JUNO performance evaluation and optimization on virtual platform

JUNO performance evaluation and optimization on virtual platform

X M Zhang[1], Z T Ma[1], C Yang[2], Y LV[2], J H Chen[2]

1 Institute of High Energy Physics, 19B Yuquan Road, Beijing 100049, P.R. China
2 Zhejiang University,866 Yuhangtang Road | Hangzhou, Zhejiang Province, 310058, P. R. China
E-mail: zhangxm@ihep.ac.cn

Abstract. To run the JUNO observatory applications on cloud, performance evaluation and optimization for JUNO software on virtual platform is necessary. With benchmark tools and the latest JUNO offline software, the paper presents the design of a complete set of evaluations to find out the best choices of virtualization infrastructures for JUNO applications. To facilitate testing procedures, automatic tools have been developed. The findings during tests and the suggestions to future improvements of the JUNO software will also be described in this paper. In the optimization part, we will describe the factors affecting performance and the ways we manage to improve the JUNO simulation and reconstruction processes in virtual platform by 10% ~20% in multi-VM cases.

1. Introduction

JUNO (Jiangmen Underground Neutrino Observatory) [1] is a multi-purpose neutrino experiment designed to measure the neutrino mass hierarchy and mixing parameters. JUNO is estimated to be in operation in 2019 with 2 PB/year raw data rate. The Institute of High Energy Physics Computing Centre (IHEPCC) in China is planned to hold most of computing resource for JUNO data processing. With the existing experiments, the IHEPCC currently supports about 12000 CPU cores and 10 PB storage. With new experiments to come in the next few years, the IHEPCC is estimated to manage twice or more servers as today. Without increasing manpower, the virtual platform is selected to meet future challenge. The cloud infrastructure called IHEPCloud[2] on OpenStack is being built up. IHEP users can submit jobs through cluster and distributed computing to IHEPCloud in an elastic way. JUNO is selected to be one of the first experiments to run on virtual platform.

Virtual platforms always bring certain performance loss comparing to physical platform. Also the loss is quite different among applications, hardware and OS environment. Before using cloud at large scale, it is necessary to know the behaviour of the JUNO software on various virtual platforms environment, including hardware, hypervisors, memory, size of VMs, etc.

2. Performance Evaluation

The JUNO Offline Software [3] is composed of three parts including framework, offline physics packages, and external libraries. In our tests, four types of data processing are concerned. They are shown in Figure 1: Physics Generator (PhyGen) and Detector Simulation (DetSim), Electronics Simulation (EleSim), PMT Reconstruction (PmtRec), Event Reconstruction (EvtRec), of which EleSim is the most IO consuming tasks, and the others are more CPU bound.

The purpose of evaluation is to find performance bottleneck of virtualization platform for later optimization, and to understand which hardware, OS and virtual machine set-up would suit the best.

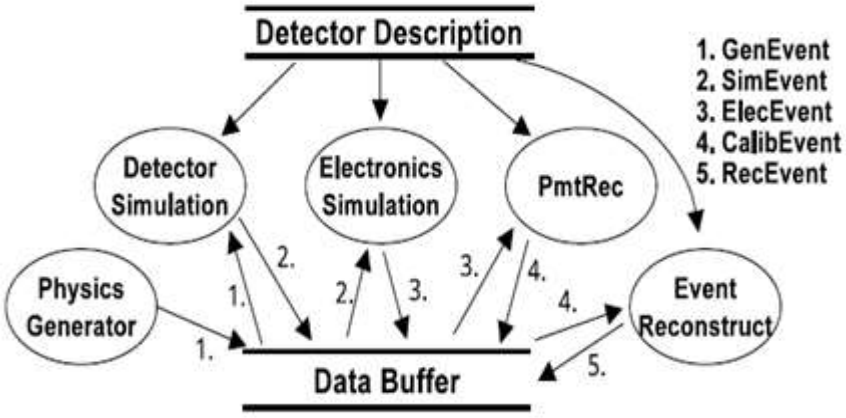

Figure 1. JUNO data processing types.

2.1. Design

Various indicators and different scenarios have been considered in JUNO performance evaluation plan, as shown in Figure 2. The indicators are used to identify loss in certain environment and scenarios, which include basic benchmarks (CPU, IO, network, memory) and performance of JUNO software (DetSim, ElecSim, PMTRec, EvtRec). The scenarios try to cover possible system setting and environment (VM size, hardware, OS, data access, KVM configuration). The results of evaluation will be presented in the rest of the paper. Automation tools are developed to facilitate the evaluation and the analysis of final results.

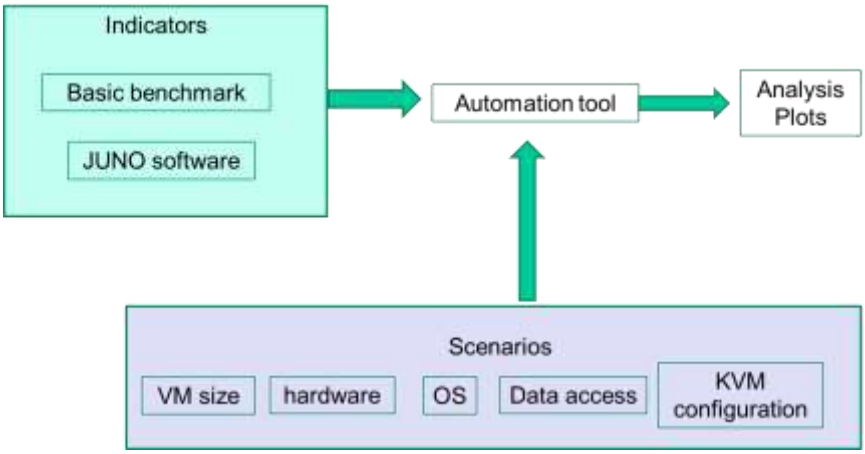

Figure 2. Evaluation architecture.

2.2. Testbed

Three types of CPU and disk were used in the tests:

- Hardware1(H1): Intel(R)Xeon(R) CPU E5-2630L v2@2.40GHz , 2CPU with 6 cores/CPU, the disk type is MM1000FBFVR from HP

- Hardware2(H2): Intel(R) Xeon(R) CPU X5650@ 2.67GHz , 2CPU with 6 cores/CPU, the disk type is ST31000340NS
- Hardware3(H3): Intel(R) Xeon(R) CPU E5-2630 v3@ 2.40GHz, 2CPU with 8 cores/CPU, the disk type is ProLiant BL460c Gen9

The memory size for each core is 4 GB. The ext4 file system is used. We chose KVM as hypervisors used in IHEPCloud. The KVM library used is libvirt 0.10.2, and the image format is qcow2. The operation system of physical and virtual machines is Scientific Linux 6.5. The JUNO software version uses J16v2r1-Pre2 in this paper. CVMFS [4] is used to deploy JUNO software with CVMFS client installed in test machines and the CVMFS and squid server located in the same local-area network.

2.3. Results and findings

All the performance loss is given comparing to physical machines of the same kind.

2.3.1. Basic benchmarks

Four basic benchmarks have been measured in the hardware mentioned above. For CPU, we tested with the HEP-SPEC06 [5] package, and found H1 has significant CPU loss about 17%, much higher than the other two. For IO, we use IOZone [6] which can measure performance of various read and write operations, covering read, re-read, initial-write, re-write, random-read, random-write, stride-write. The tests show that the loss is fluctuating a lot among different operations and the loss of read is less than that of write generally. H2 has the biggest IO loss of the three. NetPerf [7] has been used to evaluate network performance. In Table 1, the tests configured with virtio-net mode are shown, with which the network loss are small enough to be ignored. The memory access is evaluated with STREAM [8] with 4.6% loss. To summarize, IO and CPU still are the main contributors to the loss, and the hardware is one of the important factors. Further checks on hardware for reasons of loss are still needed.

Table 1. Performance loss of VMs to PMs with basic benchmarks.

	Tool	Perf Loss(H1)	PerfLoss(H2)	PerfLoss(H3)
CPU	SPECCPU 2006	17.68%	6.25%	8.83%
I/O	IOZone	10%~16%	13%~36%	4~9%
Memory	STREAM	4.6%	6.5%	2.7%
Network	NetPerf	<1%	<1%	<1%

Each VM needs to be supported by one KVM process so that it is interesting to know the external memory consumption the KVM process brought. The VMs with n cores where n ranges from 1 to 12 cores have been investigated. Each core is assigned with 4 GB memory. From Figure 3, we can see that one VM with single core the KVM process consumes 667 MB. Although when the VM size increased to 12 cores the consumption reached 1810MB, the consumption for each core greatly decrease to about 150 MB. Therefore, big VMs are expected to consume less memory for the platform itself than small ones.

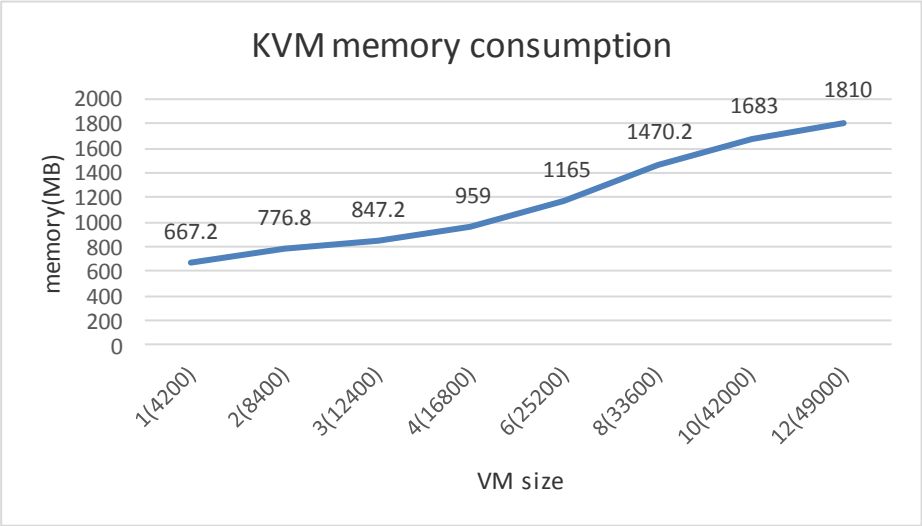

Figure 3. Memory consumption of KVM vs. VM size.

2.3.2. JUNO software

In this part, we investigate performance loss of four types of JUNO data processing with different hardware and the results are shown in Table 2.

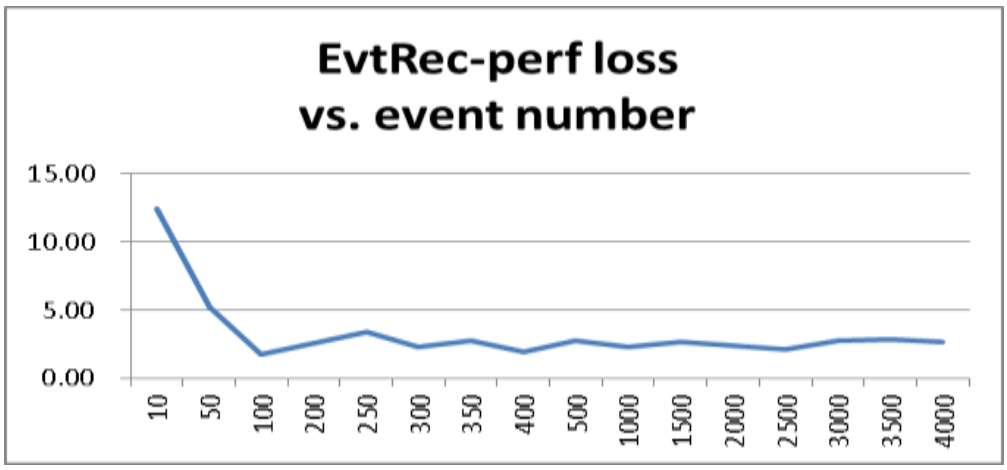

Figure 4. EvtRec performance loss (%) vs. event number.

At the beginning, the tests with the events ranging from 10 to 4000 were done in order to know the effects given by the initialization and preparation part of the software before event processing which include load of the libraries and access of geometry data, etc. The EvtRec result is shown in Figure 4. With the increase of event number, performance loss decreases before 100 events and tends to be stable after that, which shows initial part of the application caused higher penalty than event processing part. This is mostly because that initial part includes more IO intensive operations such as loading libraries and a large number of geometry parameters, etc. The DetSim process has similar result where the effect is negligible with more than 300 events in single process case, but not for multi-process case which will be described in the DetSim evaluation of this part.

Next we are going to show some interesting findings:

- Mostly the results are consistent with those of basic benchmarks, but some not. H1 shows 9.25% high CPU loss in DetSim and ElecSim has the biggest IO loss in H3. The test where ElecSim run without output has proved most of penalty come from IO. In PmtRec, it is surprising to find that the CPU loss in H3 is high with 9.5%. The reason of this high CPU loss is explained in Section 2.4.
- The CPU loss is quite different among the programs. DetSim which is based on GEANT4 has higher CPU loss than the others. The reconstruction processes have less loss than the simulation ones.
- Comparing with basic benchmarks shown in 2.3.1, the loss for JUNO is smaller both in CPU and IO.

Table 2. Performance loss in JUNO software benchmark.

	PerfLoss(H1)	PerfLoss(H2)	PerfLoss(H3)
DetSim	9.25%	5.5%	5.1%
ElecSim	14.1%	18.5%	7.5%
ElecSim* (no out)	3.4%	1.8%	2.8%
PmtRec	1.2%	0.6%	9.5%
EvtRec	1.4%	1.8%	1.3%

Here we would like to highlight the DetSim and EvtRec case where we further investigate with multi-core and multi-process scenarios with H2. With 12 core machines, the resource is arranged like the following: nVM, 12/n cores for each VM (n: 1 – 6) . Twelve processes are running in parallel in the whole set of cores. The final results tell us that with the increase of VM number, the loss increases from 5% to 20%. The worst point is 12 VMs (1 core per VM). For EvtRec, the loss has similar trend, but not so high, ranging from 1.8% to 6%. More studies on the 12 VMs DetSim case showed that the initial part is a quite IO intensive process and more than 28% have been spent for the initial part during one hour wall time, while only 2 minutes are needed for this part in a physical machine. Therefore, the IO conjunction from the initial part is the main reason for the heavy penalty in 12 VMs case. Two more methods have been applied to further prove that. One is to start the processes not in parallel, but with 5 minutes' gap, and performance loss was found to be greatly reduced from 20% to 6.21%. The other is to pre-load experiment software in the VMs before starting the processes, and the performance loss is found to go down to 7%. Therefore, heavy IO conjunction could be the bottleneck in production with high load of parallel jobs, and IO needs to be improved in the initial part, especially for the simulation program.

2.4. Optimizations

From the evaluation results shown in Table 2, we can see most of loss for JUNO software is less than 5%, but some of them still need to be improved. Therefore, we focused on the significant loss. One is the CPU loss above 9% from DetSim in H1 and PmtRec in H3. In VM configuration, CPU feature is set to be adopted from its parent physical machine. This setting is very helpful in DetSim and PmtSim. With that, the DetSim CPU loss decreased from 9.25% to 6.54%, and PmtSim loss from 9.5% to 4.5%. The other to be improved is the serious IO loss in ElecSim. Several kinds of VM disk settings have been tried and compared which are proved to have significate effects on final performance:

- Disk image type in QCOW2 – the effect of sparse space allocation causes serious penalty in the IO intensive ElecSim process. In QCOW2, the default disk setting is preallocation=off where less space is used at first but slower space extensions happen at later requests. In this mode, when the new write comes, OS needs to look up in the physical disk and allocate a new block to the virtual image which incurred a penalty. If the preallocation=full mode is used where all the space is pre-allocated at the beginning, the performance can be improved by 80%. The Figure 5 shows that space growth with growing events incurs significate penalty when event number is above 600.

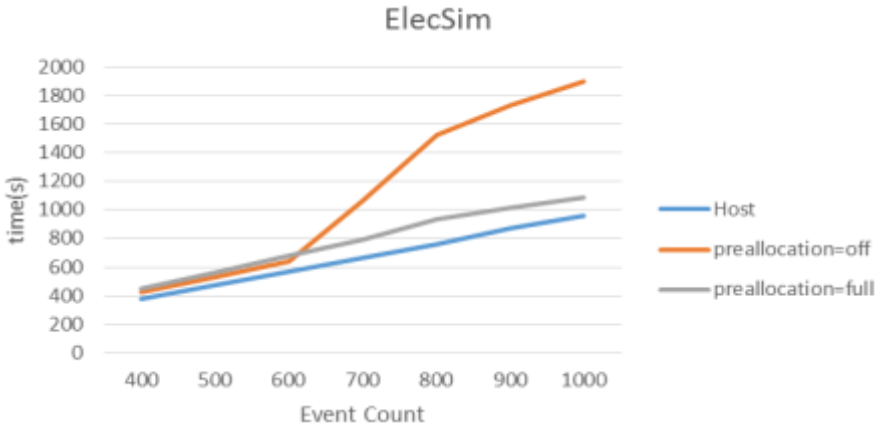

Figure 5. Comparison of two preallocation modes.

- Cache mode in disk access -- In writeback mode, both host page cache and guest KVM page cache are open, which allows good writing performance for applications. In writethrough mode, the host page cache is enabled, but the disk write cache is disabled for the guest so than the write performance might be reduced. The writeback and writethrough cache modes have been evaluated and found that the writeback can help improve IO by 10% compared to the default writethrough mode. One problem though, is that in the writeback mode the data integrity can't be assured when there are failures of transfer.

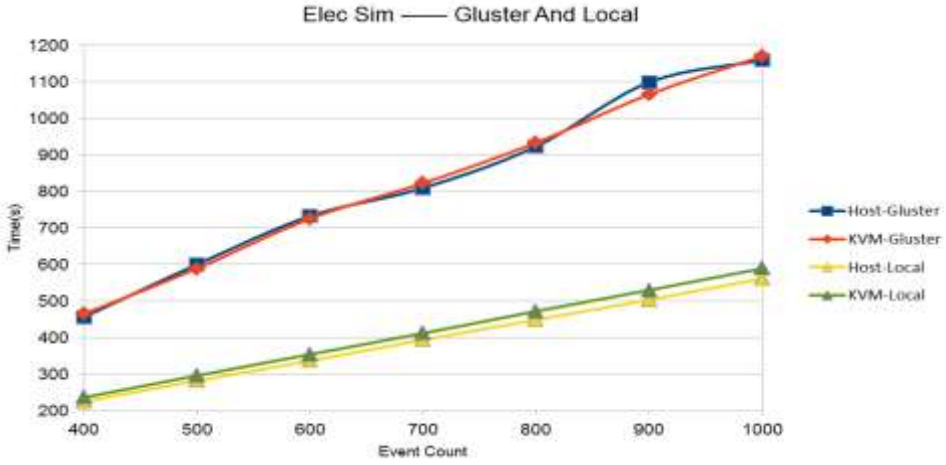

Figure 6. Comparison of Gluster and local.

- Network file system to place data – Since IO performance loss is more serious than that of network. We are interested to see performance of using the network file system to hold input and

output data instead of local disk. Also the network file system is the way to hold data in IHEPCloud so it is important to investigate that aspect. In the tests we use Gluster. The results are shown in Figure 6, where the total performance for both host and VM with data in local disk is better than the one with data in Gluster, but the loss between host and VM in Gluster case without disk I/O loss is 10% better than that of local disk.

- Other settings including CPU pinning, KSM EPT and THP have also been tried and only slight effects has been found.

3. Automatic evaluation tools

Many factors influence performance of applications on virtual platforms such as hardware, application version, KVM parameters, OS, etc. These factors keep changing during usage, eg. New Nodes with different hardware are added, OS and Cloud manager upgraded, applications upgraded to new versions. It is necessary for us to make sure these changes don't have adverse effects on performance. Therefore, automatic tools are needed to make the process easy and sustainable. Figure 7 shows the architecture of the automation tool which have been designed and developed. The tool includes four main parts: Evaluation Machine management (EM), Evaluation Scene management (ES), Evaluation Activity management (EA) and Evaluation Results management (ER). They are accessible to the users through commands and web pages. The EM is responsible for adding and deleting the machines for testing and monitoring. The ES is used to manage the necessary scripts to set up and check environment and start testing workflows in machines. The EA takes care of collecting the selected scripts and deploying to the chosen test machines. When the processes are finished, the final results will be retrieved back and transformed into JSON format, and stored in DB. The plotting and analyzing tools will be used to finally compare and analysis the data from DB.

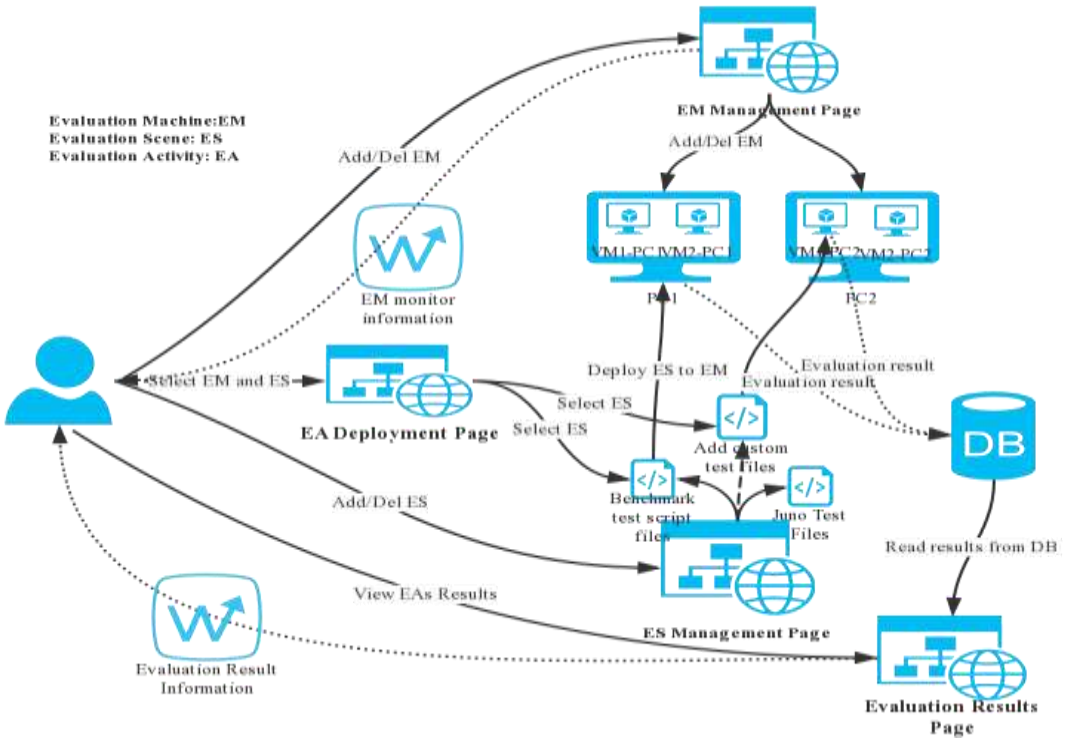

Figure 7. Architecture of Automatic evaluation tool.

4. Conclusions and future work

The JUNO Evaluations have showed CPU-bound processes are suitable to run on virtualized resources and IO penalty is still a key issue in IO intensive processes. Both hardware and software are found to be two of the important factors that influent the penalties in this paper and there are no general rules to the applications. It is therefore important to further investigate with automatic evaluation tool to cover more factors and cases. Also it is useful to try out more tunings on virtual platforms in order to achieve better performance improvements. Besides evaluating with single machine, the tools to test and analysis on cloud production environment would be also helpful to track down performance problems in real scenarios.

Acknowledgments

The authors would like to thank JUNO offline software team for their help with the JUNO software, and colleagues at the IHEP computing centre for their support. This work was funded by the National Natural Science Foundation of China (NSFC) under grant no. 11375221.

References

[1] An F P *et al* 2016 Neutrino Physics with JUNO *J. Phys.* G **43** 030401
[2] Huang Q L *et al* 2015 BESIII physical offline data analysis on virtualization platform *J. Phys.: Conf. Ser.* **664** 022024
[3] Zou J H *et al* 2015 SNiPER: an offine software framework for non-collider physics experiments *J. Phys.: Conf. Ser.* **664** 072053
[4] Buncic P *et al* 2010 CernVM - a virtual software appliance for LHC applications *J. Phys.: Conf. Ser.* **219** 042003
[5] HEP-SPEC06 benchmark, http://w3.hepix.org/benchmarks/doku.php
[6] IOzone Filesystem Benchmark, http://www.iozone.org/
[7] NetPerf, http://www.netperf.org
[8] McCalpin, John D. 1995 Memory Bandwidth and Machine Balance in Current High Performance Computers *IEEE Computer Society Technical Committee on Computer Architecture (TCCA) Newsletter*