Cost Optimization of Execution of Multi-level Deadline-Constrained Scientific Workflows on Clouds

Maciej Malawski[1](✉), Kamil Figiela[1], Marian Bubak[1,2], Ewa Deelman[3], and Jarek Nabrzyski[4]

[1] Department of Computer Science, AGH, al. Mickiewicza 30, 30-059 Kraków, Poland
{malawski,kfigiela,bubak}@agh.edu.pl
[2] ACC CYFRONET AGH, ul. Nawojki 11, 30-950 Kraków, Poland
[3] USC Information Sciences Institute, Admiralty Way, Marina Del Rey, CA 4676, USA
deelman@isi.edu
[4] Center for Research Computing, University of Notre Dame, Notre Dame, IN, USA
naber@nd.edu

Abstract. This paper introduces a cost optimization model for scientific workflows on IaaS clouds such as Amazon EC2 or RackSpace. We assume multiple IaaS clouds with heterogeneous VM instances, with limited number of instances per cloud and hourly billing. Input and output data are stored on a Cloud Object Store such as Amazon S3. Applications are scientific workflows modeled as DAGs as in the Pegasus Workflow Management System. We assume that tasks in the workflows are grouped into levels of identical tasks. Our model is specified in AMPL modeling language and allows us to minimize the cost of workflow execution under deadline constraints. We present results obtained using our model and the benchmark workflows representing real scientific applications such as Montage, Epigenomics, LIGO. We indicate how this model can be used for scenarios that require resource planning for scientific workflows and their ensembles.

Keywords: AMPL optimization · Cloud computing · Scientific workflows

1 Introduction

Nowadays, science requires processing of large amounts of data and use of hosted services for compute-intensive tasks [10]. Cloud services are used not only to provide resources, but also for hosting scientific datasets, as in the case of AWS public datasets [2]. Scientific applications that run on these clouds have often the structure of workflows or workflow ensembles that are groups of inter-related workflows [16]. Infrastructure as a Service (IaaS) cloud providers offer services, where virtual machine instances differ by performance and price [7]. Planning scientific experiments requires optimization decisions that take into account both execution time and cost.

Research presented in this paper can be seen as a step towards developing a cloud resource calculator for scientific applications in the hosted science model [10]. Specifically, we address the cost optimization problem of large-scale scientific workflows running on multiple heterogeneous clouds, using mathematical modeling with AMPL [12] and mixed integer programming. This approach allows to describe the model mathematically and use a set of available optimization solvers. On the other hand, an attempt to apply this method to the general problem of scheduling large-scale workflows on heterogeneous cloud resources would be impractical due to the problem complexity, therefore simplified models need to be developed. In our previous work [15], we used a similar technique to solve the problem where the application consists of tasks that are either identical or vary in size within a small range. As observed in [4,11], large-scale scientific workflows often consist of multiple parallel stages or levels, each of which has a structure of bag of tasks, i.e. the tasks in each level are similar. In the case of large workflows, when the number of tasks in the level is high, it becomes more practical to optimize the execution of the whole level instead of looking at each task individually, as many scheduling algorithms do [17]. Therefore, in this paper, we extend our model to deal with applications that are workflows represented as DAGs consisting of levels or layers of uniform tasks.

After outlining the related work in Sect. 2, we introduce the application and infrastructure model in Sect. 3. In Sect. 4 we provide the problem formulation in AMPL. Section 5 describes the evaluation of our model on a set of benchmark workflows, while Sect. 6 gives conclusions and future work.

2 Related Work

Our work is related to heuristic algorithms for workflow scheduling on IaaS clouds, such as the ones described in [1,3,5,17]. Our infrastructure model differs in that we assume multiple heterogeneous clouds with object storage attached to them, instead of individual machines with peer-to-peer data transfers between them. Instead of scheduling each task individually, our approach proposes a global optimization of placement of workflow tasks and data.

The deadline-constrained cost optimization of scientific workloads on heterogeneous IaaS described in [6] addresses multiple providers and data transfers between them, where the application is a bag of tasks. The global cost minimization problem on clouds, addressed in [18] focuses on data transfer costs and does not address workflows. Other approaches presented in [8,14] consider unpredictable dynamic workloads on IaaS clouds and optimize the objectives such as cost, runtime or utility function by autoscaling the resource pool at runtime.

Pipelined workflows consisting of stages are addressed in [19], where the processing model is a data flow and multiple instances of the same workflow are executed on the same set of cloud resources. Our work is different in that our goal is cost optimization instead of meeting the QoS constraints.

3 Application and Infrastructure Model

We assume that a workflow is divided into several levels (layers) that can be executed sequentially and tasks within one level do not depend on each other (see Fig. 1). Each layer represents a bag of tasks that can be partitioned in several groups (e.g. application A, B, etc.) that share computational cost and input/output size. We assume that only one task group is executed on a specific cloud instance (VM). This forbids instance sharing between multiple layers, which means that each application needs its own specific VM template.

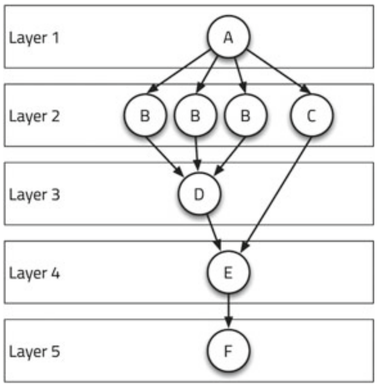


Fig. 1. Example application structure

Similarly as in [15], we assume multiple heterogeneous cloud IaaS infrastructures such as Amazon EC2, RackSpace or ElasticHosts. Clouds have heterogeneous VM instance types, with limits on the number of instances per cloud, e.g. 20 for EC2, 15 for RackSpace, etc. Input and output data are stored on Cloud Object Store such as Amazon S3 or RackSpace CloudFiles. In our model, all VM instances are billed per hour of usage, and there are fees for data transfers. In the model we can also have a private cloud where costs are set to 0.

4 Problem Formulation Using AMPL

To perform optimization of the total cost, Mixed Integer Problem (MIP) is formulated and implemented in A Mathematical Programming Language (AMPL) [12]. AMPL requires us to specify input data sets and variables to define the search space, as well as constraints and objective function to be optimized.

Input data. The formulation requires the following input sets, which represent the infrastructure model, in a similar way as we approached the problem in [15]:

- $S = \{s3, cloudfiles\}$ – defines available cloud storage sites,
- $P = \{amazon, rackspace, \ldots\}$ – defines possible computing cloud providers,
- $I = \{m1.small, \ldots, gg.1gb, \ldots\}$ – defines instance types,
- $PI_p \subset I$ – instances that belong to provider P_p,
- $LS_s \subset P$ – compute cloud providers that are local to storage platform S_s.

Each instance type I_i is described by the following parameters:

- p_i^I – fee in \$ for running instance I_i for one hour,
- ccu_i – performance of instance in CloudHarmony Compute Units (CCU) [9],

– p_i^{Iout} and p_i^{Iin} – price for non-local data transfer to and from the instance, in \$ per MiB (1 MiB = 1024 ∗ 1024 B).

Storage sites are characterized by:

– p_s^{Sout} and p_s^{Sin} characterize price in \$ per MiB for non local data transfer.

Additionally we need to provide data transfer rates in MiB per second between storage and instances by defining function $r_{i,s} > 0$.

Our application model is different from the one in [15] because it groups tasks into layers:

– L – set of layers,
– G – set of tasks groups,
– G_l – set of tasks groups belonging to layer l,
– A_t^{tot} – number of tasks in group t,
– t_t^x – execution time in hours of a single task of group t on 1 CCU machine,
– d_t^{in} and d_t^{out} – data size for input and output of one task t in MiB,
– p^R – price per request for queuing service, such as Amazon SQS, required to execute a single task,
– t^D – total time for completing workflow (deadline).

Auxiliary parameters. A set of precomputed parameters, which are derived from the main input parameters of the model includes:

– $t_{i,s}^{net} = \frac{d^{in}+d^{out}}{r_{i,s} \cdot 3600}$ – *transfer time*: time for data transfer between I_i and S_s,
– $t_{i,s}^u = \frac{t^x}{ccu_i} + t_{i,s}^{net}$ – *unit time*: time for processing a task on instance I_i using storage S_s that includes computing and data transfer time (in hours),
– $c_{i,s}^T = (d^{out} \cdot (p_i^{Iout} + p_s^{Sin}) + d^{in} \cdot (p_s^{Sout} + p_i^{Iin}))$ – cost of data transfer between instance I_i and storage S_s,
– I_i^{idx} – set of possible instance I_i indexes (from 0 to $n_i^{Imax} - 1$).

Variables. Variables that will be optimized and define the solution space are:

– $A_{t,i,x}$ – binary, 1 iff (if and only if) instance I_i with index x is launched to process task group G_t, otherwise 0;
– $H_{t,i,x}$ – int, for how many hours is instance launched;
– $T_{t,i,x}$ – int, how many tasks of G_t are processed on that instance,
– D_l^t – actual computation time for L_l,
– D_l – int, maximal number of hours that instances are allowed to run in L_l.

Objectives. Cost of running one task including instance and transfer cost is:

$$\left(t^{net} + t^u\right) \cdot p^I + d^{in} \cdot \left(p^{Sout} + p^{Iin}\right) + d^{out} \cdot \left(p^{Iout} + p^{Sin}\right) + p^R, \tag{1}$$

while the objective function represents the total cost of running multiple tasks of the application on the cloud infrastructure is defined as:

$$\underset{\text{total cost}}{\text{minimize}} \sum_{t \in G, i \in I, x \in I_i^{idx}} ((p_i^I * H_{t,i,x} + p^R + c_{i,s}^T) * T_{t,i,x}), \tag{2}$$

subject to the constraints:

1. $\sum_{l \in L} D_l \leq t^D$ ensures that workflow finishes in the given deadline,
2. to fix that $D = \lceil D^t \rceil$ we require that: $\forall_{l \in L} D_l^t \leq D_l \leq D_l^t + 1$,
3. $\forall_{t \in G, i \in I, x \in I_i^{idx}} A_{t,i,x} \leq H_{t,i,x} \leq A_{t,i,x} \cdot t^D$ ensures that H may be allocated only iff A is 1,
4. $\forall_{t \in G, i \in I, x \in I_i^{idx}} A_{t,i,x} \leq T_{t,i,x} A_{t,i,x} \cdot A_t^{tot}$ ensures that T may be allocated only iff A is 1,
5. $\forall_{t \in G, i \in I, x \in I_i^{idx}} H_{t,i,x} \leq D_l$ enforces layer deadline on instances runtime,
6. $\forall_{l \in L, t \in G_l, i \in I, x \in I_i^{idx}} T_{t,i,x} \cdot t_{t,i,s}^u \leq D_l^t$ enforces that a layer finishes work in D^t,
7. to make sure that all the instances run for enough time to process all tasks allocated to them we require: $\forall_{t \in G, i \in I, x \in I_i^{idx}} T_{t,i,x} \cdot t_{t,i,s}^u \leq H_{t,i,x} T_{t,i,x} \cdot t_{t,i,s}^u + 1$, which adjusts H respectively to T,
8. $\forall_{t \in G} \sum_{i \in I, x \in I_i^{idx}} T_{t,i,x} = A_t^{tot}$ ensures that all tasks are processed,
9. To reject symmetric solutions, we add three constraints:
   (a) $\forall_{t \in G, i \in I, x \in \{1..(n_i^{Imax} - 1)\}} H_{t,i,x} \leq H_{t,i,x-1}$,
   (b) $\forall_{t \in G, i \in I, x \in \{1..(n_i^{Imax} - 1)\}} A_{t,i,x} \leq A_{t,i,x-1}$, and:
   (c) $\forall_{t \in G, i \in I, x \in \{1..(n_i^{Imax} - 1)\}} T_{t,i,x} \leq T_{t,i,x-1}$.
10. $\forall_{l \in L, p \in P} \sum_{i \in PI_p, t \in G_l, x \in I_i^{idx}} A_{t,i,x} \leq n_p^{Pmax}$ enforces instance limits per cloud.

To keep this model in MIP class we had to take a different approach than in previous model, and schedule each virtual machine instance separately. The drawback of this approach is that we need to increase the number of decision variables. We also divided the search space by storage provider. Additionally, the deadline becomes a variable with upper bound as it may happen that shorter deadline may actually give a cheaper solution (see Fig. 3 and its discussion).

5 Evaluation

To evaluate our model on realistic data, we use CloudHarmony [9] benchmarks to parameterize the infrastructure model, and we use the Workflow Generator Gallery workflows [4] as test applications. In the infrastructure model we assumed that we have 4 public cloud providers (Amazon EC2, RackSpace, GoGrid and ElasticHosts) and a private cloud with 0 cost. The infrastructure has two storage services, S3 that is local to EC2 and CloudFiles that is local to RackSpace, so data transfers between local compute and storage are free. We tested our model with all applications from the gallery: Montage, CyberShake, Epigenomics, LIGO and SIPHT for all available workflow sizes (from 50 to 1000 tasks per workflows, up to 5000 tasks in the case of SIPHT workflow). We varied the deadline from 1 to 30 h with 1-h increment. We solve the problem for two cases, depending on whether the data is stored on S3 or on CloudFiles.

Figure 2 shows the example results obtained for the Epigenomics application and workflows of two sizes (400 and 500 tasks). For longer deadlines the private cloud instances and the cheapest RackSpace instances are used so the cost is low when using CloudFiles. For shorter deadlines the cost grows rapidly, since we reach the limit of 15 instances per cloud and additional instances must be

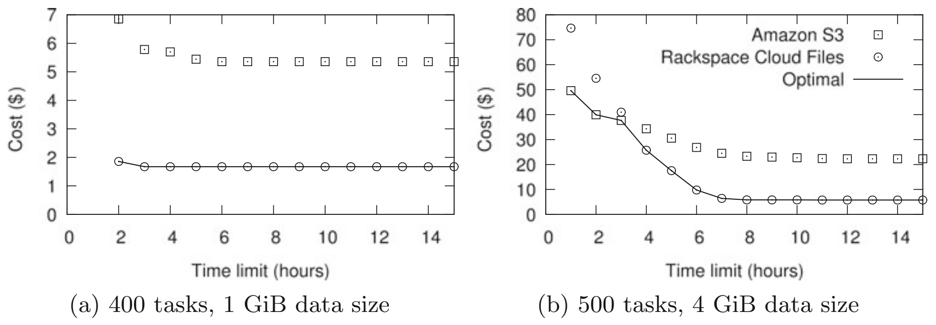


Fig. 2. Result of the optimization procedure for the Epigenomics application.

spawned on a different provider, making the transfer costs higher. This effect is amplified in Fig. 2b, which differs from Fig. 2a not only by the number of tasks but also by the data size of one layer. This means that the transfer costs are growing more rapidly, so it becomes more economical to store the data on Amazon EC2 that provides more powerful instances required for short deadlines.

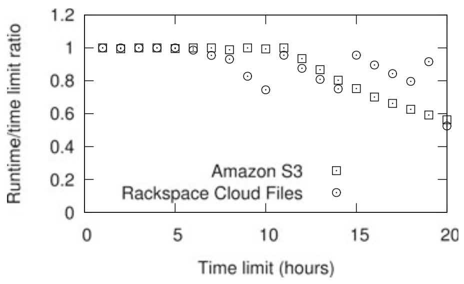
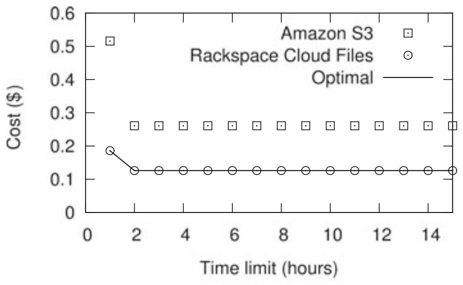


Fig. 3. Ratio of actual completion time to deadline for Epigenomics workflow with 500 tasks.

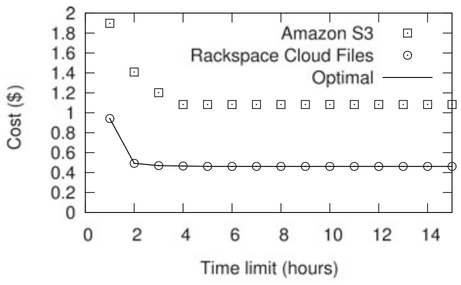
One interesting feature of our model is that for longer deadlines it can find the cost-optimal solutions that have a shorter workflow completion time than the requested deadline. This effect can be observed in Fig. 3 and is caused by the fact that for long deadlines the simple solution is to run the application on a set of the least expensive machines.

Figure 4a–d show results obtained for other workflows. These workflows are relatively small and even for short deadlines the model is able to schedule tasks on cheapest instances on a single cloud, thus resulting in flat characteristics.

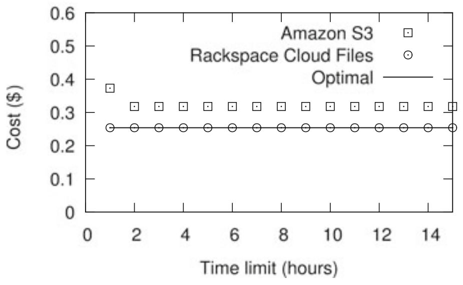
To investigate how the model behaves for workflows with the same structure, but with much longer run times of tasks, we run the optimization for Montage workflow with tasks 1000× longer. This corresponds to the scenario where tasks are in the order of hours instead of seconds. The sample results in Fig. 5a show



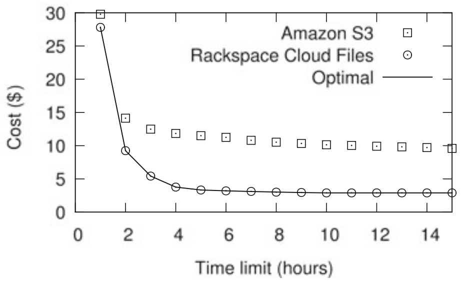
(a) CyberShake, 500 tasks



(b) LIGO, 500 tasks

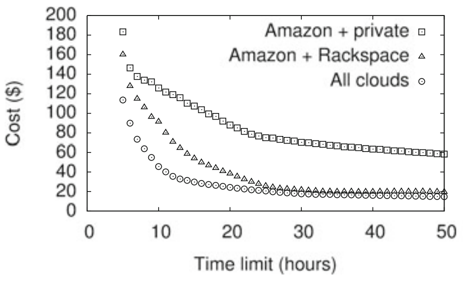


(c) Montage, 500 tasks

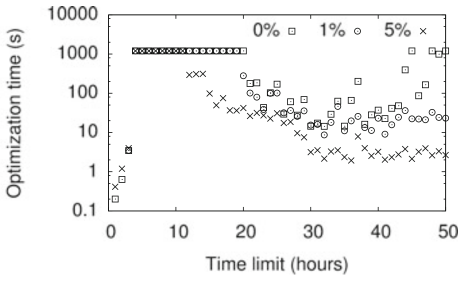


(d) SIPHT, 5000 tasks

Fig. 4. Optimal cost found by the model for different applications.



(a) Optimal cost

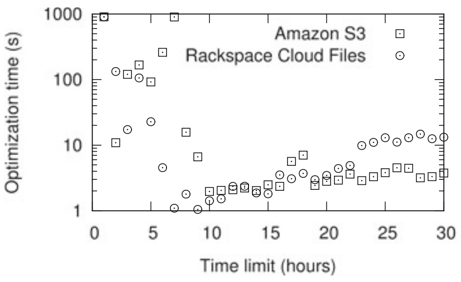


(b) Solver runtime with different MIP gap

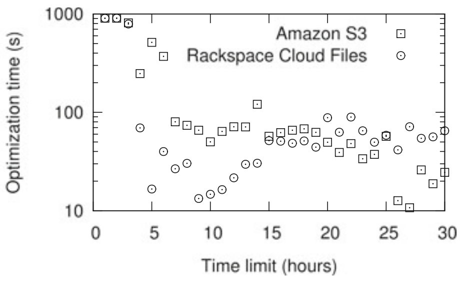
Fig. 5. Results obtained by the model for Montage 500 workflow with tasks runtimes artificially multiplied by 1000 for different cloud infrastructures.

how the cost increases much steeply with shorter deadlines, illustrating the trade-off between time and cost. The difference between Figs. 4c and 5a illustrates that the model is more useful for workflows when tasks are of granularity that is similar to the granularity of the (hourly) billing cycle of cloud providers. Additionally, Fig. 5a shows how the optimal cost depends on the cloud available.

The run time of the optimization algorithm for workflows with up to 1000 tasks ranges from few seconds up to 4 min using the CPLEX [13] solver running on a server with 4 16-core 2.3 GHz AMD Opteron processors (model 6276), with



(a) Epigenomics, 600 tasks



(b) SIPHT, 5000 tasks

Fig. 6. Solver execution wall time.

a limit set to 32 cores. Figure 6a shows that the time becomes much higher for shorter deadlines and increases for very long deadlines. This is correlated with size of search space: the longer the deadline, the search space is larger, while for shorter deadlines the problem has a very small set of acceptable solutions. The problem becomes more severe for bigger and more complex workflows like SIPHT as optimization time becomes very high (Fig. 6b).

Figure 5b illustrates how the optimization time depends on MIP gap solver setting [13]. Applying a relative MIP gap of 1 % or 5 % instead of default 0.01 % shortens optimization time in orders of magnitude. Increasing the MIP gap to 5 % did not decrease the quality of the result noticeably: the minimum cost obtained for the gap of 5 % was higher only by 3.63 % in the worst case.

6 Conclusions and Future Work

In this paper, we presented a cost optimization model for scientific workflows executing on multiple heterogeneous clouds. The model, formulated in AMPL, allows us to find the optimal assignment of workflow tasks, grouped into layers, to cloud instances. We tested our model on a set of benchmark workflows and we observed that it gives useful solutions in a reasonable amount of computing time. By solving the model for multiple deadlines, we can produce trade-off plots, showing how the cost depends on the deadline. We believe that such plots are a step towards a scientific cloud workflow calculator, supporting resource management decisions for both end-users and workflow-as-a-service providers.

In future work we plan to apply this model to the problem of provisioning cloud resources for workflow ensembles [16], where the optimization of cost can drive the workflow admission decisions. We also plan to refine the model to better support smaller workflows by reusing instances between layers, to fine-tune the model, and to test different solver configurations to reduce the computing time.

Acknowledgement. This research was partially supported by the EC ICT VPH-Share Project (contract 269978), the KI AGH grant, and by the National Science Foundation under grant OCI-1148515.

References

1. Abrishami, S., Naghibzadeh, M., Epema, D.H.: Deadline-constrained workflow scheduling algorithms for infrastructure as a service clouds. Future Gener. Comput. Syst. **29**(1), 158–169 (2013). http://www.sciencedirect.com/science/article/pii/S0167739X12001008
2. AWS: AWS public datasets. http://aws.amazon.com/publicdatasets/ (2013)
3. Barrionuevo, J.J.D., Fard, H.M., Prodan, R.: Moheft: a multi-objective list-based method for workflow scheduling. In: 4th IEEE International Conference on Cloud Computing Technology and Science Proceedings, CloudCom 2012, Taipei, Taiwan, 3–6 December 2012, pp. 185–192 (2012)
4. Bharathi, S., Chervenak, A., Deelman, E., Mehta, G., Su, M.H., Vahi, K.: Characterization of scientific workflows. In: Third Workshop on Workflows in Support of Large-Scale Science, WORKS 2008, pp. 1–10. IEEE (2008). http://dx.doi.org/10.1109/WORKS.2008.4723958
5. Bittencourt, L.F., Madeira, E.R.M.: Hcoc: a cost optimization algorithm for workflow scheduling in hybrid clouds. J. Internet Serv. Appl. **2**(3), 207–227 (2011)
6. den Bossche, R.V., Vanmechelen, K., Broeckhove, J.: Online cost-efficient scheduling of deadline-constrained workloads on hybrid clouds. Future Gener. Comput. Syst. **29**(4), 973–985 (2013). http://www.sciencedirect.com/science/article/pii/S0167739X12002324
7. Bubak, M., Kasztelnik, M., Malawski, M., Meizner, J., Nowakowski, P., Varma, S.: Evaluation of cloud providers for VPH applications. In: CCGrid2013 - 13th IEEE/ACM International Symposium on Cluster, Cloud and Grid, Computing, May 2013. http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=6546092
8. Chen, J., Wang, C., Zhou, B.B., Sun, L., Lee, Y.C., Zomaya, A.Y.: Tradeoffs between profit and customer satisfaction for service provisioning in the cloud. In: Proceedings of the 20th International Symposium on High Performance Distributed Computing, HPDC '11, pp. 229–238. ACM, New York (2011)
9. CloudHarmony: Benchmarks. http://cloudharmony.com/benchmarks (2011)
10. Deelman, E., Juve, G., Malawski, M., Nabrzyski, J.: Hosted science: managing computational workflows in the cloud. Parallel Process. Lett. **23**(2), June 2013. http://www.worldscientific.com/doi/abs/10.1142/S0129626413400045
11. Duan, R., Prodan, R., Li, X.: A sequential cooperative game theoretic approach to storage-aware scheduling of multiple large-scale workflow applications in grids. In: 2012 ACM/IEEE 13th International Conference on Grid Computing (GRID), pp. 31–39. IEEE (2012). http://dx.doi.org/10.1109/Grid.2012.14
12. Fourer, R., Gay, D.M., Kernighan, B.W.: AMPL: A Modeling Language for Mathematical Programming. Duxbury Press, Belmont (2002)
13. IBM: IBM ILOG CPLEX Optimization Studio - CPLEX User's Manual. http://pic.dhe.ibm.com/infocenter/cosinfoc/v12r5/index.jsp (2013)
14. Kim, H., El-Khamra, Y., Rodero, I., Jha, S., Parashar, M.: Autonomic management of application workflows on hybrid computing infrastructure. Sci. Program. **19**, 75–89 (2011)
15. Malawski, M., Figiela, K., Nabrzyski, J.: Cost minimization for computational applications on hybrid cloud infrastructures. Future Gener. Comput. Syst. **29**(7), 1786–1794 (2013). http://www.sciencedirect.com/science/article/pii/S0167739X13000186
16. Malawski, M., Juve, G., Deelman, E., Nabrzyski, J.: Cost- and deadline-constrained provisioning for scientific workflow ensembles in IaaS clouds. In: Proceedings of the

International Conference on High Performance Computing, Networking, Storage and Analysis, SC '12. IEEE Computer Society Press (2012). http://portal.acm.org/citation.cfm?id=2389026
17. Mao, M., Humphrey, M.: Auto-scaling to minimize cost and meet application deadlines in cloud workflows. In: Proceedings of 2011 International Conference for High Performance Computing, Networking, Storage and Analysis, SC '11. ACM, New York (2011). http://dx.doi.org/10.1145/2063384.2063449
18. Pandey, S., Barker, A., Gupta, K.K., Buyya, R.: Minimizing execution costs when using globally distributed cloud services. In: 24th IEEE International Conference on Advanced Information Networking and Applications, pp. 222–229. IEEE (2010)
19. Tolosana-Calasanz, R., Banares, J.A., Pham, C., Rana, O.F.: Enforcing QoS in scientific workflow systems enacted over cloud infrastructures. J. Comput. Syst. Sci. **78**(5), 1300–1315 (2012). http://www.sciencedirect.com/science/article/pii/S0022000011001607