International Journal of Grid and Distributed Computing
Vol. 10, No. 1 (2017), pp.245-254
http://dx.doi.org/10.14257/ijgdc.2017.10.1.21

Using Predictive Analytics for Predicting Host Availability in Desktop Grids

Muhammad Khalid Khan[1] and Tariq Mahmood[2]

PAF – Karachi Institute of Economics & Technology, Karachi, Pakistan[1]
Institute of Business Administration, Karachi, Pakistan[2]
khalid.khan@pafkiet.edu.pk[1], *tmahmood@iba.edu.pk*[2]

Abstract

Desktop grid systems are one of the largest paradigms of distributed computing in the world. The idea is to use the idle and underutilized processing cycles and memory of the desktop machines to support large scale computation. The design issues in desktop grid systems are complex because the hosts (desktop machines) participating in the computation do not work under one administrative control and can become unavailable at any point in time. The heterogeneity and volatility of computing resources, for example, diversity of memory, processors, and hardware architectures also play its role. To get fruitful results from such hostile environment, scheduling tasks to better hosts become one of the most important issues. In this paper, we have predicted the host availability in desktop grid systems by using Predictive Analytics (PA) that can help in scheduling tasks to highly available hosts. We have presented a comprehensive, high-level evaluation of standard PA techniques to predict host availability in desktop grids with the aim to determine the relatively better algorithms. We addressed both PA perspectives, i.e., classification and regression. We used the following standard classification algorithms: k-Nearest Neighbour (k-NN) for Lazy Learning technique, Naïve Bayes for Bayesian learning technique, LibSVM library for Support Vector Modeling (SVM) technique, Random Forest for Tree Induction technique, and Multi-Layer Perceptron for Neural Network technique. We found that the level of selected threshold for availability is critical for acquiring accurate predictions, and k-NN gives the best accuracy across all thresholds. Also, precision-wise, SVM gives perfect performance (100%) across all thresholds followed closely by Neural Networks. We used Multiple Linear Regression (MLR), Polynomial Regression (PR) and MLP for regression, and found that MLP gives the best performance, followed by PR and MLR.

Keywords*: Desktop Gird Systems; Resource Allocation; Predictive Analytics*

1. Introduction

There was a time when super computers were the only option for high-end computing and storage. These days, desktop grid systems have developed into more cost-effective substitutes. There is an abundance of available bandwidth for communication and desktop grids use it thoroughly for the utilization of idle processing cycles and memory of millions of computers connected through Internet, or through any other type of network. The desktop grid system infrastructure consists of *N* number of desktop machines in which one would be termed as *master* and the others would be known as *hosts*/*workers.* Practically a desktop grid system project has several servers to create tasks, distribute them, record the tasks and corresponding results, and finally, aggregate the results of a set of tasks. Normally, when a host is idle, then it is time to work on the tasks assigned by server. After finishing the tasks, the results are sent to the server. There are certain limitations to desktop grid systems which include resource management, scheduling,

ISSN: 2005-4262 IJGDC

verification of results, computation time, fault tolerance, security breaches, connectivity and bandwidth issues. The fact that hosts are not under the administrative control of server, arises various design and deployment challenges in which scheduling or resource allocation is of utmost importance. Typically, resource scheduling policies use limited number of hardware parameters along with availability and reliability for resource selection in desktop grid systems. These parameters may get altered at any given point because of many reasons such as hardware or software malfunction on host, network congestion or failure etc. A manual prediction process requires expertise and background knowledge to identify performance bottlenecks [1], correlate dynamic behavior with performance parameters, formulate different hypothesis and consider validation tests. This can potentially lead to sub-optimal resource selection [2]. These activities potentially demand enormous expertise from grid experts. Considering these problems, researchers have attempted to autonomously predict or estimate host availability using diverse types of techniques.

The related domain of making predictions is called Predictive Analytics (PA), a recent, yet well-established, field which amalgamates a set of standard techniques which can be used for estimating predictions from diverse types of data sets [3]. It belongs to supervised learning approach of the Machine Learning domain [4], and is based on a variety of statistical techniques from modeling, machine learning and data mining that analyze current and historical facts to make predictions about future or otherwise unknown events. The idea is to learn a mathematical model which acts as a mapping from the input data (called the *predictors*) to an output variable which is to be predicted (called the *label*). Then, the learnt model is used to make predictions in new, previously unknown situations. Two types of predictive models can be learnt: 1) classification models, 2) regression models. In classification, string labels (typically binary) are predicted and in regression, numerical labels are predicted. Standard PA techniques include [5] Lazy Learning, Decision Tree Induction, Bayesian Modeling, Artificial Neural Networks, Support Vector Modeling, and Regression (function fitting). Each of these techniques provides a set of standard algorithms for making predictions. PA has already been applied successfully and extensively in diverse corporate and research domains across the globe [6].

The PA process with its diverse algorithmic base, can be usefully applied to rein in the heterogeneity of desktop grids. Researchers have used many approaches for predicting resource availability in the context of clusters, cluster of grids, supercomputers, desktop computers, desktop grids, and peer-to peer systems. For instance, Reddy et. al. presented an adaptive performance tuning approach to scheduling in [7]. The focus is on adaptively decomposing jobs into tasks, selecting a set of relevant parameters, and use heuristic scheduling to improve the overall throughput of the desktop grid. The parameter set includes CPU availability, memory availability, combined mean availability, job size, job type, available grid resources and execution history. The proposed approach shows considerable benefit as compared to baseline performance tuning. Similar to [7], Zhao et. al. gave an adaptive scheduling algorithm based on resource attribute selection and load balancing in [8]. It takes advantage of resources to schedule tasks more efficiently. Nodes are selected based on the degree of matching between current and previous tasks and performance of nodes when executing previous tasks. The status of current tasks is saved by the scheduling server and used later on to allocate nodes (prediction) when similar tasks are received. Appreciable progress in resource utilization is shown as compared to standard task execution frameworks (Falkon, Sparrow, and Gearman).

Moreover, in [9], Canon et. al. addressed the problem of dealing with uncertainty of host availability. They aim to achieve stability, i.e., the actual throughput should be always less than the predicted (expected) one. Authors simulate five scheduling policies with actual traces and the Longest Processing Time (LPT) policy turns out to be most stable. In [10], cluster analysis (a non-supervised learning technique) is applied to extract

similar groupings of resource usage pattern data. The aim is to improve Quality of Services (QoS) for desktop grid clients. The identified groupings are then used to make predictions about available resources, with around 80% accuracy. On a similar pattern, a job execution service is proposed by Kang et.al. in [11] which dynamically virtualizes resources in order to use underutilized, cheap resources for improving business QoS.

The work done in [12] proposes and validates accurate model of response-time predictions for effective scheduling, without compromising QoS. They use instance-based learning and an analytical model (named SdPN) to make predictions about waiting and running time (for jobs). Linear regression is then used to discover deviations (errors) in predictions, which are then fed back to improve future predictions. Perhaps the most relevant work is done in [13], which uses the C4.5 decision tree algorithm to predict the host availability, as states *available*, *busy* or *off*, based on standard input parameters of CPU and RAM. C4.5 gives almost 90% predictive accuracy.

Brevik et. al. predicted the duration for which a machine is available using the Maximum Likelihood Estimate (MLE) parametric method, along with Random Sample and Binomial non-parametric methods in [14]. All 3 methods show that availability duration can be predicted with quantifiable confidence bounds, based on the percentile of machine availability, although Binomial gives best performance. In [15], the authors use artificial neural networks to predict the load at each node and migrate the jobs to other more free nodes. The simulation results show that the average turnaround time per job for the proposed method has got considerable improvement over no-migration and resource exclusion algorithms. Significant performance benefits (more jobs completed in less turnaround time) are achieved as compared to a resource exclusion strategy.

A Hadoop/MapReduce-based implementation for desktop grids is proposed in [16], named MRA++, which caters for the heterogeneity of nodes during data distribution, task scheduling and job control. Results show that a time delay occurs in setting up MRA++, but is offset by the efficiency achieved later on, i.e., performance improvement of more than 70%. In [17], Gil et. al. proposed a nearest neighbor (NN)-based task scheduling approach which selectively allocate tasks to those resources that are suitable for the current situation of a desktop grid environment. The experimental results show better turnaround time and number of resources consumed, compared with FCFS. In [18], k-means clustering is used to classify resources according to their own task execution availability and result-return probability. Experimental results shows better performance of K-means (as compared to FCFS) with respect to reducing turnaround time and quantity of resources consumed.

In view of the above discussion, two research questions have been developed:

- Which PA technique, in combination with a standard algorithm, can generate acceptable performance in predicting host availability?
- In classification-based prediction, does changing the threshold of generating class labels of host availability has any significant impact on predictive performance?

In the next Section, we describe the research methodology, and then we discuss the results. After that, we present conclusion and future direction.

2. Research Methodology

To answer the above mentioned questions, we conducted a comprehensive high-level evaluation of PA techniques for predicting host availability in desktop grids. By high-level, we mean that our focus is on the application of all PA algorithms while using standard parameter settings; the intent is not to tune or modify the parameters of any single algorithm extensively (as can be deduced by applications in related work). So, our results will present a comparison of PA algorithms at a

more abstract level (high-level view) to create a type of road map for desktop grid decision makers. Our set of predictors is: *Cycles Per Second* (CPS), *Number of Cores* (NoCore), *Size of Memory* (MemSize), *Amount of Free CPU* (CPUFree) and *Amount of Free Memory* (MemFree), with the label being *Availability*. We focus on six standard PA techniques: Support Vector Modeling (SVM), Artificial Neural Networks (NN), Bayesian Learning (BL), Lazy Learning (LL), Decision Tree Induction (DTI), and Regression. The particular algorithms for each technique are LibSVM library (SVM), Multi-Layer Perceptron (ANN), Naïve Bayes (BL), K-NN (LL), Random Forest (DTI), Linear and Polynomial Regression (Regression). We conduct our experiments on the two standard predictions methods, i.e., classification and regression. For the former, label values are *Available* (AV) and *Not Available* (NotAV), and for the latter, Availability values are numerical. In classification, we particularly investigate the effect of changing the threshold of availability, i.e., we apply different numerical ranges in which a machine can be labeled as "AV"/"NotAV". In the following sections, we provide the details of the experimental methodology, followed by results and conclusions.

3. Experimental Setup

To run our experiments, we gathered data from multiple computers available on the infrastructure of an educational institution having varied resource configurations. This was done to ensure accurate performance assessment of PA algorithms. Specifically, we employed 18 computers having different combinations of CPS, NoCore and RAM sizes. CPU speed ranges were from 1.8 to 3.4 GHz., No. of Core were in range of 2 to 8 and RAM sizes were in range of 2 to 16 GB. Overall we collected 1.3 million traces from 18 computers in 30 days (7 hours per day) using the time interval of 10 seconds. In our high-level PA application, we used "R" [19] to conduct the preliminary statistical analysis (descriptive and inferential) and Rapid Miner [20] to apply the PA techniques. For each technique, we selected its well-known algorithm with the standard parameter setting offered by Rapid Miner.

3.1. Descriptive Statistics

We first carried out descriptive statistics to investigate deviation from normality. We have no missing, incomplete or inconsistent values. Overall, standard deviation is small, indicating lack of any outliers. CPS and CPUFree have smaller deviations than other variables. CPS, NoCore and MemSize are weighted and normalized with respect to standard deviation whereas CPUFree and MemFree are weighted on average and then normalized with respect to standard deviation. For Availability, a worker is counted available if availability is greater 9.03×10^{-4} otherwise it is counted as unavailable. Next, we generated the histograms of predictors with normal curve to further investigate the deviation from normality. Our analysis suggested that the dynamics of our desktop grid infrastructure is deviating the distribution of potential predictors from the normal behavior, i.e., there is heterogeneity which is representative of typical desktop grid environment. We do understand that the typical behavior should represent moderate amount of kurtosis (heavy-tailed distribution) with moderate level of skew but visual analysis as well as background knowledge do not indicate any reason to reject the normality assumption for any of our predictors. Even the availability (AV) histogram shows that the distribution is normal with negligible amount of skew (-0.017) and moderate negative kurtosis (-1.171).

3.2. Inferential Statistics - Classification

After descriptive statistics, we performed inferential statistics for our classification task. Specifically, we generated thresholds of Not Available (NotAV) and Available (AV) based on percentiles of Availability. For this, we selected the following 3 percentiles:

- 25% (named as Lower Quartile LQ)
- 37.5% (named as Between Quartile BQ) and
- 50% (named as Mid Quartile MQ).

We then carried out independent Sample T-Test to investigate whether a given predictor has significant difference of means across NotAV and AV. The relevant hypotheses in this case are:

- Null Hypothesis Ho: The predictor does not influence the PC availability
- Alternate Hypothesis H1: The predictor influences the PC availability

The results for LQ, BQ and MQ are shown in Table 1, 2 and 3. First, we consider the result for LQ, where we consider the assignments (Availability <= LQ) = NotAV, and (Availability > LQ) = AV. We found that:

- At lower percentiles of availability discretization, statistical significance is achieved for CPU-related variables (CPS, NoCore, CPUFree)
- At median percentile, no statistical significance is achieved for any predictor
- Lower percentile discretization reveals that CPU-related variables are more important predictors as compared to Memory related variables (which seems logical).

Table 1. Independent Sample T-Test Result for Low Quartile

(Availability <= LQ) = NotAV, (Availability > LQ) = AV

Variable	***AV***	***NotAV***	***P-value***
CPS	*0.00046036*	*0.00045798*	*0.329*
NoCore	*0.00046382*	*0.00044757*	*0.085*
MemSize	*0.00045889*	*0.00046240*	*0.737*
CPUFree	*0.000459527*	*0.00046049*	*0.479*
MemFree	*0.000458214*	*0.00046444*	*0.616*

Table 2. Independent Sample T-Test Result for between Quartile

(Availability <= BQ) = NotAV, (Availability > BQ) = AV

Variable	***AV***	***NotAV***	***P-value***
CPS	*0.000461103*	*0.000457550*	*0.104*
NoCore	*0.00046448*	*0.000451914*	*0.136*
MemSize	*0.000457510*	*0.000463533*	*0.519*
CPUFree	*0.000459013*	*0.000461030*	*0.101*
MemFree	*0.000456171*	*0.00046564*	*0.387*

Table 3. Independent Sample T-Test Result for Mid Quartile

(Availability <= MQ) = NotAV, (Availability > MQ) = AV

Variable	***AV***	***NotAV***	***P-value***
CPS	*0.0004598948*	*0.0004596453*	*0.906*
NoCore	*0.0004606542*	*0.0004588853*	*0.829*
MemSize	*0.0004554590*	*0.0004640852*	*0.340*

CPUFree	*0.0004594690*	*0.0004600715*	*0.613*
MemFree	*0.0004538945*	*0.0004656511*	*0.274*

3.3. Inferential Statistics - Regression

We next conducted inferential statistics for our regression task. Specifically, we calculated Pearson correlation between non-discretized predictors and non-discretized availability given in Table 4. Results suggest that:

- Barring CPS, correlations with CPUFree is significant at 5% level and with MemFree, MemSize and NoCores at 10% level
- All correlations are extremely weak (hovering around 0), prohibiting any concrete conclusion regarding the strength of predictors. The only conclusion is that robust regression models like neural networks and polynomial regression need to be adopted to get some meaning out of the regression process.

Table 4. Pearson Correlation Results of Predictors with Availability

Predictors	***Pearson Correlation***	***p-value***
CPS	*-0.009*	*0.675*
NoCore	*-0.039*	*0.068*
MemSize	*-0.041*	*0.057*
CPUFree	*0.054*	*0.011*
MemFree	*-0.040*	*0.064*

We now sum up the insights obtained from our statistical analysis as follows:

- Greater the number of cores, more is the amount of free memory
- Greater the memory size, greater is the amount of free memory
- Having a larger number of cores results in more cycles per second
- Increasing the cycles per second results in lesser amount of free CPU and memory
- Lesser the memory size, greater the cycles per second

Let's relate the above mentioned insights with the identification of better hosts in desktop grid environment. It was found that greater number of cores and memory size would increase the amount of free memory as the assigned jobs will be processed quicker with higher number of cores and greater memory size. Such host has a higher chance of completing the assigned task along with the tasks of resource owners. It was also observed that large number of cores increases the cycles per second that means host can perform the assigned job quicker but if the associated memory is low than this would increase cycles per seconds cause the CPU has to work more that results in unavailability of free CPU and memory. Such host would become less favorable to desktop grid environment. An ideal host should have more CPU cores, high CPU speed, more memory and availability to become a meaningful participant in desktop grid system.

4. Results

In this section, we present the classification and regression results of our proposed PA application.

4.1. Results – Classification

The algorithms applied for classification are Naïve Bayes, K-NN, We sampled a standard 70% training and 30% test set. To test the performance, we recorded the following standard parameters:

- TP: Number of True Positives
- FP: Number of False Positives
- FN: Number of False Negative

We then recorded the accuracy and precision and recall of the positive class (AV) as follows:

- Accuracy = (T P+TN)/n (n is the number of examples considered)
- Precision (AV) = TP/TP+FP = Probability that a positive prediction is correct
- Recall (AV) = TP/TP+FN = Probability of correctly predicting all instances of positive class from test set

Test performance of algorithms for LQ, BQ and MQ are given in Table 5, 6 and 7 respectively.

Table 5. Test Performance for Low Quartile

Algorithm	***Precision AV***	***Recall AV***	***Accuracy***
K-NN	*95.4*	*85.4*	*84.3*
Naïve Bayes	*96.7*	*74.9*	*74.9*
Random Forest	*98.9*	*75.4*	*75.0*
SVM	*100.0*	*75.0*	*75.0*
Neural Networks	*99.1*	*74.9*	*74.4*

Table 6. Test Performance for between Quartile

Algorithm	***Precision AV***	***Recall AV***	***Accuracy***
K-NN	*85.3*	*79.5*	*77.1*
Naïve Bayes	*83.2*	*67.0*	*61.7*
Random Forest	*96.8*	*62.9*	*62.3*
SVM	*100.0*	*62.5*	*62.5*
Neural Networks	*97.2*	*62.4*	*61.7*

Table 7. Test Performance for Middle Quartile

Algorithm	***Precision AV***	***Recall AV***	***Accuracy***
K-NN	*69.3*	*77.0*	*74.3*
Naïve Bayes	*57.9*	*58.3*	*58.2*
Random Forest	*98.8*	*50.1*	*50.2*
SVM	*100.0*	*50.0*	*50.0*
Neural Networks	*89.9*	*49.7*	*49.5*

From the above analysis we conclude the following:

- In terms of accuracy, K-NN gives the best performance for all thresholds LQ, BQ and MQ
- Accuracy values of all algorithms are best for LQ, followed by BQ, and then MQ

- In LQ and BQ, the accuracy values of Naïve Bayes, SVM, Random Forest and NN are almost the same. In MQ, Naïve Bayes shows somewhat better accuracy as compared to other three (whose accuracy is again almost the same).
- In terms of recall, K-NN gives the best performance for all thresholds LQ, BQ and MQ
- Recall values of all algorithms are best for LQ, followed by BQ, and then MQ
- In LQ and BQ, the recall of Naïve Bayes, SVM, Random Forest and NN is approximately same. In MQ, Naïve Bayes shows somewhat better recall as compared to other three (whose recall is again almost the same).
- In terms of precision, SVM gives 100% performance in each threshold.
- In LQ, the precision of all algorithms is close to 100%. However, for BQ and MQ, the precision of more robust algorithms (NN and Random Forest) becomes better than the precision of less robust ones (Naïve Bayes and K-NN).
- The trends of result of each algorithm, with respect to both accuracy and recall, are the same across each threshold LQ, BQ and MQ. For instance, K-NN gives the highest accuracy and NN gives the lowest accuracy across each threshold.

4.2. Results – Regression

We now present our results for regression, in which we considered two regression algorithms, i.e., Multiple Linear Regression (MLR) and Polynomial Regression (PR), along with Neural Network (NN). After training, we tested each algorithm on our test set by recording the Root Mean Squared Error (RMSE), which is a standard performance measure that records the deviation of the fitted availability model from the actual values of availability in the test set [21]. The results are given in Table 8, given below:

Table 8. Test Performance by Recording the Root Mean Squared Error

Algorithm	*Root Mean Square Errors*
NN	*0.001*
PR	*95.486*
MLR	*284.351*

As we can see, NN gives the best performance, followed by PR and then MLE, showing that neural networks are the best solution if we tackle the problem of predicting host availability as a regression problem. This is logical, given the sophisticated infrastructure offered by NN, as compared to a line or curve-fitting scenario of MLR and PR.

Thus, the research questions presented in introduction can be answered as follows:

- For accuracy and recall, K-NN gave the best performance whereas in terms of precision, SVM gave better results. NN gave best results when predicting host availability is taken as regression problem.

In classification-based prediction, changing the threshold of generating class labels of host availability has significant impact on predictive performance because accuracy, precision and recall values of all algorithms are different at each threshold. All algorithms gave best performance for LQ, followed by BQ and then MQ.

5. Conclusion & Future Work

We presented a comprehensive high-level application of standard Predictive Analytics (PA) techniques to predict host availability in desktop grid systems. Our problem is to identify the best host for the task assignment through prediction results. Our predictors include: *Cycles Per Second (CPS), Number of Cores (NoCore), Size of Memory*

(MemSize), Amount of Free CPU (CPUFree) and Amount of Free Memory (MemFree). Our label is *Availability*. For classification, we discretize Availability into *Available* (*AV*) and *Not Available* (*NotAV*). For this discretization, we used three Availability percentiles i.e. 25%, 37.5% and 50%. Our classification results show that performance (precision of AV class) of selected techniques at 25^{th} percentile is better than at 37.5^{th} percentile, which is better than 50^{th} percentile. We have found that in classification, thresholding plays an effective role and k-Nearest Neighbour gives the best accuracy across all thresholds. However, if we consider precision, then support vector modeling gives perfect performance (100%) across all thresholds, followed closely by neural networks. In regression, neural networks give the best performance, followed by polynomial regression and then multiple linear regression.

As future work, we plan to further validate our results over a larger time interval (between 2-6 months) to analyze whether the data gathered for extended period, reveals more meaningful information as compared to our current understanding. We also aim to use a larger set of representative predictors (currently we have used five predictors) to evaluate the impact of using larger set of predictors in predicting host availability. Finally, we plan to validate our results online, i.e., we will apply K-Nearest Neighbour (in terms of accuracy) at run-time in a live desktop grid environment, to check whether it predicts the best available host (for availability) or not.

References

[1] M.-K. Khan, I. Hyder, B. S. Chowdhry, F. Shafiq and H. M. Ali, "A novel fault tolerant volunteer selection mechanism for volunteer computing", Sindh University Research Journal—Science Series, vol. 44, no. 3, **(2012)**, pp. 138-143.
[2] K. Khan, I. Hyder, G. Ahmed and S. Begum, "A Group based fault tolerant scheduling mechanism to improve the application turnaround time on desktop grids", International Arab Journal of Information Technology, vol. 13, no. 2, **(2016)**, pp. 274-279.
[3] http://www.amazon.com/Predictive-Analytics-Power-Predict-Click/dp/1118356853.
[4] http://www.cs.cmu.edu/~tom/mlbook.html.
[5] T. Hastie, R. Tibshirani and J. Friedman, "The Elements of Statistical Learning", Springer, **(2008)**.
[6] http://www.amazon.com/Predictive-Analytics-Power-Predict-Click/dp/1117434512.
[7] K. H. K. Reddy, D. S. Roy and M. R. Patra, "A Comprehensive Performance Tuning Scheduling Framework for Computational Desktop Grid", International Journal of Grid and Distributed Computing, vol. 7, no. 1, **(2014)**, pp. 149-168.
[8] Y. Zhao, L. Chen, Y. Li, P. Liu, X. Li and C. Zhu, "RAS: A Task Scheduling Algorithm Based on Resource Attribute Selection in a Task Scheduling Framework", In Internet and Distributed Computing Systems, Springer Berlin Heidelberg, **(2013)**, pp. 106-119.
[9] L. C. Canon, A. Essafi and D. Trystram, "A Proactive Approach for Coping with Uncertain Resource Availabilities on Desktop Grids", FEMTO-ST, Tech. Rep. RRDISC2014-1, **(2014)**.
[10] M. Finger, G. C. Bezerra and D. R. Conde, "Resource use pattern analysis for predicting resource availability in opportunistic grids", Concurrency and Computation: Practice and Experience, vol. 22, no. 3, **(2010)**, pp. 295-313.
[11] W. Kang, H. H. Huang and A. Grimshaw, "Achieving high job execution reliability using underutilized resources in a computational economy", Future Generation Computer Systems, vol. 29, no. 3, **(2013)**, pp. 763-775.
[12] J. L. Lerida, F. Solsona, P. Hernandez, F. Gine, M. Hanzich and J. Conde, "State-based predictions with self-correction on Enterprise Desktop Grid environments", Journal of Parallel and Distributed Computing, vol. 73, no. 6, **(2013)**, pp. 777-789.
[13] S. A. Salinas, C. G. Garino and A. Zunino, "An architecture for resource behavior prediction to improve scheduling systems performance on enterprise desktop grids", In Advances in New Technologies, Interactive Interfaces and Communicability, Springer Berlin Heidelberg, **(2012)**, pp. 186-196.
[14] J. Brevik, D. Nurmi and R. Wolski, "Automatic methods for predicting machine availability in desktop grid and peer-to-peer systems", In Cluster Computing and the Grid, CCGrid, IEEE International Symposium, IEEE, **(2004)**.
[15] S. Naseera and K. M. Murthy, "Prediction Based Job Scheduling Strategy for a Volunteer Desktop Grid. In Advances in Computing", Communication, and Control, Springer Berlin Heidelberg, **(2013)**, pp. 25-38.

[16] J. C. Anjos, I. Carrera, W. Kolberg, A. L. Tibola, L. B. Arantes and C. R. Geyer, "MRA++: Scheduling and data placement on MapReduce for heterogeneous environments", Future Generation Computer Systems, vol. 42, **(2015)**, pp. 22-35.
[17] J. M. Gil, S. Kim and J. Lee, "Task Replication and Scheduling Based on Nearest Neighbor Classification in Desktop Grids", In Ubiquitous Information Technologies and Applications, Springer Netherlands, **(2013)**, pp. 889-895.
[18] J. M. Gil, S. Kim and J. Lee, "Task scheduling scheme based on resource clustering in desktop grids", International Journal of Communication Systems, vol. 27, no. 6, **(2014)**, pp. 918-930.
[19] https://www.r-project.org/.
[20] https://rapidminer.com.
[21] J. Camm, J. Cochran, M. J. Fry, J. Ohlmann and D. Anderson, "Essential of Business Analytics", Cengage Learning, **(2014)**.

Authors

Muhammad Khalid Khan received Ph.D. degree in Computer Science in 2016. He also received MS-Computer Science and MBA-MIS degrees in 2005 and 2008 respectively. He is working as the Director - College of Computing and Information science at PAF-KIET, Karachi. He has more than 13 years experience in academia and close to 5 year in software industry. He has more than 20 publications on his accord.

