Implementing and Running a Workflow Application on Cloud Resources

Gabriela Andreea MORAR, Cristina Ioana MUNTEAN, Gheorghe Cosmin SILAGHI
Babes-Bolyai University, Cluj-Napoca, Romania
{gabriela.morar, cristina.muntean, gheorghe.silaghi}@econ.ubbcluj.ro

Scientist need to run applications that are time and resource consuming, but, not all of them, have the requires knowledge to run this applications in a parallel manner, by using grid, cluster or cloud resources. In the past few years many workflow building frameworks were developed in order to help scientist take a better advantage of computing resources, by designing workflows based on their applications and executing them on heterogeneous resources. This paper presents a case study of implementing and running a workflow for an E-bay data retrieval application. The workflow was designed using Askalon framework and executed on the cloud resources. The purpose of this paper is to demonstrate how workflows and cloud resources can be used by scientists in order to achieve speedup for their application without the need of spending large amounts of money on computational resources.

Keywords: *Workflow, Cloud Resource*

1 Introduction

Nowadays, scientific world is evolving rapidly and larger and more resource intensive applications are implemented on a daily bases all around the scientific and academia domains. Simultaneously, the need of computational resources that are available at any time and for low prices has increased. Scientists want to optimize the execution time of their application, but not all of them dispose of the required programming skills to implement and run parallel applications, or they lack the needed resource infrastructures. On some occasions, the available resources require advance scheduling and their heterogeneity makes their usage even more complex and difficult to handle by scientific applications. One of the solutions scientists have in order to get a better execution time and use heterogeneous resources are scientific workflows [1]. They are described as a series of structured activities and computations that arise in scientific problem-solving. In many science, engineering and business domains, the use of computation is not only heavy, but also complex and structured with intricate dependencies.

Workflows are not only used in the scientific world, they are largely used in business world too in order to optimize business processes. Testing different kind of workflows and their performance will one day help improve the way business processes are run. The workflow used as an example in this paper can be easily substituted by a business workflow, which could achieve better results by using cloud resources.

During the past few years an increasing number of frameworks for scientific workflow developments have been implemented. Some of the most popular are DAGMan [2], Pegasus [3], Triana [4], ICENI [5], Taverna [6], GridAnt [7], GridFlow [8], Gridbus [9], Askalon [10] and Kepler [11]. A detailed comparison among this workflow management projects can be found in [12]. Most of them have been developed in order to run workflows on grid resources, but lately the majority of them moved towards incorporating the use of cloud resources also.

Usually these frameworks allow the usage of different types of resources, on which the workflows can be executed: grid resources, clusters, supercomputers, and cloud resources. In our case study example the workflow will be implemented using the Askalon [13] framework, and the test will be run on cloud resources.

The present paper is structured as follows: in section 2 we present background and related work, section 3 overviews the application based on which the workflow is build,

section 4 enumerates the steps covered in order to implement the workflow, section 5 describes the conducted experiments and their results, and section 6 consists of the conclusions.

2 Background and related work

In this section we will introduce the cloud computing concept and related work concerning framework to support mapping of workflow applications on cloud resources.

Cloud Computing is the buzz work in the industry and is the newest form of SOA. In [14] Cloud computing is defined as:

"*A computing Cloud is a set of network enabled services, providing scalable, QoS guaranteed, normally personalized, inexpensive computing platforms on demand, which could be accessed in a simple and pervasive way*".

The basic types of cloud computing available on the market are: IaaS (also known as Haas), SaaS, PaaS, and Daas. The relations existing among them can be seen in Figure 1.

In [15] is described as large set of computing resources, such as storing and processing capacity, that are IPs managed. Through virtualization, they are able to split, assign and dynamically resize these resources to build ad-hoc systems as demanded by customers, the SPs. They deploy the software stacks that run their services.

SaaS [16] refers to providing on demand applications over the Internet. All the applications that run on the Cloud and provide a direct service to the customer are located in the SaaS layer. These application developers can either use the PaaS layer to develop and run their applications or directly use the IaaS infrastructure.

Daas is describe in [14] as being data in various formats and from multiple sources could be accessed via services by users on the network. Users could, for example, manipulate the remote data just like operate on a local disk or access the data in a semantic way in the Internet.

According to [17] PaaS refers to providing platform layer resources, including operating system support and software development frameworks.

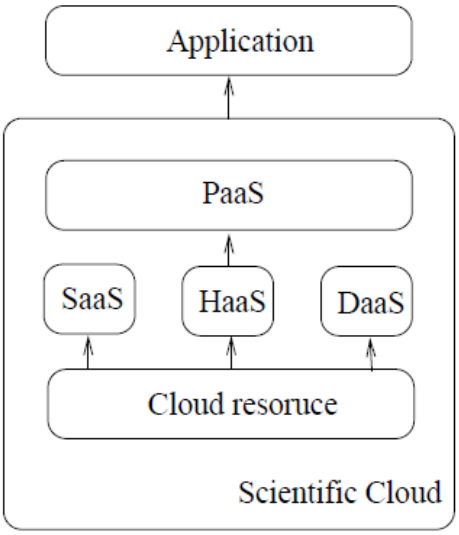

Fig. 1. Cloud models

In the past few years Cloud computing has gain a lot of appreciation from academia and research world. It is now seen as a more viable source of computational resources for scientific workflow execution, due to reduce cost, ease of management, ease of access (no more waiting for certificates in order to use grid resources, clusters, etc that did not belong to the person/group that needed them), and the possibility to install all needed software for running applications without any implication of a third party. Until now, scientific application that needed significant amounts of resources in order to perform at their true power were relying on cluster resources, grid resources, supercomputers, but not all research centers have at their disposals this large amount of computational power. Once with fast development of private and commercial cloud providers, the resource provisioning activity is no longer a time issue, but rather an issue of finding the best resources provider and optimizing costs. Here is where Cloud computing comes in, and most precisely IaaS. In the IaaS cloud model researchers are allowed to create their own cloud images with the needed performance characteristics, install and manage all the programs running on these images. Also they can instantiate different

numbers of instances, based on the previously created images, thus the need of installing software products on every resource used is inexistent. And they get all of this at reduced costs in comparison with the cost of buying and maintaining infrastructures of their own. Usually cloud providers offer cloud images that have already been configured and provided with certain software, so that the time elapsed to create the needed images can sometimes be speared.

In [18] E U C A LY P T U S (Elastic Utility Computing Architecture Linking Your Programs to Useful Systems) is described as an open source soft ware infrastructure for implementing on-premise clouds on existing Enterprise IT and service provider infrastructure. Eucalyptus enables hybrid cloud and private cloud deployments for enterprise data centers and requires no special purpose hardware or reconfiguration. Leveraging Linux and web service technologies that commonly exist in today's IT infrastructure, Eucalyptus allows customers quickly and easily to create computing clouds "on premise" that are tailored to their specific application needs.

In the past years several case studies were made on using cloud resources for running scientific workflows. In [19] an example of using combined grid and cloud resources for running three different workflows is presented, the accent lies more on the optimization of cost by carefully choosing the number and size of cloud instances used the toll used in this case in GroudSim [20].

In [21] an example of using cloud resources with the Aneka workflow management system is presented. The workflow used as a case study is based on EMO (Evolutionary Multi-objective Optimization) application that is based on a genetic algorithm. The emphasis in this paper is on the changes that workflow developing frameworks need to take into consideration when moving towards cloud resource usage.

Another example of using cloud resources for running scientific workflows is presented in [22]. The workflow management system used for running the workflows is Pegasus. Three different workflows are tested on cloud resources: Montage – an astronomy application that is more I/O intensive, Epigenom – a bioinformatics application that requires a high CPU usage, Broadband – an earthquake science application that requires large memory usage. The performance of these three workflows is tested on different numbers and types of Amazon EC2 instances.

Although case studies in the area of cloud resource usage have already been performed, the suitability of cloud resources for scientific workflow execution is a rather new topic, and more testing and case studies need to be accomplished in order to be able to say in which case, what kind of cloud resources should be use in order to reduce costs without giving up on performance.

3 Application description

This section described the baseline workflow application we used for our study. This application is implemented in Python and has as a final purpose user's data retrieval from E-bay. More precisely the application retrieves data regarding positive, negative and neutral appreciations of buyers regarding different sellers. Basically the application goes through a list of E-bay sellers and collects all the reviews they received by collecting the name of the user that placed the review and the type of the review. Due to the large number of sellers for which this data is being retrieved, we have identified the need of executing the data retrieval process in a parallel manner in order to improve the execution time. One of the most efficient way of doing so, without needing to use high parallel programming skills, is to implement a workflow that will contain one/more parallel sections that will allow parallel execution of activities without extensive programming being necessary. The data retrieved by the application will be later used for further analysis.

The data obtained by running the application can be modeled as a social network. This collection of users (buyer and seller) can be

represented as nodes of a graph with negative, neutral and positive edges according to the type of reference each user gives or receives. By employing social network analysis techniques we can obtain different kinds if information (i.e. authoritativeness, clustering, centrality) about E-bay users. The granularity of the collected information can be at the level of singles users or communities of users. For example usually a user in E-bay is ranked by his "feedback" from other users. By observing the complex relations obtained from a link analysis of the graph we can obtain rich information on power sellers, active buyers or trust within the product category. At the same time this data can give us an idea of cooperating users, malicious users and possible fraud strategies.

Through link analysis we can also observe the formation of strongly connected communities and infer ascending or descending shopping trends in the medium. In a certain time frame we can analyze the fluctuations in specific markets.

4 Preparing the application for the workflow

The workflow used for this case study was implemented in Askalon workflow management system. Askalon's components can be seen in Figure 2.

In order to be able to create a workflow for an application several steps need to be followed, this steps are presented as a suggestion in [13]:

1. Identify sections of the application that have the least connections one with the other. These sections will become independent activities in the workflow.
2. Establish sections of the code that could be run in parallel, these sections will be included in a parallel section of the workflow.
3. If some activities, previously defined, have too many correlations among them, they should be grouped together in a single activity.
4. Input and output data needed for each activity should to be identified
5. Correlations among activities must be established

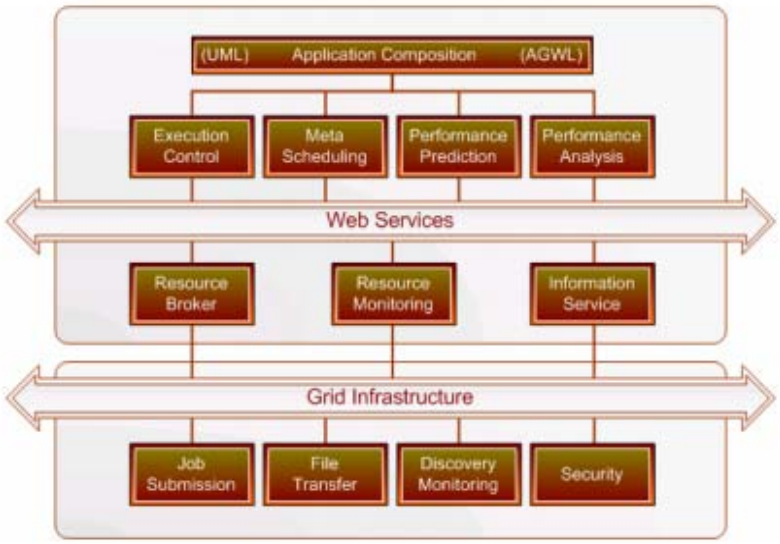

Fig. 2. Askalon's workflow management system components [13]

Based on the previous described steps our application was divided into three different activities: Nrusers, ParallelSec, CopyFiles. Figure 3 presents the workflow designed for the E-Bay data retrieval application.

Nrusers is the activity in which the number of sellers is being computed based on the input file that contains the list with all the sellers. The number of sellers is not fixed because the content of the input file containing the sellers' user names can vary.

This is one of the steps that ensure the scalability of the application. In this same activity the large file containing the users' names is divided in several different files that will serve as input for each activity in the parallel section. At the end of the activity the new files created are output and also a file named nr_parallel.txt that contains the number of file grouping the user's names is created. The number of names contained in a file can vary according to user's preferences.

ParallelSec is the section of the application that was identified as being parallel. In this activity, data about the sellers is being retrieved based on the names contained by the files previously created in the Nrusers activity. We have decided to run parallel activities for group of users in order to avoid unnecessary overload caused by multiple files transfer, due to the fact that the time needed to retrieve the data related to a single user name is small in comparison with the time needed for the files transfers. The parallel section has a number of parallel activities equal to the number of user's names files created in the previous activity. The output of this activity consists in two collections of files containing the reviews and references owned by each user.

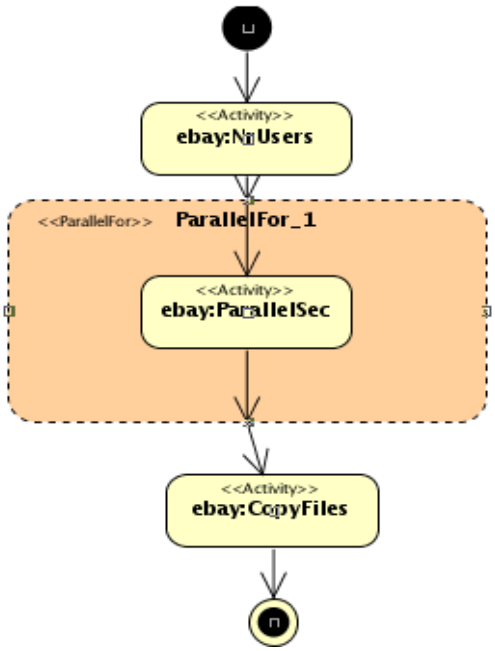

Fig. 3. Workflow for the E-bay data retrieval application

The last activity, CopyFiles, has as a master purpose the gathering off all the data contained in the two collections produced by the parallel for section of the workflow. The resulted workflow is presented in Figure 3.

5 Experiments and results

After the workflow was implemented, it was tested using different types of cloud resources. The test were run using 6000 and 60000 users' names, the groups of users used as input for each parallel activity was of 50/150 users, resulting in a total of 120 and 400 parallel activities. The tests were run on different numbers and sorts of cloud resources. The purpose of the test was to see how increasing the computational resources number and size will affect the speedup of the application. Also by doubling the number of parallel users/activities for the second series of test we what to see how increasing the problem size will affect the results. The cloud we employed is a Eucalyptus private academic cloud. The available types of instances are presented in the following table:

Table 1. Types of available cloud instances

vm types	RAM	CPUs
m1.small	1024	1
c1.medium	2048	1
m1.large	2048	2
m1.xlarge	4096	2
c1.xlarge	4096	4

In order to be able to see how the presence/absence of parallel activities influences the execution time, we also ran the application as it is, with no parallelization, on a m1.small cloud instance. Then, we ran the workflow that has the parallel section of 120 activities on multiple m1.small cloud instances in order to test the speedup achieved by using more CPUs at a time.

The speedup was computed using the following formula

$$S_p = \frac{T_1}{T_p}$$

Sp – Speedup value

T_1 – execution time of sequential algorithm

T_p – execution time of parallel algorithm with p processors

p – number of processors

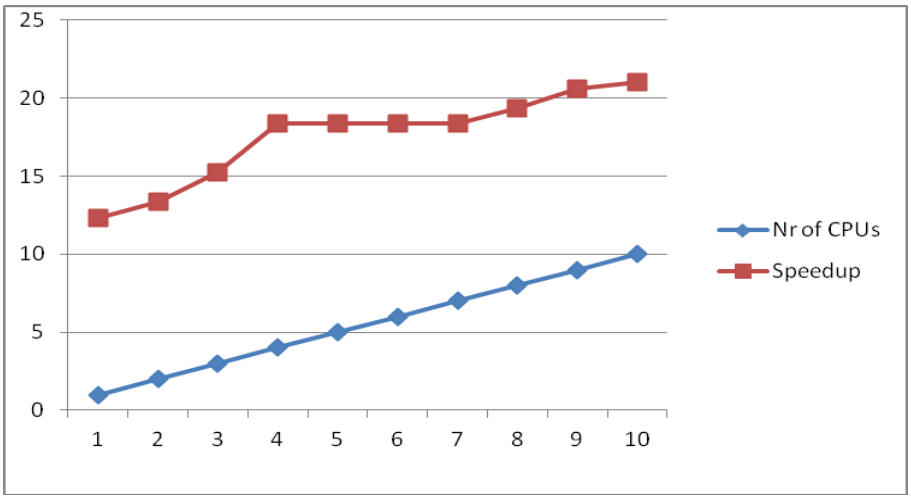

Fig. 4. Speedup achieved by using multiple m1.small cloud instances for 6000 users

For our first experiment we ran the application using m1.small cloud instances that have one CPU and 1024 RAM of memory. The maximum number of cloud instances used was 10 due to the academia's cloud limitations at our disposal. Figure 4 presents the results. By using more CPUs the speedup increases, reaching a maximum of over 21, when 10 CPUs were used. Just by running the application with parallel sections in comparison by running it in a sequential way a speedup of 12 is achieved. By using a workflow all of this has been achieved without the need of complex parallel programming being applied to the initial application.

The tested application is an I/O intensive application; it mostly reads and writes data from/to files.

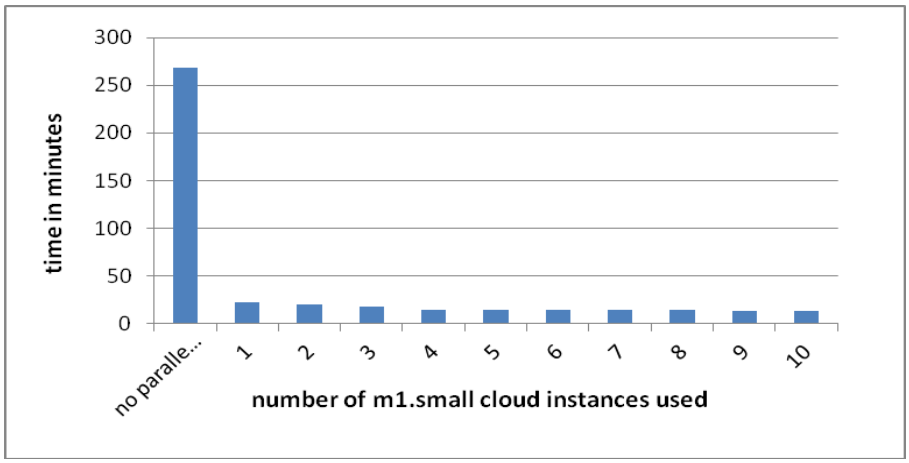

Fig. 5. E-bay workflow's execution time in minutes on m1.small instances/ comparison with execution with no parallelization for 6000 users

Figure 5 shows the time difference between several executions of the workflow using m1.small cloud instances in comparison with the time implied by running the application directly on an m1.small cloud instance without using the workflow for achieving parallelization of the execution. The execution time without parallelization is of about 268 minutes in comparison with about 13 minutes required for running the application on 10 m1.small cloud instances by using the workflow.

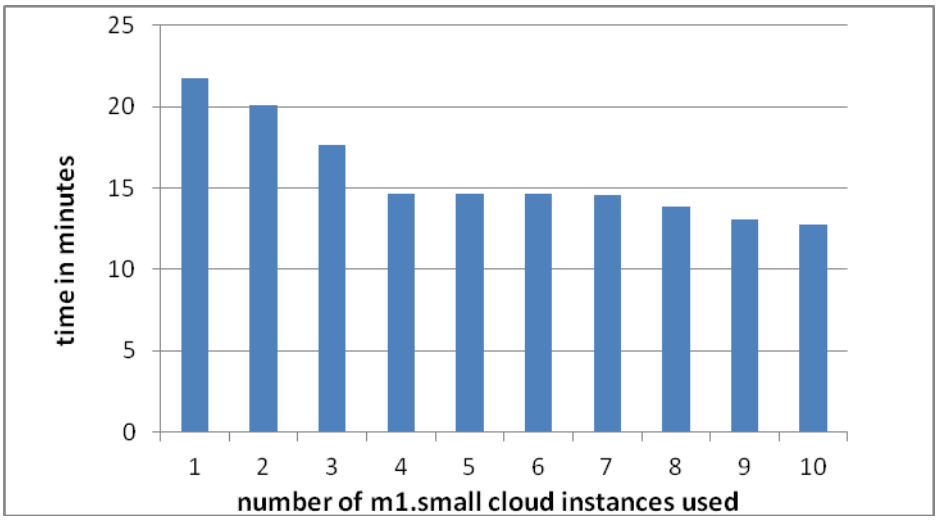

Fig. 6. E-bay workflow's execution time in milliseconds for different number of m1.small cloud instances for 6000 users

From Figure 6 we can observe that the time implied for the application to run by using increasing amounts of m1.small cloud instances is rather constant and varies only by several minutes in comparison the lowest execution time of about 13 minutes.

A second round of tests was run on m1.large cloud instances that have 2 CPUs per instance and 2048 RAM. Again, due resource limitations, the maximum number of instances was 4. The results presented in Figures 7 and 8 show that the speedup per number of CPUs is similar with the one resulted by using a similar number of CPUs but in a larger number of instances. The maximum speedup was ~23 and the lowest execution time for the workflow was, again, ~13 minutes.

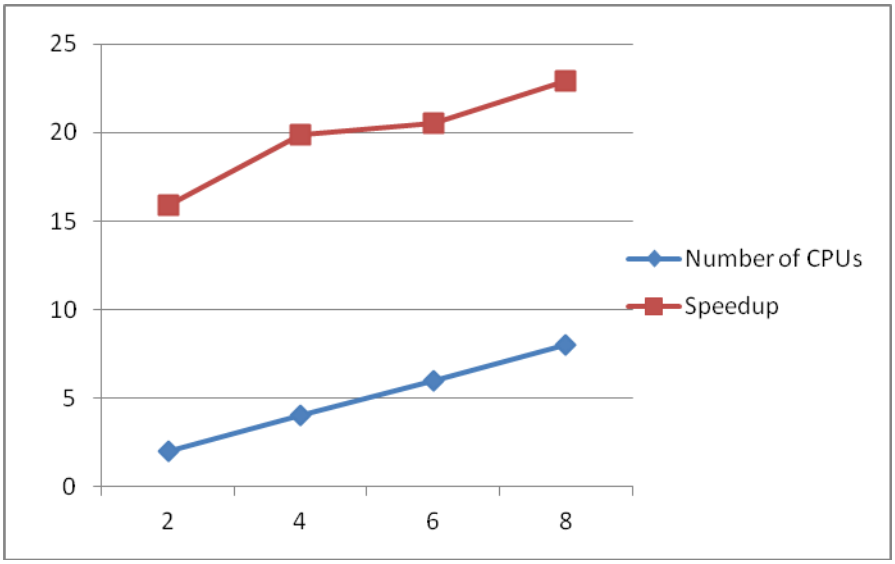

Fig. 7. Speedup based on the number of m1.large cloud instances used for 6000 users

The speedup achieved by using m1.large cloud instances can be seen in Figure 7; the maximum speedup achieved in this case was ~23 compared with the time needed for the sequential execution of the application.

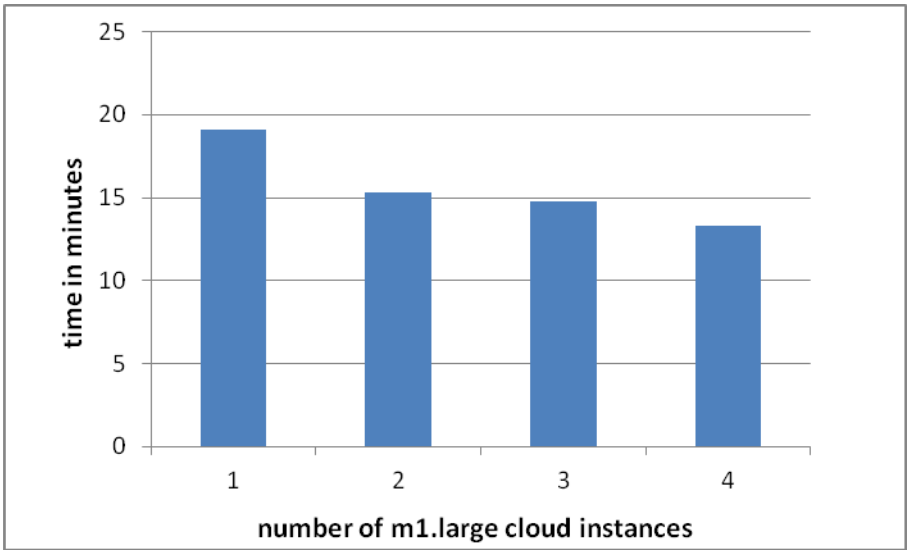

Fig. 8. E-bay workflow's execution time in minutes for different number of m1.large cloud instances for 6000 users

A comparison among time needed for the workflow execution using different number of m1.large cloud instances can be seen in Figure 8. The smallest execution time was achieved when using 4 m1.large cloud instances and it was ~13 minutes.

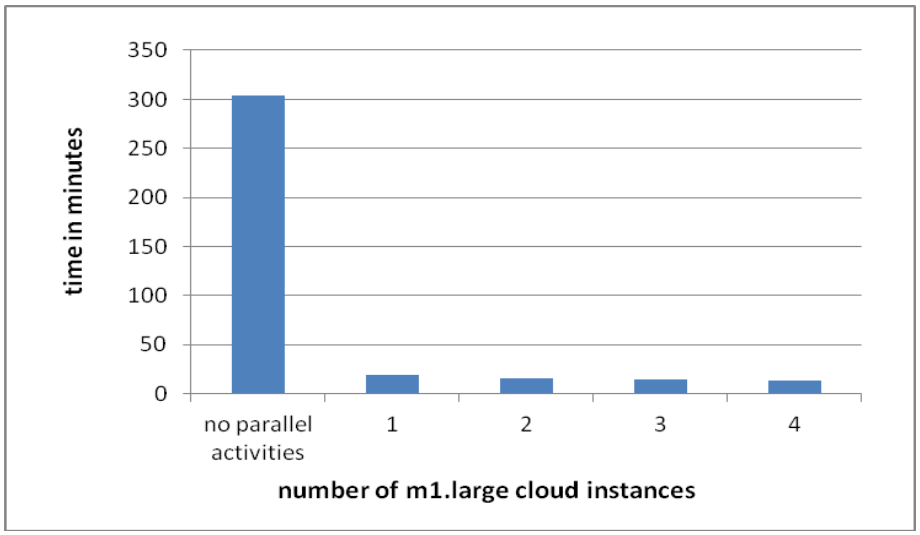

Fig. 9. E-bay workflow's execution time in minutes on m1.large instances - comparison with execution with no parallelization for 6000 users

As we can see from Figure 9, even on an m1.large cloud instance the time needed for the execution of the application with no parallelization is still rather high even if the instance disposes of larger RAM.

A third round of experiments was conducted on a number of 60000 users by using an increasing number of m1.small cloud instances. This time the number of parallel activities was 400.

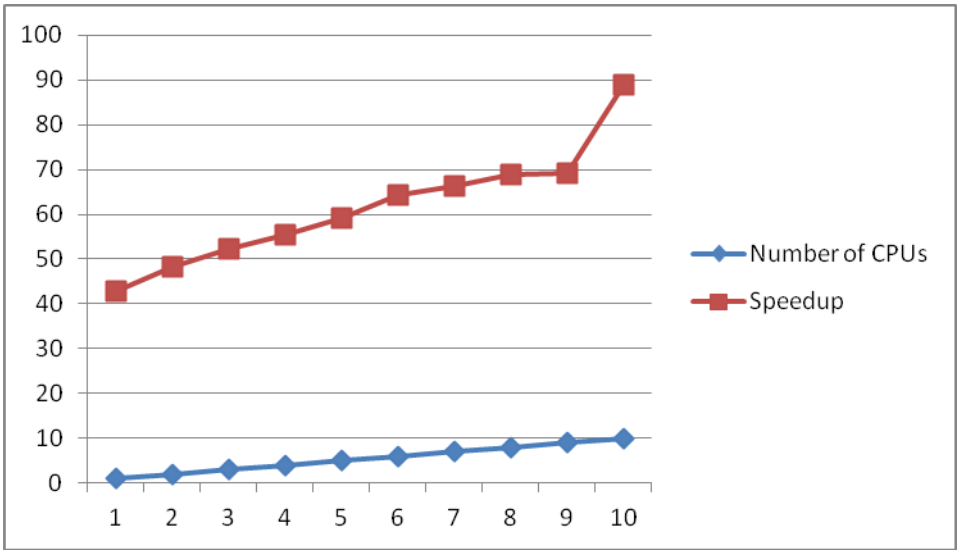

Fig. 10. Speedup achieved by using multiple m1.small cloud instances for 60000 users

From Figure 10 we can observe that by increasing the number of users and, implicitly, the number of parallel activities the achieved speedup is ~88 in comparison with the time needed for running the application in a sequential manner on a m1.small cloud instance. Thus shows that by using the cloud instances at their full potential better speedups can be achieved. During the previous experiments the number of parallel activities and the amount of computational power needed was smaller, thus resulting in a partial usage of the resources.

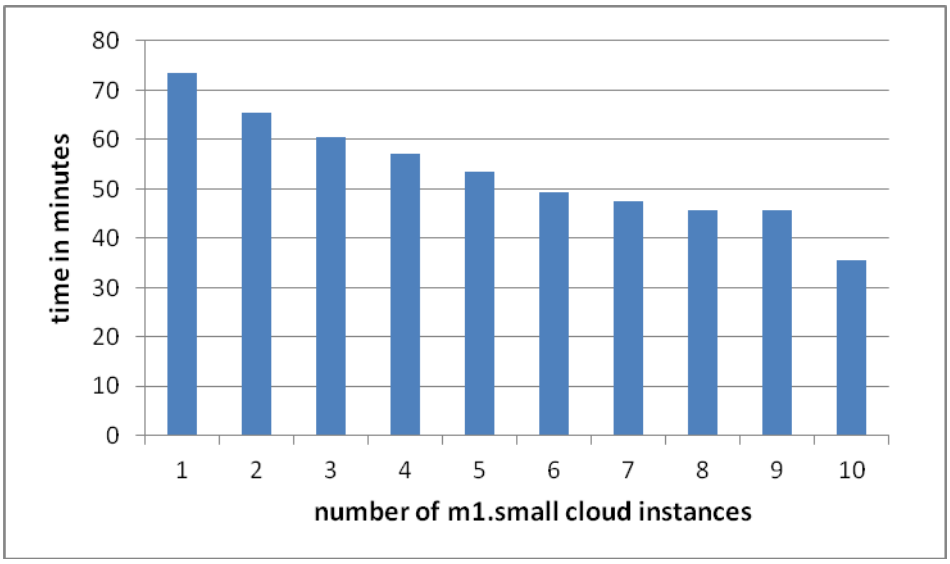

Fig. 11. E-bay workflow's execution time in milliseconds for different number of m1.small cloud instances for 60000 users

The time needed for the workflow to execute on different number of m1.small cloud instances varies from ~73 minutes for one instance to ~36 minutes for 10 m1.small cloud instances; this can be seen in Figure 11.

The fourth round of experiments was run on m1.large cloud instances for a number of 60000 users. The results are presented in Figure 12 as a comparison between the time needed to execute the workflow for 6000/60000 users. As we can observe the execution time needed for the workflow to

run on m1.large cloud instances for 60000 users varies from ~ 71 minutes on a single instance to ~44 minutes on four cloud instances.

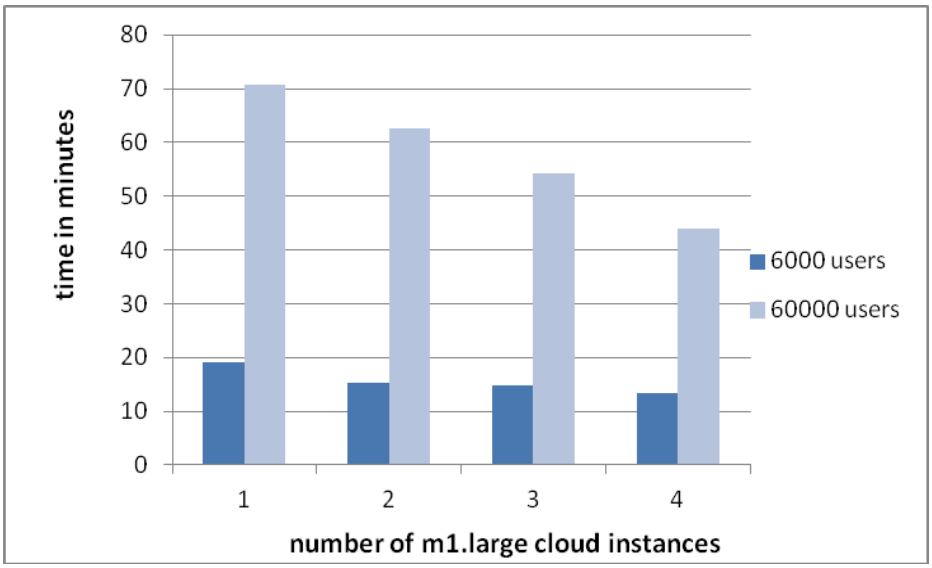

Fig. 12. Comparison between execution times on different number of m1.large cloud instances for 6000 and 60000 users

6 Conclusions

Scientific workflows are now being used on a large scale in order to help scientist run their application and take advantage of the computational resources that they have at their disposal. At the beginning, workflows appeared as a mean for scientists to better identify the steps their applications were made of and to help them scale the applications by identifying the component activities and the correlations existent among them. Nowadays, many workflow management systems exist, some of the most used being mentioned in the introduction section, but the majority of them being employed to run workflows on grid and cluster resources. Only in near past they were modified in such a manner that they could take full advantages of the new emerged kind of computational resources, cloud resources. In this paper we presented a study case of a workflow built based on a Python written E-bay data retrieval application. Our purpose was to show how in the case of this application a better speedup can be achieved by paralyzing certain sections of the application just by implementing a workflow with a parallel section and without needing high parallel programming skills. The test run proved that by using a workflow built using Askalon, we can achieve a speedup of 13 just by adding the parallel section and running the application on the same type of resource (m1.small). Also we proved that by increasing the number of CPUs used for running the workflow the speedup reach ~23 compared with the time needed to execute the application sequentially. The best result is that in the case of running the application for 6000 users the needed time without parallelization was about 268 minutes, and with multiple cloud instances and parallelization the time decreased to about 13 minutes. We also showed that by using the cloud resources at their full potential better speedups can be achieved. This results from the speedup achieved by using same number of m1.small cloud instances but running the workflow for a larger number of users, in this case 60000, and having 400 parallel activities. With these settings the speedup achieves was of ~88 compared with executing the application for the same number of users on a single cloud instance and in a sequential manner. Using cloud instance resources as their full potential is a desired outcome especially when it comes to optimizing execution costs. A coat analysis is

not part of the current work but it will be taken into account for future work.

To conclude, we can say that the use of scientific workflows can definitely improve the performance of scientific application, it can improve their scalability and the number of heterogeneous resources on which they can run can vary according to the needs. The resource type chosen for our tests were cloud resources due to several advantages that this type of resources has to offer: they are easy to procure (a lot of cloud providers are present on the market), the costs are low, and scalability is present due to the pay-as-you-go way of acquiring cloud resources. Also the need of getting certificates from third parties in order to access certain resources has disappeared in the context of cloud computing. Now scientist can provide the needed computational resources at lower prices, they can manage this resources on their own and so their attention can be focused on actually implementing and testing better application that on worrying about getting the needed amount of resources.

Business workflows can take advantage of any performances and results achieved by testing running scientific workflows on cloud resources. The observed behavior of cloud resources under certain conditions can be extrapolated to the usage of cloud resources by the scientific world to the business world.

Acknowledgement

This work is supported by the Romanian Authority for Scientific Research under project IDEI 2452. G. Morar and C. Muntean acknowledge support from the: Investing in people! Ph.D. scholarship, Project co-financed by the Sectoral Operational Programme for Human Resources Development 2007 – 2013, contract nr. POSDRU/88/1.5/S/60185 – "Innovative Doctoral Studies in Knowledge Based Society"

References

[1] M. Singh, M. Vouk. "Scientific workflows: scientific computing meets transactional workflows". In *Proceedings of the NSF Workshop on Workflow and Process Automation: State-of-the-Art and Future Directions*, 1996.

[2] T. Tannenbaum, D. Wright, K. Miller, M. Livny. "Condor - A Distributed Job Scheduler". In *Beowulf Cluster Computing with Linux*, The MIT Press, 2002.

[3] E. Deelman, J. Blythe, Y. Gil, C. Kesselman, G. Mehta, K. Vahi. "Mapping Abstract Complex Workflows onto Grid Environments". In *Journal of Grid Computing*, no. 1, pp. 25-39, 2003.

[4] I. Taylor, M. Shields, I. Wang. "Resource Management of Triana P2P Services". *Grid Resource Management*, Kluwer, Netherlands, June 2003.

[5] S. McGough, L. Young, A. Afzal, S. Newhouse, J. Darlington. "Workflow Enactment in ICENI". In *UK eScience All Hands Meeting*, Nottingham, UK, IOP Publishing, pp. 894-900, 2004.

[6] T. Oinn, M. Addis, J. Ferris, D. Marvin, M. Senger, M. Greenwood, T. Carver, K. Glover, M.R. Pocock, A. Wipat, P. Li. "Taverna: a tool for the composition and enactment of bioinformatics workflows". In *Bioinformatics*, no. 20(17), pp. 3045-3054, Oxford University Press, 2004.

[7] G. von Laszewski, K. Amin, M. Hategan, N. J. Zaluzec, S. Hampton, A. Rossi. "GridAnt: A ClientControllable Grid Workflow System". In *Proceedings of the 37th Annual Hawaii International Conference on System Sciences (HICSS'04)*, IEEE Press, 2004.

[8] J. Cao, S.A. Jarvis, S. Saini, G.R. Nudd. "GridFlow:Workflow Management for Grid Computing". In *Proceedings of the 3rd International Symposium on Cluster Computing and the Grid (CCGrid)*, IEEE Press, 2003.

[9] T. Fahringer, A. Jugravu, S. Pllana, R. Prodan, C.S.Jr, and H.L. Truong. "ASKALON: a tool set for cluster and Grid computing". In *Concurrency and Computation: Practice and Experience*, no. 17, pp. 143-169, Wiley, 2005.

[10] J. Yu, R. Buyya. "A Novel Architecture for Realizing Grid Workflow using Tuple

Spaces". In *Proceeding of the 5th IEEE/ACM International Workshop on Grid Computing (Grid 2004)*, IEEE Press, 2004.

[11] I. Altintas, A. Birnbaum, K. Baldridge, W. Sudholt, M. Miller, C. Amoreira, Y. Potier, B. Ludaescher. "A Framework for the Design and Reuse of Grid Workflows". In *Proceedings of the International Workshop on Scientific Applications on Grid Computing (SAG'04)*, LNCS 3458, Springer, 2005.

[12] J. Yu, R. Buyya, "A Taxonomy of Workflow Management Systems for Grid Computing. Technical Report", *GRIDS-TR-2005-1, Grid Computing and Distributed Systems Laboratory*, University of Melbourne, Australia, March 10, 2005.

[13] T. Fahringer, "ASKALON user Guide – version 1.0", University of Innsbruck, Austria, 2007, Available: http://www.askalon.org/documents/ASKALONUserGuide-final.pdf

[14] L. Wang, J. Tao, M. Kunze, A.C. Castellanos, D. Kramer, W. Karl. "Scientific cloud computing: Early definition and experience". In *Proceedings of the 10th IEEE International Conference on High Performance Computing and Communications*, IEEE Press, pp. 825-830, 2008.

[15] L.M. Vaquero, L. Rodero-Merino, J. Caceres, M. Lindner. "A break in the clouds: towards a cloud definition". ACM *SIGCOMM Computer Communications Review*, no. 39(1), pp 50-55, 2008.

[16] A. Lenk, M. Klems, J. Nimis, S. Tai, T. Sandholm. "What's Inside the Cloud? An Architectural Map of the Cloud Landscape", in *Proceedings of the 2009 ICSE Workshop on Software Engineering Challenges of Cloud Computing (CLOUD'09)*. IEEE Computer Society, pp. 23–31, 2009.

[17] Q. Zhang, L. Cheng, R. Boutaba, "Cloud computing: state-of-the-art and research challenges". In *Journal of Internet Services and Applications*, no. 1, pp. 7–18, 2010.

[18] "Eucalyptus Open-Source Cloud Computing Infrastructure - An Overview", Eucalyptus Systems Inc, 2009, Available: http://www.eucalyptus.com/pdf/whitepapers/Eucalyptus_Overview.pdf

[19] S. Ostermann; R. Prodan, T. Fahringer. "Dynamic Cloud provisioning for scientific Grid workflows". In *Proceedings of the 11th IEEE/ACM International Conference on Grid Computing*, IEEE Press, pp. 97-104, 2010

[20] S. Ostermann, K. Plankensteiner, R. Prodan, T. Fahringer. "GroudSim: An event-based simulation framework for computational grids and clouds". In *Proceedings of the CoreGRID/ERCIM Workshop on Grids, Clouds and P2P Computing*. Springer, August 2010.

[21] S. Pandey, D. Karunamoorthy, R. Buyya, "Workflow Engine for Clouds," In *Cloud Computing, Principles and Paradigms*, Wiley Series on Parallel and Distributed Computing, pp. 321-344, Wiley, 2011.

[22] G. Juve, E. Deelman, "Scientific workflows and clouds," In *Crossroads*, no. 16(3), pp. 14-18, ACM 2010.

Gabriela MORAR has graduated the Faculty of Economics and Business Administration, the Business Information Systems Bachelor from Babes-Bolyai University of Cluj-Napoca in 2008, later specializing in a master in Business Information Systems and Information Society. Her doctoral orientation is towards studying Service Oriented Architectures, especially Cloud Computing, and developing an efficient negotiation model for SLAs (Service Level Agreements).

Cristina MUNTEAN has graduated the Faculty of Economics and Business Administration, Business Information Systems Bachelor from Babes-Bolyai University of Cluj-Napoca in 2008, later specializing in a master in Business Information Systems and Information Society. In her doctoral research she is studying Business Intelligence techniques like web mining and applying them in order to improve website performance. She is interested in data mining, web technologies and Internet marketing.

Gheorghe Cosmin SILAGHI is an Associate Professor at the Babes-Bolyai University of Cluj-Napoca, Romania. He received a Bachelor degree in Business Information Systems in 2000 and his Engineering degree in Computer Science in 2002. In 2002, he received his M.Sc. in Artificial Intelligence from Free University of Amsterdam. G.C. Silaghi completed his Ph.D. in 2005, investigating collaborative multi-agent systems. He joined the Babes-Bolyai University in 2000 and currently holds the position of Head of the Business Information Systems department. His current research interests are focused on resource management techniques in un-trusted distributed environments, like Peer-to-Peer systems and volunteer computing. G.C. Silaghi authored papers in highly reputed journals like Journal of Grid Computing, Journal of Parallel and Distributed Computing and Security and Communication Networks.