International Journal on
"Technical and Physical Problems of Engineering"
(IJTPE)
Published by International Organization of IOTPE

ISSN 2077-3528
IJTPE Journal
www.iotpe.com
ijtpe@iotpe.com

March 2024 | Issue 58 | Volume 16 | Number 1 | Pages 277-283

A DEEP REINFORCEMENT PREDICTION MODEL FOR LIVE VM MIGRATION IN FOG

M. Patel [1] A. Jha [2] A. Mehta [3] A. Patel [1] A. Nayak [4]

1. Department of Information Technology, Devang Patel Institute of Advance Technology and Research, CHARUSAT, Gujarat, India, minalpatel.dce@charusat.ac.in, akashpatel.dit@charusat.ac.in
2. Computer Engineering Department, Madhuben and Bhanubhai Patel Institute of Technology, CVM University, Gujarat, India, akjha@mbit.edu.in
3. Department of Information Technology, Shankersinh Vaghela Bapu Institute of Technology, Gandhinagar, India akash.mehta.it@gmail.com
4. Devang Patel Institute of Advance Technology and Research, CHARUSAT, Gujarat, India amitnayak.it@charusat.ac.in

Abstract- Virtualization is the key enabler to manage various aspects of cloud and fog computing. Cloud needs to send all the data that is generated by billions of Internets of Things (IoTs) devices to distant data centers which makes the data handling very challenging. Fog computing is outspreading cloud computing by transporting data and computation on the edge of networks with reduced latency and bandwidth. It is quite difficult to fulfil on demand data processing requests of multiple IoTs for real time applications. To address this requirement, there is need to provide small cloud infrastructure viz. fog computing nearer to the IoTs users. The performance of fog infrastructure is dependent on how efficiently the workloads are managed and executed. Virtual Machine (VM) migration is the key technology to manage fog environment efficiently. With this paper, the improved pre-copy based VM migration is proposed using Deep Reinforcement Learning (DRL) to improve the prediction accuracy and hence the performance of fog environment. The simulation results outperformed existing algorithms for different workloads for VM migration.

Keywords: Fog Computing, Cloud Computing, Total Migration Time, Downtime, Deep Reinforcement Learning (DRL), Support Vector Regression (SVR).

1. INTRODUCTION

Today's era of computing can serve the computation efficiently on demand with the help of cloud computing. The basics of cloud computing provide computing for networks, servers and storage at extensive scale [1-3]. The core technology of the cloud computing is the virtualization, which enables multiple instances called virtual machines (VMs) with different operating systems to run simultaneously. The hypervisor/Virtual Machine Monitor (VMM) is a middleware that creates, executes and manages VMs. There are two types of hypervisors: Type1 and Type2. VMs can be transferred between physical hosts using VM migration where service scan be available without delay. The migration of a VMs is useful for fault tolerance, energy efficiency, load balancing, etc. of datacenter.

Infrastructure as a service in cloud is a higher level abstraction of existing hardware. Users can use that infrastructure in pay-per-use model in the form of virtual machines at any point. No one is aware about cloud infrastructure for privileged activities and user can perform various cloud servicers for deployment and storage. The fog computing is developed between cloud and edge architectures and it works for end-to-end IoTs applications by providing services like storage, computation and networking [4]. The fog is considered to be the cloud extension for services nearby to the customer. The fog is implemented with heterogeneous nodes on the Internet, and it provides layered architecture to merge people, things and services for any application to perform.

VMs runs on physical host with CPU, storage, virtual memory, networks using hypervisor and the VM migration can transfer resources from one host to another. The Xen [5] is type-1 hypervisor which can impart services for hardware to assist multiple operating systems to run concurrently. The architecture of fog has three major components as follows:

- Cloud based data framework
- Fog data central system
- Sensors management system

The data collection system includes data storage, data acquisition units, and data access. At the initial phase, data is gathered from various sources and stored at cloud data center. As per Figure 1, The data center is built by various network components, software, and data storage. The computational power can be scaled up and down as and when required for cloud. The fog data central system contains all available fog nodes managing services for

Registered: 230814 | Article Info: 1738 | Accepted: 240229
Reserved: 230927 | Available: 240331

various IoTs devices. The combination of both cloud and fog architecture is the primary aspect to deal with smarter access to applications through VM migration. Traditionally, Fog nodes gets connected with access points of wireless networks to provide fast computation with lower latencies and throughputs compare to cloud data center. Infog environment, services are not required most of the time to write in disk because of the nature of fog services which are non-persistent and time critical activities. It is observed that fog computing can provide services for runtime environment and that is migrated to destination for execution of services.

The fog computing [4] is designed based on the content delivery network and it has tremendous support for various applications. The existing techniques [7, 8] are unable to provide an optimal accuracy for the prediction of dirty pages with pre-copy approach. Our proposed work is implemented based on DRL prediction model and it is compared with existing SVR prediction model using time series analysis. In the remaining sections, background, related work and proposed fog migration prediction model are discussed. Atthe end of this paper, the simulation results followed by comparison and concluding remarks are presented.

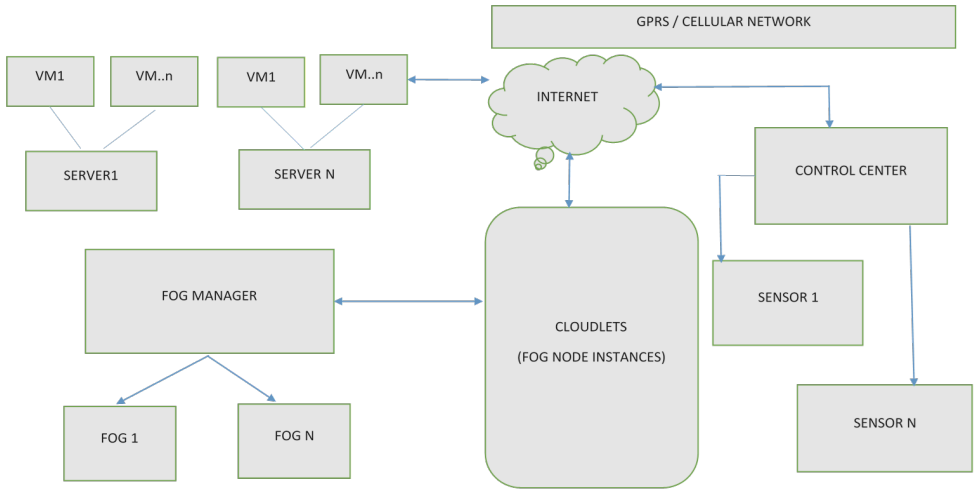

Figure 1. Architecture for Fog Computing using Cloud Datacenter

2. RELATED WORK

The main use of pre-copy migration [10], [11] is to reduce the downtime. Pre-copy is able to minimize number of iterations through managing Working Window Size (WWS) [12]. The downtime can be increased when iterations are going to complete at early stage. During post-copy-based mechanism, memory is transferred after the process state and pages are able to manage whenever required [10], [12], [13]. Authors in [14] represent a novel three-stage memory based VM migration algorithm. Existing pre-copy based VM migration approaches [10], [11], [14] are implemented to copy iteratively redundant memory pages. In post-copy-based algorithm, the problem of page fault is occurred and hencethe performance can be degraded. In [15], two methods viz. freeing page cache pages and stunning rogue processes are used. The shadow page table mechanism of Xen is able tofind accurate WWS to track dirty pages. In [16], a combined splay tree algorithm and Least Recently Used (LRU) cache and is implemented and tested with different workloads. This approach is abel to reduce both total data transferred and total migration time.

The optimized mechanism for migration [17] is designed with probability prediction algorithm. The dirty pages prediction is measured using accurate working set to reduce the retrains-mission of dirty pages. The Markov model is used to measure number of dirty pages and the probability of pages get dirtied again. The rate of dirty pages is the primary part to improve the basic pre-copy algorithm. In pre-copy approach with time series [18], pages which are modified in past and prediction can be performed accurately for dirty pages. This approach outperformed basic Xen's pre-copy in terms of migration time, down time, number of iterations, and pages transferred. The final stop-and-copy phase is started when threshold value for number of iterations is reached or threshold value for total dirty pages is also reached. In paper [18], pre-copy wity K/N time-series is discussed to ignore unwanted transfer of high dirty pages. The memory and CPU state data are used to transfer in this algorithm. The major limitation of this algorithm is that it is difficult to identify dirty pages based on page access mechanism.

For VM migration, a smart migration of pre-copy is presented to overcome the problem of system failure due to malicious attack on VMs or system failure. Using [19], downtime is estimated by each iteration competes and that can provide the information about to get starting stop-and-copy stage for failure or an attack with fog computing system. The paper [20] discusses resource allocation in fog computing for users' mobility to consider new simulation tool named MyiFogSim which is an extension of iFogSim. The user can be accessed withone hop distance in fog system and migration of VMs can be decided using user's location. The data

accessed by the user is also considered for migration of VMs. The policy for VMs migration is to be implemented with reduced latency and service downtime amongst fog nodes.

The container is lightweight and suitable for IoTs enabled applications. When cloudlets are having more data to transfer at that time, the VM is the key factor to execute and used for user devices to execute the required data for fog computing servers. The cost of data offloading in fog computing is sometimes costlier than the communication/migration cost. In the era of machine learning, the complexity of real data can be handled by various techniques. In this paper, we focused on live migration logs for our comparison in fog computing using two different methods i.e. SVR and DRL. We compared our results based on these models and proved that the deep reinforcement learning based method can give best accuracy for dirty pages prediction.

3. TECHNIQUES OF LIVE VM MIGRATION IN FOG ENVIRONMENT

In previous section, we have discussed the types of virtualizations and the working of on-demand self-service in cloud computing. The Figure 2 shows the live migration in fog computing with user movement. If user moves from one location to another, the services are available smoothly using VM migration in fog computing. Another scenario is discussed in Figure 3, in which the device is able to update the services when VM migration occurs and the second fog node is able to acknowledge the services during migration process. Live migration [21] enables the smooth transition without experiencing delay from host to destination for any running VM. In seamless migration process, the downtime is not noticeable. The recent state is available in the pre-copy while at the time of post-copy algorithm, the recent state is distributed between source and destination. The main difference between these two techniques is that the VM can be removed by pre-copy but the post-copy is not able to recover the VM.

The running state of the VM, virtual or storage disks, and existing client connections are the major activities of Xen hypervisor. The blocks which are updated during migration are recopied again. The stop-and-copy condition of Xen hyervisor is begun when the hypervisor is able to start the final iteration to copy remaining memory pages [15].

The VM Migration between two hosts performs following steps [15]:

- Step 1: pre-migration stage - Used for guaranteed migration of resources where destination host is to be chosen.
- Step 2: reservation stage - migration is being started with registered resources.
- Step 3: iterative pre-copy phase - until last iteration, memory pages are copied iteratively.
- Step 4: stop and copy phase - fix dirty pages are copied after VM is stopped and the destination system will become the primary machine.
- Step 5: activation phase - the post-migration activities are initiated by the destination host.

The process time of each iteration [15] is represented using following Equation 1:

$$Migration\ time\ of\ iteration\ (i-1) = \\ = dirtied\ pages\ in\ iteration\ (i-1)\ for\ migration\ / \\ /\ page\ dirty\ rate \quad (1)$$

Total time for migration is represented using following Equation 2:

$$Migration\ for\ total\ time = \\ = Time\ covered\ by\ all\ \text{int}eration + downtime \quad (2)$$

The pre-copy algorithm is started with rate of transfer and memory size considering as input. It terminates when stop-and-copy situation is raised. The pre-copy time is calculated by each iteration and hence the total migration time can be measured. The downtime is taken as the time of response of last iteration. In contrast to pre-copy, the post-copy [10] transfers every page once during the VM migration. It transfers the memory pages after the state of process is transferred. The post copy migration is not widely used due to its demerits compare to precopy algorithm.

A) VM Migration in Fog Computing:

In fog computing, it is possible to migrate users' data seamlessly between small scale clouds called cloudlets. Due to that, the services/data can be migrated to nearer fog server with resulting data analysis using resource allocation optimization problem [20]. The migration in fog is divided into two main parts viz. where to migrate VM and how to migrate.

The lowest latency and bandwidth requirement is considered to be the foremost purpose to get the decision for VM migration. For the same, the cloudlet at the shortest distance may be selected. Sometimes, the shortest distance between device and router is also considered for VM migration. The VM migration implementation is mainly designed with pre- copy VM migration and container migration. The techniques associated with container migration has their limitations like handling complex data compared to live VM pre-copy migration. In this work, the VM migration using pre-copy is extended with machine learning and deep learning algorithms for improvement in predicting the dirty pages accurately.

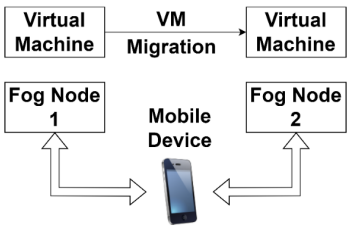

Figure 2. Live Migration in Fog Computing (User Movement)

B) Improved Pre-copy Algorithm for Machine Learning and Deep Learning:

Here, regression model with machine learning approach and deep reinforcement technique have been applied for migration.

The migration using pre-copy has been effectively used for compression and CPU scheduling in cloud computing. Fog computing paradigm requires very sensitive and small-scale information at the time of migration. The use of machine learning models along with pre-copy algorithm improves VM migration in fog environment and hence it gives optimal performance with complex modeling of workload. In proposed work, the pre-copy algorithm is chosen to improve migration of dirty pages using support vector regression and deep reinforcement learning.

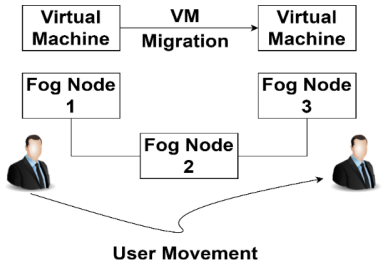

Figure. 3. Live Migration in Fog Computing (Device Management)

The models with learning strategy are used with Neural Networks (NNs) and Linear Discriminant Analysis (LDA) [24]. These techniques have their limitations to provide optimal computation cost during live VM migration. In the era of computing, the DRL mechanism is the solution to analyze a large dataset in short amount of time. Another method called Support Vector Machine (SVM) has the potential to work on linear regression problem. To achieve better prediction for dirty pages, one of the SVM library, LIBSVM, is used here. It can give better performance compared to LDA and NN [25].

The SVM technique [9] is applied to measure the global optimal output by collecting independent results irrespective of input patterns and size. In SVM, the distance called the margin is maximized between two boundary lines to identify the classes. This process is used to generate support vectors for maximum margin. The DRL is considered for a system where the size of computation task, system bandwidth and edge computation resources are dynamic for the selection of VM to be migrated in IoT networks. For that, DRL needs to allocate resources to each user to get system state with the aim of optimizing the total migration cost with three information viz. state, action and reward mechanism. In DRL [26], the agent is able to get optimal decision to find dirty page prediction. In this paper, a live VM migration using DRL model is proposed and compared with SVR model to optimize the prediction accuracy of dirty pages.

4. PROPOSED FOG MIGRATION PREDICTION MODEL

In this section, SVR and DRL models for prediction of VM migration are discussed. The strictly stationary time series is one which does not change properties with time. For any two observations, if the joint probability distribution is same then the time- series is considered to be stationary.

The non-stationary time series is used with trend and the same can be forecasted with accurate modeling. The non-stationary time series can work with regression model. The basic model of time series is as follows given in Equation 3 [24]:

$$y_t = \varphi_0 + \varphi_1 x_{t1} + \varphi_2 x_{t2} + ... + \in t \quad (3)$$

where, t= 1, 2, ..., N

The deterministic function of time is called trend or signal and it is represented by x_t, where, ϵt is called residual term and it is based on probability law. φ is the constant term, y_t is used to prepare forecasting model and N is used to represent last observation.

Algorithm 1. Prediction model using SVR

1. i) e1071 library and ii) real data set (.csv) file are imported
2. Kernel searching mechanism
3. Decide the input and output series data
4. Measure cost, gamma and epsilon values using cross-validation
5. Apply RBF kernel for smother accessing data
6. Design suitable regression model of SVR using parameters
7. Forecasting process for next data in time-series

The linear regression model involves a single predictor variable can be represented in Equation (4) as:

$$l = \beta_0 + \beta_1 x + \in \quad (4)$$

where the components are described with l is the response, x is the predictor variable, β_0 and β_1 are unknown parameters, and ϵ is an error term.

A) Support Vector Regression (SVR) Model: [27]

In this section, the SVR model is presented to predict dirty pages during live VM migration. In this paper, SVR is implemented to evaluate accuracy of real dataset [28]. The function $f(x)$ is created with two classes: +1 and -1. The value +1 means all x are at upper side of the boundary and the value-1 means all x are at below side of the boundary. The support vectors are called the points available on deciding maximum boundaries between any two classes. The SVR can provide global solution to predict data using kernel mechanism. Algorithm 1 shows the algorithm of SVR model [27].

B) Deep Reinforcement Learning (DRL) Model:

The reinforcement learning is designed with Qlearning algorithm [30], which aims to estimate Q-values to be converted to numeric values for different actions for agent. These Q- values are accurate measure of particular action for getting highest reward. The statistics of finding these Q-values are very interesting and it can be set zero first. The Q values are updated when the new information is collected while points are grabbed by the agent. The way Q-value is updated is a calculation in which observed reward is added with maximum Q-value for following state in the game. The constant term called discount is multiplied with maximum Q-value for the next state [30].

In Atari game for breakout, a mechanism for getting reward or penalty is designed based on learning algorithm [33, 34]. In developing this system, the point is how to control the paddle so that the points can be earned and dying the ball situation can be handle.

Algorithm 2. Proposed prediction model using DRL

```
1. Import MDPtoolbox, devtools and reinforcement learning library and
real data set (.csv)
2. Take dataframe data.frame(s=s,a=a,r=r,s new=s new)
3. states = c("st1", "st2") // s1 and s2 are states
4. actions = c("S", "T") // S - skip pages and T- transfer pages
5. Def environment function (state, action)
6. next state = state
7. if (state EQU state("st1") AND action EQU "S")
8. next state = state("st2")
9. if (state EQU state("st2") AND action EQU "T")
10. next state = state("st1")
11. if (state EQU state("st2") AND action EQU "S")
12. next state = state("st2")
13. if (state EQU state("st1") AND action EQU "T")
14. next state=- state("st1")
15. if (next state EQU state("st2"))
16. reward = 10
17. else
18. reward = -1
19. end
20. Run the sampleExperience with N (iterations) = 1000, env =
environment function, states = states,
actions = actions
21. Evaluate model using Reinforcement Learning with sequences, s =
"State", a = "Action",
r = "Reward", s new = "NextState"
 22. Compute policy for model
23. Print and get summary of results
```

It is possible to design this system by taking the different states so that reward is grabbed when ball hits the side of wall and bricks are smashed. To generalize this pattern, reinforcement learning is applied to develop Atari game using Q learning algorithm. Q-value for state and action is given below:

*reward + discount * max Qvalue for next state*

where, the Q-value for particular state is set to zero when the agent loses a life.

Table 1. Confusion Matrix for DRL Model

Actual Values			
Predicted Values		Positive (1)	Negative (0)
	Positive (1)	*TP* = 92	*FP* = 01
	Negative (0)	*FN* = 00	*TN* = 07

4.1. Proposed Prediction Model using DRL

The proposed model is shown in Algorithm 2 which is designed using reinforcement learning. It has divided into two states S1 and S2 and two actions performed are *S* and *T*. At the time of reading S2 sate, the reward of either 10 points or -1 point can be earned. This simulation is executed for 1000 iterations and corresponding actions and rewards are generated. At the end of tuning process, parameters of these sequences are analyzed to give final output.

Table 2. Comparison of Models [27]

Accuracy of Models		
Sr. No.	Name of Model	Accuracy (%)
1	Base Model	81
2	Simulation Model	90
3	Refined Model	90.5
4	ARIMA Model	91
5	SVR Model	95
6	Proposed DRL Model	99

C) Performance Evaluation:

Here, the output is shown with actual dirty pages and predicted dirty pages using two models - SVR and DRL. Both the models are evaluated on real dataset [28]. In SVR model, with RBF kernel, the dirty pages accuracy is evaluated using regression model. The accuracy of SVR model is approximately 95% [27].

In proposed DRL model, the dirty pages accuracy is evaluated using proposed DRL model which is approximately 99% based on the confusion matrix shown in Table 1. The confusion matrix is derived based on 100 instances of test dataset. Here, accurate data got predicted for 99 instances and got only 01 instance non-predicted. Table 2 shows the comparison of DRL model with Base model [31], Simulation model [32], Refined model [31], ARIMA model [27], and SVR model [27]. From the results, DRL based model is proved to be the accurate model to detect dirty pages during live VM migration.

5. CONCLUSIONS

In this paper, a pre-copy based live VM migration using DRL model in fog computing environment is proposed. The fog infrastructure is able to overcome cloud computing in many ways and it can useful for efficient workload management and execution. VM migration in fog is the monitoring activity to determine closer user to transfer VM state. The fog infrastructure is de-centralized and provides favorable solutions for VM migration to handle real time issues of latency and bandwidth of IoTs devices. The VM migration workload is simulated for dirty pages prediction using SVR and DRL models. The proposed approach using DRL out-performed other models with 99% prediction accuracy. The DRL is designed to get optimal decision for prediction of dirty pages during migration in advance. Our future work is to incorporate other performance features of VM migration such as total migration time, downtime, overhead, etc.

REFERENCES

[1] P. Mell, T. Grance, "National Institute of Standards and Technology", The NIST Definition of Cloud Computing, pp. 800-145, 2011.

[2] R. Buyya, C. Vecchiola, S.T. Selvi, "Mastering Cloud Computing: Foundations and Applications Programming", Newnes, 2013.

[3] R.W. Ahmad, A. Gani, S.H. Hamid, M. Shiraz, F. Xia, S.A. Madani, "Virtual Machine Migration in Cloud Data Centers: A Review, Taxonomy, and Open Research Issues", The Journal of Supercomputing, Vol. 71, pp. 2473-2515, 2015.

[4] M.P. Patel, S. Chaudhary, "Edge Computing: A Review on Computation Offloading and Light Weight Virtualization for IoT Framework", International Journal of Fog Computing (IJFC), Vol. 3, No. 1, pp. 64-74, 2020.

[5] M. Patel, S. Chaudhary, "Survey on a Combined Approach Using Prediction and Compression to Improve Pre-Copy for Efficient Live Memory Migration on Xen", International Conference on Parallel, Distributed and Grid Computing, pp. 445-450, December 2014.

[6] B.G. Tabachnick, L.S. Fidell, J.B. Ullman, "Using Multivariate Statistics", Pearson, Vol. 6, pp. 497-516, Boston, MA, USA, 2013.
[7] C. Yong, L. Yusong, G. Yi, L. Runzhi, W. Zongmin, "Optimizing Live Migration of Virtual Machines with Context-Based Prediction Algorithm", The 1st International Workshop on Cloud Computing and Information Security, Atlantis Press, pp. 441-444, November 2013.
[8] Z.H. Sun, X.L. Hu, "Live Migration for Virtual Machine Based on Kalman Prediction of Dirty Pages", Applied Mechanics and Materials, Vol. 668, pp. 1363-1367, 2014.
[9] P. Filzmoser, "Linear and Nonlinear Methods for Regression and Classification and Applications in *R*", Institute Statistics and probability Theory Working Paper, Vol. 1, No. 1, pp. 1-52, Germany, 2008.
[10] A. Shribman, B. Hudzia, "Pre-copy and Post-Copy VM Live Migration for Memory Intensive Applications", In Euro-Par 2012: Parallel Processing Workshops: BDMC, CGWS, HeteroPar, HiBB, OMHI, Paraphrase, PROPER, Resilience, UCHPC, VHPC, Rhodes Islands, Greece, August 2012, Revised Selected Papers 18 pp. 539-547, Springer, Berlin Heidelberg, Germany, 2013.
[11] C.H. Hsu, S.J. Peng, T.Y. Chan, K. Slagter, Y.C. Chung, "An Adaptive Pre-Copy Strategy for Virtual Machine Live Migration", In Internet of Vehicles-Technologies and Services: First International Conference, IOV, Beijing, China, September 2014. Proceedings 1, pp. 396-406, Springer International Publishing, 2014.
[12] M.R. Hines, K. Gopalan, "Post-copy Based Live Virtual Machine Migration Using Adaptive Pre-Paging and Dynamic Self-Ballooning", The 2009 ACM SIGPLAN/SIGOPS International Conference on Virtual Execution Environments, pp. 51-60, March 2009.
[13] M.R. Hines, U. Deshpande, K. Gopalan, "Post-Copy Live Migration of Virtual Machines", ACM SIGOPS Operating Systems Review, Vol. 43, No. 3, pp. 14-26, 2009.
[14] F. Yin, W. Liu, J. Song, "Live Virtual Machine Migration with Optimized Three-Stage Memory Copy", In Future Information Technology: FutureTech 2013, pp. 69-75, Springer Berlin Heidelberg, Germany, 2014.
[15] C. Clark, K. Fraser, S. Hand, J.G. Hansen, E. Jul, C. Limpach, I. Pratt, A. Warfield, "Live Migration of Virtual Machines", The 2nd Conference on Symposium on Networked Systems Design and Implementation, Vol. 2, pp. 273-286, May 2005.
[16] E.P. Zaw, N.L. Thein, "Improved Live VM Migration Using LRU and Splay Tree Algorithm", International Journal of Computer Science and Telecommunications, Vol. 3, No. 3, pp. 1-7, 2012.
[17] S. Mingsong, R. Wenwen, "Improvement on Dynamic Migration Technology of Virtual Machine Based on Xen", In Ifost, IEEE, Vol. 2, pp. 124-127, June 2013.
[18] B. Hu, Z. Lei, Y. Lei, D. Xu, J. Li, "A Time-Series Based Precopy Approach for Live Migration of Virtual Machines", In 2011 IEEE The 17th International Conference on Parallel and Distributed Systems, pp. 947-952, December 2011.
[19] O. Osanaiye, S. Chen, Z. Yan, R. Lu, K.K.R. Choo, M. Dlodlo, "From Cloud to Fog Computing: A Review and a Conceptual Live VM Migration Framework", IEEE Access, Vol. 5, pp. 8284-8300, 2017.
[20] M.M. Lopes, W.A. Higashino, M.A. Capretz, L.F. Bittencourt, "Myifogsim: A Simulator for Virtual MACHINE migration in Fog Computing", Companion The 10th International Conference on Utility and Cloud Computing, pp. 47-52, December 2017.
[21] R.P. Goldberg, "Survey of Virtual Machine Research", COMPUTER IEEE, pp. 34-45, 1974.
[22] S. Nathan, P. Kulkarni, U. Bellur, "Resource Availability-Based Performance Benchmarking of Virtual Machine Migrations", The 4th ACM/SPEC International Conference on Performance Engineering, pp. 387-398, April 2013.
[23] D. Chisnall, "The Definitive Guide to the Xen Hypervisor", Pearson Education, 2008.
[24] D.C. Montgomery, C.L. Jennings, M. Kulahci, "Introduction to Time Series Analysis and Forecasting", John Wiley and Sons, 2015.
[25] R. Adhikari, R.K. Agrawal, "An Introductory Study on Time Series Modeling and Forecasting", arXiv Preprint arXiv:1302.6613, 2013.
[26] X. Qiu, L. Liu, W. Chen, Z. Hong, Z. Zheng, "Online Deep Reinforcement Learning for Computation Offloading in Blockchain-Empowered Mobile Edge Computing", IEEE Transactions on Vehicular Technology, Vol. 68, No. 8, pp. 8050-8062, 2019.
[27] M. Patel, S. Chaudhary, S. Garg, "Machine Learning Based Statistical Prediction Model for Improving Performance of Live Virtual Machine Migration", Journal of Engineering, 2016.
[28] S. Nathan, U. Bellur, P. Kulkarni, "Towards a Comprehensive Performance Model of Virtual Machine Live Migration", The Sixth ACM Symposium on Cloud Computing, pp. 288-301, August 2015.
[29] R.J. Hyndman, Y. Khandakar, "Automatic Time Series for Forecasting: The Forecast Package for R (No. 6/07)", Monash University, Department of Econometrics and Business Statistics, 2007.
[30] L. Lei, Y. Tan, K. Zheng, S. Liu, K. Zhang, X. Shen, "Deep Reinforcement Learning for Autonomous Internet of Things: Model, Applications and Challenges", IEEE Communications Surveys and Tutorials, Vol. 22, No. 3, pp. 1722-1760, 2020.
[31] H. Liu, C.Z. Xu, H. Jin, J. Gong, X. Liao, "Performance and Energy Modeling for Live Migration of Virtual Machines", The 20th International Symposium on High Performance Distributed Computing, pp. 171-182, June 2011.
[32] S. Akoush, R. Sohan, A. Rice, A.W. Moore, A. Hopper, "Predicting the Performance of Virtual Machine Migration", The IEEE International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems, pp. 37-46, August 2010.

[33] A.H. Odeh, M.A. Odeh, "Increasing the Efficiency of Online Healthcare Services Software and Mobile Applications Using Artificial Intelligence Technology", International Journal on Technical and Physical Problems of Engineering (IJTPE), Issue 44, Vol. 12, No. 3, pp. 16-22, September 2020.
[34] A. Jafari, N.M. Tabatabaei, N.S. Boushehri, "Reactive Power Optimization Using Intelligent Search Algorithms Considering Voltage Stability Index", International Journal on Technical and Physical Problems of Engineering (IJTPE), Issue 28, Vol. 8, No. 3, pp. 1-8, September 2016.

BIOGRAPHIES

Name: **Minal**
Middle Name: **Parimalbhai**
Surname: **Patel**
Birthday: 29.11.1980
Birthplace: Vadodara, Gujarat, India
Bachelor: Computer Engineering, South Gujarat University (Veer Narmad University), Gujarat, India, 2002
Master: Computer Engineering, Sardar Patel University, Gujarat, India, 2007
Doctorate: Computer Engineering, Dharmsinh Desai University, Nadiad, India, 2018
The Last Scientific Position: Assist. Prof, Head of Department, Information Technology Department, Devang Patel Institute of Advance Technology and Research, Charotar University of Science and Technology, Anand, India, Since 2023
Research Interests: Fog Computing, Cloud Computing, Data Science Optimization Techniques
Scientific Publications: 6 Papers, 3 Book Chapter, 3 Theses
Scientific Memberships: ISTE

Name: **Ashwini**
Middle Name: **Birendra**
Surname: **Jha**
Birthday: 24.12.1987
Birthplace: Sitamarhi, Bihar, Gujarat, India
Bachelor: Computer Engineering, Saurashtra University, Rajkot, India, 2008
Master: Computer Science and Engineering, St. Peters University, Chennai, India, 2011
Doctorate: Computer Engineering, Gujarat Technological University, Ahmedabad, India, 2022
The Last Scientific Position: Assist. Prof., Computer Engineering Department, CVM University, Anand, India, Since 2011
Research Interests: Computer Networks, Fog Computing, Cloud Computing
Scientific Publications: 5 Papers, 1 Book Chapter
Scientific Memberships: ISTE

Name: **Akash**
Middle Name: **Kishorbhai**
Surname: **Mehta**
Birthday: 09.10.1987
Birthplace: Bhavnagar, Gujarat, India
Bachelor: Information Technology, Shantilal Shah Engineering College, Bhavnagar University, Bhavnagar, India, 2009
Master: Information Technology, Shantilal Shah Engineering College, Gujarat Technological University, Gujarat, India, 2014
The Last Scientific Position: Assist. Prof. Head of Department, Computer Engineering and Information Technology Department, SVBIT, Gandhinagar, India, Since 2022
Research Interests: Block Chain, Fog Computing, Cloud Computing
Scientific Publications: 6 Papers

Name: **Akash**
Middle Name: **Ravindrabhai**
Surname: **Patel**
Birthday: 03.10.1993
Birth Place: Bhayli, Vadodara, Gujarat, India
Bachelor: Information Technology, Gujarat Technological University, Gujarat, India. 2015
Master: Computer Engineering, Charotar University of Science and Technology, Gujarat, India, 2019
The Last Scientific Position: Assistant Professor, Devang Patel Institute of Advance Technology and Research, Charotar University of Science and Technology, Gujarat, India since 2021
Research Interests: Computer Networks, Fog Computing, Cloud Computing
Scientific Publications: 10 Papers
Scientific Memberships: ACM

Name: **Amit**
Middle Name: **Jaydevbhai**
Surname: **Nayak**
Birthday: 01.05.1984
Birthplace: Umata, Mehsana, Gujarat, India
Bachelor: Information Technology, U.V. Patel College of Engineering, Gujarat, India 2005
Master: Computer Engineering, Charotar University of Science and Technology, Gujarat, India, 2013
Doctorate: Computer Engineering, Charotar University of Science and Technology, Gujarat, India, 2020
The Last Scientific Position: Assoc. Prof. and Principal, Devang Patel Institute of Advance Technology and Research, Charotar University of Science and Technology, Gujarat, India, Since 2023
Research Interests: Computer Networks, Fog Computing, Cloud Computing
Scientific Publications: 37 Papers, 2 Book Chapters, 1 Thesis
Scientific Memberships: ACM