Availability Modeling and Analysis in a Virtualized Servers Network

Tuan Anh Nguyen, Jong Sou Park
Korea Aerospace University, Seoul, Korea
anhnt@kau.ac.kr, jspark@kau.ac.kr

Abstract

Virtualization has been considered as an inventive approach for modern information systems in the revolutions of cloud computing and big data. Virtualized systems have been taken into account in research in different granularities. In the trend of research on virtualized systems, this paper presents a new approach of virtualization in a data center. Our work aims to propose a network of virtualized servers incorporated with live VM migration to guarantee the localization of fault-tolerance. In order to maximize resource utilization and system availability, we introduce three rules of live VM migration inside the network. The proposed network system is modeled by Stochastic Reward Net (SRN) and analyzed by Stochastic Petri Net Package (SPNP). As an important measure of interest, system availability is our scope of analysis. The analysis results show new findings that are only revealed in a network view. Networking incorporated with live VM migration enhances remarkably system availability. Based on the analyses, we challenge the system developers to pay more attention on the core points of system.

1. Introduction

Industry sector is accelerating the adoption of virtualization technology to reduce the burden of IT infrastructure costs in data centers and information systems. A strict requirement in design and administration of a data center from commercial side is always considered as a prerequisite conforming the service level agreements (SLAs) [1]. As a term in each SLA, availability is usually an important measure of interest to maintain proper services. In order to deploy and maintain 24/7 applications and services in data centers, virtualized computing platforms have been emerging as one of appropriate solutions. As the core of the virtualized computing systems, virtual machine monitor (VMM) (also known as hypervisor) with the features of transparency, isolation, encapsulation and manageability makes computing systems with better scalability, migration and server consolidation [2]. Thanks to virtualization solution on hardware resources in physical systems, different and multiple computing environments called virtual machines (VMs) are created and fostered to maintain the flexibility and competency of long-life and large-scale applications. On each VM, system designers can launch required operating systems (OSs) which in turn allow multiple applications (Apps) running. Virtualization has been adopted into different levels of computing system including storage virtualization, network virtualization, client virtualization and server virtualization. Kim *et al.* [3] initiated the research trend on virtualized system. Machida *et al.* in work [5] proposed the configuration of a data center adopting virtualization called virtualized data center (VDC). Wei *et al.* [7] built a stochastic petri net (SPN) model for a typical architecture of a VDC regarding task circulation and workload. In this paper, we propose a network of virtualized servers. We call this network as a virtualized servers network (VSN). We introduce a virtualized servers system (VSS) in which two virtualized servers are combined together as a single node in the network. We construct the stochastic reward net (SRN) model of the whole network and analyze the availability and related measures of interest using Stochastic Petri Net Package (SPNP) [8]. We come up with new findings that challenge the system developers to incorporate more effective techniques to migration and concern more on the VM underlying layers including hardware layer and VMM layer to gain more system availability.

The rest of this paper is organized as follows. In Section 2, related work is introduced. Section 3 presents system architectures including VSS and VSN. The construction of analytical SRN model of VSN is described in Section 4. The numerical results of different output measures of interest are shown in Section 5. Section 6 concludes the paper with contribution and future work.

2. Related Work

Machida *et al.* [5], [6] proposed the oriental ideas of virtualization on data centers. At first, the work [5] initiated the research trend by discussing the issues of perform-ability management in a data center incorporated server virtualization and software rejuvenation, which is called virtualized data center (VDC). Furthermore, the work [6] proposed a combined server rejuvenation technique in which both VMM and VM rejuvenations are simultaneously performed. A well-organized live VM migration regarding the above rejuvenation scheduling technique is introduced. In a more detail view, Kim *et al.* [3] studied a virtualized system by using continuous time Markov chain (CTMC) models to analyze the availability of a particular virtualized system regarding hardware layer and virtualized layer. Machida *et al.* [9] focused on analysis of virtualized layers with the incorporation of rejuvenation for both VMMs and VMs. In work [4], the authors extended the above study by incorporating VM live migration. Rezaei *et al.* [10] combined time-based rejuvenation policy for VMM layer and prediction-based policy for VMs layer. Han *et al.* [2] considered another aspect of the research trend in which a workload-based rejuvenation is integrated. Wei *et al.* [7] built a stochastic petri net (SPN) model for a typical architecture of a VDC regarding task circulation and workload. The common lack of contemporary researches is either focusing on a single virtualized server in very detail or trying to build a general analysis model for the whole data center without considering thoroughly the interaction between nodes in the network of servers in a data center. Most of researches try to improve system availability by incorporating different rejuvenation strategies. Live VM migration is implemented in an ambiguous connection of two servers without explicit description. Most work focused on single or two hosts virtualized system. It is necessary to model and analyze the availability of virtualized system consisting multiple hosts. In this work, we scale up the scope of a single virtualized system and scale down the scope of a virtualized data center to study a typical configuration of servers in a network view. We

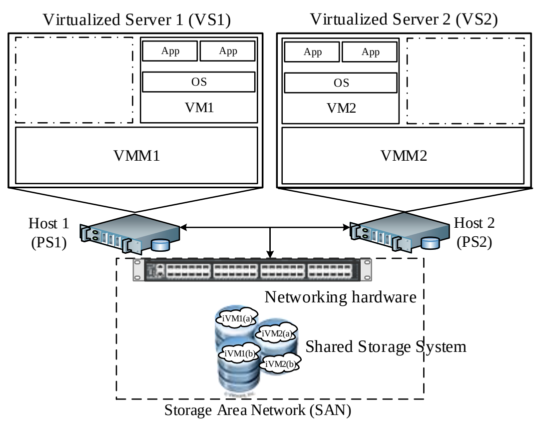


Figure 1. Virtualized Servers System Architecture

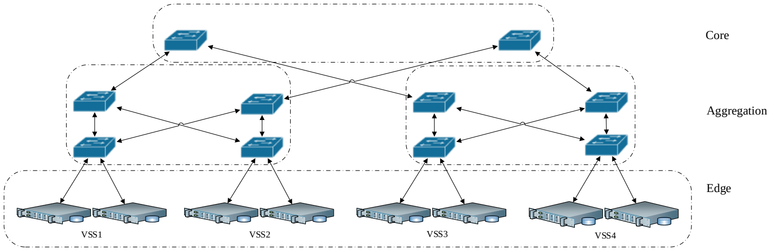


Figure 2. 3-level Fat Tree Topology of a Virtualized Servers Network

focus on the migration activity between nodes in the network. Moreover, taking the advantage of the flexibility of a set of guard function in Stochastic Reward Net (SRN) facilitates us to capture dynamic behaviors of the system in very detail.

3. System Architecture

We here present a single virtualized servers system, then we extend the system in a typical network topology.

3.1. Virtualized Servers System

The architecture of a VSS with multiple-virtual machines is depicted in Figure 1. The proposed virtualized system consists of two virtualized servers, Virtualized Server 1(VS1) and Virtualized Server 2 (VS2). Each of VSs has identical internal structure composed of a physical server computer called Host (H or PS), a virtual machine monitor (VMM), one upper virtual machines (VMs), an operating system (OS) on VM, and multiple applications/services (Apps). A Host holds the role of a physical underlying hardware platform which allows other upper software subsystems to run on. Likewise, a VMM, as a host program running on a Host, enables the physical computer to support multiple, identical execution environments to guest OSs. And similar to a physical computer, a VM consists of an OS which in turn serves as a host program to multiple Apps running on top of it. All the Apps are in charge of processing incoming requests. Nevertheless, in the scope of this paper, we disregard the involvement of OS and Apps in the modeling and analysis. And we neglect the workload effects [11], [12] which are not dominant in our research on the system evaluation. We therefore denote the OS and Apps elements same in both virtualized servers to indicate that workloads are not different. Furthermore, we incorporate a storage area network (SAN) interconnected to the both servers. SAN enables the storage system to appear as a local memory (iVM) associated with the VMs in virtualized servers.

3.2. Virtualized Servers Network

Based on the abovementioned VSS, we extend the isolated architecture of VSS by constructing a network of VSSs (see Figure 2) conforming 3-level Fat Tree topology [13] with n=4 (n means the number of ports on the switches). The network topology enables VSSs to connect efficiently to each other. We choose a typical configuration called Virtualized Servers Network (VSN). A VSN consists of eight PSs in four VSSs. Currently, we focus on the network behaviors of VSSs hence we neglect the role of networking devices in our modeling and analysis. And we assume that the migration malfunctions are not taken into account in our modeling. This point could be an extension in future works.

4. Stochastic Reward Net Model

The SRN model of VSN is shown in Figure 3. The model looks symmetrical so we neglect the index in notations in the following description to shorten model explanation unless it is necessary to mention clearly. For the sake of network analysis, we compact a complicated VSS model by using two-state model formulation. A SRN model of a VSS now consists of VM model and VM underlying layer (VMU) model. The dynamic behaviors of both layers are simply modeled as in a couple of up state and down state depicted as places (PVMup, PVMdown) and (PVMUup, PVMUdn). Initially, there are a specific number of VMs (*nVM*) running on the virtualized environment in each VSS. A VM is in upstate (PVMup) if it is running in normal state or failure-probable state. And a VM is in down state if it is in failure state, rejuvenated state or down-dependent state (i.e. VM is down because of the underlying layers). As soon as a VM falls into down state because of diverse causes including VM failure, VM/VMM rejuvenation, SAN failure; the transition TVMdn is enabled to remove a token in PVMup and deposit to PVMdn. The VM in down state can return to up state through the transition TVMrecov by recovery measures. On the other side, we squeeze the underlying layer of VMs in a VSS by assigning one token in the VMU model. Similar to VM model, the VMU is considered at first in normal state (PVMUup). The operation state of VMU is switched to down state if any of down causes occurs such as all VMMs are in failure/rejuvenation/down states or all PSs are in failure. The rate of the transitions TVMUdn, TVMUrecov, TVMdn and TVMrecov can be assigned by the values of mean time to failure equivalent (MTTFeq) and mean time to recover equivalent (MTTReq) computed in the work [14]. However, in this paper we consider these parameters as adjusted variables to observe the dependence of steady state availability (ssavail) of the whole system. We denote the ordering number (1-4) in the place and transition names to indicate those of the respective VSS. Now we construct the connection between VSSs to support VM live migration. The migration of VMs within VSN is incorporated regarding the rule of least-load optimization (LLO) and the rule of maximum capacity (MAXC). The migration of VMs between VSNs in a VDC is conformed the rule of maximum system capacity (MAXSC). And also, we don't allow to migrate a failed/down VM. Therefore, a VM in down state PVMdn is not migrated and not recovered as long as the underlying layer is in down state PVMUdn.

Rule 1(LLO): *A VM is migrated to a VSS if the number of VMs that currently exist in the VSS is the least among the VSSs in the VSN.*

Rule 2 (MAXC): *A VM is migrated to a VSS if the number of VMs that currently exist in the VSS is still less than the capacity (C) of the VSS to host.*

Rule 3 (MAXSC): *A VM is migrated inward to a VSN if the number of VMs that currently exist in the VSN is still less than the total capacity (TC) of the VSN to host. A VM is migrated outward of a VSN if the number of VMs that currently exist in the VSN is larger than the total capacity (TC) of the VSN to host.*

The LLO rule is applied to guarantee the load balancing strategy in term of the amount of VMs residing on each VSS. We here neglect the workload assigned to each VM as mentioned in VSS architecture. The MAXC and MAXSC rules are applied to guarantee the safety of computing operation in which there is no overload VSS or VSN. This is a critical requirement to not violate SLA and other commercial agreements.

To incorporate these three rules of VM migration in the SRN model of the VSN, we use a place (PVMmig) as the temporary central for migration (see Figure 3). The current running VMs in a VSS are marked to be migrated from upstate PVMup if the underlying layer is in downstate PVMUdn. All the tokens in PVMup are removed and deposited immediately in PVMmig by triggering the according immediate transition tVMmig. Here the rules 1 and 2 are applied to decide where to migrate a VM. If VSS*j* satisfies the rules, a VM in PVMmig is migrated to

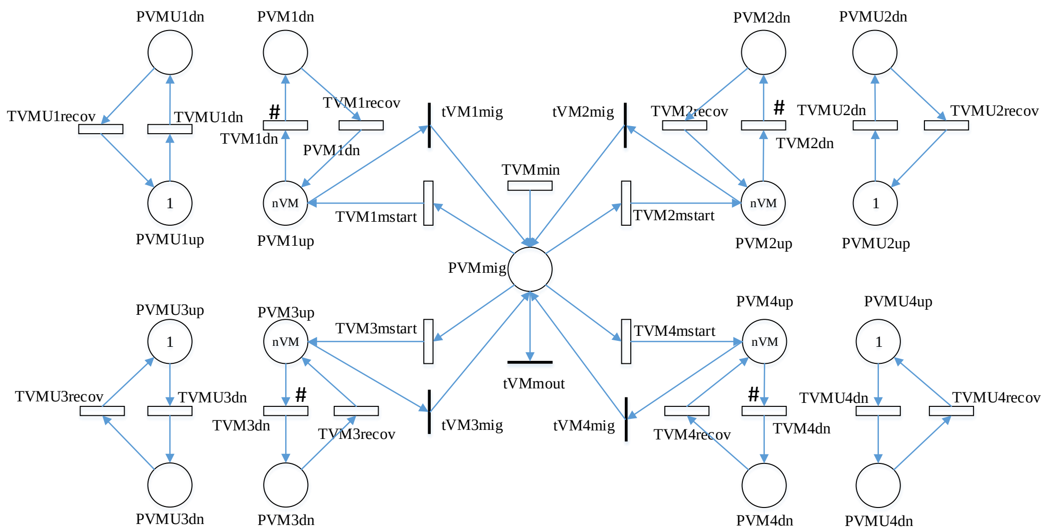


Figure 3. Stochastic Reward Net Model of a VSN

PVM*j*up through the transition TVM*j*mstart. On the other side, the migration between VSNs in a data center is incorporated by the transition TVMmin and tVMout. TVMin is enabled to migrate a VM from another VSN to the current VSN of interest if the rule MAXSC is satisfied (i.e. the current VSN is capable to host not only its own VM but also more VM from other VSNs in data center). In an opposite way, a VM is migrated outward of the current VSN of interest as tVMmout is triggered if the rule MAXSC is satisfied, i.e. the current VSN is not capable to host all the migration-waiting VMs. These migration operations are stopped as long as the rule MAXSC is violated.

A typical dependence between VMs in a VSS called place dependence is also incorporated in this model. The failure rate of TVMdn is varied upon the number of VMs running in PVMup in the VSS. We denote this dependency by putting a "#" sign next to the respective TVMdn.

All the dynamic behaviors described in Section 4 are controlled by guard functions. Table 1 shows the definition of guard functions attached to respective transitions. The guard functions gTVMmstart, gTVMmin, gtVMout are designed to protect the three rules of VM migration. All the timed transitions in the SRN model are assumed to conform exponential distribution.

Table 1. Guard function definition

Guard	Transition	Definition
gtVMmig	tVM1mig, tVM2mig, tVM3mig, tVM4mig	if(#PVMU1up==0) 1 else 0;
gTVMenable	TVM1dn, TVM2dn, TVM3dn, TVM4dn, TVM1recov, TVM2recov, TVM3recov, TVM4recov	if(#PVMU1up==1) 1else 0;
gTVMmstart	TVM1mstart, TVM2mstart, TVM3mstart, TVM4mstart	if(#PVM1up==1&&fVM1()<=fVM2()&&fVM1()<=fVM3()&&fVM1()<=fVM4()) 1 else 0; if(#PVM2up==1&&fVM2()<=fVM1()&&fVM2()<=fVM3()&&fVM2()<=fVM4()) 1 else 0; if(#PVM3up==1&&fVM3()<=fVM1()&&fVM3()<=fVM2()&&fVM3()<=fVM4()) 1 else 0; if(#PVM4up==1&&fVM4()<=fVM1()&&fVM4()<=fVM2()&&fVM4()<=fVM3()) 1 else 0;
gTVMmin	TVMmin	if(#PVMU1up==1) {mVM=mVM+fVM1();cVM=cVM+C;} if(#PVMU2up==1) {mVM=mVM+fVM2();cVM=cVM+C;} if(#PVMU3up==1) {mVM=mVM+fVM3();cVM=cVM+C;} if(#PVMU4up==1) {mVM=mVM+fVM4();cVM=cVM+C;} mVM=mVM+#PVMmig; if(mVM<cVM) 1 else 0;
gtVMmout	tVMmout	if(gTVMmin()) 1 else 0;
fVM1		return(#PVM1up+#PVM1dn);
fVM2		return(#PVM2up+#PVM2dn);
fVM3		return(#PVM3up+#PVM3dn);
fVM4		return(#PVM4up+#PVM4dn);

5. Numerical Results

We evaluate the proposed network based on the numerical analysis of the SRN model described in section 4. The default value of parameters are assigned as in Table 2. The values are extracted from literature review [14][3] and assumptions. We assume that there is one VM running and one vacancy for migration-requested VM on each PS. Therefore, each VSS has totally two running VMs and volume capacity is able to host maximum four VMs at the same time.

Table 2. Default parameter values

Parameters Name	Transitions	Description	Mean time
λ_{vmu}	TVMU1dn TVMU2dn TVMU3dn TVMU4dn	VM underlying failure rate	2654 hours
μ_{vmu}	TVMU1recov TVMU2recov TVMU3recov TVMU4recov	VM underlying recovery rate	75 hours
λ_{vm}	TVM1dn TVM2dn TVM3dn TVM4dn	VM failure rate	218 hours
μ_{vm}	TVM1recov TVM2recov TVM3recov TVM4recov	VM recovery rate	65 mins
β_{vm}	TVM1mstart TVM2mstart TVM3mstart TVM4mstart	VM migration rate	30 seconds
β_{min}	TVMmin	VM deposit	7 days
n_{VM}	x	VM amount	2
C	x	VM capacity	4

We first analyze the ssavail and downtime measures of the system under given default parameter values. To reflect the advancement of the current system, we compare it with the non-networked single VSS. The results shown in Table 3 pinpoints that networking incorporated with live VM migration enhances remarkably the availability and reduces significantly the downtime of system compared to those of the case without networking and migration.

We extend our analysis by computing the output measures to observe the migration operations inside the VSN. Table 4 shows the number of migration occurring in a year and the utilization within a year for migration. This could be a good reference as we enlarge the system architecture.

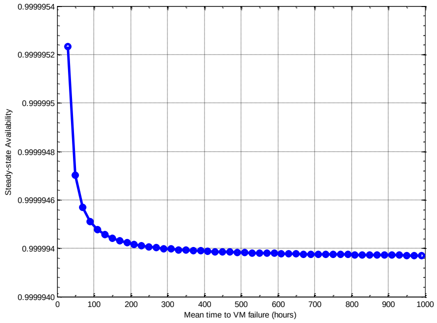
Table 3. Availability analyses of VSN under given default parameter values

	With networking and migration	Without networking and migration
Steady-state availability	0.999994408913	0.972069595425
Downtime in minutes per year	2.9386753272	14680.22064462

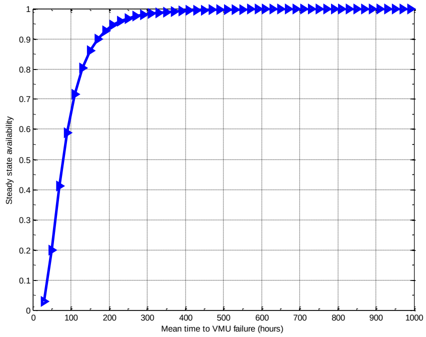
Table 4. Output analyses of VMs migration

Output measures	Value
Migration transaction per year	45.3629
Migration utilization in minutes per year	5.67

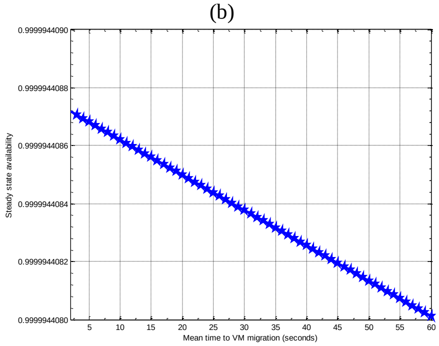
In order to determine the factors that are most influential on the availability of system, we conduct sensitivity analysis of ssavail with respect to different variables. We observe the influence of mean time to VM failure (MTTF of VM), mean time to VMU failure (MTTF of VMU) and mean time to VM migration (MTTM) on the ssavail. The procedure is as follows: first, we fix all default parameter values as in Table 2; after that we vary sequentially the value of observed variables and eventually we compute the output measure of ssavail. The results are shown in Figure 4. The variations of MTTF of VM and MTTF of VMU influence the ssavail of system in opposite trends. The ssavail of system drops down as the MTTF of VM increases whereas the ssavail of system leaps up as the MTTF of VMU increases. This finding challenges system developers to enhance the stability and long-held quality of VM underlying layers including hardware system and VMM software system. The MTTM is observed to impact the ssavail of system in monotonic tendency. The ssavail decreases linearly according to the increase of MTTM. This finding confirms the sense that the longer the VM migration operation takes, the less of the ssavail the system earns. This demands system developers to incorporate more efficient migration techniques. [15]–[19].



(a)



(b)



(c)

Figure 4. Steady-state availabilities of VSN with respect to variables
(a) Mean time to VM failure; (b) Mean time to VMM failure; (c) Mean time to VM migration

6. Conclusion

This paper contributes a network approach of virtualized servers incorporated with VM live migration. This could be considered as a preliminary research on the networking of servers in a data center toward the construction of the whole virtualized architecture of a data center. Furthermore, this research opens a broad venue of research on virtualization technology in a data center. A recursive modeling based on this research could be a proper way for future works to deal with a virtualized data center with a large number of servers.

Appendix. Stochastic Reward Net

Stochastic Reward Net (SRN) is used in sufficiently modeling many hardware and software structures of real-time computing systems [20]. To build SRN model we use three main components: places, transitions and arcs. Arcs only connect place(s) to transition(s), and transition(s) to places. There is an integer number of entities named token denoted by dot sign or integer number in the places. Transition can be enabled to transport tokens from and to places called firing. The state or condition of the system is decided by location of tokens [21], [22]. That means, a set of current location of tokens in SRN models reflects the state or condition of the system, called marking. Guard is a Boolean condition attached to each transition to perform marking-dependence. To succinctly describe many complex behaviors, marking-dependent firing rates of transitions are applied as a function of the current marking. This dependency is denoted by “#” sign next to the transition. More general dependencies are often needed and hence allowed in the SRN formalism [23]. There are other features such as input arcs; inhibit arcs, multiplicities, so that SRN models can be simplified. Stochastic Petri Net Package (SPNP) is a versatile modeling tool for performance, dependability and perform-ability analysis of complex systems [8]. It was developed by Duke University. SRN models are solved by efficient and numerically stable algorithms. Input language is CSPL (C based SPN language). CSPL is a C file so it is compiled by using a C compiler and linked with precompiled files which constitute SPNP. Moreover, it contains GUI environment using Java [24].

References

[1] P. Patel, A. Ranabahu, and A. Sheth, Service Level Agreement in Cloud Computing. 2009.

[2] L. Han and J. Xu, "Availability Models for Virtualized Systems with Rejuvenation," J. Comput. Inf. Syst., vol. 20, pp. 8389–8396, 2013.

[3] D. S. Kim, K. S. Trivedi, and F. Machida, "Availability Modeling and Analysis of a Virtualized System," in 2009 15th IEEE Pacific Rim International Symposium on Dependable Computing, 2009, vol. 1, pp. 365–371.

[4] F. Machida, D. S. Kim, and K. S. Trivedi, "Modeling and analysis of software rejuvenation in a server virtualized system with live VM migration," Perform. Eval., vol. 70, no. 3, pp. 212–230, Mar. 2013.

[5] F. Machida, D. S. Kim, J. S. Park, and K. S. Trivedi, "Toward optimal virtual machine placement and rejuvenation scheduling in a virtualized data center," in 2008 IEEE International Conference on Software Reliability Engineering Workshops (ISSRE Wksp), 2008, pp. 1–3.

[6] F. Machida, J. Xiang, K. Tadano, and Y. Maeno, "Combined Server Rejuvenation in a Virtualized Data Center," 2012 9th Int. Conf. Ubiquitous Intell. Comput. 9th Int. Conf. Auton. Trust. Comput., pp. 486–493, Sep. 2012.

[7] B. Wei, C. Lin, and X. Kong, "Dependability Modeling and Analysis for the Virtual Data Center of Cloud Computing," 2011 IEEE Int. Conf. High Perform. Comput. Commun., pp. 784–789, Sep. 2011.

[8] G. Ciardo, J. Muppala, K. Trivedi, and K. T. Gianfranco Ciardo, "SPNP: stochastic Petri net package," in Proceedings of the Third International Workshop on Petri Nets and Performance Models, PNPM89, 1989, pp. 142–151.

[9] F. Machida, D. S. Kim, and K. S. Trivedi, "Modeling and analysis of software rejuvenation in a server virtualized system," in 2010 IEEE Second International Workshop on Software Aging and Rejuvenation, 2010, pp. 1–6.

[10] A. Rezaei and M. Sharifi, "Rejuvenating High Available Virtualized Systems," 2010 Int. Conf. Availability, Reliab. Secur., pp. 289–294, Feb. 2010.

[11] K. Ye, D. Huang, X. Jiang, H. Chen, and S. Wu, "Virtual Machine Based Energy-Efficient Data Center Architecture for Cloud Computing: A Performance Perspective," in 2010 IEEE/ACM Int'l Conference on Green Computing and Communications & Int'l Conference on Cyber, Physical and Social Computing, 2010, pp. 171–178.

[12] D. Bruneo, S. Distefano, F. Longo, A. Puliafito, and M. Scarpa, "Workload-Based Software Rejuvenation in Cloud Systems," IEEE Trans. Comput., vol. 62, no. 6, pp. 1072–1085, Jun. 2013.

[13] M. Al-Fares, A. Loukissas, and A. Vahdat, "A scalable, commodity data center network architecture," ACM SIGCOMM Comput. Commun. Rev., vol. 38, no. 4, p. 63, Oct. 2008.

[14] T. A. Nguyen, D. S. Kim, and J. S. Park, "Availability Modeling and Analysis of a Virtualized Servers System." Seoul, Korea, 2014.

[15] Yingwei Luo, Binbin Zhang, Xiaolin Wang, Zhenlin Wang, Yifeng Sun, and Haogang Chen, "Live and incremental whole-system migration of virtual machines using block-bitmap." pp. 99–106, 2008.

[16] E. Harney, S. Goasguen, J. Martin, M. Murphy, and M. Westall, "The Efficacy of Live Virtual Machine Migrations Over the Internet." pp. 1–7, 2007.

[17] U. Deshpande, B. Schlinker, E. Adler, and K. Gopalan, "Gang Migration of Virtual Machines Using Cluster-wide Deduplication," in 2013 13th IEEE/ACM International Symposium on Cluster, Cloud, and Grid Computing, 2013, pp. 394–401.

[18] S. Kumar Bose, S. Brock, R. Skeoch, N. Shaikh, and S. Rao, "Optimizing live migration of virtual machines across wide area networks using integrated replication and scheduling," in 2011 IEEE International Systems Conference, 2011, pp. 97–102.

[19] Zhiyong Yang, Chunlin Li, A.-G. Yun, and Chang Liu, "A New Trigger Strategy Based on Live Migration of the Virtual Machine." pp. 677–680, 2012.

[20] C. Constanti, D. Jtiiversity, C. Constazltinescu, and K. Trivedi, "A stochastic reward net model for dependability analysis of real-time computing systems," Proc. 2nd IEEE Work. Real-Time Appl., pp. 142–146, 1994.

[21] B. Randell and J. Xu, Software Fault Tolerance. John Wiley, 1995.

[22] R. Sahner, K. S. Trivedi, and A. Puliafito, "Performance And Reliability Analysis Of Computer Systems (an Example-based Approach Using The Sharpe Software," IEEE Trans. Reliab., vol. 46, no. 3, pp. 441–441, Sep. 1997.

[23] K. S. Trivedi, Probability and Statistics with Reliability, Queueing, and Computer Science Applications, 2nd Edition. Wiley-Interscience, 2001, p. 830.

[24] K. S. Trivedi, "SPNP User ' s Manual," Options, no. 919, 1999.