ALVEC, auto-scaling by lotka volterra elastic cloud: a qos aware non linear dynamical allocation model

Goswami, Bidisha and Sarkar, Jyotirmoy and Saha, Snehanshu and Kar, Saibal and Sarkar, Poulami

Centre for Studies in Social Sciences, Calcutta

15 February 2018

ALVEC: Allocation in Lotka Volterra Elastic Cloud: A Non Linear Dynamical, QoS aware Allocation Model in Cloud

Bidisha Goswami, *Member, IEEE,* Jyotirmoy Sarkar, *Member, IEEE,* Snehanshu Saha, *Senior Member, IEEE,* and Saibal Kar

***Abstract*—Elasticity in resource allocation is still a relevant problem in cloud. There are many academic and white papers which have investigated the problem and offered solutions. Unfortunately, there is scant evidence of a noteworthy elastic allocation scheme. Elasticity is defined as the ability to adapt with the changing workloads by provisioning and de-provisioning Cloud resources. We propose ALVEC, a novel model, inspired by population dynamics and Mathematical Biology, of resource allocation in Cloud data centers which addresses dynamic allocation by auto-tuning model parameters. The proposed model, governed by a coupled differential equation known as Lotka Volterra (LV), fares better in Service level agreement (SLA) management and Quality of Services (QoS). We show evidence of true elasticity, in theory and empirical comparisons. Additionally, ALVEC is able to predict the future load and allocate VM's accordingly. A business opportunity for small scale cloud providers with a quantitative measurement of profitability has also evolved as a fallout of better SLA management.**

***Index Terms*—Cloud data centers, resource allocation, elasticity, Lotka Volterra (LV), population dynamics.**

I. Introduction

Cloud computing is an Internet-based computing that provides shared processing resources and data to computers and other devices on demand. The shared pool of computing resources can be rapidly provisioned and released with minimal management effort. The Cloud subscriber uses computational resources on short notice, on either subscription or pay-per-use model. This technology allows the enterpriser to avoid upfront infrastructure costs and lets them focus on their business, instead of on infrastructure. A further advantage is that the unit cost of operating a server in large farm is relatively lower than in small data-centers. Cloud provides virtual machines which accept the user requests and allocate the available physical resources accordingly. Cloud service provider acts as a broker between user requests and the Cloud. The major challenges that confront these Cloud service providers are provisioning the Cloud resources in a dynamic environment without compromising the quality of service and maintaining the quality of service while lowering the cost.

Bidisha Goswami and Snehanshu Saha are with the Department of Computer Science and Engineering, PES Institute of Technology South Campus, Bangalore, India USA e-mail: snehanshusaha@pes.edu.

J. Sarkar is with GE Healthcare, India.

S. Kar is with Centre for Studies in Social Sciences, Calcutta, India and IZA Bonn.

This paper has tried to implement an optimization model for Cloud data centers so that no resource should be under-provisioned / over-provisioned. When the demand of Cloud resources is highly volatile, the chances of over-provisioning / under-provisioning of resources is very common. **Some implementation of biological concept has already shown good results in computational platforms.** This paper introduces a new model inspired from population dynamics model to control the system for maximizing the utilization of every resource. This implies that no resource is under / over provisioned. **Predator-Prey is one of the application areas of population dynamics** where the population are inter-dependent on each other. The study of population dynamics focuses on all the members of a single species who live together in the same habitat and are likely to interbreed, their unique physical distribution in time and space, growing or shrinking rate of population, etc. The predator-prey behavior signifies, if food is available in large quantity, then high food consumption increases the population of predator and the large amount of prey consumption reduces the number of prey. At this point, because of scarcity of food available, number of predators decrease. This is how the predator-prey model maintains both the populations dynamically. A similar kind of inter-dependency can be observed **in the behavior of the Cloud provider and the Cloud consumer.** This paper considers the resources as prey population and Cloud customers as predators. When huge resources are available, the customers can make use of sufficient options to avail the resources. Again, when large amount of resources is consumed by the customers, the unavailability of resources can deprive the customers. This is the point where Cloud provider faces the challenge of elasticity. This either challenges the violation of SLA(Service Level Agreement), as promised direct for penalty, or additional resource provisioning which increases the cost.

This paper proposes a model based on predator-prey behavior. The model discusses various situations that can take place based on the behavior of customers and the available resources. The model has two fundamental equations which control the cloudlets and Virtual Machines(VM). The following section shows how different parameters in the said equations can control and stabilize the proposed model. The proposed model is a resource scheduling algorithm, where different levels of

dynamics in predator and prey population can be controlled by varying the parameters of the equations to control the system breakdown. The model also provides a mathematical property, known as the limit cycles, which is described in contour portraits, also known as phase portraits of the system. Limit cycle describes a qualitative limit for the stability of a system whose parameters are differed such that the system grows out of stability. The difference acquired by these parameters is measured to tell the domain of stability. This has direct application in understanding the stability of a web-server with incoming requests. Limit cycle of a system along with rate of incoming requests, can help us understand the bounds of a system.

Let us consider a hypothetical scenario in the Amazon rainforest where goats roam free without fear of being attacked or ambushed. Except natural death, the population doesn't diminish, infact, is balanced by reproduction. The grassland may lose all the green since the goat population is not controlled. Whenever that happens, it is disastrous for goats as well since they'll have nothing left to eat. This may lead to migration and other critical consequences. On the contrary, if all goats are either killed or dead because of some natural calamity, the grassland is not consumed and for the lack of predators (goats) grass may grow in an uncontrolled fashion. Evidently, the balance between the two populations need to be maintained for a healthy ecosystem.

As the model of the paper is based on the dynamic interaction between the predator and the prey, in our proposed approach, **the demand of some degree of autonomy is needed to enable the components to respond to dynamically changing circumstances.** The paper addressed this problem and additionally, solves the allocation problem by using the Lotka Volterra (LV) model.

II. Problem Statement: **REWRITE**

To propose a novel model of resource allocation in Cloud which addresses dynamic allocation, tunes the parameters of the model as per the on-demand service, shows elastic comparison to other existing models in the market, is better in SLA management, has much better Quality services with proven comparison and results, is able to predict the future load and allocate the VM accordingly and also proposes a business model for the new small scale Cloud provider with a quantitative measurement of profitability.

III. Our Contribution:**REWRITE**

- **Novelty of the model:** LV is extensively used in population biology. However, to the best of our knowledge, no application of the Lotka Volterra (LV) model in Cloud coputing, in particular or communication networks, in general is found. This has the potential to set new baseline of research in Cloud Computing.
- **Imparting Dynamic behavior to CloudSim: CloudSim is a framework for modeling and simulation of Cloud Computing infrastructures and services.** Here, the dynamic simulation model is compared with a biological model called Lotka-Volterra [33] [34] [35]. The resources on CloudSim are compared with Lotka-Volterra model. **write the section number and page number. Refer section VIII**
- **Elasticity: In the context of Cloud Computing, elasticity is defined as an ability** to adapt with the changing workloads by provisioning and de-provisioning Cloud resources. The proposed model is an elastic system which provisions / de-provisions virtual machines as per demand. Most algorithms and strategies designed handle elasticity by increasing/decreasing VM's by one, by manual intervention or predefined rules. We control the change in number of VM allocation/de-allocation exploiting the model dynamics proposed in our approach. This is a fundamental contribution and is discussed in conclusion, further. **write the section number and page number details are in subsection X-C**
- **VM prediction based on population parameters:** The proposed model specifies one upper and one lower threshold. In the case of upper threshold violation by future response time prediction, new VM's need to be added to service, to neutralize the situation. In the case of lower threshold violation, VM's need to be deallocated from the user service, as more than required number of VM's have been allocated to increase utilization of the resources. **write the section number and page number**
- **Improvement in QoS parameters: make-span, response time:** Make-span is the total duration between the beginning and the end. **Response time is the summation of waiting time and execution time.** Both are considered as important quality parameters. The experiments show significant improvement in these parameters. **write the section number and page number**
- **Parameter Tuning:** Parameter tuning implies controlling/ influencing the outcome of the parameters, which are nothing but VM's and cloudlets specified in LV model by changing the different coefficient parameters and satisfying the relevant conditions. In this paper, we have exhibited how to cater to three different situations such as Prey Increasing-Predator Decreasing, Prey Decreasing-Predator Increasing and stability of Prey-Predator by tuning the parameters. **write the section number and page number**
- **Scheduling algorithm:** The proposed scheduling algorithm mimicking the existing ecological model (non-linear in nature) address the dynamic nature adequately. The related papers show that the increase in the VM population is static and linear. The LV model decides the number of future VM allocation as per predicted need. Outside the purview of SLA, if unanticipated load needs to be handled, the model accommodates. **write the section number and page number**
- **Mitigating stiff entry barrier:Business model** Study shows huge business growth in the area of Cloud. In developing countries, many small and medium industries can target to enter this business. Presently, the Cloud market is controlled by a few major market players and small firms struggle with entry-barrier issues. **Barriers to entry are the existence of high startup costs or**

other obstacles that prevent new competitors to join the target business. The paper proposes a penalty based profit model benefiting incumbent enterprises. This is especially applicable to incumbents in Cloud data centers. **write the section number and page number**

- **Application significance:** The proposed model is the first of its kind to balance the dynamics and auto-correct over or under- utilization of resources. The applications are relevant in general cloud dynamics and data centers in particular.

Remainder of the paper is organized as follows. Section IV presents key definitions used from population biology and the relevant mathematical model. It is important to familiarize the readership with these definitions so that the mapping between LV and the Cloud problem be established clearly in section V. The analytical model presented in section V has to be solved numerically and qualitatively.These solutions along with interpretation have been documented in sections VI and VII respectively. Section VIII prsents the simulation and outcome in detail. This helps us followed by a detailed analysis of the benefits of service based outcomes in sections IX and X. One of the indirect consequences of the proposed model is discussed in section XI. This might have a major economic impact. We conclude the paper by discussing the advantages and pitfalls of our approach and background work. The following flowchart captures the workflow.

IV. Key Definitions with internal referencing

- Stability: As per dynamic stability definition, the trajectories do not change too much under small perturbations. Stability implies no significant changes in model. In cloud environment stability is a condition where no significant changes occur in the VM or cloudlets. Hence, within **a certain period of time if both the numbers do not** change, there would not be any volatility in the model. This is the condition of stability. In this paper we refer to Cloud dynamics as the increment or decrement of number of VM. The number of cloudlets do not change significantly under the discussed situation.
- Predator: Predator are the consumers of Prey. In this paper we refer to cloudlets as Predator which is consuming VM.
- Prey: Prey are consumed by the next layer of food-chain. Here is a single layer of prey, which is Virtual Machine.
- **Qualitative theory of differential equations: In mathematics, the qualitative theory of differential equations studies the behavior of differential equations by means other than finding their solutions.** The paper studies equation1 and 2 which signify the dynamics of Prey and Predator population under different circumstances.
- Equilibrium and stable condition: Equilibrium point is a constant solution to a differential equation. Figure-2 explains a phase plot with a stationary point. The tentative point has no impact on existing VM number or existing Cloudlet number. Controlling parameter from outside is able to control the system stability. This is the equilibrium point.
- Phase plane: Phase plane analysis is one of the most important techniques for studying the behavior of nonlinear systems. There is a direct method to show the existence of limit cycles. Figure-2 is a phase-plane between Virtual Machines and cloudlets which explains the area where neither population creates impact on the other and shows the area where both the population are independent of each other.
- Nullcline: In mathematical analysis, nullclines are encountered in a system of ordinary differential equations. The nullclines divide the phase portraits into regions. In this paper, the nullcline is equivalent to making equations 1 and 2 zero.

V. Our Model: Theory, Relevance and Applications

A. Relevance of Predator-Prey Model and Population Dynamics in Cloud Data Center

The biological population dynamics model, Predator-Prey, can be implemented **in the Cloud environment. This model is relevant to represent the** different scenarios involving VM and cloudlet population. VM and cloudlets can be considered as the Prey and Predator respectively. The VMs can be consumed as long as enough number of VMs exist in the system or the number of VMs can be scaled up to fulfill the need. Death or absence of **prey population ensures the gradual death of predator in the predator-prey** model (please refer to the Amazon rainforest analogy in the introduction). In the same way, if all VMs are consumed or no VM is present in the system, the cloudlets (Predator here) gradually start loosing relevance which is equivalent to death or whole scale migration to another prey population, that indirectly increases the operational cost. The Predator-Prey model can be represented in the Cloud as VM-cloudlet model as follows.

$$\frac{dP}{dt} = \alpha P - \beta PQ \tag{1}$$

$$\frac{dQ}{dt} = \delta PQ - \gamma Q \tag{2}$$

where, P is the number of VM's (Prey); Q is the number of cloudlets (Predators)
$\frac{dP}{dt}$represents the growth rates of VM and $\frac{dQ}{dt}$ represnts growth rate of cloudlets over time.
α is the upscaling rate of VMs in the case of demand (cloudlets);
β is the allocation rate of VM due to the incoming cloudlets; γ is the completion rate of cloudlets; δ is the Cloudlet incoming rate into the system. To analyze the model in detail, the trend of P and the trend of Q need to be investigated. To make the model stable, both the populations have to address the following, $\frac{dP}{dt} = 0$ and $\frac{dQ}{dt} = 0$

$$\alpha - \beta Q = 0, \delta P - \gamma = 0 \tag{3}$$

Equation 3 evaluates a stationary point $(\frac{\gamma}{\delta}, \frac{\alpha}{\beta})$. Fig. 1 represents the phase-portrait for the above dynamics. Notice from Fig. 1 that all variations of population encircle around

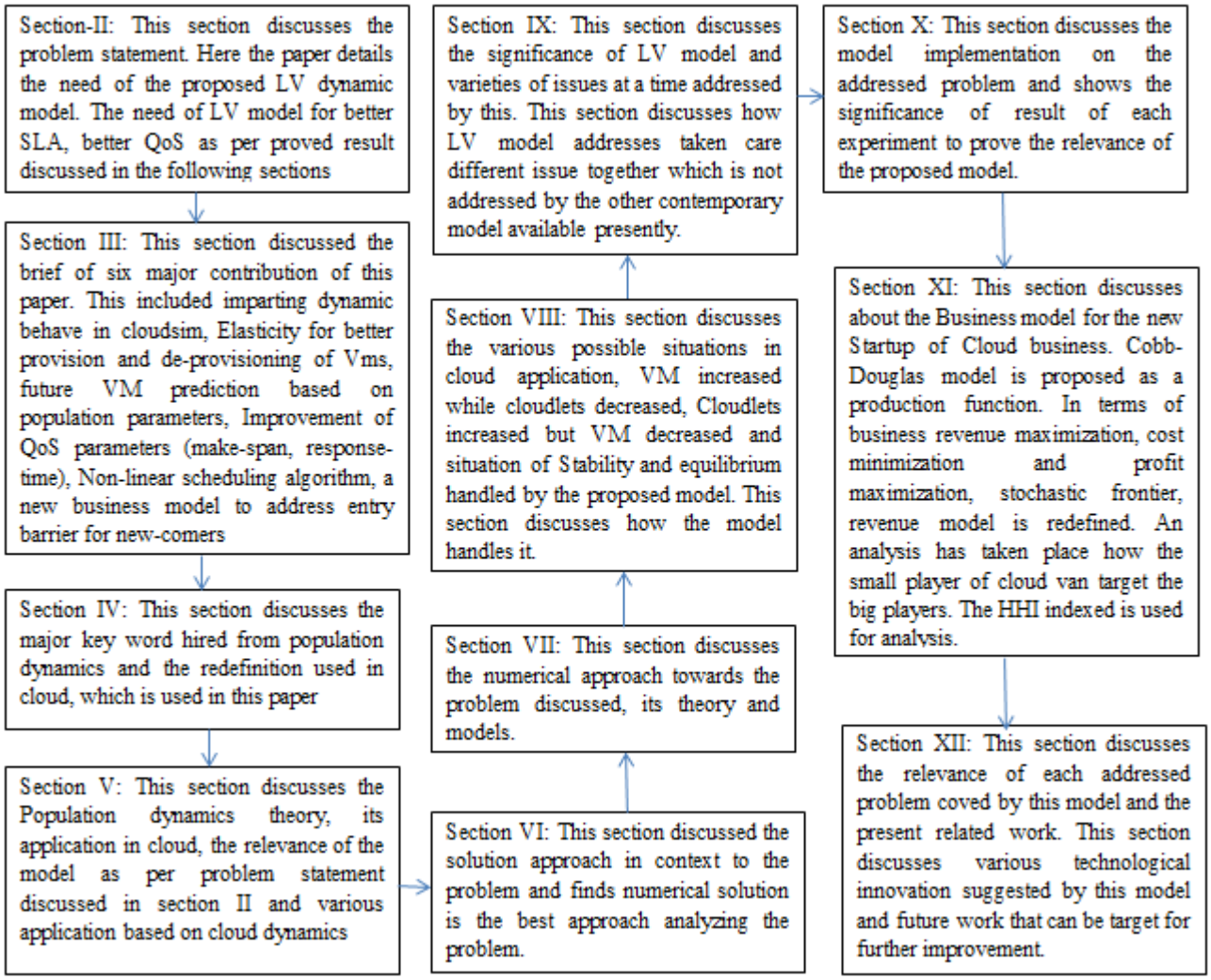

Figure 1: This is the brief overview of all the section discussed in this Paper

a stationary point.

B. Predator Prey Equilibrium and Regions

Equations 1 and 2 are the basic equations to determine the number of both the populations. This is clear that a greater number of VMs in the system is good for the model as multiple resources are present for consumption which makes it robust. Whereas, more population of cloudlets is challenging for the system as it reduces the number of VMs. To understand the stability, null-clining scenario of the predator-prey model should be discussed. In this model, two nullcline situations are possible, P-nullcline and Q-nullcline. P-nullcline is the set of points where $\frac{\partial P}{\partial t} = 0$. Similarly, Q-nullcline is the set of points where $\frac{\partial Q}{\partial t} = 0$. Now, as per the definition, equilibrium points are the locations, where growth rate of predator and prey both become zero. Hence, it can be said that P-nullcline is the location where growth rate of prey becomes zero. This signifies that prey population is neither increasing nor decreasing. On the other hand, the growth rate of predator becomes zero in the region of Q-nullcline. Apart from P-nullcline and Q-nullcline regions, the growth rate of Predator and Prey would be either positive or negative. Therefore the equilibrium points are located in the intersection of P-nullcline and Q-nullcline. The P-nullcline and Q-nullcline can be defined as below

$$\alpha P - \beta PQ = 0 \qquad \delta PQ - \gamma Q = 0 \tag{4}$$

The above equations can be rewritten as

$$P(\alpha - \beta Q) = 0 \qquad Q(\delta P - \gamma) = 0 \tag{5}$$

From the above equations, it can be derived
$P = 0$ or $\alpha - \beta Q = 0$
and
$Q = 0$ or $\delta P - \gamma = 0$
The equilibrium points are $(0,0), (\frac{\gamma}{\delta}, \frac{\alpha}{\beta})$. As we have held $\delta = 1, \beta = 1$,hence the co-ordinate points are $(0,0), (\gamma, \alpha)$. The situation where stability can be achieved is

$$\alpha P = \gamma Q \tag{6}$$

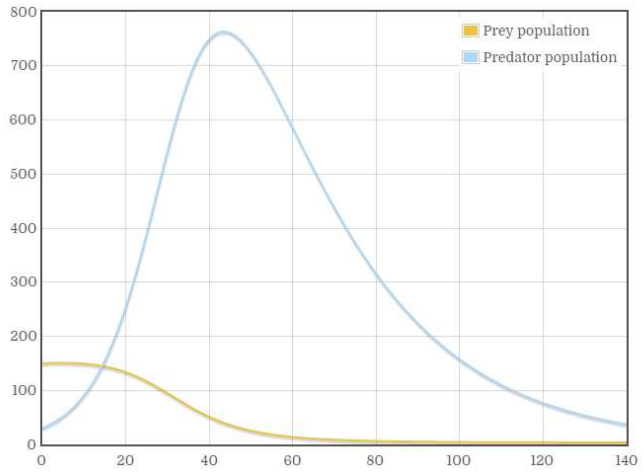

Figure 2: Classical Lotka-Volterra plot for parameters computed: The graph consists of two different species of Food chain : Prey and Predator. In the proposed cloud model, Prey is Virtual Machine (VM) and Predator is cloudlets. The intersection point of predator and prey population is the NullCline point. If the VM population starts declining, then with a phase difference the Cloudlets also decline for lack of resources.

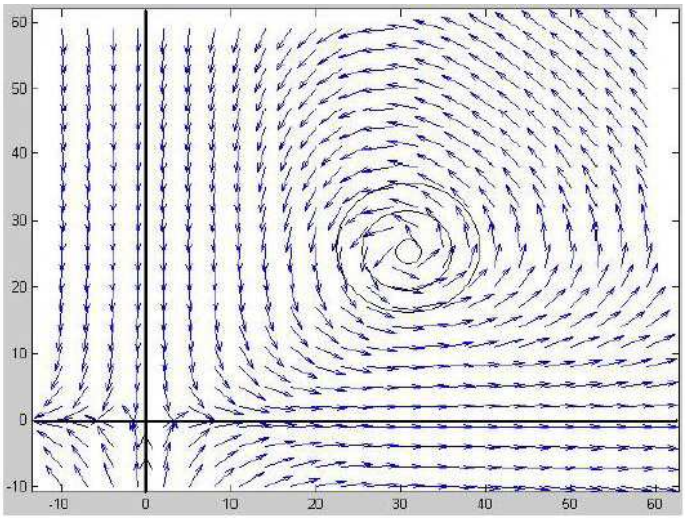

Figure 3: This explains the stationary point $(\frac{\gamma}{\delta}, \frac{\alpha}{\beta})$. The stationary point has no impact on VM or cloudlets population. Controlling these parameters enables us to control the population dynamics and also control both Q and P populations. External management of these parameters determine the stability or instability of the system.

This can be derived from equations above. As in stable situation, there is no growth for predator and prey. Hence,

$$\alpha P - \beta PQ = 0$$
$$=> \alpha P = \beta PQ$$

$$\delta PQ - \gamma Q = 0$$
$$=> \gamma Q = \delta PQ$$

After considering $\beta = 1, \delta = 1$, the above equation can be rewritten as below

$$=> \alpha P = \gamma Q$$

It is already proven that in stable situation, the value of $\alpha = Q, \gamma = P$

$$PQ = PQ$$

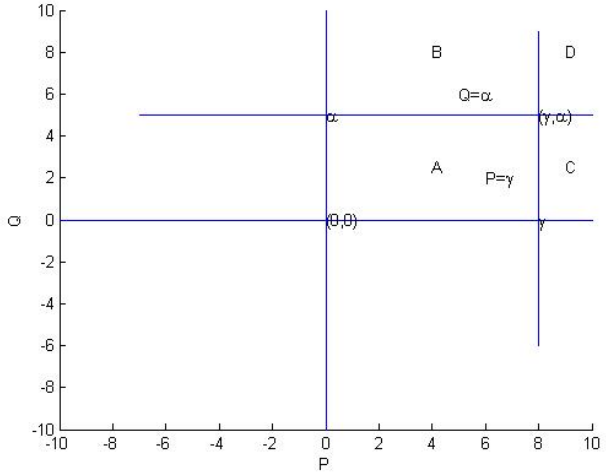

Figure 4: represents the P-nullcline and Q-nullcline equations, **where the x-axis depicts the prey P (Cloudlets) and y-axis showcases** the predator Q (VM). The equilibrium points and different regions are visible in the figure. Two equations $\alpha = Q$ and $\gamma = P$ are plotted in the graph. The intersection of these two equations yields equilibrium points (γ, α). A is the region enclosed by $P = \gamma, Q = \alpha$, P, Q axis, where growth of P(Prey) is positive and the growth of Q(Predator) is negative. B is the region, which is the upper side of $Q = \alpha$ and left side of $P = \gamma$. In this region, the growth of P(Prey) is negative and the growth of Q is negative. Region C is the right side of $P = \gamma$ and the lower side of $Q = \alpha$. The **growth rate of P is positive and the growth rate of Q is positive. Region D defines the** upper side of $Q = \alpha$ and the right side of $P = \gamma$. The growth rate of P and Q are negative and positive respectively.

Here in this section, we will discuss results obtained by simulating the proposed model. In the algorithm, three major scenarios have been highlighted and each of the scenarios need detailed discussions and elaboration. To support the findings and for illustration purpose, tables as well as figures have been added. Matlab tool is used for generating graphs and simulations. In Cloud computing, the number of VMs will be less than the number of cloudlets. Therefore, it is the reverse case of biological Lotka-Volterra model, where the **number of prey is higher than the number of** predator.

VI. Solution to the Proposed Model

- The solution to the proposed system is critical in order to exploit the solution in the simulation and to explore QoS

metrics. In this case, a closed form solution is the most convenient way of bringing out direct relations between the variables, predator (cloudlets) and prey (VM). Such direct relationship is often solicited since it explains the dynamics between the two key entities in Cloud computing. Regrettably, LV equations are inherently complex and do not admit of closed form, analytical solutions. Therefore, alternative methods to interpret and utilize the relationship between predator (cloudlets) and prey (VM) must be sought.

- There are two ways to handle this. We use the qualitative theory to interpret the solutions and represent those in the phase plane (refer to definitions and relevant theory sections, section V). The representation of the solution qualitatively re-establishes our claim that the model is relevant in the context of **resource allocation and related issues in Cloud. However, in the absence of** explicit solutions, it is difficult to proceed further in the direction of exploiting the solutions in simulation and compute/tune parameters for performance enhancement (refer to QoS figs in discussion section).
- We mitigate the problem by computing the solution to LV numerically. The numerical solution is central to our efforts in computing the parameters/coefficients in LV which further aids in accomplishing efficient VM allocation. This is accomplished by Runge Kutta and described in the next section.

VII. Numerical Solution of Lotka-Volterra

To retrieve result more accurately from LV model, we have employed the use of Runge – Kutta methods of fourth and fifth order, implemented by Fehlberge and denoted as RKF45. Numerical analysis to solve LV model is appreciated due to the inherent difficulty of solving the LV model analytically. We proceed in the following manner:

The solution domain is subdivided into a collection of discrete points. RKF45 generates an approximate solution in vector form y_n. We begin with the initial data **at time** $t_0 = 0$ **and apply an estimation formula to generate an approximation solution** at time $t_i = i * h$, i = 1, 2, ...n. The step size h is chosen suitably such that it is not too big or too small. We use RK4 and RK5 (Runge Kutta 4th order and 5th order respectively) **at each step i to produce two different approximate solutions** and compare those. The **approximation is accepted as long as the difference between the two** approximations meet a predefined tolerance. The step size may be modified to accommodate the tolerance criterion. However, we need to increase the step size if the two approximate solutions agree to more significant digits than required. **Numerical methods are sensitive to approximation** and thus the following points must be stressed:

1) We use Taylor's series expansion of the function around the iteration point at each step to approximate a function. This produces truncation error, large or small depending on the number of terms used in the expansion. If h_n denotes the difference between $n+1^{th}$ and n^{th} iteration, then a fourth order method produces an error of the form Ch^5 for some constant C. This means that a step size of magnitude $\frac{h_n}{2}$ shall reduce the error by a factor of $2^5 = 32$.
2) A 5th order Runge-Kutta method requires executing four function evaluations to obtain local truncation error of order 5. We observe, the numerical solution to the ordinary differential equation can be 5^{th} order accurate locally but may still not address the issue of global convergence adequately.
3) Round – off error is inevitable. The estimate of VM's turns out to be a ballpark figure, precisely for this reason.
4) **The standard assumption about the model does not define the exact** population dynamics for the above reasons.

We adopted Runge-Kutta-Fehlberg 45 (RKF45) method, (ode 45 in Matlab) symbolized by function evaluations with an additional evaluation to accomplish 5th order accuracy. This generates a local error of the order h^6, significantly small if h is chosen to be small enough. Please note, h is chosen to be between 0 and 1.

The parameters for simulation are computed using this method. ode45 of matlab (Appendix B), which employs Runge-kutta method is used rigorously to derive the datasets table I, III and IV corresponding to the three cases: Prey Increasing-Predator Decreasing, Prey-Predator stability and Prey Decreasing-Predator Increasing.

VIII. Simulation in CloudSim

The quality parameter, which is being compared within different simulations, is Performance/Request Completion time, which is nothing but the time difference between the first time cloudlet request is submitted to the broker and the cloudlet completion time. We have kept the VM number constant for each data point across simulations and vary the cloudlet number, as more available VM will lead to better performance, which in turn disrupt the fair comparison. All cloudlets have been submitted dynamically. In almost every simulation, the cloudlets are dynamically submitted within a timeframe, which is 1000 ms. Two situations are highlighted in this section. The cloudlets for each simulation have been submitted in three batches, for that purpose the CloudSim code has been modified. Initial batch VM, cloudlets numbers are same for each situation to have a fair comparison.

A. Case 1: Prey Increasing-Predator Decreasing

This scenario may arise, when there is no VM available to provision or number of available VM is nearly 0 and requests for VM is uprising. Such situation can be managed **by reducing the number of cloudlets and increasing the number of** VM. Say we have 30 VM available and at that moment the number of cloudlets is 50. Now, we would like to shoot up the number of VM by increasing/deceasing the constants of the proposed model. In this scenario the condition $\gamma Q > \alpha P$, which is mentioned in the algorithm, has to be met.

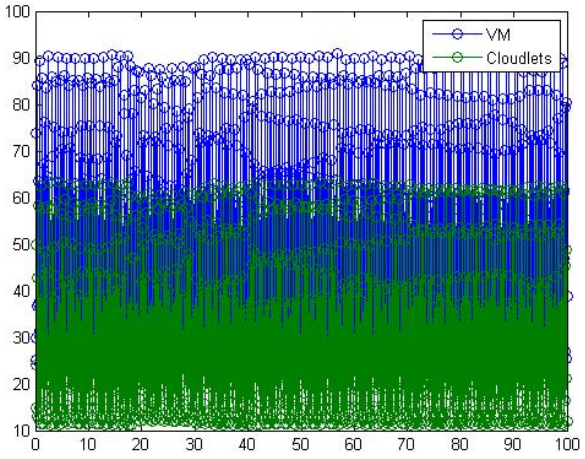

Figure 5: Variation of cloudlets and VMs wrt time (Case 1). In the figure blue, green display the VM and cloudlets respectively. Y- axis represents the time-span, which is taken 0 to 100 duration. Each data point has been plotted at 0.01 time span interval. It is palpable from the figure that VM occupies the **upper part of the figure, whereas the cloudlets cover the lower part.** Hence, in maximum cases the values of VM are higher than cloudlets. The region, where VM and cloudlets values are overlapping lies within 60 to 35. The maximum value, which is belongs to VM lies in the region nearby 90, whereas the lowest value, belonging to cloudlets is nearby 10.

Time	VM	cloudlets
0	30	50
0.1	73.817541	14.87
0.2	25.19	23.96
0.3	84	42.70
0.4	36.78	12.97
0.5	37.24	58.28
0.6	63.62	12.56
0.7	24.35	30.050
0.8	89.46	30.17
0.9	31.52	14.92
1.0	49.80	62.49
1.1	53.56	11.46
1.2	25.13	38.34
1.4	85.68	20.72
1.5	27.62	18.14
1.6	67.18	57.75
1.7	44.61	11.51
1.8	28.59	48.25
1.9	76.06	15.35
2.0	25.04	22.87

Table I. This table demonstrates a scenario (Case 1), where it is required to increase the Prey(VM) number. In the table, time-span from 0 to 20 has been taken **for better understanding of how the model works. Initially, the predator and the prey** numbers are taken as 30, 50 respectively. Time-span is displayed in the table for every 0.1 interval. As the intention was to increase the number of prey from 30, in the next immediate time-span it can be noticed that the prey number surges to 73, a two fold jump from the initial value. Apart from a few occurrences, through out the period till 2.0, the **number of prey is higher than initial value. In the case of predators, the number of predators** are less than the initial value 50. Only one occurrence at time-span 1.0, the predator number is more than initial value. The prey-predator numbers in the table and the figure will be different, if any other initial values are considered. As the proposed model is not a linear function, there is no pattern visible in the prey-predator numbers.

The Lotka-Volterra model, depicted in the Fig. 5 following the equations:

$$\frac{\partial P}{\partial t} = 30P - PQ; \frac{\partial Q}{\partial t} = -50P + PQ$$

where $\alpha = 30, \gamma = 50, \beta = 1, \delta = 1$

8 simulations have been performed where 2 simulation datasets are collected from Table I and remaining datasets are random datasets **(justify random)**. VM number is kept constant across the simulations. Initial data points remain unchanged for simulations to have a fair comparison. Lowest avg completion time is 400, which is visible in simulation 1. Though the difference among various avg completion time is few milliseconds, the performance of simulation 1, which is derived from model is better than others. Here, dynamic influx of cloudlets within 1000 ms is considered, i.e. within 1000 ms, all the cloudlet requests arrive at the data center.

The simulation 1 dataset is part of table I, which is the master dataset derived from the proposed model.

Table II contains the avg. request completion time of Simulation 2. Here the initial data point, which is nothing but the first batch of Cloudlet submission is same as 1st simulation. But for the 2nd and 3rd data points, cloudlets number are higher than the 1st simulation. The simulation 3 presents another random dataset, where the 2nd data point is lower and 3rd data point is higher in comparison than Simulation 1. The simulation 4 is the reverse of simulation 3 as the 3rd data point Cloud number is higher than simulation 1 and the 2nd data point is higher than the 2nd data point of simulation 1. Simulation 5 dataset belongs to the master dataset in table I. Simulation 6,7,8 are performed in random datasets which have been generated based on simulation 5 in a controlled manner. The 2nd, 3rd data point of simulation 6 are higher than 5th simulation which is originated from proposed model. Simulation 7 is giving almost the same performance as simulation 5. The 2nd data point is lower and 3rd is higher than the corresponding data points in simulation 5. In the case of simulation 8, the 2nd data point is higher and the 3rd data point is lower than corresponding data point in simulation 1.

We can draw the conclusion from simulations 5,6,7,8 that the avg completion time of 5 is better than others.

B. Case 2 :Prey-Predator stability

In this section, the situation is highlighted, where data center has achieved its maximum VM utilization target. Therefore, same number of VM and cloudlets need to be maintained afterwards.**It is the situation where the growth/ shrinking rate of VM, cloudlets are 0. Stability implies no change in VM, cloudlets number as time passes. Same VM, cloudlets numbers need to be maintained once the desired utilization**

Simulation No	VM	cloudlets	Avg Request Completion time
1	30	50	499.76
1	36	12	400
1	76	15	400
2	30	50	548.24
2	36	112	536.67
2	76	115	540.62
3	30	50	531.92
3	36	8	506.25
3	76	115	543.64
4	30	50	513
4	36	200	530
4	76	5	487
5	30	50	493.28
5	85	20	400
5	44	11	400
6	30	50	514.1188
6	85	200	686.6909
6	44	150	692.715
7	30	50	543.08
7	85	10	400
7	44	110	400
8	30	50	512.52
8	85	110	479.44
8	44	8	552.5

Table II. This table represents the simulations where Cloudlets (Predator) need to decrease and VM (Prey) number is supposed to increase (Case 1). Total 8 simulations have been performed. Out of these 8 simulations, for 2 simulations (Simulation No 1 and 5) data points are taken from LV model, whereas rest of the simulations consist of random data points generated in controlled manner.

is reached. This situation is applicable if it is possible to maintain the same VM number for a certain period of time and the incoming cloudlets requests do not fall below the expected number. Every data center might have a utilization threshold, beyond which it does not intend to stretch. It can be 80% of total VM utilization or can be any number based on their business model and other criteria. How stability can be attained in the proposed model, has been proved mathematically in Appendix A. **To keep the same number of VM and cloudlets what should be the α, γ parameter value of Lotka-Volterra model has also been elaborated in afore mentioned Appendix.**

The Lotka-Volterra model, which is plotted in the Fig. 6 is following:

$$\frac{\partial P}{\partial t} = 150P - PQ$$
$$\frac{\partial Q}{\partial t} = -80P + PQ$$

Where $\alpha = 150, \gamma = 80, \beta = 1, \delta = 1$
The condition which needs to be satisfied to reach stable situation is $\gamma Q = \alpha P$, where $\gamma = P$,$\alpha = P$.

C. Case 3: Prey decreases-Predator Increases

This scenario may arise when VM number reaches maximum available capacity and there is a need to allocate the VMs to incoming cloudlets to improve utilization. Such a situation, where data center needs to concentrate on provision of idle VMs, requires to decrease available VM (prey) and increase cloudlets number(Predator). Available VM number decreases as it is being provisioned to different incoming requests. Cloudlets number rises because more VMs are available and ready to serve incoming requests.

According to the algorithm, the condition $\alpha P > \gamma Q$ has to be met, where $\alpha > Q and \gamma < P$ The Lotka-Volterra model,

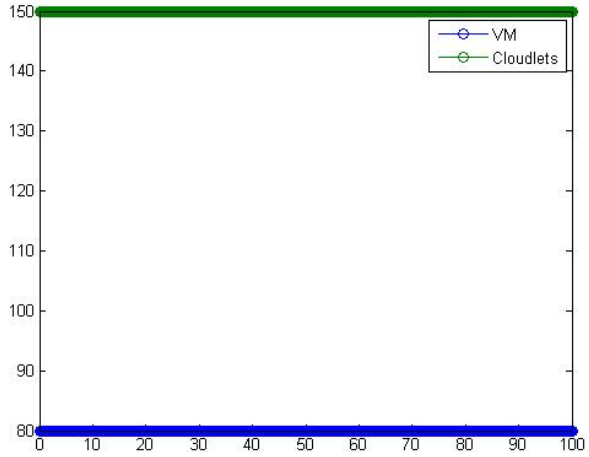

Figure 6: illustrates variation of cloudlets and VMs wrt time for stable situation (Case 2). The X axis represents the VM, cloudlets number, whereas the Y axis represents time span. Blue color depicts VM and green color represents cloudlets. It is observed from the figure that there is no change in the number of predator or prey throughout the time period (0-100). VM and cloudlets maintain the same initial values, which are 80, 150 respectively from time 0 to 100.

Time	VM	cloudlets
0	80	150
0.1	80	150
0.2	80	150
0.3	80	150
0.4	80	150
0.5	80	150
0.6	80	150
0.7	80	150
0.8	80	150
0.9	80	150
1.0	80	150
1.1	80	150
1.2	80	150
1.4	80	150
1.5	80	150
1.6	80	150
1.7	80	150
1.8	80	150
1.9	80	150
2.0	80	150

Table III. Predator Prey stability is the scenario (Case 2) where the same VM(prey) and cloudlets(Predator) numbers need to be maintained. The table III displays the predator, prey numbers at each time point, which are collected after 0.1 time interval. The table also supports the conclusion drawn from the fig 6 that there is no change in VM, cloudlets number as time passes from 0 to 100.

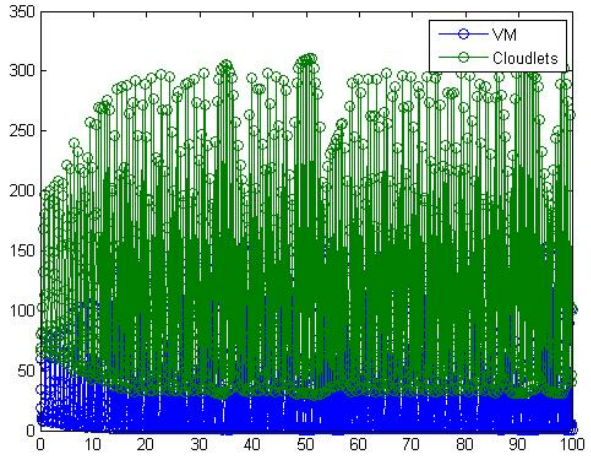

Figure 7: Variation of cloudlets and VMs wrt time, when prey number is required to decrease (Case 3). X axis represents Time-span and Y axis represents VM and cloudlets number. The values are spread out between 0 to 350. The blue region, belonging to VM occupies the lower part of the figure, whereas the upper region is covered by green, which signifies cloudlets. There is a significant region overlapped by VM and cloudlets but maximum places of the figure are free from overlapping. The maximum value in the figure, which is attained by the cloudlets, is near by the region 300. Minimum value, which is belongs to VM is near by 0.

which is plotted in the Fig. 7 is following:

$$\frac{\partial P}{\partial t} = 120P - PQ$$
$$\frac{\partial Q}{\partial t} = -30P + PQ$$

Where $\alpha = 120, \gamma = 30, \beta = 1, \delta = 1$

Total 8 simulations are conducted to calculate avg completion time for each batch. Out of these 8 simulations, 2 simulations are executed using datasets, which belong to Table IV. For comparison purpose, 6 simulations are done over random datasets but in a controlled manner. The Simulation 1 dataset is derived from the proposed model and avg Cloud request completions are calculated from Cloudsim. Simulation 2,3,4, are the random datasets to compare performance with simulation 1.

The difference of Simulation 2 with first simulation is that the 2nd and 3rd data point cloudlets number are greater than the first one.

The 2nd data point Cloudlet number of simulation 3 is higher but 3rd data point Cloudlet number is lower than the first simulation.

In simulation 4, 2nd data point Cloudlet number is lower whereas 3rd data point Cloudlet number is higher in comparison to the first simulation.

The simulation 5 dataset is derived from predator-prey model.

Time	VM	cloudlets
0	60	80
0.1	34.73	66.19
0.2	18.97	69.16
0.3	11.40	82.47
0.4	8.33	103.14
0.5	7.99	132.47
0.6	11.21	168.11
0.7	23.44	197.33
0.8	55.89	180.68
0.9	76.83	112.69
1.0	53.72	72.76
1.1	28.69	64.30
1.2	15.14	71.04
1.4	9.29	87.74
1.5	7.23	114.07
1.6	8.34	150.29
1.7	15.28	189.09
1.8	40.13	200.55
1.9	77.98	137.38
2.0	64.06	78.36

Table IV. This table captures the situation where the VM(Prey) needs to reduce but cloudlets(Predator) number is required to increase (Case 3). The table displays a few data points used to plot the figure. The initial VM, cloudlets values are 60,80. Except a few, all the VM values are less than initial VM value. In the case of cloudlets, there are a few occurrences, where cloulets values are less than initial value but maximum cloudlets values are higher than initial cloudlet value.

Simulation No	VM	cloudlets	Avg Request Completion time
1	60	80	440.25
1	11	82	2832.10
1	8	103	3165.24
2	60	80	435.95
2	11	182	5751.57
2	8	203	6505.15
3	60	80	432.45
3	11	182	3591.95
3	8	43	3636.95
4	60	80	425.1
4	11	52	2626.45
4	8	143	3310.23
5	60	80	445.90
5	55	180	1028.18
5	9	87	1074.31
6	60	80	485.1785
6	55	300	2672.91
6	9	245	2655.57
7	60	80	452
7	55	90	1071.68
7	9	175	1066.44
8	60	80	462.026
8	55	350	1570.62
8	9	15	1804.732

Table V. This is the simulation scenario, where it is required to increase cloudlets (Predator) and to reduce VM (Prey) number (Case 3). Simulation No 1 and 5 are derived from the LV model and data points for remaining simulations are randomly generated but in a controlled manner

In simulation 6 dataset, the 2nd and 3rd, both data points are higher than 5th simulation.

In simulation 7, the 3rd data point performance is better than the 5th simulation, which is derived from model. In simulation 7, the 2nd data point is lower whereas the 3rd data

point is higher in comparison to 5th simulation. In the case of simulation 8, 3rd data point cloudlets number is lower than 5th simulation and 2nd data point is higher than 5th simulation 2nd data point. The 1st and 5th simulation, where dataset derived from model is used, performed better than other random data sets. Only 1 instance, in simulation 4, data point 2 took lesser avg completion time in comparison to simulation 1, it has same VM number 11 but less number of cloudlets. Simulation 6,7,8 are random dataset in comparison to derived dataset from simulation 5. Hence, we can conclude that the dataset derived from model, performed better than random datasets, which follow no pattern.

The stable scenario is achieved, when $\alpha = Q, \gamma = P$ condition satisfies. Other two scenarios discussed above, can be achieved by fluctuating the α, β values from the stable situation. The difference between the αP and γQ determines the behavior of the data points in the table and the figure. Therefore, this model exhibits an advantage, where the constants for the predator and prey can be chosen to decide the expected behavior.

D. The modeling approach in CloudSim

CloudSim is a framework for modeling and simulation of Cloud computing infrastructures and services. Recently Cloud computing emerged as the leading technology for providing reliable, secure, sustainable and scalable computing services. There is already a wide ecosystem of Cloud environment, along with the increasing demand for energy-efficient IT technologies, demand timely, repeatable, and controllable methodologies for evaluation of algorithms, applications, and policies before actual development of Cloud products. Because utilization of real testbeds limit the experiments to the scale of the testbed and make the reproduction of results an extremely difficult undertaking, alternative approaches for testing and experimentation, leverage the development of new Cloud technologies. A suitable alternative is the utilization of simulations tools, which open the possibility of evaluating the hypothesis prior to software development in an environment where one can reproduce tests. Here, in this paper we use CloudSim to simulate in two different ways and compare the experimental results.

The initial approach is to allocate all the resources statically at the beginning of simulation. When the resources are allocated statically at the beginning of simulation, it results in over / under utilization and over / under provisioning of resources. Over-provisioning of resources occurs when the user requests gets surplus resources than demand. Under-provisioning of resources occurs when the user requests are assigned with fewer number of resources than the demand. Both over-provisioning and under-provisioning of resources result in poor optimization of resource allocation.

The next approach is to dynamically add the resources on-demand. Adding resources dynamically into the system avoids over / under provisioning of resources. Here the dynamic simulation model is compared with a biological model called Lotka-Volterra.

The resources on CloudSim compared with Lotka-Volterra model as:

- P is the number of Virtual Machines (Prey)
- Q is the number of cloudlets (Predators) where Cloudlet specifies the user request
- α is birth rate of Virtual machines in the absence of predation by cloudlets
- β is death rate of Virtual machines due to predation
- γ is natural death rate of cloudlets in the absence of Virtual Machines
- δ is reproducing rate of cloudlets

The simulation model is used to compute the parameters of Lotka-Volterra model. These parameters are used to control the system.

E. Resource Allocation algorithm using Predator Prey

Cloud computing provisions resources on the basis of demand. One of the major aspects of Cloud computing is that it allows to scale up and scale down resource allocation based on needs. Predator-Prey model, Lotka- Volterra, can be employed to understand the behavior of need based resource allocation. Cloud computing has been built upon virtualization and distributed computing to maximize resource utilization. Here, resources can be considered as prey and individual requests as predator. The objective is to establish that resources(VM) and requests(cloudlets) follow the Lotka-Volterra, Predator-Prey relationship.

Algorithm 1 Lotka-Volterra algorithm in Cloud Dynamics

```
1:  procedure LOTKA-VOLTERRA(p, q)  ▷ p is prey(VM), q is predator(Cloudlet)
2:      p ← VMs                                  ▷ Initialize VM
3:      q ← cloudlets                            ▷ Initialize cloudlets
4:      while VM = 0 do
5:          while (γ ≥ P)and(Q ≥ α) do
6:              γ ← γ + ϵ                        ▷ ϵ is infinite small number
7:              γQ ≥ αP
8:      while VM ← maxVMandcloudlets ≠ 0 do
9:          while (α > Q)and(γ < P) do
10:             α ← α + ϵ                        ▷ ϵ is infinite small number
11:             αP ≥ γQ
12:     return
```

Algorithm Explanation The algorithm starts with the initialization of prey(VM) and predator(Cloudlet). In a Cloud data center, if such situation occurs when no VM is available for allocation to newly arrived cloudlets, then the value of γ needs to increase in such a way that it satisfies $\gamma Q > \alpha P$ where $\gamma > P, \alpha < Q$. Therefore the VM number increases and cloudlets number decreases. If the VM number is near the maximum available VM and cloudlets are available then the value of α (weight of P) needs to increase so that $\alpha P > \gamma Q$ satisfies, where $\alpha > Q and \gamma < P$. Hence

Q resources (cloudlets) in the system will increase and P (available VM) will decrease. In that case, VM attains the maximum utilization level and needs to maintain the same VM and cloudlets numbers. $\gamma Q = \alpha P$ condition needs to be met, where $\gamma = P, \alpha = Q$. Considering all the scenarios $\beta, \delta = 1$.

IX. How is LV helping in achieving what was not accomplished before? :The Benefit & Cost

Ecological balance is one of the major areas of study for an ecologist. This model is important for the continued survival and existence of organisms without compromising the stability of the environment. As explained in [1], the systems are complex. The model describes a hierarchal structure of food chain and describes how every layer of predators have significant importance. Removal of any layer of predator challenges ecological stability by regulating the impacts of grazing. This ensures the overall productivity of the following layer of animals. Lotka-Volterra[2] is one of the most discussed model in food-chain system which describes the dynamics between any two corresponding layers of predator-prey relation. In service computing like Cloud computing, users can be considered as a predators. The user demands computing as a service and consumes the resources that Cloud provides. The Cloud resources are prey, which is consumed by the higher layer of food-chain, i.e users. The model proposed in paper [3] is based on the dynamic interaction between the predator and the prey. A multi-agent model was proposed to control the heterogeneous and volatile demand handling environment like Cloud. To address the volatility, some degree of autonomy is needed to enable the components to respond to dynamically changing circumstances. To address the above mentioned scenario, an elastic, autonomous and balancing model is needed to address the equilibrium under volatility. The proposed model addresses all the qualitative parameters with Lotka-Volterra model mimicking the ecological balance in ecology.

The paper uses LV(Lotka-Volterra) model to address more than one issue. These are Parameter tuning, Elasticity in VM, An improved Timeshared algorithm, Improvement in QoS, Reduction in SLA Violation, Predictive Analysis for VM allocation and Business model for the SMEs to tone down entry-barrier. The following section details about all these with explanation and experimental results.

X. Model Implementation and Outcome: Technical Discussion

A. Parameter Tuning

Lotka-Volterra model can give a different direction regarding resources provisioning and de-provisioning dynamically as per workload changes. Lotka-Volterra will be very efficient to predict the number of virtual machines based on the number of incoming Cloudlet requests and VM number when workload changes as per demand.

Algorithm Explanation

- Define maxThreshold and minThreshold of VMs and initialize VM pool.

Algorithm 2 Parameter Tuning

1: **procedure** LV-Parameter-Tuning ▷
2: $maxT \leftarrow MaximumThreshold$
3: $minT \leftarrow MinimumThreshold$
4: $VM \leftarrow Number-of-VMs-in-present-VM-pool$ ▷ Initialize VM
5: $T \leftarrow Time$
6: **while** $T \neq 0$ **do**
7: **while** $CPU - Utilization > maxT$ **do** ▷ Trigger LV, VM increasing cloudlets decreasing
8: $(new - VM > allocated - VM) and (new - VM < allocated - VM + VM$ ▷
9: $VM \leftarrow VM + additional - VM - in - pool$
10: **while** $CPU - Utilization < minT$ **do** ▷ Trigger LV, VM decreasing cloudlets increasing
11: $(new - VM < allocated - VM) and (new - VM > 0)$ ▷
12: $VM \leftarrow VM - LV - generated - number$
13: $T \leftarrow T - 1$
14: **return**

- Calculate VM utilization for every particular time interval.
- If CPU utilization $>$ maxThreshold.
- Trigger Lotka-Volterra (Prey Increasing and Predator decreasing situation). Lotka-volterra returns a set of VM number, cloudlets number.
- From this set select the particular VM number which is $>$ current allocated VM and $<$ current allocated VM $+$ VMs in pool.
- Add the additional VM from VM pool.
- if CPU utilization $<$ minThreshold
- Trigger LK (Prey Decreasing and Predator Increasing situation). select the VM number ¡currentOnlineVmnumber, and VM number > 0.
- Deallocate the VMs based on LK generated number and returned to the VM pool.

The parameter which is used as a criteria of provisioning and de-provisioning of VMs is utilization. Say a Cloud provider has decided to implement a monitoring algorithm, where the maxThreshold and minThreshold are defined as 80% and 20%. Every 30 seconds the algorithm will keep checking the VMs average utilization by using the formula given below [14].
VM utilization=

$$\frac{\sum_{I=1}^{currentcloudlets} CloudletLength(i) * CloudletPEs(i)}{PEs * MIPS} \tag{7}$$

VMs avg. Utilization=

$$\frac{\sum_{i}^{OnlineVms} VM_i Utilization}{OnlineVMs} \tag{8}$$

In 7 currentcloudlets is nothing but the number of cloudlets arriving for a particular VM. PEs is the number of processor in the VM. MIPS is the processing power of each processor core. CloudletLength is the number of instructions to be

run. CloudletPEs is the number of processors required by the Cloudlet request. In equation 8 OnlineVMs is the total number of VMs allocated for execution and $VM_iUtilization$ is the utilization of a single VM, calculated using 7. VMs avg utilization is compared every time with maxThreshold and minThreshold. If it violates the maxThreshold, it means VMs are over occupied and the number of VMs are not adequate to meet the spike of demand at that point of time. Hence, it is required to add more VMs from VM pool to serve all the incoming cloudlets requests without SLA violation and reducing the overhead on individual VM. But the question is how many VMs are needed to be pulled from the pool. Here the Lotka-Volterra algorithm plays its role by providing the VM number based on currently allocated VMs and total cloudlets number, executing in different VMs. In this particular scenario, Prey increasing and Predator decreasing condition is suitable as we need to increase online VM number. This monitoring algorithm never controls the incoming cloudlets number. If it does not satisfy the minThreshold criteria, it signifies that more than required VMs are allocated to the incoming cloudlets and VMs are under utilized. The number of VM needed to de-provision is rendered by Lotka-volterra (Prey Decreasing Predator Increasing situation) algorithm. The VMs, which are not executing any cloudlets are selected and returned back to VM pool.

B. Experiment

The monitoring algorithm is implemented in Cloudsim 3.03 version. There is a class named DataCenterBroker, which is responsible for VM creation, cloudlets submission to a particular VM, destroying the VM once it executed all the cloudlets submitted to it, etc. The DataCenterBroker class is the perfect place, from where it is feasible to monitor the VMs utilization and addition and de-provisioning of VMs based on the pre-decided threshold. Few decisions are taken prior to the experiment that the cloudlets are going to be submitted dynamically. The VM pool number will be predefined and monitor will keep checking the VMs average utilization for every 100 milliseconds. There is a CloudletScheduler, which decides the request to be allocated to a VM. In this experiment , time shared Cloudlet scheduler is used to serve the purpose. This particular scheduler allocates a time slot for every submitted Cloudlet request. On the other hand, VmAllocationPolicySimple is responsible for the determination of the host where a VM will be created. A single data center is utilized through out the experiments. The data center consists of two host machines with one host machine powered by four processors(quad core), and the other host machine contains two processors(dual-core machine). The computation speed for each processor is 1000 mips. Hence the datacenter contains two host machines, where one host machine is quad core and another host machine is dual core. We have submitted cloudlets and VMs in three phases. First phase submits predefined cloudlets and VMs before starting the simulation. Other phases add more VMs and cloudlets intermittently after the starting and before the completion of the simulation so that it can create a replica of a real time scenario. Apart form this, for every 100 milliseconds 10 cloudlets are submitted during the simulation to maintain the continuation of incoming requests. DormandPrince853Integrator of the commons-math library is used to extract values from Lotka-Volterra model. In the table VI, three parameters, Average Request Completion time, SLA violation rate and makespan time are displayed. Total three phases of submission are represented in the table and above mentioned parameters are calculated for every phase. Completion time is calculated as below

Exp No	VM	cloudlets	Avg Req Compln time	SLA violation	MakeSpan tim
1	10	98	514.1188	0.744	1697.94
1	15	135	686.6909	0.407	1446.94
1	16	155	692.715	0.419	1657.25
2	10	98	612.66	0.744	1648.94
2	15	135	336.72	0.29	1356.0
2	16	165	333.785	0.315	1633.01
3	10	98	550.51	0.755	1811.93
3	15	135	334.84	0.37	1450.48
3	16	165	312.19	0.32	1774.481
4	10	98	611.77	0.80	1630.61
4	15	135	317.35	0.31	1438.14
4	16	165	315.72	0.30	1636.961

Table VI. In the table, the performance of the Reactive scaling with LV modeling monitoring algorithm has been demonstrated. Three parameters Average Request Completion time, SLA violation rate and makespan time are displayed. Total three phases of submission are represented in the table for each experiment and above mentioned parameters are calculated for every phase. The experiments are conducted a total of 4 times and to maintain the uniformity, same VM, cloudlets numbers are used for every experiment.

$$Avg \quad Completion \quad time = \frac{\sum_{i=1}^{Cloudletnumber} Completion \quad time_i}{Number \quad of \quad Cloudlet} \tag{9}$$

For each phase the average completion time is calculated, whereas in equation 9 number of cloudlets signifies the number of cloudlets of each phase. Makespan is the total duration between the beginning and the end. Here the makespan is defined as the time difference from the last request finish time and the first request submission time of a phase.

$$makespan = finishing \quad time \quad of \quad last \quad request - submission \quad ti \tag{10}$$

Another parameter estimated along with makespan is SLA violation rate. If the completion time of any request exceeds the SLA mentioned expected completion time, then the request violates the SLA. In such a scenario, the Cloud service provider has to pay for the SLA violation. Hence in an ideal scenario, SLA violation rate should be minimum.

$$SLA \quad violation \quad rate = \frac{number \quad of \quad requests \quad violates \quad SLA}{total \quad number \quad of \quad requests} \tag{11}$$

C. VM Elasticity

Elasticity is a term, which is very common in the field of Physics and Economics but now-a-days the same term is also frequently used in Cloud Computing. In the context of Cloud Computing, elasticity is defined as an ability to adapt to the changing workloads by provisioning and de-provisioning Cloud resources automatically such that it can meet the current demand of resources at any point of time [16]. Elasticity is defined in physics as a property of material capturing the capability of returning to its original state after deformation. In economics, elasticity refers to the sensitivity of a dependent variable to the one or more arguments [15]. A brief and simple example will help to understand "How Elasticity plays its role in Cloud Computing". Consider a Website A, which is running in a certain data center and at that point of time t_0 as per the workload, 2 virtual machines are allocated. Due to the rising popularity of the website at time t_1, it has started receiving more requests and 2 virtual machines are not enough to serve all the requests. Hence, it needs to allocate more virtual machines. Consider now that 6 more virtual machines are required to cater to the changing workload. An elastic system should identify the situation and allocate 6 virtual machines immediately. Several hours later, at t_3, the number of user requests dropped significantly and 3 virtual machines are sufficient to handle all the incoming user requests. In such a scenario, an elastic system should detect the change in incoming requests and de-provision 5 virtual machines allocated earlier. Over-provisioning is a situation, where more resources are allocated than required. Such a situation needs to be avoided as service provider ends up spending more for the extra resources. Under-provisioning is a situation where lesser number of resources are allocated for the service provider and it has serious impact on the performance of the service. It may often lead to violation of service level agreement and service provider looses customers due to poor services, which will have direct impact on the profit.

Reactive Scaling

The methods used for resolving scaling decisions can be classified into two categories. One is Reactive methods and the other is Proactive methods. Reactive resource allocation is usually a method based on rules (or threshold) which determines limitations for violating a series of rules and when these violations occur, it carries out measures related to resource scaling. Though, there are three main concerns with this method

- When rule violation incident happens, the scaling decision may involve SLA violation, which affects QoS.
- It may also happen that scaling of decisions of resources due to some violations is not necessary, as the violations are temporary. Scaling up and down of resources is not required.
- The on-demand VM requires a certain amount of time to initialize, boot-up and start the applications. Therefore, new request for additional VM may fail as the VM was not ready within the required time-frame.

In this experiment, the monitoring algorithm analyzes the VMs utilization for every equal interval of time. In case it encounters a violation, it can take the right step to mitigate the situation. If the average utilization violates the maximum threshold then it is going to add a VM from the VMs pool, whereas the violation of minimum threshold will eradicate a VM and return it to the VM pool. As indicated in the tables, the same number of sets are used across, and the experiment has been repeated four times, which has produced results with small variations. Like Lotka-Volterra monitoring algorithm, the same parameters (Average Request Completion time, SLA violation, Makespan time) are explored in Reactive scaling algorithm.

Exp No	VM	cloudlets	Avg Req Compln time	SLA violation	MakeSpan tim
1	10	98	814.08	0.89	1902.4
1	15	135	339.068	0.34	1432.05
1	16	155	466.15	0.49	1902.91
2	10	98	686.63	0.82	1726.91
2	15	135	371.87	0.41	1591.05
2	16	155	384.457	0.37	1724.37
3	10	98	788.20	0.89	1826.92
3	15	135	297.29	0.207	1315.98
3	16	155	400.55	0.361	1811.34
4	10	98	786.21	0.877	1867.93
4	15	135	341.06	0.325	1509.96
4	16	155	435.70	0.451	1861.81

Table VII. The table showcases the performance of the Reactive scaling algorithm without LV model. Total four experiments are displayed and each experiment comprises of three phases. Three parameters: avg request completion time, SLA violation rate and makespan are evaluated for every experiment.

Now, if we compare the result of Lotka-Volterra with Reactive scaling algorithm, it is evident from the tables that except for a few instances, Lotka-Volterra algorithm has outperformed the Reactive scaling algorithm with respect to all the three parameters.

D. Proactive Scaling

In this section, we are going to discuss and compare the performance of Proactive scaling after integration with Lotka-Volterra model. Unlike reactive scaling, the QoS parameter that has been considered for threshold calculation is response time. In this experiment, the monitoring algorithm predicts the response time in future for equal time intervals. If the monitoring algorithm identifies any SLA violation of response time, an upscaling/downscaling decision has to be made. Response time threshold is predefined and decided when the SLA is made between Cloud provider and user. Proactive scaling requires two types of thresholds. One is upper threshold and the other is lower threshold. If any situation arises where upper threshold is violated by future response time prediction, new VMs have to be added to service to neutralize the situation. In the case of lower threshold violation, VMs need to be deallocated from the user service as more than required number of VMs are allocated to increase utilization of the resources. We have compared the performances between the two cases. In one case, Lotka-Volterra model is employed to calculate the number of VMs that need to be added or removed based on threshold violation. In any case, a VM is allocated/

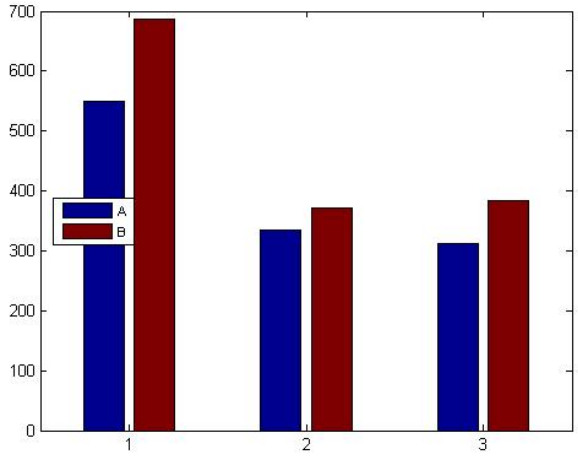

Figure 8: The Y axis represents the average completion time. 1, 2 and 3 of X axis denotes the first, second and third phase respectively. A in the figure depicts the Reactive scaling with Lotka-Volterra avg. completion time and B signifies the reactive scaling without LV avg. completion time. It is visible that A has performed better than B in all the three phases. The dataset of two tables, exp no 2 of table VI and exp no 2 of VII are used in the figure, which proves the superiority of Lotka-Volterra over the other algorithm.

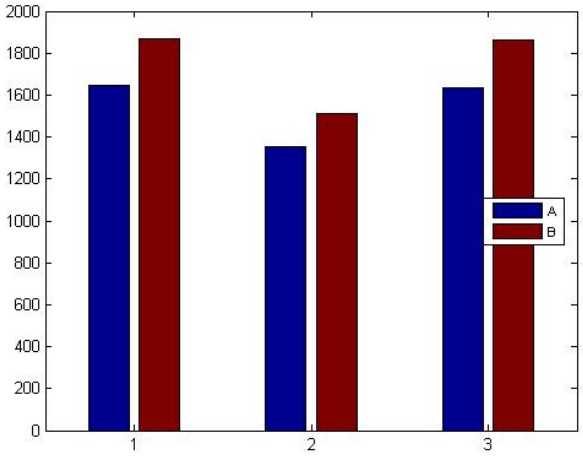

Figure 9: depicts the makespan comparison of the two algorithms. Like the previous figure, A and B are representing Lotka-Volterra and Reactive scaling respectively. The makespan time is shown alongside the Y axis whereas the X axis shows the three phases. The first phase consists of 98 cloudlets. 135 cloudlets are part of second phase whereas third phase consists of 165 cloudlets. A has outperformed B in makespan duration as th e makespan duration of A is lesser than B. Table VI, exp no 2 and Table VII, exp no 4 are illustrated in the makespan comparison figure.

deallocated based on SLA violation prediction. The method, which is used for the prediction of future response time is

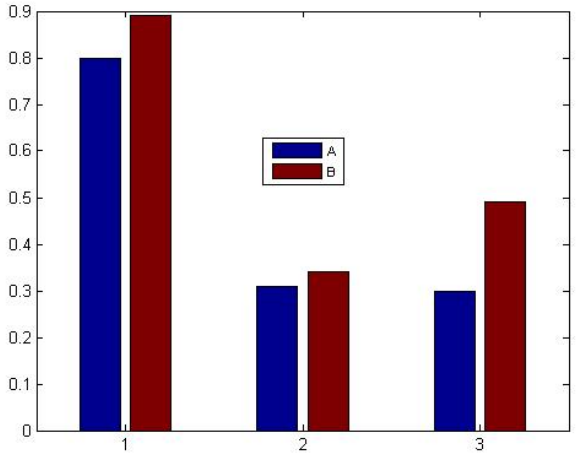

Figure 10: Illustrates the SLA violation comparison between the two algorithms. Table VI, exp no 4 and Table VII, exp no 4 are discussed in the SLA comparison figure. Deadline considered for each phase is 400 milliseconds. A request would violate the SLA, if its completion time exceeds the predefined time, which is 400 milliseconds here. Each phase SLA violation numbers of both the algorithms have been displayed side by side. A depicts the Lotka-Volterra algorithm and B signifies Reactive scaling algorithm. It is evident from the figure that SLA violation rate is lesser in the case of A than in B.

WMA (Weighted Moving Average). It is widely used and very familiar in stock market strategy. It has multiplying factors to give different weights to data at different positions [13].

$$WMA(t) = \frac{n * data_{t-1} + (n-1) * data_{t-2} + .. + 2 * data_{t-n+2} + da}{n + (n-1) + (n-2) + .. + 2 + 1} \quad (12)$$

E. Predator-Prey cloudlets Scheduling Timeshared Algorithm

In this section, a Cloudlet time sharing algorithm is explored in a new dimension. We have integrated the Lotka-Volterra model with the existing Cloudlet time sharing algorithm of Cloudsim and have taken advantage of the predator-prey equation. The algorithm decides the VM occupancy before submitting the incoming Cloudlet request to VM. If the occupancy or the VM utilization is more than the predefined threshold, Lotka-Volterra (Prey increasing-Predator deceasing) model is invoked to retrieve VM number, which is required to reduce the utilization within threshold. Here, prey denotes the number of available VM and predator the number of cloudlets. Till the time the VM number has not reached the Lotka-Volterra suggested number, all the incoming requests are pushed into a waiting list to the corresponding VM. Once the normal situation attains, that is the utilization drops below threshold, the requests from waiting list are sent for execution. In case the utilization of VM goes down the minimum threshold and incoming requests are available then more cloudlets

No	SLA VLTN WO LK	SLA VLTMN W LK	ExETN time
1st Scenario	.458	.43	1711.9
2nd Scenario	.535	.45	529.7

Table VIII. Lotka-Volterra Proactive algorithm comparison with Proactive scaling without LV has been illustrated in table VIII. Total 630 cloudlets are used during 1st scenario simulation, where 550 cloudlets are fed into the datacenter into three phases. 1st phase has pushed 98 cloudlets, 2nd phase consists of 302 cloudlets and third phase inserted 150 cloudlets. Rest of the 80 cloudlets are generated during simulation time. The SLA deadline for all the cloudlets execution time, has been decided as 2000 milliseconds. The total number of VM's allocated initially is 27 and 100 Vms in pool, which will be used in elasticity algorithm for further allocation based on situation. Upper threshold for scaling decision has been set at 400 ms and lower threshold is 100 ms. 2nd scenario consists of total 394 cloudlets as an input to the dataceneter, where all the cloudlets are pushed into three batches. First batch consists of 12 cloudlets, second batch consists of 102 cloudlets and third batch pushed 200 cloudlets. 80 cloudlets are generated during simulation time. Total number of VM in VM pool is 100. SLA violation has been set at 500 milliseconds for execution time quality of parameter. Like scenario 1, 27 VMs are allocated initially. Both the scenarios are applied on predictive scaling with Lotka-Volterra model and without Lotka-Volterra model. It is evident from the table that proactive scaling with LV has outperformed its counter part in all three sections.

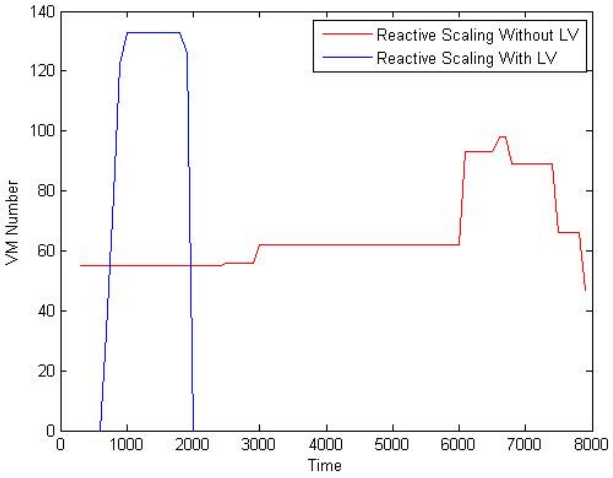

Figure 11: This figure depicts the comparison of reactive scaling algorithm with and without LV model. Y axis represents the VM number and X axis represents the time. It is apparent from the figure that the simulation of reactive algorithm with LV lasts for shortest period of time, till 2000 milliseconds. On the other hand, the reactive algorithm without LV goes on till 8000 milliseconds. But LV model uses maximum number of VMs to complete all the cloudlets task.

requests are submitted to VMs for processing as per Lotka-volterra model(Prey decreasing-Predator increasing situation). This particular situation is applicable when incoming requests

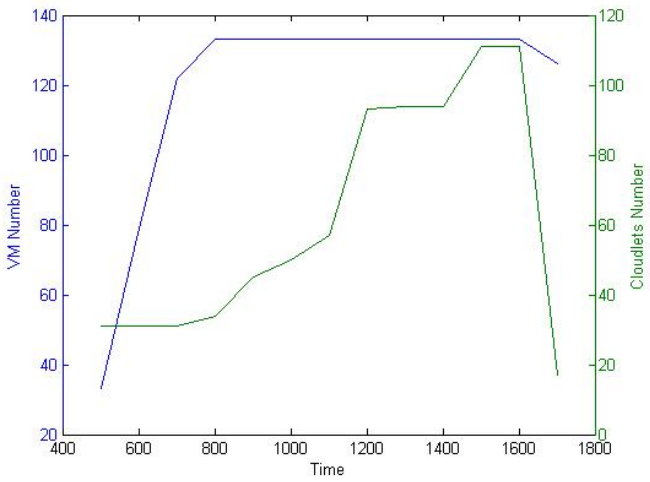

Figure 12: The figure illustrates the VM number and Cloudlet number against time for reactive scaling algorithm with LV model. X axis plots the Time and Y axis left side and right side represent the VM number and Cloudlet number respectively. The figure demonstrates the behavior of the VM and cloudlets as simulation progresses. The blue curve denotes the VM number whereas the green curve showcases the Cloudlet number. Initially, both the curves rise but VM number reaches it's threshold limit, hence the curve becomes flat afterwards. But VM number curve starts declining as the Cloudlet number starts falling.

are available. The formula used to identify the VMs utilization is below:

$$vm\ \ utilization = \frac{total\ \ vm\ \ executing\ \ the\ \ requests\ \ at\ \ that\ \ moment}{total\ \ number\ \ of\ \ available\ \ VMs} \tag{13}$$

Algorithm 3 LV-Timeshared Algorithm

procedure LV-TIMESHARED ▷
  $scloudlets \leftarrow cloudlets - submitted - for - execution$
  $cloudlets_Q \leftarrow existing - cloudlets - in - queuee$
  $T \leftarrow Time - Allocated$
  $oVM \leftarrow occupancy - of - VMs$
  while $T \neq 0$ **do**
    call procedure $LV - PARAMETER - TUNING$
    if $oVMs > MaxThreshold$ **then**
      ▷ VM increasing, cloudlets decreasing
      while $oVM > available - VM$ **do**
        add $cloudlets_Q \leftarrow scloudlets + cloudlets_Q$
    if $oVMs < MinThreshold$ **then**
      ▷ VM decreasing, cloudlets increasing
      add $cloudlets_Q \leftarrow scloudlets + cloudlets_Q$
    $T \leftarrow T - 1$
  return

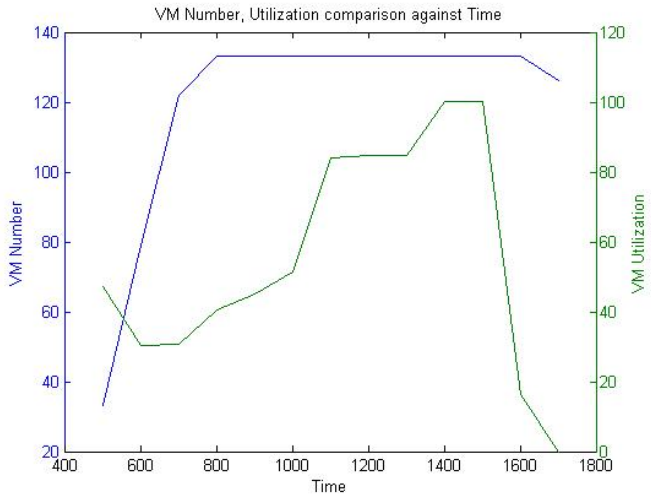

Figure 13: The figure depicts the VM number, VM utilization against time. X axis denotes the time. Y axis left side represents the VM number and Y axis right side represents the VM utilization. In the first phase the utilization starts falling as more VMs are being allocated due to the rising number of cloudlets. But the VM utilization rises up as many more cloudlets start arriving and VM number reaches its threshold level. At the last lag, both the curves(VM number, VM utilization) start declining as cloudlets arriving rate reduces. Blue, green curve depict the VM number, VM utilization respectively.

Predator-Prey cloudlets Scheduling Timeshared Algorithm

- cloudlets submitted for execution.
- Calculate occupancy of VMs.
- If(occupancy of $VMs > thresholds$) [very less number of VMs available]
- call Lotka-Volterra for Prey Increasing-Predator Decreasing [VM number will increase and Cloudlet will decrease]
- add cloudlets to waiting queue till it drops to LV suggested VM occupancy/surges till LV suggested available VMs.
- Once the LV suggested VM number reaches, start submitting the cloudlets from waiting queue.
- If the occupancy drops below the required level(Many VMS are available)
- Call Lotka-Volterra for Prey decreasing-Predator Increasing (VM occupancy will increase/available VM will reduce and Cloudlet will increase)
- Submit cloudlets without keeping them in waiting queue till it ramps up to the LV suggested number.(This situation will work in case incoming cloudlets are available)

F. Simulation of Timeshared Algorithm

Two parameters have been considered for the experiment purpose. These are average Cloudlet completion time and SLA violation rate. The Lotka-Volterra time shared algorithm has been compared with the existing Cloudlet time sharing algorithm of Cloudsim and it is shown how performance of the algorithm is improved.

VM	cloudlets	LV Time Sharing avg execution time	Cloudlet Time sharing Scheduling avg execution time	Deadline	LV Cloudlet time sharing SLA violation	Cloudlet time sharing SLA violation
60	80	437.97	472.84	450	0.2	0.3
55	180	873.007	1069.92	1000	0.37	0.65
9	87	876.26	1066.63	1000	0.45	.70

Table IX. **This table comprises of three batch executions of VM and cloudlets. Each row represents one batch execution. The comparison between Time shared scheduling algorithm of Cloudsim and Time shared scheduling algorithm with LV has been demonstrated. Two parameters, SLA violation rate and avg execution time are highlighted in the table. It is evident from the table data that time shared scheduling algorithm with LV has outperformed the other time shared algorithm.**

Experiment 1

In this experiment, the VM and cloudlets are submitted in three batches. The table explains all the batches and the corresponding VM and cloudlets numbers of each batch. Deadline is a predefined number for each Cloudlet request before the execution starts. If the Cloudlet execution time surpasses the deadline, it is considered as SLA violation. For 1st, 2nd and 3rd batch, the deadlines are 450, 1000 and 1000 milliseconds.

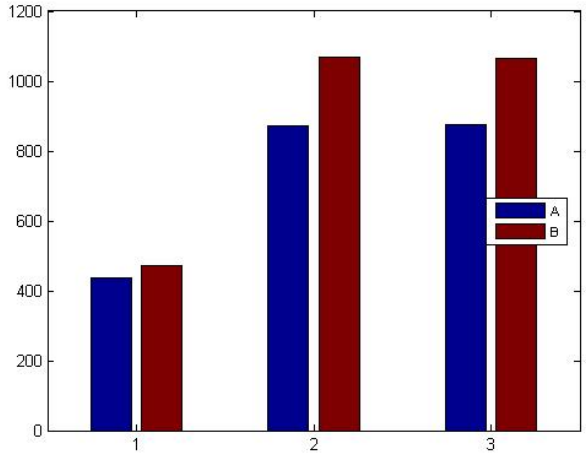

Figure 14: **Average Completion time Comparison between time shared scheduling algorithm with LV and without LV. A denotes the time shared scheduling algorithm with LV and B represents the time shared scheduling algorithm without LV. For all the batches, the avg completion time is better for time shared LV algorithm. Y axis represents the avg completion time and X axis the batches.**

Experiment 2

In the second experiment VM, Cloudlet numbers for each batch have changed. The deadlines for execution time are set

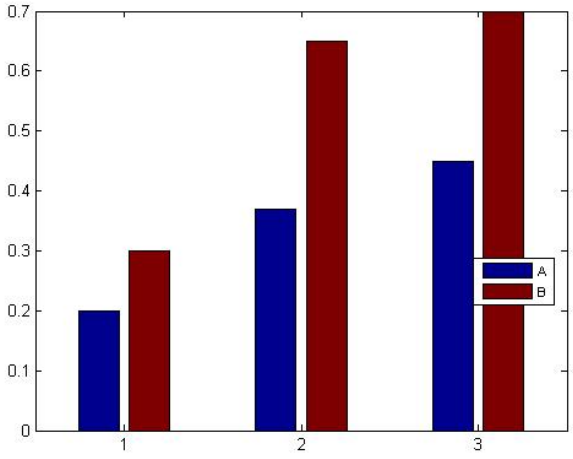

Figure 15: **SLA Comparison between time shared scheduling algorithm with LV and without LV. A, B represent time shared scheduling algorithm with LV and time shared scheduling algorithm without LV, respectively. SLA violation rate is less in the case of LV time shared algorithm.**

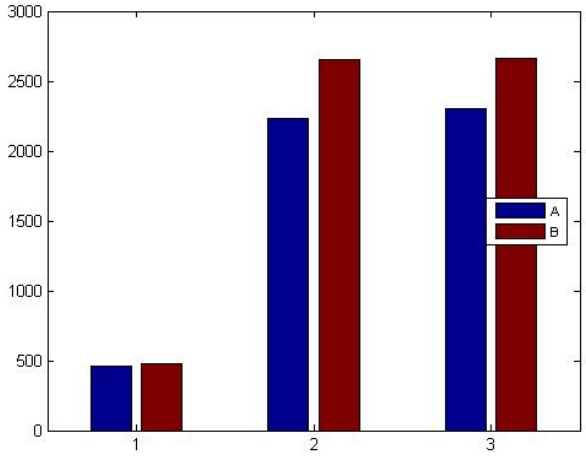

Figure 16: **Average Completion time Comparison between time shared scheduling algorithm with LV and without LV. A denotes the time shared scheduling algorithm with LV and B represents the time shared scheduling algorithm without LV. For all the batches, the avg completion time is better for time shared LV algorithm. Y axis represents the avg completion time and X axis the batches.**

VM	cloudlets	LV Time Sharing avg execution time	Cloudlet Time sharing Scheduling avg execution time	Deadline	LV Cloudlet time sharing SLA violation
60	80	459.82	478.45	450	0.225
55	300	2229.98	2654.66	3000	0.096
9	245	2300.13	2659.14	3000	0.11

Table X. **This table comprises of three batch executions of VM, cloudlets. Each row represents one batch execution. The comparison between Time shared scheduling algorithm of Cloudsim and Time shared scheduling algorithm with LV has been demonstrated. Two parameters SLA violation rate and avg execution time are highlighted in the table. It is evident from the table data that time shared scheduling algorithm with LV has outperformed the other time shared algorithm.**

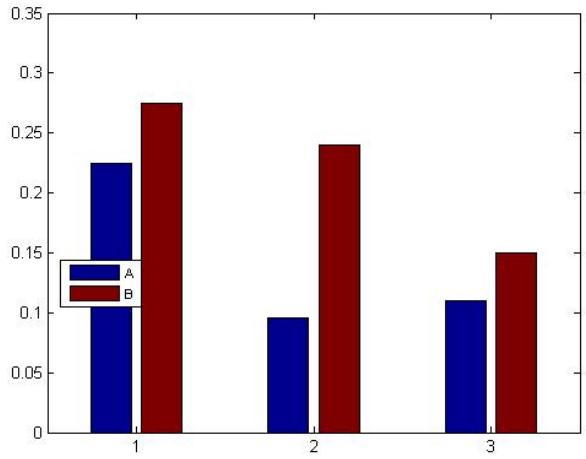

Figure 17: **SLA Comparison between time shared scheduling algorithm with LV and without LV. A, B represent time shared scheduling algorithm with LV and time shared scheduling algorithm without LV. SLA violation rate is less in the case of LV time shared algorithm.**

at 450, 3000, 3000 milliseconds for 1st, 2nd and 3rd batch.

Experiment 3

In this experiment different VM, cloudlets numbers are considered for second and third batch, though the first batch VM, cloudlets number are uniform across the experiments. Deadline for each batch is similar to the experiment 1.

G. Improved Quality of Service & Reduction in SLA Violation

- Elasticity: It is an ability of a a Cloud data center to provision and de-provision VM as per the dynamic behavior of the Cloud resource demand. Lotka-Volterra provides the flexibility to decide the required number of VMs needed to be introduced into the systems based on the number of current Cloud resource requests. VM Elasticity is discussed in detail in "Details of Novelity in Technical Contribution" section, under subsection C.
- Make span : Makespan is an another metric, used in this paper to measure the performance improvement of various algorithms such as reactive scaling, proactive scaling, Cloudlet time shared algorithm, etc, after the introduction of the Lotka-Volterra model into the afore mentioned algorithms. Reference of makespan can be observed across the paper such as subsection Reactive

VM	cloudlets	LV Time Sharing avg execution time	Cloudlet Time sharing Scheduling avg execution time	Deadline	LV Cloudlet time sharing SLA violation
60	80	462.35	463.43	450	0.375
55	90	928.47	1066.84	1000	0.5
9	175	925.68	1121.98	1000	0.53

Table XI. **This table comprises of three batch execution of VM and cloudlets. Each row represents one batch execution. The comparison between Time shared scheduling algorithm of Cloudsim and Time shared scheduling algorithm with LV has been demonstrated. Two parameters SLA violation rate and avg execution time are highlighted in the table. It is evident from the table data that time shared scheduling algorithm with LV has outperformed the other time shared algorithm.**

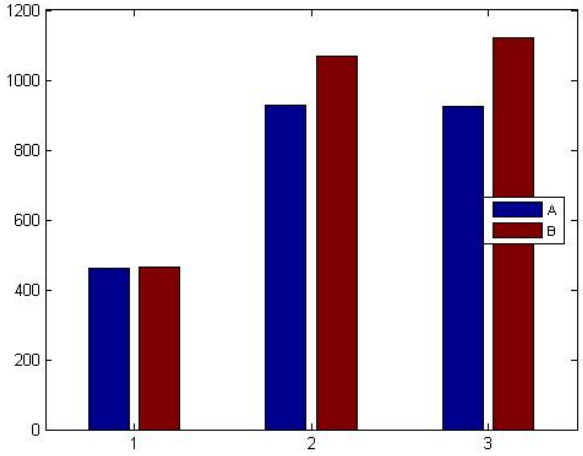

Figure 18: **Average Completion time Comparison between time shared scheduling algorithm with LV and without LV. A denotes the time shared scheduling algorithm with LV and B represents the time shared scheduling algorithm without LV. For all the batches, the avg completion time is better for time shared LV algorithm. Y axis represents the avg completion time and X axis the batches.**

Scaling, Proactive Scaling under "Details of Novelity in Technical Contribution" section.

- Response time : It is a widely used QoS parameter in Cloud computing. We have shown that the implementation of Lotka-Volterra model has improved the response time performance metric significantly. Multiple occurrences of this particular QoS can be observed in various sections of this paper such as "Details of Novelity in Technical Contribution".
- Utilization : Utilization has been used heavily in this paper. In the algorithms such as LV-Timeshared algorithm, the invocation of Lotka-Volterra has occurred whenever the utilization is touching the maximum/ minimum pre-

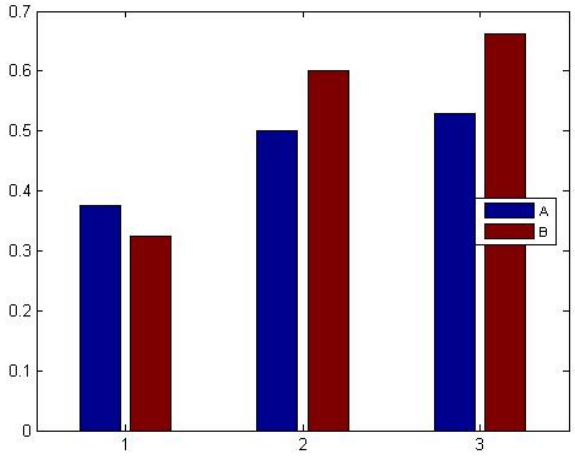

Figure 19: **SLA Comparison between time shared scheduling algorithm with LV and without LV. A, B represent time shared scheduling algorithm with LV and time shared scheduling algorithm without LV, respectively. SLA violation rate is improved after introducing LV model into time shared scheduling algorithm.**

defined threshold.

- Reduction in SLA violation: SLA violation is the most important performance metric of a Cloud datacenter. Revenue of a Cloud service provider is tightly coupled with SLA violation rate. After the introduction of LV model into various algorithms, SLA violation rate has been reduced, which has improved the quality of service and customer satisfaction. More details can be found in "Details of Novelity in Technical Contribution" section.

H. Cubic interpolation of VM number

Cubic spline interpolation divides the entire approximate interval to a set of sub-intervals and interpolates using a different polynomial for each of them. The cubic spline function *C(x)* for the tabular data, (x_1,y_1), (x_2,y_2),... (x_n,y_n) is represented using the following equation.

$$C(x) = p_0 + p_1 * x + p_2 * x^2 + p_3 * x^3 \tag{14}$$

Where, $p_0, p_1, p_2,$ and p_3 are constants. The cubic spline *C(x)*, has the following properties.

1) *C(x)* is composed of cubic polynomial pieces $C_k(x)$ *C(x)* = $C_k(x)$, if $x \in [x_k, x_{k+1}]$, k = 1,2,..n-1.
2) $C(x_k)= y_k$, k = 1,..n. (interpolation)
3) $C_{k-1}(x_k)= C_k(x_k)$, k = 2,..n-1.
4) $C'_{k-1}(x_k)= C'_k(x_k)$, k = 2,..n-1.
5) $C''_{k-1}(x_k)= C''_k(x_k)$, k = 2,..n-1.

The cubic spline interpolation can be a subject of future work where it may pave new directions along with Lotka-Volterra to address different aspects such as elasticity, scheduling algorithm etc. in cloud data center paradigm.

XI. Business Model

New generation computing is enabled by the very large array of economic data storage and processing renamed as 'Cloud Computing'. It is a kind of Internet-based computing that provides shared processing resources and data to computers and other devices on demand. The reasons of its huge popularity is mainly because the low and medium income category users have received the highest benefit. Report says that [5] developing economics has a strong impact on emergence of Cloud infrastructure. It strengthens Small and Medium level Enterpriser(SME) and employment in all economics. Dynamic growth of Cloud resources and un-committed capital are the major reason for these SMEs to opt Cloud services. This signifies more number of independent small scale businesses, more employment with an indirect impact to strengthen economy. Recently, global Cloud Infrastructure addressed a few issues for the benefit of customers and Cloud service providers. These issues highlight some major technological aspects. These include availability of high speed communication service provisioning, information flow without restriction among service providers and consumers, and also significant economics of scale and scope for the Cloud data center which truly addresses the dynamic scalability as per customer needs. [1] The report estimates the situations of developing countries like India, China, Mexico and South Africa. The estimation says that Indian Cloud market would reach $ 15 to $18 billion by year 2020. A large number of IT and ITES would run in Cloud with huge potential for economic growth and employment with a further deep assessment in education, infrastructure and the distribution of economic opportunity. Major educational embodies like NKN,UGC,AICTE are coming together to deploy higher educational institute, projects like 'meta-university' and development of programs like 'Joint-virtual degree' on the backbone of Cloud infrastructure in much cheaper investment. Cloud based network also created an impact on shift of market structure with a potential to shift more control and profit on other segments like small farmers, traders, Regional Rural Bank, rural shores and Desi crews in remote areas. The discussion on the report implies the growth and business in Cloud has huge potential market in various sectors in the developing countries which may be targets for potential Cloud providers.

At present there are many Cloud providers in the market. These service providers offer services to their customer as per a formal business agreement named as SLA i.e Service Level Agreement. As per [6] all these SLAs define the committed availability as a service commitment. However, these providers are not very specific about the steps for violation of this commitment. The major reason is ambiguous literal expression. Variability in availability and corresponding penalty and lack of clarity offered in SLA by various providers is one of the major focus in this paper. The paper proposes how the penalty should be calculated and how violation in SLA would impact profitability in both the parties, i.e provider and customer. The next focus in this paper is about the potential small players(SME) who want to target the Cloud market as Cloud service providers. The present market is fully occupied by large and very large international players. The report [6] details about the entry-deterrence strategies for the SMEs. The report analyses the strategic use of entry-deterrence of established firms and entrants quality choice in a vertically differentiated products market. The report considers the level of entrants fixed cost, the degree of consumers quality taste to fix the equilibrium point in entry-deterrence or entry-accommodation.

Consider, an enterprise has decided to be a part of Cloud business. As per several references, there are two types of costs associated with it, there are Infrastructure and Operation & Maintenance. The new joiner organizations need to invest heavily in order to set up their own Infrastructure and various domains of operation and maintenance. This involves certain trade-offs with regard to different components in Infrastructure costs, initial operational as well as recurrence maintenance. This article intends to optimize the cost structure for business models with the help of a quasi Cobb-Douglas production function. We will start with a brief description of Cobb-Douglas production function.

$$Y = PL^{\alpha}K^{\beta} \tag{15}$$

Where Y= total production output
L=Labor input
K=Capital input
P=Total factor productivity (Rephrase in terms of VM/Cloudlet
As discussed above, the two widely used mathematical models namely Harrod and Solow neutral progress to predict the technological progress can be integrated with equation (15) to accommodate time variant technological changes.

$$Y = P[A_i(t)L]^{\alpha}K^{\beta} \tag{16}$$

$$Y = PL^{\alpha}[B_i(t)K]^{\beta} \tag{17}$$

Combining equations (16) and (17) .

$$Y = P[A_i(t)L]^{\alpha}[B_i(t)K]^{\beta} \tag{18}$$

$$Y = P[AL]^{\alpha}[BK]^{\beta} \tag{19}$$

We have assumed technological progress A and B as endogenous variables, hence dependent on other parameters related with R&D. Therefore, Harrod neutral technological progress A and Solow neutral technological progress B may be represented as follows:

$$A = rL^{*\beta_1}\Gamma^{1-\beta_1} \tag{20}$$

where r is the future discount rate.
L^* is the labor involved in R&D related to Harrod technological progress.
Γ is the capital invested for R&D

$$B = rK^{*\alpha_1}\Delta^{1-\beta_1} \tag{21}$$

where r is the future discount rate as usual.
K^* is the capital invested in R&D related to Harrod technological progress whereas Δ is the labor contribution to R&D.

[1]The stochastic model is available as a working paper on arXiv.org "CDSFA Stochastic Frontier Analysis Approach to Revenue Modeling in Large Cloud Data Centers"

A. Revenue Maximization

Consider an enterprise that has to choose its consumption bundle (I,M) where I, M are infrastructure and Operation & Maintenance costs, respectively of a newly joined enterpriser. The assumption taken is that the enterprise does not want to exceed a tentative budget but wants to maximize its service. The service maximization is done using Lagrangian Multiplier. The quasi Cobb-Douglas function can be formulated as:

$$f(I,M) = [AM]^{\alpha}[BI]^{\beta} \tag{22}$$

Let m be the cost of the inputs that should not be exceeded.

$$w_1AM + w_2BI = m \tag{23}$$

w_1 is the unit cost of augmented recurring cost.
w_2 is the unit cost of augmented infrastructure cost. Optimization problem for production maximization is written as:
max $f(I,M)$ subject to m The following values of A,B obtained are the values for which the data center has maximum production after satisfying the constraints on the investment.

$$A = \frac{\alpha m}{w_1 M(\alpha+\beta)} \tag{24}$$

$$B = \frac{\beta m}{w_2 I(\alpha+\beta)} \tag{25}$$

[2] Replacing the values of A and B by using equations (20) and (21).

$$L^* = \frac{m\alpha}{rw_1M\left(\alpha+\beta\right)\Gamma^{1-\beta_1}}^{\frac{1}{\beta_1}} K^* = \frac{m\beta}{rw_2I\left(\alpha+\beta\right)\Delta^{1-\alpha_1}}^{\frac{1}{\alpha_1}} \tag{26}$$

These results are proved in Appendix A.

B. Cost Minimization

Consider an enterprise with a target level of output to be achieved by investing a minimum amount. The quasi Cobb-Douglas function is of the form:
$f = [AM]^{\alpha}[BI]^{\beta}$ where y_{tar} is the target output of the firm that needs to be achieved and w1,w2 are unit prices of Recurring cost, and infrastructure respectively. Cost minimization problem is formulated as follows:

$min_{A,B} w_1AM + w_2BI$ subject to y_{tar}
The cost of producing y_{tar} units in cheapest way is c, where

$$c = w_1AM + w_2BI \tag{27}$$

c can be written as follows:

$$c = \frac{w_1 y_{tar}^{\frac{1}{\alpha+\beta}}}{\frac{\beta w_1}{\alpha w_2 I}^{\frac{\beta}{\alpha+\beta}}} + \frac{w_2 y_{tar}^{\frac{1}{\alpha+\beta}}}{\frac{\alpha w_2}{\beta w_1 M}^{\frac{\alpha}{\alpha+\beta}}} \tag{28}$$

The details of the above results have been elaborated in Appendix B.

[2]The stochastic model is available as a working paper on arXiv.org "CDSFA Stochastic Frontier Analysis Approach to Revenue Modeling in Large Cloud Data Centers"

C. Profit Maximization

Consider an enterprise that needs to maximize its profit. The Profit function is:

$$F = f(A,B) - w_1AM - w_2BI \tag{29}$$

Profit maximization is achieved when :
$\frac{\partial f}{\partial A} = w_1M \quad and \quad \frac{\partial f}{\partial B} = w_2I$
The values are obtained after the the calculations as below:

$$A = \frac{w_1^{\frac{1}{\alpha+\beta-1}}}{M\alpha^{\frac{1-\beta}{\alpha+\beta-1}}\left(\frac{w_1\beta}{w_2}\right)^{\frac{\beta}{\alpha+\beta-1}}} \tag{30}$$

$$B = \frac{w_2^{\frac{1}{\alpha+\beta-1}}}{I\beta^{\frac{1-\alpha}{\alpha+\beta-1}}\left(\frac{w_2\alpha}{w_1}\right)^{\frac{\alpha}{\alpha+\beta-1}}} \tag{31}$$

The above results are proved in Appendix C. Substituting the values in Equation (46) we obtain

$$Y = P\frac{w_1^{\frac{1}{\alpha+\beta-1}}}{\alpha^{\frac{1-\beta}{\alpha+\beta-1}}\left(\frac{w_1\beta}{w_2}\right)^{\frac{\beta}{\alpha+\beta-1}}}^{\alpha} \frac{w_2^{\frac{1}{\alpha+\beta-1}}}{\beta^{\frac{1-\alpha}{\alpha+\beta-1}}\left(\frac{w_2\alpha}{w_1}\right)^{\frac{\alpha}{\alpha+\beta-1}}}^{\beta} \tag{32}$$

Replacing the values of A and B by equations (20) and (21), the following results are derived.

$$L^* = \frac{w_1^{\frac{1}{\alpha+\beta-1}}}{rM\alpha^{\frac{1-\beta}{\alpha+\beta-1}}\left(\frac{w_1\beta}{w_2}\right)^{\frac{\beta}{\alpha+\beta-1}}\Gamma^{1-\beta_1}}^{\frac{1}{\beta_1}} K^* = \frac{w_2^{\frac{1}{\alpha+\beta-1}}}{rI\beta^{\frac{1-\alpha}{\alpha+\beta-1}}\left(\frac{w_2\alpha}{w_1}\right)^{\frac{\alpha}{\alpha+\beta-1}}\Delta} \tag{33}$$

The output revenue, in case of profit maximization is independent of Infrastructure cost and Maintenance investment.

D. Stochastic frontier

The Service frontier can be written as:

$$y = f(K,L)TE \tag{34}$$

where TE is the technical inefficiency, the ratio of observed output to maximum possible output. If TE=1, the organization achieves maximum output. This production frontier is deterministic, as the entire deviation from maximum feasible output is attributed to technical inefficiency. It does not consider random shocks, which is not beyond the control of production function. To address the random shocks, the production frontier function can be redefined as below:

$$y = f(K,L)TEexp(v) \tag{35}$$

where v is the stochastic variable which defines the shocks, uncertainty, luck etc. Let us consider the linear logarithmic form of stochastic frontier service delivery function.

$$\ln y = K + \alpha \ln I + \beta \ln M + v - u \tag{36}$$

where y =output
I=Infrastructure cost Service delivery
M=Maintenance cost

v=random shocks
u=technical inefficiency

$$\alpha + \beta = n \tag{37}$$

CRS: n=1* Constant returns to scale*
IRS: $n > 1$* Increasing returns to scale*
DRS: $n < 1$* Decreasing returns to scale*
By solving these two equations, the following values of elasticity can be derived.

$$\alpha = \frac{\ln y - K - \ln M - v + u}{\ln \frac{I}{M}} \tag{38}$$

$$\beta = \frac{\ln y - K - \ln I - v + u}{\ln \frac{M}{I}} \tag{39}$$

The detailed proof is contained in Appendix D.

E. Revenue Model

As defined by Reference (9) For a company, this is the total amount of money received by the company for goods sold or services provided during a certain time period. It also includes all net sales, exchange of assets; interest and any other increase in owner's equity and is calculated before any expenses are subtracted. To generalize the profit model, its an equation like,

$$Profit = Revenue - Cost \tag{40}$$

[3] The 8.3 Section details about the cost minimization and 8.4 section about the quantitative way to define revenue in service computing scenario of the enterpriser. To define the cost in service computing, our business should also have the penalty for SLA violation, which is not practiced by any Cloud market player at present. The cost in this case should be the summation of the standard cost as defined in SLA agreement copy and additionally the penalty for SLA violation.

$$Cost = StandaredCost + PenaltyduetoSLAviolation \tag{41}$$

The result of our Lotka-Volterra model shows that at any given point, the utilization is more than 80. The rate of utilization of the proposed algorithm is much less penalty prone, as compared to the existing algorithm.

$$SLA_{LV} < SLA_{TDH}$$

$$=> CostSLA_{LV} < CostSLA_{TDH}$$

$$=> -CostSLA_{LV} > -CostSLA_{TDH}$$

$$=> ProfitSLA_{LV} > ProfitSLA_{TDH}$$

[3]The stochastic model is available as a working paper on arXiv.org "CDSFA Stochastic Frontier Analysis Approach to Revenue Modeling in Large Cloud Data Centers"

F. Big Player vs Small Player:

The question regarding quality of services offered by firms belonging to the information technology sector might be addressed in several ways. While, there are several impediments to measure quality of services or even products, yet quality alongside price is expected to offer a clear indication about consumer retention in a market. Indeed, one of the reasons why quality matters is because consumers derive inferences about the level of competition in the market from the perceived and observed quality of the product. Thus quality represents a fundamental aspect of competition in many markets (see Zeithaml, 2000; Darby and Karni, 1973, etc). Quality represents, perhaps the key non-price consideration that determines whether consumers will purchase a product. It is commonly expected that larger is the market power enjoyed by a few firms, the poorer would be the quality of the product and services. Hotels run by state enterprises, airlines operated by flagship carriers of the state, telecommunication or electricity services managed and operated by state controlled firms are glaring examples of poor quality of services. This has been a typical practice when the entry by private enterprises are fairly restricted by law (such as in erstwhile USSR) or by internal and external entry barriers (high import duties and strict regulations about foreign investment in India) rendering state monopoly as the only source to buy from. Evidently, the economic outcomes have largely been detrimental for both the state enterprises due to complacency and lack of quality monitoring as well as for private entrepreneurs who diverted investments to more friendly locations. This outcome is however, not expected in case of private firms competing in a market (also with public firms) to attract and retain customers. Unfortunately, despite the presence of anti-trust laws in several countries, the evolution of large private firms in certain sectors have been so overwhelming that the same quality of service question is back in the forefront. It is easy to argue that the large private firms have been able to attract a bigger pie by being quality conscious and by delivering. However, in the process, through buy-outs, licensing agreements, through joint-ventures, forced takeovers, etc these firms have gained supreme market power comparable with the prevalence of monopoly power in such sectors. Consequently, it has somewhat stifled the drive for innovation within a market, which usually leads to dynamic efficiency. In part, therefore, it is imperative that we measure the degree of concentration in such markets to observe the prevailing level of competition. We employ Hirschman-Herfindahl Index (HHI) to measure the level of competition or concentration of such firms. The Hirschman-Herfindahl Index (HHI) is a widely used technique for measuring the degree of market concentration. It is calculated by summing the squared market share of each firm competing in a given market. The value of HHI can vary between approximately zero to 10,000. The HHI is expressed as:

$$HHI = s_1^2 + s_2^2 + s_3^2 + ... + s_n^2 \tag{42}$$

Here s_n is the market share of the ith firm.

High HHI implies a few firms control the market. Thus, strategically or otherwise, firms may be less quality conscious under such circumstances. Conversely, if the market is shared by a large number of firms, the HHI value is low and poor quality of products and services could mean consumer attrition. Consequently, technological deficiencies leading to intermittent black-outs, disruption of services such as call-drops, etc are expected to be low - firstly for the fear of losing subscribers and secondly as better competition allows technological innovations and entry of more efficient firms. Thus, to begin with, let us discuss the degree of market concentration for firms present in the Asia-Pacific region. The region has generated just over USD 20 billion in data center infrastructure revenues for the world's leading technology vendors and the market and has grown by 23% from the previous year, according to data from Synergy Research Group [40].

$$HHI = 21^2+19^2+11^2+8^2+8^2+4^2+4^2+25^2 = 1708 \quad (43)$$

Since, this also captures a legal dimension, it is important to mention that the U.S. Department of Justice considers a market to be competitive if it shows a HHI score less than 1000. Conversely, if the score is between 1000 and 1800, the market is deemed as moderately concentrated, while a score above 1800 suggests a highly concentrated marketplace [39]. In the Asian-Pacific Economic Cooperation (APEC) zone, the concentration is moderate and tending towards a highly concentrated marketplace. The HHI for IaaS market share is given below

$$HHI = 27.2^2+16.6^2+11.8^2+3.6^2+2.7^2+2.4^2+35.9^2 = 2456.34 \quad (44)$$

If we exclude 'others' (i.e. firms other than Amazon, IBM, Oracle, Google and RackSpace highlighted in fig **??**) from HHI calculation, it becomes 1167.53. And yet, it cannot be considered as a competitive market. Overall, only a few firms seem to control the major share of the market for infrastructure as a service.

The coexistence of concentrated markets and disruption in services might therefore seem plausible if newer and more efficient firms typically face artificial entry barriers to such markets. Otherwise, if technological innovation is the sole criterion that helps newer (smaller) firms to compete with incumbent (bigger) firms with disproportionately large sums of investment, then except for entry barriers the market would cease to remain as concentrated as we find here. For example, innovations such as reduction in SLA violation may help small firms to gain market share provided it is not prohibitively costly for smaller firms to invest in. For all practical purposes, the strong Research and Development support coming from non-corporate sectors and open sources in many developed and developing countries provide a strong platform for smaller firms to compete with larger firms. And yet, the conditions do not seem to have matured enough to generate the desired welfare impact on the consumers at large. We shall, subsequently engage with identifying possible sources of entry barriers in this market. This might ideally include estimates of switching cost for consumers, brand loyalty, predatory pricing by incumbents, credit market imperfection in some parts of the world, role of venture capital, consumers' taste for quality, etc in relation to the market for Cloud computing.

XII. Related Work

For horizontal scaling, the user should define a fixed amount say S number of VMs to be allocated or deallocated but vertical scaling the same number S signifies the amount of resources(CPU, RAM) needs to be added [18]. The same trend can be followed in some other papers also. There are a couple of papers where upper and lower utilization threshold value of reactive scaling is the objective. Beloglazov et al, introduces adaptive threshold which is efficient to meet the high level of SLA [17]. Automated cloud-based scalability is a hot research topic in cloud computing. Fuzzy logic has been implemented in elasticity controller which enables qualitative specification of elasticity rules [19]. Fuzzy logic also utilized by Xu et al, in elasticity controller to learn the relationship between workload, resources and the learned fuzzy rules applied during resources allocation [20].Another famous approach in cloud controller is black-box surrogate model, which evolves over time and uses machine learning to predict the performance [21]. Lim et. al. [6] employed a linear equation to calculate the vm number in case of threshold violation (elasticity). The equation is heavily dependent on two parameters actuator values and sensor measurement. CPU utilization is considered as sensor variable and actuator represents the number of virtual storage instances allocated as storage nodes. The relationship between workload and CPU utilization has been established empirically. Whereas in Lotka-Volterra model, the major contributors in the equations are the number of virtual machines and the cloudlets number and certainly LV is a non-linear equation, which is reasonable as linear equation may not be efficient every time to represent the real time situation .Chieu et al. [8] have shown a dynamic scaling algorithm for automated provisioning of virtual machine resources based on threshold number of active sessions. There is a previous work , which suggests a hybrid controller, an amalgam of proactive and reactive controllers [23]. Another work subscribes the same concept and demonstrates the different possible scenarios of proactive elastic controller deployment in cloud incorporation with reactive elastic controller [22]. Tesauro et al. [9] demonstrates the strength of reinforcement learning in a sequential decision process, in which reinforcement learning trains offline on data collected while a queuing model policy controls the system. Though most of the authors consider two threshold values, upper and lower but Hasan et al, [10] have proposed 4 threshold values. ThrbU, is slightly below the upper threshold and ThroL is slightly above the lower threshold. A model-predictive algorithm is defined by Roy

et. al., is responsible for autoscaling of resources. A second order autoregressive moving average method (ARMA) is used to predict the workload and the optimization of the system behavior is achieved by minimizing various costs such as SLO violations, cost of leasing resources and reconfiguration cost [12]. Waheed et al, proposed a prototype is based on reactive scaling, which contineously keep monitoring the average response time, if it violates the required response time, it adds a VM [11]. SCADS has leveraged the utility function to scale-up and scale-down the storage resources dynamically and machine learning is demonstrated to predict the resource requirement of new quires before execution [7].On-demand cloud resource allocation plan is costlier than reservation plan. In paper [36] the author proposed an algorithm for optimal cloud resource provisioning using stochastic programming model to overcome that problem. In the next part the author applied a decomposition algorithm to divide the actual optimization problem into multiple smaller problems such that these can be solved independently and in parallel. However the methodology has several complexities. In the paper [37] and [38] agent technology is used to control dynamic environment like cloud. Singh et al. have proposed a QoS based resource provisioning and scheduling framework, where workloads are clustered using workload pattern and again reclustered by k-means clustering algorithm to identify the Qos requirements. Different scheduling policies are employed to accomplish the scheduling task [24]. The Lotka-Volterra model has been integrated with existing cloudsim timeshared algorithm and improvement is observed by evaluating the performance on different QoS parameters. Load balancing Ant Colony Optimization problem (LBACO) has been explored as a task scheduling policy which is a NP herd optimization problem. It incorporates the dynamic behavior of the cloud and balances the entire system [25]. The intitution behind the LV time shared scheduling algorithm is to improve the performance and avoiding under-provisioning/ over-provisioning situation, load balance is not accommodated int the current solution. Particle swarm optimization is an another approach exploited in a previous paper, where it has taken into consider both computation cost and data transmission cost [26].The same pso algorithm has implemented in grid environment to achieve the optimized scheduling task [28]. LV timeshared algorithm is not dedicated to a particular environment. Varalakshmi et al. [17] presented an optimal workflow-based scheduling (OWS) framework to identify a solution that can satisfy various user-desired QoS constraints, such as execution time [27]. Lotka-Volterra model is widely used in the field of biological science especially to describe the population dynamics of two interacting species. Takeuchi et al. considered the evolution system consisted of two two predator–prey deterministic systems denoted by Lotka–Volterra equations in random environment [29]. The periodic Lotka–Volterra predator-prey system is investigated with impulsive effect [30]. Chaos in three chain systems with LV model type interactions is showcased in another paper [31]. Nicola has made an attempt to establish a relationship between the LV model and predator-prey utility functions [32].

XIII. Discussion and CONCLUSION

The Lotka-Volterra model also provides us with a mathematical property known as limit cycles which is described in contour portraits, also known as phase portraits of the system. Limit cycle describes a qualitative limit for the stability of a system. Parameters of a system are differed such that the system grows out of stability and difference acquired by the parameters is measured to tell the domain of stability. This has direct application in understanding the stability of a web-server with incoming requests. Limit cycle of a system along with rate of incoming requests can help us understand the bounds of the system. As the proposed model is based on the dynamic interaction between the predator and the prey, the second contribution is addressing elasticity for highly volatile need of customers. The next section details on how, by using this model the optimum level of elasticity can be achieved. Our third contribution is a time shared algorithm. The paper implemented the experiment simulation in CloudSim. The proposed Lotka-Volterra model has the existing cloudlet time sharing algorithm of CloudSim and takes advantage of the predator-prey equation. Without externally defining any dynamic allocation scheduling algorithm, the improved time-shared algorithm decides the VM occupancy before submitting the incoming cloudlet request to VM. Other two contributions of this paper are Improvement on Quality of Services and minimization of SLA(Service Level Agreement) violation. The simulation in the CloudSim reveals that the number of the future VM has not increased or decreased as per predefined static allocation rule. On the contrary, the model decides on the number of VM. The QoS parameters which include throughput, response time, etc show that the proposed model is more suitable to address each Quality metric. The results of the experiments in the next section detail the same. SLA defines the terms and conditions among two parties as a basis for measuring agreed quality of service standards and optimization between both Cloud provider and customer. Our contributed work is quantifying the SLA violation parameter where the non-fulfillment of service should be penalized. The proposed model shows significant reduction in SLA violation, i.e low punishment. This indirectly increases the profitability. Our last contribution is proposing a business model for the small and medium scale enterprises who can target Cloud business. Here is a detailed discussion on how the proposed model quantifies the cost and profitability. An analysis has been done from the economic point on how the entry-barrier challenge can be addressed for the new Cloud service providers. It can be intuited from above that, a dynamic environment like Cloud follows lower dimensional chaos (Non linear dynamics). The motive of the project was to bring about the Cloud parameters under different situations and model them using non-linear dynamics. The parameters were calculated at the boundary conditions using a Java based

simulation platform called CloudSim. The dynamic creation of cloudlets allowed us to vary the number of cloudlets and the resources of the Cloud to create a system which was able to showcase all possible scenarios. Lotka-Volterra model suited as the best model to describe the Cloud parameters to reasonable accuracy. Phase portrait can be used to determine over-utilization of resources leading the whole system into instability. A phase portrait is plotted repeatedly by using the Cloud data. When there is a sign of instability in the system, more resources can be added to bring the system into equilibrium state. We conclude by highlighting the strengths of our contribution.

- stability implies proportioal change on VM based on changes of demand in cloudlets, this is controlled by lotka volterra. Diffetial eqation not affected by stocastic uncertainty.Therefore the ballpek figures of VMs based on cloudlets cloud be achived reliabily and efficiently. This is the interpretation of stability between VMs and Cloudlets and adroitly exploited in our work (Please refer to the subsection Prey-Predator stability).
- The difference in modeling elasticity (our approach) from other approaches available in the literature needs to be highlighted. VM's are added as per requirement in most approaches whereas we allocate VM's governed by the underlying LV model without undermining the utilization threshold.
- We achieved parameter tuning by LV to control VM population.
- We exploited the predictive analytics from the simulation of our LV model to estimate approximate VM population against the demand of cloudlets. Polynomial interpolation was applied to arrive at this estimate. As prediction accuracy is reasonable, there is less scope for overloading of systems.
- Improvement in SLA violation was accomplished. Other QoS metrics are found to be better than existing elastic work.
- The technological innovation suggested by our model should encourage further competition in a market controlled by the big players. **Upload the amazon SLA violation charges into github and refer in the main text–pending**
- VM allocation is accomplished by the proposed model, truly dynamic in fashion with out requiring overheads or look up tables. Appropriate improvisations were effected in the basic Cloudsim setting to accommodate dynamic behavior.

APPENDIX A

Lotka-Voterra Model as below:

$$\frac{dP}{dt} = \alpha P - \beta PQ \tag{45}$$

$$\frac{dQ}{dt} = \delta PQ - \gamma Q \tag{46}$$

The stability of this model is attained, when there is no growth rate for both P, Q.

$$\alpha P - \beta PQ = 0$$
$$=> P(\alpha - \beta Q) = 0$$

Either P=0 or $\alpha - \beta Q$=0.
P=0 is not possible. Hence

$$\alpha - \beta Q = 0$$
$$=> \alpha = \beta Q$$

if $\beta = 1$, than $\alpha = Q$
Similar way

$$\delta PQ - \gamma Q = 0$$
$$=> Q(\delta P - \gamma) = 0$$
$$=> \gamma = \delta P$$

if $\delta = 1$, $\gamma = P$
Therefore stability of the proposed model occur, where $\alpha = Q$,$\gamma = P$

APPENDIX B

Matlab Code for figure

```
span = 0 : 0.1 : 100;
[t, y] = ode45(@Lotka, span, [60; 80]);
plot(t, y(:, 1), '-o', t, y(:, 2), '-o')
function dydt = Lotka(t, y)
dydt = [20 * y(1) - y(1) * y(2); -20 * y(2) + y(1) * y(2)];
end
```

REFERENCES

[1] http://www.conservationnw.org/what-we-do/predators-and-prey/carnivores-predators-and-their-prey
[2] http://mathworld.wolfram.com/Lotka-VolterraEquations.html
[3] Bidisha Goswami and Snehanshu Saha : Resource Allocation in Abstraction using Predator-Prey Dynamics: A Qualitative Analysis, Volume 61– No.6, January 2013
[4] H. Kopka and P. W. Daly, *A Guide to LaTeX*, 3rd ed. Harlow, England: Addison-Wesley, 1999.
[5] Darby, M. R. and F. Karni. 1973. "Free Competition and the Optimal Amount of Fraud."Journal of Law and Economics 16 (April): 67–86
[6] Lim, Harold C., Shivnath Babu, and Jeffrey S. Chase. "Automated control for elastic storage." Proceedings of the 7th international conference on Autonomic computing. ACM, 2010.
[7] Armbrust, M., Fox, A.,Patterson, D. A.,Lanham, N., Trushkowsky, B., Trutna, J. and Oh, H.,2009, Scads: Scale-independent storage for social computing applications. In Proc. of CIDR.
[8] Chieu, T. C., Mohindra, A.,Karve, A. A., and Segal, A.,2009, Dynamic scaling of web applications in a virtualized cloud computing environment. In e-Business Engineering. ICEBE'09, 281-286.
[9] Tesauro, G., Jong,K. N.,Das,R., and Bennani, N. M.,2006,A hybrid reinforcement learning approach to autonomic resource allocation. ICAC '06: Proceedings of the 2006 IEEE International Conference on Autonomic Computing. IEEE Computer Society, 65–73.
[10] Hasan, Z. M.,Magana, E.,Clemm, A.,Tucker, L., and Gudreddi, D. L. S.,2012, Integrated and autonomic cloud resource scaling. In Network Operations and Management Symposium (NOMS), IEEE, 1327-1334.

[11] Iqbal, W.,Dailey, M., and Carrera, D.,2011, Adaptive resource provisioning for read intensive multi-tier applications in the cloud. Future Generation Computer Systems, 27(6):871-879, doi: 10.1016/j.future.2010.10.016. URL http://dl.acm.org/citation.cfm?id=1967762.1967921.

[12] Roy, N., Dubey, A., and Gokhale, A.,2011, Efficient Autoscaling in the Cloud Using Predictive Models for Workload Forecasting. In 2011 IEEE 4th Interna- tional Conference on Cloud Computing, 500-507. doi: 10.1109/CLOUD.2011.42. URL http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=6008748.

[13] Fito, O. J., Goiri, N., I., and Guitart, J., SLA-driven Elastic Cloud Hosting Provider,Barcelona Supercomputing Center Technical University of Catalonia Barcelona, Spain

[14] Aslanpour, M. S., Dashti, S. E.,2016, SLA-Aware Resource Allocation for Application Service Providers in the Cloud.Second International Conference on Web Research (ICWR)

[15] CHIANG, A. C., AND WAINWRIGHT, K. Fundamental methods of mathematical economics, 4. ed., internat. ed., [repr.] ed. McGraw-Hill [u.a.], Boston, Mass. [u.a.], 2009.

[16] Herbst, N.,Samuel, K., Ralf, R., 2013,Elasticity in Cloud Computing What It Is, and What It Is Not. Proceedings of the 10th International Conference on Autonomic Computing, San Jose, CA, June 24-28

[17] Beloglazov, A., and Buyya, R.,2010, Adaptive Threshold-Based Approach for Energy-Efficient Consolidation of Virtual Machines in Cloud Data Centers. In Proceedings of the 8th International Workshop on Middleware for Grids, Clouds and e-Science, page 4. ACM

[18] Lorido-Botrán, T.,Miguel-Alonso, J., and Lozano, A. J.,2012, Autoscaling Techniques for Elastic Applications in Cloud Environments, University of Basque Country, Tech. Rep. EHUKAT-IK-09-12.

[19] Jamshidi, P., Ahmad, A., Pahl, C.,2014, Autonomic Resource Provisioning for Cloud-Based Software. Proceedings of the 9th International Symposium on Software Engineering for Adaptive and Self-Managing Systems,95-104,doi 10.1145/2593929.2593940

[20] Xu, J., Zhao,M., Fortes,J.,Carpenter, R., and Yousif, M.,2007, On the use of fuzzy modeling in virtualized data center management.ICAC.

[21] Gambi, A., Toffetti, G., and Pezzè, M.,2013, Assurance of self-adaptive controllers for the cloud," in Assurances for Self-Adaptive Systems.

[22] Ali-Eldin, A., Tordsson, J., and Elmroth, E.,2012, An Adaptive Hybrid Elasticity Controller for Cloud Infrastructures. Network Operations and Management Symposium (NOMS),IEEE,doi 10.1109/NOMS.2012.6211900

[23] Urgaonkar,B., Shenoy,P., Chandra,A., Goyal,P., and Wood,T.,2008, Agile dynamic provisioning of multi-tier Internet applications, ACM Transactions on Autonomous and Adaptive Systems (TAAS),3(1), 1. 2008.

[24] Singh, S., Chana, I.,2016, Resource provisioning and scheduling in clouds: QoS perspective. J. Supercomput,72(3), 926-960.

[25] Li, K., Xu, G., Zhao, G., Dong, Y., Wang,D.,2011, Cloud task scheduling based on load balancing ant colony optimization. In: Chinagrid Conference (ChinaGrid), 2011 Sixth Annual. IEEE, pp 3–9

[26] Pandey, S., Wu, L., Guru, S., Buyya, R., 2010, A particle swarm optimization-based heuristic for scheduling workflow applications in cloud computing environments. In: Advanced information networking and applications (AINA), 24th IEEE international conference, Perth, Australia

[27] Varalakshmi, P., Ramaswamy, A., Balasubramanian, A., Vijaykumar, P., 2011, An optimal workflow based scheduling and resource allocation in cloud. Advances in computing and communications. Springer, Berlin, Heidelberg

[28] Zhang, L., Chen, Y., Sun, R., Jing, S., and Yang, B., 2008, A task scheduling algorithm based on pso for grid computing. International Journal of Computational Intelligence Research, 4(1).

[29] Takeuchi, Y. , Du, H. N., Hieu, T. N., Sato, K., 2006. Evolution of predator–prey systems described by a Lotka–Volterra equation under random environment, Journal of Mathematical Analysis and Applications, 323, 938-957

[30] S.Y. Tang, L.S. Chen, The periodic predator-prey lotka-volterra model with impulsive effect, J. Mech. Med. Biol. 2 (2002) 267–296.

[31] Liu, XN., Chen, LS.,2003 Complex dynamics of Holling type II Lotka–Volterra predator–prey system with impulsive perturbations on the predator. Chaos, Solitons & Fractals,16(2):311–20.

[32] Serra, N.,2014,Utility Functions and Lotka-Volterra Model: A Possible Connection in Predator-Prey Game. Journal of Game Theory, 3(2), 31-34, DOI: 10.5923/j.jgt.20140302.03

[33] Kolmogoroff, N. A.,1936, Sulla teoria di Volterra per la lotta per l'esistenza, Giornale Ist. Ital. Attuari, 7 (74-80).

[34] Keller,A. A., 2011. Stochastic delay Lotka-Volterra system to interacting population dynamics. System, 1(1), 0.

[35] Goel, S.N., Maitra, C. S., Montroll, W. E.,1971, On the Volterra and Other Nonlinear Models of Interacting Populations, Academic Press, New York.

[36] Chaisiri, S., Lee, S. B., and Niyato, D.,2012,Optimization of Resource Provisioning Cost in Cloud Computing, IEEE TRANSACTIONS ON SERVICES COMPUTING, 5(2).

[37] Luck, M., McBurney, P., and Preist, C., and the AgentLink Community, Agent Technology: Enabling Next Generation Computing, Agent Technology, a roadmap, page 94- 1: 94©Agentlink

[38] Kang, M., Wang, L., and Taguchi, K., 2004, Modeling Mobile Agent Applications in UML2.0 Activity Diagrams, In Proceedings of the Sixth International Conference on Enterprise Information Systems, Porto, Portugal, 519-522.

[39] http://www.investopedia.com/terms/h/hhi.asp, accessed on 22/1/2016

[40] https://www.srgresearch.com/articles/hp-ibm-and-dell-lead-burgeoning-apac-data-center-infrastructure-market, accessed on 20/10/2016.

PLACE PHOTO HERE

Michael Shell Biography text here.

John Doe Biography text here.

Jane Doe Biography text here.