Cluster Comput (2017) 20:2805–2819
DOI 10.1007/s10586-017-1018-x

Modeling and predicting execution time of scientific workflows in the Grid using radial basis function neural network

Farrukh Nadeem[1] · **Daniyal Alghazzawi**[1] · **Abdulfattah Mashat**[2] · **Khalid Fakeeh**[1] · **Abdullah Almalaise**[1] · **Hani Hagras**[3]

Received: 1 November 2016 / Revised: 14 April 2017 / Accepted: 24 June 2017 / Published online: 12 July 2017
© Springer Science+Business Media, LLC 2017

Abstract With the maturity of electronic science (e-science) the scientific applications are growing to be more complex composed of a set of coordinating tasks with complex dependencies among them referred to as workflows. For optimized execution of workflows in the Grid, the high level middleware services (like task scheduler, resource broker, performance steering service etc.) need in-advance estimates of workflow execution times. However, modeling and predicting workflow execution time in the Grid is complex due to several tasks in a workflow, their distributed execution on multiple heterogeneous Grid-sites, and dynamic behaviour of the shared Grid resources. In this paper, we describe a novel method based on radial basis function neural network to model and predict workflow execution time in the Grid. We model workflows execution time in terms of attributes describing workflow structure and execution runtime information. To further refine our models, we employ principle component analysis to eliminate attributes of lesser importance. We recommend a set of only 14 attributes (as compared with total 21) to effectively model workflow execution time. Our reduced set of attributes improves the prediction accuracy by 16%. Results of our prediction experiments for three real-world scientific workflows are presented to show that our predictions are more accurate than the two best methods from related work so far.

Keywords Scientific workflow applications · Distributed execution of scientific workflows · Workflow execution time prediction in the Grid

✉ Farrukh Nadeem
fabdullatif@kau.edu.sa

Abdulfattah Mashat
asmashat@uj.edu.sa

Hani Hagras
hani@essex.ac.uk

[1] Department of Information Systems, Faculty of Computing and Information Technology, King Abdulaziz University, Jeddah, Saudi Arabia

[2] University of Jeddah, Jeddah, Saudi Arabia

[3] School of Computer Science and Electronic Engineering, The Computational Intelligence Centre, University of Essex, Colchester, UK

1 Introduction

Computational Grids enable application developers to aggregate heterogeneous computational and storage resources scattered around the globe for large-scale scientific and engineering research. Scientific workflow applications (later referred as scientific workflows or just workflows) have recently emerged as an important paradigm for representing and managing complex scientific computations. Typically a workflow consists of a set of tasks (software executions or data transfers), which are coordinated by control and data flow dependencies to solve a complex problem. Workflow applications are usually executed on the Grid through workflow management systems like Askalon [14], GridFlow [4], Pegasus [10] etc. for their automatic execution. The runtime environment of such workflow management systems schedule and manage the execution of workflow tasks on multiple Grid-sites with the objective to minimize the workflow execution time. Workflow execution time is widely considered as a metric to measure workflow performance [14].

Workflow schedulers, enactment engines and performance analysis services are commonly part of the runtime environments that rely on execution time modeling of scientific applications in order to take crucial strategic decisions and to determine the causes for performance problems.

The scheduler needs execution time estimates for individual workflow tasks as well as for the entire workflow in order to determine an effective mapping between tasks and Grid-sites. The enactment engine commonly controls the actual execution of the workflow which requires workflow execution time models for runtime steering, automatic scaling and load balancing of workflow tasks. Performance analysis services need performance characterization and execution time predictions for diagnosing performance bottlenecks at run time and during post-mortem analysis. Grid runtime environments thus greatly benefit by reasonably accurate execution time models of scientific workflow applications. Above all, the application user requires estimates about application execution time to plan execution of his application. Furthermore, the resource owners/administrators need execution time estimates to manage usage of their resources.

It is widely known that the execution time of a workflow application is very difficult to estimate due to inherent architectural and functional heterogeneity of the Grid. Moreover, the workflow structure and complex dependencies between workflow tasks including input data dependencies, external load and non-deterministic behaviour of the Grid make execution time prediction for Grid workflows a notoriously difficult task. To this end, we employ radial basis function neural network (RBF-NN) to model and predict workflow execution time in the Grid. For RBF-NN modeling, we parameterized workflows in terms of workflow static (workflow tasks, dependencies among tasks, input problem-size, etc.) as well as dynamic attributes (describing workflow runtime environment). We then reduce the dimensionality of workflow attributes through principal component analysis. We observe that the predictions through the proposed methodology are more accurate than the previous methods by Nadeem et al. based on templates [42] and local learning framework [43] (see Sect. 5 and Fig. 7 for details). The three major contributions are as follows: first, prediction accuracy of the proposed method is the highest as compared with methods in the related work; second, we rank workflow attributes based on their importance for modeling workflow execution time; last but not the least, we recommend a reduced set of workflow attributes required to effectively model workflow execution time by eliminating attributes of lesser importance. Adopting our recommended set of attributes only not will save a lot of time, effort and system resources for collecting the trace data but also improve prediction accuracy. In addition, this will also facilitate the future efforts for modeling workflow execution time.

The rest of the paper is organized as follows. Section 2 explains distributed execution of workflows in the Grid. Section 3 describes the main method of RBF-NN. The dimensionality reduction of workflow attributes through principal component analysis is addressed in Sect. 4. Section 5 highlights our experiments for evaluating the proposed approach. A review of the related work is presented in Sect. 6. Finally, we conclude in Sect. 7.

2 Understanding distributed execution of scientific workflows in the Grid

An application workflow (or simply workflow) is a well-defined pattern or systematic organization of tasks designed to achieve a specific goal. In general, a scientific workflow application can be represented as a directed acyclic graph (DAG), where nodes represent the individual tasks and edges correspond to inter-task dependencies. Today's scientific workflows consist of tens/hundreds of tasks [18]. To execute workflow tasks efficiently and to harness the maximum potential of the Grid, the workflows are usually executed through a high level middleware (like Pegasus [10], Askalon [14], Kepler [37], Triana [23], Taverna [56], etc.) in a distributed manner on multiple Grid-sites.

Depending upon the nature of workflow tasks, the tasks may be executed sequentially or in parallel on one or multiple Grid-sites. In case of exploiting more than one Grid-site, there will be data transfers before execution of some tasks. On shared resources, the tasks may be subject to wait due to other tasks executing and occupying all the resources. Besides, the execution of workflow tasks may be delayed due to local scheduling policies defined by the resource owners, who may define higher priorities for some users and lesser for others. After start of an task execution, the execution time of the task depends upon the resource computational power and the external load due to other applications running at the same time. The overall execution of the workflow depends upon the available Grid-sites, their computational powers, Grid-sites' states at the time of workflow execution and way the the workflow tasks are mapped to the Grid-sites. The execution time of the workflow may vary widely if different sets of Grid-sites are selected. In addition, the execution time may also vary with problem-size of the workflow.

2.1 Attributes affecting execution of scientific workflows in the Grid

To model execution time of scientific workflows in the Grid effectively, it is important to understand different attributes that define the workflow performance. Broadly, these attributes affecting workflow execution time can be divided into two classes: static attributes and dynamic attributes [42] . The static attributes mainly consist of workflow structure attributes including workflow tasks, their dependencies, and problem-size. The dynamic attributes cover the execution environment, where the workflow is executed. These include Grid-sites, scheduling policy, and Grid-sites' states. The detailed attributes are described as under.

- Workflow structure
  - *Workflow name* represents name of the workflow/ application.
  - *Workflow tasks* represent components of the workflow (executables or data transfers).
  - *Control flow dependencies* describe the order of execution of workflow tasks.
  - *Dataflow dependencies* indicate that output data of one task will be used as input for another.
- Application
  - *Problem-size* represents the input problem-size of a workflow execution. It is one of the important attributes that define workflow performance and was explicitly stored in our trace data.
  - *Executables* names of softwares corresponding to workflow tasks.
  - *Versions* represent versions of executables (e.g. MPI version/serial version etc.).
  - *File-sizes* represent the size of input files.
- Execution environment
  - *Grid-sites* names of the Grid-sites selected for workflow execution.
  - *Time of submission* time (hour of the day) of submission of workflow for execution.
  - *Scheduling strategy* name of scheduling algorithm used to map workflow tasks to selected Grid-sites.
- Resource state
  - *Jobs in queue* number of jobs already in the queue at time of job submission.
  - *CPUs in the queue* number of CPUs required for the jobs already in the queue.
  - *Jobs already running* number of jobs running at the time of task submission.
  - *Already occupied CPUs* number of CPUs occupied by the jobs running at the time of task submission.
  - *Jobs running in parallel* the number of jobs running concurrently with the submitted task.
  - *CPUs occupied in parallel* the number of CPUs occupied by the jobs running concurrently with the submitted task.
- Resource sharing policy
  - *User name* name of the user submitting the workflow. The resource owners define different resource sharing policies for different users.
  - *Group/VO* name of the group or (VO). The users may experience different resource sharing policies under the umbrella of groups created for different projects.

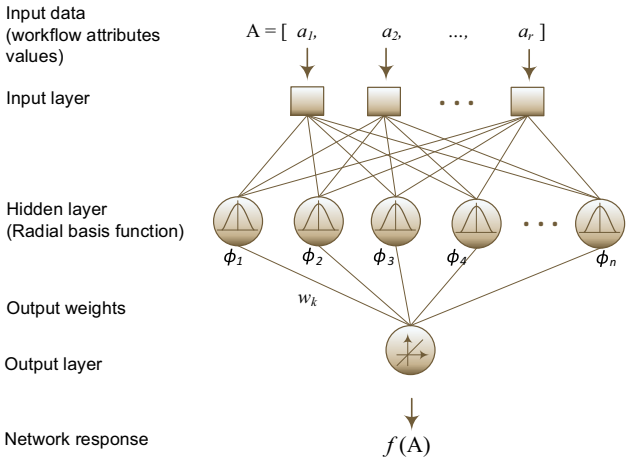


Fig. 1 Architecture of $r - n - 1$ RBF-NN

- Network
  - *Bandwidth* available network bandwidth for data transfers during workflow execution.
  - *Latency* network latency at the time of the data transfers during workflow execution.
  - *Number of Parallel transfers* used to transfer data. Usually a higher number of parallel transfers data more efficiently.

3 Modeling workflow execution time using radial basis function neural network

Radial basis function neural network is an artificial neural network that uses radial basis function as network activation function. It is a popular tool for interpolating multivariate scattered data due to its fast training and global approximation capabilities. Using RBF-NN for workflow execution time modeling, we aimed at finding an interpolation function $f(A)$ that can map sets of workflow attributes values $A_1, A_2, A_3, ..., A_m$ to corresponding the workflow execution times $t_1, t_2, t_3, ..., t_m$, i.e. $f(A_1) \approx t_1$. The architecture of our RBF-NN is shown in Fig. 1. It is a typical three-layered neural network. The first layer is the input layer, which takes sets of workflow attributes values $A_i = \{a_1^i, a_2^i, a_3^i, ..., a_r^i\}$ as input to the network. Here a_1^i represents value of attribute a_1 in ith set (data row). The second layer (hidden layer) is the layer of neurons, where each neuron i computes the Euclidean distance d_i between an input set A_i and the center ($d_i = \|A_i - c\|$), and applies the Gaussian function as a radial function:

$$\phi_i(d_i) = e^{-\beta d_i^2} \tag{1}$$

The centers were determined through Weka toolkit [20] using *k-mean clustering* algorithm on the training dataset where

the cluster centers were taken as centers. The value of β was determined as:

$$\beta = \frac{1}{2\sigma^2} \tag{2}$$

where σ is the width parameter of Gaussian function and was set to average distance of the cluster center from all the points in the cluster. i.e.

$$\sigma = \frac{1}{s}\sum_{i=1}^{s} \|A_i - c\| \tag{3}$$

Here s is the number of training samples that belong to a cluster, A_i is the ith training sample in the cluster.

The output of second layer ($\phi(d_i)$) is communicated through weights w_i to the third layer, which approximates $f(A)$ as a linear combination of n neurons such that

$$f(A) = \sum_{i=1}^{n} w_i \phi_i(d_i) \tag{4}$$

The weights w_i were determined by solving the following linear system.

$$\begin{bmatrix} \phi_1(d_1) & \phi_2(d_1) & \phi_3(d_1) & \dots & \phi_n(d_1) \\ \phi_1(d_2) & \phi_2(d_2) & \phi_3(d_2) & \dots & \phi_n(d_2) \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \phi_1(d_m) & \phi_2(d_m) & \phi_3(d_m) & \dots & \phi_n(d_m) \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_m \end{bmatrix} = \begin{bmatrix} t_1 \\ t_2 \\ \vdots \\ t_m \end{bmatrix} \tag{5}$$

The main motivation behind using RBF-NN is its capacity to handle several workflow attributes and develop execution time models with high accuracy even when the workflow attributes have complex distributions in m-dimensional space. One disadvantage of RBF-NN is that it does not differentiate between relative importance of workflow attributes and thus tries to optimize the weights of all of them. This affects the accuracy of the model. To compensate this affect, we employed principal component analysis (see Sect. 4) to eliminate the attributes of lesser importance and consider the more important attributes only in our model. This leads to improved accuracy of the model.

4 Dimensionality reduction through principal component analysis

Principal component analysis (PCA) [26] is a popular mathematical procedure to reduce dimensionality of a large dimension space into a small dimension space without loss of much information. PCA extracts the important information from a given dataset of correlated values and uses orthogonal transformations to convert them to a dataset of non-correlated values referred to as *principal components*. The principal components can be reduced as per need.

The goal of our PCA is to reduce the dimensions of 21-dimensional trace dataset $A_i = \{a_1^i, a_2^i, a_3^i, ..., a_{21}^i\}$ (see Sects. 2.1 and 5.1 for details) to a q-dimensional dataset where $q < 21$ provided that we do not loose much information from original dataset.

In the first step, we standardized the trace dataset of each attribute a by subtracting the mean (μ) and dividing by the standard deviation (σ).

$$a_{std} = \frac{a_{orig} - \mu}{\sigma} \tag{6}$$

Here a_{orig} and a_{std} represent original and standardized values of attribute a. The standardization brought the centroid of the whole dataset to origin (zero).

Then, the covariance matrix $C_{21\times 21}$ of our 21-dimensional standardized data was computed as:

$$C = \begin{bmatrix} cov(a_1,a_1) & cov(a_1,a_2) & cov(a_1,a_3) & \dots & cov(a_1,a_{21}) \\ cov(a_2,a_1) & cov(a_2,a_2) & cov(a_2,a_3) & \dots & cov(a_2,a_{21}) \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ cov(a_{21},a_1) & cov(a_{21},a_2) & cov(a_{21},a_3) & \dots & cov(a_{21},a_{21}) \end{bmatrix} \tag{7}$$

where

$$cov(X,Y) = \frac{\sum_{i=1}^{m}(X_i - \bar{X})(Y_i - \bar{Y})}{n-1} \tag{8}$$

In the next step, we found the eigenvalues by solving the determinant $|C - \lambda I| = 0$, where λ represents eigenvalues and I represents identity matrix of order 21×21. For our 21-dimensional space of workflow attributes, there were 21 eigenvalues. The eigenvalues represent the amount of variance corresponding to each principal component. We sorted the eigenvalues in decreasing order. For each eigenvalue, the corresponding eigenvector was obtained by solving $[C - \lambda I] * [X] = 0$, where λ is replaced by the eigenvalue and X represents its eigenvector. Thus, for our 21 attributes, there were 21 eigenvalues and 21 eigenvectors.

In the next step, we computed principal components by linear combinations [26] of eigenvectors with the standardized trace dataset obtained through Eq. 6. The first principal component has the highest possible variance and thus can be best used to describe the data along its axis. Each subsequent principal component has the next highest possible variance.

We chose the final set of principal components iteratively through a series of execution time modeling (using RBF-NN) experiments, where the number of principal components was set to 21, 20, 19, 18, ... After the first experiment, in each subsequent experiment, the principal component with the

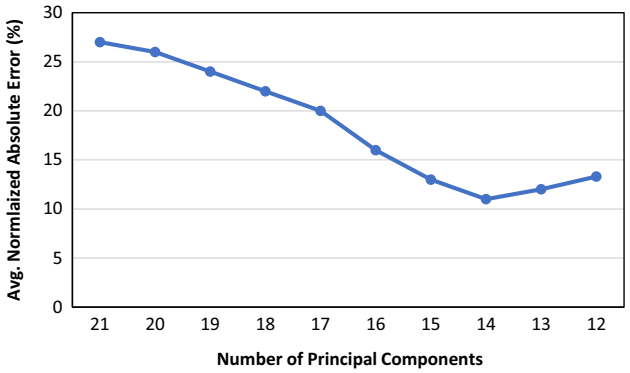


Fig. 2 Reduction in average normalized absolute error with reduction in principal components for MeteoAG workflow

lowest variance was eliminated (to see its effect on prediction accuracy). It means that the principal components having the lowest variance were eliminated one by one. The main motivation behind this approach was to exploit the maximum potential of RBF-NN to affectively model the principal components. The experiments were continued till the accuracy of RBF-NN (with the reduced number of principal components) improved and were stopped as soon as the accuracy started decreasing. A decrease in accuracy of RBF-NN from reduced number of principal components means that some valuable information was lost due to the last eliminated principal component. The smallest set of principal components for which the accuracy of RBF-NN improved was selected as final set. In our dimensionality reduction experiments, we were able to reduce number of principal components from 21 to 14 and still reduce the prediction error by 16% for MeteoAG workflow—average normalized absolute error for MeteoAG workflow before(after) PCA was (27%)(11%) (see Sects. 5.2 and 5.4 for more details). Figure 2 shows the reduction in prediction error with reduction in principal components. Similar results were achieved for other workflows.

In our all later experiments (see Sect. 5), we used the final set of principal components.

5 Experiments

This section describes the scientific workflows considered in our study, the testbed, and details and the outcome of our experiments for evaluating the proposed method.

5.1 Scientific workflows

We evaluated the proposed approach for three real-world applications: MeteoAG [49], Wien2K [44], and Invmod [8]. These three workflows differ largely with respect to workflow structures, number of sequential and parallel activities, average execution times etc. The MeteoAG workflow produces meteorological simulations for atmospheric fields of heavy precipitation. These simulations are used to forecast alpine watersheds and thunderstorms over the western part of Austria. The MeteoAG workflow showing its tasks is depicted in Fig. 3a. Invmod application simulates *water flow and balance model* to help study the effects of climatic changes on the water balance. It also provides improved discharge estimates for extreme floods. The Invmod workflow is shown in Fig. 3b. Wien2k is a program package for performing electronic structure calculations of solids using density functional theory based on the full-potential augmented plane-wave and local orbital methods. The Wien2K workflow of is shown in Fig. 3c.

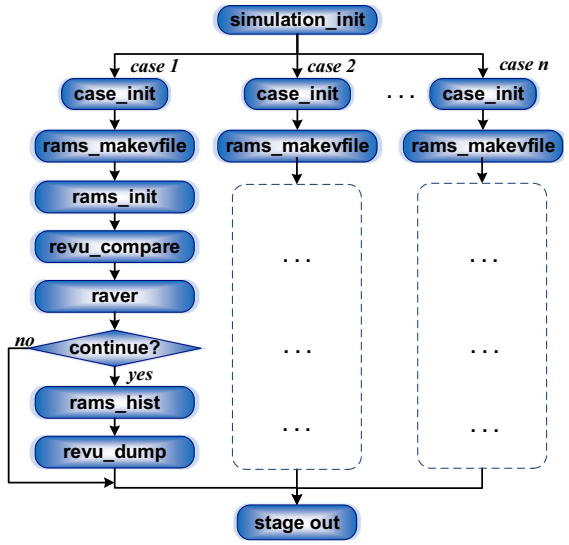
These workflows were executed using workflow development and runtime environment Askalon [14] in Austrian Grid environment. See the following section for more details.

5.2 Testbed

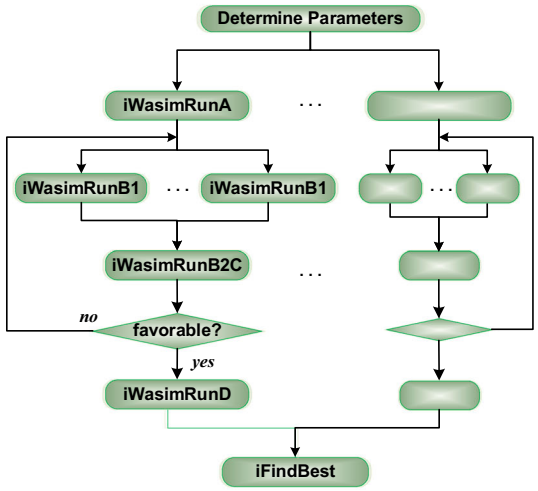
Austrian Grid is National Grid infrastructure of Austria that consists of several Grid-sites dispersed geographically in different cities of Austria. Most of the Grid-sites are heterogeneous, managed by independent authorities and implement different resource sharing policies for different users. A summary of the the Grid-sites used in our experiments and number of workflow tasks executed on them are show in Table 1. We executed the workflows mentioned in Sect. 5.1 in Austrian Grid at different times (hours/days) using several different problem sizes of the workflows on a range of Grid-sites (1–10). Moreover, the LRM queues at Grid-sites also showed different states. Overall, the trace data of the workflows reflected a lot of different execution scenarios. Table 2 shows some statistics of the trace data obtained from these executions.

5.3 Data pre-processing

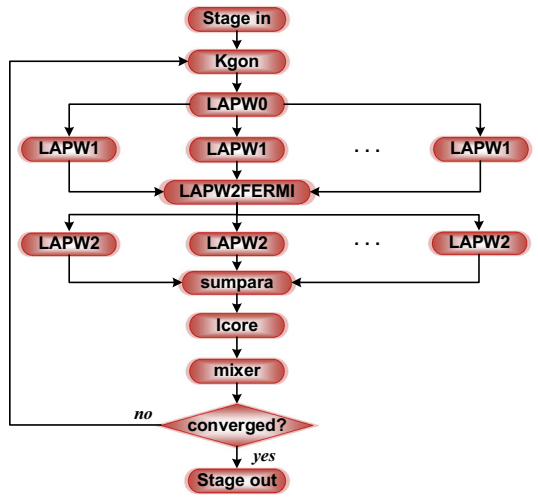
Before proceeding towards RBF-NN modeling, we need to prepare our data for it. First of all, the available dataset was cleaned to remove all incomplete or failed executions. In next step, the outlier data was removed using *distance-based subspace outlier detection* [1]. Next, all of the nominal values (like application name, Grid-sites, etc.) were represented by numeric values using *one-to-N coding* [21] scheme, with the exception of problem-size which was represented by *thermometer coding* [21]. The workflow structure was considered as a whole and each distinct structure of a workflow was encoded uniquely using *one-to-N coding*. Any workflow structure with different tasks or control/data flow dependencies is considered a distinct workflow structure. Reasons behind considering the workflow as a whole are multifold. First, coding individual components of the workflow is very



(a) MeteoAG Workflow.



(b) Invmod Workflow.



(c) Wien2K Workflow.

Fig. 3 Workflows of scientific applications considered in the study

complex and infeasible for a dataset of a several workflows. Second, in case of big workflows (with large number of activities), the number of input attributes will increase largely and make the model learning slow and with reduced accuracy (because we cannot reduce the individual components of the workflow in PCA because it does not make sense. Eventually, the learning phase of the model will "try to" optimize all input attributes. This will result in reduced optimization of other important attributes). Last but not the least, dimension reduction phase will also be very complex through common methods.

At this step, it is important to normalize data to bring all the attribute values to the same scale. We normalized data using the following formula that brings all the variables values in the range of [0,1].

$$a_{norm} = \frac{a - a_{\min}}{a_{\max} - a_{\min}} \tag{9}$$

where a, a_{norm}, $a_{\min}$, $a_{\max}$ represent original, normalized, the minimum and the maximum values respectively.

5.4 RBF-NN training and evaluation

In our early experiments, we modeled RBF-NN with all workflow attributes. One input unit (in the input layer of RBF-NN) was allocated in RBF-NN for each Grid-site. During the learning phase, if a Grid-site was selected in an experiment, its code was given as an input, otherwise 0 was given as an input. Three input units were allocated to represent each resource state; first, to represent state of the queue at the time of submission; second, to represent jobs running at the time of submission; third, to represent jobs running in parallel at the time of execution. Each distinct resource state was encoded uniquely using *one-to-N* coding. One input unit was allocated for each other attribute as well as the workflow structure. The trace dataset was divided into two parts: training data (80%) and test data (20%). The training data was used to develop execution time models and the test data was used to evaluate the execution time models.

We designed our RBF-NN evaluation experiments to look for the answers of following specific questions.

- Which workflow attributes are more important than others in modeling workflow execution time in the Grid?
- How effectively the proposed approach can model and predict the workflow execution time?
- Does the variations in number of Grid-sites and problem-size have any impact on prediction accuracy?
- How effective is the proposed approach as compared with previous approaches [41,42]?

Table 1 Summary of Grid-sites used from Austrian Grid and workflow tasks executions on them

Grid-site[a]	Location	Architecture	CPUs	OS	LRM	Tasks
GS-1	Innsbruck	Ethernet Pentium 4	20	Linux	PBS	8963
GS-2	Innsbruck	AMD Opteron 244	6	Linux	Torque/Maui	1452
GS-3	Linz	SGI Altix 3000 Itanium 2	64	Linux	PBS	48,780
GS-4	Innsbruck	SGI Altix 350 Itanium 2	16	Linux	Torque	85,188
GS-5	Innsbruck	Pentium 4	20	Linux	PBS	24,093
GS-6	Linz	AMD Athelon(tm)	16	Linux	Torque	13,434
GS-7	Innsbruck	Intel(R) Xeon(TM)	12	Linux	Torque/Maui	3596
GS-8	Innsbruck	AMD Opteron 848	80	Linux	SGE	26,294
GS-9	Salzburg	SGI Altix 350 Itanium 2	16	Linux	Fork	17,021
GS-10	Innsbruck	AMD Opteron 248	198	Linux	SGE	12,988

[a] The Grid-sites names have been anonymized

Table 2 Statistics of trace data for Wien2K, Invmod and MeteoAG workflows

Workflow name	When			Execution count	Task count	Exe. time (min)		
	From	To	Duration			Min	Max	Mean
MeteoAG	Jan 2007	Dec 2007	12 months	10,974	142,671	2.31	145.45	24.74
Wien2K	Jan 2006	June 2007	18 months	10,330	165,279	1.61	137.37	12.71
Invmod	July 2006	June 2007	18 months	7184	64,657	1.74	74.93	18.36

The details of our experiments to find the answers of these questions are described in subsequent paragraphs.

We ranked the workflow attributes on the basis of their variance in the data (computed in Sect. 4). The first 14 attributes and their ranks are shown in Table 3—since we were able to reduce dimensions from 21 to 14. The rank 1 in the table indicates the highest rank. The higher the rank of the attribute, the more is the variance in data due to that attribute. It means the higher the rank of the attribute, the more important it is for modeling workflow execution time. Thus the Table 3 shows the attributes in order of their importance for modeling workflow attributes. We found that *workflow tasks* and *user name* are the most and the least important workflow attributes for modeling workflow execution times, respectively. Likewise, we observed that the Grid-sites are the second most important workflow attribute.

In our experiments, we measured the workflow execution time in seconds. The effectiveness of RBF-NN models is evaluated using two metrics. First, the *average absolute prediction error* of n predictions defined as $\sum_{i=1}^{n} |t_i^p - t_i^r|/n$. Here, t_i^p and t_i^r represent i_{th} predicted and real values. To compare prediction accuracies for workflows executed with different sets of attributes (see Sect. 2.1), we also normalized average absolute prediction error by average real time. Second, in terms of the *average absolute relative error* defined as $\sigma_{rel} = (t_p - t_r)/(t_p + t_r)$. It is noteworthy that our selected metrics for prediction accuracy concur with the same by Nadeem et al. in [41]. This allows us to compare our approach with theirs. During our evaluation, we repeated each experiment (for a fixed problem-size and number of Grid-sites) 10 times (for different values of other attributes) and their average values are shown in this paper.

Table 3 Ordered workflow attributes after PCA

Rank	Attributes
1	Workflow tasks
2	Grid-sites
3	Problem-size
4	Scheduling strategy
5	Jobs already in queue
6	Jobs already running
7	Parallel running jobs
8	Para. occupied CPUs
9	CPUs req. in queue
10	Occupied CPUs
11	No. of parallel transfers
12	Bandwidth
13	Executables/versions
14	User-name

We evaluated the proposed approach for its prediction accuracy for workflows executed with different problem-sizes and number of Grid-sites. These two are among the main static attributes that we can vary by choice. The scheduling strategy is varied in subsequent experiments depicted in

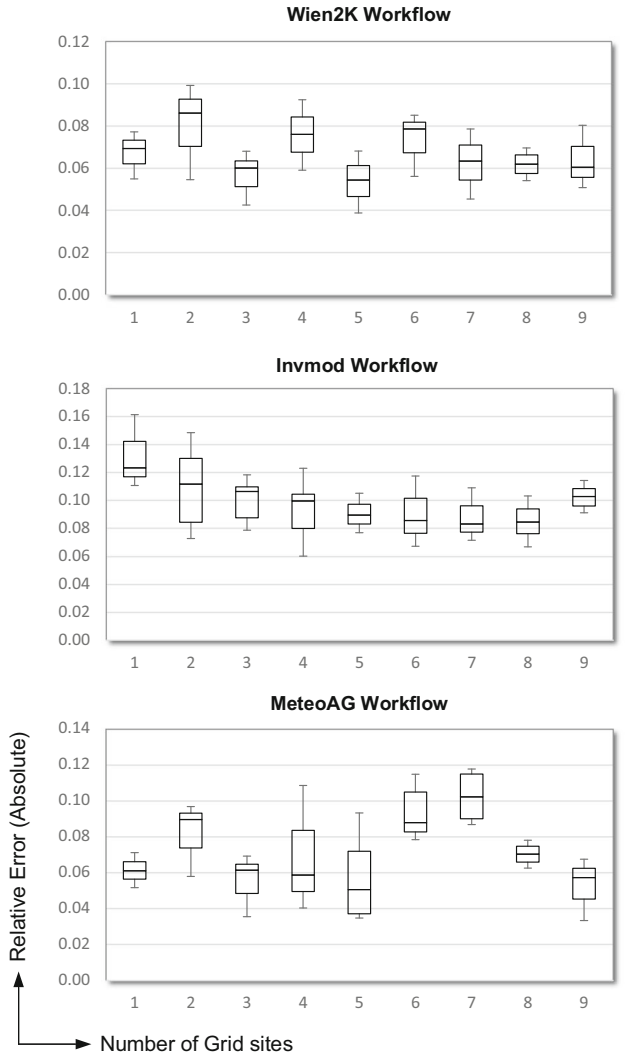


Fig. 4 Absolute relative error for MeteoAG (*bottom*), Invmod (*middle*) and Wien2k (*top*) workflows for different number of Grid-sites where problem-size was varied from problem-size = 1 to problem-size = 3

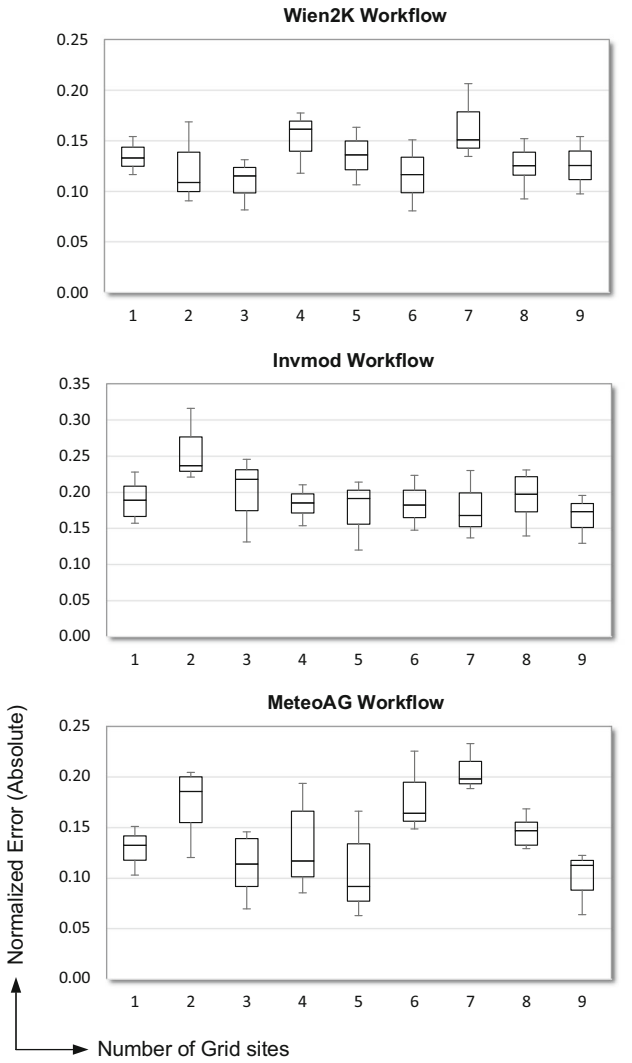


Fig. 5 Normalized absolute error for MeteoAG (*bottom*), Invmod (*middle*) and Wien2k (*top*) workflows for different number of Grid-sites where problem-size was varied from problem-size = 1 to problem-size = 3

Fig. 8 shown later. We do not vary the executables in current study. The rest of the attributes are determined from the runtime environment and thus can not be varied by choice.

Figure 4 shows the average absolute relative error for MeteoAG, Invmod and Wien2K workflows when executed on different number of Grid-sites for three different problem-sizes (problem-size = 1, 2, 3). It should be noted that the execution time of a workflow for a given problem-size may increase or decrease when executed on higher number of Grid-sites – it depends upon the selected Grid-sites and mapping of workflow tasks on them. The absolute relative error for MeteoAG workflow ranged between 3 and 11% with an overall mean of 10%. We observed the minimum(maximum) relative error for Grid-sites=9 and problem-size = 3 (Grid-sites = 6 and problem-size = 3). The absolute relative error for Invmod workflow ranged between 7 and 16% with an overall mean of 12%. The minimum(maximum) relative error was observed for Grid-sites = 3 and problem-size = 2 (Grid-sites = 2 and problem-size = 3). The absolute relative error for Wien2K workflow ranged between 6 and 15% with an overall mean of 11%. The minimum(maximum) relative error was observed for Grid-sites = 5 and problem-size = 1 (Grid-sites = 2 and problem-size = 1). We did not notice any patterns of relative error with changes in problem-size and number of Grid-sites. Overall, we found that the relative error in predictions through the proposed method decreased by 36%(9%), 22%(8%) and 52%(10%) for MeteoAG, Invmod and Wien2K workflows respectively, as compared with the methods based on templates [42] (local learning framework [41]). For a fair comparison, the same parameters were used in methods based on templates and local learning framework.

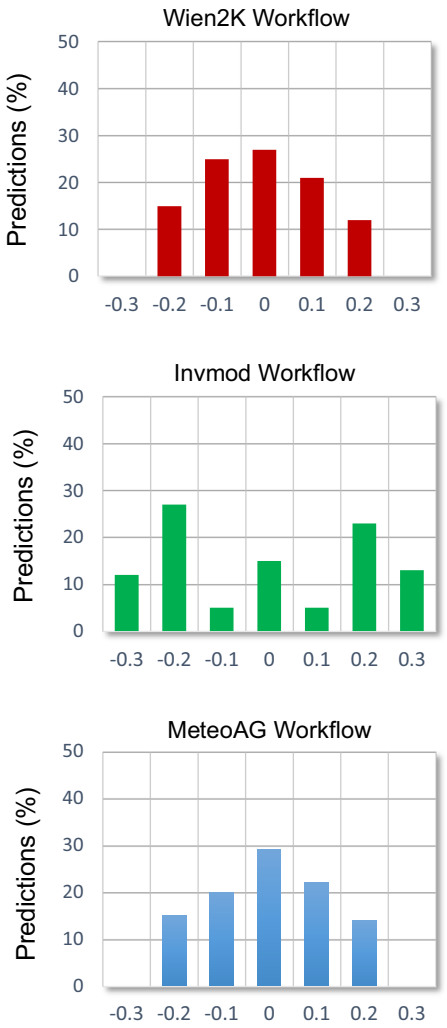


Fig. 6 Distribution of normalized absolute error for MeteoAG (*left*), Invmod (*middle*) and Wien2k (*right*) workflows

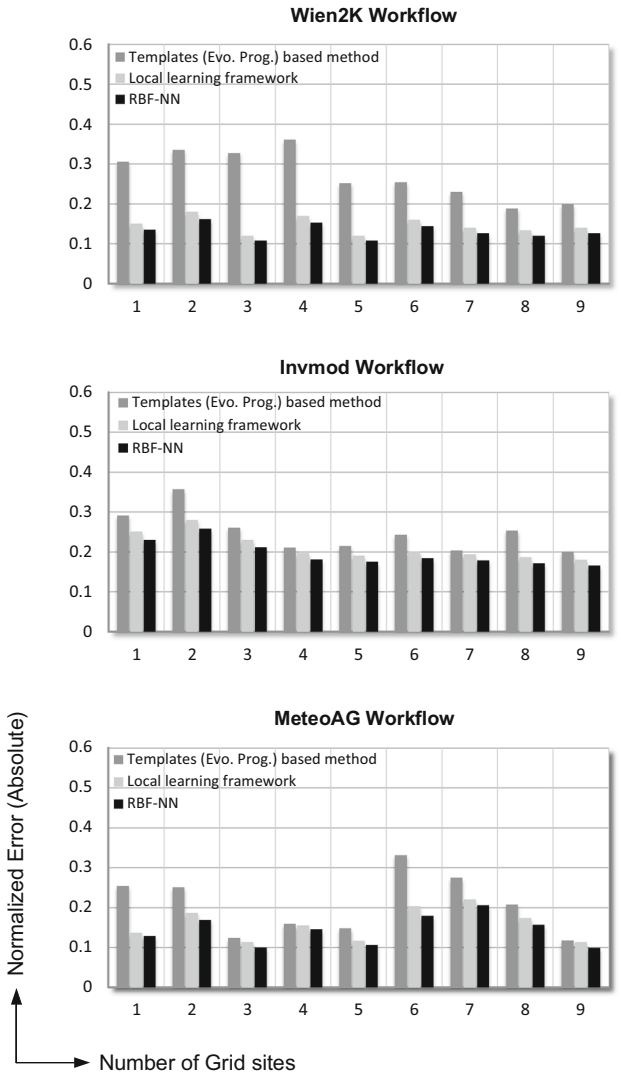


Fig. 7 A comparison of prediction accuracies [in terms of normalized absolute error (average)] of the proposed approach with two methods based on templates and local learning framework for MeteoAG (*bottom*), Invmod (*middle*) and Wien2k (*top*) workflows for different number of Grid-sites

Figure 5 shows average normalized absolute error for MeteoAG, Invmod and Wien2K workflows when executed on different number of Grid-sites for three different problem-sizes (problem-size = 1, 2, 3). The overall mean normalized error for MeteoAG workflow was 11% with the minimum(maximum) of 6%(24%). The minimum(maximum) normalized error was observed for Grid-sites = 9 and problem-size = 3 (Grid-sites = 4 and problem-size = 3). The mean normalized error for Invmod workflow was 15% with the minimum(maximum) of 12%(32%). The minimum(maximum) normalized error was observed for Grid-sites = 5 and problem-size = 1 (Grid-sites = 2 and problem-size = 2). The mean normalized error for Wien2K workflow was 10% with the minimum(maximum) of 8%(21%). The minimum(maximum) normalized error was observed for Grid-sites = 6 and problem-size = 2 (Grid-sites = 7 and problem-size = 1). We did not observe any patterns in normalized error with variations in problem-size and number of Grid-sites. Figure 6 shows the distribution of normalized error for the three workflows. These distributions did not follow any well known model.

Figure 7 shows a comparison of the proposed approach with two other approaches based on templates [42] and local learning framework [41]. We observed that the predictions through the proposed approach are the most accurate (have the least error). The average normalized error through the proposed approach are already described above. The average normalized error through templates based method was 21, 25, and 27% for MeteoAG, Invmod and Wien2K workflows respectively. The average normalized error through local learning framework was 16, 21, and 15% for MeteoAG, Invmod and Wien2K workflows respectively. We found that the normalized error in predictions through the proposed

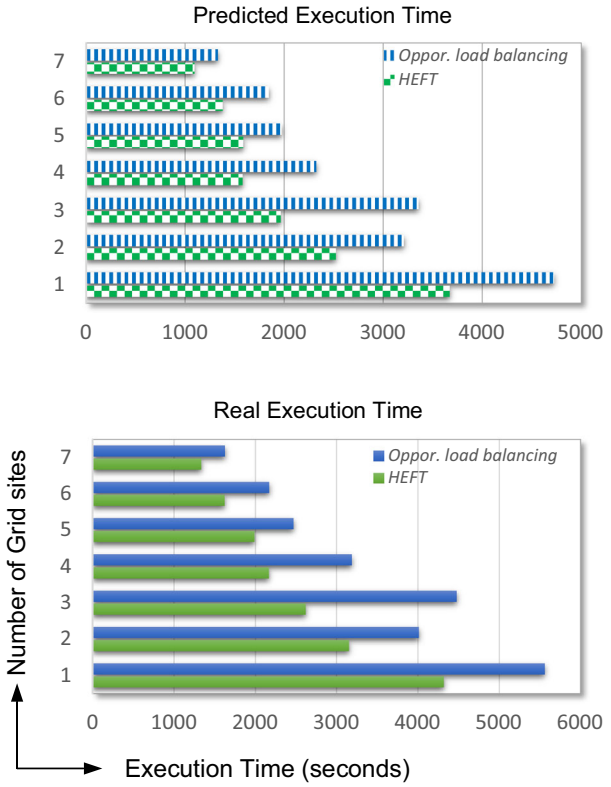


Fig. 8 A comparison of execution times of Invmod workflow using two scheduling algorithms

method decreased by 47%(30%), 39%(29%) and 63%(31%) for MeteoAG, Invmod and Wien2K workflows respectively, as compared with the methods based on templates [42] (local learning framework [41]).

In our further evaluations, we investigated that how effectively our predictions can guide the selection of different optimization strategies employed for Grid high level services. In this regard, we compared real and predicted execution times of Invmod workflow using two scheduling algorithms: *opportunistic load balancing* and *HEFT* [14]. Figure 8 shows comparisons of real and predicted execution time of Invmod workflow using two scheduling algorithms. It can be observed that ranking of the two scheduling algorithms is the same in real as well as in predicted execution times. Thus, despite of the prediction accuracies, the proposed method can still effectively guide for the selection of the optimization strategies.

For our dataset, using Weka toolkit on MacBook Pro, Intel core i7, 2.3 GHz quad core with 16 GB memory it took less than 30 s to build RBF-NN for each workflow and a few milliseconds for each prediction.

6 Related work

There have been several attempts in the past to predict the execution time of single task applications employing a variety of methods (e.g. combining application models with machine profiles [5,46,53,57,59,63], analytical models [3,9,67], statistical modeling [2,63,64], hybrid methods combing analytical models with statistical methods [6], through historical data [17,27,31,39], time series [25,35], data mining methods [31,52] particularly neural networks [24,29], application benchmarks [15,48,58], partial program executions [66], simulation [55], and using skeletons [54]). Performance modeling and prediction for parallel applications through user provided performance functions has been addressed in [44]. Most of these efforts target predictions for individual Grid-sites, and do not account for performance variations due to changes in problem-sizes of the application. In our previous attempts, we have used application benchmarks [15] to predict the execution time of single tasks for different problem-sizes and machine-sizes. This approach achieved reasonable prediction accuracy for very regular (e.g. only uniform memory access behaviour) and simple codes. Moreover, benchmarking must be done for all problem-sizes and number of Grid-sites for which predictions are required. This can dramatically increase the training phase effort. Moreover these benchmarks ignore external load on the Grid-sites. In contrast to these efforts, we introduce a novel prediction method based on RBF-NN for estimating the execution time of larger, more complex workflow applications on heterogeneous Grid-sites considering the external load. This new method can also derive predictions for the problem-sizes and number of Grid-sites for which no historic information is given.

There also have been some attempts to predict the performance of individual Grid resources such as CPU load [25,28,65], network bandwidth [13,61], and queue wait times [30]. The prediction of critical paths of a workflow has been examined in [34,50]. Prediction for job failures and its impact on performance has been addressed in [32]. There have been several efforts to use machine learning to predict tasks execution times [31], the effects of compiler performance optimization [11], scheduling [47] and the performance behaviour of networks [13].

Workflow profiling was used by Da Silva et al. [51] to predict run time and resource consumption of individual workflow tasks. Liu et al. [36] proposed a time series pattern based method to predict the execution time of workflow tasks. Execution time of workflow tasks was also predicted by Miu et al. [38] considering task input features and historical data.

Execution time of workflow applications was modeled through analytical models in [28] and [33]. Eder et al. [12] employed probabilistic models to estimate execution times of scientific workflows. These approaches are limited to execution time modeling of workflow applications on single Grid-site assuming homogeneous CPU architectures. Moreover, the analytical models used assume fixed problem-sizes.

In contrast, we target at execution time prediction of the workflow applications which are executed on geographically dispersed heterogenous Grid-sites. Furthermore, our method takes into account the application problem-sizes and external load on the Grid-sites.

An effort closer to our focus has been introduced by Glatard et al. in [19]. The authors used probabilistic models to analyze workflow execution time in the Grid by considering execution times of individual tasks, and data transfers between the tasks and model the remainder of the execution time (the overheads in different execution phases) as a random variable. Gelenbe et al. [16] and Mussi et al. [40] also considered the execution time of a task graph as a random variable and determined its distribution from the graph parameters. These approaches assume simplified application workflows, ignoring complex control flows between tasks, and loops over different (sets of) tasks—which are driven by the problem-size. Moreover, variations in execution time because of input data are not taken into account. Contrarily, the proposed approach considers complex workflow structure attributes as well as the problem-size used for the workflow execution. Furthermore, our work also considers the impact of different optimizations employed to execute the workflows, such as different scheduling algorithms for mapping workflow tasks to Grid-sites.

The authors in [7] combine the task execution times and workflow structure to model execution time of the workflow as random variable and provide its distribution function. Their approach assumes the availability of accurate runtime models of the individual tasks on different Grid-sites. In addition, the authors assume a predefined mapping of workflow tasks on the Grid-sites and does not take into account any runtime optimization, e.g. in dynamic scheduling. Contrarily, our approach does not depend on any runtime models of individual tasks. We model the execution time of the whole workflow (including individual tasks) dynamically and predict it as a definite quantity (in contradiction to probabilistic models). Moreover, our approach does not assume a predefined mapping of workflow tasks on the Grid-sites, rather it considers scheduling strategy as run time optimization.

Nadeem et al. employed similarity templates [42,43] and local learning framework [41] to predict execution of workflow applications. The authors considered the most of the major attributes describing workflow execution at different Grid infrastructural levels (like Grids-site, network, etc.), particularly the attributes to reflect workflow structure similarities defined by Wombacher et al. [62]. The attributes considered in their study were large in number, which resulted in high inaccuracies (sometimes even more than 50%). In the proposed method, we employed principal component analysis [60] to investigate the impacts of individual workflow attributes and to reduce the number of attributes to be considered in execution time modeling. We employed RBF-NN to model and predict workflow execution time from the reduced set of attributes. The prediction accuracy of the proposed method is higher than that of the methods by Nadeem et al.

There have been some recent efforts of predicting workflow execution time in the Cloud environments. Hiden et al. [22] proposed a performance data capture and modeling architecture to generate execution time models in the Cloud. The models are generated from the partial execution of applications and are dynamically updated as additional performance data is available. Pietri et al. [45] predicted the execution time of scientific workflows in the Cloud. The authors considered workflow structures and number of resources in their model and achieved reasonable accuracy in their predictions.

7 Conclusion and future work

Grid workflow applications consist of several tasks that have data and control flow dependencies among them. High level Grid services require workflow execution time predictions to best exploit the available resources. However, predicting workflow execution time is challenging due to complex workflow structures, distributed execution of workflows on multiple Grid-sites and dynamic nature of the Grid. In this paper, we parameterize each workflow execution in terms of the attributes describing its static as well as runtime information and use a novel method based on RBF-NN to model and predict overall execution time of the workflow. To further optimize the accuracy of RBF-NN models, we evaluated the relative importance of workflow attributes to model workflow execution time. Using principal component analysis, we recommend a reduced set of 14 attributes (out of total 21 attributes considered initially) that highly affect the overall execution time of a workflow. From our reduced set of workflow attributes, we are able to improve the prediction accuracy by 16% (as compared with prediction accuracy using all 21 attributes). This reduction in workflow attributes will also be helpful in improving accuracy of future efforts of workflow performance modeling. To the best of our knowledge, our approach yields the lowest prediction error as compared with similar approaches in related work. The normalized error in our predictions is 46% and 30% less than that from the two other methods in related work (using templates and local learning framework) that yielded the highest prediction accuracy so far. In future, we plan to validate the proposed approach with more workflows and by varying the size of training data. We also plan to consider more sophisticated machine learning methods to model and predict workflow execution time.

Acknowledgements This project was funded by the National Plan for Science, Technology and Innovation (MAARIFAH)—King Abdulaziz City for Science and Technology - the Kingdom of Saudi Arabia—Award Number (12-INF2716-03). The authors also, acknowledge with thanks Science and Technology Unit, King Abdulaziz University for technical support.

References

1. Aggarwal, C.C.: Outlier Analysis. Springer, New York (2013)
2. Barnes, B.J., Rountree, B., Lowenthal, D.K., Reeves, J., de Supinski, B., Schulz, M.: A regression-based approach to scalability prediction. In: Proceedings of the 22nd Annual International Conference on Supercomputing, ICS '08, pp. 368–377. ACM (2008)
3. Brehm, J., Worley, P.H.: Performance prediction for complex parallel applications. In: Proceedings of the 11th International Parallel Processing Symposium, 1997, pp. 187–191. IEEE (1997)
4. Cao, J., Jarvis, S.A., Saini, S., Nudd, G.R.: Gridflow: Workflow management for grid computing. In: Proceedings of the 3rd International Symposium on Cluster Computing and the Grid, CCGRID '03, pp. 198–206. IEEE Computer Society (2003)
5. Cao, J., Jarvis, S.A., Spooner, D.P., Turner, J.D., Kerbyson, D.J., Nudd, G.R.: Performance prediction technology for agent-based resource management in grid environments. In: Proceedings of the 16th International Parallel and Distributed Processing Symposium, IPDPS '02, pp. 265–274. IEEE Computer Society (2002)
6. Carrington, L., Snavely, A., Wolter, N.: A performance prediction framework for scientific applications. Future Gener. Comput. Syst. **22**(3), 336–346 (2006)
7. Chirkin, A.M., Belloum, A.S.Z., Kovalchuk, S.V., Makkes, M.X.: Execution time estimation for workflow scheduling. In: Proceedings of the 9th Workshop on Workflows in Support of Large-Scale Science, WORKS '14, pp. 1–10. IEEE Press (2014)
8. D. Theiner et al.: Reduction of calibration time of distributed hydrological models. In: Proceedings of the International Conference on Hydroinformatics. Acropolis, Nice (2006)
9. Bacigalupo, D.A., et al.: An investigation into the application of different performance prediction methods to distributed enterprise applications. J. Supercomput. **34**(2), 93–111 (2005)
10. Deelman, E., Vahi, K., Juve, G., Rynge, M., Callaghan, S., Maechling, P.J., Mayani, R., Chen, W., da Silva, R.F., Livny, M., Wenger, K.: Pegasus, a workflow management system for science automation. Future Gener. Comput. Syst. **46**, 17–35 (2015)
11. Dubach, C., Cavazos, J., Franke, B., Fursin, G., O'Boyle, M.F., Temam, O.: Fast compiler optimisation evaluation using code-feature based performance prediction. In: Proceedings of the 4th International Conference on Computing Frontiers, pp. 131–142 (2007)
12. Eder, J., Pichler, H.: Probabilistic calculation of execution intervals for workflows. In: Proceedings of the 12th International Symposium on Temporal Representation and Reasoning, 2005, TIME 2005, pp. 183–185 (2005)
13. Eswaradass, A., Sun, X.H., Wu, M.: Network bandwidth predictor (NBP): a system for online network performance forecasting. In: Proceedings of the Sixth IEEE International Symposium on Cluster Computing and the Grid, 2006, CCGRID 06, vol. 1, pp. 265–268 (2006)
14. Fahringer, T., Prodan, R., Duan, R., Hofer, J., Nadeem, F., Nerieri, F., Podlipnig, S., Qin, J., Siddiqui, M., Truong, H.L., Villazon, A., Wieczorek, M.: ASKALON: A development and Grid computing environment for scientific workflows. In: Workflows for e-Science, pp. 450–471. Springer (2007)
15. Nadeem, F., et al.: Soft benchmarks-based application performance prediction using a minimum training set. In: Proceedings of the International Conference on e-Science and Grid Computing, p. 71 (2006)
16. Gelenbe, E., Montagne, E., Suros, R., Woodside, C.M.: A performance model of block structured parallel programs. In: International Workshop on Parallel Algorithms & Architectures. Luminy (1986)
17. Gibbons, R.: A historical application profiler for use by parallel schedulers. In: Proceedings of the Job Scheduling Strategies for Parallel Processing. Springer (1997)
18. Gil, Y., Deelman, E., Ellisman, M., Fahringer, T., Fox, G., Gannon, D., Goble, C., Livny, M., Moreau, L., Myers, J.: Examining the challenges of scientific workflows. Computer **40**(12), 24–32 (2007)
19. Glatard, T., Montagnat, J., Pennec, X.: A probabilistic model to analyse workflow performance on production grids. In: Proceedings of the 8th IEEE International Symposium on Cluster Computing and the Grid, 2008, pp. 510–517. Lyon (2008)
20. Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I.H.: The weka data mining software: an update. SIGKDD Explor. Newsl. **11**(1), 10–18 (2009)
21. Haykin, S.: Neural Networks: A Comprehensive Foundation, 3rd edn. Prentice-Hall, Inc., Upper Saddle River (2007)
22. Hiden, H., Woodman, S., Watson, P.: A framework for dynamically generating predictive models of workflow execution. In: Proceedings of the 8th Workshop on Workflows in Support of Large-Scale Science, WORKS '13, pp. 77–87. ACM (2013)
23. Taylor, I., et al.: Distributed computing with triana on the grid. Concurr. Comput. Pract. Exp. **17**(9), 1197–1214 (2005)
24. Ipek, E., de Supinski, B.R., Schulz, M., McKee, S.A.: An approach to performance prediction for parallel applications. In: Proceedings of the 11th International Euro-Par Conference on Parallel Processing, Euro-Par'05, pp. 196–205. Springer, Berlin, Heidelberg (2005)
25. Iverson, M.A., Özgüner, F., Potter, L.: Statistical prediction of task execution times through analytic benchmarking for scheduling in a heterogeneous environment. IEEE Trans. Comput. **48**(12), 1374–1379 (1999)
26. Jolliffe, I.: Principal Component Analysis. Springer, New York (2002)
27. Kapadia, N.H., Fortes, J.A., Brodley, C.E.: Predictive application-performance modeling in a computational grid environment. In: Proceedings of the Eighth International Symposium on High Performance Distributed Computing, pp. 47–54. IEEE (1999)
28. Kim, K.H., Ellis, C.: Performance analytic models and analyses for workflow architectures. Inf. Syst. Front. **3**(3), 339–355 (2001)
29. Lee, B.C., Brooks, D.M., de Supinski, B.R., Schulz, M., Singh, K., McKee, S.A.: Methods of inference and learning for performance modeling of parallel applications. In: Proceedings of the 12th ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming, PPoPP '07, pp. 249–258. ACM (2007)
30. Li, H., Chen, J., Tao, Y., Gro, D., Wolters, L.: Improving a local learning technique for queue wait time predictions. In: Proceedings of the Sixth IEEE International Symposium on Cluster Computing and the Grid. Singapore (2006)
31. Li, H., Groep, D., Templon, J., Wolters, L.: Predicting job start times on clusters. In: CCGRID '04: Proceedings of the 2004 IEEE International Symposium on Cluster Computing and the Grid, pp. 301–308 (2004)
32. Li, H., Groep, D., Wolters, L., Templon, J.: Job failure analysis and its implications in a large-scale production grid. In: Proceedings of the Second IEEE International Conference on e-Science and Grid Computing, 2006, e-Science '06, pp. 27–27 (2006)
33. Li, J., Fan, Y., Zhou, M.: Performance modeling and analysis of workflow. IEEE Trans. Syst. Man Cybern. A **34**(2), 229–242 (2004)
34. Liang, D.: Dynamic prediction of critical path instructions. In: Proceedings of the 7th International Symposium on High-Performance Computer Architecture, HPCA '01, pp. 185–193. IEEE Computer Society (2001)

35. Liu, X., Chen, J., Liu, K., Yang, Y.: Forecasting duration intervals of scientific workflow activities based on time-series patterns. In: Proceedings of the IEEE Fourth International Conference on eScience, 2008, eScience '08, pp. 23–30 (2008)
36. Liu, X., Ni, Z., Yuan, D., Jiang, Y., Wu, Z., Chen, J., Yang, Y.: A novel statistical time-series pattern based interval forecasting strategy for activity durations in workflow systems. J. Syst. Softw. **84**(3), 354–376 (2011)
37. Ludäscher, B., Altintas, I., Bowers, S., Cummings, J., Critchlow, T., Deelman, E., Roure, D.D., Freire, J., Goble, C., Jones, M., Klasky, S., McPhillips, T., Podhorszki, N., Silva, C., Taylor, I., Vouk, M.: Scientific Process Automation and Workflow Management. Chapman & Hall, Boca Raton (2009)
38. Miu, T., Missier, P.: Predicting the execution time of workflow activities based on their input features. In: High Performance Computing, Networking, Storage and Analysis (SCC), 2012, pp. 64–72 (2012)
39. Mohr, B., Wolf, F.: Kojak—a tool set for automatic performance analysis of parallel programs. In: Euro-Par 2003 Parallel Processing, pp. 1301–1304. Springer (2003)
40. Mussi, P., Nain, P.: Evaluation of parallel execution of program tree structures. SIGMETRICS Perform. Eval. Rev. **12**(3), 78–87 (1984)
41. Nadeem, F., Fahringer, T.: Predicting the execution time of grid workflow applications through local learning. In: Proceedings of the Conference on High Performance Computing Networking, Storage and Analysis, SC '09, pp. 33:1–33:12. ACM (2009)
42. Nadeem, F., Fahringer, T.: Optimizing execution time predictions of scientific workflow applications in the grid through evolutionary programming. Future Gener. Comput. Syst. **29**(4), 926–935 (2013)
43. Nadeem, F., Thomas, F.: Using templates to predict execution time of scientific workflow applications in the grid. Proceedings of the 9th IEEE/ACM International Symposium on Cluster Computing and the Grid, 2009, CCGRID '09, pp. 316–323. IEEE Computer Society, Shanghai (2009)
44. Blaha, P., et al.: WIEN2k: an Augmented Plane Wave Plus Local Orbitals Program for Calculating Crystal Properties. Institute of Physical and Theoretical Chemistry, Vienna (2001)
45. Pietri, I., Juve, G., Deelman, E., Sakellariou, R.: A performance model to estimate execution time of scientific workflows on the cloud. In: Proceedings of the 9th Workshop on Workflows in Support of Large-Scale Science, WORKS '14, pp. 11–19. IEEE Press (2014)
46. Pllana, S., Fahringer, T.: Performance prophet: a performance modeling and prediction tool for parallel and distributed programs. In: Proceedings of the International Conference on Parallel Processing, pp. 509–516 (2005)
47. Priore, P., Fuente, D.D.L., Gomez, A., Puente, J.: A review of machine learning in dynamic scheduling of flexible manufacturing systems. Artif. Intell. Eng. Des. Anal. Manuf. **15**(3), 251–263 (2001)
48. Prodan, R., Nadeem, F., Fahringer, T.: Benchmarking grid applications for performance and scalability predictions. In: K.C. Li, C.H. Hsu, L.T. Yang, J. Dongarra, H. Zima (eds.) Handbook of Research on Scalable Computing Technologies. IGI Global (2010). doi:10.4018/978-1-60566-661-7.ch005. http://www.igi-global.com/bookstore/TitleDetails.aspx?TitleId=498
49. Felix, S., et al.: Performance, Scalability and quality of the meteorological grid workflow. In: Proceedings of the 2nd Austrian Grid Symposium, Innsbruck (2006)
50. Salverda, P., Tu ker, C., Zilles, C.: Accurate critical path prediction via random trace construction. In: Proceedings of the 6th Annual IEEE/ACM International Symposium on Code Generation and Optimization, CGO '08, pp. 64–73. ACM (2008)
51. da Silva, R.F., Juve, G., Rynge, M., Deelman, E., Livny, M.: Online task resource consumption prediction for scientific workflows. Parallel Process. Lett. (2015). doi:10.1142/S0129626415410030
52. Smith, W., Foster, I., Taylor, V.: Predicting application run times with historical information. J. Parallel Distrib. Comput. **64**(9), 1007–1016 (2004)
53. Snavely, A., Carrington, L., Wolter, N., Labarta, J., Badia, R., Purkayastha, A.: A framework for performance modeling and prediction. In: Supercomputing, ACM/IEEE 2002 Conference, pp. 21–21 (2002)
54. Sodhi, S., Subhlok, J., Xu, Q.: Performance prediction with skeletons. Cluster Comput. **11**(2), 151–165 (2008)
55. Susukita, R., Ando, H., Aoyagi, M., Honda, H., Inadomi, Y., Inoue, K., Ishizuki, S., Kimura, Y., Komatsu, H., Kurokawa, M., Murakami, K.J., Shibamura, H., Yamamura, S., Yu, Y.: Performance prediction of large-scale parallell system and application using macro-level simulation. In: Proceedings of the 2008 ACM/IEEE Conference on Supercomputing, SC '08, pp. 20:1–20:9. IEEE Press (2008)
56. Oinn, T., et al.: Taverna: a tool for the composition and enactment of bioinformatics workflows. Bioinformatics **20**(17), 3045–3054 (2004)
57. Taylor, V., Wu, X., Geisler, J., Stevens, R.: Using kernel couplings to predict parallel application performance. In: Proceedings of the 11th IEEE International Symposium on High Performance Distributed Computing, HPDC '02, pp. 125–139. IEEE Computer Society (2002)
58. Tirado-Ramos, A., Tsouloupas, G., Dikaiakos, M., Sloot, P.: Grid resource selection by application benchmarking for computational haemodynamics applications. In: Computational Science, ICCS 2005, vol. 3514, pp. 534–543. Springer, Berlin, Heidelberg (2005)
59. Vraalsen, F., Aydt, R., Mendes, C., Reed, D.: Performance contracts: Predicting and monitoring grid application behavior. In: Grid Computing, GRID 2001, vol. 2242, pp. 154–165. Springer, Berlin, Heidelberg (2001)
60. Witten, I.H., Frank, E.: Data Mining: Practical Machine Learning Tools and Techniques, 2nd edn. Morgan Kaufmann, San Francisco (2005)
61. Wolski, R., Spring, N.T., Hayes, J.: The network weather service: a distributed resource performance forecasting service for metacomputing. Future Gener. Comput. Syst. **15**(5–6), 757–768 (1999)
62. Wombacher, A., Rozie, M.: Piloting an empirical study on measures forworkflow similarity. In: Proceedings of the IEEE International Conference on Services Computing, SCC '06, pp. 94–102. IEEE Computer Society (2006)
63. Wu, Q., Datla, V.V.: On performance modeling and prediction in support of scientific workflow optimization. In: Proceedings of the 2011 IEEE World Congress on Services, SERVICES '11, pp. 161–168. IEEE Computer Society (2011)
64. Wu, X., Taylor, V., Paris, J.: A web-based prophesy automated performance modeling system. In: Proceedings of the International Conference on Web Technologies, Applications and Services (WTAS2006), pp. 17–19 (2006)
65. Yang, L., Foster, I., Schopf, J.M.: Homeostatic and tendency-based CPU load predictions. In: Proceedings of the International Parallel and Distributed Processing Symposium, 2003, pp. 9–17. IEEE (2003)
66. Yang, L.T., Ma, X., Mueller, F.: Cross-platform performance prediction of parallel applications using partial execution. In: Proceedings of the 2005 ACM/IEEE Conference on Supercomputing, SC '05, pp. 40–44. IEEE Computer Society (2005)
67. Yero, E.J.H., Henriques, M.A.A.: Contention-sensitive static performance prediction for parallel distributed applications. Perform. Eval. **63**(4), 265–277 (2006)

Farrukh Nadeem is gold medalist in B.Sc. and completed M.Sc. Computer Science from University of Punjab, Pakistan. He has a second master degree "Master of Business Administration (MBA)" from Islamia University, Pakistan. He completed his Ph.D. with distinction in computer science in 2009 from the University of Innsbruck, Austria. He is an associate professor at Faculty of Computing and Information Technology, King Abdulaziz University, Jeddah. He holds several distinctions and awards during his educational career. He has been involved in several Austrian research projects and is working on couple of Saudi research and development projects. He has gained professional trainings on Cloud Computing and High Performance Computing. He has set up a "Grid Computing Infrastructure" at Faculty of Computing and Information Technology, King Abdulaziz University, Jeddah. He is a member of program committees of several conferences and editorial board member of journal of Modern Education and Computer Science. Farrukh has authored more than 29 conference and journal research papers, including four book chapters. He has been awarded President (King Abdulaziz University) Certificate of Appreciation and cash award for one of his journal publication. His main research interests include performance modeling and prediction, distributed systems, particularly the Grid and the Cloud, and decision support systems.

Daniyal Alghazzawi obtained his Bachelor's degree with honor in Computer Science from King Abdulaziz University in 1999. He completed his Master's degree and Doctorate in the field of Computer Science at the University of Kansas at the United States in 2007. He also obtained another Master's degree in Teaching and Leadership from University of Kansas in 2004, which helped him to develop his teaching and leadership skills. He also obtained the certificate of Management International Leadership (LMI) and has been the Head of the Information Systems department, Faculty of Computing and Information Technology for over five years during which he organized many workshops, and international and domestic conferences. Currently he is serving as a Vice dean of Development at deanship of IT. In 2010 he joined University Essex and since then has published with them several researches in the filed of smart environment. In 2017, he got promoted as a consideration for his output in research field which exceeded 90 researches and books in the filed of intelligent systems and Information security. He has 3 pending patents. He also served as a reviewer and editor for international journals, workshops and contests. He is also the head of the Information Security Research Group at King Abdulaziz University. His research interests include smart e-learning, information security and computational intelligence.

Abdulfattah Mashat graduated with a Bachelor of Science from King Abdulaziz University in 1989. He completed his MS and PhD in computer science from University of Leads, UK in 1999. He is currently the President of University of Jeddah (UJ), Jeddah, Saudi Arabia. He came to his present position after spending three years as Vice-President for Development at King Abdulaziz University (KAU), Jeddah, Saudi Arabia. He was also appointed as the Dean of the Faculty of Computer and Information Technology. During his tenure as the College Dean he lead successful effort for ABET accreditation of the Faculty's three programs. He also has been Director of Information Technology Center, King Abdulaziz University for two years. Prior to his Deanship position he spent several years as the Dean of Admission at the University and was instrumental in the development and implementation of a sophisticated online electronic system called "On Demand University Service" (ODUS Plus). He is a member of several professional and scientific organizations including ACM, IEEE, and also serves on the governing boards of several Saudi universities and on advisory boards of several national and international ranking systems. Among his current interests is developing and implementing key performance indicators for multitudes of university activities. He has authored more than 55 conference and journal research articles. His research interests include performance assessment of asysnchronous transfer mode networks, algorithms and protocols for multimedia systems, and models for video traffic VBR.

Khalid Fakeeh obtained his BS Computer Science from Northrop University in 1985. He completed his MS and PhD in Computer Science from The George Washington University in 1990 and 1993, respectively. He has also completed several short courses and trainings in several fields including leadership management, information technology strategy, distance learning, Effective Personal Productivity LMI, Leadership MQT International Inc., Microsoft Research, etc. He is a professor at Department of Information Systems, Faculty of Computing and Information Technology, King Abdulaziz University, Jeddah, Saudi Arabia. He has been head of Department of Information Systems for eight years and Dean of Faculty of Computing and Information Technology for five years. He also has been the adviser to Ministry of Information, Jeddah, Saudi Arabia. He has authored more than fifty conference and journal publications. He is also an editorial board member of the International Arab Journal of Information Technology and International Journal of Information Technology and Web Engineering. He has over 20 years experience in Expert and Decision Support Systems with a special interest in Ethics and Computing and the socio-psychological aspects of Information Systems as well as media perceptions and cultural and organizational aspects of Information Systems.

Abdullah Almalaise obtained his BS of Computer Science from University of Southern Mississippi, USA, in 1990. He completed his Master of Management Information Systems from University of Illinois at Springfield, IL, USA in 1992. He completed his doctorate in computer science in 2003 from George Washington University, United Stated of America. He has overall 20 years of experience in education field, mainly in teaching and curriculum building of several courses. He has worked on many administrative posts in educational institutes. Currently, he is working as a Vice Dean for Graduate Studies and Scientific Research at Faculty of Computing and Information Technology, King Abdulaziz University Jeddah, Saudi Arabia. He is also head of Computer Skills Department. He has been, and Chairman Department of Information Systems during 2012–2014. During 2010–2012, he was Chairman Computing and Information Technology Department, Jeddah Community College, and Chairman, Management Information Systems Department College of Business Administration (CBA), King Abdulaziz University from 2005 to 2008. He was also Chair of General Studies Department Computer Science Department, Technical College of Jeddah. He has expertise in training and supervising the team to learn technical, research and academic activities. He has more than 45 conference and journal publications and two books. Three of his books are under publications. His research interests include Collaborative Software, Distributed Systems Conflict Measurements, e-Business and e-Government, Information Systems, and Artificial Intelligence.

Hani Hagras received the B.Sc. and M.Sc. degrees in electrical engineering from Alexandria University, Alexandria, Egypt, and the Ph.D. degree in computer science from the University of Essex, Colchester, U.K. He is a Professor in the School of Science and Electronic Engineering, Director of the Computational Intelligence Centre and the Head of the Fuzzy Systems Research Group in the University of Essex, UK. His major research interests are in computational intelligence, notably type-2 fuzzy systems, fuzzy logic, neural networks, genetic algorithms, and evolutionary computation. His research interests also include ambient intelligence, pervasive computing and intelligent buildings. He is also interested in embedded agents, robotics and intelligent control. He has authored more than 300 papers in international journals, conferences and books. He is a Fellow of the Institute of Electrical and Electronics Engineers (IEEE) and he is also a Fellow of the Institution of Engineering and Technology (IET (IEE). He was the Chair of IEEE Computational Intelligence Society (CIS) Senior Members Sub-Committee. His research has won numerous prestigious international awards where most recently he was awarded by the IEEE Computational Intelligence Society (CIS), the 2013 Outstanding Paper Award in the IEEE Transactions on Fuzzy Systems and he was also awarded the 2006 Outstanding Paper Award in the IEEE Transactions on Fuzzy Systems. He is an Associate Editor of the IEEE Transactions on Fuzzy Systems. He is also an Associate Editor of the International Journal of Robotics and Automation, the Journal of Cognitive Computation and the Journal of Ambient Computing and Intelligence. He is a member of the IEEE Computational Intelligence Society (CIS) Fuzzy Systems Technical Committee and IEEE CIS conference committee. Prof. Hagras chaired several international conferences where he served as the General Co-Chair of the 2007 IEEE International Conference on Fuzzy systems London.