Workflow Scheduling and Offloading for Service-based Applications in Hybrid Fog-Cloud Computing

Saleh M. Altowaijri
Department of Information Systems, Faculty of Computing and Information Technology
Northern Border University, Rafha 91911, Kingdom of Saudi Arabia

***Abstract*—Fog and edge computing has emerged as an important paradigm to address many challenges related to time-sensitive and real-time applications, high network loads, user privacy, security, and others. While these developments offer huge potential, many efforts are needed to study and design applications and systems for these emerging computing paradigms. This paper provides a detailed study of workflow scheduling and offloading of service-based applications. We develop different models of cloud, fog and edge systems and study the scheduling of workflows (such as scientific and machine learning workflows) using a range of system sizes and application intensities. Firstly, we develop several Markov models of cloud, fog, and edge systems and compute the steady-state probabilities for system utilization and stability. Secondly, using steady-state probabilities, we define a range of system metrics to study the performance of workflow scheduling and offloading including, network load, response delay, energy consumption, and energy costs. An extensive investigation of application intensities and cloud, fog, and edge system sizes reveals that significant benefits can be accrued from the use of fog and edge computing in terms of low network loads, response times, energy consumption and costs.**

***Keywords*—Workflow scheduling; workflow offloading; cloud computing; fog computing; edge computing; scientific workflows**

I. Introduction

In recent years, new computing paradigms, named fog computing [1]–[5] and edge computing [6]–[11] have emerged as an extension of cloud architecture to the edge of the network to support the computational demands of real-time, latency-sensitive, and location-aware service-based applications (SBA) of largely distributed Internet-of-Things (IoT) devices/sensors. Fog and edge computing are considered among the most important archetypes in the current world. While some communities differ in the precise definitions of and differences between fog and edge computing or nodes, we prefer the following definitions. Edge computing refers to the computing at the edge of the networks, near the device and IoT layers. Fog computing refers to the computing at the intermediate layers between cloud data centers and IoT devices (many works have considered such definitions, see [3], for example). Edge and Fog layers have been proposed to bridge the gap between the cloud and IoT devices by enabling data management, computing, networking, storage, and application services at the intermediate layers and edge of the network while offering the possibility to interact with the cloud. Many applications have been proposed to benefit from fog and edge computing such as smart districts [9], SMS (Short Message Service) spam detection [12], networked healthcare [3], smart societies [2], QoS management in networks [8], Smart airport [9], Distributed Artificial Intelligence (AI) as-a-service (DAIaaS) [9], and many others applications [13]–[18]. However, the development and management of fog-based and edge-based systems for SBA face many challenges that need to be tackled. These include investigating and designing applications and systems for these emerging computing paradigms. One of the core challenging issues is workflow scheduling and offloading in such a dynamic, geo-distributed, heterogeneous environment where the set of computing nodes contains edge nodes, fog nodes, and cloud datacenters such as discussed in many works in the literature [19]–[23].

Further research is needed, for instance, for investigating dynamic scheduling of multiple workflows executions, i.e., invocations of multiple sets of linked elementary IT-enabled services, in hybrid edge-fog-cloud computing environments while ensuring the individual Quality-of-Service (QoS) requirements of all the workflows and their services and reducing services latency, energy consumption, and costs.

This paper provides a detailed study of workflow scheduling and offloading of service-based applications. We abstracted high-level challenges and requirements of cloud, fog and edge systems and developed different models of cloud, fog and edge systems, and study the scheduling of workflows (such as scientific and machine learning workflows) using a range of system sizes and applications intensities. Firstly, we develop several Markov models of cloud, fog, and edge systems and compute the steady-state probabilities for system utilization and stability. Secondly, using steady-state probabilities, we define a range of system metrics to study the performance of workflow scheduling and offloading including, network load, response delay, energy consumption, and energy costs. An extensive investigation of application intensities and cloud, fog, and edge system sizes reveals that significant benefits can be accrued from the use of fog and edge computing in terms of low network loads, response times, energy consumption, and costs.

The proposed workflow scheduling and offloading models can be utilized in practice to study a range of applications and derive several benefits. Firstly, different well-known standardized workflow can be plugged in our proposed workflow scheduling models to study their various performance behaviors including network load, average response delay, energy consumption, and energy cost, and this can be done for a range of cloud-only, cloud-fog, and cloud-fog-edge systems. Some examples of standardized workflows include Montage workflow, SIPHIT workflow, epigenomics workflow, LIGO workflow, Cyber-Shake workflow, and more; see [23], for explanations and use cases of these workflows, and [19]–[22] for additional examples for practical utilization of our work. We elaborate this further in the methodology, results, and the discussion sections.

The rest of the paper is organized as follows. Section II reviews related work. Section III details the methodology and design. Section IV provides an analysis of the results. Section V discusses the practical utilization of the proposed models and concludes the paper. Section VI provides future research directions.

II. Literature Review

The focus of this paper is on combining the use of edge, fog and cloud computing for executing service-based applications. Fog computing has been attracting a lot of attentions in the last few years from researchers all over the world to bring out its potential. It has been seen as a complement of cloud computing to allow satisfying the increasingly sophisticated applications demanded by users that combine the use of time-sensitive services and intensive-processing services, such as big data analysis which can be performed only in the cloud. The coupling of fog and cloud computing requires providing scheduling and offloading mechanisms that allows to manage the execution of multiple workflows of interconnected services in fog and cloud resources.

The problem of scheduling and offloading in fog computing environments has been an active research topic for the last few years. Many researchers have been extensively focusing on providing solutions to this problem [24]–[32]. However, the research on fog computing, in general, and on workflow/task scheduling and offloading, in particular, is still in its early stages and the problem is not completely solved and there is still a lot of challenges that need to be addressed [33].

In [34], Zeng et al. tackled the problem of minimizing the maximum task completion time in Fog computing supported software-defined embedded system (FC-SDES) by jointly considering task scheduling and task image placement. The authors considered a scenario where tasks (requests) can be processed either on the client node or a fog (edge) node and task images can be saved on storage servers. Based on that scenario, they formulated their optimization problem as a mixed-integer non-linear programming (MINLP) problem. Then, in order to tackle its computational complexity, the authors proposed a heuristic algorithm for task completion time minimization based on the concept of "partition and join". The main consideration in the proposed algorithm is that, by balancing the load between client nodes and fog nodes, the overall computation and transmission latency of all requests, therefore, their completion time, can be significantly minimized.

Similarly, Chen et al. [35], formulated the task offloading problem in ultra-dense network as a mixed-integer non-linear programming problem. Their aim, in this work, was to minimize the delay while saving the battery lifetime of user's device. To do so, the optimization problem has been divided into two sub-problems, i.e., task placement and resource allocation sub-problems. Based on the solution of the two sub-problems, the authors proposed an offloading scheme which considers the battery lifetime of user's device while reducing the task duration.

In [36], Huang et al. focused on providing a solution to the problem of computational offloading for multimedia workflows in mobile cloud computing. They proposed an energy-efficient offloading method using Differential Evolution (DE) algorithm to optimize the energy consumption of the mobile devices with time constraints.

Targeting the problem of task scheduling in smart factory, Wan et al. [37], introduced a method for energy-aware load balancing and scheduling (ELBS) based on Fog computing. They first formulated a load balancing optimization function by taking into account the energy consumption of the equipment in the smart factory. Then, they introduced a multi-agent system for achieve the dynamic scheduling of equipment workload with the task scheduling mechanism.

Considering a scenario where both edge/fog and cloud computing are used to serve mobile users, Zhao et al. [31], proposed to maximize the probability of tasks satisfying the delay requirement by jointly scheduling them either to the edge/fog network or to the cloud and allocating computational resources in the edge/fog network. So, they proposed to offload tasks with stringent delay bounds to resources in the edge level while the ones with loose delay bounds to resources in the cloud level. The proposed solution has been introduced to allow users with different delay requirement to be simultaneously served.

In [38], the authors presented a ranking-based method for task scheduling in fog-cloud computing networks. The aim of the proposal is to schedule user's requests based on their different preferences and fog nodes' constraints. To do so, the authors proposed to use linguistic quantifiers and fuzzy quantified propositions to rank fog nodes from the most to the least satisfactory one based on their requirements, then, the one that satisfies more user task preferences will be selected as a destination fog node.

Another work for task scheduling in hybrid fog-cloud computing has been proposed in [39]. In their work, Aburukba et al. modeled the problem of scheduling IoT service requests as an optimization problem using integer linear programming to minimize latency. They proposed a heuristic optimization approach in order to find feasible solutions with a good quality in a reasonable computational time. The genetic algorithm (GA) has been chosen and customized to schedule the IoT service requests to achieve the objective of minimizing the overall latency.

Even though there is many research works for scheduling and offloading in fog computing and hybrid fog-cloud computing, most of them fail to meet the scalability and mobility of nodes criterion. Also, they consider the scheduling of a single task which is not applicable for workflows composed of a set of linked tasks. The dependency between tasks in workflows adds more challenge to the scheduling and offloading problem. More importantly, many efforts are needed to develop high-level understanding of cloud-fog-edge systems.

III. Methodology and Design

A. Workflow Scheduling Cloud-Fog-Edge Model

Consider a workflow scheduling system that is programmed to manage its capacity periodically -- such as monthly, weekly, daily, or hourly -- by scrutinizing the quantitative variations in the workload. A possible method that can be used for planning is building Markov models and solving these models for their steady-state probabilities.

Let us represent the demand of a workflow scheduling system in terms of the aggregate computational nodes by λ_A, where the subscript 'A' represents the 'aggregate' demand per hour. The demand λ represents the inter-arrival times that are exponentially distributed. The aggregate demand includes the demand for cloud, fog, and edge servers that are represented by λ_C, λ_F, and λ_E, respectively. We can write the aggregate demand in a mathematical form as in the following equation.

$$\lambda_A = \lambda_C + \lambda_F + \lambda_E. \quad (1)$$

Now consider that the aggregate hourly capacity of the collective cloud, fog, and edge system is μ_A, where, as for the arrival rate λ, the subscript 'A' represents the 'aggregate' hourly capacity in terms of the number of nodes. These nodes can be the physical nodes in the system or virtual machines. Similar to Equation (1), the following equation gives the breakdown of the aggregate capacity, which is the sum of the cloud, fog, and edge capacities, respectively.

$$\mu_A = \mu_C + \mu_F + \mu_E. \quad (2)$$

The total number of physical or virtual nodes in the system are represented by N_A, which is the sum of the total number of cloud, fog, and edge nodes, represented by N_C, N_F, and N_E, as formulated in Equation (3) below. Note that the capacities defined in Equation (2) are the hourly service capacities of the system while Equation (3) defines the number of cloud, fog, and edge nodes in the system.

$$N_A = N_C + N_F + N_E. \quad (3)$$

Note that "capacity" in this paper implies to be the server capacity in terms of the physical nodes or virtual machines the cloud, fog, and edge are able to provide in terms of their hourly rates. Note also that the three arrival rates or demands and capacities in the two equations given above can assume any reasonable values and their quantities do not affect our model. Indeed, not only that the model is independent of the values λ and μ can assume, the number of sub-arrival rates and capacities can also be extended to any number of clouds, fogs and edges. That is, using the model described above, we can model any number of clouds, fogs and edges by embedding in Equations (1) and (2) their individual arrival rates and server node capacities.

Fig. 1 depicts the CTMC (Continuous Time Markov Chain) transition diagram of our proposed cloud-fog-edge workflow scheduling model. There are three parts of the transition diagram, one part each for cloud, fog, and edge. The symbols used in the figure have already been defined in the earlier paragraphs and equations. The cloud layer model is depicted in the top row, as is evident by the use of "C" subscript in all the variables and parameters (arrival and departure rates, and node capacities), followed by the fog layer (use of the subscript "F") and edge layer (use of the subscript "E") in the second and third rows, respectively.

Inside the cloud layer, we have the system moving from the zero or idle state with no task in the system to be executed to one task, two tasks, until "c" tasks, where "c" could be any state between zero and N_C. Once the maximum number of tasks allocated to the cloud(s) have reached, the fog can start receiving tasks, moving from state one, to two, to "f" where "f" could be any state between one and N_F. Once the maximum number of tasks allocated to the fog(s) have reached, the edge can start receiving tasks, moving from state one, to two, to "e" where "e" could be any state between one and N_E. Note that any reasonable values can be assigned to the quantities in Equations (1), (2), and (3), with the exception that arrival rate cannot exceed the capacity otherwise the system will not be stable. Note also that the system can be equally modelled as first receiving the tasks in the edge layer followed by the fog and cloud. Similarly, we can conveniently place another set of parameters -- let us call them n_C, n_f, and n_e, to replace N_C, N_F, N_E, respectively – such that they can be any number between zero and N_x $(x \in \{C, F, E\}$. This would enable the model to allocate any maximum number of nodes in cloud, fog, and edge layers up to the maximum capacities of the three layers. That is, we can model such that any or all of the three layers do not have to work to their full capacities to avoid instability and provide higher reliability. Finally, note that the arrival and departure rates are dependent on the specific state the system is in but in the figure (Fig. 1) these are shown the same for simplicity (e.g., λ_C and μ_C).

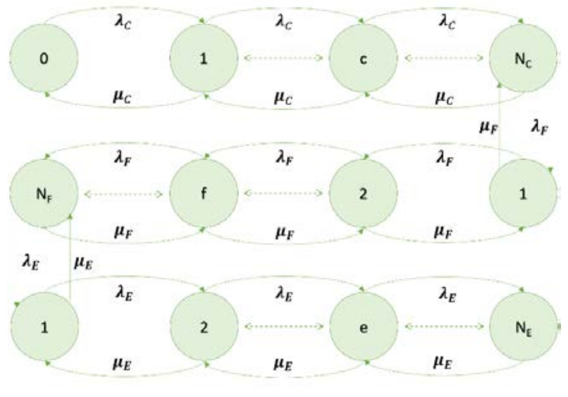

Fig. 1. The Cloud-Fog-Edge Workflow Scheduling Model – CTMC Transition Diagram.

$$\begin{pmatrix}
-\lambda_C & \lambda_C & 0 & \cdots & \cdots & \cdots & \cdots & \cdots \\
\mu & -(\mu_C + \lambda_C) & \lambda_C & 0 & \cdots & \cdots & \cdots & \cdots \\
0 & 2\mu_C & -(2\mu_C + \lambda_C) & \lambda_C & 0 & \cdots & \cdots & \cdots \\
0 & 0 & 3\mu_C & -(3\mu_C + \lambda_C) & \lambda_C & 0 & \cdots & \cdots \\
\vdots & \vdots & \cdots & \ddots & \cdots & \cdots & \cdots & \cdots \\
0 & 0 & 0 & N_C\mu_C & -(N_C\mu_C + \lambda_F) & \lambda_F & \cdots & \cdots \\
\vdots & \vdots & \cdots & \cdots & \ddots & \cdots & \cdots & \cdots \\
0 & 0 & 0 & 0 & N_F\mu_F & -(N_F\mu_F + \lambda_E) & \lambda_E & \cdots \\
\vdots & \vdots & \vdots & \vdots & \cdots & \cdots & \ddots & \cdots \\
0 & 0 & \cdots & \cdots & \cdots & 0 & N_E\mu_E & -(N_E\mu_E)
\end{pmatrix}$$

Fig. 2. The Cloud-Fog-Edge Workflow Scheduling Model: CTMC Generator Matrix Q.

Fig. 2 depicts the CTMC generator matrix of our proposed cloud-fog-edge workflow scheduling model. The model itself is depicted in Fig. 1 and has been explained. This generator matrix, represented by A, can be used to calculate the steady-state probabilities vector, x, using the following equation.

$$Ax = 0. \tag{4}$$

The vector x is a probability vector and therefore the sum of the vector is normalised to the value of 1. The details about the numerical solution of Markov chains can be found in the nominal book [40], or our earlier works such as [41], and other works such as [42].

We define a range of cloud, fog and edge system sizes in terms of the computational nodes and task arrival and departure rates and solve them for the steady-state probabilities. We use these steady-state probabilities to compute the system utilization and stability as formulated in Equations (5) and (6). The system utilisation (U) is defined as the maximum relative state that has the highest probability for the system to be in, in terms of the number of computational nodes.

$$U = \frac{i}{N} \mid x[i] = \max(x). \tag{5}$$

The relative state in the equation above is computed by dividing the state number that has the highest probability in the vector (indicating that the system will be in this state with the highest probability) by the total capacity of the system (the hourly capacity in terms of the number of nodes); higher this number, higher will be the utilisation of the system in terms of the number of busy computational nodes. A higher utilisation is desired to make the best use of the available computational resources. However, a higher utilisation could reduce the stability of the system.

The system stability defines inverse of the maximum relative state that has the highest probability and can be computed by dividing 1 by the system utilization.

$$S = \frac{1}{U} \tag{6}$$

The average network workload in bytes per second is represented by NWL and its computation is formulated by the following equation.

$$\text{NWL} = \frac{\psi * \sum_{i=1}^{\kappa} (\eta_i * t_i)}{\kappa}, \qquad \eta_i \in \eta, t_i \in \{\kappa\}. \tag{7}$$

In the equation above, η_i is the network load of the task number i, given in bytes, t_i is the time the task number i takes to be transferred over the network, and η is the set of all tasks in a given state. There are κ tasks in a given state, and $\{\kappa\}$ is the set of all κ time durations required to complete all tasks. In our experiments, we have used a fixed size of 5MB for the network load for all tasks. The times for each task are also fixed according to the latency of different networks (cloud-fog latency is 100ms, fog-edge latency is 2ms, and edge latency is 0.1ms). Not that this is not a limitation of our approach; a distribution of task sizes and task network transfer time can easily be used in these equations. Finally, ψ is a factor that depends on the state the system is in and this is computed using the following equation.

$$\psi = 1 + \omega * U^{\sigma}, \quad \forall \psi, \psi \leq 1 + \omega. \tag{8}$$

The equation stats that ψ can be computed using system utilization U, ω, and σ, but its value cannot exceed $1 + \omega$, means that the value of U^{σ} cannot exceed 1. The parameter ω is a regulation weight given to the network workload calculation that regulates the factor ψ. We set it to 1.0. A lower value for this parameter will have a lower effect on the network load computation and vice versa. This can be set by the user based on their knowledge of the system or focus of the study. The parameter σ is set to 5 in our calculations. It is a dampening factor over utilization so that the effect of utilization is balanced over the various operational states of the workflow scheduling system. A higher value of the dampening fact σ will create higher dampening implying that the values of ψ will increase slowly for the earlier states towards the higher-numbered states (see Fig. 1 and Fig. 2).

The average response delay (RD) of the system can be computed by the following equation. The parameter ψ is the same factor that depends on the state the system is in and this was computed using Equation (8). The variable d_i is the delay of job "i" (the time it takes to complete the job) and there are τ jobs in the system, each with its own delay.

$$\text{RD} = \frac{\psi * \sum_{i=1}^{\tau} d_i}{\tau}, \qquad d_i \in \{\tau\}. \tag{9}$$

The network energy consumption per hour (NE) is calculated by the following equation. The network energy consumption depends on the network load, NWL, and the estimated energy ζ. Several studies have reported the network energy consumption. We use the values reported in [9], [43], which is 0.54 kWh/GB. NWL has already been computed earlier. Since it was computed in MBps, we have added in the equation a denominator of 1000 to convert the network load into GBps. The value is multiplied by Ƭ which is the time factor, in this case it is 3600 (the number of seconds in an hour).

$$\text{NE} = \frac{Ƭ * \zeta * NWL}{1000}. \tag{10}$$

Finally, we compute the network energy cost (EC) per hour as given in the following equation. The cost depends on the network energy consumption, NE, calculated earlier, and the unit price of energy (γ), which we have taken from [44] as GBP 0.174 per kilowatt-hour (kWh).

$$\text{EC} = \gamma * NE. \tag{11}$$

IV. Results and Analysis

A. *Workflow Scheduling System (Cloud)*

We use the CTMC model described in Fig. 1 and Fig. 2 and model a cloud-only workflow scheduling system. We set the number of cloud nodes to 16,000, and the number of fog and edge nodes to zero each. This gives according to Equation (3), $\boldsymbol{N_A} = 16{,}000 + 0 + 0 = 16{,}000$. We study different CTMC system with varying aggregate arrival rates, λ_A, beginning from one task per hour ($\lambda_A = 1$) to 500 tasks per hour ($\lambda_A = 500$), up to 16,000 tasks per hour ($\lambda_A = 16{,}000$). The service rate or departure rate or hourly capacity of this system is kept at constant, which is 16,000 tasks per hour

($\mu_A = 16{,}000$). These settings result in 33 different CTMC models that we solve using iterative methods and compute the steady-state probabilities for each model. These are plotted in Fig. 3. The figures shows that the x-axis that plots the number of states varies between 0 and 16,000, and the y-axis provides probability values ranging from 1E-19 to 1.0 on a logarithmic base 10 scale. The system is idle in the zeroth state. Initially, with lower arrival rates, the probabilities of the lower-numbered states are higher compared to the higher-numbered states, and in these cases the maximum probabilities fall to a certain low, near-zero values (manifested in the vertically dropping lines before the state number 16,000). As we move towards higher arrival rates, the probabilities for the lower-numbered states start decreasing and the probabilities for the higher-numbered states start increasing. This trend continues until the probabilities rise towards the high-numbered states and do not fall vertically even after reaching the states nearer the state number 16,000. This shows that these cloud scheduling systems will be operating with high computational node utilization but with low stability and the risk for the system to drop the tasks off the system or its waiting queues.

Fig. 4 plots the utilization and stability of the 33 cloud workflow systems that we have described in the previous paragraph. Note that the utilization (y-axis, blue line) rises with the increasing arrival rates (x-axis), while the stability of the system (orange line) decreases with the increasing arrival rates. This is an expected behaviour from such systems.

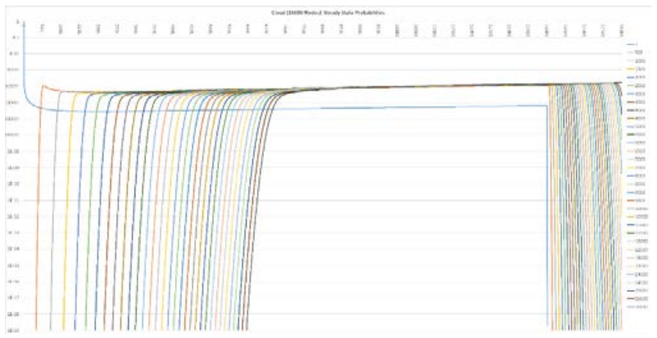

Fig. 3. Steady State Probabilities: Cloud Only System (16000 Nodes).

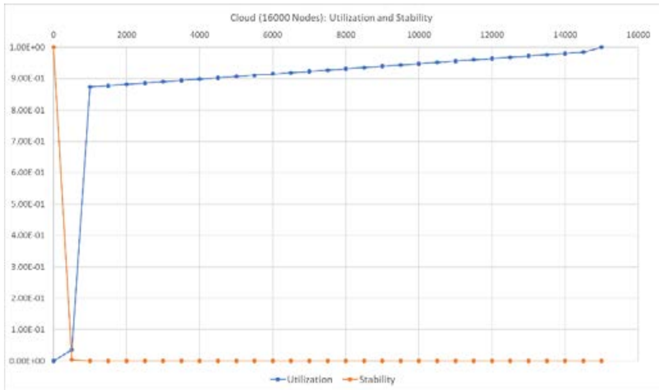

Fig. 4. Cloud (16000 Nodes): Utilization and Stability.

B. *Workflow Scheduling System (Cloud-Fog-Edge)*

We now model cloud-fog-edge systems where the workflows are scheduled to all three layers. Fig. 5 plots the steady-state probabilities of 23 workflow scheduling systems with different configurations. Some execute on cloud only (the top nine systems represented by Cloud-1, Cloud-1000, … Cloud-8000). While others run on fog (Fog-333 to Fog-333) and edge (Edge-166 to Edg-3166) layers. The numbers alongside Cloud-, Fog-, and Edge- represent the arrival rates for those nodes types. The strange numbering is used to avoid lines coming on top of each other and causing difficulty in reading and differentiating the plots. Note that since the fog and edge layers add to the capacity of the cloud, the probabilities for the higher-numbered states near the state number 16,000 are zero, indicating that those systems will not be unstable.

C. *Network Load*

The network workload (NWL) computations were explained earlier in Section III along with its Equation (7). In this section, we will study the network workload related performance of various cloud, cloud-fog, and cloud-fog-edge systems.

Fig. 6 depicts the network workload in GBps for a cloud-only workflow scheduling system containing 16,000 nodes. There are a total of 31 different systems that have been modelled and their network load has been computed. These 31 systems relate to different workloads on the systems in terms of the tasks being received by the system, beginning from 1 task, to 500 tasks, up to 15000 tasks per hour. The capacity in all of these 31 systems has been kept constant. The minimum network load is for the system with one task; it is actually 0.51 GBps but is rounded off to 1GBps in the figure. Note that the network load consistently rises to reach 15,315 GBps for the busiest workflow scheduling system. Note that the increase in the network load is due to the equal load of each job as have been explained in Section III. However, this increase can be varied by using a distribution of network loads related to different tasks, and these network loads and tasks can even be varied based on different system states. Moreover, note that the increase in the network load is not linear. This is due to the factor ψ, which is dependent on the steady-state probabilities and system utilisation.

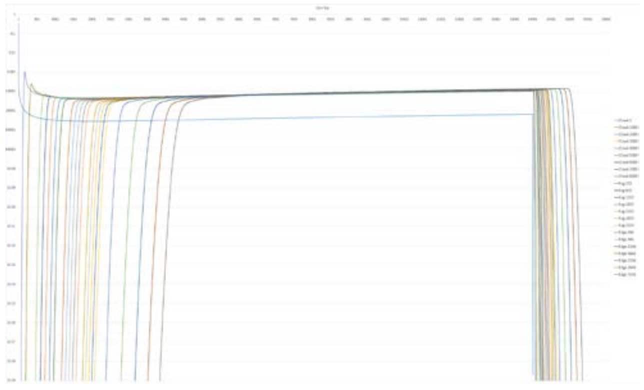

Fig. 5. Steady-State Probabilities: Cloud-Fog-Edge System (8000-4000-4000 Nodes).

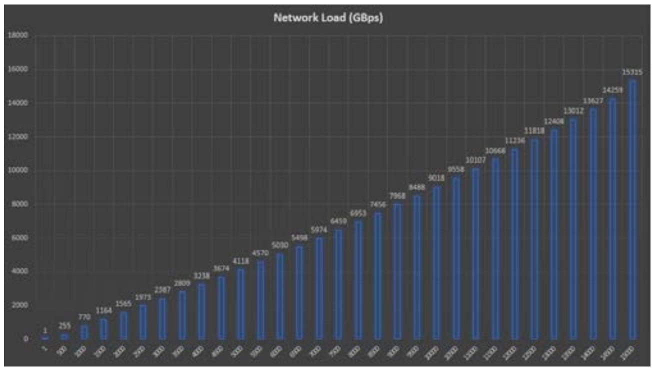

Fig. 6. Network Load in GBps (Cloud with 15,000 Nodes).

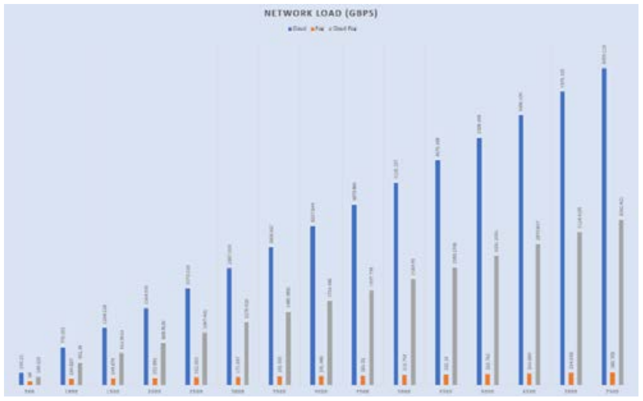

Fig. 7. Network Load in GBps (Cloud-Fog System with 7,500 Nodes Each).

Fig. 7 shows the network workload in GBps for a cloud-fog workflow scheduling system containing 7,500 nodes each in the cloud and fog. As mentioned earlier, an equal number of nodes are used for simplicity of explanation and it does not pose any limitations on the design of the systems. There are a total of 15 different systems for cloud and fog each that have been modelled and their network loads have been computed. These 15 x 2 systems relate to different workloads on the systems in terms of the tasks being received by the system, beginning from 500 tasks, up to 7,500 tasks. The capacity in all of these 30 systems has been kept constant. The cloud network load is depicted using blue bars and the fog network load is depicted using orange bars. The third set of bars in the grey colour represents the aggregate network load of the cloud-fog systems. The minimum network load is for the system with 500 tasks; it is 255 GBps for cloud and 84 GBps for the fog system, with ~170 GBps the aggregate network load. The network load consistently rises to reach 6,459 GBps (cloud), ~266 (fog), and 3362 GBps (aggregate) for the busiest workflow scheduling system. Note that the consistent increase in the network load is due to the equal load of each job as has been explained earlier. The relatively smaller values for fog systems is due to the smaller network latencies, implying a much lower time period for the fog tasks to travel over the fog-edge networks compared to the fog-cloud networks.

Fig. 8 plots the network workload in GBps for a cloud-fog-edge workflow scheduling system containing 5000 nodes each in the cloud, fog, and edge layers. An equal number of nodes are used for simplicity of explanation and it does not pose any limitations on the design of the systems. There are a total of 10 different systems for cloud and fog each, which have been modelled. These 10 x 3 systems relate to the different workloads on the systems in terms of the tasks being received by the system, beginning from 500 tasks, up to 5000 tasks per hour while the capacity all the systems is constant. The network load is depicted using blue, orange, and grey bars for cloud, fog and edge respectively. The edge values are relatively small and therefore their bars are not visible but the labels can be seen on the right of the orange bars. The fourth set of bars in the yellow colour represents the aggregate network load of the cloud-fog-edge systems. The minimum network load is for the system with 500 tasks; 255 GBps (cloud), 58 GBps (fog), 5 GBps (edge), and 106 (aggregate). Note that the network load for the fog system with 500 nodes was 84 in the cloud-fog system depicted in Fig. 7; the higher value in that case is due to the cloud system that had 7500 maximum nodes. Since we modelled systems with cloud scheduling first, the fog is scheduled after the 7500 cloud nodes and this increase the overall load of system and in turn the factor ψ, which is dependent on the steady-state probabilities and system utilisation. The network load consistently rises to a maximum of 4,118 GBps (cloud) and 1433 GBps (aggregate) for the busiest workflow scheduling system with 5000 nodes each in the cloud, fog, and edge. The reason for relatively smaller values for fog (and edge) systems has been explained in the previous paragraph.

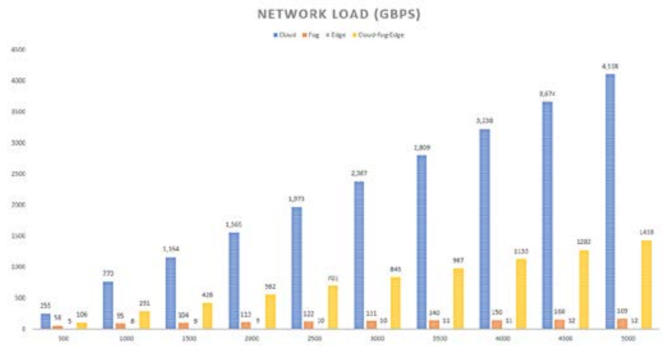

Fig. 8. Network Load in GBps (Cloud-Fog-Edge System with 5,000 Nodes Each).

D. Average Response Delay

The calculations of average Response Delay (RD) were explained earlier in Section III and Equation (9). We now in this section will analyse the system performance related to response delay of various cloud-fog, and cloud-fog-edge systems. We have modelled cloud-only and several other different configuration systems but we will limit our analysis to the cloud-fog and cloud-fog-edge systems for the sake of brevity.

Fig. 9 depicts the average response delay in milliseconds (ms) for a cloud-fog workflow scheduling system containing 7,500 nodes each in the cloud and fog. This system is similar to the one depicted in Fig. 7. There are a total of 15 different systems for cloud and fog each that have been modelled and their average response delays have been computed according to Equation (9). The cloud response delay is depicted using blue bars and the fog delay is depicted using orange bars. The third set of bars in the grey colour represents the aggregate response delay of the cloud-fog systems. The minimum delay is for the system with 500 tasks; 404ms, 102ms, and 253ms for cloud, fog, and aggregate delays, respectively. The increase in the delay is consistent and non-linear, however, minimal. The minimal increase is because the system serves the jobs in parallel. Also, this minimal value and consistent increase is due to the equal load of each job as has been explained earlier. This increase in the delay depends on the system and task characteristics that can be tuned using the factor ψ and using tasks with some low or high-variance distributions.

Fig. 10 depicts the average response delay in milliseconds (ms) for a cloud-fog-edge workflow scheduling system containing 5000 nodes each in the cloud, fog, and edge. This system is similar to the one depicted in Fig. 8. There are a total of 10 different systems for cloud, fog, and edge, each, which have been modelled and their average response delays have been computed according to Equation (9). The cloud, fog, edge, and aggregate response delays are depicted using blue, orange, grey, and yellow bars. The minimum delay is for the system with 500 tasks; 404ms, 102ms, 100ms, and 202ms for cloud, fog, edge, and aggregate delays, respectively. The increase in the delay is consistent and non-linear, however, minimal. We have explained the reasons for this while explaining Fig. 9. Note that using a cloud-fog-edge system as opposed to cloud-only or cloud-fog system significantly decreases the aggregate delay bringing to half of it (from 652 to 326 ms). Obviously, increasing the relative number of edge nodes compared to cloud and fog can significantly bring down the aggregate delays.

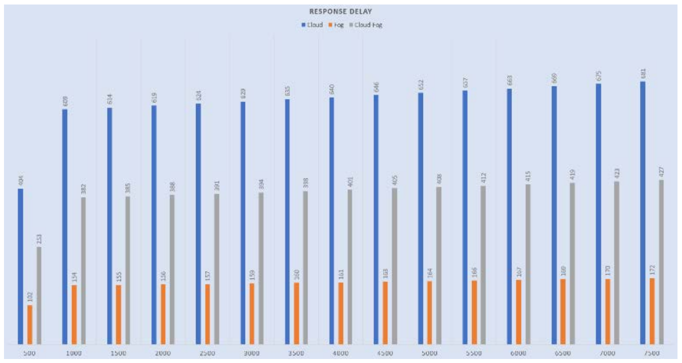

Fig. 9. Response Delay -- Milliseconds -- Cloud-Fog System with 7,500 Nodes Each.

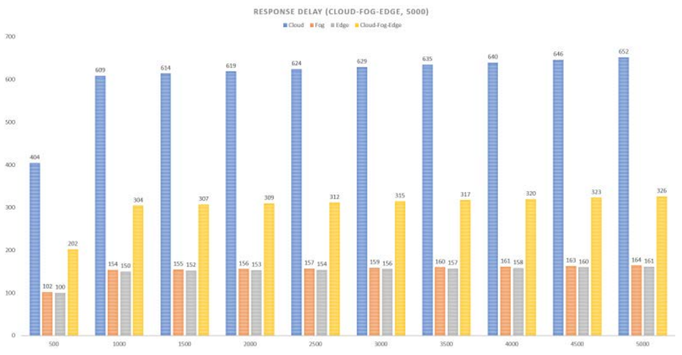

Fig. 10. Response Delay -- Milliseconds -- Cloud-Fog-Edge System with 5000 Nodes Each.

E. Energy Consumption

The network energy consumption per hour (NE) was defined and explained in Section III and calculated using Equation (10). We, in this section, analyse the system performance related to network energy consumption of various cloud-fog-edge systems. We have modelled cloud-only and cloud-fog, and several other different configuration systems, however, for the sake of brevity, we will limit our analysis in this section to the cloud-fog-edge systems.

Fig. 11 depicts the network energy consumption in MWh for a cloud-fog-edge workflow scheduling system containing 5000 nodes each in the cloud, fog, and edge. This system is similar to the one depicted in Fig. 10, however, it provides network energy consumption data. There are a total of 10 different systems for cloud, fog, and edge, each, which have been modelled and their energy consumption have been computed according to Equation (10). The cloud, fog, edge, and aggregate network energy consumption are depicted using blue, orange, grey, and yellow bars. The minimum energy consumption is for the system with 500 tasks; 496, 112, 10, and 206 MWh for cloud, fog, edge, and aggregate energy consumption, respectively. The increase in the consumption is consistent, non-linear, and reaches roughly 16 times (496 to 8006) as opposed to the 10 times increase in the number of nodes (500 to 5000). We have explained the reasons for this while explaining Fig. 9. Note that using a cloud-fog-edge system as opposed to cloud-only or cloud-fog system significantly decreases the aggregate energy consumption (from 8006 to 2786 MWh).

F. Energy Cost

The energy cost (EC) per hour was defined and explained in Section III and calculated using Equation (11). We here analyze the system performance related to network energy cost of various cloud-fog-edge systems. We have modelled cloud-only and cloud-fog, and several other different configuration systems; however, for the sake of brevity, we will limit our analysis in this section to the cloud-fog-edge systems.

Fig. 12 depicts the energy cost in GBP (x million) for a cloud-fog-edge workflow scheduling system containing 5000 nodes each in the cloud, fog, and edge. There are a total of 10 different systems for cloud, fog, and edge, each, which have been modelled and their energy cost have been computed according to Equation (11). The cloud, fog, edge, and aggregate monthly network energy consumption are depicted using blue, orange, grey, and yellow bars. The minimum monthly energy cost is for the system with 500 tasks; 62, 14, 1, and 26 million GBP for cloud, fog, edge, and aggregate monthly cost respectively. The increase in the cost is consistent, non-linear, and reaches roughly 16 times (62 to 1003) as opposed to the 10 times increase in the number of nodes (500 to 5000). Note that using a cloud-fog-edge system as opposed to cloud-only or cloud-fog system significantly decreases the aggregate energy consumption (from 1003 to 349 million GBP).

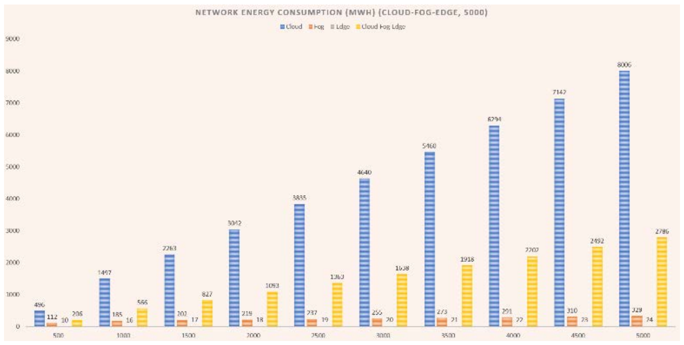

Fig. 11. Network Energy Consumption -- MWh -- Cloud-Fog-Edge System with 5000 Nodes Each.

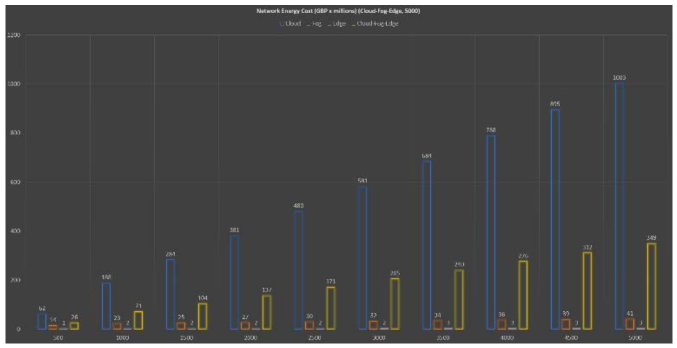

Fig. 12. Monthly Energy Cost – GBP x million -- Cloud-Fog-Edge System with 5000 Nodes Each.

V. Discussion, Utilization and Conclusion

Fog and edge computing has emerged as an important paradigm to address many challenges related to time-sensitive and real-time applications, high network loads, user privacy, security, and others. These developments offer huge potential; however, many efforts are needed to study and design applications and systems for these emerging computing paradigms.

This paper provided a detailed study of workflow scheduling and offloading of service-based applications. We developed different models of cloud, fog and edge systems and studied the scheduling of workflows using a range of system sizes and application intensities. Firstly, we developed several Markov models of cloud, fog, and edge systems and computed the steady-state probabilities for system utilization and stability. Secondly, using steady-state probabilities, we defined a range of system metrics to study the performance of workflow scheduling and offloading including, network load, response delay, energy consumption, and energy costs. An extensive investigation of application intensities and cloud, fog, and edge system sizes revealed that significant benefits can be accrued from the use of fog and edge computing in terms of low network loads, response times, energy consumption and costs.

The proposed workflow scheduling and offloading models can be utilized in practice to study a range of applications and derive several benefits. Firstly, different well-known standardized workflow can be plugged in our proposed workflow scheduling models to study their various performance behaviors including network load, average response delay, energy consumption, and energy cost, and this can be done for a range of cloud only, cloud-fog, and cloud-fog-edge systems. Some examples of standardized workflows include Montage workflow, SIPHIT workflow, epigenomics workflow, LIGO workflow, Cyber-Shake workflow, and more; see [23], for explanations and use cases of these workflows, and [19]–[22] for additional examples for practical utilization of our work. Let us take the epigenomics workflow as an example that captures the execution workflows related to the operations involved in genome sequences. Such a workflow can be embedded in our proposed Markov model by defining the execution workflows within the Markov chain and thereby we can study how that workflow will behave for cloud-only, cloud-fog, and cloud-fog-edge systems in terms of the network load, average response delay, energy consumption, and energy cost of the system.

The computational loads used by the different tasks modeled in this paper are the same. Similarly, the network loads in terms of the bytes sent around the network are also the same. However, this is not the limitation of the model. The equations developed in the models do use different computational and network loads and other parameters. The proposed models can capture the additional network load due to the task offloading or the different execution times of tasks in nodes due to the differences in their computational performance such as device speed and power that may also lead to higher energy consumption by the devices and networks. This is because the models define separately each of the tasks' computational and network loads, as well as computational and network characteristics of the devices and networks in the cloud, fog and edge layers. These could be easily changed based on various workflows to study their performance. However, the developed system is a Markov chain and therefore it does use exponential distribution to capture the arrival and departure rates.

VI. Future Work

The future work will focus on investigating variations in the cloud, fog, and edge system configurations, variations in application intensities, and variations in task sizes. Moreover, we plan to investigate system utilization and stability models and their effects on the computations of system characteristics including energy consumption and costs, response times, and network throughput.

Acknowledgment

The author gratefully acknowledges the approval and the support of this research from the Deanship of Scientific Research study by the grant no CIT-2019-1-10-F-8452, Northern Border University, Arar 91431, Kingdom of Saudi Arabia.

References

[1] K. Tange, M. De Donno, X. Fafoutis, and N. Dragoni, "A Systematic Survey of Industrial Internet of Things Security: Requirements and Fog Computing Opportunities," IEEE Commun. Surv. Tutorials, vol. 22, no. 4, pp. 2489–2520, Oct. 2020, doi: 10.1109/COMST.2020.3011208.

[2] Y. Arfat et al., "Enabling Smarter Societies through Mobile Big Data Fogs and Clouds," in Procedia Computer Science, 2017, vol. 109, pp. 1128–1133, doi: 10.1016/j.procs.2017.05.439.

[3] T. Muhammed, R. Mehmood, A. Albeshri, and I. Katib, “UbeHealth: A personalized ubiquitous cloud and edge-enabled networked healthcare system for smart cities,” IEEE Access, vol. 6, pp. 32258–32285, 2018, doi: 10.1109/ACCESS.2018.2846609.

[4] A. Yousefpour et al., “All one needs to know about fog computing and related edge computing paradigms: A complete survey,” J. Syst. Archit., vol. 98, pp. 289–330, 2019, doi: 10.1016/j.sysarc.2019.02.009.

[5] L. A. Tawalbeh, R. Mehmood, E. Benkhlifa, and H. Song, “Mobile Cloud Computing Model and Big Data Analysis for Healthcare Applications,” IEEE Access, vol. 4, pp. 6171–6180, 2016, doi: 10.1109/ACCESS.2016.2613278.

[6] Y. Han, X. Wang, V. C. M. Leung, D. Niyato, X. Yan, and X. Chen, “Convergence of Edge Computing and Deep Learning: A Comprehensive Survey,” 2019.

[7] W. Yu et al., “A Survey on the Edge Computing for the Internet of Things,” IEEE Access, vol. 6, pp. 6900–6919, 2018, doi: 10.1109/ACCESS.2017.2778504.

[8] T. Mohammed, A. Albeshri, I. Katib, and R. Mehmood, “UbiPriSEQ—Deep reinforcement learning to manage privacy, security, energy, and QoS in 5G IoT hetnets,” Appl. Sci., vol. 10, no. 20, 2020, doi: 10.3390/app10207120.

[9] N. Janbi, I. Katib, A. Albeshri, and R. Mehmood, “Distributed Artificial Intelligence-as-a-Service (DAIaaS) for Smarter IoE and 6G Environments,” Sensors, vol. 20, no. 20, p. 5796, Oct. 2020, doi: 10.3390/s20205796.

[10] A. Marchisio et al., “Deep Learning for Edge Computing: Current Trends, Cross-Layer Optimizations, and Open Research Challenges,” in Proceedings of IEEE Computer Society Annual Symposium on VLSI, ISVLSI, Jul. 2019, vol. 2019-July, pp. 553–559, doi: 10.1109/ISVLSI.2019.00105.

[11] Z. Zhou, X. Chen, E. Li, L. Zeng, K. Luo, and J. Zhang, “Edge Intelligence: Paving the Last Mile of Artificial Intelligence With Edge Computing,” Proc. IEEE, vol. 107, no. 8, pp. 1738–1762, May 2019, doi: 10.1109/JPROC.2019.2918951.

[12] S. Bosaeed, I. Katib, and R. Mehmood, “A Fog-Augmented Machine Learning based SMS Spam Detection and Classification System,” 2020, doi: 10.1109/FMEC49853.2020.9144833.

[13] J. C. Guevara, R. da S. Torres, and N. L. S. da Fonseca, “On the classification of fog computing applications: A machine learning perspective,” J. Netw. Comput. Appl., vol. 159, p. 102596, Jun. 2020, doi: 10.1016/J.JNCA.2020.102596.

[14] G. Javadzadeh and A. M. Rahmani, “Fog Computing Applications in Smart Cities: A Systematic Survey,” Wirel. Networks 2019 262, vol. 26, no. 2, pp. 1433–1457, Dec. 2019, doi: 10.1007/S11276-019-02208-Y.

[15] M. Abdel-Basset, R. Mohamed, M. Elhoseny, A. K. Bashir, A. Jolfaei, and N. Kumar, “Energy-Aware Marine Predators Algorithm for Task Scheduling in IoT-Based Fog Computing Applications,” IEEE Trans. Ind. Informatics, vol. 17, no. 7, pp. 5068–5076, Jul. 2021, doi: 10.1109/TII.2020.3001067.

[16] S. N. Srirama, F. M. S. Dick, and M. Adhikari, “Akka framework based on the Actor model for executing distributed Fog Computing applications,” Futur. Gener. Comput. Syst., vol. 117, pp. 439–452, Apr. 2021, doi: 10.1016/J.FUTURE.2020.12.011.

[17] M. Abdel-Basset, D. El-Shahat, M. Elhoseny, and H. Song, “Energy-Aware Metaheuristic Algorithm for Industrial-Internet-of-Things Task Scheduling Problems in Fog Computing Applications,” IEEE Internet Things J., vol. 8, no. 16, pp. 12638–12649, Aug. 2021, doi: 10.1109/JIOT.2020.3012617.

[18] M. Goudarzi, H. Wu, M. Palaniswami, and R. Buyya, “An Application Placement Technique for Concurrent IoT Applications in Edge and Fog Computing Environments,” IEEE Trans. Mob. Comput., vol. 20, no. 4, pp. 1298–1311, Apr. 2021, doi: 10.1109/TMC.2020.2967041.

[19] M. Farid, R. Latip, M. Hussin, and N. A. W. A. Hamid, “A Survey on QoS Requirements Based on Particle Swarm Optimization Scheduling Techniques for Workflow Scheduling in Cloud Computing,” Symmetry 2020, Vol. 12, Page 551, vol. 12, no. 4, p. 551, Apr. 2020, doi: 10.3390/SYM12040551.

[20] M. Kumar, S. C. Sharma, A. Goel, and S. P. Singh, “A comprehensive survey for scheduling techniques in cloud computing,” J. Netw. Comput. Appl., vol. 143, pp. 1–33, Oct. 2019, doi: 10.1016/J.JNCA.2019.06.006.

[21] I. Grosof, M. Harchol-Balter, and A. Scheller-Wolf, “The Finite-Skip Method for Multiserver Analysis,” vol. 1, no. 1, Sep. 2021, Accessed: Dec. 21, 2021. [Online]. Available: https://arxiv.org/abs/2109.12663v1.

[22] M. Ala’anzy and M. Othman, “Load Balancing and Server Consolidation in Cloud Computing Environments: A Meta-Study,” IEEE Access, vol. 7, pp. 141868–141887, 2019, doi: 10.1109/ACCESS.2019.2944420.

[23] M. Adhikari, T. Amgoth, and S. N. Srirama, “A Survey on Scheduling Strategies for Workflows in Cloud Environment and Emerging Trends,” ACM Comput. Surv., vol. 52, no. 4, Aug. 2019, doi: 10.1145/3325097.

[24] R. Deng, R. Lu, C. Lai, T. H. Luan, and H. Liang, “Optimal Workload Allocation in Fog-Cloud Computing Toward Balanced Delay and Power Consumption,” IEEE Internet Things J., vol. 3, no. 6, pp. 1171–1181, Dec. 2016, doi: 10.1109/JIOT.2016.2565516.

[25] R. Deng, R. Lu, C. Lai, and T. H. Luan, “Towards power consumption-delay tradeoff by workload allocation in cloud-fog computing,” IEEE Int. Conf. Commun., vol. 2015-September, pp. 3909–3914, Sep. 2015, doi: 10.1109/ICC.2015.7248934.

[26] J. Oueis, E. C. Strinati, S. Sardellitti, and S. Barbarossa, “Small cell clustering for efficient distributed fog computing: A multi-user case,” 2015 IEEE 82nd Veh. Technol. Conf. VTC Fall 2015 - Proc., Jan. 2016, doi: 10.1109/VTCFALL.2015.7391144.

[27] M. Aazam and E. N. Huh, “Fog computing micro datacenter based dynamic resource estimation and pricing model for IoT,” Proc. - Int. Conf. Adv. Inf. Netw. Appl. AINA, vol. 2015-April, pp. 687–694, Apr. 2015, doi: 10.1109/AINA.2015.254.

[28] K. Habak, M. Ammar, E. W. Zegura, and K. A. Harras, “Workload management for dynamic mobile device clusters in edge femtoclouds,” 2017 2nd ACM/IEEE Symp. Edge Comput. SEC 2017, Oct. 2017, doi: 10.1145/3132211.3134455.

[29] K. Intharawijitr, K. Iida, and H. Koga, “Analysis of fog model considering computing and communication latency in 5G cellular networks,” 2016 IEEE Int. Conf. Pervasive Comput. Commun. Work. PerCom Work. 2016, Apr. 2016, doi: 10.1109/PERCOMW.2016.7457059.

[30] H. J. Jeong, I. Jeong, H. J. Lee, and S. M. Moon, “Computation offloading for machine learning web apps in the edge server environment,” Proc. - Int. Conf. Distrib. Comput. Syst., vol. 2018-July, pp. 1492–1499, Jul. 2018, doi: 10.1109/ICDCS.2018.00154.

[31] T. Zhao, S. Zhou, X. Guo, and Z. Niu, “Tasks scheduling and resource allocation in heterogeneous cloud for delay-bounded mobile edge computing,” IEEE Int. Conf. Commun., Jul. 2017, doi: 10.1109/ICC.2017.7996858.

[32] S. Agarwal, S. Yadav, and A. K. Yadav, “An Efficient Architecture and Algorithm for Resource Provisioning in Fog Computing,” undefined, vol. 8, no. 1, pp. 48–61, Jan. 2016, doi: 10.5815/IJIEEB.2016.01.06.

[33] A. Yousefpour et al., “All one needs to know about fog computing and related edge computing paradigms: A complete survey,” J. Syst. Archit., vol. 98, pp. 289–330, Sep. 2019, doi: 10.1016/J.SYSARC.2019.02.009.

[34] D. Zeng, L. Gu, S. Guo, Z. Cheng, and S. Yu, “Joint Optimization of Task Scheduling and Image Placement in Fog Computing Supported Software-Defined Embedded System,” IEEE Trans. Comput., vol. 65, no. 12, pp. 3702–3712, Dec. 2016, doi: 10.1109/TC.2016.2536019.

[35] M. Chen and Y. Hao, “Task Offloading for Mobile Edge Computing in Software Defined Ultra-Dense Network,” IEEE J. Sel. Areas Commun., vol. 36, no. 3, pp. 587–597, Mar. 2018, doi: 10.1109/JSAC.2018.2815360.

[36] T. Huang et al., “Energy-Efficient Computation Offloading for Multimedia Workflows in Mobile Cloud Computing,” Lect. Notes Inst. Comput. Sci. Soc. Telecommun. Eng. LNICST, vol. 270, pp. 113–123, Nov. 2018, doi: 10.1007/978-3-030-12971-2_7.

[37] J. Wan, B. Chen, S. Wang, M. Xia, D. Li, and C. Liu, “Fog Computing for Energy-Aware Load Balancing and Scheduling in Smart Factory,” IEEE Trans. Ind. Informatics, vol. 14, no. 10, pp. 4548–4556, Oct. 2018, doi: 10.1109/TII.2018.2818932.

[38] M. A. Benblidia, B. Brik, L. Merghem-Boulahia, and M. Esseghir, "Ranking fog nodes for tasks scheduling in fog-cloud environments: A fuzzy logic approach," 2019 15th Int. Wirel. Commun. Mob. Comput. Conf. IWCMC 2019, pp. 1451–1457, Jun. 2019, doi: 10.1109/IWCMC.2019.8766437.

[39] R. O. Aburukba, M. AliKarrar, T. Landolsi, and K. El-Fakih, "Scheduling Internet of Things requests to minimize latency in hybrid Fog–Cloud computing," Futur. Gener. Comput. Syst., vol. 111, pp. 539–551, Oct. 2020, doi: 10.1016/J.FUTURE.2019.09.039.

[40] W. J. Stewart, Numerical Solution of Markov Chains (Probability: Pure & Applied). New York: CRC, 1991.

[41] S. Altowaijri, R. Mehmood, and J. Williams, "A Quantitative Model of Grid Systems Performance in Healthcare Organisations," 2010 Int. Conf. Intell. Syst. Model. Simul., pp. 431–436, 2010, doi: 10.1109/ISMS.2010.84.

[42] R. Mehmood, R. Meriton, G. Graham, P. Hennelly, and M. Kumar, "Exploring the influence of big data on city transport operations: a Markovian approach," Int. J. Oper. Prod. Manag., vol. 37, no. 1, pp. 75–104, 2017, doi: 10.1108/IJOPM-03-2015-0179.

[43] A. Andrae and T. Edler, "On Global Electricity Usage of Communication Technology: Trends to 2030," Challenges, vol. 6, no. 1, pp. 117–157, Apr. 2015, doi: 10.3390/challe6010117.

[44] Selectra, "Energy bills: How much is the average electric bill?" https://selectra.co.uk/energy/guides/billing/electricity (accessed Nov. 29, 2021).