IJCSE International Journal of Computer Sciences and Engineering Open Access
Research Paper Vol.-7, Issue-5, May 2019 E-ISSN: 2347-2693

Load Balancing Strategy based on Genetic Algorithm for Cloud Computing

Tulsidas Nakrani[1*], Dilendra Hiran[2], Chetankumar Sindhi[3]

[1] Research Scholar, Faculty of Computer Science, PAHER University, Udaipur, India
[2] Principal, Faculty of Computer Applications, PAHER University, Udaipur, India
[3] Lead, Sr. Software Engineer, Nividous Software Solutions Pvt Ltd, Ahmedabad, India

[*]*Corresponding Author: nakranitulsidas@gmail.com, Tel.: +91-98797-92211*

DOI: https://doi.org/10.26438/ijcse/v7i5.11061111 | Available online at: www.ijcseonline.org

Accepted: 17/May/2019, Published: 31/May/2019

Abstract— today, cloud computing technology is becoming popular because it provides on-demand services for distributed resources like databases, servers, software, infrastructure, etc. Web traffic and service provisioning is increasing day by day. Load balancing is the biggest challenge in cloud computing, which distribute the workload dynamically across the different nodes to make sure that no node is overwhelmed or underutilized. That can be considered as an optimization problem. A good load balancing must adopt its strategy to the changing environment and the types of tasks. This paper proposes a new load balancing strategy which is based on genetic algorithm. The algorithm thrives to balancing the load of the cloud infrastructure while trying minimizing the make span of a given tasks set. The proposed load balancing policy is simulated using Cloud Analyst. The results of the simulation for sample application show that the proposed algorithm surpassed the existing algorithm like Round Robin, First Come First Serve, and Stochastic Hill Climbing.

Keywords-Cloud Computing, Cloud Analyst, Load balancing, Genetic algorithm

I. INTRODUCTION

The latest large scale distributed Computing is called Cloud. Describes a category of sophisticated IT services on demand or from cloud service providers such as Google, Amazon and Microsoft [1]. This IT infrastructure is used by companies and individuals to admittance from everywhere in the globe. A possible cloud service provider offers software, storage and software as "service". Cloud computing adapts pay as you go modal which can helps the organization to save hardware and software cost [2]. This technology is broadly accepted by industry because of exponential growth of it. As the size of the cloud increases, cloud computing service providers require mass demand management. The biggest challenge is to maintain the same or better performance every time an epidemic occurs. Therefore, despite the bright prospect of cloud computing, many critical aspects must be explored for their perfect realization [3]. One of these problems is Load balancing.

It is considered as one of the essentials to use the full assets of parallel and circulated frameworks. Load Balancing permits appropriation of outstanding task at hand crosswise over at least one server, data centres, hard drives, or other registering assets, along these lines giving Cloud Service Providers (CSP) a component to convey application demands over any number of utilization arrangements situated in server farms. Burden adjusting components can be extensively sorted as concentrated or decentralized, dynamic or static, and occasional or non-intermittent. There has been not much research on load balancing procedures in distributed computing condition [4]. It utilizes Minimum Execution Time (MET) to allot request to each activity in subjective way to the hubs on which it is relied upon to be executed quickest, paying slight reverence to the current load on that centre. Utilization of some current allocation systems like Min-Min, Round Robin and FCFS for load balancing additionally exists in writing. An insightful policy for load balancing proposed by B. Jana et. al. [5]. They propose a novel model to adjust information circulation to improve distributed computing execution in information serious applications, for example, dispersed information mining. Some soft computing strategies like Ant Colony [5] is also discussed in literature.

In this paper soft computing approach called genetic algorithm has been proposed which use the natural selection procedure system. For analysis of an algorithm CloudAnalyst visual simulator is used. The different algorithms like FCFS, RR and Stochastic hill climbing are compared with the outcome of this algorithm [6]. We organized the rest of the paper follows. Section-II covers related work is done regarding load balancing technique. Section-III covers methodology to propose the GA algorithm for load balancing. Section-IV presents the results of the simulation and its analysis with a general description of CloudAnalyst. Finally, section-V covers Conclusion and future scope.

II. RELATED WORK

The load balancing technique, FCFS with the policy of the nearest data centre broker to allocate assets for virtual machines, the results of the FCFS algorithm are compared with the existing known algorithms that include RR and the accelerated algorithm. Response time(RT) is shorter in some clusters than in RR and accelerated algorithm [6].The problem of load distribution in different hosts of a scattered system solved in present work to improve the utilization of resources and the response time of the work by analyzing a variant of the RR algorithm. Overload and load situations are avoided. Load balancing ensures all processors and node in the system runs approximately the similar workload at any given time. The proposed algorithm shows a improved retort time than the former algorithms[7]. A Stochastic Hill climbing local optimization approach is used to assign jobs to virtual machines (VMs). There are two main families of procedures for solving an optimization problem. Complete methods that guarantee both the search for a valid transfer of values to variables, and the proof that there is no such activity. These methods often show good performance and guarantee a correct and optimal response for all inputs. In the cloud computing, in the worst-case they need exponential time that is not acceptable. The other methods cannot guarantee proper answers for all input. Instead of that these methods find satisfying task to solve exertion with high probability. A variant of the Hill Climbing Stochastic Hill Climbing algorithm is one of imperfect approaches to solving these optimization problems. The local and stochastic optimization algorithm be a cycle that moves continuously in the upward direction value that is uphill. It stops when it reaches a peak where no neighbour has a higher value. This variant randomly selects upward movements and the probability of selection may vary with the increase in upward movement. Therefore, activities are associated with a set of actions by making minor changes to the original activity. Each element of the kit is evaluated based on some criteria designed to approach a valid task to improve the results of the status assessment.

III. METHODOLOGY

Even if Cloud computing is energetic in nature but load balancing problem is formulate by assigning N jobs to M processing unit at any given point of time[8]. Following symbols are used to formulate the load balancing dilemma using genetic algorithm.

(P_{uv}) is calculated for all processing unit. Each vector consists of $N_{ips.}$ The cloud service provider needs to pay estimate penalty (C_d) to customer in the event of job finishing late than predefine deadline given by service provider.

Table 1: Symbols used to formulate genetic algorithm for load balancing

P_{uv}	Processing unit vector
N_{ips}	No of instructions executed by the machine per second (in Millions)
C_{ei}	Cost of execution of instruction
C_d	Cost of delay in execution of instruction
J_{uv}	Job unit vector
t	Type of service required by the job
N_{ic}	Number of instructions present in the job
T_{jv}	Job arrival time
T_{wc}	Worst case completion time
w_1,w_2	Predefined weight

$$P_{uv} = f(N_{ips,} C_{ei,} C_d) \quad (1)$$

Likewise

$$J_{uv} = f(t, N_{ic,} T_{jv}, T_{wc}) \quad (2)$$

Where t represents the type of service required by the job like SAAS,PAAS and IAAS. The Cloud service supplier needs to distribute *N* Jobs to all *M* processors such that objective function Z is minimized as shown in (3).

$$Z = w_1 * C_{ei,} (N_{ic} * N_{ips}) + w_2 * T_{wc} \quad (3)$$

It is very difficult to decide / optimize weights, a criterion could be that the more the factor is general, the greater the weight. Logic is the preference or importance of users for one factor in particular compared to the other. Here the optimization has been performed on the set of weights of the data. The weights are measured as weight1(w1) = 0.8 and weight(w2) = 0.2, so their total is 1.

Therefore, the trouble of load balancing is intricate and that can be considered as a computationally obdurate problem. Such a trouble cannot be manipulated by linear programming, so it is rather hard to find the best possible solution globally using deterministic polynomial time algorithms or rules. GAs [9] is considered one of the largely widely used artificial intelligent techniques used primarily for effective search and optimization. It is a stochastic local search algorithm that is based on natural and genetic selection mechanisms. GAs has been revealed to be very resourceful and stable in finding optimal global solutions, especially in the complex and / or large research space. In this manuscript, GA is anticipated as a load balancing policy for CC to find optimal solution i.e. to find the processors to assign a workload to that processor.

The arrival of a job is considered linear and the rescheduling of the works is not considered, since the solution will have an optimal global nature. A proposed algorithm is explained in this section.

A. Proposed Algorithm

A simple GA has three main operations like selection, operation and replacement. The advantage of this technique is

that it can manage a large research space, applicable to complex objective functions and can avoid being stuck in an local optimal solution. The GA working principle used in load balancing in CC is shown in Figure 2. The GA details are described below.

1).Generation of the initial population: GA works in the representation of a fixed bit string of an individual solution. Therefore, all possible solutions in a solution space are encoded in binary strings. From this, an initial populace of ten (10) many chromosomes is randomly selected.

2). Crossover: The purpose of this phase is to select most of the time people better than torque set for the crossover. The fitness value of every single chromosome is calculated using the fitness function proposed in 3. This series of chromosomes is randomly exposed to a single point where the portion depends on the cut on one side of a site. The passage is replaced with the other side. This generates a new couple of folks.

3). Mutation: a very small value (0.05) is now taken as the probability of mutation. Depending on the value of the mutation, the chromosome bits alternate from 1 to 0 or from 0 to 1. The production of this is a new pairing set prepared for crossing.

This GA procedure is recurring until the most suitable chromosome is found (optimal solution) or the termination condition (maximum iteration number) is exceeded.

The proposed algorithm steps are as follows:

Step 1: After encoding in binary string, it arbitrarily initialize a populace of dispensation units.[Start].

Step 2: Estimate the strength value of every populace using (3) [Fitness].

Step 3: Do following until utmost iteration are exceed or best possible solution is not found.

Step 3 (a): Consider the chromosome with the smallest strength two times and reduce the chromosome with the maximum fitness value to build the coupling group [Selection]

Step 3 (b): cross a single point by randomly selecting the crossing point to form a new offspring. [Crossover].
Step 3 (c): mutate new offspring with a probability of mutation of (0,05) [Mutation].

Step 3 (d): put the fresh progeny new population and use this population for the next iteration cycle [Acceptance].

Step 3 (e): test for the final condition [test].

Step 4: stop

IV. RESULTS AND ANALYSIS OF SIMULATION

The anticipated GA algorithm is pretend by considering an "Internet Banking" situation of an intercontinental bank in a CloudAnalyst toolkit [10].

A. Cloud Analyst

To maintain the infrastructure and relevance level needs arising from the cloud computing concept, such as on-demand virtualization modeling, resource simulators are required. Few simulators are available such as CloudSim[11] and CloudAnalyst. In this paper simulation tool , CloudAnalyst is used. In Figure 1 (a) shows GUI of simulation tool and (b) shows its architecture. It is GUI based simulation tool and we can make different experiment on it.

CloudAnalyst uses the functionality of CloudSim and runs a GUI-based simulation. It allows setting parameters to configure a simulation environment to study any cloud research problem [12, 13]. Depending on the parameters that the tool calculates, the simulation result also shows them in graphic form.

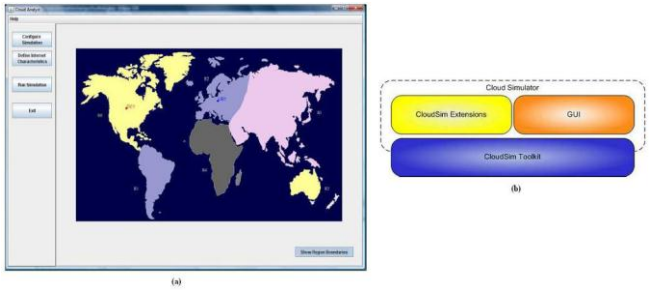

Figure 1. CloudAnalyst (a)CloudAnalyst GUI (b)Architecture of Cloud Analyst

A hypothetical configuration was generated using CloudAnalyst. Where, the world is separated into 6 "Regions" that coincide with the 6 main continents of the world. Six "User Bases" are considered as a model of a collection of user representative the six main continents of the globe. It has been considered a particular instance region for all client bases and it is supposed that there are a number of users registered online during peak hours, of which only one out of 20 is online throughout the initial a small amount of hours. Table 2 shows the details of the client bases used for the experimentation. Each simulated "data centre host" has a particular number of dedicated virtual machines (VMs) for the application. Each machine has 4 GB RAM and 100 GB of storage space and each machine has 4 CPUs and each CPU has a capacity of 10,000 MIPS.

B. Simulation Configuration

Different scenarios are consider for testing from a single centralized data centre in the cloud (DC). Therefore, all requests from users around the world are processed by this unique DC which has 75, 50 and 25virtual cloud configuration (CC) machines assigned to the request. This simulation construction is shows details in Table 3 with the calculated global average response time (RT) in ms for GA, SHC, RR and FCFS. Figure 2. shows performance analysis. Combination of 25,50 and 75 Virtual machines for two data center are shown in table 4 and the performance analysis is shown in Figure 3. Subsequently three(3),four(4),five(5) and Six(6) data centres are considered with a combination of 75,50 and 25 VMs for each CC as shown in Tables 5, 6, 7 and 8.

Table 2. Configuration of simulation environment

Sr. No	User Base	Region	No of users in Peak hrs.	No of users in off Peak hrs.
1.	UB1	0-North America	4,10,000	85,000
2.	UB2	1-South America	5,10,000	1,25,000
3.	UB3	2-Europe	3,40,000	75,000
4.	UB4	3-Asia	7,80,000	1,35,000
5.	UB5	4-Africa	1,35,000	22,000
6.	UB6	5-Oceania	1,45,000	45,500

C. Simulation Configuration

The study of the complexity of any algorithm includes time-space complexity of an algorithm. The basic operations performed in the genetic algorithm are the calculation of fitness and the selection operation, the crossover operation and the mutation operation. In the genetic algorithm, the initialization of the population is measured to be pre-processed, so its complication is not considered for analysis. To encode in a binary sequence a time complexity is maximum n1, for the evaluation of the function (3) it is maximum (c × k) to verify the cost c of the k chromosomes. The selection process has a time complexity is maximum m, for a crossing point, the time complexity is at most m, where m is the length of chromosome and for the mutation wherever m. The three operations of GA are repeated iteratively until the ending process criteria are satisfied, so that the total time complexity G is given by,

$$G = O\{n_1 + (c \times k) + (n_2 + 1)(m + m + m)\} \quad (4)$$

Table 3: Simulation status and average response time

Sr. No.	Cloud Config.	No of VMs in each Data Center	Res. Time using GA (in ms)	Res. Time using SHC (in ms)	Res. Time using RR (in ms)	Res. Time using FCFS (in ms)
1.	CC1	25	328.02	328.05	331	331.12
2.	CC2	50	327.95	328.02	328.85	328.45
3.	CC3	75	245.12	328.75	328.98	328.98

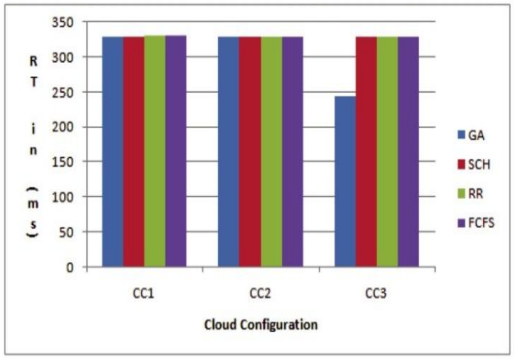

Figure 2. Analysis of Performance of GA,FCFS, RR and SHC Results based on one data center

Table 4. Simulation setting and average response time

Sr. No	Cloud Config.	No of VMs in each Data Center	Res. Time using GA (in ms)	Res. Time using SHC (in ms)	Res. Time using RR (in ms)	Res. Time using FCFS (in ms)
1.	CC1	25	360.75	364.98	371.25	376.32
2.	CC2	50	356.01	359.00	367.47	373.01
3.	CC3	75	354.99	360.01	365.01	371.00
4.	CC4	25,50	351.01	357.05	363.05	369.12
5.	CC5	25,75	351.59	356.97	364.24	367.22
6.	CC6	75,50	351.98	357.03	362.04	359.98

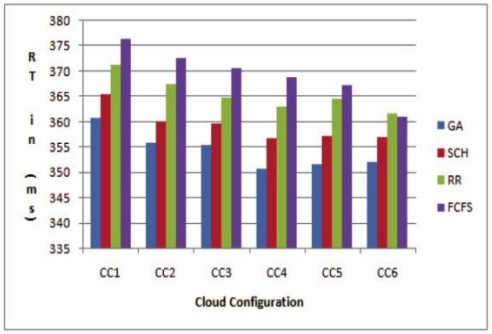

Figure 3. Analysis of Performance of GA,FCFS, RR and SHC Results based on two data center

Table 5. Simulation scenario and average response time

Sr. No.	Cloud Config.	No of VMs in each Data Center	Res. Time using GA (in ms)	Res. Time using SHC (in ms)	Res. Time using RR (in ms)	Res. Time using FCFS (in ms)
1.	CC1	25	350.02	356.62	360.97	363.14
2.	CC2	50	349.98	355.05	362.29	363.32
3.	CC3	75	345.89	350.53	355.98	361.37
4.	CC4	25,50, 75	345.78	349.89	356.01	360.66

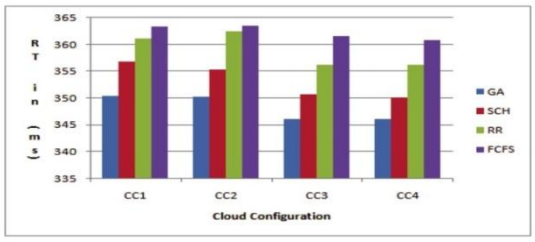

Figure 4. Analysis of Performance of GA,FCFS, RR and SHC

Results based on three data centre

Table 6. Simulation scenario and average response time

Sr. No	Cloud Config.	No of VMs in each Data Center	Res. Time using GA (in ms)	Res. Time using SHC (in ms)	Res. Time using RR (in ms)	Res. Time using FCFS (in ms)
1.	CC1	25	348.76	354.26	359.27	360.85
2.	CC2	50	345.45	350.62	356.84	359.86
3.	CC3	75	340.56	346.37	351.99	358.35
4.	CC4	25,50, 75	337.79	344.18	351.02	355.89

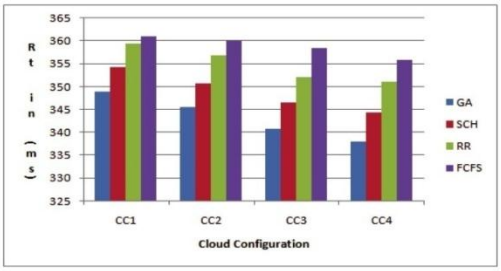

Figure 5. Analysis of Performance of GA,FCFS,RR and SHC

Results based on four data center

Table 7. Simulation scenario average response time

Sr. No	Cloud Config.	No of VMs in each Data Center	Res. Time using GA (in ms)	Res. Time using SHC (in ms)	Res. Time using RR (in ms)	Res. Time using FCFS (in ms)
1.	CC1	25	335.62	342.76	348.56	352.04
2.	CC2	50	325.98	332.85	339.75	345.45
3.	CC3	75	322.91	329.45	335.86	342.78
4.	CC4	25,50, 75	319.97	326.65	333.98	337.96

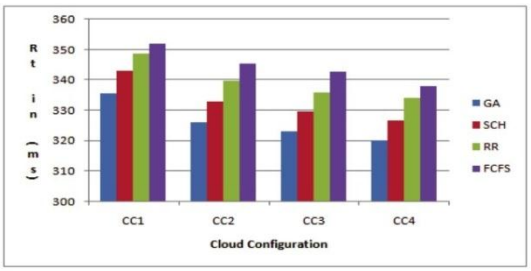

Figure 6. Analysis of Performance of GA,FCFS, RR and SHC

Results based on five data center

Table 8. Simulation scenario and calculated overall average response time

Sr. No	Cloud Config.	No of VMs in each Data Center	Res. Time using GA (in ms)	Res. Time using SHC (in ms)	Res. Time using RR (in ms)	Res. Time using FCFS (in ms)
1.	CC1	25	330.55	336.95	341.85	349.25
2.	CC2	50	322.99	331.55	338.12	344.01
3.	CC3	75	321.52	327.76	333.47	339.85
4.	CC4	25,50, 75	315.30	323.54	331.47	338.30

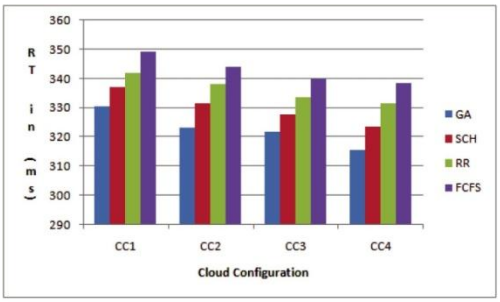

Figure 7. Analysis of Performance of GA,FCFS,RR and SHC

Results based on six data center

V. CONCLUSION AND FUTURE SCOPE

In this paper, a load balancing strategy based on genetic algorithms for cloud computing has been designed to provide proficient use of assets in the cloud environment. The investigation of the results shows that the proposed load balancing strategy not only outperforms some existing techniques, but also guarantees the QoS requirement of the customer job. Although it has been assumed that all jobs have the same priority, which may not be the real case, this can be accommodated in the JUV and can later be taken care of in the fitness function. A very simple GA approach was also used; however, variation and cross-selection strategy might be applied as future work to achieve more efficient and adequate results.

REFERENCES

[1] R. Buyya, J. Broberg, and A. M. Goscinski, *Cloud Computing: Principles and Paradigms*. John Wiley & Sons, 2010.

[2] M. D. Dikaiakos, D. Katsaros, P. Mehra, G. Pallis, and A. Vakali, "Cloud Computing: Distributed Internet Computing for IT and Scientific Research," IEEE Internet Computing, vol. 13, Issue. 5, pp. 10–13, 2009.

[3] R. Mishra, "Ant colony Optimization: A Solution of Load balancing in Cloud," IJWesT, vol. 3, Issue. 2, pp. 33–50, 2012.

[4] B. Mondal, K. Dasgupta, and P. Dutta, "Load Balancing in Cloud Computing using Stochastic Hill Climbing-A Soft Computing Approach," Procedia Technology, vol. 4, pp. 783–789, 2012.

[5] B. Jana, M. Chakraborty, and T. Mandal, "A Task Scheduling Technique Based on Particle Swarm Optimization Algorithm in

Cloud Environment," in Soft Computing: Theories and Applications, pp. 525–536, 2019.
[6] F. Saeed, "Load Balancing on Cloud Analyst Using First Come First Serve Scheduling Algorithm," in Advances in Intelligent Networking and Collaborative Systems, pp. 463–472, 2019.
[7] G. Liu and X. Wang, "A Modified Round-Robin Load Balancing Algorithm Based on Content of Request," in 2018 5th International Conference on Information Science and Control Engineering (ICISCE), pp. 66–72, 2018.
[8] K. Dasgupta, B. Mandal, P. Dutta, J. K. Mandal, and S. Dam, "A Genetic Algorithm (GA) based Load Balancing Strategy for Cloud Computing," Procedia Technology, vol. 10, pp. 340–347, 2013.
[9] P. Devarasetty and S. Reddy, "Genetic algorithm for quality of service based resource allocation in cloud computing," Evol. Intel., 2019.
[10] B. Wickremasinghe, R. N. Calheiros, and R. Buyya, "CloudAnalyst: A CloudSim-Based Visual Modeller for Analysing Cloud Computing Environments and Applications," in 2010 24th IEEE International Conference on Advanced Information Networking and Applications, pp. 446–452, 2010.
[11] R. N. Calheiros, R. Ranjan, A. Beloglazov, C. A. F. De Rose, and R. Buyya, "CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms," Software: Practice and Experience, vol. 41, issue. 1, pp. 23–50, 2011.
[12] A.A. Ekre, N.M. Nimbarte, S.V. Balamwar, "An Empirical Proposition to Load Balancing Effectuate on AWS Hybrid Cloud", International Journal of Scientific Research in Computer Science and Engineering, Vol.6, Issue.4, pp.9-17, 2018
[13] Deepti Sharma, Vijay B. Aggarwal, "*Dynamic Load Balancing Algorithms for Heterogeneous Web Server Clusters*", International Journal of Scientific Research in Computer Science and Engineering, Vol.5, Issue.4, pp.56-59, 2017

Authors Profile

Mr. T V Nakrani pursed Bachelor of Computer Application from North Gujarat University, Patan, Gujarat , India in 2003. and Master of Computer Application from Gujarat University, Ahmedabad, India in 2006. He is currently pursuing Ph.D. and currently working as Assistant Professor in Department of Computer Application(MCA), Sankalchand Patel University, Visnagar, Gujarat, India since 2011. His main research work focuses on Cloud Computing technology. He has 13 years of teaching experience and 3 years of Research Experience.

Dr. Dilendra Hiran pursued Ph.D in Computer Science from Pacific University, Udaipur in 2015. He Completed his Mmaster of Science in Mathematics and Computer Science in 1999 and Bachelor of Science in Mathamatics from MLSU, Udaipur in 1994. He is currently working as Principal, Faculty of Computer Application at Pacific University, Udaipur

Dr. Chetankumar K Sindhi pursued Ph.D. in Computer Science and Application from Hemchandracharya North Gujarat University, Patan in 2011. He completed his Master of Science in Industrial Mathematics and Computer Science & Application, Bachelor of Science in Mathematics from the same University respectively in 2005 and 2002. He is currently working as a Lead, Sr. Software Engineer in Nividous Software Solutions Pvt Ltd. He has 14 years of experience in Academics, Industry, and Research.

