A Project Report on

“Performance Impact due to Virtualization on Personal Laptop System”

Submitted by

Akshay R. Thorat

G01032733

Guided by

Prof. Daniel A. Menasce'

George Mason University

CS 672: Computer System Performance Evaluation

Abstract

As Cloud Computing is growing in present scenario which uses virtualization technique for providing Platform as a Service (PaaS). With virtualization, migration, failover management, resource management can be easily done. The same hardware can be shared with multiple entities independently without interacting with each other. Multiple Operating systems (OS) can be run on the system.

Performance modelling for System could be beneficial to facilitate best service for the user and which can help deciding various factors in service as Service Level Agreement (SLA), Cost, Availability etc.

Now with Virtualization, depending on the type of virtualization provided, there is occurrence of overhead which is introduced as there is no full control over the hardware of the system and translation is needed between instructions from virtual machine and bare hardware.

This project aims to provide method by which one can build performance model for the intended System with integrating overhead in the classes present. Here personal windows machine is used to find the Service Demand and Response time in various scenarios. This method can be scaled for large system to find the SLA and bottlenecks in the system

Keywords: SLA, Virtual Machine (VM), Service Demands, Response Time, Overhead

Contents

Introduction

For the modelling analysis, here I have used VM Ware Workstation which is Type 2 Hypervisor [1]. In type 2 hypervisor, Virtual Machine Manager (VMM), which is responsible for managing Virtual Machine (we can see it as coordinator between local OS and OS in Virtual Machine), is one of the process in local OS and managed by the OS. So in in non-privileged mode, it cannot access many of the hardware assistance features of modern CPUs like trap and emulate. The VMM detects the sensitive instruction and perform action on behalf of the OS, as it needs binary translation for this there is occurrence of overhead.

Overhead due to VM consists of resource usage by the operating system. Overhead has two components: a constant component and a variable component. Constant component is for activities performed by an OS that do not depend on the level of system load, such as the CPU time required to handle an I/O interrupt. The variable component of overhead corresponds to these activities that are dependent on the system load. [2]

To build performance model with overhead, there are two methods available

1. Workload class for representing Overhead

    - This includes overhead of OS activates performed on behalf of application programs, in this case of VMM.
    - Because of Variable nature of the overhead, service demand of this class needs to be made load-dependent.
    - The dependency in overhead parameters and workloads can make this approach difficult and error prone.

2. Distribute overhead among classes

    - Now the overhead is incorporated within the class.
    - Relative fraction is calculated for dividing the workload
    - Service demand is load independent and can be used for scaling and analysis.

In this project, second method is used where relative factor is calculated and assigned to class accordingly.

Building Performance Model

The input parameters needed to build performance model are

1. Workload Intensity

    - This can be seen as the indication of load of the system
    - If the rate at which transaction arrived to the system is provided, then workload is specified in arrival rate and if it is in terms of number of batch jobs submitted to the system then concurrency level is used.
    - In the tested system, as workload is specified by arrival rate.

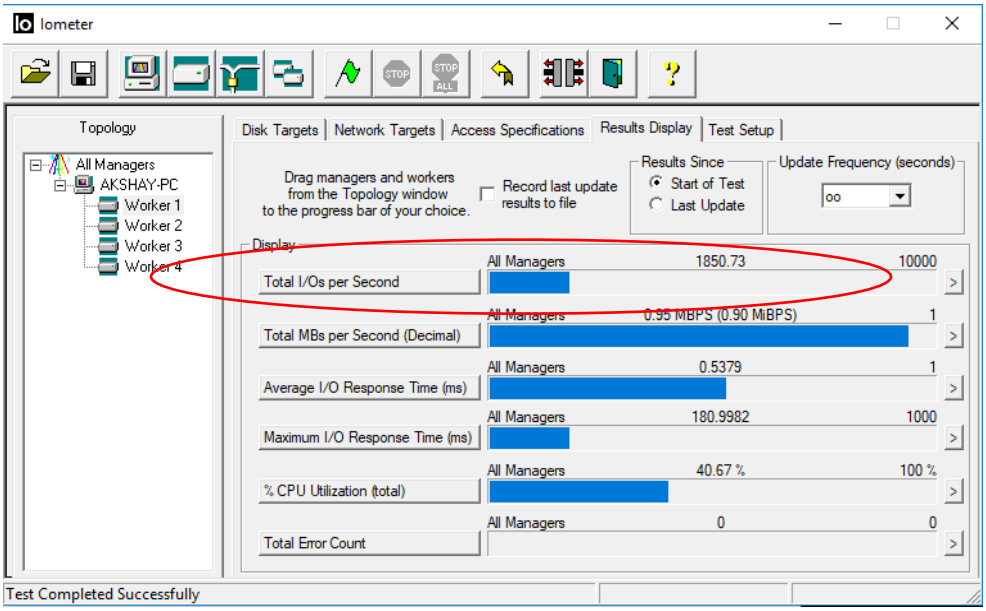
2. Service Demand (D_i)

    - It specifies total average time provided by the resource to the class of workload.
    - D_i generally is load independent and does not rely on the system load.
    - This parameter can be used to check the scaling of the system if load on the system is increased
    - It can provide insights of the system, in terms of response time and can be helpful of defining SLA, availability and cost of the system.

Obtaining Input Parameters

1. With Iometer (Arrival rate)

    Workload of the system can be characterized into CPU bound or IO bound. In IO bound, the process or workload spends its maximum time in disk ie performing IO, so this result in high disk utilization. For the generation of workload, Iometer [3] is used, which generates IO bound workload which can be used to check capacity of disk in the system.



We can see that arrival can be obtained in I/O's per second. Here we are considering Open Queuing Network where arrival rate is equal to system throughput at equilibrium.

2. Data collector Sets

To measure utilization of various devices on the system, windows provide system tool named Performance monitor.

Computer Management -> System Tools -> Performance -> Monitoring tool -> Performance Monitor.

Here user can define DataCollector Set which can be configured with some performance counters. This performance counter measures the specified field and stores the results. In the testing below are the counters which are used to gather the data.

PhysicalDisk(_Total)\% Idle Time - Udisk can be found with (100 - ideal time)

PhysicalDisk(_Total)\Disk Transfers/sec - Total IO performed which is same as obtained with Iometer

Processor(_Total)\% Idle Time - Ucpu (100 - ideal time)

Process(vmware-vmx)\% Processor Time - $U_{cpu,VM,}$ specifies utilization by VM(per-class)

Process(Dynamo)\% Processor Time - $U_{cpu,Iometer}$, utilization by Iometer

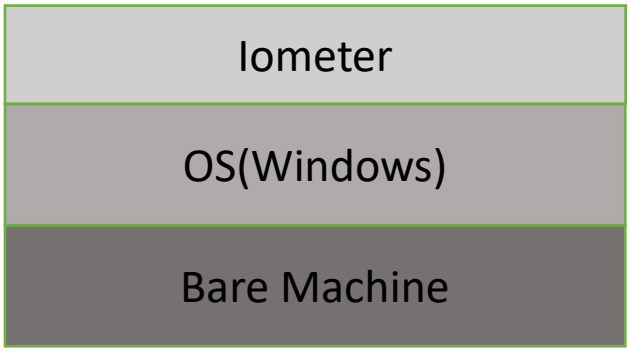
Process(vmware-vmx)\IO Data Operations/sec – Specifies number I/O performed by VM, used to find $U_{disk,VM}$

Process(Dynamo)\IO Data Operations/sec - number I/O performed by Iometer , used to find $U_{disk,Iomter}$

The measurements are included for 2 min where in data collector set, at 10 sec interval snapshots are taken. Same with Iometer, the workload generator ran for 2 min.

Case 1: Iometer is ran in Local OS and Virtual OS independently

1. In Local OS



Iometer ran for 2 min and data collector at 10 sec intervals, below are the results obtained

Time	PhysicalDisk(_Total)\% Idle Time	PhysicalDisk(_Total)\Disk Transfers/sec	Processor(_Total)\% Idle Time
02:45.5			
02:55.5	13.929	1694.385	76.633
03:05.5	0.017	1772.073	77.101
03:15.5	1.374	1478.195	81.584
03:25.5	2.209	1685.488	80.234
03:35.5	0.005	1654.74	82.291
03:45.5	0.001	1599.853	81.718
03:55.5	0.009	1557.144	81.733
04:05.5	0.001	1592.24	83.056
04:15.5	0.004	1638.038	83.54
04:25.5	0.005	1679.264	83.194
04:35.5	0.001	1766.128	75.195
Avg. Value	1.596	1647.05	80.571
	98.404		19.429

Udisk = 98.40% , X0 = 1647(same by Iometer) , Ucpu = 19.429%

By service demand law,

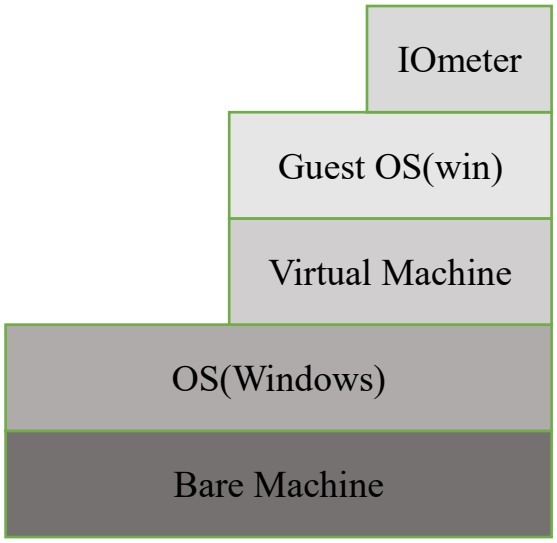
$$Di = \frac{Ui}{X0}$$

$$Dcpu = \frac{Ucpu}{X0} = \frac{0.1943}{1647} = 0.000118 \text{ sec}$$

$$Ddisk = \frac{Udisk}{X0} = \frac{0.984}{1647} = 0.000597 \text{ sec}$$

2. Inside Virtual Machine

Here Iometer is running inside VM.



VMware Workstation is installed on local OS and guest OS is Windows 10 same as local OS as we are trying keep all parameters same in both OS.

Below are the results obtained from the tests performed.

Let's calculate the parameters needed to build the performance model.

Time	PhysicalDisk(_Total)\ % Idle Time	PhysicalDisk(_Total)\Disk Transfers/sec	Process(vmware-vmx)\IO Data Operations/sec	Processor(_Total)\ % Idle Time
19:02.4	37.72	991.49	989.78	67.54
19:12.4	28.72	1020.69	1045.17	63.50
19:22.4	51.22	876.31	889.51	59.84
19:32.4	39.24	887.86	885.06	66.87
19:42.4	39.04	1037.15	1031.55	67.32
19:52.4	14.11	1114.71	1110.91	69.31
20:02.4	33.92	1010.34	1007.34	69.81
20:12.4	0.58	1034.53	1022.92	66.88
20:22.4	1.88	947.32	938.83	68.98
20:32.4	25.96	1084.12	1080.72	68.65
20:42.4	14.48	1058.94	1052.33	69.39
Average	26.08	1005.77	1004.92	67.10

Udisk = 100 - 26.0784 = 73.92

Ucpu = 100 - 67.09849423 = 32.90

X0 = 1005.769602 IO/s

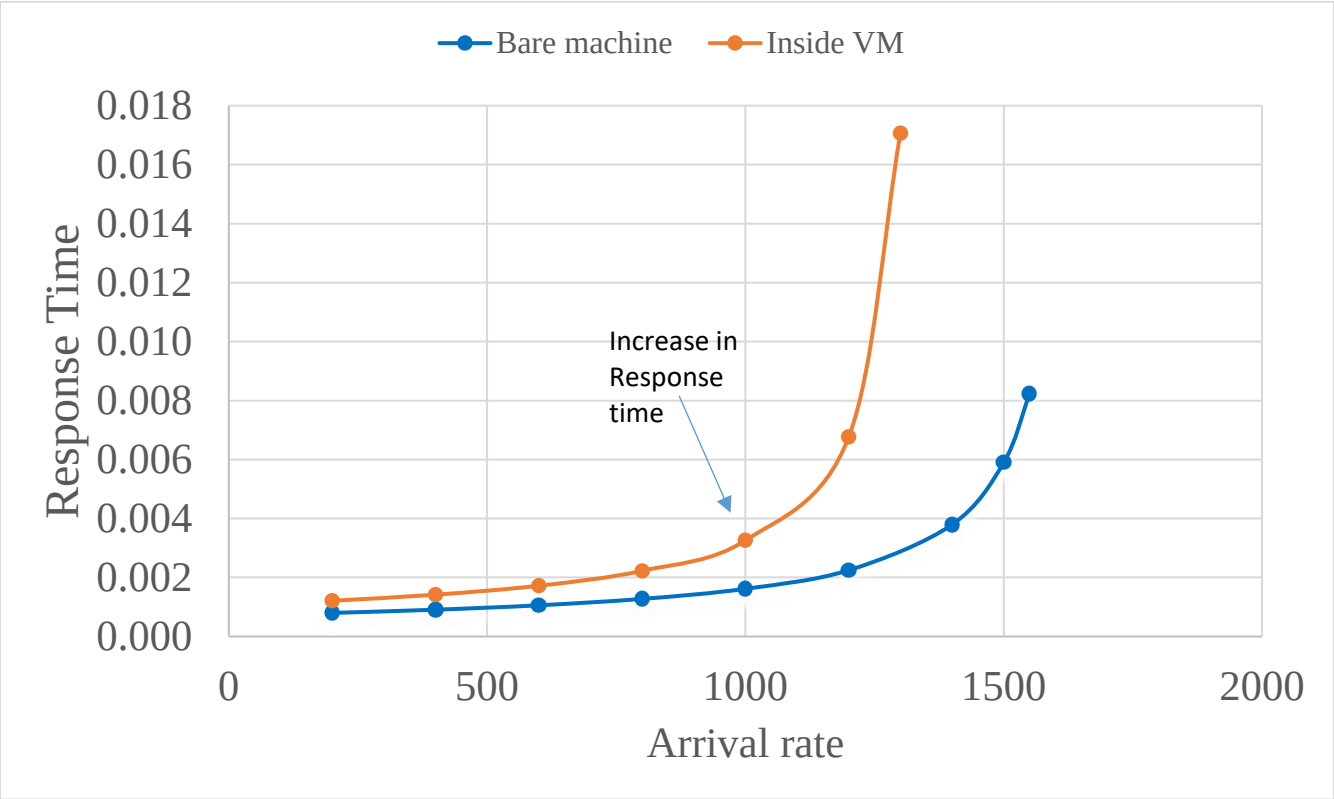
As nothing is running inside the Local OS, the Utilization measured by data collector is true measurement. There is no need to calculate relative factors and overhead is incorporated in the utilization values.

Now with the same method used above Di can be calculated.

	Local OS	Inside VM
Dcpu	0.000118	0.000327
Ddisk	0.000597	0.000735

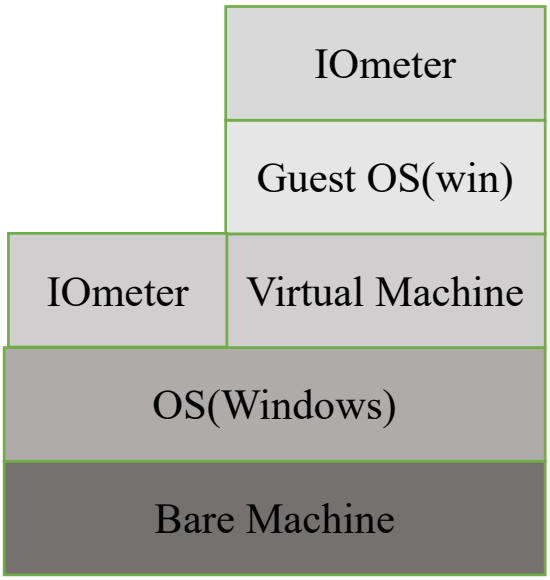
From above table it is clear that the service demands for devices when workload was inside VM is higher than running on local OS. This is due to overhead of running Virtual Machine Manager (VMM).

Now we have obtained Service demands which are load independent, we can use them to form performance model, I have used OpenQN.xls [2] to get values of Response time (R) for various arrival rate.



- As expected, for same arrival rate, R(in sec) is different.
- R in less on Bare machine (local OS) and more inside VM.
- We can see that at arrival rate as 1000 io/sec, there is exponential increase in R.
- After that for small increase in arrival R changes rapidly.

Case 2: Iometer and Iometer inside VM running simultaneously on Local OS



- In local OS, Iometer and VM both are running
- Update data collector with counter as
    - Process(vmware-vmx)\% Processor Time
    - Process(Dynamo)\% Processor Time
- In case 2, five measurements are taken, each with 12 snapshots at 10 sec interval

Below are the results obtained aggregating 12 of measurements into 1 and taken 5 times.

Global	**Global**	**VM ware**		**Local**	
Processor(_Total)\% Idle Time	**PhysicalDisk (_Total)\% Idle Time**	**Process(vmware-vmx)\IO Data Operations/sec**	**Process(vmware-vmx)\% Processor Time (divide by 2)**		
Ucpu	**Udisk**	**X0**	**Ucpu**	**X0**	**Ucpu**
31.18	98.87	639.70	21.14	1342.09	6.54
30.52	97.13	630.02	20.55	1330.62	6.49
29.41	99.39	623.15	19.66	1298.28	6.31
30.71	99.81	626.85	19.71	1304.72	6.13
32.43	96.10	645.39	23.69	1396.45	6.96

Now average value of the measurements is taken

Global		VM		Local	
Ucpu	**Udisk**	**X0**	**Ucpu**	**Xo**	**Ucpu**
30.85	98.26	633.02	20.95	1334.43	6.48

Here, we have to find relative fraction ($f_{i,r}$) to include overhead inside the class itself only.

- For Ui, we have to find $f_{i,r}$

  Ucpu = 30.85 % , $f_{cpu,local} = \frac{Ulocal}{Ulocal+Uvm}$

  Ucpu,local(Iometer) = .3085 * $\frac{20.95}{20.95+6.48}$ = .2356 ,

  Ucpu,vm = .3085-.2356 = .0729

- For Udisk,

  $f_{disk,local} = \frac{\text{no. of disk operation by local}}{\text{no. of disk operation by local+no. of disk operation by VM}}$,

  which will give $U_{disk,local}$ and $U_{disk,vm}$

- Using this data Di,r is calculated with service demand law,

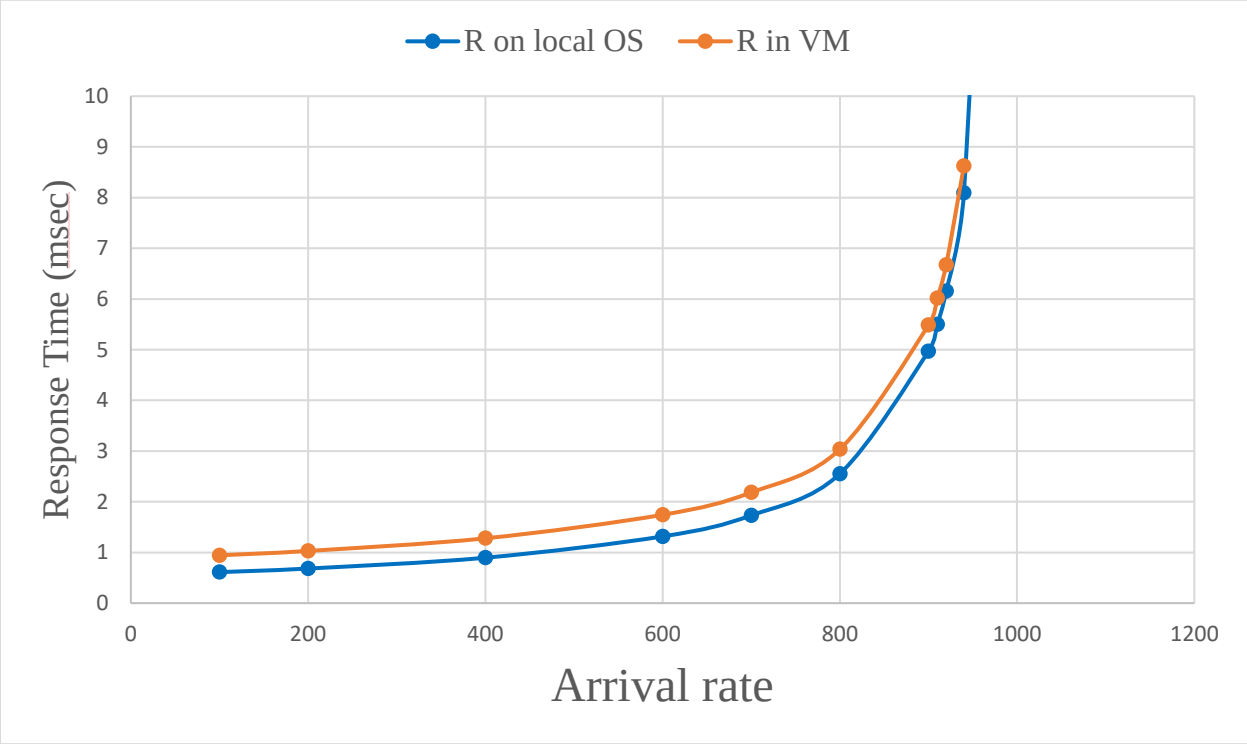
  $$\boldsymbol{Di,r} = \frac{\boldsymbol{Ui,r}}{\mathbf{X_{0,r}}}$$

Local		**VM**	
Ucpu	**Udisk**	**Ucpu**	**Udisk**
0.24	0.32	0.07	0.67

With above values calculated we can find Di,r

Dvm,cpu	3.72E-04	Dlocal,cpu	5.46E-05
Dvm,disk	5.12E-04	Dlocal,disk	4.99E-04

Now we have obtained Service Demands, we can build performance model and find response times for various arrival rate.



- In this case, open Queuing n/w with two class and two queues
- At 800, exponential change, due to high disk utilization
- After certain point, both response time come to converge as utilization of disk is getting to saturation and queue length is increasing.
- Optimal R value at 1334.432 and 633.022 (which is given by Iometer)

Conclusion

- For same type of workload service demands and Response times are different because of the overhead
- Same method can be followed for large scale system
- With modelling, SLA can be defined efficiently
- Organizations can find bottlenecks in their infrastructure
- From easily available data what if analysis can be done

References

[1] A. SILBERSCHATZ , P. GALVIN and G. GAGNE, “Virtual Machine” in Operating System Concepts Ninth Edition, 2013

[2] D. A. Menascé, V. Almeida, and Larry W. Dowdy, Performance by Design: Computer Capacity Planning by Example, Prentice Hall, 2004.

[3] Iometer workload generator, http://www.iometer.org/

[4] https://en.wikipedia.org/wiki/Hardware-assisted_virtualization