

Performance Analysis of NI Real Time Hypervisor using Decision making Algorithms

Mesala Sravani¹ Shiwali Sonwani² Sasi Kumar Gurumurthy³

M.Tech,(CSE) , Department of CSE, VIT , Vellore^{1,2}

Assistant Professor, Department of CSE, VIT, Vellore³

Abstract: NI Real-Time Hypervisor programming utilizes development of virtualization for running LabVIEW Real-Time close by to Windows XP, Red Hat Enterprise Linux or Windows 7 on stand out controller. In present days the multi criteria choice making systems are playing an important role for determination of best among the available choices. Multiple Criteria decision Making (MCDM) systems are focused around accumulating importance to the concept of "closeness to perfect". In this paper four multi-criteria decision making(MCDM) techniques Simple Additive Weighting (SAW) Method, Weighted Product Method (WPM), Grey Resolution Analysis (GRA), and TOPSIS algorithms are analyzed for Windows or Linux as general purpose operating systems. LabVIEW Real time system and the general purpose operating systems access the shared memory and for synchronizing it they use either blocking or polling methods. The test results of the system are considered and their efficiency and performance is calculated using the above four algorithms.

Key words: MCDM, LabVIEW, Operating systems, Hypervisor

I. INTRODUCTION

Each one designing application has it special necessities. A few frameworks need exact timing and hard constant execution, while others may oblige broadly useful OS abilities like rich client interface illustrations. Number of progressive applications effectively utilize more than one working framework to perform the current workload .During the process of running numerous working frameworks in a configuration has customarily implied utilizing one or more sets of registering equipment, NI Real-Time Hypervisor programming [1] empowers specialists to combine frameworks by running Lab VIEW Real-Time and either Windows XP, Windows 7, or Red Hat Enterprise Linux on a solitary controller simultaneously. Likewise, the Real-Time Hypervisor is solely intended to maintain the determinism of LabVIEW Real-Time programs.

The following are the benefits of NI Real Time Hypervisor Systems [1]:

- Reduce Cost and Physical Footprint
- Partition I/O Devices, RAM, and CPU Cores Quickly
- Take Advantage of a Variety of Options for Sharing Data Between OS's
- Make Better Use of Your Multicore Hardware
- Get Up and Running Faster

To run two Operating Systems parallelly on a solitary machine, the NI Real-Time Hypervisor uses virtualization engineering. The Operating systems basically are preferred based on their performance and efficiency. The performance and efficiency of these systems is tested using decision making algorithms and the best Operating system can be effectively used. This paper discusses the simulation of the four Operating Systems in terms of their

performance and efficiency which is calculated using various decision making algorithms.

II. VIRTUALIZATION TECHNOLOGY

Virtualization is a term which implies the reflection of machine assets. In practice, virtualization engineering empowers running numerous Operating Systems in parallelly on the same figuring equipment. This innovation has been utilized for a considerable length of time as a part of the IT area and designers are progressively exploiting virtualization to lessen the equipment necessities of their practical applications also. For a given framework, virtualization is performed by a bit of programming termed as a virtual machine screen (VMM) or a hypervisor. These terms are frequently utilized conversely. Singular OS occurrences running on a hypervisor are alluded to as virtual machines (VM's). Basically, hypervisor programming is in charge of overseeing access to I/O gadgets (counting those imparted between OS's), encouraging between OS correspondence, and, sometimes, booking virtual machines (while running on imparted CPU's).

Virtualization programming is classified into two categories as : facilitated and exposed metal. Facilitated VMM's run depending on a "host" OS and rely on it for booking and I/O access. Conversely, exposed metal hypervisors collaborate straightforwardly with machine fittings and don't depend on a host OS. Exposed metal programming bundles are appropriate for designing applications in light of the fact that they can be exceptionally intended to help running constant Operating system's in virtual machines. Furthermore, exposed metal virtualization programming permits individual VM's to get to I/O gadgets utilizing the local drivers.

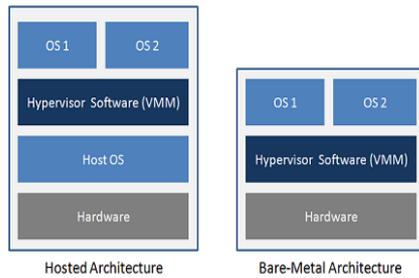


Figure 1 While hosted virtualization software runs on top of a host OS, bare-metal software runs directly on the underlying computer hardware.

NI Real Time Hypervisor Software Architecture[1]

The LabVIEW Real-Time and Windows XP or Red Hat Enterprise Linux are run in parallel as virtual machines by an uncovered NI-Real-time Hypervisor. At the point when introduced on an underpinned controller, NI Real-Time Hypervisor programming parcels the CPU centres in the framework as per a client arrangement. The NI Real-Time Hypervisor is focused on low-level Virtual Logix VLX virtualization programming.

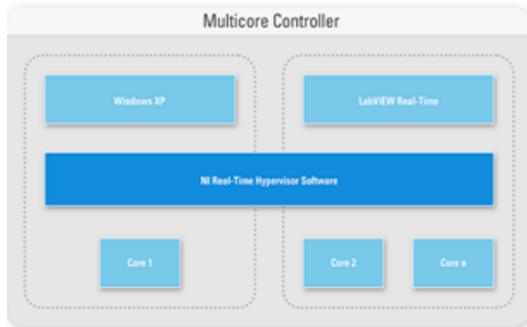


Figure 2. NI Real-Time Hypervisor for Windows programming runs Windows XP on one or more processor centres and LabVIEW Real-Time on the remaining centres

At the point when either LabVIEW Real-Time or the host Operating System endeavours to get to an imparted asset or impart to the next Operating framework, the hypervisor is naturally called by exceptional virtualization offers that are incorporated with the processor. Particularly, NI Real-Time Hypervisor fittings uses multicore Intel processors with builtin Intel-VT innovation. For execution purpose, the NI Real-Time Hypervisor parcels I/O modules and RAM between OS's (not withstanding CPU centres). Fancied OS assignments can be entered for each of the I/O modules in the undercarriage and show the sought division of RAM utilizing an implicit utility called the NI Real-Time Hypervisor Manager. Sometimes, the NI Real-Time Hypervisor Manager may ask for physically moving the I/O modules to distinctive undercarriage openings to evade intrude on clashes. Since every hypervisor call brings about some execution overhead, the Real-Time Hypervisor is called just when important.

III. SHARED MEMORY

NI Real-Time Hypervisor software includes a shared memory feature. With this feature, up to 95 MB of system memory is conserved, which is to be shared by both operating systems on the NI Real-Time Hypervisor

system. The data written in this section of memory can be directly accessed by both Operating Systems. Both the virtual Ethernet port and the Shared Memory feature use system memory; however, Shared Memory provides a high-throughput low-level API that should be used if you are frequently transferring large amounts of data between operating systems, or if you are storing data that if frequently accessed by both operating systems. Hypervisor Shared Memory VI's are used to transfer data between various operating systems.

IV. SYNCHRONIZING SHARED MEMORY DATA TRANSFER

Shared Memory data transfer requires that access to the data to be synchronized in some manner. A deadlock can occur if both operating systems attempt to write to the memory at the same time. There are two ways of synchronizing shared memory data transfer

➤ **Polling** : Synchronizes shared memory data transfer by using the **offset** terminal of the Read Shared Memory and Write Shared Memory VI's. You use this terminal to designate two elements of a shared memory block as polling flags that indicate when one OS should write data and when the other OS should read that data, thus preventing both OSes from writing to the data block at the same time.

➤ **Blocking** : This is the method which uses triggers to synchronize shared memory data transfer by using the "Send Shared Memory Trigger VI" on one OS and "Wait on Shared Memory Trigger VI" on the other OS. When the data flow reaches a "Wait on Shared Memory Trigger VI" on one OS, the thread on that OS blocks until the OS receives a cross-OS interrupt (XIRQ) from a "Send Shared Memory Trigger VI" on the other OS. If data flow reaches a "Send Shared Memory Trigger VI" first, the XIRQ remains in memory until data flow reaches the "Wait on Shared Memory Trigger VI" on the other OS.

V. BENCHMARKING TEST RESULTS

To benchmark shared memory data transfer, some standard tests are designed by NI R&D. The test's benchmark the shared memory transfer rates with polling and blocking synchronization methods with both Red Hat Enterprise Linux (RHEL) 5.5 and Windows XP for the general-purpose OS. Hardware and software used in the tests are:

Operating System	Parameter Block Size in KB						
	16	32	64	128	256	528	1064
Windows Blocking	0.07	0.09	0.11	0.14	0.19	0.24	0.28
Windows Polling	0.06	0.08	0.1	0.14	0.2	0.27	0.35
Linux Blocking	0.29	0.32	0.36	0.41	0.68	0.80	0.89
Linux Polling	1	1	1	1	1	1	1

Table.1 Benchmark Test Result

Hardware and Partitioning Configuration

- ❖ NI PXIe-8133 controller, no other I/O Modules installed
- ❖ 4 GB RAM split evenly between different Operating Systems. The maximum 95 MB is allocated to Shared Memory.
- ❖ 3 centers are appointed to Labview Real-Time and 1 center to the universally useful OS (Windows XP or RHEL)
- With the RTOS, threads can be assigned to run on specific processors (as was done in these tests—See attached VIs); whereas, with a general purpose OS this cannot be specified. Multiple cores on the general purpose OS can slow execution due to thread switching between cores. Additionally, Windows can decrease the execution priority of CPU-intensive applications, a feature users cannot disable.

Software

- ❖ NI Real-Time Hypervisor 2.0 or 3.0
- ❖ LabVIEW 2012 with the following components installed:
 - LabVIEW Real-Time 2012 SMP
 - Hypervisor Shared Memory components
 - SMP Support installed on the RT target
- ❖ Ethernet driver set to polling in MAX and Legacy USB disabled in the BIOS to decrease jitter

Simple Additive Weight (SAW)

Simple Additive weight(SAW) [2] which is likewise called as Weighted Sum Method (WSM) or scoring strategy. Weighted entirety strategy is a basic and most conspicuously utilized method. Weighted average technique is the premise for this calculation. An assessment score is ascertained for every option by multiplying the weight of relative importance with the scaled value allotted to the alternative of that particular attribute. Then the products of all the criteria are added. The utilization of SAW scoring is reliant on identification of destinations, identifying objectives and alternatives, assessment of choices, determination of sub-goal weights, added substance total of weighted fractional inclination values, sensitive examination and analysis. It uses immediate rating on the institutionalized scales just in simply qualitative qualities. Scores are calculated for numerical attributes by the normalized values to match the standardised scale.

$$Y_{ij} = \frac{U_{ij}}{U_j^{max}} \dots\dots (1)$$

$$Y_{ij} = \frac{U_j^{min}}{U_{xj}} \dots\dots (2)$$

The WSM method, underlying additive values function and alternative score $Y=Y(A_i)$ are calculated by addition

of weighting normalized values $W_j Y_{ij} \forall j = \{1, \dots, m\}$ before eventually ranking alternatives.

$$Y_i = \sum_{j=1}^m w_j Y_{ij} \dots\dots (3)$$

$$\text{For } Y \in R^{n \times m} \text{ with } i = \{1, \dots, n\}, \\ j = \{1, \dots, m\}; Y_{ij}, W_j, W_j \in (0, 1)$$

Weighted Product Method(WPM)

Weighted Product Method (WPM)[3,4] is an alternate MADM scoring system which is fundamentally synonym to WSM technique. The primary contrast in this system is the fact that, rather than addition, multiplication is done. The score X_i of the system i in equation (4) is obtained by the product of weights of the following attributes:

$$X_i = \prod_{j \in N} Y_{ij}^{w_j} \dots\dots\dots (4)$$

Here Y_{ij} denotes attribute j of the home operating system i , w_j denotes the weight of attributed j , and $\sum_{j=1}^N W_j = 1$. In equation (4), w_j is positive force for profit measurements $Y_{ij}^{w_j}$, and a negative force for cost measurements $Y_{ij}^{-w_j}$. As the parameter standardization is not obliged (i.e., it is discretionary), the calculated score of a framework by WPM will not have an upper bound [4], along these lines it is advantageous to look at the score of individual working framework with those of the positive perfect framework A^{**} . This framework is determined to be the framework with the best values in all the metrics. For preference metric, the best regard is the greatest. In case of a cost metric, the best regard is the least.

VI. (TOPSIS) Technique for Order Preference by Similarity to Ideal Solution

TOPSIS[3,6] known as a standout amongst the most traditional MADM techniques, is focused around the thought that the picked option which has the briefest separation from the Positive Ideal Solution (PIS) and on the other side, the most distant separation of the Negative Ideal Solution (NIS). The request of inclination is dead set on the premise of the closeness to a positive perfect arrangement and the most un great comparability to a negative arrangement. At present TOPSIS, the execution appraisals and the weights of the criteria are given as precise qualities. TOPSIS methodology is used to settle multi-objective nonlinear programming issues. The steps of TOPSIS model are as follows:

- The decision matrix which is normalized is calculated.
- The weight of decision matrix which is normalized is determined.
- The Positive Ideal solution and Negative Ideal Solutions are determined
- The separation measures for each alternative from the positive and negative ideal solutions are calculated.
- The relative closeness to the ideal solution for each alternative is calculated.
- Based on preference order the ranking is given.

TOPSIS algorithm is a technique which involves construction of normalized decision matrix. Each element is determined as

$$n_{ij} = \frac{a_{ij}}{(\sum a_{ij}^2)^{1/2}}$$

Where $i=1, \dots, m$; $j = 1, \dots, r$

The weight matrix which is a normalized matrix is determined and calculated using the formula

$$W_j = W_j n_{ij}$$

The positive and negative ideal solution are determined as shown below.

Positive ideal solution

$$A^* = \{Y_1^*, \dots, Y_n^*\},$$

where, $Y_j^* = \{max_i(Y_{ij}) \text{ if } j \in J; min_i(Y_{ij}) \text{ if } j \in J'\}$

Negative ideal solution

$$A' = \{Y_1, \dots, Y_n\},$$

where $Y_j = \{min_i(Y_{ij}) \text{ if } j \in J; max_i(Y_{ij}) \text{ if } j \in J'\}$

The separation between alternative and the positive perfect solution are given by :

$$R_i^* = [\sum_j (Y_j^* - Y_{ij})^2]^{1/2}, i = 1, \dots, m$$

The negative alternate arrangement and every option are differentiated by a separation which is given as:

$$R_i' = [\sum_j (Y_j' - Y_{ij})^2]^{1/2}, i = 1, \dots, m$$

Finally the relative closeness to the perfect arrangement U_i^* is ascertained as

$$U_i^* = R_i' / (R_i^* + R_i'), 0 < U_i^* < 1$$

Grey Relation Analysis

According to Grey relation theory[4,5] the random process is believed to be a grey quantity variable in an area of certain amplitude and time zone. The correlations among factors and candidates of a system can be obtained using Grey relational analysis. The major advantage with this method lies with the fact that with inadequate information, the qualitative and quantitative relationships are estimated and calculated from numerous factors. Grey relational analysis is calculated as follows.

- i) The Original system X_o is initialized
- ii) From the initial original sequence, the ideal value is identified.

$$Y^+(X(n)) = \{X_1^+, X_2^+, \dots, X_i^+(n)\}$$

$$\{(Max_n. X_i(n)/l \in L_1), (Min_n. Z_i(K)/l \in L_2)\},$$

$$i=1,2,\dots,7; n=1,2, \dots, 16$$

L_1 is a set of benefit attributes and L_2 is a set of cost attributes.

- iii) The Original sequence is normalized as

$$X_i^*(n) = \frac{X_i(n)}{Y^+(X_o(n))}$$

- iv) Grey relational deviation sequence is determined by the formula

$$\Delta_{oi} = \|X_o(n) - X_i(n)\|$$

- v) Maximum and Minimum deviation are identified by the equations

$$\Delta_{max} = |X_o(n) - X_i(n)| ;$$

$$\Delta_{min} = |X_o(n) - X_i(n)|$$

- vi) Grey relational coefficient is calculated by

$$\beta(X_o(n), X_i(n)) = \frac{\Delta_{min} + \eta \Delta_{max}}{\Delta_{oi}(n) + \eta \Delta_{max}}$$

- vii) Grey relational grade for 4 alternatives is calculated as

$$\beta(X_i, X_l) = \sum_{m=1}^n \alpha_m \beta(X_i(n), X_l(n));$$

Here the total probability $\sum_{m=1}^n \alpha_m = 1$;

$$\text{Hence } \beta(X_i, X_l) = \frac{1}{n} \sum_{m=1}^n \beta(X_i(n), X_l(n))$$

Result's and Discussions

Simple Additive Method(SAW)

Operating System	Windows Blocking	Polling Windows	Linux Blocking	Polling Linux
Weight	0.0901	0.0990	0.30	0.5109
Rank	4	3	2	1

Table 2 Weights for SAW method

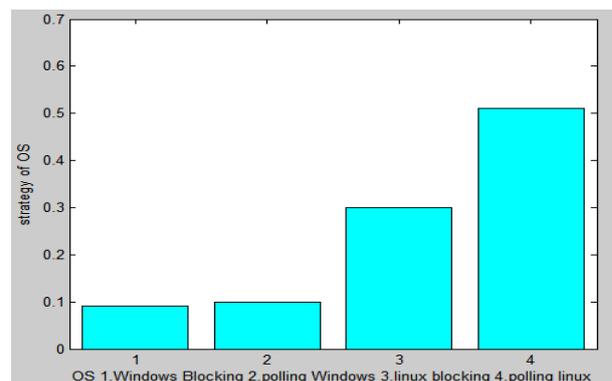


Figure 3 Bar chart for SAW method

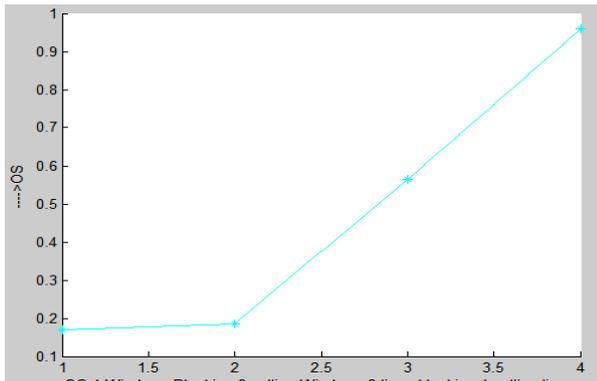


Figure 4 Simulation Plot for SAW method

The simulation results of SAW method show that the Linux operating system using polling method is the best method for synchronizing shared memory in NI real time systems as it has ranked Linux operating system using as the best among the four systems which were considered.

Weighted Product Method(WPM)

Operating System	Windows Blocking	Polling Windows	Linux Blocking	Polling Linux
Weight	0.0902	0.0929	0.29	0.5268
Rank	4	3	2	1

Table 3 Weights for WPM method

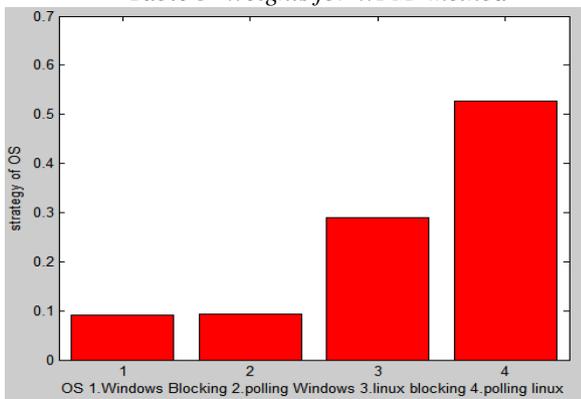


Figure 5 Bar chart for WPM method

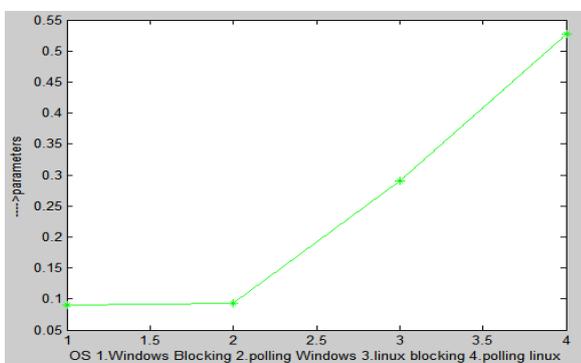


Figure 6 Simulation Plot for WPM method.

The simulation results of WPM method show that the Linux operating system using polling method is the best method for synchronizing shared memory in NI real time

systems as it has ranked Linux operating system using as the best among the four systems which were considered.

Grey Resolution Analysis (GRA)

Operating System	Windows Blocking	Polling Windows	Linux Blocking	Polling Linux
Weight	0.6501	0.5990	0.6765	1.1667
Rank	4	3	2	1

Table 4 Weights for SAW method

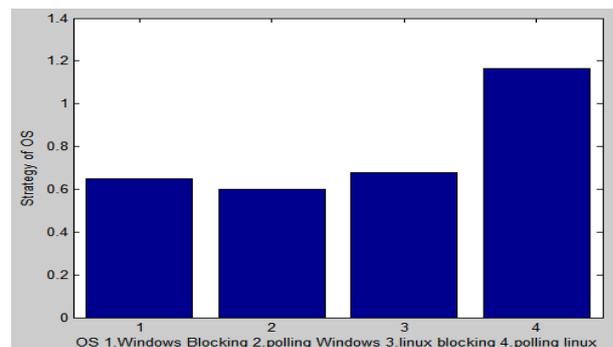


Figure 7 Bar chart for GRA method

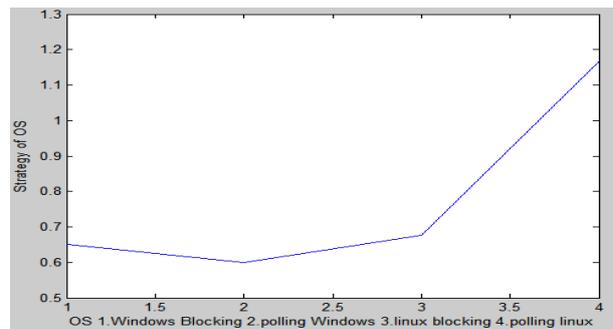


Figure 8 Simulation Plot for GRA method.

The simulation results of GRA method show that the Linux operating system using polling method is the best method for synchronizing shared memory in NI real time systems as it has ranked Linux operating system using as the best among the four systems which were considered.

VII. TOPSIS algorithm

Operating System	Windows Blocking	Polling Windows	Linux Blocking	Polling Linux
Weight	0.0071	0.0395	0.4805	1.0
Rank	4	3	2	1

Table 5 Weights for SAW method

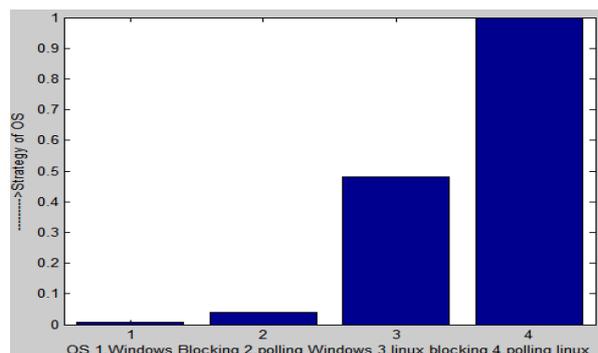


Figure 9 Bar chart for TOPSIS method

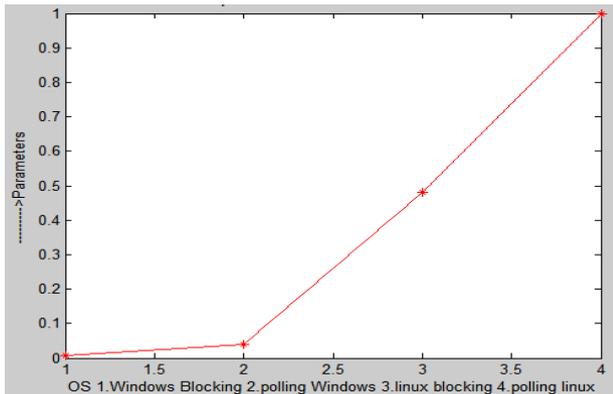


Figure 10 Simulation Plot for TOPSIS method.

The simulation results of TOPSIS method show that the Linux operating system using polling method is the best method for synchronizing shared memory in NI real time systems as it has ranked Linux operating system using as the best among the four systems which were considered.

VIII. EFFICIENCY

Efficiency of the Linux system using polling method for the four algorithms is given in the table below.

OS	TOPSIS	WSM	WPM	GREY
Efficiency of Polling Linux(%)	46.53	37.4	46.53	18.4

Table 6 Efficiency of polling Linux

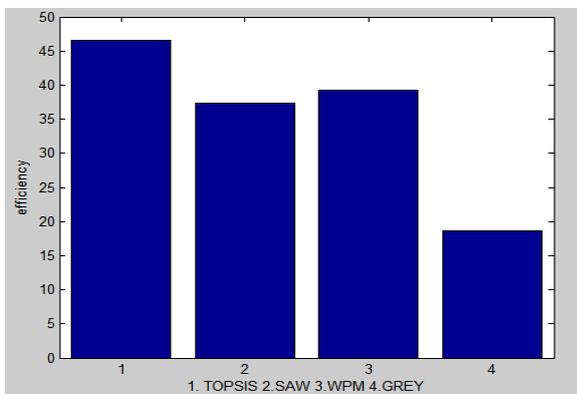


Figure 11 Bar chart for Efficiency of Polling Linux

CONCLUSION

The Multi model Decision-Making (MCDM) are picking up vitality as potential devices for examining complex true issues because of their innate capacity to judge diverse choices (Choice, strategy, policy, scenario can also be used synonymously) on different criteria for conceivable determination of the best/suitable option (s). These alternatives may be further explored in-depth for their final implementation. These decision making algorithms are used to evaluate the performance of various computing systems. The simulated results show that the Linux system using polling method is the best system in terms of various attributes that are considered.

REFERENCES

1. www.zone.ni.com
2. www.ni.com/white-paper/12603
3. Mortaza Honarmande Azimia, Houshang Taghizadehb, Nasser Fegh-hi Farahmandb and Jafar Pourmahmoud, "Selection of industrial robots using the Polygons area method", International Journal of Industrial Engineering Computations, 5 (2014); 631-646.
4. D. Sameer Kumar1, S. Radhika and K. N. S.Suman, "MADM Methods for Finding The Right Personnel in Academic Institutions", International Journal of u- and e- Service, Science and Technology Vol.6, No.5 (2013), pp.133-144
5. Martin Aruldoss, T. Miranda Lakshmi, V. Prasanna Venkatesan, "A Survey on Multi Criteria Decision Making Methods and Its Applications" American Journal of Information Systems, 2013, Vol. 1, No. 1, 31-43
6. S.R.Gangurde and M.M Akarte, "Ranking of product alternatives based on customer-designer preferences", IEEE International Conference on Industrial Engineering and Engineering Management - IEEM, pp. 75-79, 2010
7. Hsin-Hung Wu, "A Comparative Study of Using Grey Relational Analysis in Multiple Attribute Decision Making Problems", Journal: Quality Engineering, vol. 15, no. 2, pp. 209-217, 2002

BIOGRAPHIES



Mesala Sravani was born in India, A.P, in 1992. She received B.Tech degree from JNTU Kakinada in 2014. She is presently pursuing her M.Tech degree at Vellore Institute Of Technology (VIT), Vellore, TN.



Shiwali Sonwani was born on 9-8-1990. She received her B.Tech degree from Sri Govindaram Seksaria Institute of tech and science, Indoor, MP. She is presently pursuing her M.Tech degree at Vellore Institute Of Technology(VIT), Vellore, TN.



Prof. G.Sasi Kumar received his B.E. degree in Computer science from the Madurai Kamaraj University, Madurai in 2003, his M.E., from the Anna University Chennai in 2005 and Pursuing Ph.D. in VIT University, Vellore from 2009. He was a Faculty of J.J. College of Engineering and Technology for eight months and he was with Kuriji college of Engineering and Technology for two years and eight months, 2006-2009, heading the Information Technology Department during 2007-2009. He is now with the School of Computing Science and Engineering, VIT University, Vellore from March 2009.