International Journal of Control and Automation
Vol. 13, No. 2s, (2020), pp. 36 - 50

A Hybrid Self-Adaptive PSO and QoS Based Machine Learning Model for Cloud Service Data

*Sirisha Potluri[1], Katta Subba Rao[2]

[1, 2] *Department of Computer Science and Engineering, Koneru Lakshmaiah Education Foundation, Vaddeswaram, AP, India, 522502.*

[1]*sirisha.vegunta@gmail.com*, [2]*subbarao_cse@kluniversity.in*

Abstract

All the users present in the cloud expect services with high quality since they pay for those services. To provide expected quality of service to the users cloud providers need to maintain sufficient resources. To manage these resources in an efficient manner and in order to maintain the QoS, a capable scheduling algorithm is required. To address this issue, a self-adaptive based allocation of various resources can be used, which has the ability to change the resource allocation dynamically according to the different conditions in the cloud environment. There are many such methods and approaches available in literature in which a set of rules are defined with respect to each service to take better decisions for efficient resource allocation. Self-adaptive based allocation of resources can be achieved by using machine learning approach and control theory. In this paper we propose a hybrid self-adaptive PSO and QoS based machine learning model for cloud service data. The proposed model has three modules. The first module contains an improved quality of service-based cloud service ranking and recommendation model to find the essential top ranked services in the cloud environment. The second module contains a hybrid PSO optimization model integrated with recommendation and ranking model for efficient task scheduling in cloud computing environment. In the third module we proposed a hybrid self-adaptive PSO and QoS based machine learning model for cloud service data. We have compared the performance of the existing algorithms with the proposed algorithm based on average accuracy of each task, average error rate of each task, task level runtime for response time prediction and job level runtime for response time prediction. During every comparison the proposed algorithm is giving better results.

Keywords: *Cloud computing, Quality of service, Machine learning, Control theory, Self-adaptive resource allocation, Scheduling, PSO*

1. INTRODUCTION

Due to dynamism, elasticity and vagueness present in the computing environment, resources allocation to the submitted tasks is a highly challenging task. Cloud users expects high quality of service in terms of response time, cost and throughput, in order to maintain and guarantee these requirements the cloud service provider need to have more number of resources. Simply make use of more number of resources does not give an

effective solution. An efficient mechanism is required to balance the client requirements such as quality of service and cloud service provider requests such as consumption of fewer resources. To address this issue, a self-adaptive based allocation of various resources can be used, which has the ability to change dynamically according to the different conditions in the cloud environment. There are many such methods and approaches available in literature in which a set of rules are defined with respect to each service to take better decisions for efficient resource allocation. During this process the system takes a much amount of administration overhead and lot of difficulty involved in implementation. To avoid these drawbacks further researches proposed two different types of approaches [1-5].

2. SELF-ADAPTIVE BASED APPROACHES FOR RESOURCE ALLOCATION

2.1. Based on machine learning

One of these approaches is based on machine learning technique which makes use of historical data to gain domain related knowledge of the various software services of the cloud environment. This model predicts the values for various metrics of quality of service based on the current workload of the cloud environment and followed by using some efficient scheduling algorithms to allocate the resources. The main drawback and limitation of this model is that to build the prediction model based on QoS, it requires large and large amount of data which is inadequate, insufficient and does not cover dynamic workload conditions [6].

2.2. Based on control theory

The second approach is based on control theory, which uses feedback mechanism so as to dynamically adjust the resource allocation process in cloud computing environment. By using this approach set of parameters namely control inputs are identified which actually effect the virtual machine allocation. To identify such set of control inputs to effectively schedule the resources large number of feedback based iterations are required. Due to this lot of overhead on virtual machine has been created in connect and disconnect of them during resource allocation process [7].

3. LITERATURE SURVEY

J. Panerati et al. proposed autonomic computing which provides expected services to users by overcoming the difficulties and complexities existing in the computing environment. This in turn makes the operations of the service provider more complex and difficult. To overcome these difficulties and to achieve the objectives of autonomic computing a novel framework with monitoring infrastructure has been introduced in this paper. This architecture has two modules. One of these is used to take decisions regarding scheduling of tasks and another module is to take care of workload distribution of the processing elements or cores [8].

Sadip Midya et al. proposed a prediction model driven by quality of service which is based on iterative approach using historical data. This model makes use of machine learning concepts to get the domain specific knowledge and PSO based algorithm to take run time decisions regarding resource allocation. The objective of this framework is to achieve efficient scheduling using large number of feedback based iterations [9].

Xing Chen et al. proposed an efficient model to address the issues of dynamic behavior, elastic nature and uncertainty environment of the computing environment. Since conventional approaches are completely based on rules, this approach is used to predict the quality of service based on the historical data and resource allocation is achieved by using genetic algorithm approach. By using this resource allocation happens in online and identifies the best reasonable resource allocation plan exists in the environment [10].

Xing Chen et al. proposed architecture based on self-adaptive approach to dynamically the resources based on demand in the computing environment. This approach is based on two approaches namely feedback loops based prediction model and PSO based decision model to schedule the resources. Prediction model tries to improve the QoS prediction based on the demand and PSO based algorithm tries to improve the resource allocation compared to the existing allocation [11].

J. Lin et al. proposed a framework to handle the on demand computing requirements when load is dynamically varying with increasing number of requests. Conventional approaches have limited scope to respond to such cases. To overcome this problem the author has suggested a looping mechanism which contains four modules namely monitoring the environment, analyze the dynamic requirements, plan for the reasonable resource allocation and finally execute the plan to achieve the maximum utilization of the resources. This model has efficiently constructed the management strategy using machine learning approach [12].

S. Gong et al. proposed an efficient model which can handle the dynamically fluctuating cloud service requests in the computing environment. Single resource scheduling model may not be sufficient to handle the dynamic requirements which are submitted to the computing environment. This model proposed self-adaptive based on multi variable control approach [13].

Marios Avgeris et al. proposed a two level resource allocation model which provides the mechanism for allocation of resources and a mechanism for admission control to execute the submitted tasks. At the lower level of the proposed model a set of linearly working systems are used to meet the system requirements and at the higher level an optimized tracking algorithm is used to maximize the usage ratio of the resources [14].

F. Guo et al. proposed a framework based on block chain which is self-adaptive in nature to handle the ongoing dynamic requests made by cloud users. Block chain based technology is used to improve the throughput of the system. Using deep reinforcement

learning approach and edge computing approaches used in this model the intractable problems are addressed in this model [15].

V. Podolskiy et al. proposed a model which can minimize the service level agreement/ objectives violations using vertical scaling of the collocated containers/ processing elements [16].

S. Liu et al. proposed a model which is evolutionary in nature and using new grouping mechanism to reasonably allocate the resources to the cloud users. This algorithm has proved competitive efficiency when compared with the state of the art algorithms in the literature [17].

4. PROPOSED MODEL FRAMEWORK

Self-adaptive resource allocation for cloud tasks is proposed using iterative QoS prediction model [18]. The proposed model frame work can be visualized as shown in the figure 1. When multiple tasks are submitted to the cloud server there exist a set of virtual machines to execute them. Using various parameters such as minimum response time, maximum response time, average response time, throughput, cost etc. training dataset has been prepared. To achieve optimization, in this framework we are proposing a QoS based PSO model which uses parameters such as response time and throughput constraints. To extract the efficient patterns and to achieve best optimization we use a wrapper model. To take a decision on available patterns and to choose the best pattern so as to make the model as self-adaptive learning model we can make use of Nonlinear SVM, Decision tree and nonlinear regression.

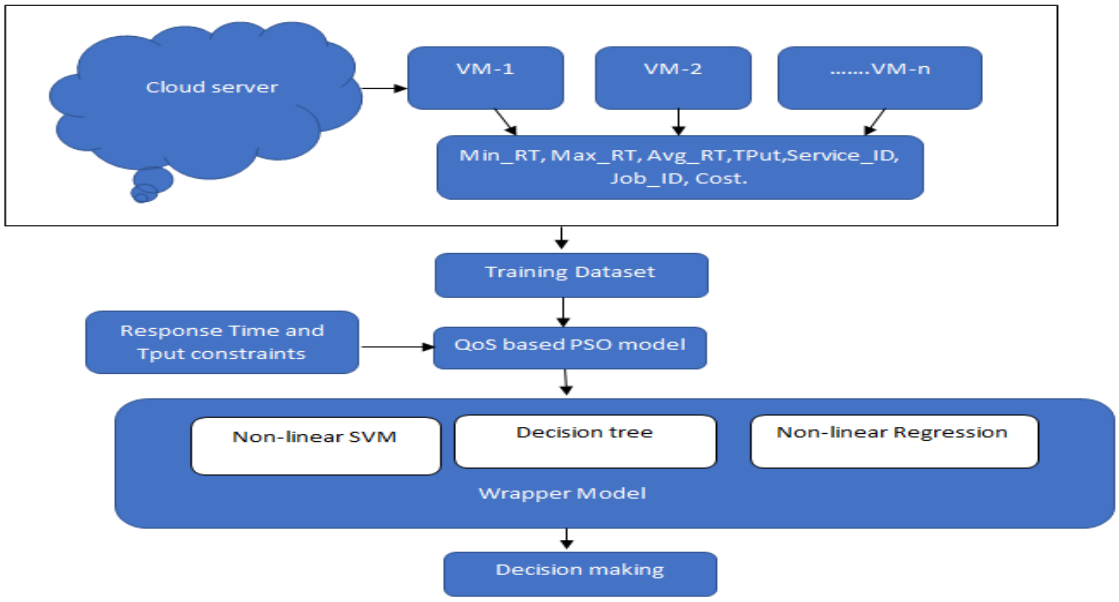


Figure 1. Proposed framework

The proposed model has three modules. The first module contains an improved quality of service-based cloud service ranking and recommendation model to find the essential top ranked services in the cloud environment. The second module contains a hybrid PSO optimization model integrated with recommendation and ranking model for efficient task scheduling in cloud computing environment. In the third module we proposed a hybrid self-adaptive PSO and QoS based machine learning model for cloud service data.

4.1. Improved quality of service-based cloud service ranking and recommendation model

QoS based service ranking model is proposed in cloud computing environment to find the essential top ranked services. Proposed model is implemented in two phases. In the first phase, similarity computation between the users and their services is considered. In the second phase, computing the missing values based on the computed similarity measures is calculated.

Algorithm:

Step 1: Computing the task rating similarity requested by the active and normal user is given by

$\bar{Lr}_a$ and $\bar{Lr}_n$ represents the average log likelihood estimator of all services invoked by active and normal users to service .

The contextual task similarity requested by the active cloud and normal cloud users is given by

$$simR(r_a, r_n) = \frac{\sum_{s \in S}(r_{a,s} - \bar{Lr}_a)(r_{n,s} - \bar{Lr}_n)}{\sqrt{\sum_{s \in S}(r_{a,s} - \bar{Lr}_a)^2}\sqrt{\sum_{s \in S}(r_{n,s} - \bar{Lr}_n)^2}} * (\max(T_a, T_n))$$

Step 2: In this step, the similarity computation of the users based on the QoS values is given by

The similarity between the users interacting with the service s is

$$simU(v_a, v_n) = simU(QoS(v_a, v_n)) = \frac{\sum_{s \in S}(v_{a,s} - \bar{v}_{a,s})(v_{n,s} - \bar{v}_{n,s})}{\sqrt{\sum_{s \in S}(v_{a,s} - \bar{v}_{a,s})^2}\sqrt{\sum_{s \in S}(v_{n,s} - \bar{v}_{n,s})^2}}$$

Step 3: QoS missing value prediction using the weighted similarity measures as

Step 4: Normalized confidence score for users is computed as

$$\text{Normalized Confidence score for users} = \phi_U = \text{NCWU} = N(\text{simU}(v_a, v_n).(\sum_{u \in U} \frac{\text{simU}(v_a, v_n)}{\sum_{u \in U} \text{simU}(v_a, v_n)}))$$

Step 5:Normalized confidence score for tasks is computed as

$$\text{Normalized Confidence weighted score for tasks} = \phi_T = \text{NCWS} = N(\text{simR}(r_a, r_n).(\sum_{s \in S} \frac{\text{simR}(r_a, r_n)}{\sum_{s \in S} \text{simR}(r_a, r_n)}))$$

Step 6:Weighted score for users is computed as

$$\text{weighted score of users} = w_u = \frac{\tau.\phi_U}{\tau(\phi_U) + (1-\tau)\phi_T}$$

Step 7:Weighted score for tasks is computed as

$$\text{weighted score of tasks} = w_T = \frac{(1-\tau).\phi_T}{\tau(\phi_T) + (1-\tau)\phi_U}$$

Step 8:Weigted predicted score for QoS missing value is computed as

$$\text{Predicted value for QoS missing values} = \frac{2(w_u.w_T)}{(w_u + w_T)}.\text{Max}\{\phi_{U[i]}, \phi_{T[j]}\}$$

Step 9: Apply PSO algorithm for users to tasks ranking.

4.2. A hybrid PSO optimization model integrated with recommendation and ranking model

To enhance the performance of PSO algorithm by optimizing task response time a hybrid PSO based task selection and recommended system has been proposed.

Algorithm:

Step-1: Initialize number of tasks, number of data centers, number of VMs and number of particles.

Step-2: Compute Weighted score for users Wu and tasks on the training dataset Wt.

Step-3: To each VM in the VMList
Do
VM response time in the cloud server V_{diff}.
Where $V_{diff=}V_{max-}\ V_{min}$
Done

Step-4: Compute QoS using the min, max and average response time.

The cost function of the QoS model is given as cost function is shown below

$$F(c) = V_{diff}\,(\,RT_{max}s\ \rho\ +\ RT_{min}\,(1-p_k).Wq\,)$$

$$W_q = RT_\mu\ /\ \lambda_e$$

Where

Effective Arrival Rate λ_e :

$$\lambda_e = \mu\ (1-P_k)$$

$$P_n = \left(\lambda^n\ /\ S!\ S^{n-s}\mu^n\right)\ P_{o,}$$

λ = mean arrival rate of tasks in each service.

ρ : price

μ = the average response time of each server

s = Number of Servers

K = Total number of cloud services;

Let P_n be the probability that there are n tasks in the cloud services in the running state n.

Step-5: Randomly assign each particle to each task.

Step-6: Randomly initialize each particle position and velocity.

Step-7: Set min position, max position and max velocity.

Step-8: To each particle compute fitness value and update global best and neighbor best positions.

Step-9: Update particle's position and velocity by using each instance CPUU and Gaussian values as

$$\text{GaussianValue}: gv = \frac{1}{2.\pi}e^{\sum(gbest-pbest)}$$

$$\text{UpdatedVelocity}: V_{i+1} = w * v_i + C.(V_{diff}) * (pbest_i - x_i) + (C * gv / PS)) * (gbest_i - x_i)$$

$$w = 0.9$$

$$PS := \text{ParticleSize}$$

Step-10: Repeat Step-8 until all tasks are scheduled

Step-11: Stop

4.3. Self-Adaptive Machine learning Model for cloud service data

4.3.1. Hybrid Non-linear SVM

In this proposed non-linear SVM, a hybrid kernel function is used to estimate the response time-based rules on the training data. The objective function of the proposed non-linear SVM model is given as

$$\min_{w,b,\chi,\rho}\{\frac{1}{2}w^{T}w - v.\eta + \chi\sum_{i=1}^{l}\lambda_i\}$$

s.t.

$$y_i(w^{T}\cdot\phi(x_i,y_i)+b) \geq \eta - \lambda_i$$

$$,\lambda_i \geq 0, i = 1\ldots l, \eta \geq 0$$

$$\phi(x_i,y_i) = e^{\chi.\log(\sum|y_i^2|)}; \text{if}\,(x_i > y_i)$$

$$= e^{\chi.\log(\sum|x_i^2|)}; \text{if}\,(x_i < y_i)$$

$$= e^{\chi.\log(\sum|x_i - y_i|^2)}; \text{if}\,(x_i == y_i)$$

The decision boundary is given by

$$\text{sgn}(\sum_{i=1}^{l} y_i.\phi(x_i,y_i)+b)$$

4.3.2. Decision tree Construction

In the decision tree construction, a hybrid entropy measure is used to generate the decision patterns on the input training data. In the decision tree construction process, enhanced entropy is computed to each attribute for node selection process. The attribute with highest entropy value is selected as the decision tree node for pattern generation. This process is repeated until the number of attributes is satisfied. An improved entropy measure used in the decision tree construction is give as:

Enhanced entropy:

$$\text{Pr} = -\text{Prob}(D_i).\log(\text{Prob}(D_i))$$

$$\text{Ent}(D) = \sum_i \text{Pr}$$

$$\text{ModEnt}(D) = \frac{\text{Ent}(D).\log(\text{Ent}(D))}{\sqrt{\text{Ent}(D)*\log(2)}}$$

5. RESULTS AND DISCUSSIONS

Experimental results are simulated using the amazon aws server and cloud service training data. Results are developed using the NetBeans and JDK environment with different third party libraries. Sample training data is shown in table 1. Table 2, describes the automatic patterns generated on the training data using the proposed machine learning model. These patterns or rules are used to automate the cloud server for different types of services or tasks.

Table 1. Sample Training Data

AVAILABILITY	MAX_RT	MIN_RT	AVG_RT	THROUGH	SERVICE_ID	IDX	COST	JOB_ID	TASK_INDEX	TASK_USAGE_IDX
363	3758	1146.36	120	1.96	41	1012	3.432794	39308692	0	752165
252	1945	806.03	166	2.71	41	1014	5.328627	39308692	1	898663
322	6817	1463.86	123	2.02	41	1017	1.293431	39308692	1	95
95	689	255.51	269	4.45	41	1021	3.432794	39308692	0	752165
222	2105	639.1	199	3.3	41	1023	3.432794	39308692	0	752165
109	1461	385.64	223	3.69	41	1027	1.293431	39308692	1	95
108	1436	361.83	227	3.71	41	1028	3.313186	39308692	1	752166
167	1733	449.48	209	3.44	41	1030	1.928611	39308692	1	132267
182	6915	863.31	156	2.54	41	1032	1.704608	39308692	0	613661
4565	9430	6759.34	32	0.52	42	1037	6.958529	46542951	1	898665
5309	8361	6679.89	37	0.61	42	1039	1.856176	46542951	0	132268
3031	9045	5113.02	39	0.62	42	1042	1.265173	46542951	0	421856
3132	8797	6008.46	30	0.49	42	1043	1.649461	46542951	1	97
3151	8521	6196.94	37	0.59	42	1047	6.502647	46542951	1	614260
95	577	223.55	279	4.63	42	1049	6.958529	46542951	1	898665
4506	9410	6297.02	34	0.55	42	1052	20.68987	46542951	0	277529
113	1725	286.63	233	3.86	42	1058	6.958529	46542951	1	898665
92	6957	696.28	169	2.74	42	1060	6.958529	46542951	1	898665

Table 2. Automatic Patterns Generated using Proposed Decision tree

```
IDX < 840.5
|  SERVICE_ID < 24.5
|  |  SERVICE_ID < 23.5
|  |  |  MIN_RT < 147.25 : 0.47 (171/0.25)
|  |  |  MIN_RT >= 147.25
|  |  |  |  JOB_ID < 3418387.5 : 0.61 (151/0.34)
|  |  |  |  JOB_ID >= 3418387.5
|  |  |  |  |  COST < 2.24
|  |  |  |  |  |  COST < 1.33
|  |  |  |  |  |  |  IDX < 282
|  |  |  |  |  |  |  |  IDX < 180 : 0 (2/0)
|  |  |  |  |  |  |  |  IDX >= 180
|  |  |  |  |  |  |  |  |  TASK_USAGE_IDX < 751350 : 2.38 (13/0.39)
|  |  |  |  |  |  |  |  |  TASK_USAGE_IDX >= 751350 : 0 (1/0)
|  |  |  |  |  |  |  IDX >= 282 : 0.79 (28/0.17)
|  |  |  |  |  |  COST >= 1.33 : 0.5 (52/0.37)
|  |  |  |  |  COST >= 2.24
|  |  |  |  |  |  SERVICE_ID < 14.5
|  |  |  |  |  |  |  THROUGHPUT < 4.01
|  |  |  |  |  |  |  |  MAX_RT < 1401 : 4 (1/0)
|  |  |  |  |  |  |  |  MAX_RT >= 1401
|  |  |  |  |  |  |  |  |  AVAILABILITY < 63.5 : 0 (2/0)
|  |  |  |  |  |  |  |  |  AVAILABILITY >= 63.5
|  |  |  |  |  |  |  |  |  |  MIN_RT < 439.32 : 2 (6/0.33)
|  |  |  |  |  |  |  |  |  |  MIN_RT >= 439.32 : 0.67 (3/0.22)
|  |  |  |  |  |  |  THROUGHPUT >= 4.01
|  |  |  |  |  |  |  |  COST < 6.61 : 3.8 (5/0.16)
|  |  |  |  |  |  |  |  COST >= 6.61 : 2 (2/0)
|  |  |  |  |  |  SERVICE_ID >= 14.5
|  |  |  |  |  |  |  SERVICE_ID < 17.5
|  |  |  |  |  |  |  |  IDX < 255.5 : 0.64 (11/0.23)
|  |  |  |  |  |  |  |  IDX >= 255.5
|  |  |  |  |  |  |  |  |  MIN_RT < 242.87
|  |  |  |  |  |  |  |  |  |  COST < 7.03 : 2 (2/0)
|  |  |  |  |  |  |  |  |  |  COST >= 7.03 : 0.33 (3/0.22)
|  |  |  |  |  |  |  |  |  MIN_RT >= 242.87
|  |  |  |  |  |  |  |  |  |  MAX_RT < 2896 : 1 (1/0)
|  |  |  |  |  |  |  |  |  |  MAX_RT >= 2896
|  |  |  |  |  |  |  |  |  |  |  TASK_USAGE_IDX < 848.5 : 1.5 (2/0.25)
|  |  |  |  |  |  |  |  |  |  |  TASK_USAGE_IDX >= 848.5 : 2.83 (6/0.14)
|  |  |  |  |  |  SERVICE_ID >= 17.5 : 0.43 (14/0.24)
|  |  SERVICE_ID >= 23.5
|  |  |  THROUGHPUT < 5.07
|  |  |  AVG_RT < 288
|  |  |  |  COST < 1.92
|  |  |  |  |  AVAILABILITY < 31 : 4.5 (2/0.25)
|  |  |  |  |  AVAILABILITY >= 31 : 6.33 (3/0.22)
|  |  |  |  COST >= 1.92
|  |  |  |  |  IDX < 485.5
|  |  |  |  |  |  TASK_USAGE_IDX < 751373
|  |  |  |  |  |  |  MIN_RT < 123 : 2 (2/0)
|  |  |  |  |  |  |  MIN_RT >= 123 : 0 (1/0)
|  |  |  |  |  |  TASK_USAGE_IDX >= 751373 : 4 (1/0)
|  |  |  |  |  IDX >= 485.5
|  |  |  |  |  |  COST < 16.22
|  |  |  |  |  |  |  AVAILABILITY < 35.5 : 4 (2/0)
|  |  |  |  |  |  |  AVAILABILITY >= 35.5 : 2.33 (3/0.22)
|  |  |  |  |  |  COST >= 16.22
|  |  |  |  |  |  |  MAX_RT < 956.5 : 7 (1/0)
|  |  |  |  |  |  |  MAX_RT >= 956.5 : 5 (1/0)
|  |  |  AVG_RT >= 288
|  |  |  |  COST < 7.21 : 0 (4/0)
|  |  |  |  COST >= 7.21
|  |  |  |  |  AVAILABILITY < 23 : 0 (1/0)
|  |  |  |  |  AVAILABILITY >= 23 : 4.67 (3/0.22)
|  |  THROUGHPUT >= 5.07
|  |  |  MAX_RT < 884.5 : 3 (1/0)
|  |  |  MAX_RT >= 884.5
|  |  |  |  MIN_RT < 70.06 : 4 (1/0)
|  |  |  |  MIN_RT >= 70.06 : 7 (2/0)
|  SERVICE_ID >= 24.5 : 0.34 (308/0.3)
```

Table 3. Task level response time prediction

Task level Prediction		
Models	Accuracy	Error Rate
RT + QoS	0.78	0.227
SVM + QoS	0.847	0.157
NLREG + QoS	0.746	0.268
IRT + QoS	0.967	0.046

Table 4. Job level Response time prediction

Job Level Prediction		
Models	Accuracy	Error Rate
RT + QoS	0.793	0.213
SVM + QoS	0.864	0.138
NLREG + QoS	0.783	0.218
IRT + QoS	0.957	0.057

Task level response time and job level response time values are mentioned in table 3 and 4. These values are compared for various models based on QoS namely Random Tree, Support Vector Machine, Non Linear Regression and Improved Random Tree.

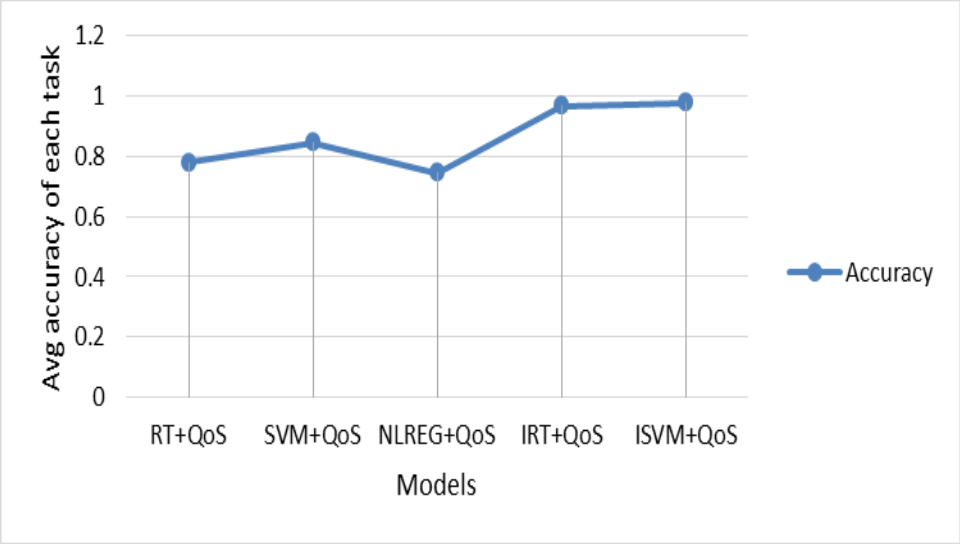


Figure 2. Average accuracy of the each task using the machine learning models

The average accuracy values of each task against various machine learning models have been shown in figure 2. Similarly the average error rates of each task against various machine learning models have been shown in figure 3.

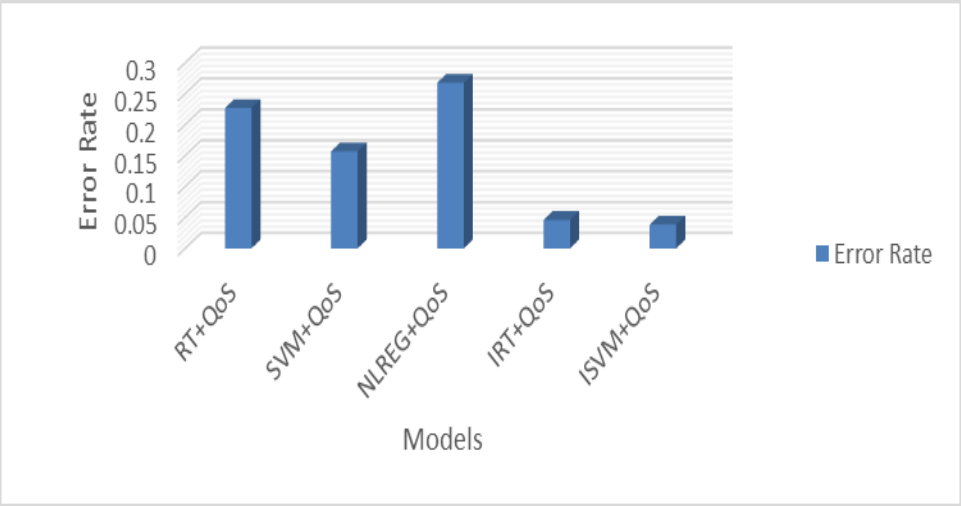


Figure 3. Average error rate of the each task using machine learning models

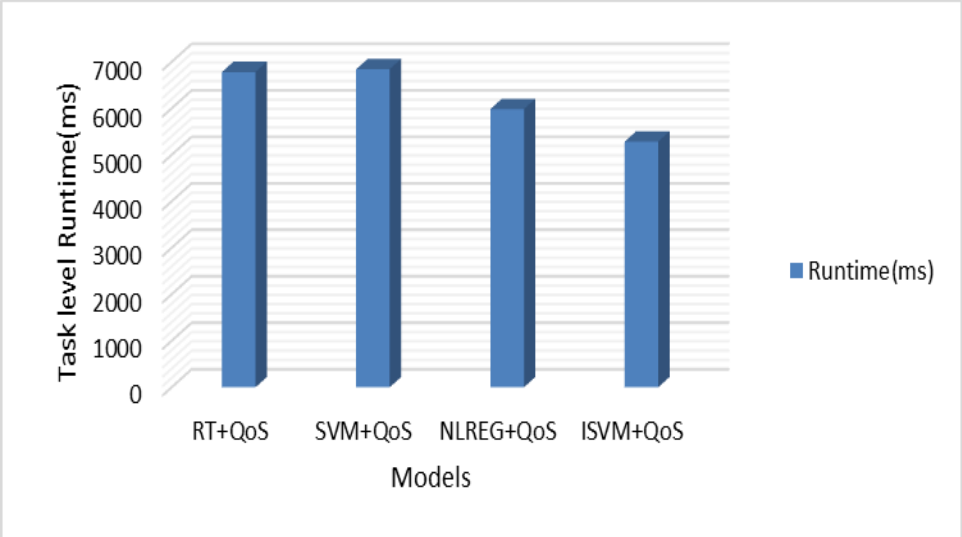


Figure 4. Task level runtime of the machine learning models for response time prediction

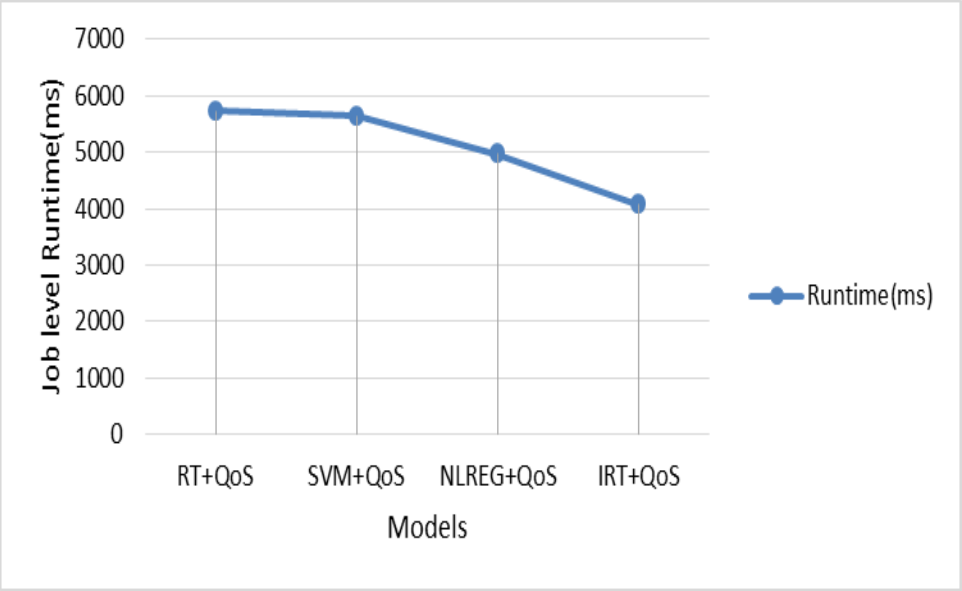


Figure 5. Comparison of job level response time prediction using different machine learning models

The task level runtime values of each task against various machine learning models have been shown in figure 4. Similarly the comparison of job level response time prediction using different machine learning models has been shown in figure 5.

6. CONCLUSION AND FUTURE WORK

Cloud users expects high quality of service in terms of response time, cost and throughput, in order to maintain and guarantee these requirements the cloud service provider need to have more number of resources. Simply make use of more number of resources does not give an effective solution. An efficient mechanism is required to balance the client requirements such as quality of service and cloud service provider requests such as consumption of fewer resources. To address this issue, a self-adaptive based allocation of various resources can be used, which has the ability to change dynamically according to the different conditions in the cloud environment. The proposed model has three modules. The first module contains an improved quality of service-based cloud service ranking and recommendation model to find the essential top ranked services in the cloud environment. The second module contains a hybrid PSO optimization model integrated with recommendation and ranking model for efficient task scheduling in cloud computing environment. In the third module we proposed a hybrid self-adaptive PSO and QoS based machine learning model for cloud service data. We have compared the performance of the existing algorithms with the proposed algorithm based on average accuracy of each task, average error rate of each task, task level runtime for response time prediction and job level runtime for response time prediction. During every comparison the proposed algorithm is giving better results.

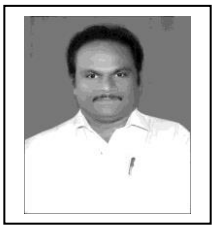
REFERENCES

[1] Devi R, Shanmugalakshmi R, Cloud providers ranking and selection using quantitative and qualitative approach, Computer Communications, Volume 154, **2020**, Pages 370-379.
[2] Mohit kumar, SC Sharma, PSO-COGENT: Cost and Energy Efficient scheduling in Cloud environment with deadline constraint, Sustainable Computing: Informatics and Systems, **2018**. DOI: 10.1016/j.suscom.2018.06.002.
[3] Chandra shekar Jatoth, G R Gangadharan, Ugo Fiore, Optimal fitness aware cloud service composition using modified invasive weed optimization, Swarm and Evolutionary Computation, Volume 44, February **2019**, Pages 1073-1091.
[4] Najme Mansouri, Behnam Mohammad Hasani Zade, Mohammad Masoud Javidi, Hybrid task scheduling strategy for cloud computing by modified particle swarm optimization and fuzzy theory, Computers & Industrial Engineering, 130, **2019**, pp. 597-633.
[5] G. Rjoub and J. Bentahar, "Cloud Task Scheduling Based on Swarm Intelligence and Machine Learning," 2017 IEEE 5th International Conference on Future Internet of Things and Cloud (FiCloud), Prague, **2017**, pp. 272-279.
[6] Hao Meng, Daichong Chao, Qianying Guo, Xiaowei Li, Delay-sensitive Task Scheduling with Deep Reinforcement Learning in Mobile-edge Computing Systems, IOP Conf. Series: Journal of Physics: Conf. Series 1229 (**2019**) 012059.
[7] V. Janarthanan, P. Gohari, A.Saffar, Formalizing Real-Time Scheduling Using Priority-Based Supervisory Control of Discrete-Event Systems, IEEE TRANSACTIONS ON AUTOMATIC CONTROL, VOL. 51, NO. 6, JUNE **2006**.
[8] J. Panerati, Filippo Sironi, Matteo Carminati, Martina Maggio, Martina Maggio, Piotr J. Gmytrasiewicz, Donatella Sciuto, Marco D. Santambrogio, "On self-adaptive resource allocation through reinforcement learning," 2013 NASA/ESA Conference on Adaptive Hardware and Systems (AHS-2013), Torino, **2013**, pp. 23-30.
[9] Sadip Midya. Asmita Roy, Koushik Majumder, Santanu Phadikar, An adaptive resource placement policy by optimizing live VM migration for ITS applications in vehicular cloud network, Transactions on emerging telecommunications technologies, **2019**.
[10] Xing Chen, Junxin Lin, Bing Lin, Tao Xiang., Self-learning and self-adaptive resource allocation for cloud-based software services, Concurrency and Computation Practice and Experience , **2018**.
[11] Xing Chen, Haijiang Wang, Yun Ma, Xianghan Zheng, Self-adaptive resource allocation for cloud-based software services based on iterative QoS prediction model, Future Generation Computer Systems 105, December **2019**.
[12] J. Lin, Y. Dai, X. Chen and Y. Wu, "Resource Allocation of Cloud Application Through Machine Learning: A Case Study," 2017 International Conference on Green Informatics (ICGI), Fuzhou, **2017**, pp. 263-268.
[13] S. Gong, B. Yin, Z. Zheng, K. Cai, "Adaptive Multivariable Control for Multiple Resource Allocation of Service-Based Systems in Cloud Computing," in IEEE Access, vol. 7, pp. 13817-13831, **2019**.
[14] Marios Avgeris, Dimitrios Dechouniotis, Nikolaos Athanasopoulos, Symeon Papavassiliou, Adaptive Resource Allocation for Computation Offloading: A Control-Theoretic Approach, Article No.: 23, CM Transactions on Internet Technology, April **2019**.
[15] F. Guo, F. R. Yu, H. Zhang, H. Ji, M. Liu and V. C. M. Leung, "Adaptive Resource Allocation in Future Wireless Networks With Blockchain and Mobile Edge Computing," in IEEE Transactions on Wireless Communications, vol. 19, no. 3, pp. 1689-1703, March **2020**.
[16] V. Podolskiy, M. Mayo, A. Koay, M. Gerndt and P. Patros, "Maintaining SLOs of Cloud-Native Applications Via Self-Adaptive Resource Sharing," 2019 IEEE 13th International Conference on Self-Adaptive and Self-Organizing Systems (SASO), Umea, Sweden, **2019**, pp. 72-81.
[17] S. Liu, L. Liu, X. Liu, Y. Wang and B. Bai, "A New Evolutionary Algorithm Based on Self-Adaptive Grouping and Efficient Resource Allocation," 2019 IEEE Intl Conf on Dependable, Autonomic and Secure Computing, Intl Conf on Pervasive Intelligence and Computing, Intl Conf on Cloud and Big Data Computing, Intl Conf on Cyber Science and Technology Congress, Fukuoka, Japan, **2019**, pp. 22-27.

[18] Xing Chen, Haijiang Wang, Yun Ma, Xianghan Zheng, Self-adaptive resource allocation for cloud-based software services based on iterative QoS prediction model, Future Generation Computer Systems, Volume 105, April **2020**, pp. 287-296.

Authors

Ms. Sirisha Potluri (Corresponding Author)
Research Scholar
Department of CSE
Koneru Lakshmaiah Education Foundation
Andhra Pradesh, India- 522502
Mail id: sirisha.vegunta@gmail.com



Dr. Katta Subba Rao
Professor
Department of CSE
Koneru Lakshmaiah Education Foundation
Andhra Pradesh, India- 522502
Mail id: subbarao_cse@kluniversity.in