# Expert System and Heuristics Algorithm for Cloud Resource Scheduling

**Mamatha E** (sricsrmax@gmail.com)
Dept of Engineering Mathematics, GITAM University, Bangalore, India

**Sasritha S**
Dept of Engineering Mathematics, GITAM University, Bangalore, India

**CS Reddy**
School of Computing, SASTRA University, Thanjavur, India

## ABSTRACT

*Rule-based scheduling algorithms have been widely used on cloud computing systems and there is still plenty of room to improve their performance. This paper proposes to develop an expert system to allocate resources in cloud by using Rule based Algorithm, thereby measuring the performance of the system by letting the system adapt new rules based on the feedback. Here performance of the action helps to make better allocation of the resources to improve quality of services, scalability and flexibility. The performance measure is based on how the allocation of the resources is dynamically optimized and how the resources are utilized properly. It aims to maximize the utilization of the resources. The data and resource are given to the algorithm which allocates the data to resources and an output is obtained based on the action occurred. Once the action is completed, the performance of every action is measured that contains how the resources are allocated and how efficiently it worked. In addition to performance, resource allocation in cloud environment is also considered.*

**Keywords:** *Cloud computing, Scheduling and Expert System, Heuristic Models*

**JEL Classification:** *C87*

## INTRODUCTION

Cloud Computing, the long-held dream of computing industry, has the capability to change large IT industry, making software even more usable as a service and changing the way IT hardware is made and purchased. Developers with creative ideas for new Internet services no longer need the large capital costs in hardware to install their service or the human expense to operate it [1-5]. They need not be bothered about over-provisioning for a service whose attractiveness does not meet their predictions, thus wasting the

resources, or under-provisioning for one that becomes wildly accepted, thus missing potential customers and revenue.

The prominent feature of cloud computing that it allows the consumption of services over internet with subscription based model. Based on the level of abstraction, various models for cloud computing like Infrastructure-as-a-Service (IaaS), Platform-as-a-Service (PaaS) and Software-as-a-Service (SaaS). This model of service consumption is extremely suitable for many workloads and cloud computing has become highly successful technology [5]. It allows its users to pay for what they use and also remove the upfront infrastructure cost. Cloud service providers receive resource requests from a number of users through the use of virtualization. It is very essential for cloud providers to operate very efficiently in multiplexing at the scale to remain profitable. The increase in the use of cloud computing has risen to develop massive data centers with very large number of servers. The resource management at this scale is the concerned issue. Scheduling is responsible for arbitrate of resources and is at the center of resource management.

The issue of efficiency at this rate and developing model of consumption of cloud providers needs new approaches and techniques to be applied to the age old problem of scheduling. Virtual machine is the primary unit of scheduling in this model. In this study, we deal with problem of virtual machine scheduling over physical machines. We aim to understand and solve the various aspect of scheduling in cloud environments [6,7]. Specifically, we leverage different fine grained monitoring information to make better scheduling decisions and used learning based approach of scheduling in widely different environments. There is increasing concern over energy consumption by cloud data centers and cloud operators are focusing on energy savings through effective utilization of resources [8,9]. SLAs is also very important for them for the performance of applications running. We propose algorithms which try to minimize the energy consumption in the data center duly maintaining the SLA ensures. The algorithms try to use very less number of physical machines in the data center by dynamically rebalancing the physical machines based on their resource utilization. The algorithms also do an optimization of virtual machines on a physical machine, reducing SLA violations.

For large scale distributed system autonomic management [10,11] is one of the most wanted features and even more important in dynamic infrastructures such as Clouds. This is self-managing such as self-healing, self-regulating, self-protecting, and self-improving. Good work in both academia and industry has been already carried out. Tsai [12] reported an overview of the early efforts in developing Metaheuristic scheduling for autonomic systems for storage management. Computing Grids have benefited

from the application of autonomic models for management of resources and the scheduling of applications [13,14,15]. Solutions for secure Cloud platforms have been proposed in the literature [16,17]. However, existing works are yet to address issues related to recognition of attacks against SaaS with the aim of exploiting elasticity. The basic idea of the proposed algorithm is to leverage the strengths of heuristic algorithms, such as swarm optimization [18], Scheduling Computer and Manufacturing Processes [19], Hadoop map-task scheduling [21,22] and ant colony optimization [20], by integrating them into a single algorithm.

One of the important tasks for cloud provider is scheduling data packets for better achievement and efficient resource pooling along with elasticity. In cloud computing scenarios scheduling algorithm becomes very significant where the cloud service providers have to operate at very much competent to be competitive and take advantage at scale[23,24]. The wide acceptance of cloud computing means data centers with many more machines and the usage model is much different than traditional clusters, like hour boundaries, auction based prices are to name a few. Thus scheduling in cloud data center is more challenging than traditional cluster schedulers. Also, these data center run many different kinds of applications with varying expectations from infrastructure. Resource usage patterns in traditional data centers are have less variance in than the unpredictability faced by cloud data centers.

## PROBLEM ANALYSIS AND DEFINITION

Since Rule based algorithms are straight forward and can be easily implemented, so there is lot of possibility to improve the performance of these algorithms especially in cloud environments. Conventional models and its corresponding algorithms are pretty good for small scale environments, however owing to the advent improvement of computer and related internet technological improvements; there is a huge range to enhancements in these algorithms to get better performance in large scale system. Extreme use of number of servers in the recent period has reduced the usage of traditional scheduling techniques. The resource management at this scale is the concerned as an issue. Scheduling is responsible for arbitrate of resources and is at the center of resource management. The issue of efficiency at this rate and developing model of consumption of cloud providers wants new methodologies and techniques that to be extended to apply to presently available old algorithms of scheduling.

The main objective of the paper is to develop and implement more efficient scheduling algorithm suitable for cloud system. Since parallel and

distributed computing technologies was widely used to improve the diversity performance of computer systems, a number of models and thoughts have been anticipated for different approaches and congenital limitations in diverse eras. Whatever it may be the contemplation it is for; the way to competently use computer resources is a key research matter. Among all of them, scheduling is indispensable in the success of escalating the performance of the computing system. The wide acceptance of cloud computing means data centers with many more machines and the usage model is much different than traditional clusters, like hour boundaries, auction based prices are to name a few. Thus scheduling in cloud data center is more challenging than traditional cluster schedulers.

## PRESENT SYSTEM AND PROPOSED SCHEDULING ALGORITHM

In this research paper, by the simple scheduling problems, we mean problems for which all the solutions can be checked in a reasonable time by using classical exhaustive algorithms running on modern computer systems. In comparison, with the large scale scheduling problems, like the problems for which not all the solutions can be examined in a reasonable time by using the same algorithms running on the same computer systems. These observations make it easy to understand that exhaustive algorithms will take a prohibitive amount of time to check all the candidate solutions for large scheduling problems because the number of candidate solutions is simply way too large to be checked in a reasonable time. As a result, researchers have paid their attention to the development of scheduling algorithms that are efficient and effective, such as heuristics. Workflow is used with the automation of procedures where by files and data are passed between participants according to a defined set of rules to achieve an overall goal.

A workflow management system is the one which manages and executes workflows on computing resources. Workflow Scheduling: It is a kind of global task scheduling as it focuses on mapping and managing the execution of inter-dependent tasks on shared resources that are not directly under its control. The authors classify and review hyper-heuristic approaches into the following four categories: based on the random choice of low level heuristics, greedy and puckish, meta heuristic-based, and those employing learning mechanisms to manage low level heuristics. The hyper heuristics can be used to operate at a higher level of abstraction. Meta heuristic techniques are expensive techniques that require knowledge in problem domain and heuristic technique. Hyper heuristic technique does not require problem

specific knowledge. In order to solve hard computational search problems the hyper heuristic techniques can be used. The hyper heuristic techniques can be operated on the search space of heuristics.

**Proposed System:** The basic idea of the proposed algorithm is to use the diversity detection and improvement detection operators to balance the intensification and diversification in the search of the solutions during the convergence process. The proposed algorithm, called hyper-heuristic scheduling algorithm (HHSA).The parameters max and ni, where max denotes the maximum number of iterations the selected low-level heuristic algorithm is to be run; ni the number of iterations the solutions of the selected low-level heuristic algorithm are not improved. Line 2 reads in the tasks and jobs to be scheduled, i.e., the problem question. Line 3 initializes the population of solutions Z = fz1; z2;:::; zNg, where N is the population size. Online 4, a heuristic algorithm Hi is randomly selected from the candidate pool H = fH1;H2; :::;Hng.

Hyper-heuristics are high level problem independent heuristics that work with any set of problem dependent heuristics and adaptively apply and combine them to solve a specific problem. This could be due to the fact that variants of differential evolution, which we mainly use as basic heuristics due to their competitive performance and simple configuration, strongly depend on the population distribution. Hyper-heuristics might be regarded as a special form of genetic programming, the key intuition underlying research in this area is that, for a given type of problem, there are often a number of straightforward heuristics already in existence that can work well (but perhaps not optimally) for certain sorts of instances of that type of problem. Perhaps it is possible to combine those existing heuristics into some more elaborate algorithm that will work well across a range of problems.

Cloud Computing denotes both the applications delivered as services, the hardware and systems software in the datacenters that provide those services. The services has been refered to as Software as a Service (SaaS). The datacenter hardware and software together combinedly called a Cloud. When a Cloud is of the kind pay-as-you-go manner to the public, then it is a Public Cloud; if the service is being sold then it is a Utility Computing. The term Private Cloud refers to internal datacenters of a business or other organizations, thart are not made available to the public. Thus, Cloud Computing is the combination of SaaS and Utility Computing, but does not comprise Private Clouds. Anyone can be users or providers of Software as a Service, and users or providers of Utility Computing.

**Advantage:**

- Statistical multiplexing is a method that can be used to maximize utilization when compared to a private cloud.
- Cloud computing can also offer services at a cost of a medium-sized datacenter and can even make a good profit.

**Disadvantage:**

- The one important disadvantage is the chance of data loss.
- User can leave the site permanently after facing a poor service; this negative feedback may result in a permanent loss of a portion of the revenue stream.

## WORKFLOW ENGINE:

A Workflow engine is a software service that is used to provide the run-time environment in order to create, maintain and develop workflow instances. The representation of a workflow process is in a form which supports automated manipulation.

Invoked Applications: Interfaces to support interaction with a variety of IT applications

Workflow Client Applications: Interfaces to support interaction with the user interface. Administration and Monitoring: Interfaces to provide system monitoring and metric functions to facilitate the management of composite workflow application environments. It can be seen that scheduling is a functional module of a Workflow Engine, becoming a significant part of workflow management systems.

Workflow is concerned with the automation of procedures whereby files and data are passed between Participants according to a defined set of rules to achieve an overall goal. A workflow management system defines, maintains and develops workflows on computing resources.

## VIRTUAL GRID EXECUTION SYSTEM (VGES):

vgES provides an uniform qualitative resource abstraction over grid and cloud systems. We apply vgES for scheduling a set of deadline sensitive weather forecasting workflows. Specifically, this paper reports on our experiences with:

1. Virtualized reservations for batch queue systems,
2. Coordinated usage of Tera Grid (batch queue),Amazon EC2 (cloud), our own clusters (batch queue) and Eucalyptus(cloud) resources, and

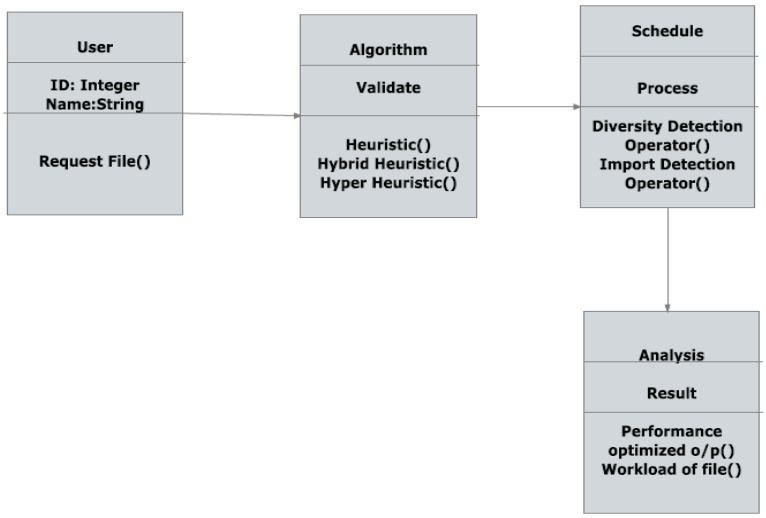
3. Fault tolerance through automated task replication. The combined effect of these techniques was to enable a new workflow planning method to balance the performance, reliability and the cost considerations. These results point toward improved resource selection and execution management support for a variety of e-Science applications over grids and cloud systems.

This paper brings together many of the results from the VGrADS project demonstrating the effectiveness of virtual grids for scheduling LEAD workflows. In the process, it demonstrates a seamless merging of cloud and HPC resources in service of a scientific application. It also applies advanced scheduling techniques for both performance improvement and fault tolerance in a realistic context. LEAD has been run as a distributed application since its inception, but VGrADS methods have opened new capabilities for resource management and adaptation in its execution. This paper details the vgES implementation of virtual grids and their use in fault tolerant workflow planning of workflow sets with time and accuracy constraints. Our experiments show the efficiency of the implementation and the effectiveness of the overall approach.

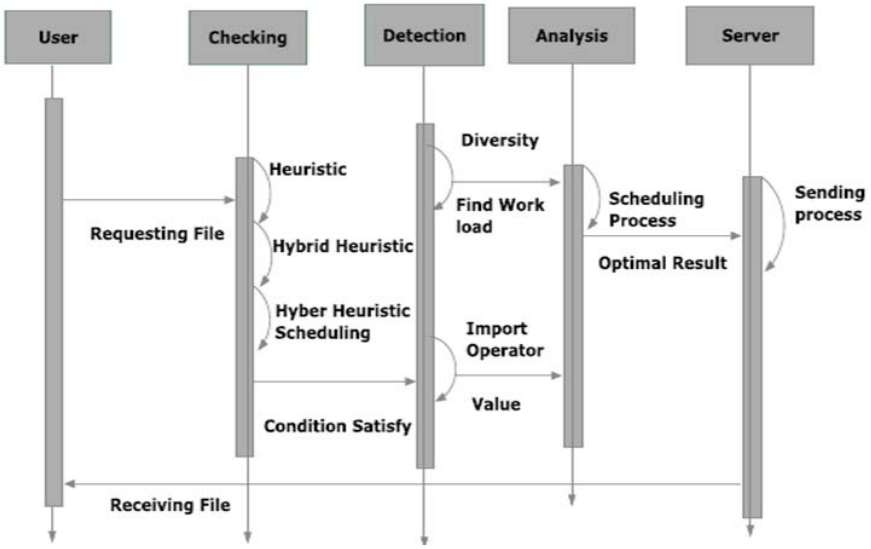
**Modules Description:** A simple random method is used to select the low-level heuristic Hi from the candidate pool H. The diversity detection operator is used by HHSA to decide "when" to change the low-level heuristic algorithm Hi. This mechanism implies that the higher the temperature, the higher the opportunity to escape from the local search space to find better solutions. The timer is fixed at startup and end up mode of an application. We associate with such an event the workload dependent.

**Hyper heuristic Algorithm:** Time-based consists in setting a timer that schedules the time instant when the Scheduled task has to be performed. The timer is fixed at startup mode of an application. Hyper-heuristic algorithms can then maintain a high search diversity to increase the chance of finding better solutions at later iterations while not increasing the computation time. A time-based rejuvenation policy intends to identify the optimal time to rejuvenate with respect to one or more performance indices. The VMM does not degrade, and therefore, it is only necessary to keep memory of the age that was reached at the workload changing point.

**Class Diagram to Scheduling Algorithm**




**Sequence Diagram for Cloud Scheduling Algorithm**

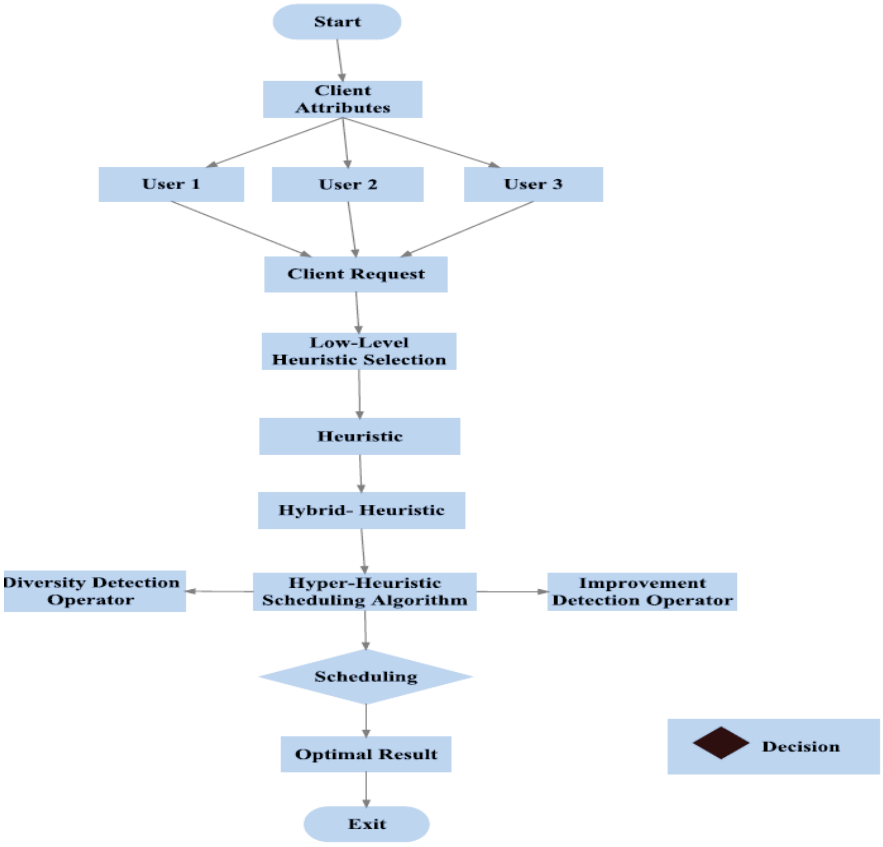


## ALGORITHM DESCRIPTION:

A hyper-heuristic is a heuristic search method that learns to automate, repeatedly by the incorporation of machine learning techniques, in which the process of selecting, combining, and generating or adapting a number of simpler heuristics (or components of such heuristics) to efficiently solve the computational search problems. One of the main motivations for studying the hyper-heuristics is to build systems that which can handle the classes of problems rather than solving just a single problem. There might be multiple heuristics from which one can choose for solving the problem, and then the each heuristic has its own strength and also the weakness.

The idea is to automatically devise algorithms by combining the strength and compensating for the weakness which are known heuristics. In the typical hyper-heuristic framework there consists of the high-level methodologies and a set of low-level heuristics (either constructive or perturbative heuristics). When a problem instance is given, the high-level method selects which low-level heuristic should be applied at any of the given time, based upon the current problem state, or the search stage.

**Scheduling Flow Diagram**




**Primitive model of Server Side Sample Code:**

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using System.Data.SqlClient;
namespace server
{
```

```
public partial class Form2 : Form
{
SqlConnection cn = new SqlConnection("Data Source=SRUJAN\\
SQLEXPRESS;Initial Catalog=schedule;Integrated Security=True");
SqlCommand cmd;
SqlDataReader dr, dr1;
public Form2()
{
InitializeComponent();
}
private void Form2_Load(object sender, EventArgs e)
{
cn.Open();
}
private void button6_Click(object sender, EventArgs e)
{
//+cal1();
//string sta = rsc + "N State";
//textBox6.Text = sta.ToString();
}
private void button4_Click(object sender, EventArgs e)
{
}
string t, t1, t2;
private void button2_Click(object sender, EventArgs e)
{
SqlCommand cmd = new SqlCommand("select * from vm1load", cn);
SqlDataReader dr = cmd.ExecuteReader();
while (dr.Read())
{
t1 = dr[0].ToString();
}
dr.Dispose();
cmd.Dispose();
if (t1 == null)

{
MessageBox.Show("Start sharinf file");
}
else
```

```
{
textBox7.Text = t1.ToString();
}
}
private void button3_Click(object sender, EventArgs e)
{
cmd = new SqlCommand("select * from vm2load", cn);
dr = cmd.ExecuteReader();
while (dr.Read())
{
t2 = dr[0].ToString();
}
dr.Dispose();
cmd.Dispose();
if (t2 == null)
{
MessageBox.Show("Start sharinf file");
}
else
{
textBox1.Text = t2.ToString();
}
}
private void button1_Click(object sender, EventArgs e)
{
//int v=Convert.ToInt32(textBox1.Text);
int v1 = Convert.ToInt32(textBox7.Text);
int v2 = Convert.ToInt32(textBox1.Text);
if (v1 < 100)
{
label1.Visible = true;
label1.Text = " Failure Occured Unable to Load";
}
if (v2 < 100)
{
label2.Visible = true;
label2.Text = "Failure Occured Unable to Load";
}
}
private void pictureBox1_Click(object sender, EventArgs e)
```

```
{
if (textBox7.Text == “” && textBox1.Text == “”)
{
MessageBox.Show(“Start any of Your Server And Proceed”);
}
else
{
Form2 f2 = new Form2();
f2.Show();
}
}
private void button4_Click_1(object sender, EventArgs e)
{
Form4 f4 = new Form4();
f4.Show();
this.Hide();
}
private void label9_Click(object sender, EventArgs e)
{
}
}
}
```

## CONCLUSION

The proposed algorithm uses two detection operators to automatically determine when to change the low level heuristic algorithm and a perturbation operator to fine tune the solutions obtained by each low-level algorithm to further improve the scheduling results in terms of make span. As the simulation results show, the proposed algorithm can not only provide better results than the traditional rule-based scheduling algorithms, it also outperforms the other heuristic scheduling algorithms, in solving the workflow scheduling and Hadoop map-task scheduling problems on cloud computing environments. With the incorporation of genetic programming into hyper-heuristic research, a new level of approaches are found that we have termed ‘heuristics to generate heuristics’. These approaches provide richer heuristic search spaces, and thus the freedom to create new methodologies for solving the underlying combinatorial problems.

In addition, the simulation results show further that the proposed algorithm converges faster than the other heuristic algorithms evaluated in this

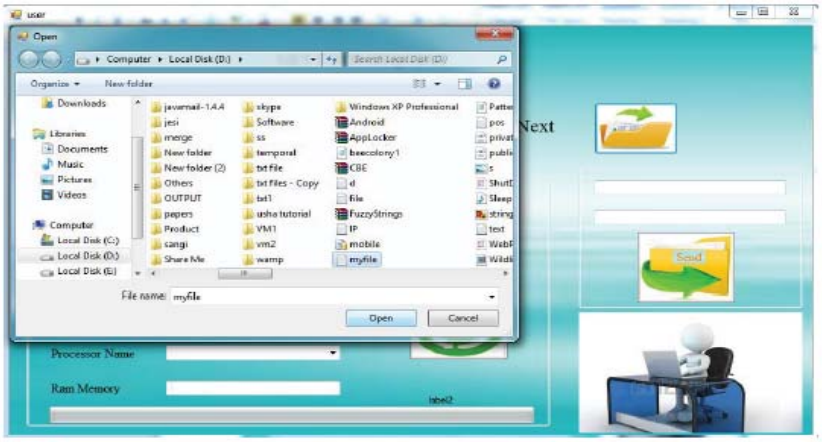
study for most of the datasets. In brief, the basic idea of the proposed hyper heuristic algorithm is to leverage the strengths of all the low level algorithms while not increasing the computation time, by running one and only one low-level algorithm at each iteration. This is fundamentally different from the so-called hybrid heuristic algorithm, which runs more than one low level algorithm for each iteration; thus requiring a much longer computation time.

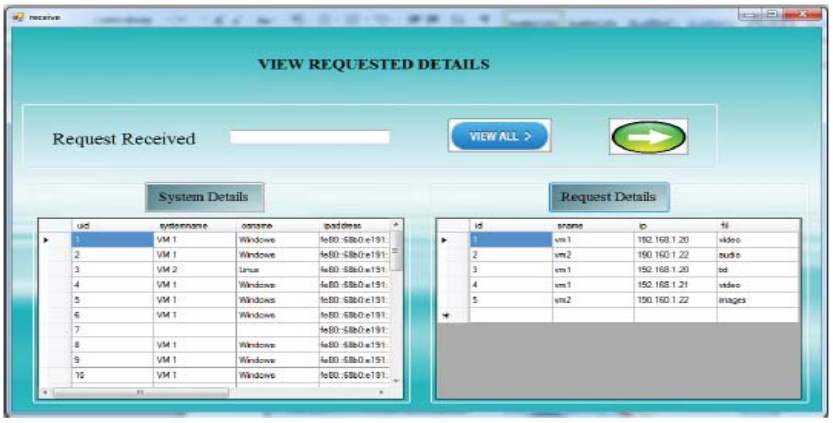
## REFERENCES

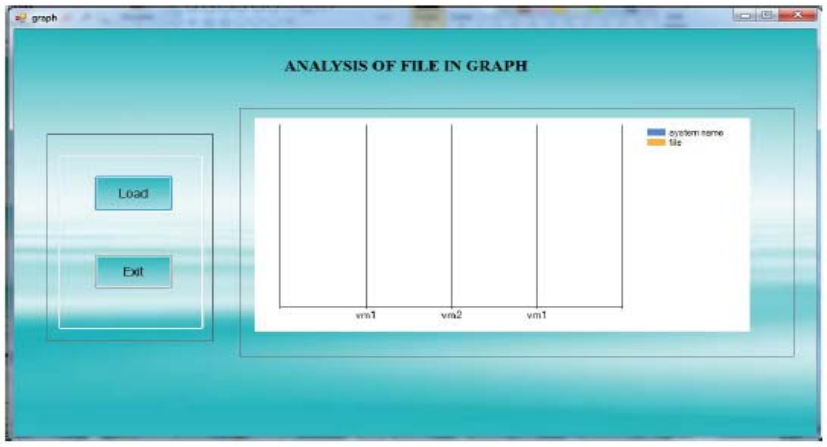

1. **P. Chr´etienne, E. G. Coffman, J. K. Lenstra**, and **Z. Liu, Eds.**, 1995, *Scheduling Theory and Its Applications*. John Wiely & Sons Ltd.
2. **A. Allahverdi, C. T. Ng, T. C. E. Cheng**, and **M. Y. Kovalyov**, 2008, *"A survey of scheduling problems with setup times or costs,"* European Journal of Operational Research, vol. 187, no. 3, pp. 985–1032.
3. M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica, and M. Zaharia, "A view of cloud computing," Communications of the ACM, vol. 53, no. 4, pp. 50–58, 2010.
4. **Reddy, C. S., et al.**, 2016, *"Obtaining Description for Simple Images using Surface Realization Techniques and Natural Language Processing"* Indian Journal of Science and Technology 9.22.
5. **I. Foster, Y. Zhao, I. Raicu**, and **S. Lu**, 2008 , "*Cloud computing and grid computing 360-degree compared"* in Proceedings of the Grid Computing Environments Workshop, pp. 1–10.
6. **Mamatha, E., C. S. Reddy** and **Ramakrishna Prasad**,, 2012, *"Mathematical Modeling of Markovian Queuing Network with Repairs, Breakdown and fixed Buffer"* i-Manager's Journal on Software Engineering 6.3 (2012): 21.
7. **Tsai, Chun-Wei, et al.**, 2014, *"A hyper-heuristic scheduling algorithm for cloud"* IEEE Transactions on Cloud Computing 2.2 (2014): 236-250.
8. **X. Lei, X. Liao, T. Huang, H. Li, and C. Hu,** 2013, *"Outsourcing large matrix inversion computation to a public cloud"* IEEE Transactions on Cloud Computing, vol. 1, no. 1, pp. 78–87
9. **Mamatha, E., C. S. Reddy, and K. R. Prasad**, 2016, *"Antialiased Digital Pixel Plotting for Raster Scan Lines Using Area Evaluation"*, Emerging Research in Computing, Information, Communication and Applications. Springer Singapore, 461-468
10. **J. H. Holland**, 1975, *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence*. University of Michigan Press.
11. **Mamatha et al.**, *An Efficient Line Clipping Algorithm in 2D Space, in press*, International Arab Journal of Information Technology,
12. **C. W. Tsai** and **J. Rodrigues**, *"Metaheuristic scheduling for cloud: A survey",* 2014, IEEE Systems Journal, vol. 8, no. 1, pp. 279–297.
13. **Y. T. J. Leung**, 2004, *Handbook of Scheduling: Algorithms, Models and Performance Analysis*. Chapman & Hall/CRC.
14. **K. M. Elsayed** and **A. K. Khattab**, 2006, *"Channel-aware earliest deadline due fair scheduling for wireless multimedia networks"* Wireless Personal Communications, vol. 38, no. 2, pp. 233–252.
15. **Mamatha, et al.**, 2008, *"Performance evaluation of homogeneous parallel processor system of Markov modeled queue"* i-Manager's Journal on Software Engineering 3.2: 58.
16. **Z. Wu, X. Liu, Z. Ni, D. Yuan** and **Y. Yang**, 2011, *"A market-oriented hierarchical scheduling strategy in cloud workflow systems"* The Journal of Supercomputing, pp. 1–38.
17. **M. R. Garey, D. S. Johnson** and **R. Sethi**, 1976, *"The complexity of flowshop and jobshop scheduling"* Mathematics of Operations Research, vol. 1, no. 2, pp. 117–129.

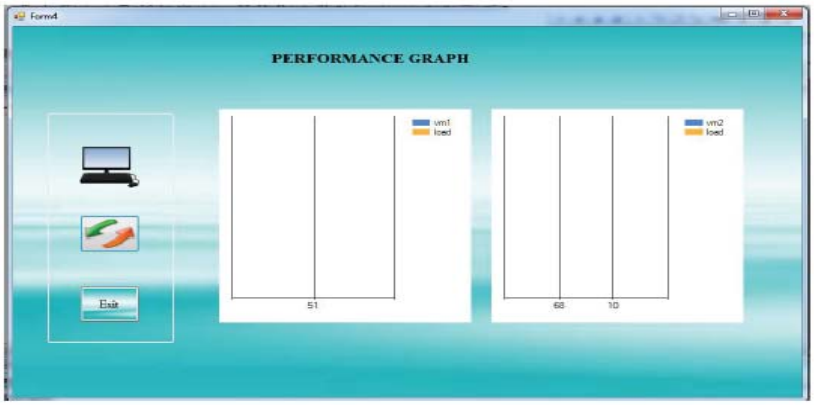
18. J. **Bła˙zewicz, K. H. Ecker, E. Pesch, G. Schmidt** and **J. We¸glarz**, 2001, *Scheduling Computer and Manufacturing Processes*. Springer-Verlag New York, Inc.
19. **D. Shi** and **T. Chen,** 2013, *"Optimal periodic scheduling of sensor networks: A branch and bound approach"* Systems & Control Letters, vol. 62, no. 9, pp. 732–738.
20. **M. Dorigo** and **L. M. Gambardella**, 1997, *"Ant colony system: A cooperative learning approach to the traveling salesman problem"* IEEE Transactions on Evolutionary Computation, vol. 1, no. 1, pp. 53–66.
21. **Apache Hadoop.** [Online]. Available: http://hadoop.apache.org
22. **Y. M. Huang** and **Z. H. Nie**, *"Cloud computing with Linux and Apache Hadoop"* [Online]. Available: http://www.ibm.com/developerworks/aix/library/au-cloud apache.
23. **Mamatha, E., C. S. Reddy** and **S. Krishna Anand**, 2016, *"Focal point computation and homogeneous geometrical transformation for linear curves"* Perspectives in Science
24. **M. Rahman, X. Li,** and **H. Palit**, 2011, "Hybrid heuristic for scheduling data analytics workflow applications in hybrid cloud environment," in Proceedings of the IEEE International Symposium on Parallel and Distributed Processing Workshops, pp. 966–974.

## RESULTS AND DIAGRAMS




graph
ANALYSIS OF FILE IN GRAPH
Load
Exit
system name
file
vm1
vm2
vm1


Form4
PERFORMANCE GRAPH
Exit
vm1
load
51
vm2
load
68
10