American Scientific Research Journal for Engineering, Technology, and Sciences (ASRJETS)

ISSN (Print) 2313-4410, ISSN (Online) 2313-4402

© Global Society of Scientific Research and Researchers

http://asrjetsjournal.org/

Serverless Computing and Scheduling Tasks on Cloud: A Review

Omar Alqaryouti[a*], Nur Siyam[b]

[a,b]*The British University in Dubai, Dubai, United Arab Emirates*

[a]*Email: omar.alqaryouti@gmail.com*

[b]*Email: nur.siyam@gmail.com*

Abstract

Recently, the emergence of Function-as-a-Service (FaaS) has gained increasing attention by researchers. FaaS, also known as serverless computing, is a new concept in cloud computing that allows the services computation that triggers the code execution as a response for certain events. In this paper, we discuss various proposals related to scheduling tasks in clouds. These proposals are categorized according to their objective functions, namely minimizing execution time, minimizing execution cost, or multi objectives (time and cost). The dependency relationships between the tasks plays a vital role in determining the efficiency of the scheduling approach. This dependency may result in resources underutilization. FaaS is expected to have a significant impact on the process of scheduling tasks. This problem can be reduced by adopting a hybrid approach that combines both the benefit of FaaS and Infrastructure-as-a-Service (IaaS). Using FaaS, we can run the small tasks remotely and focus only on scheduling the large tasks. This helps in increasing the utilization of the resources because the small tasks will not be considered during the process of scheduling. An extension of the restricted time limit by cloud vendors will allow running the complete workflow using the serverless architecture, avoiding the scheduling problem.

Keywords: FaaS Clouds; Serverless Computing; Scheduling; Workflows; Cloud Computing; Scheduling Algorithms.

1. Introduction

Cloud computing is an evolving trend in the Information Technology (IT) and distributed computing. Clouds provide on-demand dynamic delivery of high quality and low-cost applications, infrastructure, and further IT resources as services through internet with payment per usage pricing approach [1]. Furthermore, cloud computing is a model that enables expanded access to IT resources with minimal management efforts.

----

* Corresponding author.

Customers can customize their cloud usage requirements in terms of storage, servers, applications, operating system, and development environment. Moreover, the customers can utilize the usage of the cloud services according to the defined service level agreement (SLA) that includes the desired Quality of Service (QoS) constrains [1]. With various cloud computing, individuals and enterprise organizations are able to store and process data, manage applications, and deploy applications with the support of virtualization resources [2,3].

Cloud computing is classified into three main environment types; private clouds, public clouds, and hybrid clouds. Private clouds exist within the same organization offering special benefits. Public clouds are usually located on a data center and available for public and are managed by vendors. On the other hand, hybrid clouds consist of a combination of private and public clouds [2]. In term of services, Cloud computing is divided into three main models; Infrastructure-as-a-Service (IaaS), Platform-as-a-Service (PaaS), and Software-as-a-Service (SaaS) [2]. The IaaS is considered as the base layer. In IaaS, the servers, storage, virtualization, hardware and networking are provided to customers through vendors. The customers will be able to manage their applications, databases, security and operating systems. While in PaaS, the vendors will additionally manage the databases, security and operating systems. SaaS is the model in which the customer's applications are hosted by the vendor. The vendors manage everything including the applications and make the services available for the customer. Figure **1** illustrates the three cloud computing models.

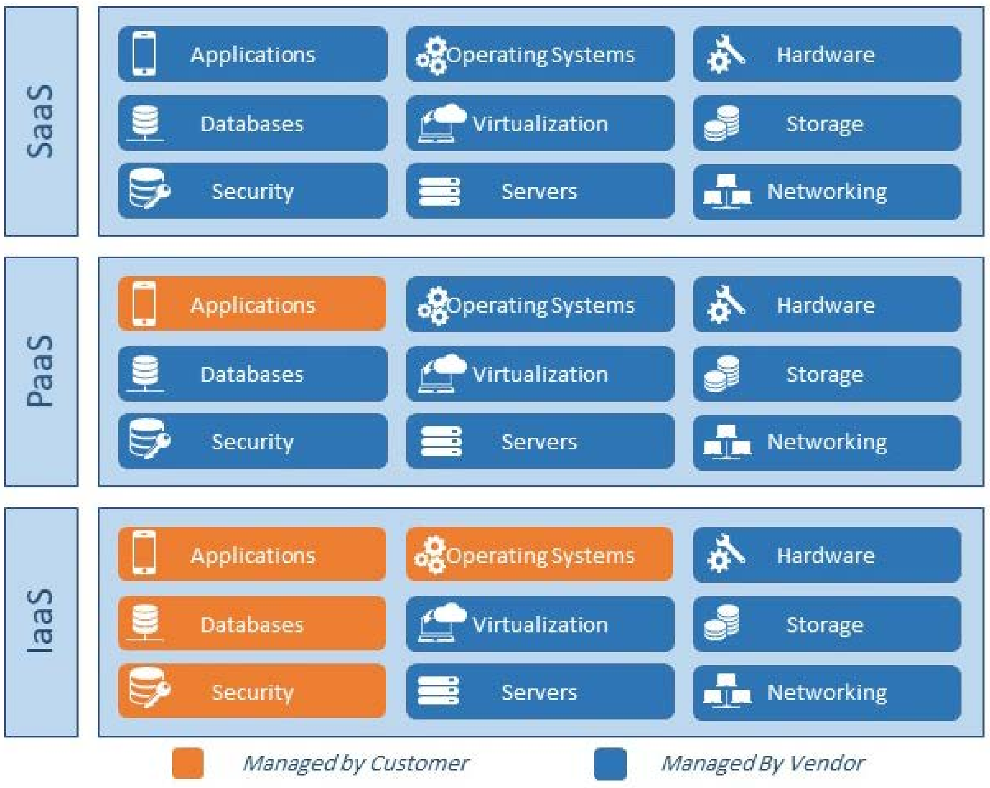

Figure 1: Cloud Computing Models

With the massive growth of data, scientists in different arenas of science, astronomy, engineering and physics are developing broad applications. These applications normally consist of dependent tasks that form workflows, such as the Montage and the Epigenomic workflows (*see Figure 2*). For instance, Montage workflows consist of several tasks that take as an input multiple images from the mosaics and use them to project a single image for a specific location. Such workflows are typically complex and extensive in nature, since they consist of large number of processes with extended number of tasks that have dependency constraints. This kind of complex applications require a high performance environment so that they can be performed within a given time constraint [4,6].

Applications workflows can be represented as a Directed Acyclic Graph (DAG), in which each computational task is denoted by a node, and the relation between the precedence-constrained tasks is denoted by directed edges [4,7]. In particular, the nodes (tasks) represent computations jobs while the edges (relation) define the dependencies between nodes. Tasks with dependency constraints cannot start before the parent tasks execution is completed. Workflows may have multiple entry tasks and one exit task. Entry tasks are those tasks with no parent tasks, whereas exit tasks are those tasks with no child tasks. *Figure 2* shows some example of well-known scientific workflows. Based on the size of the workflow, the execution of its tasks will be performed across several virtual machines. This leads to the following general optimization problem:

Given n number of tasks that forms a workflow, determine the "right" number of virtual machines to execute these tasks such that the execution time and/or cost is minimized.

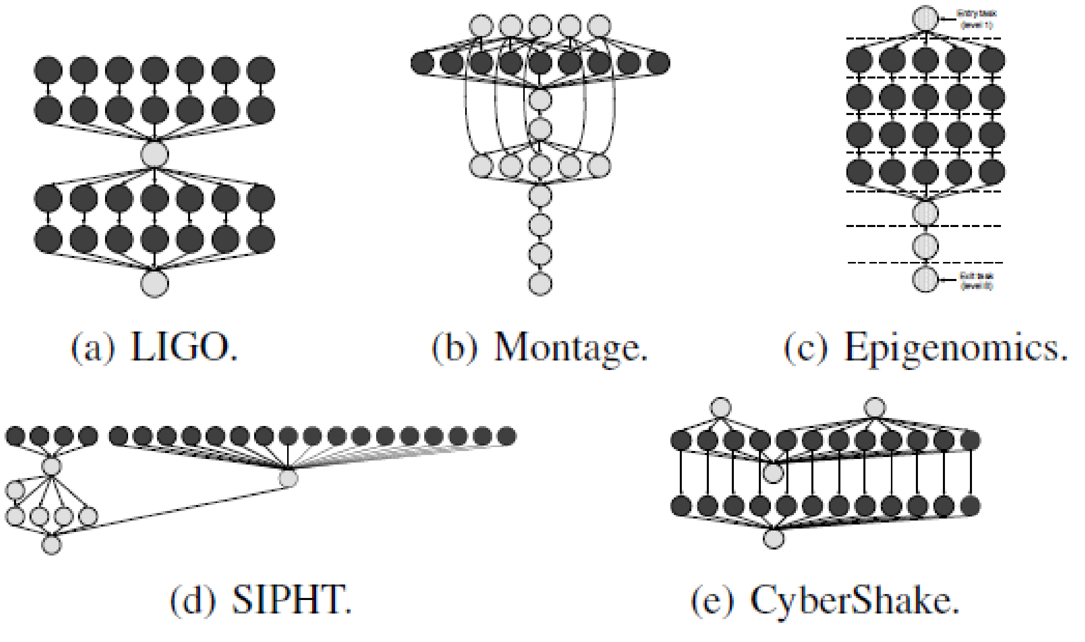

Figure 2: Examples of Scientific Workflows

To address this problem, researchers have studied the deployment of the workflows across various environments such as clusters and grids. This scheduling problem, and many of its variations are considered as NP-Complete problems [8]. Several variations of this problem have been investigated in the literature. For instance, in [1,9,10,11], the authors addressed the problem of finding the cheapest schedules while satisfying pre-determined deadline on the execution. Conversly, in [5,12,13,14,15,16] the authors proposed scheduling methods that aim to determine the fastest schedule, while satisfying a pre-determined budget constraint.

Recently, the use of Function-as-a-Service (FaaS) has emerged as a promising approach to simplify this scheduling problem. FaaS that is also known as serverless computing started to gain the attention of researchers in public clouds. FaaS is a new concept in cloud computing that allows the computation of services that triggers the code execution as a response for certain events. Additionally, FaaS is similar to PaaS in the essence that it allows software development and releases the users from the hassle of the management of servers and operating systems. However, the main difference is visible when it comes to the pricing scheme. In PaaS, the user will pay for the running time period of the thread. Whereas, in FaaS, the user will pay for the execution time of the event of a particular function [17]. FaaS is an enormously new expansion in cloud computing such as AWS Lambda by Amazon [18], Google Cloud Functions by Google [19] and Microsoft Azure by Microsoft [20].

Such paradigm has great impact on increasing the performance of large scale systems such as serverless websites, web applications, mobile backends, Cyber-Physical Systems (CPS), IoT backends, data processing, and IT automation. For instance, employing a serverless approach on websites where traffic differs significantly, reduces the cost of services when there is no load on the website while providing prompt scaling of resources when needed. Another example of large scale systems that benefit from a serverless architectures are the IoT backends. The increasing number of mobile backends connected to IoT platforms arises the need for a system that can flexibly scales. For example, Bustle.com (a news and fashion website), and Roomba (a cleaning robot) are both based on a serverless architecture on AWS Lambda by Amazon. The transition to a serverless approach allowed these companies to cut costs and enabled them to focus on improving their services rather than managing and maintaining the infrastructure. Additionally, FaaS is considered a beneficial approach for applications that require large amount of data processing in real time as well as parallel processing. Examples of such applications include the media company Thomson Reuters, where the serverless approach is used to analyze product data consumption more efficiently [21]. Similarly, the entertainment company Netflix uses AWS Lambda for different applications such as parallel processing of media data, backup updates, and security processes validation [22]. Moreover, the integration of serverless computing and CPS can help in connecting the cyber components of the system with the physical ones and facilitates their interaction. For example, sensors placed on physical farming machinery that are in need for a spare part, trigger the function on the serverless backend to purchase a new part automatically [18,23].

In this paper, we discuss various proposals related to scheduling scientific workflows in clouds. In addition, we clarify how the structure of this scheduling problem can be simplified using FaaS. The rest of the study is structured as follows. In section 2, we survey various approaches that addresses the workflow scheduling problem according to their objectives. Section 3 provides a discussion of the problem and how FaaS can help in increasing the utilization of the resources. Lastly, section 4 summarizes the key findings and discusses future work.

2. Scheduling Proposals

Challenges of scheduling workflows have been studied extensively in the literature. Typically, the focus can be either on the reduction of execution time, the reduction of execution cost, or the reduction of both execution time and cost. In this section, we discuss proposals that addressed this problem while aiming to emphasize the benefits of using FaaS in the process of scheduling. This section categorizes the proposals based on their objective functions into: (1) Minimizing execution time (2) Minimizing execution cost (3) bi-objective (time and cost).

2.1. Minimizing Execution Time

Many Proposals have addressed the problem of minimizing the execution time under the presence of budget constraints [5,12,13,15,16] Typically, the problem of minimizing the execution time is dominated by the critical path length. The length of this path establishes a lower bound on the execution time.

Wu and his colleagues [5] proposed an iterative approach that aim to iteratively re-allocate the tasks of the workflows to achieve the underlined goals. Likewise, Arabnejad and Barbosa [12] also proposed an iterative scheduling approach. In each iteration, the proposed approach tries to improve the current schedule based on the left budget. However, Sakellariou and his colleagues [15] proposed two heuristic solutions to schedule workflows. The objective of these solutions is to meet the budget constraint by iteratively modify the schedule of the workflow to adjust the relevant cost and minimize the total execution time of the workflow.

Topcuoglu, Hariri and Wu [16] proposed the well-known heterogeneous earliest finish time (HEFT) algorithm. The underline goal of this algorithm is to minimize the execution time of the workflow. It uses a greedy approach where in each stage it schedules the tasks with the highest priority. The allocation strategy in each stage aims to determine the resource which results in achieving the earliest execution time for the current task to be scheduled.

Lee and his colleagues [13] developed an efficient critical-path-first algorithm that stretches out the schedule in order to preserve the completion shortest probable time proactively. Additionally, the authors developed an algorithm to gain more efficient resources. The proposed algorithm compacts the schedule through re-ordering the tasks for maximum utilization of the schedule idle slots according to the dependency constraints. This approach confirmed that the efficient utilization of the abundant resources allows optimized performance in terms of resource usage and makespan.

The main challenge in designing time minimization approach is the budget constraint and the structure of the workflow. Budget constraints limit the number of resources that can be used and this bound the space of optimization. The structure of the workflow reflects the dependency relationships between the tasks which has a major influence on the expected performance of any scheduling approach. The dependency between the tasks restricts the order of execution which may result in delaying the execution for some of the tasks. Hence, a task cannot start execution before receiving all required data from its parent tasks. This problem can be simplified if we can reduce the impact of the dependency constraints of the parent tasks.

2.2. Minimizing Execution Cost

The problem of reducing the execution cost under the presence of the deadline constraints have been studied in the literature [1, 9, 10, 11]. Abrishami, Naghibzadeh and Epema [1] proposed the IC-PCP algorithm. In this algorithm, we are given a pre-determined execution deadline and the objective is to construct the cheapest schedule such that all tasks are executed before the given deadline. Starting from the exit task, this algorithm works by determining the latest possible time for each task to be executed that result in meeting the overall deadline. In the first iteration, the deadline will be assigned to the exit task. After that, the latest possible execution time for all tasks involved in the critical path will be determined. This is established through the latest possible execution time for each task's successor task. This process will be repeated to all paths routed at the critical path. The process of allocating resources in this algorithm aims to determine the cheapest resource usage that can execute these tasks on time.

Lee and Lian [9] studied how the workload can be dynamically transmitted among private and public clouds. For instance, many organizations may outsource their ICT to public clouds, while other organizations have their security and governance concerns considering infrequent workload cloud bursting. In this paper, the authors proposed a novel algorithm called Cloud Bursting Scheduler (CBS). The ultimate goal is to minimize the tasks and jobs running costs. It involves the energy, operating, and hardware costs. CBS considers the private clouds electricity rates that varies during the day and the variety of public clouds constant time rental rate. As a result, CBS reported significant cost saving when simulating workloads and improved performance of resource utilization.

Lee and his colleagues [10] proposed a novel approach that comprises resource management module for efficient cost utilization for both users and vendors. The resource management module integrated the pricing scheme according to the real usage, adoptive and fine-grained resource allocator. This approach reported saving cost from the user's perception, whereas operating clouds revenue is increased from the vendor's side.

Farahabady and his colleagues [11] demonstrated the Contention Aware Resource Allocation (CARA) system. The CARA aims to improve the efficiency of data centers. This system is designed according to the predictive control model that qualifies reasonable decision making along with upcoming system states. In CARA, the resource usage correlation among its form of shared and isolated models is essential for workload consolidation. This approach improved the resource utilization without a substantial impact on the QoS.

Typically, the cost involves the communication cost and the execution cost. The users will pay for the total renting time of the resources regardless whether these resources are used or not. It is expected that the resources will be idle for a significant amount of time due to the data dependency constrains between the tasks. By using FaaS, we can significantly improve the utilization of resources. Hence, the small tasks can be executed using this service and this will help in reducing the scheduling overhead.

In designing a cost-efficient approach, the main factor is the deadline constraints. The value of the deadline constraint establishes a restricted bound on the cost of the execution. Increasing the time deadline results in creating flexibility in terms of the number of resources that we can use which typically reduces the cost.

2.3. Approaches with multi Objectives

Many proposals have investigated the problem of optimizing the execution cost and time [6,14,24,25,26,27 28,29,30,31,32]. Malawski and his colleagues [24] proposed different scheduling methods that aim to maximize the total number of scheduled workflows while respecting the time and cost constraints. In this direction, Zheng and his colleagues [25] have investigated a similar problem. The authors have to manually decide whether the resulted schedule will be accepted or rejected.

An efficient Pareto-based method was proposed in [26]. This method provides the users with multiple schedules to choose from. By using this method, the users have to choose the best schedule that meets their objectives. The number of obtained solutions is an essential factor for achieving higher performance.

Lee, Subrata and Zomaya [14] described a unique algorithm of workflow applications scheduling in grids to minimize the application execution time and improve the resource usage. This algorithm is called Adaptive Dual Objective Scheduling (ADOS). The focus of this scheduling algorithm is on both time and use of resources as a dual objective methodology. The algorithm combines two core parameters. The first one is the static heuristic scheduling scheme. Whereas, the second one is the dynamic rescheduling technique. This approach has proved its significance in terms of resource utilization, execution time, and robustness to the changing resource performance.

Prodan and Wieczorek [27] proposed an algorithmic solution known as DCA, which requires determining the main and sub -objectives namely cost and time. DCA comprises two scheduling stages where the main objective is handled in the first stage of scheduling and the secondary objective is optimized by the second stage of scheduling.

Almi'Ani and his colleagues [6] proposed a cluster-based approach that uses a slack parameter which determines the priority of each objective function. The value of this parameter can be between 0 and 1. Increasing this value results in establishing clusters that focus on reducing the time execution. On the other side, reducing this value, establishes clusters that focus on reducing the execution cost. In this direction, the RDAS algorithm [32] employs strategies from fair allocation, to achieve the allocation schedule which address both objective functions.

Leslie and his colleagues [28] developed the RAMC-DC framework that utilized Amazon's various services to guarantee the QoS in terms of cost effectiveness, execution compliance and the reliability of the services. The framework integrates a variety of novel strategies such as cost, execution time, resource allocation and bidding. RAMC-DC utilizes Amazon EC2 service to provide efficient cost. Furthermore, it conveys performance and reliability of on-demand occurrences as well as minimized cost of instant occurrences.

Lu and his colleagues [29] developed a model of MapReduce scheduler that deals with various workload parameters known as Workload Characteristic Oriented Scheduler (WCO). The model followed dynamic scheduling decisions as well as a static analysis strategy for task selection to estimate the tasks workload characteristics. WCO was designed to achieve two main targets of resource usage efficiency and performance enhancement. The experiments results proved the workload diversity effectiveness of WCO.

Lee and Zomaya [30] presented a novel algorithm for task scheduling known as Artificial Immune System with Duplication (AISD). In the problem of scheduling tasks in heterogeneous has been examined. AISD algorithm starts with the generation and refinement of various schedules through the clonal selection approach within the immune system. Next, it improves the scheduling process by duplicating the tasks within the schedule. Finally, it removes the inefficient tasks from the schedule process. The proposed methodology reported promising results. In particular, when scheduling the intensive tasks that are represented in graphs. The performance evaluation was calculated by accurately coordinating between the three phases.Farahabady, Lee and Zomaya [31] developed a framework call PANDA. This framework is able to schedule static Bag-of-Tasks (BoT) throughout the private and public clouds resources. The BoT applications are enormously parallel and its operations are independent. PANDA integrates FPTAS scheduling algorithm. FPTAS creates schedules with the optimal trade-off in terms of cost and performance (Pareto-Optimality). The performance evaluation of this approach revealed the quality of PANDA in scheduling with optimal solution assurance to be within the boundaries of measurable distance.

The complexity of this problem is a result of its conflicted objectives. Reducing the cost normally results in increasing the execution time, whereas reducing the execution time may result in increasing the cost. The table below (*TABLE I*) summarizes the various scheduling proposals in the literature.

Table1: summary of scheduling proposals in the literature

Reference	**Paper Title**	**Year**	**Minimize Time**	**Minimize Cost**	**Time Constraint**	**Budget Constraint**
Wu and his colleagues [5]	“End-to-end delay minimization for scientific workflows in clouds under budget constraint”	2009	✓			✓
Arabnejad and Barbosa [12]	“A Budget Constrained Scheduling Algorithm for Workflow Applications”	2014	✓			✓
Lee and Zomaya [13]	“Stretch Out and Compact: Workflow Scheduling with Resource Abundance”	2013	✓			
Sakellariou and his colleagues [15]	“Scheduling workflows with budget constraints”	2007	✓			✓
Topcuoglu, Hariri, and Wu [16]	“Performance-effective and low-complexity task scheduling for heterogeneous computing”	2002	✓			
Abrishami, Naghibzadeh and Epema [1]	“Deadline-constrained workflow scheduling algorithms for Infrastructure as a Service Clouds”	2013		✓	✓	
Lee and Lian [9]	“Cloud Bursting Scheduler for Cost Efficiency”	2017		✓		

Lee and his colleagues [10]	“Fine-Grained, Adaptive Resource Sharing for Real Pay-Per-Use Pricing in Clouds”	2015		✓		
Farahabady and his colleagues [11]	“A model predictive controller for contention-aware resource allocation in virtualized data centers”	2016		✓		
Lee, Subrata and Zomaya [14]	“On the performance of a dual-objective optimization model for workflow applications on grid platforms”	2009	✓	✓		
Malawski and his colleagues [24]	“Cost optimization of execution of multi-level deadline-constrained scientific workflows on clouds”	2015	✓	✓	✓	
heng and Sakellariou [25]	“Budget-Deadline Constrained Workflow Planning for Admission Control”	2013	✓	✓	✓	✓
Durillo, Fard and Prodan [26]	“MOHEFT: A multi-objective list-based method for workflow scheduling”	2012	✓	✓		
Prodan and Wieczorek [27]	“Bi-Criteria Scheduling of Scientific Grid Workflows”	2010	✓	✓		
Almi’Ani and Lee [6]	“Partitioning-based workflow scheduling in clouds”	2016	✓	✓		
Leslie and his colleagues [28]	“Exploiting performance and cost diversity in the cloud”	2013	✓	✓		
Lu and his colleagues [29]	“Workload characteristic oriented scheduler for MapReduce”	2012	✓	✓		
Lee and Zomaya [30]	“An artificial immune system for heterogeneous multiprocessor scheduling with task duplication”	2007	✓	✓		
Farahabady, Lee, and Zomaya [31]	“Pareto-optimal cloud bursting”	2014	✓	✓		
Almi'Ani, Lee and Mans [32]	“Resource Demand Aware Scheduling for Workflows in Clouds”	2017	✓	✓		

3. Discussion

As discussed in the previous section, the dependency relationships between the tasks play an important role in determining the efficiency of the scheduling approach. This dependency may result in reducing the utilization of the resources. As shown in *Figure 3*, due to this relationship, we may end up with unused time slots in the virtual machines and the user will have to pay for the cost of those unused time slots. This problem can be reduced by adopting a hybrid approach that combines both the benefit of FaaS and IaaS [17]. FaaS has resulted in allowing software developers to leverage the serverless architecture to deploy parts of the application business logic, particular functions, or actions. These functions can be triggered within milliseconds. Such event driven serverless architecture enables users rapid and scalable developments and deployments. In this architecture, users only pay for the period of execution time.

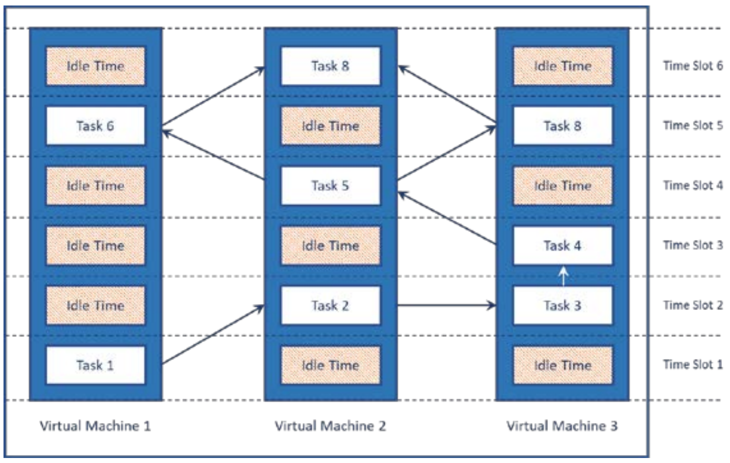

Figure 3: Idle Time Due to Dependency Relationship Between Tasks

FaaS is expected to have a significant impact on the process of scheduling scientific workflows. As we mentioned, the scheduling of the task will involve different virtual machines and due to the precedency constraints, some of these virtual machines may stay in idle state during the workflow execution time. Typically, the majority of tasks have less than 300 seconds running time [17]. For instance, AWS Lambda allows to run tasks up to 300 seconds while Google Cloud Functions allows up to 540 seconds as a limit for execution time [17]. Using FaaS, we can run the small tasks remotely and focus only on scheduling the large tasks. This helps in increasing the utilization of the resources. Since not considering the small tasks during the process of scheduling will reduce the total idle time in the resources (See Figure 4).

Combining FaaS and IaaS results in reducing the expected cost and running time for the workflow. In FaaS, users pay only for the amount of execution. Thus, in terms of cost, running the small tasks on FaaS instead of IaaS will not increase the execution cost. In contrary, it will reduce the execution cost since it helps in avoiding wasting the resources (reduce idle time). For the same factors, FaaS is expected to have a significant impact on reducing the execution time.

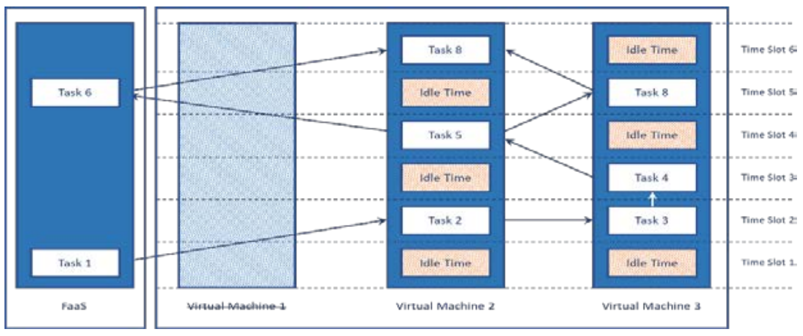

Figure 4: Adopting FaaS to Resolve the Scheduling Problem

4. Conclusion and Future Prospects

In this paper, we discussed and presented proposals that investigated the problem of scheduling scientific workflows on clouds. We mainly addressed three variations of this problem namely (1) minimizing the execution time (2) minimizing the execution cost (3) minimizing both the execution time and execution cost. We discussed the challenges of these problems and showed that by using FaaS and IaaS combined we can simplify the structure of this problem and therefore achieve better schedules in terms of cost and time. Due to the restriction on the running time in FaaS, an extension for the running time limit by the cloud vendors could allow the option of totally running the complete workflow using the serverless architecture and therefore, avoid this rescheduling problem.

References

[1] S. Abrishami, M. Naghibzadeh, and D. H. J. Epema, “Deadline-constrained workflow scheduling algorithms for Infrastructure as a Service Clouds,” Futur. Gener. Comput. Syst., vol. 29, no. 1, pp. 158–169, 2013.

[2] L. K. Arya and A. Verma, “Workflow scheduling algorithms in cloud environment - A survey,” 2014 Recent Adv. Eng. Comput. Sci. RAECS 2014, no. April, 2014.

[3] A. Bardsiri and S. Hashemi, “A Review of Workflow Scheduling in Cloud Computing Environment,” Int. J. Comput. Sci. Manag. Res., vol. 1, no. 3, pp. 348–351, 2012.

[4] M. A. Rodriguez and R. Buyya, “Deadline Based Resource Provisioning and Scheduling Algorithm for Scientific Workflows on Clouds,” IEEE Trans. Cloud Comput., vol. 2, no. 2, pp. 222–235, 2014.

[5] C. Q. Wu, X. Lin, D. Yu, W. Xu, and L. Li, “End-to-end delay minimization for scientific workflows in clouds under budget constraint,” IEEE Trans. Cloud Comput., vol. 3, no. 2, pp. 169–181, 2015.

[6] K. Almi’Ani and Y. C. Lee, “Partitioning-based workflow scheduling in clouds,” Proc. - Int. Conf. Adv. Inf. Netw. Appl. AINA, vol. 2016–May, pp. 645–652, 2016.

[7] T. Ryan and Y. C. Lee, “Effective Resource Multiplexing for Scientific Workflows,” 2015.

[8] M. Mao and M. Humphrey, “Auto-scaling to minimize cost and meet application deadlines in cloud workflows,” Proc. 2011 Int. Conf. High Perform. Comput. Networking, Storage Anal. - SC ’11, p. 1, 2011.

[9] Y. C. Lee and B. Lian, “Cloud Bursting Scheduler for Cost Efficiency,” 2017 IEEE 10th Int. Conf. Cloud Comput., pp. 774–777, 2017.

[10] Y. C. Lee, Y. Kim, H. Han, and S. Kang, “Fine-Grained, Adaptive Resource Sharing for Real Pay-Per-Use Pricing in Clouds,” Proc. - 2015 Int. Conf. Cloud Auton. Comput. ICCAC 2015, pp. 236–243, 2015.

[11] M. R. Hoseiny farahabady, Y. C. Lee, A. Y. Zomaya, Z. Tari, and A. Song, “A model predictive controller for contention-aware resource allocation in virtualized data centers,” Proc. - 2016 IEEE 24th Int. Symp. Model. Anal. Simul. Comput. Telecommun. Syst. MASCOTS 2016, no. 2, pp. 277–282, 2016.

[12] H. Arabnejad and J. G. Barbosa, “A Budget Constrained Scheduling Algorithm for Workflow Applications,” J. Grid Comput., vol. 12, no. 4, pp. 665–679, 2014.

[13] Y. C. Lee and A. Y. Zomaya, “Stretch Out and Compact: Workflow Scheduling with Resource Abundance,” Proc. - 13th IEEE/ACM Int. Symp. Clust. Cloud, Grid Comput. CCGrid 2013, pp. 219–226, 2013.

[14] Y. C. Lee, R. Subrata, and A. Y. Zomaya, “On the performance of a dual-objective optimization model for workflow applications on grid platforms,” IEEE Trans. Parallel Distrib. Syst., vol. 20, no. 9, pp. 1273–1284, 2009.

[15] R. Sakellariou, H. Zhao, E. Tsiakkouri, and M. Dikaiakos, “Scheduling workflows with budget constraints,” Integr. Res. GRID Comput., pp. 189–202, 2007.

[16] H. Topcuoglu, S. Hariri, and M. Wu, “Performance-effective and low-complexity task scheduling for heterogeneous computing,” Parallel Distrib. Syst. …, vol. 13, no. 3, pp. 260–274, 2002.

[17] Q. Jiang, Y. C. Lee, and A. Y. Zomaya, “Serverless Execution of Scientific Workflows,” 2017, pp. 706–721.

[18] Amazon, “AWS Lambda - Serverless Compute,” Amazon Web Services, Inc, 2014. [Online]. Available: https://aws.amazon.com/lambda/. [Accessed: 11-Nov-2017].

[19] Google, “Cloud Functions - Serverless Environment to Build and Connect Cloud Services | Google Cloud Platform,” Google Cloud Platform, 2016. [Online]. Available: https://cloud.google.com/functions/. [Accessed: 11-Nov-2017].

[20] Microsoft, “Microsoft Azure Cloud Computing Platform & Services,” Azure.microsoft.com, 2016. [Online]. Available: https://azure.microsoft.com/en-us/. [Accessed: 11-Nov-2017].

[21] A. AWS, “Optimizing Enterprise Economics with Serverless Architectures,” 2017.

[22] A. AWS, “Netflix & AWS Lambda Case Study,” 2014. [Online]. Available: https://aws.amazon.com/solutions/case-studies/netflix-and-aws-lambda/. [Accessed: 29-Nov-2017].

[23] D. Wu, D. W. Rosen, L. Wang, and D. Schaefer, "Cloud-based design and manufacturing: A new paradigm in digital manufacturing and design innovation," Comput. Aided Des., vol. 59, pp. 1–14, 2015.

[24] M. Malawski, K. Figiela, M. Bubak, E. Deelman, and J. Nabrzyski, "Cost optimization of execution of multi-level deadline-constrained scientific workflows on clouds," Sci. Program., vol. 2015, 2015.

[25] W. Zheng and R. Sakellariou, "Budget-Deadline Constrained Workflow Planning for Admission Control," J. Grid Comput., vol. 11, no. 4, pp. 633–651, 2013.

[26] J. J. Durillo, H. M. Fard, and R. Prodan, "MOHEFT: A multi-objective list-based method for workflow scheduling," CloudCom 2012 - Proc. 2012 4th IEEE Int. Conf. Cloud Comput. Technol. Sci., pp. 185–192, 2012.

[27] R. Prodan and M. Wieczorek, "Bi-Criteria Scheduling of Scientific Grid Workflows," J. Grid Comput., vol. 8, no. 4, pp. 493–510, 2010.

[28] L. M. Leslie, Y. C. Lee, P. Lu, and A. Y. Zomaya, "Exploiting performance and cost diversity in the cloud," IEEE Int. Conf. Cloud Comput. CLOUD, pp. 107–114, 2013.

[29] P. Lu, Y. C. Lee, C. Wang, B. B. Zhou, J. Chen, and A. Y. Zomaya, "Workload characteristic oriented scheduler for MapReduce," Proc. Int. Conf. Parallel Distrib. Syst. - ICPADS, pp. 156–163, 2012.

[30] Y. C. Lee and A. Y. Zomaya, "An artificial immune system for heterogeneous multiprocessor scheduling with task duplication," Proc. - 21st Int. Parallel Distrib. Process. Symp. IPDPS 2007; Abstr. CD-ROM, 2007.

[31] M. R. Hoseiny Farahabady, Y. C. Lee, and A. Y. Zomaya, "Pareto-optimal cloud bursting," IEEE Trans. Parallel Distrib. Syst., vol. 25, no. 10, pp. 2670–2682, 2014.

[32] K. Almi'Ani, Y. C. Lee, and B. Mans, "Resource Demand Aware Scheduling for Workflows in Clouds," in The 16th IEEE International Symposium on Network Computing and Applications (NCA 2017), 2017, p. to appear.