ISSN 1392–124X (print), ISSN 2335–884X (online) INFORMATION TECHNOLOGY AND CONTROL, 2015, T. 44, Nr. 3

Private Cloud Infrastructure for Applications of Mechanical and Medical Engineering

Arnas Kačeniauskas, Ruslan Pacevič, Miglė Staškūnienė

Department of Graphical Systems, Vilnius Gediminas Technical University
Saulėtekio al. 11, Vilnius, LT-10223, Lithuania
e-mail: arnas.kaceniauskas@vgtu.lt

Dmitrij Šešok, Dainius Rusakevičius

Faculty of Fundamental Sciences, Vilnius Gediminas Technical University
Saulėtekio al. 11, Vilnius, LT-10223, Lithuania

Audrius Aidietis, Giedrius Davidavičius

Centre of Cardiology and Angiology, Vilnius University Hospital "Santariškių Klinikos"
Santariškių g. 2, Vilnius, LT- 08661, Lithuania

crossref http://dx.doi.org/10.5755/j01.itc.44.3.7379

Abstract. The paper presents the development of the private cloud infrastructure providing advanced services for research in mechanical and medical engineering. The cloud services have been developed by using Apache jclouds API and JetS3t Toolkit to enhance management of Eucalyptus cloud infrastructure and to increase accessibility of engineering software. The performance of the developed cloud infrastructure has been assessed testing CPU, memory IO, disk IO, network and the deployed software services. The obtained results have been compared with the performance of the native hardware. The technical implementation of deployed cloud services has been evaluated by utilising a number of use cases from different research areas of mechanical and medical engineering.

Keywords: private cloud infrastructure, platform as a service, software as a service, Eucalyptus, performance.

1. Introduction

Cloud computing is a promising paradigm delivering computing utilities as IT services. Cloud computing is where the evolution lines of the service-oriented, grid computing, utility computing, autonomic computing and virtualization paradigms meet [9]. The capabilities of different applications are exposed as sophisticated services that can be accessed over a network. A cloud service model [2] represents a layered high-level abstraction of the main classes of cloud services that can be classified into three main layers: software services, platform services and infrastructure services. Cloud computing provides the tools and technologies [6] to build IT services with much affordable prices compared to traditional computing techniques.

Enterprise level software for applications of mechanical and medical engineering is usually deployed as HPC solution [20], [3]. However, there are lot of software for engineers, which is running on desktop computers [4]. Such kind of software has some licensing restrictions and confines mobility of the user. Deployment of the engineering software on cloud infrastructure [19] increases mobility of users. Moreover, better exploitation can be achieved because clouds feature flexible management of resources.

Currently, one of the most popular cloud computing frameworks, Eucalyptus [24] implements infrastructure services enabling users to run and control virtual machine instances across a variety of physical resources. The open-source cloud platform developed as a linux-based collaborative project with the most active community is OpenStack [18]. OpenStack Compute deploys automatically the provisioned virtual compute instances, OpenStack Object Storage provides redundant storage of static objects and

OpenStack Image Service provides service discovery, registration and delivery for virtual disk images. OpenNebula [7] is an open-source distributed virtual machine manager, which focuses on data centre virtualisation and enterprise private cloud computing, and has often been used in conjunction with OpenStack and Eucalyptus. Research in cloud computing may be seen as particularly demanding with regards to infrastructure and equipment, being a barrier for entry for sections of the scientific community. Moreover, it is hardly possible to provide precise guidelines regarding the optimal platform for each type of research and applications [27].

The performance of cloud systems is a key concern [8]. Commercial testbeds are naturally realistic, but they cannot provide to the scientist dependable experiments and enough control. Cloud computing still lacks case studies and best practices on how to harness the employed technologies and to avoid the pitfalls that may result in immature design of engineering application.

The paper presents the development of the infrastructure, platforms and software services for applications of mechanical and medical engineering on a private university cloud. The main focus is on the software service level built on the top of the provided platforms to implement "use on demand" principle for cloud users. The developed services including management tools, launchers and interactive visualization programs enhance accessibility and usability of different engineering software on Eucalyptus cloud computing infrastructure. The performance analysis aimed to explore the feasibility and potential for utilizing cloud services to address computing needs of university researches in the fields of mechanical and medical engineering.

Other parts of the paper are organized as follows. In Section 2, the hosted cloud infrastructure based on Eucalyptus is described. Section 3 presents the developed services according to the cloud service model. Section 4 provides the details of the developed software services and the use cases from mechanical and medical engineering. The performance analysis of the cloud infrastructure and the developed software services is discussed in Section 5, while the concluding remarks are presented in Section 6.

2. Private cloud infrastructure

Private cloud infrastructure (Fig. 1) based on Eucalyptus open source solution [24] was developed in Vilnius Gediminas Technical University. The capabilities of the developed infrastructure include processing, storage and network resources that can be used to run platforms as well as the applications of mechanical and medical engineering developed on top of the deployed platforms. KVM hypervisor [5] was employed for hardware virtualization, which created an abstraction layer between computing resources and the applications that use them. Walrus storage service was included within Eucalyptus to store and retrieve persistent data organized as buckets and objects.

Eucalyptus adopts a hierarchical design [33]. The main cloud components are Cloud Controller (CLC) and Walrus Storage. They communicate with the Cluster Controller (CC) and the Storage Controller (SC). However, in the developed infrastructure, the Cloud Controller, the Cluster Controller (CC) and Storage Controller (SC) were hosted on the single physical server, which simplified administration and reduced communication cost. The Cluster Controller and Storage Controller communicate with the Node Controllers (NCs) through the network between machines hosting these components. Finally, virtual machines (VMs) run on the nodes that host the Node Controllers.

The Eucalyptus version 3.4.0 was used on top of CentOS release 6.4. The cloud infrastructure was composed of two different types of nodes that were connected to 1Gbps Ethernet LAN. Hardware characteristics of the first type of nodes are listed below: Intel® Core2Quad Q6600 2.40 GHz CPU, 4 GB DDR2 800 RAM and 320 GB HDD. Hardware characteristics of the second employed architecture are listed below: Intel®Core i7-3770 3.40 GHz CPU, 16 GB DDR3 1600 MHz RAM and 2x1 TB HDD.

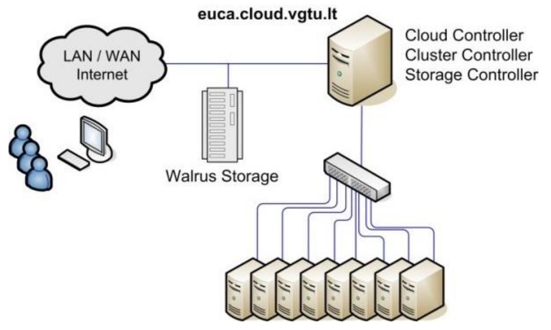

Figure 1. Private cloud infrastructure based on Eucalyptus

3. Cloud services

As the area of cloud computing was emerging, the systems developed for the cloud were stratified into three main subsets of systems. The NIST SPI model [21] classifies the services provided by the cloud systems into three main layers: Software as a Service (SaaS), Platform as a Service (PaaS) and Infrastructure as a Service (IaaS) (Fig. 2). The Eucalyptus based IaaS allows the service consumer to use infrastructure capabilities based on demand. The capabilities of the developed infrastructure include computing, storage and network, or any other basic resources that can be used to deploy and run platforms.

PaaS allows the service consumer to define, develop, configure, deploy and manage cloud applications. Operating systems, development tools and management tools are considered as part of the PaaS layer. The providers of the platform services supply the developers with a programming-language-level environment of well-defined APIs to facilitate the interaction between the environments and the cloud applications. The developers of engineering applications are supplied by Fortran programming language environment [32], because of its popularity in this particular field of interests. Other development platforms as services are provided on the basis of the popular numerical modelling software Ansys [23] and Matlab [29] that are widely used by researchers and engineers to solve different applications of mechanical and medical engineering.

Visualization Toolkit (VTK) [28] is deployed as the platform for development of visualization applications. VTK is an open source, object-oriented software library for 3D computer graphics and visualization. VTK applications are platform independent, which is very attractive for heterogeneous cloud architectures. A lot of visualization systems and environments have been developed by using VTK [10], [16]. Academic numerical software, developed by university researchers, often lacks required visualization capabilities, therefore, a wide variety of visualization algorithms provided by VTK can fill this gap on the cloud infrastructure.

User friendly tools for management of Eucalyptus infrastructure have been developed as platform services. The client is based on jclouds API [11] that provides unified access to EC2 services. Java JetS3t API [12] was used to interact with Eucalyptus Block Storage Service while HTTP protocol was used to implement the file transfer. Environment launchers have been designed for developers to configure software and to define custom settings. Furthermore, the SaaS developers can integrate user interface components of the PaaS system to their applications.

In Fig. 2, the SaaS layer contains software services developed on top of the provided platforms. The software service Grillage optimizes foundations by using the Fortran code. Another optimization software service Truss is developed by using Matlab. The special purpose visualization software VisApps is developed on top of the VTK platform to visualize the computational results of mechanical engineering. The software service HeartFlows is developed by using ANSYS Fluent to compute blood flow in aorta. Several virtual machine images with different software installations and different computational resources are prepared for SaaS users.

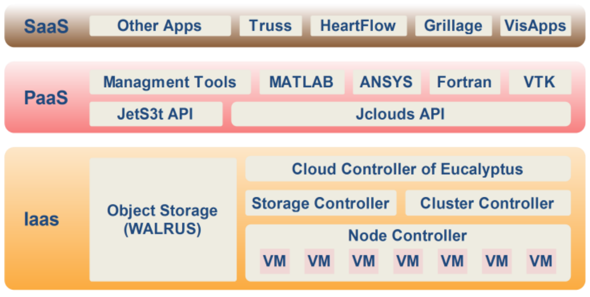

Figure 2. Cloud services

4. Developed software services

The developed software services were deployed on the cloud infrastructure to address needs of university researches in the fields of mechanical and medical engineering.

4.1. Visualization software service VisApps

Visualization software as a cloud service is developed on the top of the VTK platform to visualize results of computations. VTK is the largest open source toolkit perfectly suitable for fast development of the visualization software. On the developed cloud infrastructure, visualization software services are provided by special purpose small visualization programs empowered by VNC [26] or more universal visualization e-services based on other technologies of remote rendering [25], [15]. The developed visualization software services allow users to perform remote interactive analysis of computational results in the areas of mechanical and medical engineering, which is the challenging goal and the advanced feature of remote computational infrastructures.

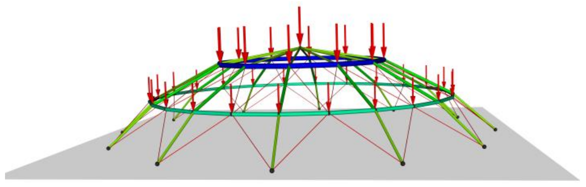

Figure 3. Visualization of a 120-bar dome truss

Fig. 3 shows the render window of the simple visualization software developed for Truss application, which performs sizing and shape optimization. It contains visualization of a 120-bar dome truss. Bars are coloured depending on the computed cross-sectional area. Vertical loads at unsupported joints are represented by red arrows.

4.2. Optimization software service Truss

The software service Truss is developed for optimization of truss structures by using Matlab Genetic Algorithm Toolbox. The developed computational software service illustrates the specialized usage of the Matlab software for global optimization problems on the cloud infrastructure. The developed software service includes the application launcher, the automatic data preparation software and the interactive visualization software. In the most cases, cloud user works with interactive graphical windows of Matlab, which runs on the considered virtual machine, but the researcher can also use the interactive visualization software for analysis of the computational results ant the cloud management tools for acquiring and configuring new resources on-demand that are necessary for the next iteration of the analysis cycle.

Well-known spatial truss structure [34], [17] with 10 nodes and 25 trusses (Fig. 4) is considered as a benchmark to minimize its weight W. The optimization problem is formulated as follows:

$$\text{Minimize } W(A) = \gamma \sum_{i=1}^{n} L_i A_i \qquad (1)$$

$$\text{Subject to} \quad \delta_{\min} \le \delta_j \le \delta_{\max} \quad j=1, \dots, m$$

$$\sigma_{\min} \le \sigma_i \le \sigma_{\max} \quad i=1, \dots, n$$

$$A_{\min} \le A_i \le A_{\max} \quad i=1, \dots, n$$

where design variable A_i is a cross-sectional area of member i, n is the number of members, m is the number of nodal displacements δ_j, L_i is the length of member i, σ_i is the stress of member i and γ is the material density equal to 2767.99 kg/m^3 (0.1lb/in^3). The modulus of elasticity E= 68947.6 MPa (10^7 psi). The limiting displacement is ±8.89 mm (0.35 in) along three perpendicular axes. Compressive stress limitations are from 6.759 to 35.092 ksi. Allowable tensile stresses are taken to be 275.79 MPa (40 ksi). Symmetrical structure is modelled, therefore, 25 beams are divided into 8 groups. The nodal loads $F_i(F_x, F_y, F_z)$ are prescribed: F_1(1.0, 10.0, -5.0), F_2(0, 10.0, -5.0), F_3(0.5, 0, 0), F_6(0.5, 0, 0) (kips). Discrete design variables are selected from the list of American Institute of Steel Construction (AISC) Code – 64 standard cross-sections from 0.111 in^2 to 33.5 in^2.

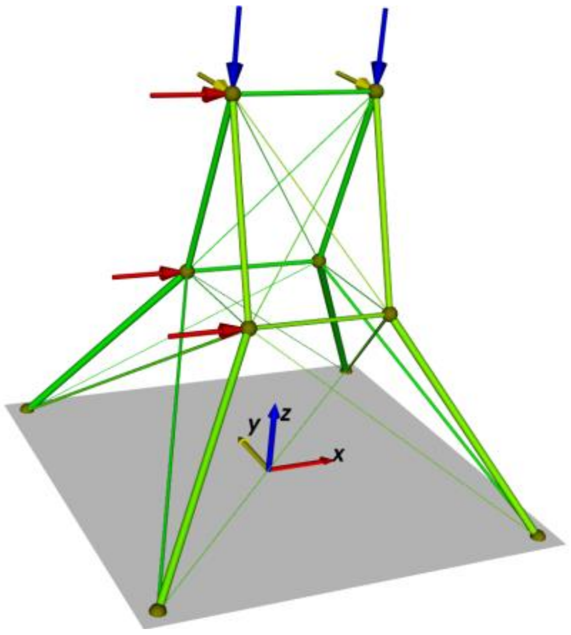

Figure 4. A 25-bar spatial truss structure

20 optimization experiments were performed (population size – 50 individuals, number of iterations – 50). Optimization results and comparisons are presented in Table 1.

Table 1. Comparison of optimal solutions

Variables (in^2)	Wu [34]	Li [17]	This work
A_1	0.307	0.111	3.470
A_2-A_5	2.130	2.130	0.111
A_6-A_9	2.930	2.880	3.630
A_{10}-A_{11}	0.111	0.111	0.196
A_{12}-A_{13}	0.111	0.111	0.196
A_{14}-A_{17}	0.563	0.766	0.785
A_{18}-A_{21}	1.620	1.620	1.228
A_{22}-A_{25}	2.880	2.620	4.180
Weight (lb)	553.915	551.14	561.78
Weight (kg)	251.25	249.99	254.82

4.3. Optimization software service Grillage

The software service Grillage is designed to optimize reactions of supports in grillages. In previous research, grillage optimization software was deployed on computer clusters [31] and grids [30]. The developed cloud services provide the interactive exploration of computational results and the flexible acquirement of necessary resources available on-demand that are the main advantages of cloud computing against the other computational infrastructures employed for optimization of grillages. Moreover, resources of virtual machines and interactive graphical environment of the universal Fortran platform service can be effectively used for new code development and enhancement of the software service.

The objective function $P(x)$ is computed as the biggest difference between the vertical and allowable reaction of the pole:

$$\text{Minimize } P(x) = \max_{1 \le i \le N_s} \left| R_i(x_i) - R_{allowable} \right| \qquad (2)$$

where design variable x_i is a coordinate of support i, $R_{allowable}$ is allowable reaction, R_i is calculated reaction, N_s is the number of supports. Design variables are restricted by the topology of a grillage.

Finite Element Method is employed to calculate values of objective function (2). Results of optimization are presented in the Table 2 and Fig. 5, where supports are represented by blue springs and loads are visualized by arrows.

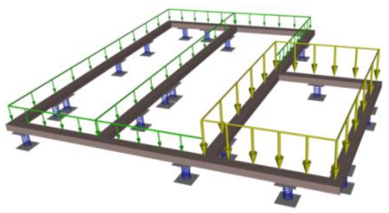

Figure 5. The optimized grillage

Table 2. Results of computations

Support No	Coordinate	Support No	Coordinate
1	2.02	10	39.67
2	7.16	11	42.04
3	6.10	12	33.85
4	47.34	13	0.56
5	12.44	14	20.70
6	13.22	15	41.22
7	18.67	16	46.44
8	21.49	17	25.95
9	25.18	18	33.13

4.4. Computational software service HeartFlows

Blood flow through aortic valve can be described by numerical models based on viscous incompressible Navier-Stokes equations [13]. The developed software service HeartFlows based on Ansys Fluent platform is applied for numerical analysis of low flow, low pressure gradient aortic stenosis, which is a highly challenging condition in terms of diagnosis and therapeutic management [1]. Computational flow analysis is a representative example of Ansys Fluent usage in various fields of engineering [23]. Advanced software and large computational times have been associated with the numerical analysis of complex flows [3], [14]. Thus, cloud infrastructure and software services are perceived as a promising avenue for future advances in this multidisciplinary area of medical engineering. Large computations can be carried out on clusters of virtual machines acquired on-demand and managed by the Eucalyptus Cluster Controller (Fig. 2) while the interactive exploration of computational results can be performed by using the developed software services on a single virtual machine saving computational resources. Moreover, semi-automatic medical image segmentation, which requires user interaction, can also be performed on virtual machines employing interactive graphics services to obtain patient-specific geometry.

Fig. 6 shows the screenshot of Ansys Fluent software illustrating the flow through aortic valve in the vertical cross-section. The colour map visualizes distribution of the pressure field. Computed velocity field is represented by arrow glyphs. The complex flow pattern with vortexes following the open valve can be observed in aortic root. The pressure drop is the main target of computations in the case of low pressure gradient aortic stenosis.

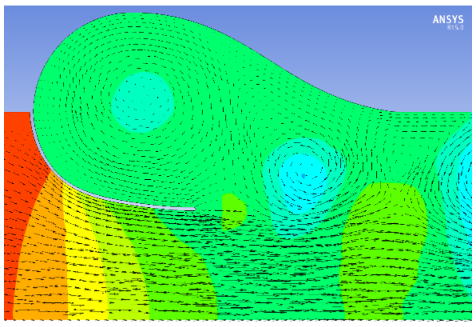

Figure 6. Blood flow in a cross section of the aortic root

5. The performance analysis

The performance analysis aimed to explore the feasibility and potential for utilizing cloud infrastructure. It is very important to know how well data are treated amongst disk, CPU, memory and network since bottlenecks in such virtualized resources will likely lead to bottlenecks in applications.

5.1. Memory Performance

The STREAM benchmark is a synthetic benchmark program that measures sustainable memory bandwidth and the corresponding computation rate for simple vector kernels. The problem size of 150 million is considered to work with datasets much larger than the available cache on any allocated system.

As shown in Fig. 7, there are no significant performance variations among different memory operations. The largest observed difference in bandwidth measured on the native machine and the virtual machine was equal to 3.9% in the case of ADD operation. The smallest measured difference was equal to 2.8% of the native bandwidth.

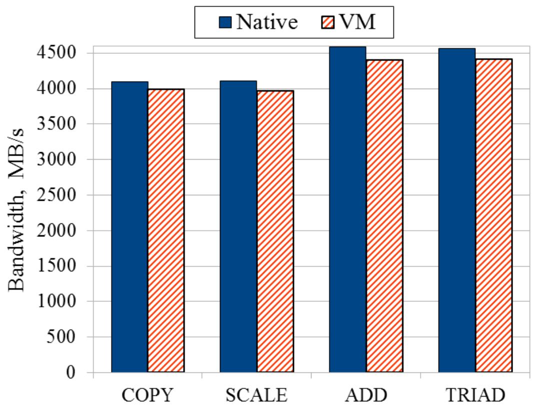

Figure 7. Memory bandwidth measured by STREAM

5.2. Network performance

The performance of virtual networking in Eucalyptus infrastructure was examined measuring the network bandwidth and the round-trip time. Iperf was employed to measure the network bandwidth, while Ping was used to obtain the round-trip time. The results of the networking benchmark are presented in Table 3. They are arranged that each row presents measurements of a different network connection. The first line shows the performance of the network communication among the virtual machine and the external server (VM<->Server). Hardware of the external server is different than hardware hosting the virtual machine, therefore, two-way communication is examined. The performance of the network among the native CentOS machine and the external server is presented for the sake of comparison in the second row (Native<->Server). The performance of the virtual network connecting two identical virtual machines was also investigated in the third row (VM<->VM). There are two sets of measurements in Table 3: network bandwidth measured by Iperf and the averaged round-trip time obtained by Ping.

Table 3. Network performance

Software	Iperf, Mbit/s		Ping, ms	
Communication	X->Y	X<-Y	X->Y	X<-Y
VM<->Server	767	653	0.575	0.599
Native<->Server	932	923	0.357	0.307
VM<->VM	913	913	0.550	0.550

The quantitative comparison of the performance of the virtual network shows relatively disappointing results. The network bandwidth among VM and the external server made only 82% of the network bandwidth among the native CentOS machine and the external server. Other authors [8] also pointed out that networking, especially receiving from the external servers, remained a bottleneck in cloud infrastructures. The measurements of the averaged round trip time revealed even more discouraging results. However, only minor overheads (up to 2%) were observed measuring the bandwidth of the virtual network among the virtual machines of the Eucalyptus cloud.

5.3. Disk Performance

Researchers generate and analyse large data sets to derive scientific insights. Understanding the disk I/O performance is critical to understand the performance of scientific applications in cloud infrastructures. In the presented study, two different kinds of data storage are addressed: instance storage, which is the disk physically installed in the node hosting VM, and attached volume storage, which is mounted on Elastic Block Storage. The obtained I/O performance of virtualized data storages is compared with that of the native hardware. The disk I/O performance is measured with the file system benchmark tool IOzone, which can generate and measure a variety of file operations. We use automatic mode, file size of 512MB and 64kB record size.

Table 4 shows the main results of the disk buffered I/O tests in MB/s. The six performance metrics (read, re-read, random read, write, rewrite, random write) are presented in the table. In general, the disk I/O performance on the local instance made more than 90% of the native performance. However, it does not hold for read operation, which reveals slightly lower performance equal to 86%. **Clearly,** the local disk on the instance performs better than the EBS volumes on the VMs, which is due to the relatively low network bandwidth. The observed **performance** difference among VM and EBS volume did not exceed 23% of the disk I/O on the local instance storage.

Table 4. Disk I/O performance

IOzone	read	reread	ranread	write	rewrite	ranwrite
Native	3084	4059	3895	61	64	62
VM	2665	3716	3623	57	61	61
Volume	2140	3714	3574	46	49	47

5.4. CPU and application performance

LINPACK benchmark, which measures the floating point operations per second (flop/s) for a set of linear equations, was performed to evaluate the general CPU performance. The pre-compiled version of LINPACK from the Intel website was applied for benchmarking unoptimized performance with default parameters since this is what the typical user would get on cloud infrastructures. In average, the CPU performance on the virtual machine made only 76% of the native performance. However, it is more important to measure the time needed to compute the benchmarks of the investigated codes. Thus, the following results reveal performance with respect to the developed software services Grillage, Truss and HeartFlow. The relatively small benchmark problems with limited execution time were considered in order to make quantitative comparison.

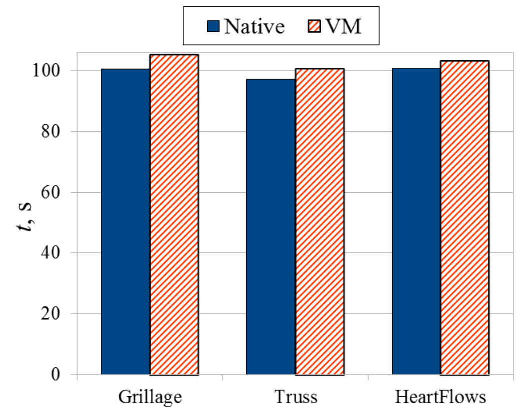

Figure 8. The performance of the considered applications

The measured performance of application benchmarks (Fig. 8) revealed that the performance difference of the native hardware and the virtual machine varies from 2.4% to 4.7% of the benchmark time on the native machine. The largest difference was obtained in the case of the highly efficient Fortran code Grillage, which was able to exploit the advantages of the native hardware. Higher level codes showed good performance on the virtualized resources in the case of the considered benchmark.

5.5. Launch time of virtual machines

One great advantage of the cloud is the elasticity, the ability to dynamically provision resources in response to demand. A cloud resource generally and virtual machine instance in particular has several stages, all of which can take time and so can attract cost. The sequence from the instant request to the end of the CentOS image boot was considered to be the launch time of VM in the performed benchmark. The VM launch time for a multiple-instance (up to 8-instance) acquisition request was investigated in the case of two nodes with eight cores of physical hardware.

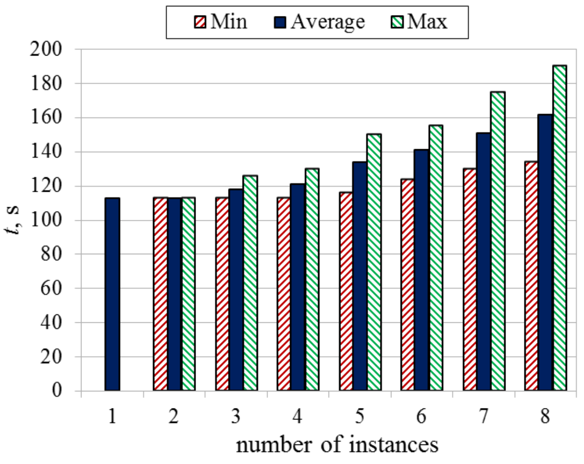

Figure 9. Virtual machine launch time

The VM launch time for the multiple-instance acquisition requests is presented in Fig. 9. Minimal, average and maximal launching times of VMs are shown. The variance of launch time depended on the number of acquired instances. The standard deviation reached 15% of the averaged launch time in the case of the 8-instance acquisition request. The average launch time of VM became up to 43% longer, when a large number of instances is acquired at the same time. It can be explained by the performance reduction of the highly loaded hardware.

6. Conclusions

In this paper, the infrastructure, platform and software have been developed to provide applications of mechanical and medical engineering as services within private cloud. The results of the performance analysis and quantitative comparison with the native hardware have revealed good potential of the developed infrastructure. The investigated performance of the virtual machines of the cloud infrastructure has been close to the performance of the native hardware measuring memory bandwidth, disk I/O and bandwidth of the local virtual network. However, the significant overhead has been observed when virtual machines communicated with the external server. CPU performance measured with LINPACK has been significantly lower in comparison with the native hardware, but only minor overheads have been observed measuring performance of the developed software services. The developed cloud computing solution has improved accessibility of the engineering software for the researchers, students and teachers. The trials from the case study have shown that the developed cloud infrastructure and the deployed services are able to satisfy needs of researchers in diverse fields of engineering.

Acknowledgements

Simulations of blood flow through the aortic valve are supported by the Research Council of Lithuania in the framework of the project "Patient-specific simulation of the flow through the aortic valve" (MIP-052/2014).

References

[1] **J. Adda, C. Mielot, R. Giorgi, F. Cransac, X. Zirphile, E. Donal, C. Sportouch-Dukhan, P. Réant, S. Laffitte, S. Cade, Y. Le Dolley, F. Thuny, N. Touboul, C. Lavoute, J.F. Avierinos, P. Lancellotti, G. Habib.** Low-flow, low-gradient severe aortic stenosis despite normal ejection fraction is associated with severe left ventricular dysfunction as assessed by speckle-tracking echocardiography: a multicenter study. *Circulation: Cardiovascular Imaging*, 2012, Vol. 5, No. 1, 27–35.

[2] **S.A. Ahson, M. Ilyas.** Cloud Computing and Software Services: Theory and Techniques. *CRC Press, London*, 2011.

[3] **S. Bastrakov, I. Meyerov, V. Gergel, A. Gonoskov, A. Gorshkov, E. Efimenko, M. Ivanchenko, M. Kirillin, A. Malova, G. Osipov, V. Petrov, I. Surmin, A. Vildemanov.** High Performance Computing in biomedical applications. *Procedia Computer Science*, 2013, Vol. 18, 10–19.

[4] **T.H. Beach, O.F. Rana, Y. Rezgui, M. Parashar.** Cloud computing for the architecture, engineering & construction sector: requirements, prototype & experience. *Journal of Cloud Computing: Advances, Systems and Applications*, 2013, Vol. 2, No. 1, 1–8.

[5] **T. Deshane, Z. Shepherd, J.N. Matthews, M. Ben-Yehuda, A. Shah, B. Rao.** Quantitative comparison of Xen and KVM. *Xen Summit, June 23-24, Boston, MA, USA*, 2008.

[6] **R. Dukaric, M.B. Juric.** Towards a unified taxonomy and architecture of cloud frameworks. *Future Generation Computer Systems*, 2013, Vol. 29, No. 5, 1196–1210.

[7] **J. Fontan, T. Vazquez, L. Gonzalez, R. Montero, I. Llorente.** OpenNebula: the open source virtual machine manager for cluster computing. *In: Open Source Grid and Cluster Software Conference, San Francisco, CA, USA,* 2008.

[8] **L. Gillam, B. Li, J.O'Loughlin, A.P. Tomar.** Fair benchmarking for cloud computing systems. *Journal of Cloud Computing: Advances, Systems and Application*s, 2013, Vol. 2, No. 1, 1–6.

[9] **M. Hamdaqa, L. Tahvildari.** Cloud computing uncovered: a research landscape. *Advances in Computers,* 2012, Vol. 86, 41–85.

[10] **C.D. Hansen, C.R. Johnson.** The Visualization Handbook. *Elsevier, Amsterdam*, 2005.

[11] **Jclouds:** http://www.jclouds.org/.

[12] **JetS3t:** http://jets3t.s3.amazonaws.com/toolkit/toolkit.html.

[13] **A. Kačeniauskas.** Development of efficient interface sharpening procedure for viscous incompressible flows. *Informatica,* 2008, Vol. 19, No.4, 487–504.

[14] **A. Kačeniauskas.** Solution and analysis of CFD applications by using grid infrastructure. *Information Technology and Control*, 2010, Vol. 39, No. 4, 284–290.

[15] **A. Kačeniauskas, R. Pacevič.** VizLitG: grid visualization e-service enabling partial dataset transfer from storage elements of glite-based grid infrastructure. *Journal of Grid Computing*, 2011, Vol. 9, No. 4, 573-589.

[16] **A. Kačeniauskas, R. Pacevič, A. Bugajev, T. Katkevičius.** Efficient visualization by using ParaView software on BalticGrid. *Information Technology and Control*, 2010, Vol. 39, No. 2, 108–115.

[17] **L.J. Li, Z.B. Huang, F. Liu.** A heuristic particle swarm optimization method for truss structures with discrete variables. *Computers & Structures*, 2009, Vol. 87, 435–443.

[18] **O. Litvinski, A. Gherbi**. Experimental evaluation of OpenStack compute scheduler. *Procedia Computer Science*, 2013, Vol. 19, 116–123.

[19] **T. Ludescher, T. Feilhauer, P. Brezany.** Cloud-based code execution framework for scientific problem solving environments. *Journal of Cloud Computing: Advances, Systems and Applications*, 2013, Vol. 2, No. 1, 1–11.

[20] **V. Mauch, M. Kunze, M. Hillenbrand**. High performance cloud computing. *Future Generation Computer Systems*, 2013, Vol. 29, No. 6, 1408–1416.

[21] **P. Mell, T. Grance.** The NIST definition of cloud computing. Recommendations of the National Institute of Standards and Technology Special Publication 800-145, National Institute of Standards and Technology, USA, 2009.

[22]

[23] **Y. Nakasone, S. Yoshimoto, T.A. Stolarski.** Engineering Analysis with ANSYS Software. *Elsevier Ltd.*, 2006.

[24] **D. Nurmi, R. Wolski, C. Grzegorczyk, G. Obertelli, S. Soman, L. Youseff, D. Zagorodnov.** The Eucalyptus open-source cloud-computing system. In: *The 9th IEEE/ACM International Symposium on Cluster Computing and the Grid*, 2009, pp. 124–131.

[25] **M. Polak, D. Kranzlmüller**. Interactive videostreaming visualization on grids. *Future Generation Computer Systems*, 2008, 24(1), 39–45.

[26] **T. Richardson, Q. Stafford-Fraser, K.R. Wood, A. Hopper**. Virtual network computing. *IEEE Internet Computing,* 1998, 2(1) 33–38.

[27] **G. Sakellari, G. Loukas.** A survey of mathematical models, simulation approaches and testbeds used for research in cloud computing. *Simulation Modelling Practice and Theory*, 2013, Vol. 39, 92–103.

[28] **W. Schroeder, K. Martin, B. Lorensen.** Visualization Toolkit: An Object-Oriented Approach to 3D Graphics, 4th Edition. *Kitware. Inc.*, 2006.

[29] **S.K. Sen, G.A. S haykhian**. MatLab tutorial for scientific and engineering computations. *Nonlinear Analysis: Theory, Methods & Applications*, 2009, Vol. 71, No. 12, 1005–1020.

[30] **D. Šešok, R. Belevičius, A. Kačeniauskas, J. Mockus**. Application of GRID computing for optimization of grillages. *Mechanika*, 2010, Vol. 82, No. 2, 63–69.

[31] **D. Šešok, J. Mockus, R. Belevičius, A. Kačeniauskas.** Global optimization of grillages using simulated annealing and high performance computing. *Journal of Civil Engineering and Management: International Research and Achievements*, 2010, Vol. 16, No. 1, 95–101.

[32] **L.J. Slooten, F. Batle, J. Carrera**. An experimental approach to the performance penalty of the use of classes in Fortran 95. *Advances in Engineering Software*, 2011, Vol. 42, No. 10, 735–742.

[33] Understanding the Eucalyptus Architecture. Eucalyptus Ins*tallation Guide. Eucalyptus Systems, Inc*., 2014.

[34] **S-J. Wu, P-T. Chow**. Steady-state genetic algorithms for discrete optimization of trusses. *Computers & Structures*, 1995, Vol. 56, No. 6, 979–991.

Received June 2014.