Machine Intelligence by Central Pivot Ranges (MICPR): An Optimal Resource Scheduling Strategy for Cloud Services

Uma Maheswara Rao (inkolluchanti@gmail.com)
Koneru Lakshmaiah Education Foundation
JKR Sastry
Koneru Lakshmaiah Education Foundation

Research Article

Keywords: Central Pivot Range (CPR), Infrastructure as a Service (IaaS), Bandwidth Consumption, Load balancing algorithms, Cloud Resource Sensing and scheduling

Posted Date: June 8th, 2022

DOI: https://doi.org/10.21203/rs.3.rs-1632741/v1

Machine Intelligence by Central Pivot Ranges (MICPR): An Optimal Resource Scheduling Strategy for Cloud Services

[1]Uma Maheswara Rao I **[2]Dr. JKR Sastry**
[1]Research Scholar [2]Professor
[1,2]Dept., of Computer Science and Engineering, Koneru Lakshmaiah Education Foundation, Andhra Pradesh, India
[1]inkolluchanti@gmail.com ***[2]drsastry@kluniversity.in***

Abstract: Alongside the increasing demand for cloud computing solutions, the other critical factor in focusing on cloud computing solutions is effective cloud resource sensing and scheduling. In the real-time scenario, there are numerous resource scheduling models available to support effective managing resource scheduling. However, resource sensing and scheduling challenges remains in the increasing complexities of background processing in the cloud servers. Thus, in this manuscript, a machine intelligence model that provides significant insights into how the load balance can be sensed and handled using the sever performance's critical metrics is proposed. The model is an automated machine intelligence (MI) approach that uses pivot range (CPR), support, and resistance models for estimating if the load of the system as per the metrics chosen is running heavy load or normal. Accordingly, the decision can be triggered automatically by MICPR to choose the fundamental resource sensing and scheduling algorithms. The experimental study of the model refers to potential ways in which the system can be feasible, and if implemented on a large scale, it can help reduce the backend load on the cloud servers.

Key Word: *Central Pivot Range (CPR), Infrastructure as a Service (IaaS), Bandwidth Consumption, Load balancing algorithms, Cloud Resource Sensing and scheduling.*

1 Introduction

Cloud computing has become an integral part of today's fast computing solutions. In the initial days of cloud computing models, numerous studies focused on the implications, data security issues, and loss of controls as the critical challenges in cloud computing. However, the scope of performance, ease of access, and reduced cost of IT infrastructure management has led to the paradigm shift in how organizations prefer the cloud computing models [1].

Today, across the industry verticals, and the kind of enterprise solutions adapted, there is heavy reliance on cloud computing solutions. From simple email servers to complex enterprise applications, social media applications, e-governance systems, and financial work-flow processing applications, there are scores of solutions reliant on the cloud computing models for managing effective processing, work-flow, and security management. It is imperative from the emerging security developments in the cloud computing models that across SaaS, IaaS, and PaaS classifications, the scope of cloud computing security is strengthened, and the cloud computing phenomenon is emerging high [1].

Alongside the increasing demand for cloud computing solutions, the other critical factor in cloud computing solutions is effective cloud resource scheduling. The fundamental purpose of developing the cloud computing model is about increasing the flexibility of IT resource management, reduce the risks of severe downtime conditions, increase the speed of response to the host systems, in addition to anytime anywhere access. Unlike the traditional models wherein

the work-flow's back-up, systems are managed over back-up drives or back-up servers. The cloud solutions perform multi-tasking in contemporary models [2].

Also, in the IaaS (Infrastructure as a Service), there are scores of applications from various organizations being deployed into private and public cloud services. With numerous small- and large-scale applications being launched for cloud computing, it is imperative that the servers' resource capability have to be managed effectively. Thus, there is a need for a more robust set of solutions that can improve the overall resource scheduling process [1], [2].

In the real-time scenario, there are numerous resource scheduling models adapted, which can help in effective ways of managing the resource scheduling. Some of the resource scheduling models are based on conventional solutions, and certain models are based on machine learning solutions and other contemporary models. One of the complexities encountered with the existing set of resource scheduling solutions is the load on the servers for the backend applications like the resource scheduling solutions.

1.1 Problem Statement

Many of the earlier proposed models cover a distinct set of resource scheduling issues. Some of the common areas wherein significant models were proposed in the cloud resource scheduling domain are:

- Load balancing algorithms.
- Power and Fault Tolerance based scheduling models.
- Work-flow processing reliability models
- Network capability management solution
- Network and server security enhancement during the resource scheduling

For a rationale view, in a standard cloud computing environment, there is a need for the resource management teams to ensure the server capacity utilized effectively, the security and reliability in the work-flow and task processing is handled well, and to ensure uninterrupted service quality to the end-users. In terms of achieving such a picture-perfect execution, it is of paramount importance that the models' processes must be more robust and help in having sustainable resource management conditions [3].

For instance, considering the fair term requirement of all the above-said factors in a resource scheduling algorithm, if multiple sets of algorithms are to be run in the background, such sequential computing takes a certain load on the servers. There is a need for a multi-layered approach to curtail such load factors, wherein the cloud servers have the systems deployed but activated based on the as-is essential basis [3].

1.2 Case Scenario

For instance, when an IaaS server has some thirty-odd large-scale applications from fintech companies deployed in a server network, one of the critical challenges for the server teams is managing the load for all the applications effectively the QoS. However, multiple conditions might lead to overbalancing over the severs. Following are some of the possibilities in general based on the usage of the cloud servers [4].

Firstly, in a generic scenario, the fintech applications might encounter huge work-flow processing tasks during the first week of the month; if so, the admin's task is to ensure load for

various applications being managed from multiple servers integrated to the central server. Thus, the need for an application to process such requirements is different.

Similarly, in the DDoS attack kind of conditions, when a specific server is being affected, there is a need to switch the services to alternate servers. Thus, different kinds of applications need to be deployed to handle the requirements. There could be more such scenarios wherein, depending on the requirement, the necessity of deploying a solution or algorithm model could differ [4].

Hence, if there is an internal ecosystem model followed, wherein the multiple set of models or algorithms for addressing the requirements are deployed and depending on the context of the solutions can be activated, it shall help in managing the optimal ways of load balancing over the central servers, and the teams monitoring the application process [4], [5].

1.3 Outline of the Proposed Model

To overcome the limitations above in the cloud servers' current arrangements, a simple and precise solution that can be resourceful to work as the top-layer process is essential. The solution should work as a surface system or a privy system that monitors the cloud server performances based on various metrics and provides insights into the team to develop a comprehensive system.

Thus, focusing on the statistical inference solutions, this study attempts a novel model wherein the surface system is lighter on the servers and can help the teams make informed decisions towards managing the cloud resource scheduling management. One of the proposed solution's fundamental objectives is to ensure that the data integral to the cloud servers' performance should be considered in developing a comprehensive solution. By focusing on utilizing the existing information from the system, the scope of minimizing the additional application load over the system can be managed resourcefully.

Followed by the outline discussed in the model, the related work in terms of literature available on the various gaps in the resource scheduling models shall be discussed in the next section. Accordingly, the proposed application and its experimental case scenario analysis are presented for detailed review in the other section.

2 Related Work

In [6], the study focuses on the conditions wherein the resources in a cloud computing environment must target cloud resource scheduling conditions. The principal policy applied in the model is to ensure the right kind of classification of the applications in the IaaS environment into high, or low, or average load factors based on the hidden Markov model. Accordingly, a suitable load balancing application shall be deployed for the implementation, which will help in effective load balancing for the resources. However, one of the key challenges observed in the model is the basis of classification, wherein only the Markov model is adapted.

Markov model's structure is to ensure the presumption about every stage has a certain timeline and load. While the model can be resourceful in a conventional scenario, in the dynamic applications environment, wherein the session tasks have more volatile conditions, adapting the Markov model can be complex. However, the report's key takeaway is the rationale in choosing the surface layer approach as a reasonable condition to classify the resource utilization requirements.

In [7], a comprehensive review of the literature over the cloud resource scheduling algorithms is proposed, wherein the key factors impacting the resource schedules are evident. For

instance, information on how the metrics are used, the processes, and the different algorithms adapted. Such a detailed classification of the literature supports a one-stop review wherein some critical metrics like the importance of bandwidth, session tasks volumes, understanding the load factors, and other such dynamics are imperative. The study's key insight is about identifying the processes integral to managing the cloud resource scheduling and how effectively the key attributes used can support the ineffective performance of the model.

The authors of the study [8] proposed a cloud resource scheduling model in the private cloud computing environment. Targeting the need for effective scheduling, the model proposes using job tasks in hand and resource availability as the two key dimensions for managing the scheduling dimension. The study claims a good outcome from the simulated experimental study of the proposed model. A key insight from the model is about focusing on the category of tasks and scheduling algorithm development that can account for resource availability. However, one of the key factors to be considered is about optimal ways of using the cloud resources available and manage the resource juggling during the underutilization capacities.

In the study [9], a combination of machine learning models and the statistical solution in UPPAL-SMC is adapted for the resource scheduling process. The model's emphasis is about adapting the complex QoS queries for the resource allocation instances in terms of variations, enabling the parameters tuning for improving the overall QoS, alongside handling the process of quick optimization of overall workflow QoS under customized requirements and resource variation conditions. The model proposes implementing the statistical scope of implementing the resource scheduling process for cloud computing. More specifically, the model targets the execution of QoS metrics as a critical factor.

A literature review study [10] has covered the comparative analysis of the cloud resource scheduling process across various industry verticals and segments. Fundamentally the cloud computing environment of distinct sectors might be having various requirements. For instance, in some sectors, load management could be critical, and in some, the response rate and processing abilities could be complex factors. Thus, there is a need for the organization to have adequate inputs in place for choosing the right kind of cloud resource scheduling models that can help in improvising the overall process outcome. The study has effectively classified the distinct set of challenges and requirements for various cloud computing environments. More specifically, the study has covered the scope of issues in the cloud manufacturing process.

Another study [11] has proposed the TSS model (Task Scheduling System) wherein for each of the tasks, there is a clear and systematic approach for resource scheduling based on the general distribution. The underlying principle followed for the task scheduling module is to adapt the weighted sum of make-span and flowtime as the objective metrics and accordingly deploying the ACO (Ant Colony Optimization) and GA (Genetic Algorithm) for addressing the issue of task scheduling. Simulation results from the study are used for advocating the efficacy of the model. Adapting the evolutionary computing models is a pragmatic approach for resource scheduling. However, the constraints in understanding the impact levels and the additional load on the server are the other key aspects to consider in the implementation.

Study [12] relies on a hybrid computational algorithm model wherein the first level categorization and allocation model is based on the heuristic algorithm like GA and the bacterial foraging algorithms. The process followed in the model uses the first set algorithm to minimize the make span. The second algorithm is used to reduce energy consumption and economic and

ecological factors. Thus, the model considers a diverse range of issues that affect the cloud servers' performance and accordingly deploys a robust solution.

Based on the "Deep Q-network (DQN) algorithm", Zhiping Peng et al.[13] suggested an online resource scheduling system. By modifying the proportion of the benefit of two optimization targets, the framework may make a trade-off between the two optimization goals of energy usage and task make-span. Ramamoorthy s et al., [14], suggested a "multi constraint aware multiple objective resource scheduling optimization (MCAMO)" technique that sought to deal with many objectives by imposing multi restraints during resource scheduling in infrastructural cloud services. The suggested method is unique in that it addresses the limits of the given workflows while also achieving the goals of efficient resource allocation. Many of the limits evincied in previous contributions have been handled in these contemporary models [13] [14]. These models, on the other hand, are more focused on infrastructure costs and financial concerns. As a result, it is clear that these contributions frequently fall short of achieving ideal resource usage in terms of minimum make-span, maximum resource utilisation, and least latency.

In a summary of the limited review of literature, it is evident that there are scores of cloud resource scheduling models that could be more efficient in handling the QoS problems, optimal utilization of the resources, customization of the algorithms, or combination of models, which can help in improving the overall development and solution. However, there are limited studies that have relied extremely on the conditions of statistical approaches or leveraging some of the statistical estimations that can work as simple and effective screeners for the cloud resource scheduling process.

3 Machine Intelligence based Resource Scheduling by Pivot Ranges

Machine Intelligence by Central Pivot Ranges (MICPR) is a new and simple approach proposed resource sensing and scheduling in cloud services. The system is termed as surface application, since the model can be layered as a support system over any of the existing algorithms or models. The purpose of the defined model is to reduce the load of the launching of appropriate cloud scheduling models based on the pivot range of load occupancy observed on the cloud servers used for the cloud network.

The underlying principle used in the model is the application of central pivot range and pivot range points, which are used as the scale of the metrics for understanding the scheduling load and balance accordingly. The pivot points are a statistical solution used in prediction of the scenario in next vertical of the work-flows in the considered context. For instance, the critical time for completing a task or time frame of a session that can change the resulting outcome is seen as pivot points.

Technically, the calculation of pivot points is based on a certain dataset, and the sequential set of support and Resistance depicting the datasets are also imperative in the model. The critical reason behind the application of support and resistance calculations is to understand where the specific task can encounter Resistance or envisage support.

In terminology to the cloud-computing resource scheduling, every server based on its capability model, session tasks, time consumed for managing the load for various applications, and other such metrics, if the pivot points of performance, possible Resistance, and the minimum support conditions are estimated, it shall help the server administration in taking the decisions for altering the load management decisions.

Also, in terms of understanding the dynamics of lagging patterns in load management, the VWA (Volume Weighted Average) is used as an indicator for confirmatory analysis. The purpose of using the volume-weighted average is to understand if the current load capacity is within the volume region to handle or there is a need for alternate systems to handle the load capacity as a confirmation model.

3.1 Metrics and Assessment Factors

3.1.1 Pivot Points

Pivot points refer to the chartists' significant levels to determine the directional movement, potential support, and resistance levels for a scribe movement. The estimation of the pivot points is based on work-flows of the prior periods, high, low, and close (or mean) for a metric, which helps understand future support and resistance levels.

The pivot estimation models are classified in the statistical models as potential support or resistance levels. While there are a distinct set of pivot point models like the Standard Pivot Points, Demark Pivot Points, Camarilla Pivot Points, Fibonacci Pivot Points, and Central Pivot Points, in this manuscript, focusing on the minimalistic approach required, the emphasis is on the Central Pivot Point, which is termed as CPR (Central Pivot Range).

3.1.2 Volume Weighted Average

In line with the proposed scheduling model, the Volume Weighted Average (VWA) has used as a potential system across the metrics estimation. In the case of the VWA, the real-time analysis of the volume-weighted averages refers to pragmatic conditions in the metric's ongoing action.

While the combination of lead and lag indicators is used for analysis, one can predict the possible action, and the other can confirm the emerging action. Thus, the combination of the lead and lag approach is used for estimation.

3.1.3 Support and Resistance

The terms support and Resistance are the possible points for a metric, estimated using a defined mathematical model. Once the support and resistance models are identified for a system, it is much easier to estimate the possible support and reversal areas.

In terminology to the cloud computing scenario, when a cloud server load reaches the resistance levels, the administration needs to have a close watch to understand if the load has to be shifted to alternate servers. By doing so, the performance of the server can be handled near its pivot point. As vice versa, if the systems in their current position are managing at the support points, the load levels for the respective service can be increased to maintain balance among the network servers.

The proposed model's fundamental structure is to understand if a server's load in terms of a metric estimation is assessed for its position. If the unit value for a metric is within the region of R1(resistance-1) to S1 (support-1), the load balance is in control. Else, the necessary action can be initiated by the server administration team.

3.1.4 Metrics

The metrics chosen for the process of adopting the cloud resource scheduling are multi-dimensional. In pragmatic working conditions, the need for load balancing is different based on the cloud computing environment and its functions. For instance, in some cases, the scope of response rate is important, and in few conditions, the scope of processing unit bandwidth

consumption is a key factor. While in certain cloud computing models, the scope is more about make-span management.

Thus, in the context of the proposed model, various dimensions can be used for analysis depending on the context and requirement. In complex cloud environment conditions, the model facilitates the application of the system for multi-dimensional analysis. And the sever administrators can have a view of the various dynamics important for decision making.

Some of the common metrics to facilitate the analysis are

- Bandwidth Required per Make-span (brm): This metric indicates the volume weighted average of the bandwidth consumed per each make span of the each time frame. The said metric shall be estimated as follows

  For each make-span m, for each resource $cR = \{r_j \exists j = 1,2,3...,|cR|\}$, find the volume weighted average of the bandwidth consumed as,

  For each work-flow $wF_m = \{w_k \exists k = 1,2,3,...,w_{|wF_m|}\}$ of the given make-span, find the product of bandwidth consumed per unit of time and number of time units survived by corresponding work-flow. Further aggregate the resultant values of all work-flows, and fraction the resultant aggregate value by the total number of time units tU_m survived the corresponding make-span.

  $\underset{j=1}{\overset{|cR|}{\forall}} \{r_j \exists r_j \in cR\}$ Begin // for each cloud resource

  $$rwbt_{(r_j \to m)} = \frac{\sum_{k=1}^{|wF_m|} \{bcu * tU_k\}}{tU_m}$$

  End

- Tasks Rendered per Resource (trr): This metric is the Weighted Average of total tasks made for each make span to the target resource. The said metric shall be estimated as follows

  For each make-span m, find the volume weighted average of the total tasks rendered towards each resource $cR = \{r_j \exists j = 1,2,3...,|cR|\}$, which is as follows

  For each of the given make-span m, for each resource $cR = \{r_j \exists j = 1,2,3...,|cR|\}$, find the sum of tasks rendered under all work-flows, and find the fraction of resultant sum for total number of work-flows rendered in corresponding make-span, as stated in following notation.

  $\underset{j=1}{\overset{|cR|}{\forall}} \{r_j \exists r_j \in cR\}$ Begin // for each cloud resource

  $$trr_{(r_j \to m)} = \frac{\sum_{k=1}^{|wF_m|} \{|sT_k^j|\}}{|wF_m|}$$ //denotes the weighted average of the overall scheduled tasks sT of the make-span m to resource r_j

wF_m : Denotes the set of workflows scheduled in make-span m

sT_k^j : Denotes the addressed tasks of the k^{th} work-flow to j^{th} resource

End

- Tasks addressed per Resource (tar) : This metric is the Volume Weighted Average of total tasks addressed by the target resource per each make span. The said metric shall be estimated as follows:

  For each make-span m, for each resource $cR = \left\{ r_j \exists j = 1,2,3...,|cR| \right\}$, find the volume weighted average of the total tasks addressed, which is as follows.

  Find the sum of tasks addressed under all work-flows wF_m, and find the fraction of resultant sum for total number of work-flows rendered in corresponding make-span.

  $\underset{j=1}{\overset{|cR|}{\forall}} \left\{ r_j \exists r_j \in cR \right\}$ Begin // for each cloud resource

  $$tar_{(r_j \to m)} = \frac{\sum_{k=1}^{|wF_m|} \left\{ \left| aT_k^j \right| \right\}}{|wF_m|}$$ //denotes the weighted average of the overall scheduled tasks sT of the make-span m to resource r_j

  wF_m : Denotes the set of workflows scheduled in make-span m

  aT_k^j : Denotes the addressed tasks of the k^{th} work-flow to j^{th} resource

  End

- Resource to Workflow Binding Time $(rwbt)$: This metric is the Weighted Average of the time bonded between work-flows and the target resource per each make span. The said metric shall be estimated as follows

  For each make-span m, for each resource $cR = \left\{ r_j \exists j = 1,2,3...,|cR| \right\}$, find the weighted average of the time units, the target resource bonded to each work-flow $\left\{ w \exists w \in wF_m \right\}$ of the given make-span m, which is as follows

  For each of the given make-span, for each resource, find the sum of the time units that resource bonded to the workflows, and find the fraction of resultant sum for total number of work-flows rendered in corresponding make-span.

  $\underset{j=1}{\overset{|cR|}{\forall}} \left\{ r_j \exists r_j \in cR \right\}$ Begin // for each cloud resource

  $$rwbt_{(r_j \to m)} = \frac{\sum_{k=1}^{|wF_m|} \left\{ tU_k \right\}}{\left| wF_m \right|}$$

  End

- Make Span Execution Tenure (met) : This metric indicates the total time taken to complete the tasks of the work-flows engaged in target make-span. The said metric shall be estimated

as follows, for each Make-span, aggregate the time units taken to complete by each work-flow

$\forall_{j=1}^{|cR|}\{r_j \exists r_j \in cR\}$ Begin // for each cloud resource

$$met_{(r_j \to m)} = \sum_{k=1}^{|wF_m|}\sum_{l=1}^{|w_k|}\{tU_l\}$$

End

- Resource to Workflow Binding Latency $(rwbl)$: This metric denotes the total latency in time bonded by resource to work-flows of the target make-span. The said metric value is a weighted average of the latency between aggregate of the time units taken to complete each task of the target work flow, and total time units the resource and target workflow are coupled, which shall be estimated as follows:

For each make-span, for each work-flow, the absolute difference between time units spent by the corresponding workflow and the aggregate of the time units spent for all the tasks listed in corresponding workflow.

Further, shall find the mean of the latencies observed for all the workflows listed in the given make-span, which denotes the Resource to Work-flow Binding Latency.

$\forall_{j=1}^{|cR|}\{r_j \exists r_j \in cR\}$ Begin // for each cloud resource

$$rwbl_{(r_j \to m)} = \frac{\sum_{k=1}^{|wF_m|}\left\{tU_k - \sum_{l=1}^{|w_k|}\{tU_l\}\right\}}{|wF_m|}$$

End

3.1.5 Time Frame

The other critical factor for observation in the model is the timeframe used for addressing the analysis. In the context of proposed scheduling strategy, the time frame τ is a coefficient annotated in the context of target domain of cloud services.

Simultaneously, the pivot points and CPR are used based on the metric values of the previous time frame's high, low, and mean values.

Whereas for the VWA, the timeframe can be flexible as per the user requirement. This will enable the detailed view of weighted average movement over a scenario and accordingly make informed decisions.

3.1.6 Central Pivot Range (CPR)

Central Pivot Range is a universal statistical computation that comprises 3 level calculation, a central pivot point, top central levels denoted as TC, and the bottom central levels denoted as BC .

The estimation of CPR is calculated as follows.

$$TC = (Pivot - BC) + Pivot$$

$$Pivot = (High + Low + Close)/3$$

$$BC = (High + Low)/2 \text{ [15]}$$

Let {

Mv The chosen metric ($bandwidth$ / $session\ time$ / $make\ span$)

- **Level-1 Estimating** CPR

{

CPR Is estimated using the values of the metric discovered from the make-spans of the previous time-frame.

- Mv Is the Metric Value.
- Tc be the Top Central Value
- Bc be the Bottom Central value
- P Be the pivot value.

Identify the $Mv - Cv, Mv - Hd, Mv - Ld$

$$Pivot = (High + Low + Close)/3\{High = Hd, Low = Ld, Close = Cd\}$$

$$TC = (Pivot - BC) + Pivot$$

$$BC = (High + Low)/2 \text{ [15]}$$

The CPR estimation is an inclusive line frame of Tc , P , and Bc in sequential order.

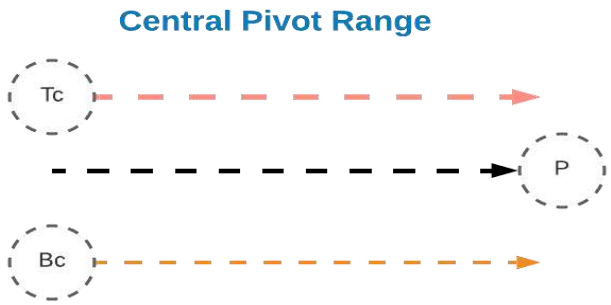


Figure 1: The formation of the CPR range

The formation of the CPR range shall be as Figure 1. However, depending on the volumes and the range of operation for the metric on the previous day, the wide or narrow, or moderate CPR range shall form.

}

- **Level-2 Estimating Support and Resistance Levels**

$R1 = (P * 2) - Low$ (Resistance-1)

$R2 = P + (High - Low)$ (Resistance -2)

$S1 = (P * 2) = High$ (Support-1)

$S2 = P - (High - Low)$ (Support-2)

Technically, $R2 > R1$ and the $S1 > S2$, overall, the times. The CPR range shall be in Figure 2 $R1$ and $S1$.

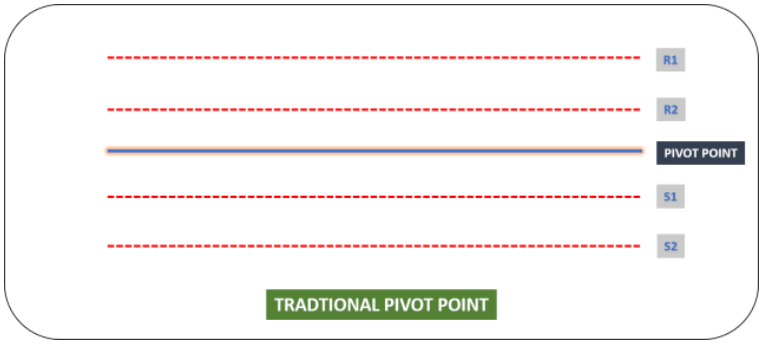


Figure 2: The CPR range Traditional PIVOT Point

3.1.7 Depiction of Pivot Ranges Calculation

This process considers the set of make-spans rendered in previous time-frame τ

Further, discovers values of each metric (i). Bandwidth Required per Make-span (brm), (ii). Tasks Rendered per Resource (trr), (iii). Tasks addressed per Resource (tar), (iv). Resource to Workflow Binding Time $(rwbt)$, (v). Make Span Execution Tenure (met), and (vi). Resource to Workflow Binding Latency $(rwbl)$ of the each make-span of the previous time-frame τ.

Further, for each metric, the discovered values from all make-spans shall be collected as a vector fv, and invokes the method $pivotRanges(fv)$ that receives the vector fv as argument. The method $pivotRanges(fv)$ performs the following steps

$pivotRanges(fv)$	Begin //finds pivot ranges fc of each feature

//Discovers the mean of the vector fv values//

$$\mu = \frac{1}{|fv|}\sum_{i=0}^{\|fv|}\{e_i \exists e_i \in fv\}$$ // average of the values listed in vector fv

//Discovers the deviation of the vector fv values//

$$\delta = \frac{1}{|fv|}\sum_{i=0}^{|fv|}\left\{\sqrt{(\mu - e_i)^2} \exists e_i \in fv\right\}$$ //deviation of the values listed in vector fv

Finds the pivot from high, low, mean, and deviation of the vector fv

$$P = \frac{1}{3}(e_{\min} + e_{\max} + \mu) - \delta$$ //the notation P denotes the central-pivot of the values listed in vector fv,

//Discovers the lower resistance of the vector fv values//

$R_{\min} = \{\log(x) \exists x = 100\} - \{e_{\min} \exists e_{\min} \in fv\}$ //lower resistance $R_{\min}$ of the central-pivot P

// Discovers the higher resistance of the vector *fv* values//

$$R_{\max} = \left\{ \begin{matrix} \{\log(x) \exists x = 10\} + \\ \{e_{\max} - e_{\min} \exists [e_{\min}, e_{\max}] \in fv\} \end{matrix} \right\}$$ //higher resistance $R_{\max}$ of the central-pivot P

// Discovers the min pivot support of the vector *fv* values //

$S_{\min} = \{\log_P(x) \exists p = 10, x = 100\} - \{e_{\max} \exists e_{\max} \in mV\}$ //lower support values $S_{\min}$ of the central-pivot P 0

//Discovers the max pivot support of the vector *fv* values//

$$S_{\max} = \left\{ \begin{matrix} \{\log(x) \exists x = 10\} - \\ \{e_{\max} - e_{\min} \exists [e_{\min}, e_{\max}] \in mV\} \end{matrix} \right\}$$ //higher support value $S_{\max}$ of the central-pivot P

$smc = \{R_{\min}, R_{\max}, S_{\min}, S_{\max}\}$

return smc

End

3.1.8 Decision Making about Resource Scheduling

For a chosen resource $cR = \{r_j \exists j = 1, 2, 3..., |cR|\}$ of the cloud, for a chosen metric $\{m \exists m \in \{brm, trr, tar, rwbt, met, rwbl,\}\}$ of the scheduling process,

The resource available state categorized under Likert scale [16], which is as follows

---

$\forall \{m \exists m \in \{brm, trr, tar, rwbt, met, rwbl,\}\}$

$$lv = \left\{ \begin{matrix} 4 \exists mv < S_{min} \; else \\ 3 \exists mv < S_{max} \; else \\ 2 \exists mv < R_{min} \; else \\ 1 \exists mv < R_{max} \; else \\ 0 \exists mv > R_{\max} \end{matrix} \right\}$$

If the metric value *mv* of the metric *m* is less than lower support, then likert value shall be 4

If the metric value *mv* of the metric *m* is less than higher support, then likert value shall be 3

If the metric value *mv* of the metric *m* is less than lower range, then likert value shall be 2

If the metric value *mv* of the metric *m* is less than higher range, then likert value shall be 1

If the metric value *mv* of the metric *m* is greater than higher range, then the likert value is 0

After discovering the Likert values of each metric, the scheduling state of each resource can be estimated by the product of the likert values of the stated metrics of the corresponding resource.

$\forall_{j=1}^{|cR|} \{r_j \exists r_j \in cR\}$ Begin // for each cloud resource

$s = 1$

$\forall \{m \exists m \in \{brm, trr, tar, rwbt, met, rwbl,\}\}$ Begin

$s = s * lv_f$ //product of Likert values of the resource r_j of different metrics

End

$sr_j = s$ // available status of the resource r_j is estimated, which is denoted by the notation sr_j

End

Further, the resources are prioritized to schedule in descending order of their available status. The resource having zero as available status denotes that corresponding resource is not fit to schedule under one or more scheduling metrics stated, hence load shift is recommended for that resources having available status as 0

4 Case Study

For the proposed experimental analysis, using the dataset of bandwidth values and the Volume of session tasks received, the information is computed into a simulated study. Accordingly, the spreadsheet analysis is carried out using the information imperative.

Based on the data estimation method portrayed in the model, the estimations of a pivot, Resistance and support information is calculated for various set of data related to bandwidth and session tasks. For the Mv values, the bandwidth consumption is considered as the key to estimating the load factor.

In calculating the VWA for the metric value Mv, the session tasks received in each timeframe (here considered as a day volume) are used for estimation. Based on the data, the current Mv value is estimated to understand huge variation in the model.

The data is generated for 47 days of datasets, and a random basis selection of 13 days dataset is chosen for verification of the model.

Table 1: the calculation of CPR, Resistance, and support levels

Day	Pivot Point	BC	TC	R1	R2	S1	S2	Mv
1	545.3	544.5	546.2	595.7	644.3	496.7	446.3	752.0
2	612.3	628.0	596.7	806.7	1032.3	386.7	192.3	565.0
3	736.3	652.5	820.2	1020.7	1137.3	619.7	335.3	630.0
4	602.0	567.0	637.0	703.0	734.0	571.0	470.0	705.0
5	691.0	618.0	764.0	1002.0	1167.0	526.0	215.0	846.0
6	604.3	593.0	615.7	661.7	696.3	569.7	512.3	738.0
7	662.3	648.5	676.2	801.7	913.3	550.7	411.3	618.0
8	660.0	620.0	700.0	858.0	976.0	542.0	344.0	664.0

9	593.0	557.5	628.5	804.0	944.0	453.0	242.0	621.0
10	717.3	671.5	763.2	937.7	1066.3	588.7	368.3	869.0
11	509.7	497.0	522.3	588.3	641.7	456.3	377.7	858.0
12	549.0	530.0	568.0	677.0	767.0	459.0	331.0	876.0
13	698.7	701.0	696.3	904.3	1114.7	488.3	282.7	794.0

The Table 1 refers to the calculation of CPR, Resistance, and support levels. The unit column VWA represents the Volume weighted average for the thirteen days, based on the simulation datasets.

The following table indicates the random selection of 13 days for the decision making from the level-1 stage, wherein the decision is imperative in the following Table 2.

Table 2: Indicates the random selection of 13 days for the decision making from the level-1 stage

Day	**Pivot Point**	**R1**	**R2**	**S1**	**S2**	**Mv**	**Decision**
1	545.3	595.7	644.3	496.7	446.3	752.0	**Load Shift**
2	612.3	806.7	1032.3	386.7	192.3	565.0	Load Maintain
3	736.3	1020.7	1137.3	619.7	335.3	630.0	Maintain
4	602.0	703.0	734.0	571.0	470.0	705.0	Load Monitor
5	691.0	1002.0	1167.0	526.0	215.0	846.0	Load Monitor
6	604.3	661.7	696.3	569.7	512.3	738.0	**Load Shift**
7	662.3	801.7	913.3	550.7	411.3	618.0	Load Maintain
8	660.0	858.0	976.0	542.0	344.0	664.0	Load Maintain
9	593.0	804.0	944.0	453.0	242.0	621.0	Load Maintain
10	717.3	937.7	1066.3	588.7	368.3	869.0	Load Maintain
11	509.7	588.3	641.7	456.3	377.7	858.0	**Load Shift**
12	549.0	677.0	767.0	459.0	331.0	876.0	**Load Shift**
13	698.7	904.3	1114.7	488.3	282.7	794.0	Load Maintain

The symbolic representation of the load factor decision is depicted in the graph below.

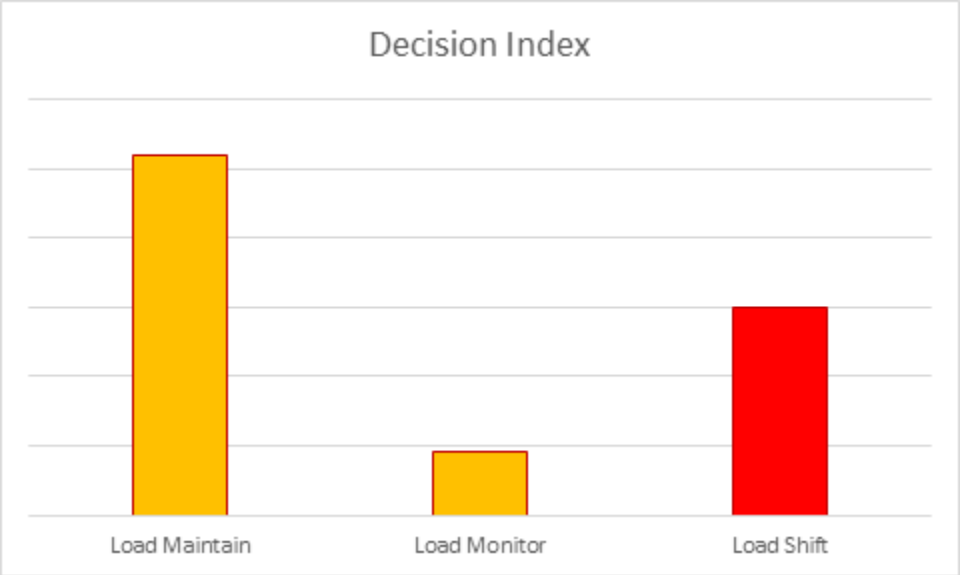


Figure 3: the load maintain condition is depicted, and in certain conditions, the Load Shift is reflected.

From the graph depicted figure 3, it is evident that in most instances, the load maintain condition is depicted, and in certain conditions, the Load Shift is reflected.

By adopting such a simple symbolic representation of the model based on the possible conditions of load shift required, the different verification processes of VWA analysis shall be conducted for verification. Among the 13 days estimations, four days have shown the need for server load shift.

Accordingly, the VWA is estimated for the process. Accordingly, the load conditions for the period shall be estimated, and the conditional decisions. Based on the data analysis, among the four critical slots identified, three of them required the load shift in certainty, and in one case, the process of close monitoring can still be deployed.

In a significant development, when the admin teams can understand the resource scheduling requirement at surface level, the necessary resource scheduling algorithms could be deployed for choosing which set of applications needs resource schedule balancing and accordingly manage the services.

5 Experimental Study and Performance Analysis

This section describes the setting in which simulations were run, followed by a discussion of the simulation outcomes.

5.1 Experimental setup

We consider data processing tasks from a variety of Workflow Sources in order to run the simulations. The experiment replicates a variety of Workflow Sources sending workflows to cloud resources in order to complete specified tasks. To run the simulations, we created our own

scheduling program, with each resource having the requisite CPU capacity of 8 to 12 workflows per second (WFPS), RAM capacity of 16 MB, and link bandwidth capacity of 50 to 90 Mbps, with the link bandwidth capacity ranging from 500 to 900 Mbps.

Between 100 and 500 milliseconds of latency exists between cloud resources and each make-span. We compare our approach to two recent scheduling algorithms, MCAMO [13] and DQN [14], because they are widely employed in the cloud-related scheduling literature. The DQN method prioritizes the jobs that take the least amount of time to complete by scheduling them first. When the workflow count increased, DQN's performance degrades, resulting in poor resource usage and a long make-span. MCAMO, on the other hand, is based on the principle of prioritizing jobs with the longest Process time in order to reduce the overall make-span. Unlike the DQN and MCAMO algorithms, which only analyses the cloud resources side, our approach considers a broader range of parameters like CPU utilization, Memory Usage, bandwidth usage, and latency.

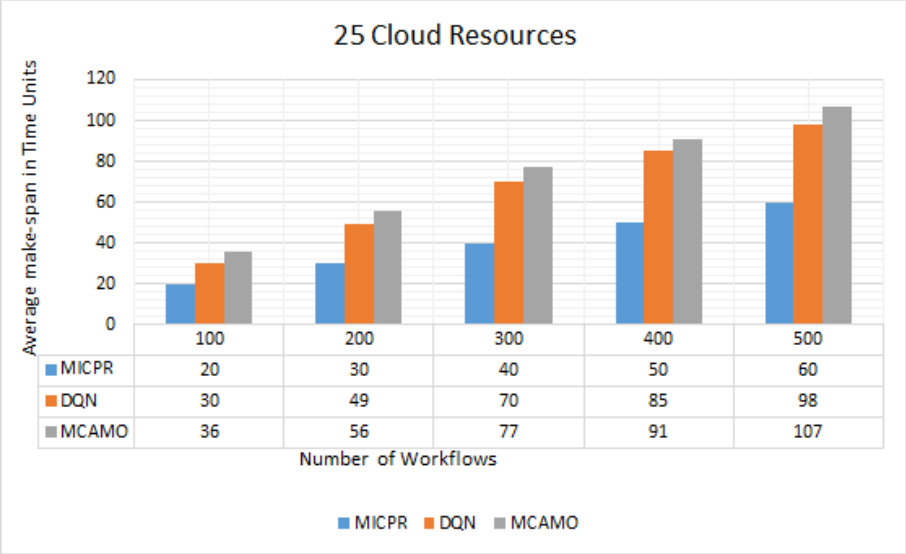
5.2 Experimental Results

The simulations are primarily focused on three key metrics: average task completion time, resource consumption, and overall solution Process time. To carefully analyze the adaptability of our system across a wide range of real-world instances, we evaluate each statistic under numerous realistic scenarios.

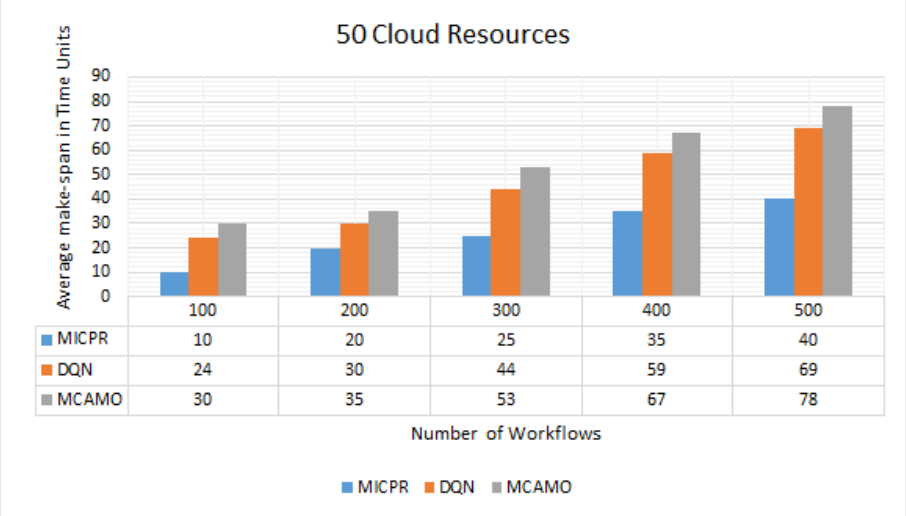
5.2.1 Average make-span

Let’s start by reviewing the length of time required by the various compared options. The "make-span" is the measure of time spanning two intervals with continuous transactions, which is worth noticing. The number of resources is fixed in the very first series of studies (Figure 4) while the variety of transaction sources is varied. The goal is to see how increasing the number of sources producing tasks affects the overall make-span. The number of cloud resources is maintained at 25, 50,100 as shown in Figure 4 (a), 4 (b), and Figure 4 (c) throughout the various scenarios in respective order. Whereas the number of Workflows ranges from 100 to 500.

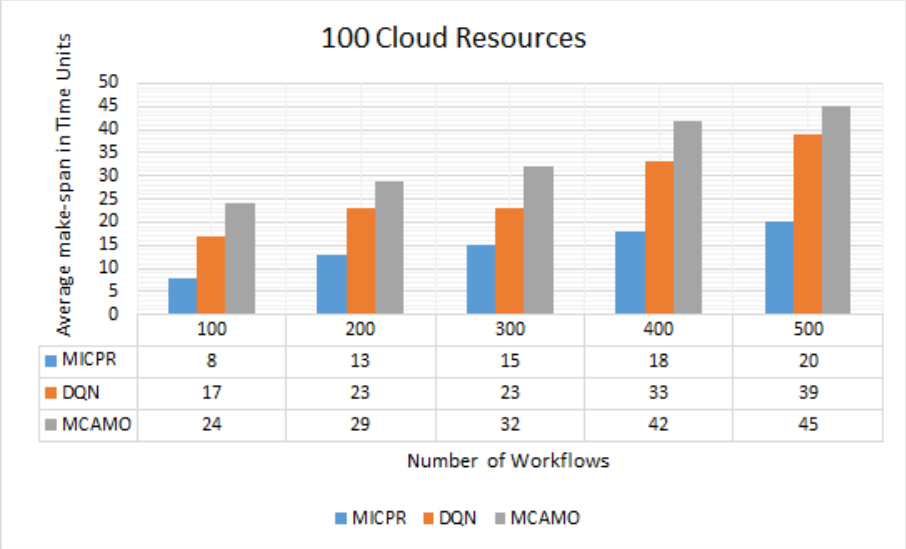
The first conclusion to be derived using Figure 4, which indicating that, in comparison to the MCAMO and DQN techniques, the proposed MIPR provided the reduced make-span. The reason for the DQN and MCAMO strategies' low performance when compared to MIPR is that they only examine task completion time, ignoring other crucial metrics like resource consumption and latency, which duly projects significant impact on the make-span. The second conclusion to be derived through Figure 4 is, the bigger the count of Workflows, the longer the make-span. This is due to the more number of Workflows relying on fixed set of cloud resources, which projects that, each resource loaded with more number of tasks. This inevitably leads to longer task wait times and, as a result, a longer total make-span.



(a) 25 Cloud Resources



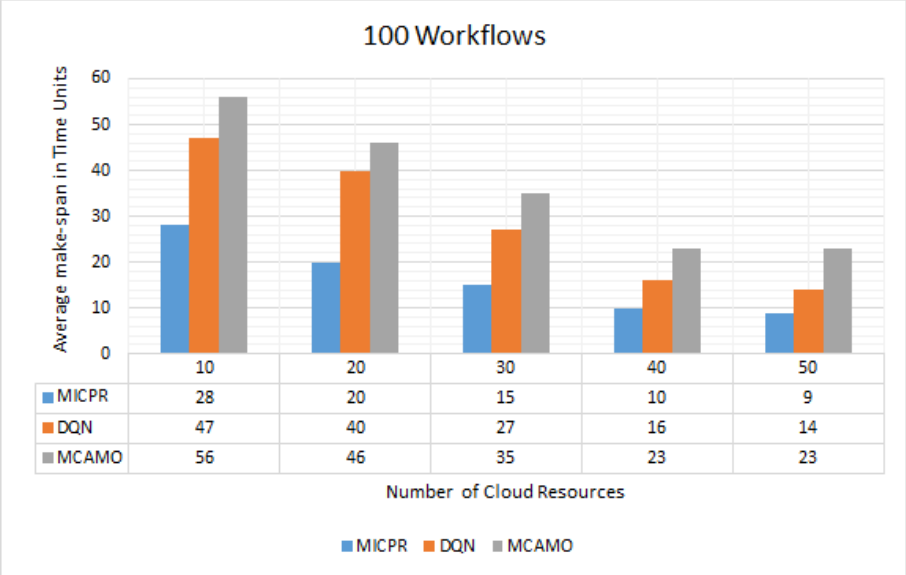
(b) 50 Cloud Resources



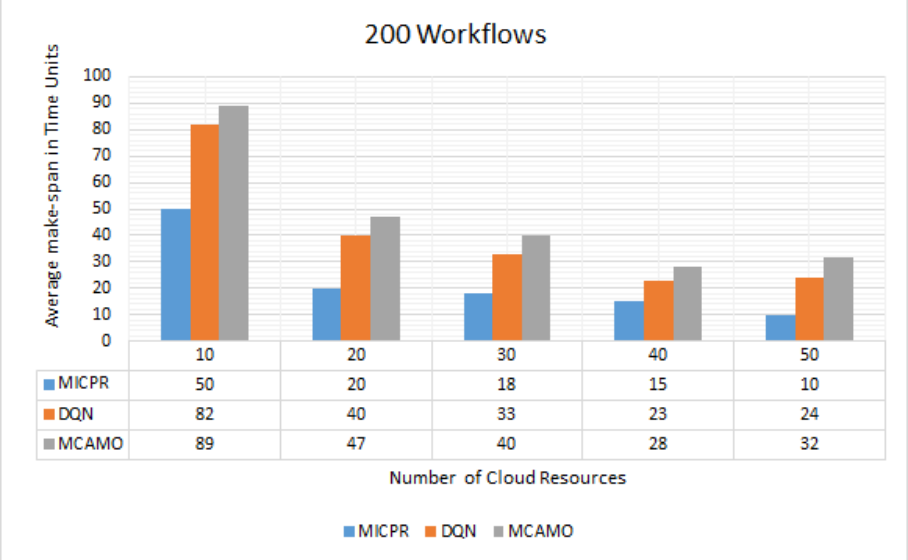
(c) 100 Cloud Resources

Figure 4: Average Make-spans at fixed cloud resources and variable workflows

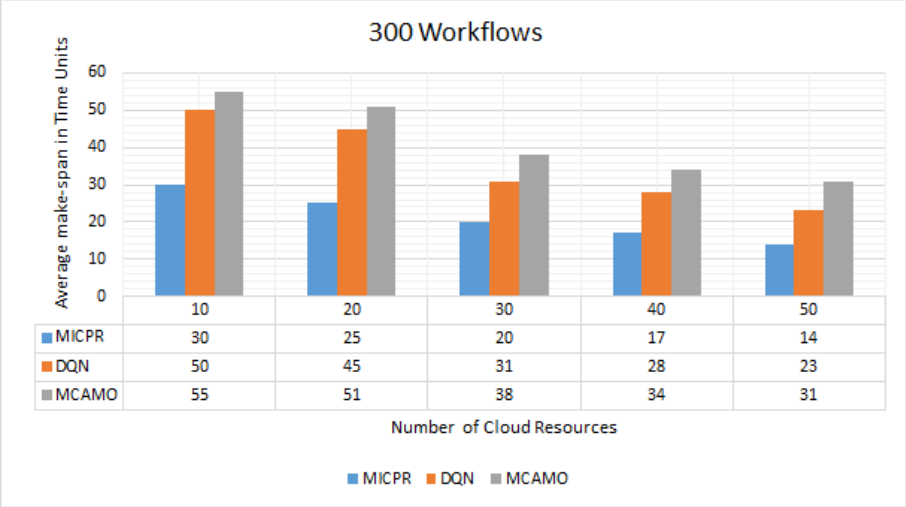
Further study measures the make-span of transaction workflows in Figure 5, where fixed workflows and variable count of cloud resources have been considered. The fixed count of Workflows considered as 100, 200, and 300 (figure 5). The findings show that the minimal make-span tenure for constant count of workflows and variable count of cloud sources. The rationale for this is that variable and continuous cloud resources are available to handle constant count of workflows, reducing wait times and, as a result, the time it takes to complete these operations. In terms of make-span, Figure 5 indicates that our MIPR technique excels the DQN and MCAMO algorithms. Even the quantity of workflows varies, the MIPR performs better than the contemporary models. Overall, our system beats the DQN and MCAMO methods minimizing the make-span and demonstrating improved sustainability as the workflows and cloud resources grows.



(a) 100 Workflow Resources



(b) 200 Workflow Resources



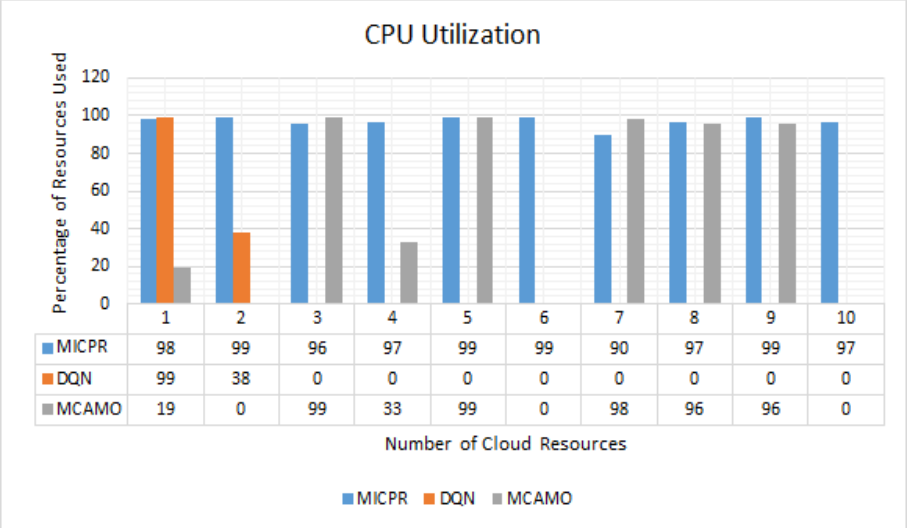
(c) 300 Workflow Resources

Figure 5: Average Make-span against fixed number of workflows and variable cloud resources

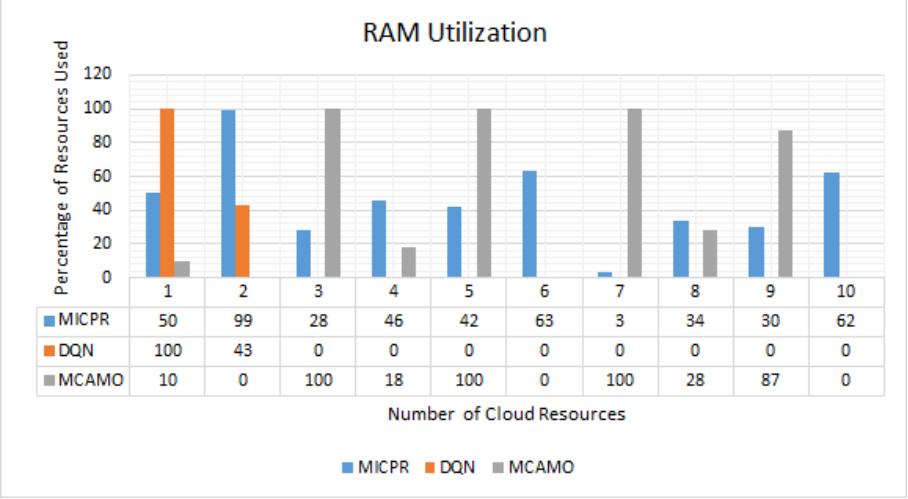
5.2.2 Resource Utilization

This phase of analysis verifies cloud resource usage ratio while serving workflows. The CPU utilization, RAM utilization, and the usage of link bandwidth are the metrics considered in this regard. The analysis of these metrics informs about the infrastructure costs that each of the evaluated options imposes on cloud network providers. The distinct trials for this set of tests performed with, varying the number of workflows as well as the amount of cloud resources that aimed to determining the sustainability of the various methodologies evaluated.

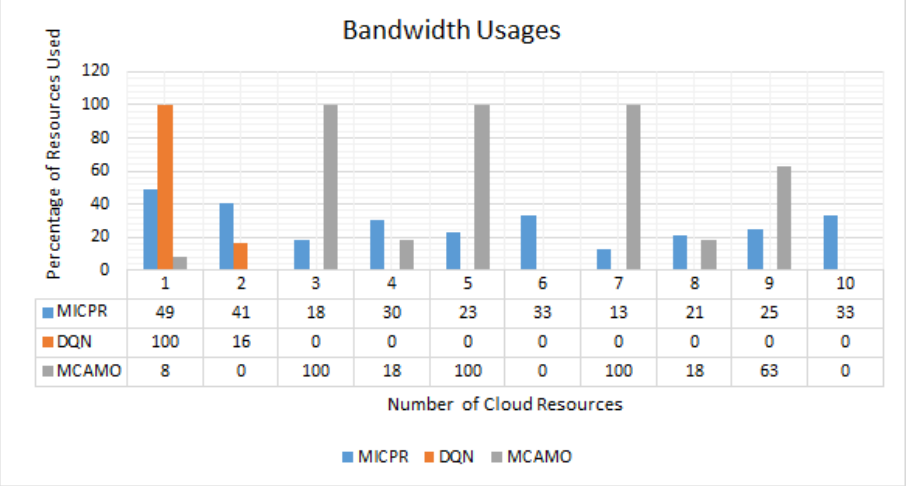
Figure 6 shows the stated metrics impact in a cloud system with 10 cloud resources and 100 workflows. In this area, we also conduct additional testing. The proportion of cloud resources was reduced to 5 in further experiment, while the proportion of workflows was fixed at 100. The phase of experiments considered the proportion of cloud resources to 15 against 150 workflows to 150 in another trial. The study of the last two trials are presented in figure 7. The first conclusion denotes that is increase in the count of cloud resources against given workflows results in lower cloud resource use.



(a) CPU Utilization



(b) RAM Utilization

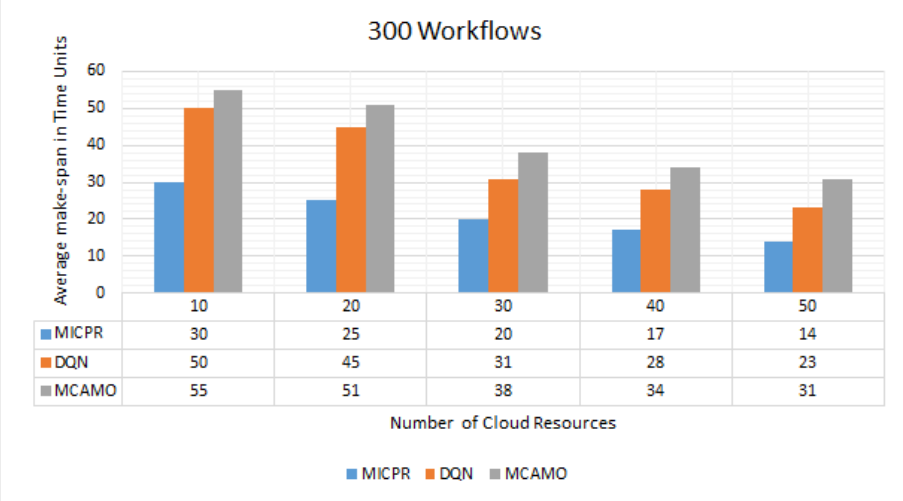


(c) Bandwidth Utilization

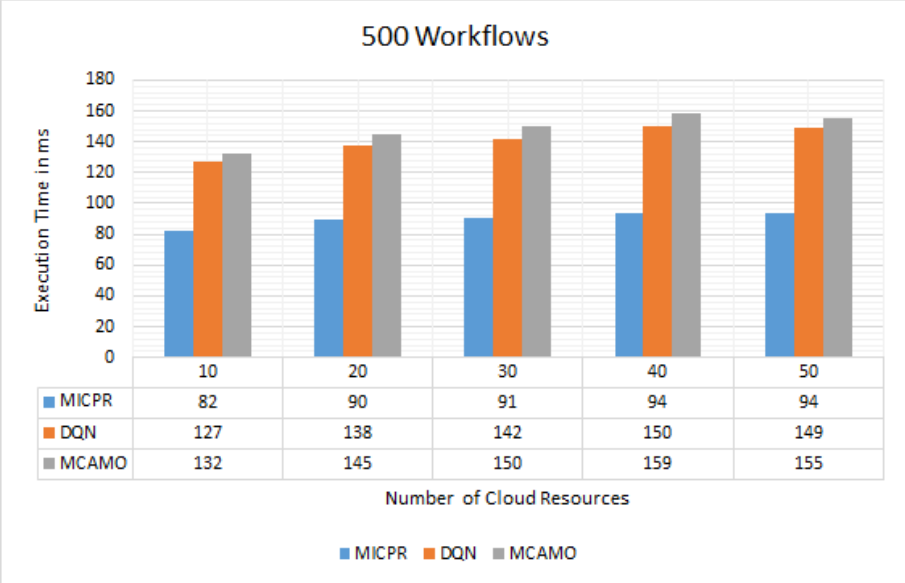
Figure 6: Resource Utilization for 10 cloud resources and 100 Workflows

This is to be expected, as having more cloud resources available to fulfil the fixed count of workflows reduces the load on each cloud resource. The second finding is that for various combinations of workflows and cloud resources, the DSRPR evincing nearly 100% cloud resource usage.

The resource utilization implied by the DQN and MCAMO algorithms, but at the other end, is not constant, with some cloud resources being completely utilized at times and others being underused or not used at all. The main reason for this is that the DQN and MCAMO approaches are time-based and simply disregard the cloud resources' preference and limits when making scheduling options. In contrast, the MIPR considers each resource's fitness of diversified metrics in scheduling process, allowing these cloud resources to achieve better usage and reducing the additional cloud resources to service the Sources. It has the virtue of lowering the infrastructural cost.



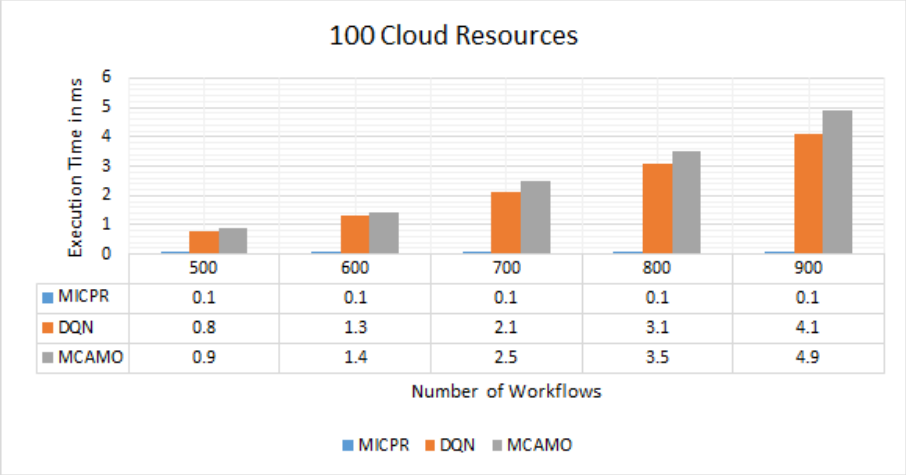
(a) 300 Workflows and Cloud Resources



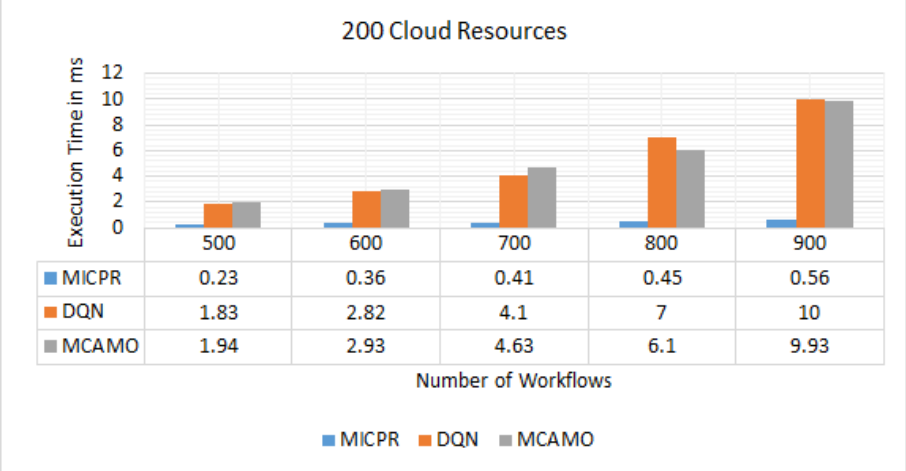
(b) Workflows and Cloud Resources

Figure 7: Process time against fixed count of workflows and variable count of cloud resources

The processing time imposed by the various contemporary methods is the third statistic that analyzed in further phase of experiments. To that purpose, the experiments performed in multiple trails, each with a different amount of cloud resources and workflows. Figure 7a shows the number of sources fixed at 300 and the proportion of cloud resources increased from 5 to 25. Figure 7b shows a deployment with a fixed number of 500 sources and a variable count of cloud resources ranging from 10 to 50. Figure 8a shows the cloud resources fixed at 100 and the workflows increased to 900. In Figure 8b, the cloud resources is fixed at 200, but the workflows is varied from 500 to 900. The Figure 7 denotes that as the process time is proportionate to the count of workflows, which is since, the broader variety of workflows that needs cloud resources. Furthermore, as the cloud resources grows in count, reduces the time it takes for different algorithms to run. The reason for this is that having more cloud resources means that sources must consider a greater number of inputs when generating their preference lists. In addition, as shown in Figure 8, our MIPR strategy takes less time to execute than the DQN and MCAMO techniques.



(a) 100 Cloud Resources and Workflows



(b) 200 Cloud Resources and Workflows

Figure 8: Process time against fixed count of cloud resources and variable count of workflows.

6 Conclusion

Resource scheduling is a profound challenge for organizations, and globally, various companies rely on the need for effective cloud computing resource scheduling for effective management of the IT infrastructure. Review of the existing resource scheduling models reflects the there are certain gaps in the system and scope for improvement. Thus, in this manuscript a contemporary statistical analysis model of cloud resource load indicator is proposed, which can be resourceful for sensing the load factors on the cloud servers in a network. Using the model as a surface system, the necessary deployment of robust resource sensing and scheduling algorithms can be applied. By adopting such surface application programs, the backend load factor on the servers can be curtailed, and performance management of the cloud networks is managed effectively.

Funding: No funding support for this research

Data Availability: The authors have provided information of data availability in the manuscript.

Conflicts of interest: The authors declare that they have no conflict of interest

Ethical approval: This article does not contain any studies with animals performed by any of the authors

References

[1] Madni, S. H. H., Abd Latiff, M. S., & Coulibaly, Y. (2016). Resource scheduling for infrastructure as a service (IaaS) in cloud computing: Challenges and opportunities. Journal of Network and Computer Applications, 68, 173-200.

[2] Strumberger, I., Bacanin, N., Tuba, M., & Tuba, E. (2019). Resource scheduling in cloud computing based on a hybridized whale optimization algorithm. Applied Sciences, 9(22), 4893.

[3] Chen, J., Xu, J., & Hui, B. (2017, July). Cloud Computing Resource Scheduling based on Improved Semantic Search Engine. In Proceedings of the 2nd International Conference on Intelligent Information Processing (pp. 1-5).

[4] Madni, S. H. H., Abd Latiff, M. S., & Coulibaly, Y. (2016). Resource scheduling for infrastructure as a service (IaaS) in cloud computing: Challenges and opportunities. Journal of Network and Computer Applications, 68, 173-200.

[5] Madni, S. H. H., Abd Latiff, M. S., & Coulibaly, Y. (2017). Recent advancements in resource allocation techniques for cloud computing environment: a systematic review. Cluster Computing, 20(3), 2489-2533.

[6] Mehta, H., Prasad, V. K., & Bhavsar, M. (2017). Efficient resource scheduling in cloud computing. International Journal of Advanced Research in Computer Science, 8(3), 809-815.

[7] Singh, S., & Chana, I. (2016). A survey on resource scheduling in cloud computing: Issues and challenges. Journal of grid computing, 14(2), 217-264.

[8] Loganathan, S., & Mukherjee, S. (2015). Job scheduling with efficient resource monitoring in cloud datacenter. The Scientific World Journal, 2015.

[9] Chen, M., Huang, S., Fu, X., Liu, X., & He, J. (2016). Statistical model checking-based evaluation and optimization for cloud workflow resource allocation. IEEE Work-flows on Cloud Computing, 8(2), 443-458.

[10] Liu, Y., Wang, L., Wang, X. V., Xu, X., & Zhang, L. (2019). Scheduling in cloud manufacturing: state-of-the-art and research challenges. International Journal of Production Research, 57(15-16), 4854-4879.

[11] Cui, H., Liu, X., Yu, T., Zhang, H., Fang, Y., & Xia, Z. (2017). Cloud service scheduling algorithm research and optimization. Security and Communication Networks, 2017.

[12] Srichandan, S., Kumar, T. A., & Bibhudatta, S. (2018). Task scheduling for cloud computing using multi-objective hybrid bacteria foraging algorithm. Future Computing and Informatics Journal, 3(2), 210-230.

[13] Ramamoorthy, S., Ravikumar, G., Saravana Balaji, B., Balakrishnan, S., & Venkatachalam, K. (2020). MCAMO: multi constraint aware multi-objective resource scheduling optimization technique for cloud infrastructure services. Journal of Ambient Intelligence and Humanized Computing, 1-8.

[14] Peng, Z., Lin, J., Cui, D., Li, Q., & He, J. (2020). A multi-objective trade-off framework for cloud resource scheduling based on the deep Q-network algorithm. Cluster Computing, 1-15.
[15] https://blog.quantinsti.com/pivot-point-strategy/
[16] Xue Y, H. M. (2017). Active learning of classification models with likert-scale feedback. In Proceedings of the 2017 SIAM International Conference on Data Mining, (pp. 28-35).