IOSR Journal of Engineering (IOSR JEN) **www.iosrjen.org**
ISSN (e): 2250-3021, ISSN (p): 2278-8719
PP 01-04

Performance Evaluation of Low Latency Communication Alternatives in A Containerized Cloud Environment

Blessed Sam[1], Mr.S.Prasanna[2].
B Pg Student Mailam Engineering College
Associate Professor Cse Mailam Engineering College

Abstract**:** *Reliability in cloud systems is an important aspect of delivering stable cloud services for users. Focusing on improving successful execution of tasks under resource constraints, this work proposes an enhanced and effective resource management method to achieve reliability within the cloud environment. The proposed method employs an adaptive reinforcement learning algorithm merged with the queuing theory to schedule user requests. There are many dynamic changes in the cloud environment in terms of resource availability and attributes that make a reliable task execution difficult to guarantee. As a solution to this problem, our approach employs a task scheduler, which can effectively adapt to those dynamic changes and successfully schedule user requests. We developed an adaptive action-selection method that aims to control the action selection dynamically (i.e., suitable virtual machine selection), considering the queue buffer size and uncertainty value function. To evaluate the performance of our approach, we conduct several experiments and compare our approach with greedy and random job scheduling policies, in terms of successful task execution, utilization rate, and response time. The numerical results demonstrate the efficiency of our method.*

I. Introduction

Cloud computing is a method of calculating or computing a service that is provided through the Internet by deploying various models and layers of abstraction. Reliability in cloud computing is a significant issue for quality of service. One of the main drivers for cloud computing is cost and reliability. Because personal computers and their software are becoming more complex, the installation, removal, and updates of such computer systems demand significant time. Instead, outsourcing the computation tasks can eliminate these problems. The cloud of computer grids offers such services on a fee-based schedule, which in the long term can cost users considerably more than buying, maintaining, and upgrading servers. Moreover, resource sharing improves the overall usage of the same. Owing to a large number of virtual machines (VM) performing in the cloud data center, it is difficult to guarantee that all the VMs continuously perform satisfactorily. There is a lack of reliability in cloud services because many VMs exist, increasing the risk of failure. Thus, it is important to enhance the reliability of services based on VMs within a cloud computing environment. Reinforcement learning (RL) provides a plain knowledge-free trial-and-error methodology in which a learner attempts various actions in several system states, and learns from the penalties of each. Developing self-adaptive methods for task scheduling policy in service applications has become a key research field, especially in utility optimization of distributed systems. Various methods have been proposed, based on RLs, for automatically learning management policies. To measure the indicators of performance, such as completion time and waiting time, we deploy a Markov request queue model, developed considering the shared resources among VMs and several types of failures. However, no consideration is given for the subtask's buffer size, which is illogical in an environment like cloud computing.

1.1 PRELIMINARIES:

In the proposed model, the scheduler analyzes the incoming input tasks, followed by the scheduling strategy utility values, which then reviews the better methodology to assign incoming tasks to the appropriate VMs. The task scheduling method in our model can be formulated as a Markov decision process (MDP), in which the state space is denoted by S, action set is represented by A, and the immediate reward function represented by *r(st, at)*. The task is assigned to the chosen scheduling strategy, where it is scheduled to the appropriate VM.

Reinforcement Learning:

The technique of RL is based on increasing the likelihood of a behavior by offering a reward for behavior accuracy. An RL problem can be defined as an MDP, in which repeated steps can enhance processing. Without a priori knowledge, RL captures the performance of a target application and its policy. RL scope focuses on direct interaction between an agent and its environment, Fig. 1 shows the basic structure of an RL.

The agent is the decision-maker, learning from past experiments to execute the best action for each state of the environment by maximizing the returned reward.

$Q(s, a) = r\ (s, a) + © \ max_{a'}\ (Q(s', a'))$ (1)

where r(s, a) is the immediate reward, and _ is the relative value of delayed rewards against immediate rewards (0 to 1).s*'* is the new state after action, *a'*. *a* and *a'* are actions in states, s and s*'*, respectively.
2) To evaluate performance metrics, such as average waiting time or queue length for tasks, the classical queuing theory has been widely used. The mean arrival rate for client requests in our model is _, and those requests are queued until they are processed. In the proposed model, one or more VMs can be made available to the requests at a mean service rate, µ. We used Kendall's notation [9] to designate and categorize queuing models. A queue can be described in shorthand notation as A/B/C/K/N/D or A/B/C. The elements, K, N, and D, are noncompulsory; we assume that K = 1, N = 1, and D =first-in first- out in the absence of those elements.

1.2 TASK SCHEDULING METHOD:
We developed our experiments using the CloudSim framework, an extensible framework that supports simulation modeling of service scheduling, virtual resource allocation, and other functions [10]. To evaluate the performance of our system, we conducted several experiments and compared the results of our prosed system with two scheduling methods: greedy scheduling policy and random scheduling policy. The experimental setup in our simulation system is stated in Table 1. The RL parameters values were set: _=0.5 (Gamma value), _=0.5 (Alpha value), _=0.01 (Epsilon value), and _=10.0, for the Poisson distribution.

policy output.
1, ( )
2, ( )
3, ( )
4, ( )
0, ( )
if VM queue is full fu
if VM queue is middle mi
M
S M where f if VM queue is over middle om
f
if VM queue is semi free sm
if VM queue is free fr
= □ □
□
____ ___
(5)
Fig. 2 shows the working model of our proposed system

II. Proposed System

In the proposed model, the scheduler analyzes the incoming input tasks, followed by the scheduling strategy utility values, which then reviews the better methodology to assign incoming tasks to the appropriate VMs. The task scheduling method in our model can be formulated as a Markov decision process (MDP), in which the state space is denoted by S, action set is represented by A, and the immediate reward function represented by *r(st, at)*. The task is assigned to the chosen scheduling strategy, where it is scheduled to the appropriate VM.

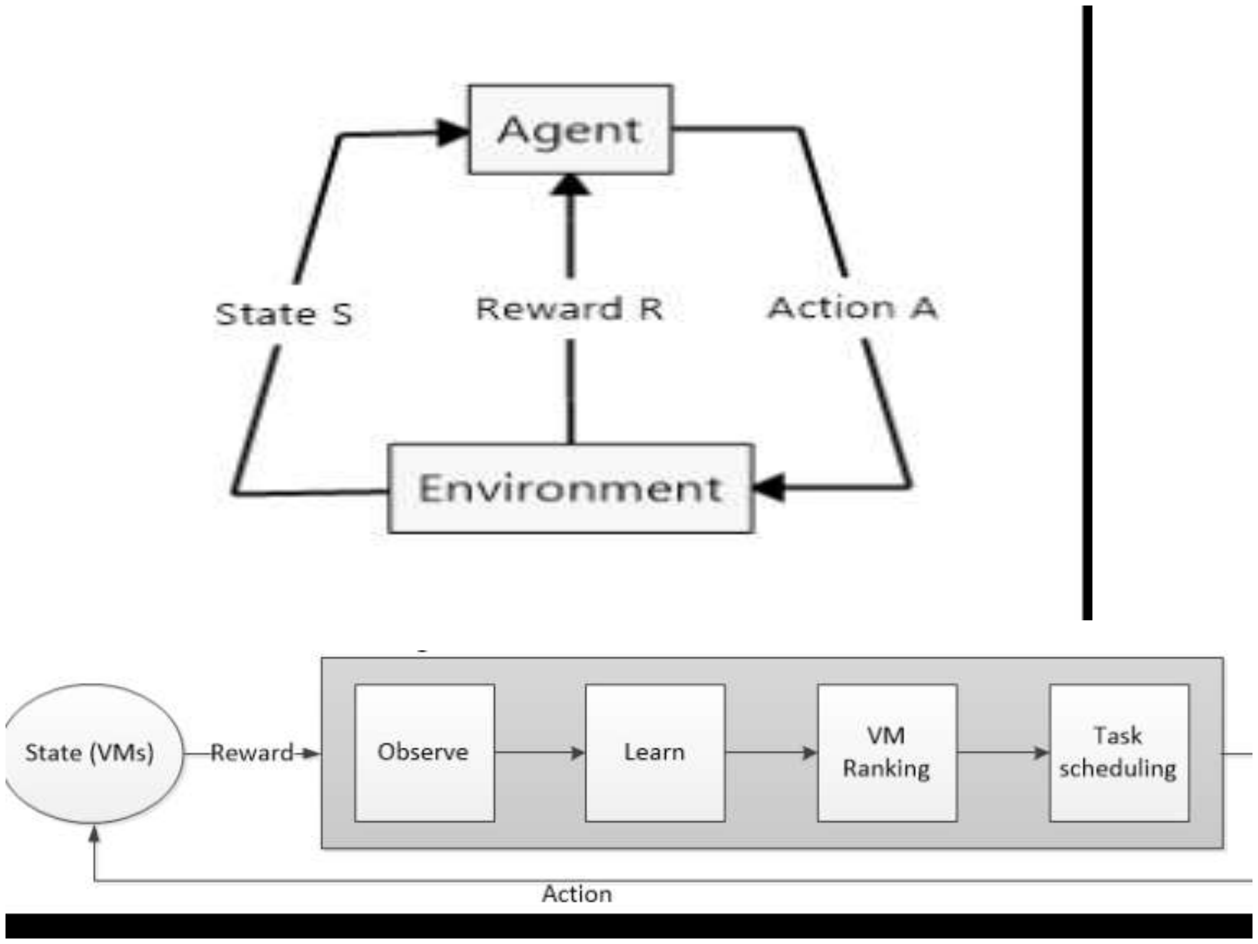

Figure 2. The working model of the proposed system.

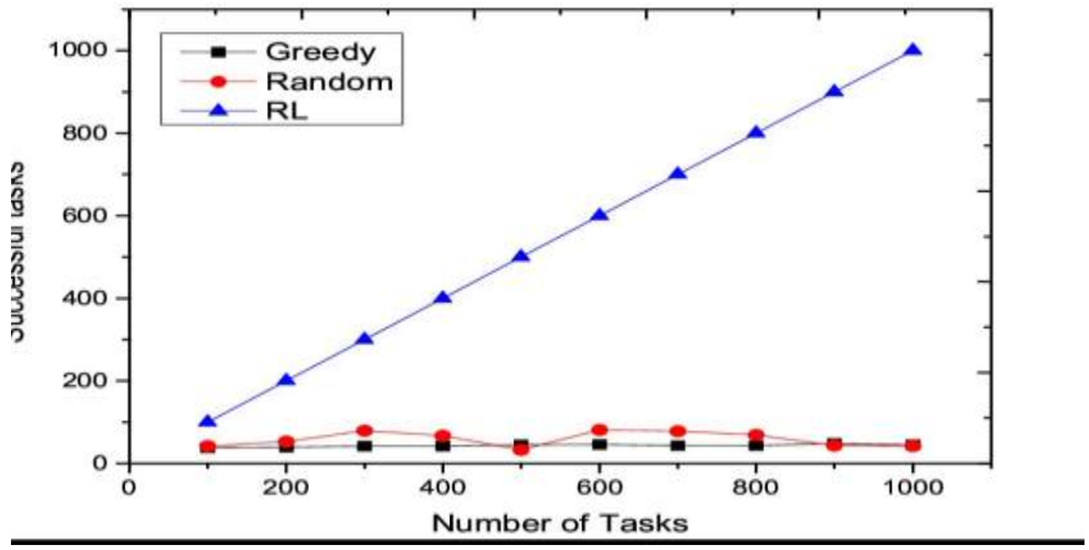

Figure 3. Comparison of successful tasks under various task numbers

EXPERIMENTS AND RESULTS:

We developed our experiments using the Cloud Sim framework, an extensible framework that supports simulation modeling of service scheduling, virtual resource allocation, and other functions [10]. To evaluate the performance of our system, we conducted several experiments and compared the results of our proposed system with two scheduling methods: greedy scheduling policy and random scheduling policy. The experimental setup in our simulation system is stated in Table 1. The RL parameters values were set: _=0.5 (Gamma value), _=0.5 (Alpha value), _=0.01 (Epsilon value), and _=10.0, for the Poisson distribution.

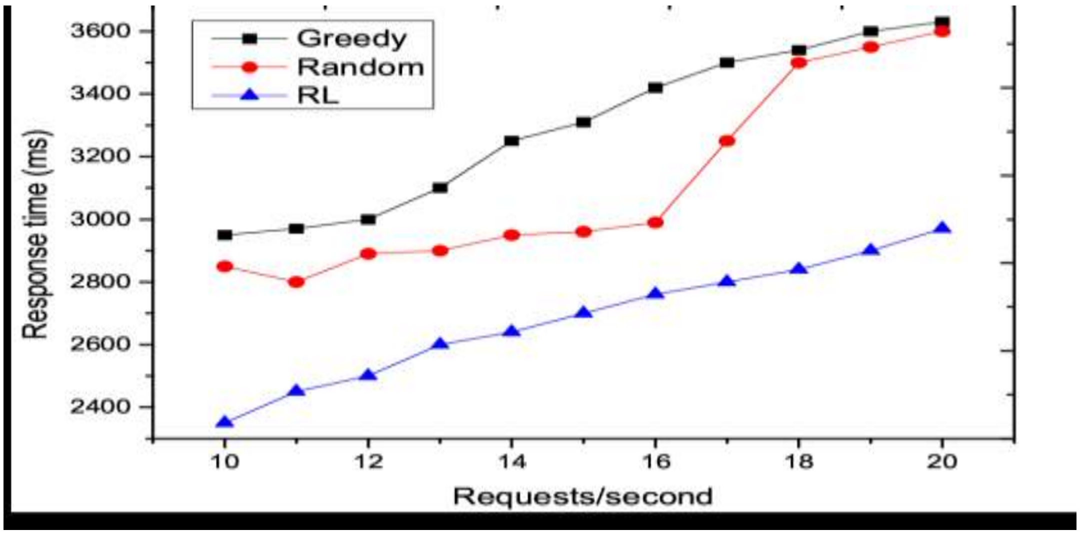

Figure 5. Comparison in term of the response time.

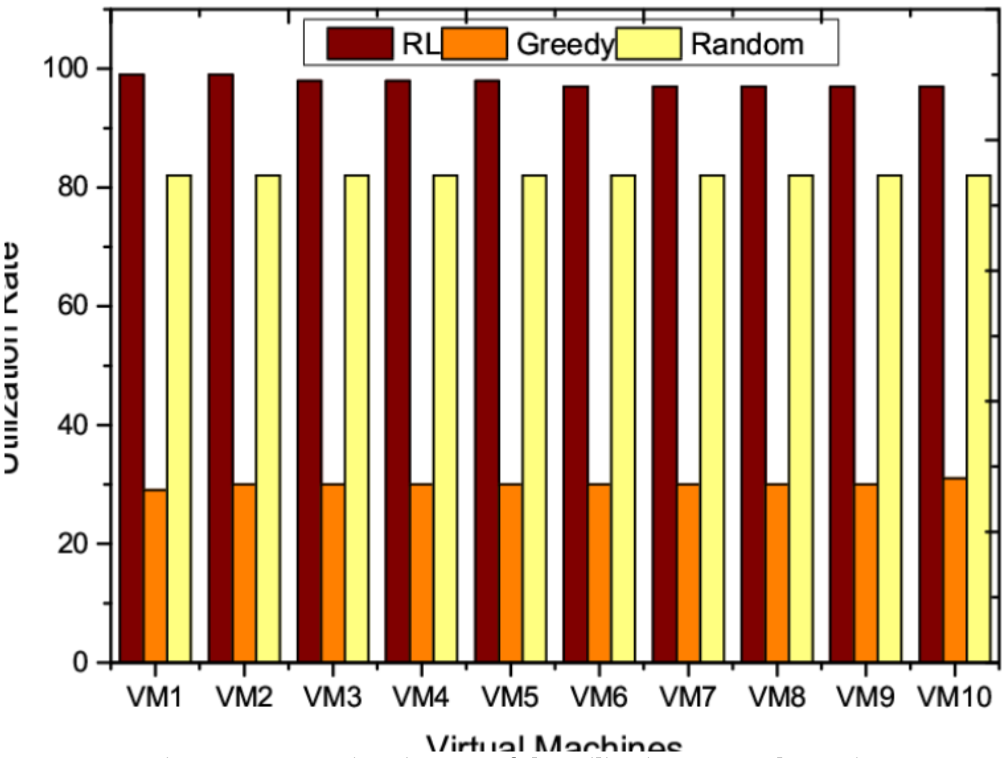

Figure 4. Comparison in term of the utilization rate under various VMs.

III. Conclusion

We developed our experiments using the CloudSim framework, an extensible framework that supports simulation modeling of service scheduling, virtual resource allocation, and other functions [10]. To evaluate the performance of our system, we conducted several experiments and compared the results of our proposed system with two scheduling methods: greedy scheduling policy and random scheduling policy. The experimental setup in our simulation system is stated in Table 1. The RL parameters values were set: _=0.5 (Gamma value), _=0.5 (Alpha value), _=0.01 (Epsilon value), and _=10.0, for the Poisson distribution.