A Study on Workflow Execution Time Prediction

Mohamed Ismail
B.Tech Student
Department of Computer Science Engineering
Federal Institute of Science And Technology (FISAT), India

Sajith Geo Paul
B.Tech Student
Department of Computer Science Engineering
Federal Institute of Science And Technology (FISAT), India

Rohith
B.Tech Student
Department of Computer Science Engineering
Federal Institute of Science And Technology (FISAT), India

Thomas Mani
B.Tech Student
Department of Computer Science Engineering
Federal Institute of Science And Technology (FISAT), India

Dr. Jyothish K. John
Head of Department
Department of Computer Science Engineering
Federal Institute of Science And Technology (FISAT), India

Abstract

Several applications depend on the performance of workflow execution time prediction for differing input data. In any case, such estimates are hard to produce in the cloud. Cloud computing conditions give an expansive scope of focal points for the organization of logical application, especially the potential to provide a great number of resources under a pay-per-use plan. This trademark meets the needs of researchers who define applications as a collection of workflows. Scientific workflows consist of tasks with conditions among them. These workflows are huge in size and require a great number of resources to run. Some of the major drawbacks of the current model are the limited accuracy and the impractical pre-requisites required. The model we propose involves machine learning ensembles employing multiple learners capable of balancing out the weaknesses of each with the strengths offered by others. The proposed approach also uses extracted attributes as parameters to predict the execution time of workflow applications in dynamic environments such as cloud.

Keywords: Cloud Computing, Runtime, Hardware, Machine Learning, Computational Modeling, Predictive Models, Analytical Models, Performance Prediction, Workflow Application Execution Time

I. Introduction

Cloud computing is the delivery of computing services, especially storage and computing power, without direct management by the user. It is the best possible way to provide enterprise applications and preferred solutions for companies aiming to develop their infrastructure or to launch new products. It is widely regarded as the future of computing, eliminating the need for maintaining individual server systems for organisations. Migration to cloud-based systems, however, is considered a challenge for firms that run high performance software such as workflows. Determining the best-suited cloud provider for workflows is hence considered a major obstacle for cloud migration.
A cloud service provider with the following features can be determined to be the best for a particular workflow software:

- Availability of complete support for the workflow system on the required VM or the cloud computing provider.
- Minimal time required to complete the execution of the workflow application in the cloud environment.

This can be determined by taking into account several factors such as CPU utilisation, Peak Memory used, Bandwidth etc.

A workflow application is a software application which can be subdivided into processes. The process usually requires a series of steps to be automated via software. A workflow is a directed graph in which the vertices represent the tasks of the workflow and the edges represent data or control dependencies among tasks. Examples of workflow applications include Montage, Wein2k and Blender etc.

The rest of the paper is organized as Section II surveys various workflow execution time prediction methods, Section III is the comparative study of the methods, Section IV has the problem statement and Section V contains the conclusion.

II. Machine Learning based Approaches

A. *Supervised Learning Approaches*

Farrukh Nadeem, Daniyal Alghazzawi, Abdulfattah Mashat, Khalid Faqeeh and Abdullah Almalaise [1] proposed an approach to efficiently predict execution times of workflow applications run in grid environments using supervised learning algorithms. This approach makes use of two multi-stage machine learning ensembles namely: Mixture of Experts and Dynamic Selection of Expert. Mixture of Experts is a mixture of multiple strong learners capable of generating their own predictions. These individual learners are referred to as experts. The predictions generated by individual experts are then weighted to determine the final output. The weights are added to the experts by means of a gated network. The resultant values are then combined together to develop the final output of the model. The model offers improved accuracy as the weaknesses of individual experts can be evened out using the strengths offered by other experts. Dynamic Selection of Expert is another method that exercises the strength offered by multiple learners. The method evaluates the accuracy of each learner and picks the one with the highest accuracy.

Andréa Matsunaga and José Fortes [2] proposed a model based on machine learning that is adept in predicting the resources and time utilized by applications. Resource consumption by an application can be useful for execution time prediction. This approach identifies the best machine learning methods and predicts utilization of resources by applications such as execution time, memory and disk requirements. This work uses supervised machine learning methods. It measures the impact of attributes on prediction accuracy and evaluates the performance on increasing the number of attributes, accounting system's performance and application-specific characteristics. The system is made to learn using a collection of historical data of previous executions of the application. Each previous observation was used as training data, providing a set of attributes and the actual outcome. This work extends the Predicting Query Runtime (PQR) algorithm. Regression functions are added at the leaves of the PQR tree to provide a fine-grained prediction.

Michael A. Iverson, Füsun Özgüner, and Lee Potter [3] proposed a method for predicting execution time for scheduling tasks in heterogeneous environments. The work makes use of a hybrid approach to generate observations capable of describing the input features of the data alongside the capabilities of the machine involved using techniques like analytic benchmarking. These observations are then used to predict execution time using statistical techniques. The execution time is regarded as the output of a function that utilizes the parameters specified in the data input. The approach uses k-Nearest Neighbours regression algorithm to estimate the execution time. Non-parametric regression techniques are used in this research as the estimate generated is solely dependent on the availability of previous observations. The algorithm developed for predicting the execution time consists of two parts. The first part is responsible for computing the execution time of tasks performed on a single machine. The second part performs the parameterization of different machines.

Tudor Miu and Paolo Missier [4] proposed a model that makes use of supervised machine learning methods in predicting the execution time of workflow applications utilizing datasets containing input features. This work exploits the observation that better execution time prediction is done by using historical data to train regression models which then help to forecast the execution time depending upon select features of inputs. This work focuses on services that are workflow activities and have a history of executions which contains input datasets and corresponding execution time. This historical data is used to make the execution time estimates for any new inputs. Predictions are done for individual subtasks instead of entire workflow compositions.

Muhammad Hafizhuddin Hilman, Maria A. Rodriguez and Rajkumar Buyya [5] proposed a model based on an online incremental machine learning approach. Features related to the run-time parameters and records generated by a resource consumption monitor are retrieved by the model. Machine learning methods are used to extract patterns and information from the data available to generate predictions on events. Multiple prediction models are implemented for each workflow application. Availability of multiple models aid the system in optimizing the prediction by fine tuning each model as per the requirements of the specific task. The workflows are modelled as directed acyclic graphs. The execution of workflows is assumed to be done by a Waas platform. Workflows are constantly monitored for resource usage. There are two phases. In the first phase of the prediction, the pre-runtime configurations related to the required resource type is extracted by the algorithm. In the next phase, the resource consumption will be estimated from the given pre-runtime features of a task. The estimated resource consumption will be processed and then will be used as a feature to determine task execution time.

Hugo Hiden, Simon Woodman and Paul Watson [6] developed a framework to capture live data to build predictive models. These models are used to forecast the attributes of workflows. The models can also be updated when needed or at a fixed interval. The framework is used to estimate the performance of workflows based on collected historical data. The framework also allows us to combine multiple predictive models of unit operations of a workflow and thus multiple models of unit operations can be combined to create models of different workflows. To compute the total execution time, the execution time of the constituent blocks is found out. The execution time of each block is found by defining a relationship between execution time and various block parameters like block settings, machine characteristics, input size, etc. There are three broad categories of model types: a linear model based on a mix of execution data stored in the performance database, a non-linear model in which the execution time obeys a non-linear relationship with the gathered performance data, and when there exists no correlation between the duration prediction for the block and the observed execution data, a model depending on the average execution time computed from the observed execution on the block. Based on the nature of the block, a model is adopted. The system builds a model of each type and the model showing the best performance is chosen. These models are updated based on new performance data generated. The update is done periodically or when a required amount of data is obtained.

B. Reinforcement Learning Approaches

Enda Barrett, Enda Howley and Jim Duggan [7] proposed a approach for scheduling where a genetic algorithm is used to evolve solutions to workflow scheduling problems. The Markov Decision Process is then employed to choose from evolved schedules such that the chosen one maximizes the output in the long run. The learning agent learns to use the observed rewards for each case.

Genetic algorithms are used which are basically stochastic search and optimization techniques based on evolution. All possible solutions are initially evaluated in an iterative manner and the best solution is chosen and the next generation of solutions are created from it. The successive solutions improve the productivity of the system. Each solution is the linking of a subtask to a resource. The suitability of solutions is determined by a fitness function. Each possible schedule generated is represented by a Markov Decision Process model which in turn is solved by the learning agent to calculate its efficiency. This approach is capable of adapting with respect to dynamically changing environments such as cloud environments and can account for uncertainties such as peak times.

Athanassios M.Kintsakis, Fotis E.Psomopoulos, Pericles A.Mitkas [8] proposed an approach that uses Reinforcement Learning to optimise the scheduling of workflow applications. This method is an efficient alternative to existing Workflow Management Systems (WMS). This approach closely monitors the execution of workflow applications and the dependencies associated with each subtask. The scheduling of tasks is represented using a sequence-to-sequence neural network architecture. The neural architecture is then used to schedule decisions within the workflow. Reinforcement Learning is then applied to this architecture to improve it based on the rewards obtained for performing appropriate scheduling decisions. In this process, historic task execution data is collected, compared and a near-optimal scheduling decision corresponding to the highest reward is determined. In the sequence-to-sequence architecture used, scheduling decisions are responsible for mapping a set of inputs to a set of outputs. The mapping is performed by comparing the reward functions for the available cases.

C. *Unsupervised Learning Approaches*

Seyong Lee, Jeremy S. Meredith and Jeffrey S. Vetter [9] proposed a method to accurately predict the execution times using the source code of the workflow involved. In this method, the source code of an input program is analysed using an Open Accelerated Research Compiler. The compiler performs a static analysis of the program by analysing and digesting its source code. Aspen attributes are then added to it. The Aspen IR postprocessor then analyses it to check for errors or possible optimisations. Aspen prediction tools are then used to process the model created. This in turn generates an application performance model using Aspen. The COMPASS model basically creates a performance model of the input program along with annotations. These annotations can then be overridden by the user to tweak the decision-making process. Aspen is a language used for generating representations of applications. Aspen tools such as resource counters are then used to measure the resource usage of each block. The resource utilisation of each block is analysed and the number of iterations of each block is taken to determine the total resource requirement. A hardware sensitivity analyser is used to determine the hardware parameters and their effects on each block of the application. The performance model is then evaluated.

Ilia Pietri, Gideon Juve, Ewa Deelman and Rizos Sakellariou[10] proposed a prediction model capable of estimating the execution time for a given number of resources. This model accounts for the structure of the workflow and its runtime characteristics. A hierarchical clustering-based estimation model is implemented to evaluate the structure of the workflow. The model splits the workflow tasks into separate levels where the tasks at the same level are independent of each other. The execution time for each level is obtained by measuring the total execution time of the individual blocks in the workflow. The assignment of tasks to different workflows can be done using two approaches. One of the approaches is a Top-Down approach where the level of a task is calculated by considering the longest path from the entry node within the application. Another approach is the Bottom-Up approach where the level of task is based on the longest path the exit node of the application. The tasks of a workflow are assigned to levels by invoking level assignment method. The characteristics of the levels are found out. Then the maximum runtime, minimum runtime, maximum no of usable slots, etc are used to estimate the execution time for each level. Thus, the total execution time of a workflow can be predicted.

III. COMPARATIVE STUDY

Table 1 and Table 2 shows the comparative study of the above discussed papers based on their methodology, advantages and disadvantages.

Table – 1
Comparison Table

Reference paper	*Methodology*	*Advantage*	*Disadvantage*
[1]	*Uses two stage machine learning ensemble methods to generate predictions on execution times of workflow applications in grid environments.*	*Flexible and scalable. Evaluates and accounts for the differences in execution environments. Can be migrated with little effort.*	*Requires system state attributes and properties to be declared as parameters. Properties of workflow application required to determine dataflow dependencies cannot be guaranteed.*
[2]	*Identifies the best machine learning methods and predicts the utilization of resources by applications. The work extends Predicting Query Runtime (PQR) algorithm.*	*Performance improves by accounting system performance and application-specific attributes. Applicable to a wide range of applications and systems. PQR2 has better accuracy as it is able to adapt to scenarios with different characteristics.*	*Attribute information specific to each application is not readily available.*

[3]	*A Hybrid model which consists of analytic benchmarking techniques and statistical methods, predicts the execution time.*	*Can make accurate predictions using few past observations. The quality of the predictions improves with time. Information about design of the algorithms is not required by the approach.*	*Requires information about the sub tasks within the system. It also requires at least one entry corresponding to the historic data to function.*
[4]	*Execution time prediction is done by using historical data to train regression models which then help to forecast the execution time depending on the selected input features.*	*Does not require pre-requisites for functioning. Input features can be extracted inexpensively.*	*Process is data intensive. Requires human expertise. Large variation in prediction quality for each model.*
[5]	*Model extracts pre-runtime configurations and uses it to estimate resource consumption which is then used as a feature for prediction.*	*Accounts for longer execution time. Online Incremental Machine Learning is used.*	*Batch offline Evaluation though unfit for WaaS workflows, outperforms Online Incremental methods.*
[6]	*A framework to capture live data to build predictive models. The execution time is obtained by defining a relationship between the execution time of blocks and various block parameters.*	*The architecture is basic and can link with other systems. Variety of fallback prediction are also provided.*	*Model prediction degrades with increase in size of workflow.*
[7]	*A scheduling approach where a genetic algorithm is used to evolve solutions to workflow scheduling problems.*	*Allows the system to keep up with the constantly changing underlying environment. Accounts for peak times during execution and helps in scheduling accordingly.*	*Learning optimal value functions for less visited states is challenging. Performance is heavily dependent on gaining better experience.*
[8]	*The scheduling of tasks is represented using a sequence-to-sequence neural network architecture. Scheduling decisions are responsible for mapping a set of inputs to a set of outputs. The mapping is performed by comparing the reward functions for the available cases.*	*Can efficiently use historic data to schedule tasks. The scheduling decisions taken using Random Forest ensemble methods provide improved accuracy.*	*System does not have complete control of the tasks called for execution. Fails to account for duplicates of tasks that may occur within workflows.*
[9]	*Predicts the execution time by analysing the source code of an input program using an Open Accelerated Research Compiler.*	*Runtime predictions with error values close to zero in case of some applications. This approach is flexible and can be used on any workflow application.*	*The model requires source code of workflow applications for runtime prediction which is not always available.*
[10]	*The model splits the workflow tasks into separate levels where the tasks at the same level are independent of each other and the execution time for each level is obtained by measuring the total execution time of the individual blocks in the workflow.*	*Job submission delays are accounted.*	*Requires prior knowledge on the number of sub-tasks. Requires information about the physical hardware used by the cloud vendor.*

IV. Problem statement

Cloud computing or pay-per-use service is fast gaining prominence as they offer numerous advantages over their conventional counterparts. Cloud computing offers rapid provisioning of resources and provides effortless scalability eliminating the need for high-end hardware at the client side. Scientific workflows are increasingly being executed on cloud systems to improve the accessibility of resources as well as for economic reasons. Workflow applications generally consist of a large number of components or sub-tasks within them often making them resource-intensive and time-consuming. The sub-tasks within these applications are generally connected to data and control flow dependencies. Effective resource optimization and faster execution are aspects crucial to the execution of a workflow application. Previous attempts at predicting execution times, however, require prior knowledge about the workflow application and its source code as well as detailed information about the hardware used. This, however, cannot be guaranteed. Moreover, the change in execution environments involved also causes the results to be erroneous and unreliable.

V. Conclusion

To devise an efficient method to determine the execution time of workflow applications in the cloud and other resource-sharing environments, we researched a number of papers as part of our literature survey. The advantages and disadvantages offered by each methodology were identified. In the absence of a well-defined dataset, statistical prediction and analytic benchmarking were identified as the best way to build and organise a dataset. Sample datasets were created using the above methods in different execution environments. Different machine learning techniques were studied and compared, leading us to the conclusion that multi-stage machine learning techniques offered better accuracy than the other methods. This approach will be further used in our proposed methodology. The major drawbacks in previous research works were that they lacked adequate accuracy while some required parameters such as source code and internal structures as their prerequisites which cannot be guaranteed. Hence, the methodology we propose requires minimum parameters as requirements and generates an accurate prediction. Multi-stage machine learning ensembles such as MOE and DSE were found to improve accuracy considerably. Hence, these techniques involving multiple learners will be incorporated into the model. Thus, we would like to improve the accuracy and reliability of prediction of execution time in workflows.

References

[1] F. Nadeem, D. Alghazzawi, A. Mashat, K. Faqeeh and A. Almalaise, "Using Machine Learning Ensemble Methods to Predict Execution Time of e-Science Workflows in Heterogeneous Distributed Systems," in *IEEE* Access, vol. 7, pp. 25138-25149, 2019, doi: 10.1109/ACCESS.2019.2899985.

[2] A. Matsunaga and J. A. B. Fortes, "On the Use of Machine Learning to Predict the Time and Resources Consumed by Applications," *2010 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing*, Melbourne, VIC, Australia, 2010, pp. 495-504, doi: 10.1109/CCGRID.2010.98.

[3] M. A. Iverson, F. Ozguner and L. Potter, "Statistical prediction of task execution times through analytic benchmarking for scheduling in a heterogeneous environment," in *IEEE Transactions on Computers*, vol. 48, no. 12, pp. 1374-1379, Dec. 1999, doi: 10.1109/12.817403.

[4] T. Miu and P. Missier, "Predicting the Execution Time of Workflow Activities Based on Their Input Features," *2012 SC Companion: High Performance Computing, Networking Storage and Analysis*, Salt Lake City, UT, USA, 2012, pp. 64-72, doi: 10.1109/SC.Companion.2012.21.

[5] M. H. Hilman, M. A. Rodriguez and R. Buyya, "Task Runtime Prediction in Scientific Workflows Using an Online Incremental Learning Approach," *2018 IEEE/ACM 11th International Conference on Utility and Cloud Computing (UCC)*, Zurich, Switzerland, 2018, pp. 93-102, doi: 10.1109/UCC.2018.00018.

[6] Hugo Hiden, Simon Woodman, and Paul Watson. 2013. 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)]*. Association for Computing Machinery, New York, NY, USA, 77–87. DOI:https://doi.org/10.1145/2534248.2534256.

[7] E. Barrett, E. Howley and J. Duggan, "A Learning Architecture for Scheduling Workflow Applications in the Cloud," *2011 IEEE Ninth European Conference on Web Services*, Lugano, Switzerland, 2011, pp. 83-90, doi: 10.1109/ECOWS.2011.27.

[8] Athanassios M. Kintsakis, Fotis E. Psomopoulos, Pericles A. Mitkas, Reinforcement Learning based scheduling in a workflow management system, Engineering Applications of Artificial Intelligence, Volume 81, 2019, Pages 94-106, ISSN 0952-1976, https://doi.org/10.1016/j.engappai.2019.02.013. (https://www.sciencedirect.com/science/article/pii/S0952197619300351).

[9] S. Lee, J. S. Meredith, and J. S. Vetter, "Compass: A framework for automated performance modeling and prediction," in *Proceedings of the 29th ACM on International Conference on Supercomputing*, ser. ICS '15. New York, NY, USA: ACM, 2015, pp. 405–414. [Online]. Available: http://doi.acm.org/10.1145/2751205.2751220

[10] I. Pietri, G. Juve, E. Deelman and R. Sakellariou, "A Performance Model to Estimate Execution Time of Scientific Workflows on the Cloud," *2014 9th Workshop on Workflows in Support of Large-Scale Science*, New Orleans, LA, USA, 2014, pp. 11-19, doi: 10.1109/WORKS.2014.12.