Disaggregation for Energy Efficient Fog in Future 6G Networks

Opeyemi O. Ajibola, *Member, IEEE*, Taisir E. H. El-Gorashi, and Jaafar M. H. Elmirghani, *Fellow, IEEE*

***Abstract*—We study the benefits of adopting server disaggregation in the fog computing tier by evaluating energy efficient placement of interactive apps in a future (6G) fog network. Using a mixed integer linear programming (MILP) model, we compare the adoption of traditional server (TS) and disaggregated server (DS) architectures in a fog network that comprises selected fog sites. We also propose a heuristic for energy efficient and delay aware placement of interactive fog apps in a fog network which effectively mimics the MILP model formulated in this paper. Compared to a non-federated fog computing layer, federation of selected fog computing sites over the metro-access network enables significant reductions of the total fog computing power consumption (TFPC). Relative to the use of TSs in the fog network, the adoption of DSs improves the energy efficiency of the fog network and enables up to 18% reduction in TFPC. To minimize response time, more instances of interactive fog apps are provisioned in a fog network that is implemented over a network topology with high delay penalty. Our result also shows that the proximity of metro-central offices and radio cell sites to geo-distributed users makes them important fog sites for provisioning delay-sensitive fog applications.**

***Index Terms*—Disaggregated servers, fog network, disaggregation, fog computing, composable infrastructures, optical access and metro networks, software defined infrastructures, energy efficient networks, MILP.**

I. Introduction

CLOUD computing became an integral part of the global society over the last two decades because it enabled improved capital and operational efficiencies relative to the traditional distributed computing architecture. Adoption of cloud computing in its different service offerings such as Infrastructure as a Service (IaaS), Platform as a Service (PaaS) and Software as a Service (SaaS) spans across a variety of personal, enterprise and public applications. Some of the key drivers of this trend include digital transformation, artificial intelligence and machine learning and Internet of Things (IoT) [1]. By 2022, it is estimated that the global spending on IoT will reach $1.2 trillion [2]. Furthermore, Cisco predicts that about 500 billion smart devices will be connected by 2030 [3].

The growing uptake of the cloud computing and IoT paradigms is expected to enable a new range of emerging and future Internet applications at the edge of telecommunication networks. However, the centralized architecture of the traditional cloud computing paradigm is a major inhibiting factor to the emergence of some of these applications which require real-time or near-real-time computation. In addition, the volume, velocity, and variety of data generated by geo-distributed IoT-devices and end-users of these emerging Cloud-IoT applications combined with traditional network traffic can also overwhelm network infrastructures. Therefore, leading to significant increase in the cost of owning and operating networks and their carbon footprint. Furthermore, the centralized cloud computing architecture can degrade the performance of some future applications due to increased network congestion. Hence, the concept of edge/fog computing [4]–[6] has been proposed in recent times to address some of these challenges.

The fog computing paradigm extends cloud computing to the network edge to support emerging and future Internet applications and services. This is enabled via the introduction of a new intermediate computation tier called the fog computing tier. The fog computing tier is located between the centralized cloud computing tier and geo-distributed IoT-devices and end-users of emerging applications at the network edge. The fog tier comprises heterogeneous devices and nodes called fog nodes. These heterogeneous fog nodes/devices adopt heterogeneous network infrastructures (both wireless and wired) for connectivity. The geo-distributed heterogeneous nodes may also be orchestrated collectively as a fog federation within a given area to enable a fog as a service (FaaS) or fog Infrastructure as a service (IaaS, FIaaS) business model [7]–[9].

Like the cloud computing tier, fog nodes possess compute, storage, and networking capabilities. Virtualization might also be supported in fog nodes to ensure efficient use of the computation capacity as obtainable in the cloud computing tier [7]. However, fog nodes are unable to enjoy other benefits obtainable in centralized cloud computing infrastructures. For example, the ability to strategically position massive computational resources at little or no opportunity cost is limited in the fog computing tier. Furthermore, the

Manuscript received 30 July 2021; revised 26 November 2021, 12 January 2022, and 12 February 2022; accepted 14 March 2022. Date of publication 17 March 2022; date of current version 19 August 2022. This work was supported in part by the Engineering and Physical Sciences Research Council (EPSRC); in part by the INTelligent Energy aware NETworks (INTERNET) under Grant EP/H040536/1; in part by the SwiTching And tRansmission (STAR) under Grant EP/K016873/1; and in part by the Terabit Bidirectional Multi-User Optical Wireless System (TOWS) Project under Grant EP/S016570/1. All data is provided in the results section of this paper. The editor coordinating the review of this article was D. Van Veen *(Corresponding author: Opeyemi O. Ajibola.)*

The authors are with the School of Electronic and Electrical Engineering, University of Leeds, Leeds LS2 9JT, U.K. (e-mail: o.ajibola@ultracellnetworks.com; t.e.h.elgorashi@leeds.ac.uk; j.m.h.elmirghani@leeds.ac.uk).

Digital Object Identifier 10.1109/TGCN.2022.3160397

fog computing nodes cannot fully maximize the benefits of using commodity hardware as obtainable in hyper-scale cloud datacenters (DCs) because they are relatively smaller in size. In recent times, large DC infrastructure providers such as Facebook and Microsoft have also begun to explore the use of the server disaggregation concept as a tool to further improve the overall efficiency of DC infrastructure [10].

Server disaggregation proposes the physical or logical separation of traditional server (TS) intrinsic resources into pools of homogeneous resources, which can be composed, decomposed, and recomposed on-demand over high bandwidth and low latency networks, to support applications. This concept addresses the limitations associated with TSs such as poor resource modularity and lifecycle management, the need for purpose-built servers in computing clusters, high power consumption and capital expenditure resulting from inefficient resource utilization [11]–[14]. Adoption of server disaggregation concept in the fog computing tier could improve fog computing efficiency to approach efficiencies that are traditionally attributed to the cloud computing tier.

In this paper, we explore the gains that server disaggregation can enable in fog networks relative to TS architecture and study the impact of fog computing on metro networks. In [15], we conducted an initial study to demonstrate some potential benefits of adopting disaggregated servers (DSs) in the fog computing tier, concluding that the network bottlenecks may inhibit optimal performance and that users experience may suffer due to the resulting increased hop counts between users and fog app instances. This paper extends our initial work as follows; (i) it provides for the first time the complete MILP model, which has been extended by considering queuing delay in the network topology; (ii) it considers emerging fog apps with varying delay sensitivity; (iii) it investigates the impact of user distribution on the fog network; and finally, (iv) it develops a fast and scalable policy (heuristic) which mimics the MILP model for practical deployment in large scenarios, and to verify the MILP.

The rest of this paper is organized as follows. We review the concepts of fog nodes federation and resource disaggregation and related literatures in Section II. Section III presents the system setup and the MILP model formulated to investigate the placement of applications in fog networks. Section IV presents a scalable heuristic that is comparable to the formulated MILP model. Section V gives the evaluation scenarios along with numerical results and discusses the insights obtained by solving the MILP model and heuristic. Finally, the paper is concluded in Section VI along with a brief discussion of future work and future directions.

II. Fog Networks and Disaggregation

A. Fog Networks

The fog computing layer is an intelligent intermediate layer between centralized cloud computing servers and geo-distributed connected devices and end-users. This layer provides distributed computing infrastructure at network edges (i.e., metro-access network). Hence, the fog computing layer complements the centralized cloud computing infrastructure by extending cloud-like services closer to end-users and connected things for improved performance and to support new classes of applications. The fog computing layer can also enable reductions in computing and network infrastructures [16]–[18].

The traditional definition of fog computing classifies any device with compute, storage and network connectivity as a fog node [19]. However, some locations and devices may be more optimal than others because of factors such as energy efficiency, resource capacity, node availability, resource reusability and utilization efficiency. In recent times, wired edge network devices [20] and wireless edge network devices alike are often equipped with extra computing capability to support the hosting of non-network functions and Machine Learning (ML) and Artificial intelligence (AI) functions at the edge of the network. This trend is expected to be sustained in future 6G networks as the use of ML and AI capability increases at the edge of future networks [21]. Therefore, the traditional network locations, i.e., central offices (COs) and radio cell sites (CSs) where such devices are located may be relatively more optimal for some applications.

For instance, compared to the use of dedicated fog computing nodes, the availability and use of spare computing capacities at traditional network locations, i.e., COs and radio CSs for fog computing can promote sustainability and improved energy efficiency. This is because such locations are centralized at the network edge and can be easily accessed by multiple distributed devices and end-users at the network edge. Therefore, reducing the number of active dedicated fog computing nodes at the network edge and their corresponding power consumption. The adoption of commodity hardware to support network function virtualization at the edge of the network also supports the use of some traditional network locations to provide shared fog computing capability. As a result, the energy efficiency of NFV [22] can be leveraged in the fog computing layer. Additionally, existing computing capacities at other edge locations such as enterprise offices and public locations can also be made available to all end-users and devices to further promote sustainability in the fog computing layer.

However, efficient sharing of fog computing capacity at the network edge requires the federation of independent fog nodes via coordinated orchestration and control of distributed fog nodes over the network infrastructure. Such a group of linked fog nodes with non-application specific resource is called a federated fog network [23]. To ensure low latency, the span of each federation of fog nodes could be limited to satisfy the requirements of some applications. The rising demand for the federation of distributed fog nodes (fog network) is motivating the emergence of service-oriented access models for computing resource in the fog computing tier in the form of Fog Infrastructure as a Service (FIaaS) and Fog as a Service (FaaS) [23], [24]. The adoption of service-oriented consumption of federated fog computing capacity can reduce the number of required fog nodes. Fog networks can also enable new revenue streams which can offset the total cost of ownership incurred by providers that deploy fog computing nodes [23].

In spite of the opportunities that the federation of fog nodes enables, a practical implementation of federated fog node in the multi-domain network edge has inherent challenges. For instance, a holistic framework is required to manage the federation of fog node across multiple operating domains [25], [26]. Such as a framework should support the discovery of fog nodes, dynamic management of compute, network and storage resources, ensure compliance with service level agreements and establish trust and security mechanisms in the fog network.

B. Resource Disaggregation

Resource disaggregation proposes the separation of the resource components of traditional computing servers into physical or logical pools of homogenous resource types, i.e., homogenous resources are in the same pool. The concept addresses the problems of resource stranding and fragmentation which inhibit efficiencies, modularity, agility, and effective resource lifecycle management in traditional computing infrastructure. Resource disaggregation solves these problems by relaxing the rigid physical resource utilization boundary of traditional servers to permit on-demand use of un-collocated resource components (via networks) to form logical computing hosts. Disaggregation of the server intrinsic resource components can be performed at different scales, i.e., rack-scale, pod-scale and DC-scale [14], [27], [28]. Furthermore, the concept of resource disaggregation can be implemented physically or logically over a corresponding physical hardware. On the one hand, physical disaggregation ensures that the allocation of physical components leads to the creation of physical pools (nodes/racks/pods) of homogenous components. On the other hand, logical disaggregation virtually implements resource disaggregation on homogenously and/or heterogeneously resourced pools to support on-demand creation of logical servers to run applications.

Resource disaggregation promises significant improvements in the efficiency of computing infrastructure at both the fog and cloud computing tiers of the cloud of things continuum. However, such infrastructure requires software defined (SD) techniques [29] and appropriate network connectivity to maximize the potential benefits enabled by resource disaggregation. A computing infrastructure that implements resource disaggregation builds on the efficiencies, flexibility and agility enabled by SD-techniques to enable even greater efficiency. Additionally, an appropriate network topology that supports on-demand creation of logical hosts from disaggregated resource components is essential for practical implementation of any form of resource disaggregation [30]. This is because resource disaggregation exposes high bandwidth and low latency intra-host communication of traditional servers onto the DC network.

C. Related Works

Comprehensive studies of energy efficient communication networks have been done in [31]–[36] as a result of increasing adoption of digital solutions and services, which are often domiciled in remote DCs of cloud computing services providers. However, the recent introduction of the fog computing layer motivates an extension of similar studies to the fog computing layer. The authors of [16]–[18] have also conducted extensive studies on how the introduction of the fog computing layer can enable significant power savings in the cloud of things architecture relative to the adoption of a 2-tier architecture. The work in [9] also showed that a 3-tier cloud of things architecture is better than a 2-tier cloud of things architecture especially when energy consumption and latency are used as comparison metrics; hence, a 3-tier cloud of things architecture is adopted in this paper. However, the goal of this paper is to minimize the number of computational nodes in the lowest tier of the 3-tier architecture by employing the concept of server disaggregation for the first time in this context. In [7], the authors showed that the execution of large tasks experiences significant processing delay in the fog computing layer. Hence, scaling fog computing capacity at the expense of higher financial cost is required. In [23], a platform which orchestrates distributed fog nodes over the network to form fog networks that support on-demand deployment of applications and services was proposed. The authors in [37] introduced a programming model for present and emerging geo-distributed, massive and latency-sensitive applications. The programming model was validated using two latency sensitive applications. In [19], the authors proposed a high-level policy for placing applications in the fog computing tier based on application latency requirements only. The policy is generic and does not consider the impact of factors such as energy efficiency, resource utilization, networks, and disaggregation on optimal application placement. Workload offloading [38], [39] and workload assignment [40], [41] are two different approaches used to study the minimization of response time in fog computing tier in the literature. Given a set of fog workloads, the workload assignment approach attempts to assign such workloads to fog computing nodes while optimizing a specific cost such as response time or energy. On the other hand, the workload offloading approach aims to design policies that offload fog computing requests to other fog nodes or to the cloud while optimizing a specific cost.

The workload assignment approach is adopted in this paper by formulating a MILP model to assign varying classes of interactive fog applications. This is because the workload offloading approach may be less appropriate for interactive workloads with stringent delay requirements. A heuristic is proposed from the insights obtained from the MILP model. In contrast to existing literatures, this work is focused on the fog computing tier only and its associated access and metro networks. The things and cloud computing layers of the cloud of thing architecture are not explicitly considered. However, the impact of cloud destined network traffic on the overall performance of the fog computing system is modeled and a cost is associated with fog servers that are deployed very close to the things layer.

Like the works of the authors in [16]–[18], [40], this work explores power consumption, cost and latency in the fog computing layer. However, the novelty of this work is that it considers disaggregation of fog servers and different classes

TABLE I
A COMPARISON OF THE CONTRIBUTIONS OF THIS PAPER WITH THAT OF THE RELATED WORKS IN THE LITERATURE

Ref	Cloud of Things Architecture		Propositions and Evaluation Metrics				Methodology		Server Architecture		Fog Node Federation
	3 tiers	2 tiers	Latency (Delay)	Power (Energy)	Orchestration (Policy)	Cost	Workload Offloading	Workload Assignment	TS	DS	
[7]	✓	✓	✓	✓	✓	✓	-	-	✓	-	-
[9]	✓	✓	✓	✓	-	-	-	-	✓	-	✓
[19]	✓	-	✓	-	✓	-	-	-	✓	-	-
[39]	✓	-	✓	-	✓	-	✓	-	✓	-	✓
[13]	-	-	-	✓	-	-	-	-	✓	✓	-
[16]	✓	-	-	✓	✓	-	-	✓	✓	-	-
[17]	✓	-	-	✓	✓	-	-	✓	✓	-	-
[18]	✓	-	-	✓	✓	-	-	✓	✓	-	-
[40]	✓	✓	✓	✓	-	✓	-	✓	✓	-	-
[23]	✓	-	✓	-	✓	-	-	-	-	-	✓
[30]	-	-	-	✓	-	-	-	-	✓	✓	-
[37]	✓	✓	✓	-	✓	-	-	-	-	-	-
[38]	✓	-	✓	-	✓	-	✓	-	✓	-	✓
[41]	✓	✓	✓	✓	-	-	-	✓	✓	-	-
[42]	-	-	-	✓	-	-	-	-	-	✓	-
This paper	✓	-	✓	✓	✓	✓	-	✓	✓	✓	✓

of delay sensitive (interactive) applications. Other factors that may influence performance in such a setup are also studied. The adoption of DSs in place of TSs in fog networks departs from the state of the art since server disaggregation is expected to improve efficiency. In spite of expected improvement, it is also important to evaluate how adoption of DS in fog networks would affect the performance of metro-access network. This is because metro-access network provides the enabling connectivity between federated fog nodes that form fog networks. Furthermore, the consideration of different classes of delay sensitive applications allows a comprehensive study that can inform the development of effective placement strategies for diverse interactive fog applications in fog networks.

In contrast to [13] where the authors focused on the design of an energy efficient network for DCs with disaggregated servers, we do not consider DC networks in this paper to simplify our evaluation scenario. Moreover, in previous works [30], [42], we formulated a MILP model that placed workloads energy efficiently in composable DC networks. Therefore, this paper focuses on the application of the resource disaggregation concept in the fog computing layer of the cloud of things architecture. Furthermore, we adopt logical disaggregation of TSs as a representation of the resource disaggregation concept. Table I gives a comparison of the contributions of this paper with those of other related works in the literature.

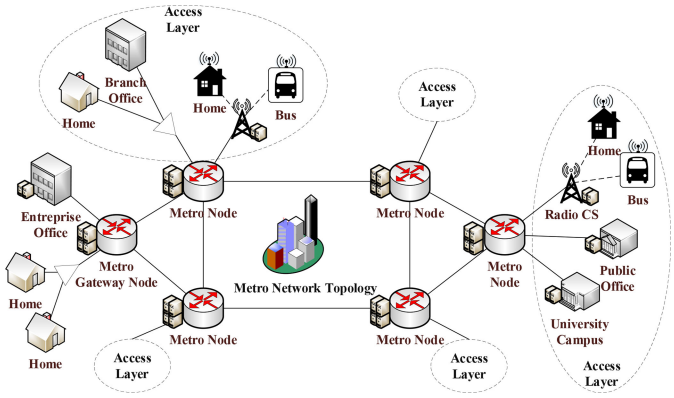


Fig. 1. Metro fog network.

III. MILP MODEL FOR FOG APPLICATIONS PLACEMENT

A. System Setup

Only functions of emerging fog applications that can be performed in the fog computing tier, such as data pre-processing, aggregation, and intelligence (e.g., facial recognition) are considered. Distributed fog nodes in selected locations are federated by linking fog computing capacity together via metro-access network to create a pool of computing capacity in the fog network. The federated fog nodes include specialized wired/wireless network equipment in centralized locations which can support generic application. Additionally, existing computing infrastructure owned by enterprises and public institutions, as shown in Fig. 1, are also included to the metro fog network. The preferred fog computing sites support mission critical traditional applications, i.e., virtual machine (VM) and/or virtual network functions (VNF) required by their owners. Each traditional app (TA) is associated with specific fog computing sites in the network topology and the spare computing resource capacity in such sites is made available in the pool of federated fog computing capacity.

Furthermore, a scenario where the fog computing layer must process all delay sensitive application is considered. This is because such applications cannot be supported by the centralized cloud computing architecture. If a delay sensitive application is not provisioned in the fog network, a local fog node must be provisioned at the source of the request for that application. Every provisioned instance of any fog application leads to corresponding pre-processing and post-processing traffic in the network. Regular network traffic from (to) each

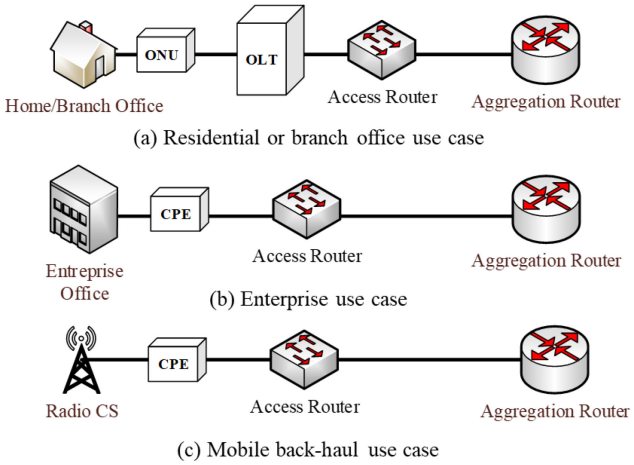


Fig. 2. Access layer use cases and network architectures in metro fog network.

access node comprises of traditional network traffic and the traffic from (to) other applications which are processed in centralized cloud computing node. We only consider delay in metropolitan area network (MAN) and access network by evaluating total delay as a sum of propagation delay and congestion delay experienced on each link in the network topology.

A scenario where fog applications traffic between a fog computing site and end-users follows a single path is considered to simplify delay calculations. Such single low latency path is provisioned by the network service provider to support instances of interactive fog apps created in the fog network. Similar to the work of the authors in [41], a maximum delay threshold is adopted for interactive fog apps during delay-aware placement. Two classes of emerging fog apps are considered. The first class comprises mission critical fog applications such as vehicle-to-everything and industrial process control which require sub-milliseconds (end-to-end) delay, and at most a delay of 20 ms or less. The second class comprises emerging fog apps which are moderately sensitive to delay and can tolerate delay that is above 20 ms. As there are different use cases at the access network layer for a federation of fog nodes in the network, it is expected that the network components traversed by network traffic in the access layer will also differ according to each use case. Fig. 2 gives illustrations of such use cases and their corresponding access network architecture.

B. MILP Model Description

In this sub-section, a MILP model that efficiently assigns instances of interactive applications into distributed fog computing nodes within a MAN topology is presented. The model minimizes network power consumption, fog computing power consumption, resulting power consumption of rejected fog applications and the approximated total queuing delay incurred in the network. Given:

i A MAN topology that comprises sets of metro and access network nodes and corresponding interconnecting physical link capacities as illustrated in Fig. 1.

ii The availability of fog computing capacity in selected fog sites/network nodes in the MAN topology.

iii The locations of clusters of end-users/IoT-devices with explicit demand for an instance of fog applications.

The MILP model determines the number of instances of each fog app that can be provisioned and the optimal location of each provisioned instance while enforcing defined constraints. The data traffic at a given node is proportional to the number of users in that node. Hence, both pre-processing and post processing data traffic of each fog app instance are defined in Gbps per user. Furthermore, only one instance of a given fog app is allocated to all users of that fog app in each access node. The model parameters are given in Table II, Table III, and Table IV. The model variables are given in Table V. Linear approximations are made as required to ensure linearity.

Delay related variables in the MILP model are derived as follows:

$$PD_{mn}^{sdec} = PD_{mn}\mathbb{H}_{mn}^{sdec}$$
$$\forall s, d \in N, \forall e \in E, \forall c \in C, \forall m \in N, \forall n \in NB_m \quad (1)$$

Equation (1) gives the propagation delay experienced by flow L_{sdec}on physical link (m, n) on the path selected for the flow.

$$WL_{sdec} = \sum_{m \in N} \sum_{n \in NB_m} \left(TL_{mn}^{sdec} + PD_{mn}^{sdec} \right)$$
$$\forall s, d \in N, \forall e \in E, \forall c \in C \quad (2)$$

Equation (2) gives the total delay experienced by flow L_{sdec} on all physical links, it is a sum of congestion delay and propagation delay experienced on all physical links on the path.

$$RD_{sdec} = WL_{sdec} + WL_{dsec}$$
$$\forall s, d \in N, \forall e \in E, \forall c \in C \quad (3)$$

Equation (3) gives the round-trip delay experienced between a node with users of an emerging fog app and the network node hosting the instance assigned to the users at that node.

$$TQ = \sum_{m \in N} \sum_{n \in NB_m} \mathbb{W}_{mn} \quad (4)$$

Equation (4) gives the approximated total delay experienced on all physical links of the network topology.

The following equations present the derivation of variables that aid the calculation of network traffic and power consumption.

$$\varphi_{esa} = \sum_{c \in C} \sum_{x \in A} \aleph_{eca} CN_{cx} AM_{xs}$$
$$\forall s \in N, \forall a \in AN, \forall e \in E \quad (5)$$

Equation (5) gives the variable φ_{esa} which depends on the placement of emerging fog application. A scenario where the compute capacity of an instance of emerging fog app $e \in E$ is greater than the maximum compute capacity required by the

TABLE II
LIST OF MILP MODEL NETWORK SETS AND PARAMETERS

N	Set of all network nodes
NB_m	Set of neighbor nodes of network node $m \in N$
U	Set of metro network nodes, $U \subseteq N$
U_m	Set of neighbor metro nodes of metro node $m \in U$
G	Set of gateway nodes in metro network topology, $G \subseteq U$
AN	Set of access network nodes $AN \subseteq N$
AN_a	Set of metro nodes that are neighbors of access network node $a \in AN$
AC_a	$AC_a = 1$ if access network node $a \in AN$ has a consumer premises equipment. Otherwise, $AC_a = 0$.
AP_a	$AP_a = 1$ if access network node $a \in AN$ has a PON ONU. Otherwise, $AP_a = 0$.
PL_{mn}	Bandwidth of physical link (m,n) $m \in N, n \in NB_m$
Δ_{an}	$\Delta_{an} = 1$ if node $n \in N$ is an access network node $a \in AN$. Otherwise, $\Delta_{an} = 0$
RT_{mn}	Regular traffic on physical link (m,n) $m \in N, n \in NB_m$
PD_{mn}	Propagation delay on physical link (m,n) $m \in N, n \in NB_m$
LP_{mn}	Set of linear pieces (linear approximations) used to linearize the delay curve of the delay experienced on link (m,n) $m \in N, n \in NB_m$
∇_{mnq}	Rate of linear piece $q \in LP_{mn}$ of the linear approximation of the delay experienced on link (m,n) $m \in N, n \in NB_m$
ζ_{mnq}	Intercept of linear piece $q \in LP_{mn}$ of the linear approximation of the delay experienced on link (m,n) $m \in N, n \in NB_m$
LU_{mn}	Upper bound of queuing delay experienced on link (m,n) $m \in N, n \in NB_m$
CE	Consumer premises equipment power consumption
MA	Metro Ethernet access switch energy per bit
MG	Metro Ethernet aggregation switch energy per bit
NU	PON ONU power consumption
OL	PON OLT energy per bit
δ	Queuing penalty of the network topology in Watt per second.

cluster of users of that app in all access node is considered. Hence, $\varphi_{esa} > 1$ is avoided.

$$\mathbb{y}_{sdec} = \sum_{x \in A} \sum_{a \in AN} \mathbb{x}_{eca} FU_e UA_{ea} \Delta_{as} CN_{cx} AM_{xd} \quad \forall s, d \in N, \forall e \in E, \forall c \in C \tag{6}$$

Equation (6) gives the pre-processing traffic between users in access nodes and instances of emerging fog applications placed in fog computing sites.

$$\mathbb{z}_{sdec} = \sum_{x \in A} \sum_{a \in AN} \mathbb{x}_{eca} FD_e UA_{ea} \Delta_{ad} CN_{cx} AM_{xs} \quad \forall s, d \in N, \forall e \in E, \forall c \in C \tag{7}$$

TABLE III
LIST OF MILP MODEL FOG APPLICATIONS SETS AND PARAMETERS

F	Set of fog apps
T	Set of traditional fog apps, $T \subseteq F$
E	Set of emerging fog apps, $E \subseteq F$
FC_f	Compute resource demand of fog app $f \in F$
FM_f	Memory resource demand of fog app $f \in F$
FS_f	Storage resource demand of fog app $f \in F$
FU_e	Uplink data rate per user of emerging fog app $e \in E$
FD_e	Downlink data rate per user of emerging fog app $e \in E$
TS_{tn}	$TS_{tn} = 1$ if traditional fog app $t \in T$ is associated with network node $n \in N$. Otherwise, $TS_{tn} = 0$
EG_{en}	$EG_{en} = 1$ if node $n \in G$ is the gateway node of emerging fog app $e \in E$. Otherwise, $EG_{en} = 0$
EA_{ea}	$EA_{ea} = 1$ if users in node $a \in AN$ make request for an instance of emerging fog app $e \in E$. Otherwise, $EA_{ea} = 0$.
UA_{ea}	Number of users in node $a \in AN$ requesting an instance of emerging fog app $e \in E$.
ED_e	Emerging fog app $e \in E$ maximum delay threshold.
γ	A unitless cost coefficient of the power consumed as a result of rejecting traditional fog apps.
$\emptyset$	A unitless cost coefficient of the power consumed as a result of rejecting emerging fog apps.

Equation (7) gives the post-processing traffic between users in access nodes and instances of emerging fog applications placed in fog computing sites.

$$L_{sdec} = \mathbb{y}_{sdec} + \mathbb{z}_{sdec} \quad \forall s, d \in N, \forall e \in E, \forall c \in C \tag{8}$$

Equation (8) gives the total traffic between a pair of nodes due to the creation of an instance of an emerging fog app in the fog network.

$$\mathbb{k}_{sd} = \sum_{a \in AN} \sum_{e \in E} \varphi_{esa} FD_e UA_{ea} EG_{ed} \quad \forall s \in N, \forall d \in G \tag{9}$$

Equation (9) gives the post-processing traffic between instances emerging fog applications placed in fog computing sites and gateway metro nodes.

$$\Lambda_{mn} = \sum_{s \in N} \sum_{d \in N} \sum_{e \in E} \sum_{c \in C} \mathcal{H}_{mn}^{sdec} \quad \forall m \in N, \forall n \in NB_m \tag{10}$$

Equation (10) gives the delay sensitive traffic routed over a physical link by summing all the delay sensitive flows over the link.

$$\lambda_{mn} = \sum_{s \in N} \sum_{d \in N} \mathbb{h}_{mn}^{sd} + RT_{mn} \quad \forall m \in N, \forall n \in NB_m \tag{11}$$

TABLE IV
LIST OF MILP MODEL FOG COMPUTING NODES SETS AND PARAMETERS

C	Set of CPU resource components.
M	Set of memory resource components.
S	Set of storage resource components.
$\mathbb{C}_j$	Capacity of CPU component $j \in C$
IC	Idle power consumption as a fraction of the maximum CPU power consumption.
CP_j	Maximum power consumption of CPU $j \in C$
ΔC_j	Power factor of CPU $j \in C$; $\Delta C_j = \frac{CP_j - IC \cdot CP_j}{\mathbb{C}_j}$
$\mathbb{M}_j$	Capacity of memory component $j \in M$
IM	Idle power consumption as a fraction of the maximum memory power consumption.
MP_j	Maximum power consumption of memory $j \in M$
ΔM_j	Power factor of memory $j \in M$; $\Delta M_m = \frac{MP_j - IM \cdot MP_j}{\mathbb{M}_j}$
$\mathbb{S}_j$	Capacity of storage component $j \in S$
IS	Idle power consumption as a fraction of the maximum storage power consumption.
SP_j	Maximum power consumption of storage $j \in S$
ΔS_j	Power factor of storage $j \in S$; $\Delta S_j = \frac{SP_j - IS \cdot SP_j}{\mathbb{S}_j}$
$M\Delta C$	Power factor of CPU component with highest power consumption.
CPM	Maximum power consumption of CPU component with highest power consumption.
$M\Delta M$	Power factor of memory component with highest power consumption.
MPM	Maximum power consumption of memory component with highest power consumption.
$M\Delta S$	Power factor of storage component with highest power consumption.
SPM	Maximum power consumption of storage component with highest power consumption.
A	Set of computing nodes
CN_{cx}	$CN_{cx} = 1$ if CPU component $c \in C$ is placed in computing node $x \in A$. Otherwise, $CN_{cx} = 0$.
MN_{mx}	$MN_{mx} = 1$ if memory component $m \in M$ is placed in computing node $x \in A$. Otherwise, $MN_{mx} = 0$.
SN_{sx}	$SN_{sx} = 1$ if storage component $s \in S$ is placed in compute node $x \in A$. Otherwise, $SN_{sx} = 0$.
AM_{xn}	$AM_{xn} = 1$ if compute node $x \in A$ is placed in network node $n \in N$. Otherwise, $AM_{xn} = 0$.
Q	A large enough number.

Equation (11) gives the delay tolerant traffic routed over a physical link by summing all the delay tolerant flows over the link with the given regular traffic on that physical link.

$$\Gamma_{mn} = \Lambda_{mn} + \lambda_{mn} \quad \forall m \in N, \forall n \in NB_m \tag{12}$$

TABLE V
LIST OF MILP MODEL VARIABLES

c_{fc}	$c_{fc} = 1$ if an instance of fog app $f \in F$ is in CPU component $c \in C$. Otherwise, $c_{fc} = 0$.
m_{fm}	$m_{fm} = 1$ if an instance of fog app $f \in F$ is in memory component $m \in M$. Otherwise, $m_{fm} = 0$.
s_{fs}	$s_{fs} = 1$ if an instance of fog app $f \in F$ is in storage component $s \in S$.Otherwise, $s_{fs} = 0$.
$\mathbb{c}_c$	$\mathbb{c}_c = 1$ if CPU $c \in C$ is active. Otherwise, $\mathbb{c}_c = 0$.
$\mathbb{m}_m$	$\mathbb{m}_m = 1$ if memory $m \in M$ is active. Otherwise, $\mathbb{m}_m = 0$.
$\mathbb{s}_s$	$\mathbb{s}_s = 1$ if storage $s \in S$ is active. Otherwise, $\mathbb{s}_s = 0$.
$\mathbb{x}_{eca}$	$\mathbb{x}_{eca} = 1$ if the instance of emerging fog app $e \in E$ in CPU component $c \in C$ is allocated to users in access node $a \in AN$. Otherwise, $\mathbb{x}_{eca} = 0$.
$\mathbb{v}_{ea}$	$\mathbb{v}_{ea} = 1$ if the emerging app $e \in E$ requested by node $a \in AN$ has been provisioned. Otherwise, $\mathbb{v}_{ea} = 0$.
φ_{esa}	$\varphi_{esa} \geq 1$ if an instance of emerging fog app $e \in E$ in node $s \in N$ is allocated to users of that app in access node $a \in AN$. Otherwise, $\varphi_{esa} = 0$.
$\mathbb{k}_{sd}$	Post-processing traffic from instances of all fog apps in network node $s \in N$ to gateway node $d \in G$.
$\mathbb{y}_{sdec}$	Pre-processing traffic from users in node $s \in N$ to node $d \in N$ that hosts an instance of emerging fog app $e \in E$ placed in CPU component $c \in C$ in node $d \in N$.
$\mathbb{z}_{sdec}$	Post-processing traffic from compute node $s \in N$ to users in node $d \in N$. Node $s \in N$ hosts an instance of emerging fog app $e \in E$ placed in CPU component $c \in C$. The instance of emerging fog app $e \in E$ placed in CPU component $c \in C$ was allocated to users in node $d \in N$.
L_{sdec}	Traffic from node $s \in N$ to node $d \in N$ due to the presence of emerging fog app $e \in E$ in CPU component $c \in C$.
$\mathbb{h}_{mn}^{sd}$	Volume of $\mathbb{k}_{sd}$ traffic routed on physical link (m, n)
λ_{mn}	Volume of cloud bound traffic on physical link (m, n).

(*continued*)

Equation (12) gives the total traffic over a link by summing delay sensitive and delay tolerant traffic routed over the link.

Total network power consumption *TNPC* is given by:

$$TNPC = ANPC + MNPC \tag{13}$$

MNPC is the total metro network power consumption and is given by

$$MNPC = \sum_{m \in U} (\mathbb{u}_m + \mathbb{g}_m + \mathbb{q}_m) MG \tag{14}$$

TABLE V
(Continued.) LIST OF MILP MODEL VARIABLES

$\mathcal{H}_{mn}^{sdec}$	Flow of latency sensitive traffic (emerging fog applications traffic) L_{sdec} on physical link (m,n)
$\mathbb{H}_{mn}^{sdec}$	$\mathbb{H}_{mn}^{sdec}=1$ if a flow of L_{sdec} is present on physical link (m,n). Otherwise $\mathbb{H}_{mn}^{sdec}=0$.
Λ_{mn}	Volume of latency sensitive traffic on physical link (m,n)
Γ_{mn}	Total traffic on physical link (m,n)
$\mathbb{t}_t$	State of traditional fog app $t \in T$.
$TCRTA$	Total cost of rejected traditional fog apps in Watt.
$TCREA$	Total cost of rejected emerging fog app in Watt.
α_t	Power penalty as a result of rejecting traditional fog app $t \in T$ in Watt.
β_e	Power penalty as a result of rejecting emerging fog app $e \in E$ in Watt.
$\mathbb{W}_{mn}$	M/M/1 queuing delay experienced on physical link (m,n)
TL_{mn}^{sdec}	TL_{mn}^{sdec} is the queuing delay experienced by flow L_{sdec} on physical link (m,n) on the path selected for the flow.
PD_{mn}^{sdec}	PD_{mn}^{sdec} is the propagation delay experienced by flow L_{sdec} on physical link (m,n) on the path selected for the flow.
WL_{sdec}	Total delay of flow L_{sdec} on all physical links. Sum of congestion in physical links (queuing delay) and propagation delay on physical links on the path.
RD_{sdec}	Round trip delay between a node containing users of an emerging fog app and the network node hosting the instance assigned to the users.
TQ	Approximated total queuing delay experienced on physical links of the network topology.

where $\mathbb{u}_m$ is the traffic relayed by a metro node m and is given by

$$\begin{aligned}\mathbb{u}_m = &\sum_{s\in N}\sum_{d\in N}\sum_{n\in NB_m}\mathbb{h}_{mn}^{sd} \\ &+\sum_{n\in NB_m}\sum_{s\in N}\sum_{d\in N}\sum_{e\in E}\sum_{c\in C}\mathcal{H}_{mn}^{sdec} \\ &\forall m\in U, s,d\in N,\ s\neq m,\ d\neq m\end{aligned} \tag{15}$$

$\mathbb{g}_m$ is the traffic received by a metro node m and is given by

$$\begin{aligned}\mathbb{g}_m = &\sum_{s\in N}\sum_{n\in NB_m}\mathbb{h}_{nm}^{sm} \\ &+\sum_{s\in N}\sum_{n\in NB_m}\sum_{e\in E}\sum_{c\in C}\mathcal{H}_{nm}^{smec} \\ &+\sum_{n\in NB_m} RT_{nm} \\ &\forall m\in U\end{aligned} \tag{16}$$

and finally, $\mathbb{q}_m$ is the traffic transmitted by a metro node m and is given by

$$\begin{aligned}\mathbb{q}_m = &\sum_{d\in G}\sum_{n\in NB_m}\mathbb{h}_{mn}^{md} \\ &+\sum_{d\in N}\sum_{n\in NB_m}\sum_{e\in E}\sum_{c\in C}\mathcal{H}_{mn}^{mdec} \\ &+\sum_{n\in NB_m} RT_{mn} \\ &\forall m\in U\end{aligned} \tag{17}$$

ANPC is the total access network power consumption and is given by

$$\begin{aligned}ANPC = &\sum_{a\in AN}(CE\ AC_a + NU\ AP_a) \\ &+\sum_{a\in AN}\sum_{m\in AN_a}(\Gamma_{am}+\Gamma_{ma})(MA + AP_a\ OL)\end{aligned} \tag{18}$$

The total fog computing power consumption $(TFPC)$ is given by:

$$TFPC = TCPC + TMPC + TSPC \tag{19}$$

where *TCPC* is the total power consumption of CPU resources in fog network and is given by

$$TCPC = \sum_{c\in C}\left((IC\ CP_c\mathbb{c}_c) + \sum_{f\in F}\Delta C_c\ c_{fc}FC_f\right) \tag{20}$$

TMPC is the total power consumption of memory resources in fog network and is given by

$$TMPC = \sum_{m\in M}\left((\mathrm{IM}\ MP_m\mathbb{m}_m) + \sum_{f\in F}\Delta M_m\ m_{fm}FM_f\right) \tag{21}$$

and *TSPC* is the total power consumption of storage resources in fog network and is given by

$$TSPC = \sum_{s\in S}\left((\mathrm{IS}\ SP_s\mathbb{s}_s) + \sum_{f\in F}\Delta S_s\ s_{fs}FS_f\right) \tag{22}$$

The total cost of rejected traditional fog apps in the distributed fog network is given by

$$TCRTA = \sum_{t\in T}\alpha_t \tag{23}$$

where

$$\begin{aligned}\alpha_t = &((IC\ CPM + M\Delta CFC_t) \\ &+ (IM\ MPM + M\Delta M\ FM_t) \\ &+ (IS\ SPM + M\Delta SFD_t))\mathbb{t}_t \\ &\forall t\in T\end{aligned} \tag{24}$$

where the state of traditional application t is given by

$$\begin{aligned}\mathbb{t}_t = &\sum_{c\in C}(1-c_{tc}) \\ &\forall t\in T\end{aligned} \tag{25}$$

The total cost of rejected emerging fog apps in the distributed fog network is given by

$$TCREA = \sum_{e\in E}\beta_e \tag{26}$$

where

$$\beta_e = ((IC\ CPM + M\Delta C\ FC_e) + (IM\ MPM + M\Delta M\ FM_e) + (IS\ SPM + M\Delta S\ FS_e)) \sum_{a\in AN} (1 - \mathbb{v}_{ea})\ UA_{ea} \quad \forall e \in E \tag{27}$$

where $\mathbb{v}_{ea}$ indicates if the emerging application e requested by node a has been provisioned or not, and is given by

$$\mathbb{v}_{ea} = \sum_{c\in C} \mathbb{x}_{eca} \quad \forall e \in E, \forall a \in AN \tag{28}$$

It is important to note that the cost of rejecting an app is defined as the maximum power consumed if it is accepted, i.e., the power consumption in the case where inactive components must be turned-on to support the app.

The model is defined as follows:

Objective: Minimize

$$TNPC + TFPC + \gamma TCRTA + \emptyset TCREA + \delta TQ \tag{29}$$

The objective of the model is to minimize a weighted sum of the TNPC, TFPC, the total cost of rejected traditional, emerging fog applications and the cost of approximated total delay in the network as given by the expression in (29). Note that both traditional and emerging fog apps are provisioned or rejected based on the cost coefficient of terms in the objective. The unit of the terms in the objective function and their corresponding cost-coefficients jointly ensure a consistent unit (i.e., Watt) for the sum of terms in the objective function. Setting γ to a high value ensures that TCRTA is significantly higher than TCREA. Hence, higher priority is given to provisioning traditional fog apps in the objective function. $\emptyset$ may also be varied to increase or decrease the cost of rejected emerging fog apps. By modelling the TCRTA and TCREA, the additional power that will be consumed by provisioning traditional apps and emerging fog apps respectively with dedicated fog servers are modelled as cost of apps rejection. The value of δ dictates the weight of approximated total queuing delay in the objective function. $\delta \ll 1$ represents a network with trivial queuing delay penalty while $\delta \gg 1$ represents a network with significant queuing delay penalty.

Subject to the following constraints:

Fog DC related constraints

$$\sum_{c\in C} c_{fc} = \sum_{m\in M} m_{fm} \quad \forall f \in F \tag{30}$$

$$\sum_{c\in C} c_{fc} = \sum_{s\in S} s_{fs} \quad \forall f \in F \tag{31}$$

Constraints (30) and (31) ensure that the number of instances of CPU resources provisioned for any (traditional or emerging) fog app is equal to the number of instances of memory and storage resources provisioned for that app across the distributed fog sites.

$$\sum_{c\in C} c_{fc} CN_{cx} = \sum_{m\in M} m_{fm} MN_{mx} \quad \forall f \in F, \forall x \in A \tag{32}$$

$$\sum_{c\in C} c_{fc} CN_{cx} = \sum_{s\in S} s_{fs} SN_{sx} \quad \forall f \in F, \forall x \in A \tag{33}$$

Constraints (32) and (33) are the locality constraints when the traditional server architecture is adopted in compute nodes. They ensure that the CPU, memory, and storage components used to provision a given instance of a fog app are in the same compute nodes.

$$\sum_{x\in A}\sum_{c\in C} c_{fc} CN_{cx} AM_{xn} = \sum_{x\in A}\sum_{m\in M} m_{fm} MN_{mx} AM_{xn} \quad \forall f \in F, \forall n \in N \tag{34}$$

$$\sum_{x\in A}\sum_{c\in C} c_{fc} CN_{cx} AM_{xn} = \sum_{x\in A}\sum_{s\in S} s_{fs} SN_{sx} AM_{xn} \quad \forall f \in F, \forall n \in N \tag{35}$$

Constraints (34) and (35) are the locality constraints when the disaggregated server architecture is adopted in compute nodes. They ensure that the CPU, memory, and storage components used to provision a given instance of a fog app are in the same fog computing site (network node) but not necessarily in the same compute node.

$$\sum_{x\in A}\sum_{c\in C} c_{tc} CN_{cx} AM_{xn} = TS_{tn} \quad \forall t \in T, \forall n \in N \tag{36}$$

Constraint (36) is the workload locality constraint for traditional fog apps associated with a given network node.

$$\sum_{c\in C} c_{fc} CN_{cx} \leq 1 \quad \forall f \in F, \forall x \in A \tag{37}$$

$$\sum_{m\in M} m_{fm} MN_{mx} \leq 1 \quad \forall f \in F, \forall x \in A \tag{38}$$

$$\sum_{s\in S} s_{fs} SN_{sx} \leq 1 \quad \forall f \in F, \forall x \in A \tag{39}$$

Constraints (37) - (39) are SLA constraints which ensure robustness of the fog network. They ensure that only an instance of fog app f is provisioned within a given compute node. Hence, the impact of a compute node failure is minimized for a fog app with multiple instances.

$$\sum_{f\in F} FC_f c_{fc} \leq \mathbb{C}_c \quad \forall c \in C \tag{40}$$

$$\sum_{f\in F} FM_f m_{fm} \leq \mathbb{M}_m \quad \forall m \in M \tag{41}$$

$$\sum_{f\in F} FS_f s_{fs} \leq \mathbb{S}_s \quad \forall s \in S \tag{42}$$

Constraints (40) - (42) denote resource capacity constraints for each CPU, memory, and storage component in the fog network. They ensure that each resource component's capacity across all fog computing sites is not exceeded.

$$\sum_{f\in F} c_{fc} \geq \mathbb{c}_c \quad \forall c \in C \tag{43}$$

$$\sum_{f\in F} c_{fc} \leq \mathcal{Q}\mathbb{c}_c \quad \forall c \in C \tag{44}$$

$$\sum_{f\in F} m_{fm} \geq \mathbb{m}_m \quad \forall m \in M \tag{45}$$

$$\sum_{f\in F} m_{fm} \leq \mathcal{Q}\mathbb{m}_m \quad \forall m \in M \tag{46}$$

$$\sum_{f\in F} s_{fs} \geq \mathbb{s}_s \quad \forall s \in S \tag{47}$$

$$\sum_{f\in F} s_{fs} \leq \mathcal{Q}\mathbb{s}_s \quad \forall s \in S \tag{48}$$

Constraints (43) - (48) derive the state of CPU, memory, and storage resources components across all fog computing sites.

Fog app instance related constraints

$$\sum_{c\in C} \mathbb{x}_{eca} \leq 1 \quad \forall e \in E, \forall a \in AN \tag{49}$$

Constraint (49) ensures that the cluster of users requesting an emerging fog app is assigned at most one instance of that application.

$$\sum_{a\in AN} \mathbb{x}_{eca} \geq c_{ec} \quad \forall c \in C,\ e \in E \tag{50}$$

$$\sum_{a\in AN} \mathbb{x}_{eca} \leq \mathcal{Q}c_{ec} \quad \forall c \in C,\ e \in E \tag{51}$$

Constraints (50) and (51) ensure that each instance of an emerging fog app in a CPU component is allocated to one or more user clusters in access nodes. Otherwise, the instance should not be created.

$$\mathbb{x}_{eca} = EA_{ea}\mathbb{v}_{ea}\, c_{ec} \quad \forall c \in C,\ e \in E,\ a \in AN \tag{52}$$

$$\mathbb{x}_{eca} \leq EA_{ea}\, c_{ec} \quad \forall c \in C,\ e \in E,\ a \in AN \tag{53}$$

$$\mathbb{x}_{eca} \leq EA_{ea}\mathbb{v}_{ea} \quad \forall c \in C,\ e \in E,\ a \in AN \tag{54}$$

$$\mathbb{x}_{eca} \geq EA_{ea}(\mathbb{v}_{ea} + c_{ec}) - 1 \quad \forall c \in C,\ e \in E,\ a \in AN \tag{55}$$

Constraint (52) derives $\mathbb{x}_{eca}$ which gives the instance of an emerging fog app in a CPU that is assigned to users of that fog app in an access node. $\mathbb{x}_{eca} = 1$ if and only if users of an emerging fop app are present in an access node, an instance of that fog app is in a CPU component and that instance has been assigned to users of that fog app in the corresponding access node. Constraints (53) - (55) implement linearly the product of parameter and variables given in Constraint (52).

Network related constraints

$$\sum_{n\in NB_m} \mathbb{h}^{sd}_{mn} - \sum_{n\in NB_m} \mathbb{h}^{sd}_{nm} = \begin{cases} \mathbb{k}_{sd} & m = s \\ -\mathbb{k}_{sd} & m = d \\ 0 & otherwise \end{cases} \quad \forall s, m \in N, \forall d \in G : s \neq d \tag{56}$$

Constraint (56) enforces flow conservation for post-processing traffic to the cloud in the physical layer of the network.

$$\sum_{n\in NB_m} \mathcal{H}^{sdec}_{mn} - \sum_{n\in NB_m} \mathcal{H}^{sdec}_{nm} = \begin{cases} L_{sdec} & m = s \\ -L_{sdec} & m = d \\ 0 & otherwise \end{cases} \quad \forall s, d, m \in N, e \in E, c \in C : s \neq d \tag{57}$$

Constraint (57) enforces flow conservation for delay sensitive flows in the physical layer of the network.

$$\mathcal{H}^{sdec}_{mn} \geq \mathbb{H}^{sdec}_{mn} \quad \forall s, d, m, n \in N, e \in E, c \in C : s \neq d \tag{58}$$

$$\mathcal{H}^{sdec}_{mn} \leq \mathcal{Q}\ \mathbb{H}^{sdec}_{mn} \quad \forall s, d, m, n \in N, e \in E, c \in C : s \neq d \tag{59}$$

Constraints (58) and (59) give the binary equivalent of $\mathcal{H}^{sdec}_{mn}$.

$$\sum_{n\in NB_m} \mathbb{H}^{sdec}_{mn} \leq 1 \quad \forall s, d, m \in N, e \in E, c \in C : s \neq d \tag{60}$$

Constraint (60) ensures that the flow L_{sdec} is not bifurcated over multiple paths.

$$\Gamma_{mn} \leq PL_{mn} \quad \forall m \in N,\ n \in NB_m \tag{61}$$

Constraint (61) enforces capacity constraint on each physical link (m, n).

Network delay related constraints

$$TL^{sdec}_{mn} = \mathbb{W}_{mn}\mathbb{H}^{sdec}_{mn} \quad \forall s, d, m \in N, n \in NB_m, e \in E, c \in C : s \neq d \tag{62}$$

$$TL^{sdec}_{mn} \leq LU_{mn}\mathbb{H}^{sdec}_{mn} \quad \forall s, d, m \in N, n \in NB_m, e \in E, c \in C : s \neq d \tag{63}$$

$$TL^{sdec}_{mn} \leq \mathbb{W}_{mn} \quad \forall s, d, m \in N, n \in NB_m, e \in E, c \in C : s \neq d \tag{64}$$

$$TL_{mn}^{sdec} \geq \mathbb{W}_{mn} - LU_{mn}\left(1 - \mathbb{H}_{mn}^{sdec}\right)$$
$$\forall s, d, m \in N, n \in NB_m, e \in E, c \in C : s \neq d \quad (65)$$

Constraint (62) estimates the queuing delay experienced by flow L_{sdec} on physical link (*m*, *n*) which is given by the product of $\mathbb{W}_{mn}$ and $\mathbb{H}_{mn}^{sdec}$. Constraints (63) - (65) linearize Constraint (62). LU_{mn} is the upper bound of the queuing delay experienced on each physical link, it is required to ensure that the delay experienced on a physical link does not exceed a predefined threshold.

$$\mathbb{W}_{mn} \geq \nabla_{mnq}\Gamma_{mn} + \zeta_{mnq}$$
$$\forall m \in N, n \in NB_m, q \in LP_{mn} \quad (66)$$

Constraint (66) represent piecewise linear approximation of queuing delay experienced on physical link (*m*, *n*). This is required because M/M/1 delay is a non-linear function.

$$RD_{sdec} \leq ED_e$$
$$\forall s, d \in N, e \in E, c \in C : s \neq d \quad (67)$$

Constraint (67) represents the round-trip delay constraint for an emerging fog app *e*. The round-trip delay experienced by an emerging fog app *e* must not exceed the predefined threshold.

IV. Heuristic for Energy Efficient and Delay Aware Placement of Fog Applications

The complexity of the MILP model formulation increases exponentially as the size of the fog network is scaled-up. This is because MILP models are known to be NP-hard and are consequentially computationally intractable [43]. For instance, the satisfiability problem, a known NP-hard problem, is directly reducible to a special case of a MILP resource allocation problem (where the non-binary variables are replaced with binary variables) [44]. Therefore, MILP problem is also NP-hard and a fast heuristic that mimics the MILP model is more effective for large fog networks. Consequently, a heuristic that leverages a centralized orchestration and management framework for a network of distributed fog computing nodes is proposed. Centralized control is an essential tool that can enable the efficiencies that will approach exact solutions of obtained from solving a MILP model formulation. The heuristic depends on centralized control of distributed fog nodes and on global knowledge derived from control information exchange to achieve high efficiency in a fog network. The heuristic optimally provisions both mission critical traditional applications and (delay sensitive) emerging fog applications in a fog network when possible.

Given a set of input fog apps (i.e., mission critical traditional apps and emerging fog apps), the algorithm attempts to provision instance(s) of these applications in a fog network in an energy efficient manner while considering the delay requirements of each application. Applications that cannot be provisioned are rejected and users whose delay requirements are not satisfied by provisioned instance(s) are also rejected. The algorithm supports the use of TS and DS architectures in the fog network. Other inputs to the algorithm include the user distribution and delay requirement of each emerging fog app; the distribution of fog compute nodes in the network topology; the features and characteristics of each resource components at each fog site; and the load and propagation delay on each network link.

A. HEEDAP Algorithm Description

A high-level description of the heuristic for energy efficient and delay aware placement (HEEDAP) of applications in fog networks is illustrated in Fig. 3. At inception, the HEEDAP algorithm processes the set of input mission critical TAs which are associated with specific network nodes by sorting the list of TAs (in each network node) in descending order of CPU demand intensity. If a tie occurs, memory demand intensity is initially adopted to break the tie followed by storage demand intensity. The output of this process is the "local job list" (LJ-list) created at each network node.

Secondly, the HEEDAP algorithm processes the set of input emerging fog apps to the fog network in two stages. The initial stage identifies fog apps that are highly sensitive to network delay. These fog apps form the secondary job list at each fog computing site if some users of such delay sensitive fog app are directly connected to the corresponding network node via wireless media. This secondary job list created at each fog computing site is called the "pseudo-local job list" (PLJ-list). Emerging fog apps that are classified as delay sensitive are subsequently ejected from the list of input emerging fog apps. The PLJ-list at each network node is also arranged in descending order of resource demand intensity as described for the LJ-list. In the second stage, the HEEDAP algorithm sorts the list of input (moderately sensitive) emerging fog apps in descending order of resource intensity to create the "real fog job list" (RFJ-list). The RFJ-list also (implicitly) holds information about the number of users at each network node which is a source of any request made for each emerging fog app. After input apps processing, HEEDAP creates a temporary copy of the RFJ-list. This temporary copy is the "pseudo fog job list" (PFJ-list) which is refreshed after each complete iteration of the algorithm. It is important to note that an iteration of the HEEDAP algorithm is complete when the PFJ-list of that iteration is empty. Furthermore, a union of LJ-list and PLJ-list at each network node and the RFJ-list form the list of applications in the fog network.

Whilst the list of applications in the fog network is not empty (this is the first check of the HEEDAP algorithm), at each network node with compute capacity, the mission critical traditional fog app at the top of the LJ-list is set as a "query app". The query app at each network node is placed energy-efficiently; new components may be activated to support the query app as required. If the query app could not be placed, it is rejected and removed from the LJ-list. The state of all compute components in each network node is recorded and stored by the central orchestrator. Thereafter, at each network node, a candidate app in the LJ-list is identified to maximize the utilization of the idle resource capacity (IRC) of active components in this network node where possible. If a candidate app is not found in the LJ-list, the PLJ-list is checked for a candidate app. The search for a candidate local or pseudo-local app in each network node gives higher

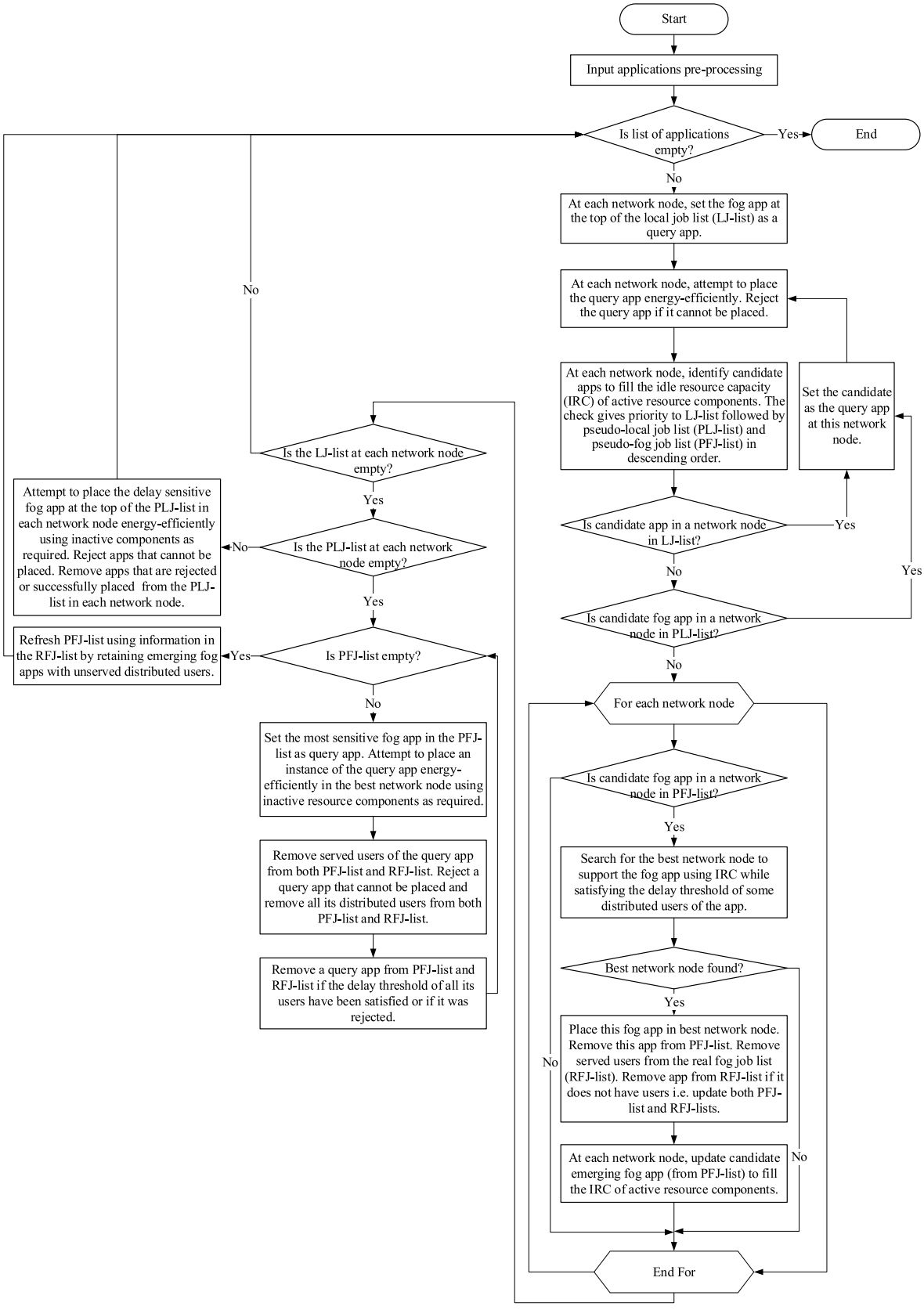


Fig. 3. Flow chart of HEEDAP algorithm.

priority to maximum utilization of the IRC of active CPU components because CPU components consume more power than memory and storage components (as illustrated later in Table VI). When the DS architecture is adopted, inactive memory and storage components within the same network node may be used to complement available IRC of CPU

component. A similar approach may be adopted when the TS architecture is deployed and a single compute node has multiple intrinsic CPU, memory, and storage components; otherwise, the resource locality constraint of TS architecture is enforced.

If a candidate app is not found in the LJ-list or PLJ-list of a network node, the PFJ-list is searched to identify a moderately sensitive emerging fog app that can maximize utilization of the IRC of active resource components at such network node. This search gives priority to the emerging fog app with more stringent delay requirement if the node is within the round-trip delay threshold of one or more unserved users of that fog app. The search conducted at each network node provides control data that supports the placement of emerging fog apps in the PFJ-list in subsequent steps of the HEEDAP algorithm. Such control information provides the global knowledge to the algorithm.

In each network node, if a candidate app is found in the LJ-list, the app is selected as the new query app and placed energy-efficiently. Hence, the algorithm gives higher priority to mission critical traditional fog apps of the fog computing infrastructure provider. Otherwise, if a candidate app is found in the PLJ-list, the app is also selected as the new query app and is placed energy-efficiently. Relative to moderately sensitive emerging fog apps, this gives greater priority to emerging fog apps that have greater delay sensitivity. On the other hand, if a candidate app is not found in LJ-list or PLJ-list in each network node and one is found in the PFJ-list, the algorithm checks if other network nodes can also host this candidate emerging fog app using IRC before a best network node is selected. Thus, global knowledge of the central orchestrator must be consulted before the best network node is selected from the list of all contending network nodes that can host the candidate moderately sensitive emerging fog app using IRC.

The best network node for a given moderately sensitive emerging fog app in the PFJ-List is the network node that leads to the smallest increase in TFPC after the placement of the candidate emerging fog app, i.e., the most energy efficient network node. However, ties may occur when energy efficiency is used as the decision metric. Hence, the CO that is closest to the metro gateway node is given the highest priority when a tie occurs. This ensures that the (number of hops traversed or) traffic in the network is minimized. Furthermore, if the list of contending network nodes for the emerging fog app comprise of only access network nodes, the network node with the highest user density is given higher priority. Otherwise, if some contending network nodes have the same user density for the emerging fog app, then lower delay to the metro gateway node is used as the decision metric to select the best network node.

Once the best network node for a given emerging in the PFJ-list fog app is found, an instance of the app is provisioned in that network node and all users of the app that can be served by this new instance (without violating delay requirements and network capacity constraints) are removed from the RFJ-list. Furthermore, the fog app and its served users are also removed from the PFJ-list of the present iteration. Additionally, the provisioned emerging fog app is replaced as a candidate emerging fog app at all corresponding fog computing sites where it was previously a candidate by electing a new candidate fog app from the PFJ-list to maximize the utilization of the IRC of active CPU component at such network nodes. This approach ensures fair placement for each emerging fog app in the fog network. This placement strategy, which fills the IRC of active CPU components in the fog network with emerging fog apps in the PFJ-list, is repeated at all network nodes to place all elected candidate emerging fog apps in the active iteration.

After exhausting all opportunities to use the IRC to provision an instance of each candidate emerging fog app in the PFJ-list in the active iteration, the size of the LJ-list is checked. If the LJ-list is not empty, the HEEDAP algorithm repeats all procedures described above; hence, the algorithm attempts to provision all mission critical traditional apps before considering emerging fog apps for placement. On the other hand, if the LJ-list is empty, another check is made to confirm that the PLJ-list in each network node is empty. If the PLJ-list in a network node is not empty, the HEEDAP algorithm attempts to provision the fog apps at the top of the PLJ-list in each network node. Such attempts may activate inactive resource components as required since the LJ-list in the network node is now empty. Hence, after mission critical TAs, emerging fog apps with higher sensitivity to delay have the highest priority. Emerging fog apps in the PLJ-list of a network node that are placed successfully are removed from the PLJ-list of that network node. Fog apps that are rejected are also removed from the PLJ-list of the corresponding network node. The HEEDAP algorithm thereafter repeats all procedures described above to place all moderately-sensitive emerging fog apps with available IRC of active resource components. It is important to note that if both LJ-list and PLJ-list of a network node are empty, only fog apps in the PFJ-list of the active iteration will be considered for energy efficient placement using IRC as given in the previous procedures described above.

On the other hand, if the PLJ-list is empty, a new check is made to confirm if the PFJ-list of the active iteration of the fog network is empty or not. If the PFJ-list is not empty, the emerging fog app with the highest delay sensitivity in the PFJ-list is set as query app and energy-efficient placement of the query app is attempted. Inactive resource components may be activated as required to place an instance of this query app. If a query app cannot be placed, the app is deleted from the PFJ-list and RFJ-list along with the information about all un-served distributed users of that app. Otherwise, if an instance of the query app was successfully provisioned, users of the query app whose delay threshold has been satisfied by the new instance are removed from both PFJ-list and RFJ-list. To ensure fairness when placing emerging fog apps, the provisioned query app is also removed from the PFJ-list of the active iteration, but the details of unserved distributed users are kept in the RFJ-list. Thereafter, the HEEDAP algorithm repeats all previous steps in this paragraph to place one instance of each moderately sensitive emerging fog apps in the PFJ-list until the list is emptied.

An empty PFJ-list implies that an active iteration of the HEEDAP algorithm has been completed and that a single instance of each emerging fog app has been provisioned.

Since the HEEDAP algorithm uses the RFJ-list to maintain global knowledge of users of some emerging fog apps whose delay requirement remain unfulfilled, this knowledge is used to refresh PFJ-list. The refreshed PFJ-list comprise of all emerging fog apps with one or more unsatisfied users. The HEEDAP algorithm subsequently returns to the first check at the top of the algorithm to begin a new iteration since this check will be negative. However, if the delay requirements of all users of all emerging fog apps have been satisfied or the emerging fog apps has been rejected because their delay threshold could not be satisfied, the first check of the algorithm will be positive and the HEEDAP algorithm stops.

The HEEDAP algorithm calculates delay by considering the sum of the delay experienced on a link and link's propagation delay as the delay cost on each link in the network topology as given in the MILP model. The path with the smallest total delay is always selected as the shortest path between two nodes. It is assumed that the information of the network topology such as propagation delay and historical traffic (load) on each network link is available as input to the HEEDAP algorithm. Given this information, Dijkstra's shortest path algorithm is used determine the shortest path between two network nodes using total (propagation plus congestion) delay as the cost metric.

In the HEEDAP algorithm, resource locality constraint distinguishes a fog computing site with DS architecture from a fog computing site with TS architecture. To reduce the complexity of control and orchestration required for the algorithm in a large fog network deployment, i.e., big fog networks can be sub-divided into multiple small units. The algorithms can be deployed in a stand-alone mode in each small unit. Criteria for deciding the division thresholds for big fog networks include delay, number of network nodes and fog application user distribution. It is also important to note that an emerging fog app is a candidate app in a network node if and only if network capacity exists on a selected shortest path that satisfies the delay threshold of some users of that fog app after the placement of that fog app into the node. Otherwise, the fog app is not an acceptable candidate for that network node. Similarly, the users of placed fog app at a given network node are removed from the RFJ-list if and only if the delay threshold of such users are satisfied within specified link capacity constraint of links on the selected shortest path between users of that app and the network node where the instance has been provisioned.

B. HEEDAP Computational Complexity

The computational complexity of HEEDAP algorithm depends on the following input parameters.

- l The total number of unique requests for an instance of each application in the fog network.
- n The number of network nodes in the fog network.
- s The maximum number of compute nodes (servers) in any network node in the fog network.
- c The maximum number of unique compute components in any compute node in the fog network.
- x The maximum number of VMs/VNFs in the LJ-list of any network node.
- y The maximum number of fog apps in the PLJ-list of any network node.
- z The maximum number of delay tolerant emerging fog apps in the RFJ-list or PFJ-list of the fog network.

The number of unique requests for an instance of each application in the fog network is derived by (68). It depends on the number of application instance demands in FJ-list, PFJ-list and RFJ-list at each network node.

$$l = (x + y + z)n \tag{68}$$

It is important to note that the Dijkstra's shortest path algorithm which can be implemented with a computation complexity of $O(n^2)$ is always implemented by HEEDAP when an emerging fog app in PFJ-list and RFJ-list is considered for placement in any network node. In the worst case, the complexity of different segments of the HEEDAP algorithm is as follows.

- The computational complexity of placing or rejecting all requested instances of each application in the fog network is $O(l)$. This determines when HEEDAP will stop.
- The computation complexity of attempting to place or reject a query app in all network node is $O(n{\cdot}s{\cdot}3{\cdot}c)$. This is because in the worst case, all CPU, memory and storage components in each compute node of each network node must be considered before placing a query app.
- The computational complexity of attempting to find the next query app in each network node of the fog network is $O(n(x+n^2y+3n^2\cdot z))$. This is because, in the worst case, all application instance requested at each network node would be considered in the search to fill active components.
- The computational complexity of placing or rejecting a candidate query app from the PFJ-list at each network node is $O(n^2{\cdot}3n^2)$. This because a network wide search is conducted to determine the best network node for a candidate query app from the PFJ-list before the final decision is made on the placement location of an instance of the candidate query app.
- The computational complexity of searching for a new candidate query app from PFJ-list following the placement of a candidate query app from PFJ-list is $O(n^2\cdot z\cdot 3n^2)$.
- The computation complexity of placing (or rejecting) the most resource intensive application in the PFJ-list of each network node energy efficiently following the placement of all fog apps in the LJ-list is $O(n\cdot s\cdot 3\cdot c\cdot n^2)$.
- Following the placement (or rejection) of all fog apps in the FJ-list and PLJ-list in the fog network, the worst-case computation complexity of finding a suitable fog app in the PFJ-list and then placing or rejecting an instance of that app in PFJ-list is $O(z(n\cdot 3n^2+n\cdot s\cdot 3\cdot c))$.

An expression of the worst-case computational complexity of the HEEDAP algorithm is given by $O(h)$, where h is as given in (69). The expression in (69) can be further simplified as given in (70). Hence, the worst-case computational complexity of HEEDAP is as given in (70). Therefore, HEEDAP is a polynomial time algorithm. Compared to the computationally intractable MILP model formulation, HEEDAP is a

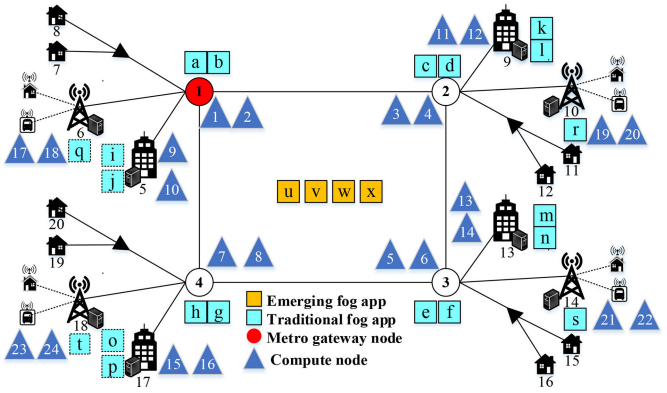


Fig. 4. Fog network system setup. The figure shows the evaluated network topology, the preferred fog computing sites, and the allocation of compute nodes to the fog sites. The figure also shows the traditional fog apps associated with each fog computing site and un-provisioned emerging fog apps.

much faster and efficient algorithm. Additionally, effective use global knowledge in the algorithm is expected to ensure lower average execution time in a practical scenario. This is because global knowledge of fog network is obtained after placing or rejecting the first application in all network nodes. Such knowledge would often facilitate rapid instantiation of application instances by HEEDAP.

$$h = l\Big((3n \cdot s \cdot c) + n\Big(x + n^2 \cdot y + 3n^2 \cdot z\Big) + \Big(3n^4\Big) + \Big(3n^4 \cdot z\Big) + \Big(3n^3 \cdot s \cdot c\Big) + \Big(3n^3 \cdot z + 3n \cdot s \cdot c \cdot z\Big)\Big) \quad (69)$$

$$h = l\Big(n\Big(x + n^2 y\Big) + \Big(n^3 sc\Big) + nz\Big(n^3 + sc\Big)\Big). \quad (70)$$

V. Evaluation and Results

A. Evaluation Scenarios and Input Parameters

The MILP model and HEEDAP algorithm are used to study the impact of adopting DS architecture in the fog computing layer of the cloud of things continuum relative to the use of TS architecture. A small network topology comprising of 4 (metro) central offices (COs) and 16 access nodes is adopted as the default evaluation scenario. This helps to minimize execution time of the MILP model which grows as the size and complexity of the problem increases. The access nodes comprise 4 radio cell sites (CSs), 4 enterprise offices (EOs), which are connected to the metro ring via 40Gbps links, and 8 homes, which are connected to the metro ring via 40Gbps Next-Generation Passive Optical Network 2 (NG-PON2) links. Connected to each metro CO are an EO, a radio CS and two residential houses as illustrated in Fig. 4. By using 40Gbps last mile links between metro and access network nodes, network bottlenecks that can lead to the rejection of emerging fog apps as reported in [15] are avoided.

To further maintain simplicity, the evaluation scenario allocates two servers (or compute nodes comprising of commodity hardware) to each fog computing sites. The fog computing sites comprise metro COs, EOs, and radio CSs in the network topology as illustrated in Fig. 4. When the TS architecture is adopted, the utilization scope of each server's intrinsic resource components is limited to that server. On the other hand, when the DS architecture is adopted, servers are logically disaggregated to expand the utilization scope of the intrinsic resources present in each server at fog computing sites. However, access to such disaggregated resource components is limited to the corresponding fog computing site hosting each component. A common configuration is adopted for all servers distributed across the metro network topology, each server comprises of one CPU, one memory and one storage components. The characteristics of each compute component used to evaluate the MILP model are given in Table VI. The power consumption profile of each compute component comprises of an idle portion and another portion that is linearly dependent on the component's utilization. Servers are not allocated to residential houses when fog computing sites are federated to form a fog network. On the other hands, in the absences of fog federation, each fog computing apps must be instantiated at the location (including residential houses) of the users.

TABLE VI
Component Capacity and Power Features

Parameter	Value
CPU capacity, CPU peak power, and idle fraction of peak CPU power	3.6 GHz, 130 W and 0.7
Memory capacity, memory peak power, and idle fraction of peak memory power	32 GB, 11.85 W and 0.7
Storage capacity, storage peak power, and idle fraction of peak storage power	320 GB, 6.19 W and 0.7
Metro Ethernet CPE (on-off)	75 W [46]
Metro Ethernet aggregation router (load proportional)	0.9 W/Gb [47]
Metro Ethernet access router (load proportional)	0.243 W/Gb
PON optical line terminal (load proportional)	1.75 W/Gb
PON optical network unit (on-off)	15 W [48]

Each designated fog computing site within the federation of fog computing nodes has one or two in-situ VM/VNFs for mission critical traditional applications (TA), as illustrated in Fig. 4, which must be provisioned at the node. Each CO node has 2 VNFs; each EO has 2 VMs; and each radio CS has 1 VNF. The resource demand of each mission critical TA is illustrated in Table VII. Four types of emerging fog applications, which have distributed users in the access layer, are considered. Based on the assumption that all applications required by each enterprise are either hosted locally as VMs or hosted remotely in centralized cloud DCs, distributed users of emerging fog applications are not associated with EOs. All user demands for a fog app in each access node are grouped together to form a cluster of user demand in that node. In all scenarios, a group of 5 end-users, which are attached to each radio CS via wireless media, collectively form a single clustered demand for each emerging fog app at the access layer. While a single end-user located in each residential house,

TABLE VII
RESOURCE DEMAND OF FOG APPLICATIONS

Fog App	CPU demand (GHz)	Memory demand (GB)	Storage demand (GB)	Pre-processing data rate per user (Gbps)	Post-processing data rate per user (Gbps)
A	1.8	7.2	80	-	-
B	1.8	26	240	-	-
C	2.7	10.8	240	-	-
D	0.9	13	160	-	-
E	0.9	3.6	160	-	-
F	2.7	32	160	-	-
G	1.8	26	80	-	-
H	2.7	10.8	80	-	-
I	2.7	32	80	-	-
J	1.8	7.2	160	-	-
K	1.8	26	240	-	-
L	2.7	10.8	80	-	-
M	0.9	3.6	160	-	-
N	0.9	13	240	-	-
O	2.7	10.8	80	-	-
P	1.8	26	80	-	-
Q	1.8	7.2	160	-	-
R	2.7	10.8	80	-	-
S	0.9	3.6	160	-	-
T	0.9	13	160	-	-
U	1.8	26	120	0.9	0.45
V	2.7	32	160	0.11	0.05
W	1.8	7.2	80	0.83	0.41
X	0.9	3.6	40	0.43	0.22

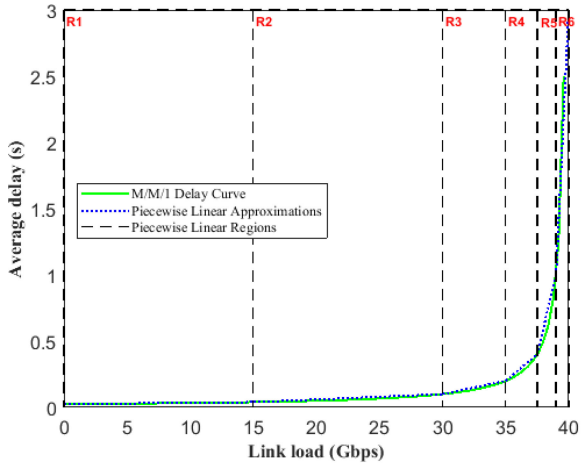
forms a single clustered demand for each emerging fog app in the access layer.

Both VM/VNF of traditional fog apps and emerging fog apps have a mix of resource intensity as illustrated in Table VII. Relative to the maximum compute resource capacity some apps are CPU intensive while others are memory intensive. Relative to the capacity of computes resource adopted, fog app "U" has medium CPU demand, high memory demand and low storage demand. Fog app "V" has high CPU demand, high memory demand and medium storage demand, Fog app "W" has medium CPU demand, low memory, and low storage demand. Fog app "X" has low CPU, memory, and storage demands. Relative to other emerging fog apps, fog apps U and X have the highest pre and post processing traffic per user as illustrated in Table VII. In spite of varying processing demands by fog applications, the MILP model and heuristic provision each instance of a fog application by proportionally utilizing CPU, memory and storage components in the corresponding fog computing sites. However, the resource locality constraint must be satisfied for the corresponding server architecture being considered.

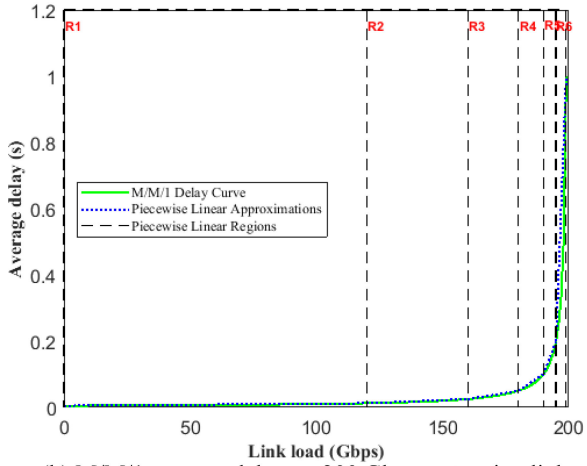
Although the placement of apps in centralized cloud DCs is not explicitly considered, the impact of cloud destined traffic on the overall performance of metro and access network tiers is considered and it is further assumed that traffic to and from the centralized cloud DC is part of the regular traffic traversing the network topology. A scenario where priority is given to traffic of emerging fog apps in the fog network is considered while traffic of other (including traditional) applications and services contribute to the regular traffic traversing metro and access networks. Therefore, the traffic of traditional apps is embedded into the regular traffic. The regular traffic in the metro-access network is an input parameter to the MILP model and the heuristic. In this paper, the range of regular traffic on the metro ring and access links are 114 – 120 Gbps and 4 – 5 Gbps, respectively. Therefore, regular traffic utilizes about 60% and 12.5% of the capacity of a metro and access link, respectively. It is important to note that the regular traffic in the metro-access network can also be predefined based on a reference traffic model.

The network traffic that is associated with each instance of an emerging fog app is routed over the metro-access network without traffic splitting. The presence of the traffic of emerging fog apps in the metro-access network changes the state of the network. The pre-processing and post-processing data rates per user for each emerging fog app considered are given in Table VII. After processing at the optimal location, processed data is sent to the requesting users at the edge of the network and to the central cloud for further/historical analysis and persistent storage. It is assumed that the ratio of post-processing data rate to pre-processing data rate is about 50% as illustrated in Table VII. In the worst-case scenario, if the request made by a given user (located at access node 7, a residential house with a single user for each emerging fog app) for all emerging fog apps is satisfied about 6% of the access link capacity will be utilized. On the other hand, if the requests made by all users in node 6 (a radio CS with multiple users for each emerging fog app) are satisfied, about 30% of the access link capacity will be utilized. Generally, compared to the fog related traffic, regular traffic is dominant in the network topology.

A scenario where shared network elements such as Ethernet access and aggregation routers and optical line terminals (OLTs) are assumed to have a load proportional power profile is considered. On the other hand, dedicated network components such as consumer premises equipment (CPE) and optical network units (ONUs) have an on-off power profile. Table VI shows the power profile of each network element and their corresponding values. The exponential M/M/1 delay curve of each network link is divided into 6 piecewise linear segments (or pieces) to implement piecewise linearization of the non-linear delay curve. Fig. 5 gives the piecewise linear approximation of both 200 Gbps and 40 Gbps network links using the 6 piecewise linear segments. Each piecewise linear segment is a linear approximation of a segment of the non-linear M/M/1 delay curve (i.e., the green curve) as shown in Fig. 5. Each linear segment (i.e., a blue line) falls within an approximate linear region (i.e., R1, R2, R3, R4, R5 or R6) of the M/M/1 delay curve as illustrated in Fig. 5. We have examined the use of more than 6 piecewise linear segments and no appreciable improvement in accuracy was obtained. When using a much lower number of linear segments (two or three segments) the converse was observed. The predefined upper bound for link load on both 200 Gbps and 40 Gbps network links are 195 Gbps and 39 Gbps, respectively. Both values enforce a corresponding upper bound for queuing delay on each link. These values maybe varied based on expected network performance as desired by the network service provider on the corresponding link.



(a) M/M/1 average delay on 40 Gbps access link



(b) M/M/1 average delay on 200 Gbps metro ring link

Fig. 5. M/M/1 average delay on metro-access links.

This study evaluates the energy efficient placement of delay sensitive emerging fog applications in the presence of mission critical traditional fog applications in a shared distributed fog network. The MILP model is solved using the 64-bit AMPL/CPLEX solver on the ARC3 supercomputing node with 24 CPU cores and 128 GB of memory [45]. Analysis of results from the model focuses on metrics such as TFPC, TNPC, number of fog app instances created, round-trip delay experienced by users of emerging fog applications. The number of active resource components across all fog computing sites in fog network and the corresponding average utilization of each active component type across the fog network is also adopted as an evaluation metric. To obtain optimal results, the results show that the MILP model effectively bin-packs applications resource demands onto fog computing resources to achieve optimal resource power and utilization efficiencies within capacity constraints and limited resource utilization scope.

B. Energy Efficient Placement in Non-Federated Fog Layer

The non-federated fog layer scenario assumes that a fog network is not created. Hence, distributed fog nodes are not linked by the metro-access network and the spare computing capacity at selected fog sites in Fig. 4 are inaccessible.

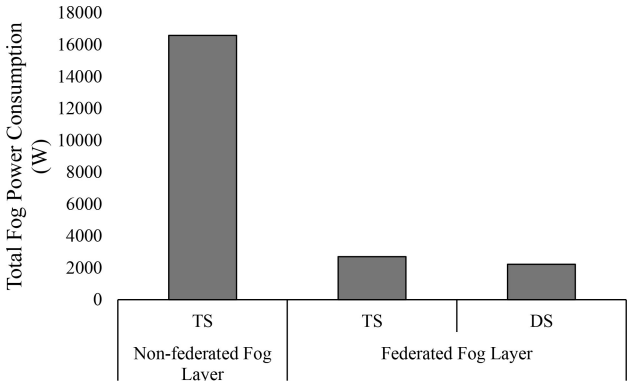


Fig. 6. Total fog power consumption of non-federated and federated fog layers.

Therefore, dedicated fog computing capacity is required for each cluster of users that have a request for a given emerging fog app. To simulate the non-federated fog layer scenario, the MILP model is solved to optimally place TAs into TSs when $\emptyset = 0$, i.e., cost of rejecting any emerging fog app instance is zero. Consequently, all emerging fog app instance requests are rejected and must be provisioned with standalone computing capacity at the source of the user request. Fig. 6 shows the TFPC due to the activation of computing nodes in the selected network location and due to the use of fog computing nodes at the source of the fog app instance demand. Non-federated fog computing layer consumes significantly high fog computing power consumption since computing capacity is required at the source of user demand for an instance of each emerging fog application.

Fig. 7 and Fig. 8 show the corresponding utilization of each compute components across distributed fog nodes under TS and DS architectures respectively after optimal placement of mission critical TAs. These results are also obtained by solving the MILP model to optimally place all fog apps when $\emptyset = 0$, i.e., cost of rejecting any emerging fog app instance is zero. Consequently, no instance of emerging fog app is provisioned in the fog network. Fig. 7 and Fig. 8 show the presence of unused computing capacity to support emerging fog applications in preferred fog computing sites after TAs have been provisioned under both TS and DS architectures. Relative to the TS architecture, the DS architecture has greater average active resource utilization and reduced number of active components across fog computing sites. It is expected that these advantages of DS architecture over TS architecture will be maintained when attempts are made to also place emerging fog apps into the fog network. However, to achieve optimal efficiency, the placement of each TA within each network node may be revised according to the characteristics of the emerging fog apps under consideration. Furthermore, subsequent analysis of application placement is focused on emerging fog applications alone since the placement of mission critical TAs is fixed to specific fog computing sites in the system setup.

C. Energy Efficient Placement Under Low Delay Penalty

Under this scenario, a fog network is created via the federation of computing capacity at selected fog sites over the

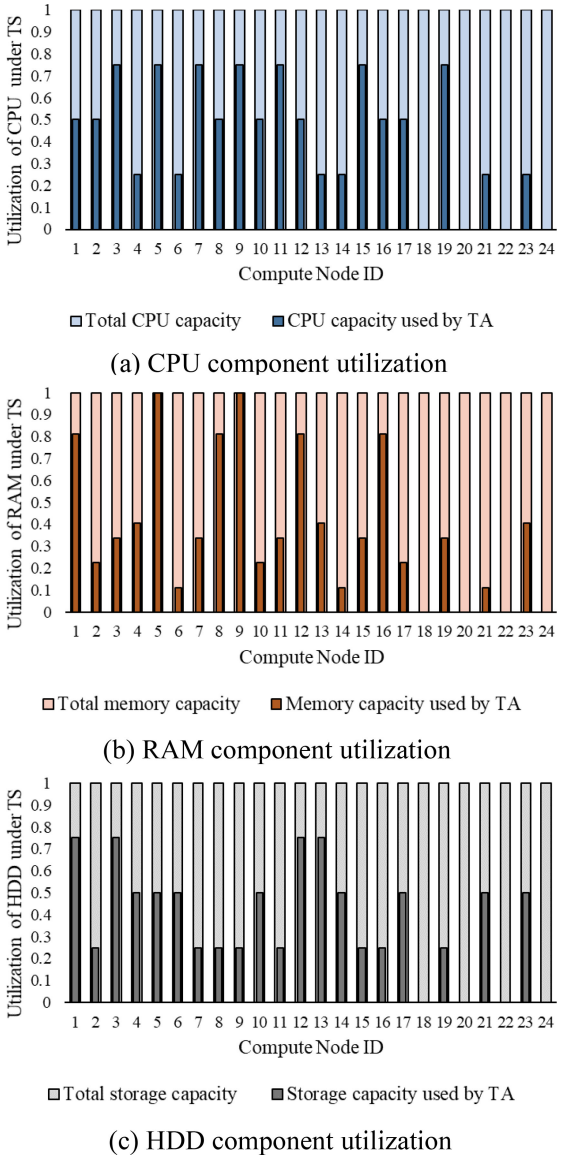


Fig. 7. Resource component utilization under TS architecture.

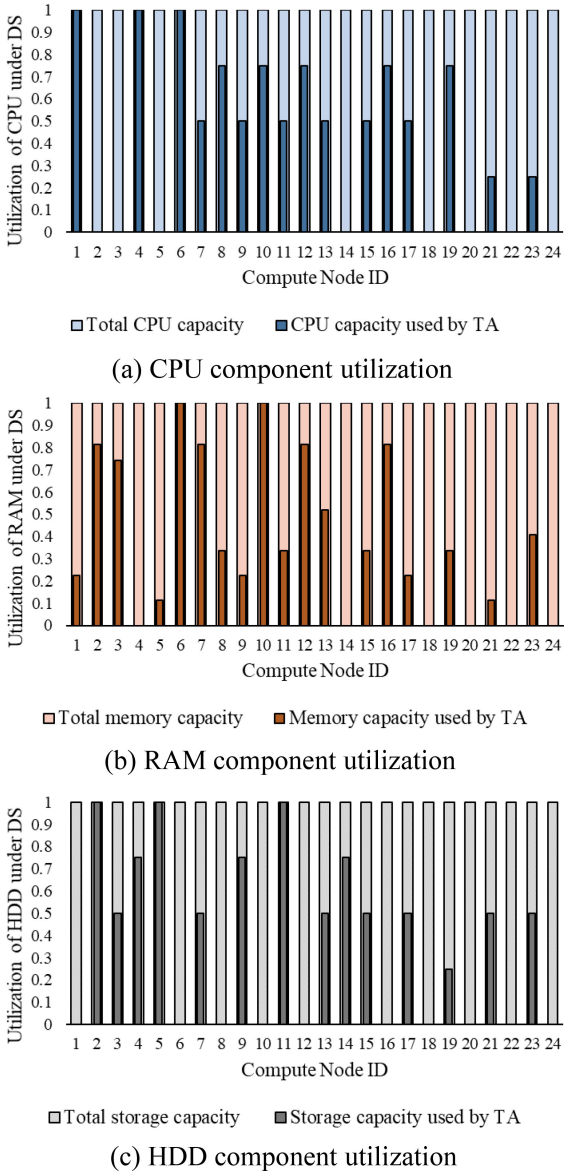


Fig. 8. Resource component utilization under DS architecture.

metro-access network in Fig. 4. Furthermore, the VMs/VNFs of both mission critical TAs and emerging fog applications are optimally placed into the fog network. All emerging fog apps considered are moderately sensitive to end-to-end delay in the network and the network has a trivial queuing delay penalty, i.e., $\delta \ll 1$ when the MILP model is solved. The scenario is evaluated when TS architecture is adopted and when the DS architecture is adopted in fog computing servers.

1) Placement Under Traditional Server Architecture: The illustration in Fig. 9 shows the optimal placement of emerging fog applications when TSs are deployed in the fog network. A single instance of each emerging fog app is provisioned in the fog network. The instance provisioned for each emerging fog app satisfies the computing capacity requested by all geo-distributed users of that app. These provisioned instances are strategically placed in the fog network to minimize both TNPC and TFPC. All mission critical TAs are also placed in the required network nodes to minimize the high cost of rejecting them as defined in the objective. Therefore, both TCREA and TCRTA are both zero. Network power consumption is dominant because a small fog network with a limited number of fog computing nodes, users and applications is considered. Moreover, fixed regular traffic in the network topology accounts for a significant portion of the TNPC.

Given enough CPU, memory and storage resource capacity in metro COs, the placement of fog apps as illustrated in Fig. 9 shows that there is high preference for metro COs when the selection of optimal locations for emerging fog apps is made. Relative to other network nodes, COs are centrally located, closer to geo-distributed users and closer to the metro gateway to the cloud. Hence, placement of fog apps in COs reduces the number of hops traveled by the fog traffic and therefore reduces TNPC. As illustrated in Fig. 9, instances of three emerging fog apps are placed in metro COs. Because homogenous servers are adopted across the distributed fog

Scenario	Low delay penalty				High delay penalty		Low delay penalty with delay sensitive fog app "X"			
Server Architecture	TS		DS		TS	DS	TS		DS	
Network Node (type)	MILP	HEEDAP	MILP	HEEDAP	MILP		MILP	HEEDAP	MILP	HEEDAP
1 (CO)	◆★	◆★			◆★		◆	◆		
2 (CO)					◆	◆●				✚
3 (CO)	●	●			●		●	●		
4 (CO)			◆★	◆★		◆★				◆
5 (EO)										
6 (Radio CS)	✚	✚	●		★	●	★✚	★✚	★✚◆	★
7 (Home)										
8 (Home)										
9 (EO)										
10 (Radio CS)			★		★	★	★	★	★	★
11 (Home)										
12 (Home)										
13 (EO)										
14 (Radio CS)				●	◆★	✚	★	★	★	★
15 (Home)										
16 (Home)										
17 (EO)										
18 (Radio CS)			✚	✚	◆★✚	●★	★	★	★●	★●
19 (Home)										
20 (Home)										
● - Fog app U ✚ - Fog app U ◆ - Fog app W ★ - Fog app X										

Fig. 9. Energy efficient placement of emerging fog apps in a fog network.

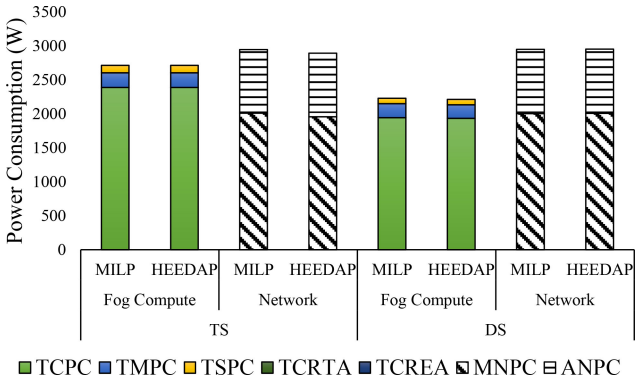
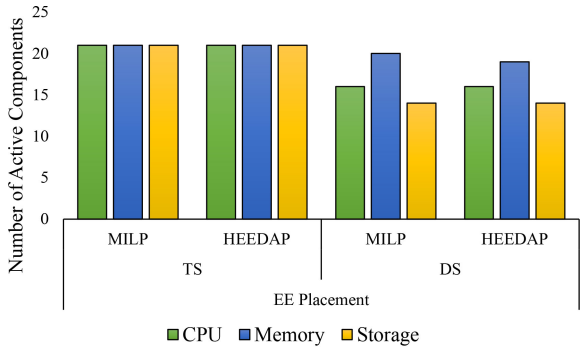
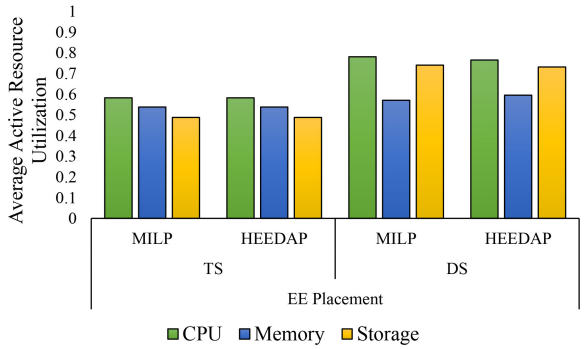


Fig. 10. Power consumption under energy efficient placement in a fog network with low delay penalty.



(a) Number of active components across fog sites



(b) Average utilization of active components across fog sites

Fig. 11. Resource components utilization across fog sites in the network under energy efficient placement in a fog network with low delay penalty.

computing nodes, the power consumption incurred by hosting a given emerging fog app in inactive resource components is the same across all network nodes. Likewise, power consumption incurred by hosting a given emerging fog app in the idle resource capacity of active resource components is also equal across all network nodes. Hence, multiple candidate metro COs which lead to the same increment of TFPC may exist for a given emerging fog app. Such ties are broken by selecting the metro CO which enables lower total (i.e., fog computing plus network) power consumption. Network node 1, which is also the metro gateway node to the core network in the network topology, wins such tie breaks. This is because the placement of emerging fog apps close to the metro gateway helps to reduce the number of hops traversed in the network topology and consequently the TNPC. Emerging fog apps "W" and "X" are placed in network node 1 as shown in Fig. 9. However, when resource capacity in network node 1 is limited, other candidate metro COs with adequate resource capacity are considered. Consequently, network node 3 is selected to host emerging fog app "U" as shown in Fig. 9.

In the absence of enough resource capacity in fog computing nodes attached to metro COs, fog sites in the access network must be selected to support emerging fog apps in the fog network if such fog sites have adequate computing capacity. However, multiple candidate access nodes may also present equal compute energy efficiency to host a given emerging fog application (as result of the homogeneous power profile of fog computing nodes across the distributed fog network). Hence, network energy efficiency is used as a decision metric to select the optimal network node in such scenarios. For example, emerging fog app "V" which requires a dedicated server because of the intensive nature of its memory demand is placed in network node 6, an access node as illustrated in Fig. 7. This node is selected over other network nodes (10, 14 and 18) due to its proximity to the metro gateway node which promotes lower total network traffic because the number of hops traversed by cloud bound traffic is reduced. It is important to note that unused servers are also present in network nodes 10, 14 and 18 as illustrated in Fig. 4.

The HEEDAP algorithm achieves comparable results as those reported when the MILP model is solved as shown in Fig. 10. As shown in Fig. 10 and Fig. 11, the HEEDAP algorithm achieves the same TFPC, number of active resource components and average active resource utilization as the MILP model when the TS architecture is deployed in the fog network. This demonstrates the efficacy of the HEEDAP algorithm at mimicking the compute energy efficiency achieved by

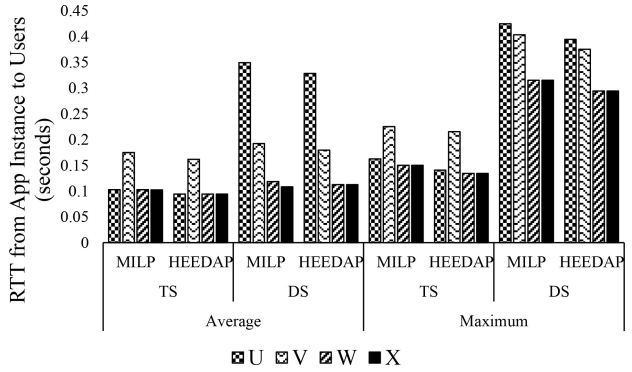


Fig. 12. Round trip delay from app instance to users under energy efficient placement in a fog network with low delay penalty.

the MILP model in a similar system setup that adopts homogenous resources across the fog network. A single instance of each emerging fog app is provisioned in the fog network to serve all distributed users as reported when the MILP model was solved as shown in Fig. 9. All traditional apps in the fog network are also provisioned. Both TCREA and TCRTA obtained by HEEDAP are both zero as shown in Fig. 10. The placement of the instance created for each emerging fog app obtained via the HEEDAP algorithm is also an exact match with those obtained by solving the MILP model when TS architecture is deployed in fog nodes. Consequently, the average and maximum RTT from app instance to users obtained by HEEDAP are also comparable to those obtained by solving the MILP model when TS architecture is adopted in fog network nodes as shown in Fig. 12. The TNPC obtained using the HEEDAP algorithm is marginally (about 2%) lower than that of the MILP model as shown in Fig. 10. Disparity in path selection made for cloud bound traffic is responsible for this difference, while the MILP minimizes overall congestion in the network topology by distributing such traffic as necessary (which leads to higher network power consumption), HEEDAP always selects the shortest path which in turns minimizes network power consumption.

2) Placement Under Disaggregated Server Architecture: Replacing TSs with DSs in the fog network leads to changes in the optimal placement of emerging fog applications as shown in Fig. 9. Improved consolidation of both traditional and emerging fog apps enabled by the adoption of DSs in the distributed fog network is responsible for the revised placement observed. Consequently, Fig. 11b shows corresponding increases in the average utilization of active resources components in the fog network as result of the revision in server architecture. All traditional apps in the fog network are provisioned and the demand of all distributed users of emerging fog apps are satisfied. A primary instance of each emerging fog app is provisioned in the network node that leads to optimal energy efficiency when the DS architecture is deployed in the fog network. Furthermore, additional instances of an emerging fog app are created if such instances lead to marginal rise in TFPC while achieving significant drop in the TNPC. Reductions in TNPC is achieved because the creation of additional instance(s) enable reductions in the number of hops traversed. This explains the creation of two instances of emerging fog app "X". The instance in network node 18 is responsible for distributed users of the application in network nodes 6, 7, 8, 14, 15, 16, 18, 19 and 20. A second instance of emerging fog app "X" in network node 10, which is provisioned using IRC of active resource components, is responsible for distributed users in network nodes 10, 11, and 12. Hence, the number of hops between instances of app "X" and their distributed users is minimized.

The revisions in fog apps placement observed when DSs are deployed is responsible for the fall (about 18%) in TFPC relative to results obtained when TSs are deployed in the fog network as shown in Fig. 10. Reduction in the TCPC is responsible for over 90% of the fall seen in TFPC. This is because power consumption of CPU component is significantly higher than that of memory and storage components. Disaggregation enables significant improvements in CPU utilization efficiency (as shown in Fig. 11b) via improved consolidation of CPU demands of mission critical traditional apps and emerging fog apps in each fog computing site. Hence, the number of active CPU component reduced when DS are deployed to replace TS in the fog network as shown in Fig. 11a. The same is also true for HDD components and their corresponding utilization efficiency. However, the 33% drop in the number of HDD component observed in Fig. 11a does not lead to significant fall in the TFPC because HDD components have a lower peak power consumption relative to CPU and memory components as illustrated in Table VI. Fig. 11a only shows a marginal drop in the number of active memory components. This is because several considered applications have high memory demand relative to the capacity of the homogenous memory components as given in Table VII. Hence, a significant improvement in active memory component utilization could not be realized as shown in Fig. 11b.

Fig. 10 shows a marginal increase in the TNPC obtained by solving the MILP model after TSs are replaced with DSs in the fog network. This marginal rise is as a result of increased hop count between the instance of some emerging fog apps and their users. Furthermore, the hop count between instance of emerging fog app "W" and "X" and the metro gateway node also increases after a change in server architecture. Hence, network traffic traverses through more network equipment when DSs are deployed in the fog network. Fig. 12 gives the average and maximum round trip time between distributed user of each emerging fog app and the provisioned instances of the app. Relative to the deployment of TS architecture in the fog network, both average and maximum RTT increased when DS architecture is adopted in the fog network. Users of emerging fog app "U" experience relatively higher delay as shown in Fig. 12 compared to other emerging fog apps. Although the placement of fog app "U" in network node 6 enables optimal energy efficiency in the fog network, this choice also leads to high congestion on the link that connects network node 6 to node 1. Consequently, users of emerging fog apps "V", "W" and "X", which are in node 6, experience the corresponding maximum delay illustrated in Fig. 12. Since, the moderate delay thresholds of all emerging fog apps under this scenario are satisfied, such performance is acceptable. However, the

performance obtained under both TS and DS architectures violates the delay threshold (i.e., $\tau \leq 20ms$) for delay sensitive fog apps. Hence, another subsection considers a scenario where requests for a delay sensitive emerging fog app are present in the fog network.

The same trends observed from the analysis of the solved MILP model are also observed when the HEEDAP algorithm is deployed to perform energy efficient placement of moderately sensitive emerging fog apps in a fog network that adopts DS architecture. The TFPC obtained by HEEDAP algorithm is almost equal to that obtained by solving the MILP model. The TFPC of the HEEDAP algorithm is marginally (1%) lower because only a single instance of emerging fog app "X" is provisioned when HEEDAP algorithm is deployed as shown Fig. 9. This contrasts with the creation of two instances of the same emerging fog app when the MILP model was solved. Consequently, relative to the TNPC obtained by solving the MILP model, the TNPC obtained by HEEDAP algorithm is marginally higher because the total volume of traffic in the network is higher. Furthermore, the placement of emerging fog apps obtained by the HEEDAP algorithm as shown in Fig. 9 is largely comparable to those obtained by solving the MILP model. However, compared to results obtained by solving the MILP model, the placement of emerging fog app "U" as obtained by the HEEDAP algorithm is different since the app is placed in node 14. This revised placement is responsible for the fall in the average and maximum RTT experienced by the distributed users of the app as shown in Fig. 12. This is because node 14 is farther away from node 1 which is also the metro gateway node to the cloud. Hence, the congestion on the paths to node 14 is lower.

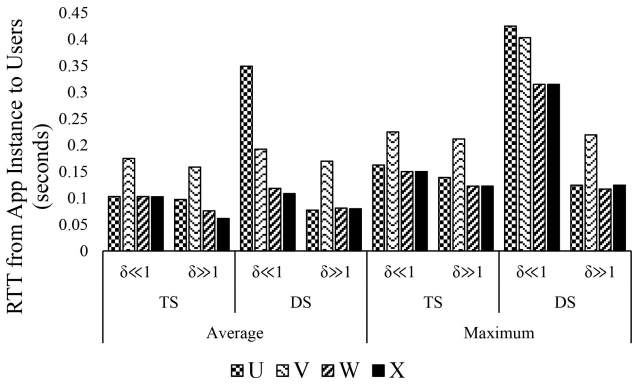


Fig. 13. Round trip delay from app instance to users under EE placement scenarios.

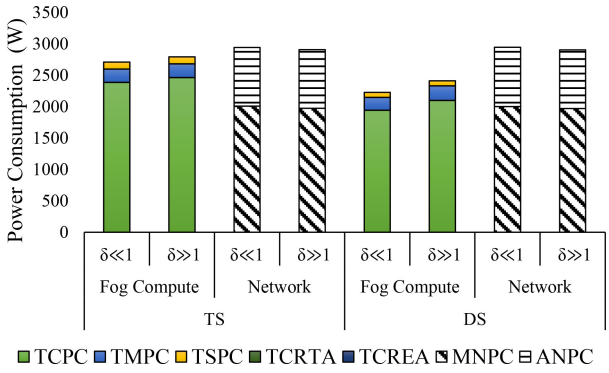


Fig. 14. Power consumption under EE placement scenarios.

D. Energy Efficient Placement Under High Delay Penalty

In this subsection, a fog network is created and $\delta \gg 1$; hence, network delay penalty is high. This represents a network where the network operator desires minimal impact of emerging fog apps on regular traffic. All results in this sub-section are obtained by solving the MILP model.

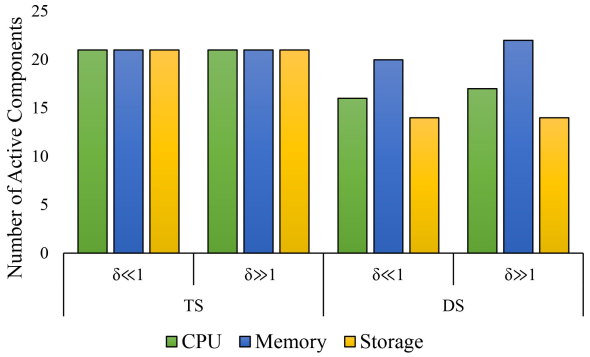
1) Placement Under Traditional Server Architecture: Under this scenario, multiple instances of some emerging fog apps are created when the TS architecture is adopted as shown in Fig. 9. This reduces the number of hops between instances of replicated fog apps and their users. Consequently, the total volume of traffic traversing the network topology is reduced and the delay experienced on each link of the network topology is also minimized. Relative to result obtained under low delay penalty ($\delta \ll 1$), the average and maximum RTT between the instance of fog apps and their distributed user falls when $\delta \gg 1$ as shown in Fig. 13. However, to ensure a balanced trade-off between minimizing TFPC and the total approximated delay, only applications (app "W" and app "X") with low-medium resource demand intensity are replicated as shown in Fig. 9. The primary instances for both apps are placed in centrally located fog sites (i.e., COs). These primary instances are responsible for users in directly attached access network nodes and for most distributed users in the network topology. On the other hand, additional instances of emerging fog app "W" and app "X" are provisioned in some radio CSs. This strategy reduces the traffic associated with densely populated user clusters attached to each radio CS in the network and consequently reduces network congestion. Because the app "W" and app "X" have small compute footprint, they are easily provisioned with IRC of active TSs. Hence, minimal increase in TFPC is incurred compared to the $\delta \ll 1$ scenario as shown in Fig. 14. Furthermore, relative to results obtained when $\delta \ll 1$, additional compute components are not activated to provision the additional instances of app "W" and "X" as shown in Fig. 15a. However, replication of fog app instances leads to the increase observed in the utilization of active resource components in the fog network as shown in Fig. 15b. Relative to the low delay penalty scenario where TSs are deployed, there is a 3% rise in the TFPC when the high delay penalty scenario is considered under a similar setup. On the other hand, the TNPC consumption fell by 2% as shown in Fig. 14.

2) Placement Under Disaggregated Server Architecture: A similar trend is observed when DSs are deployed in the fog network. Replicas of certain emerging fog apps are made, as shown in Fig. 9. This strategy enables reductions in the volume of traffic traversing the network topology and increased network congestions that may arise due to the creation of a single instance of each fog app. Consequently, distributed users of fog apps experienced lower average and maximum round trip time to assigned fog instances as shown in Fig. 13. Under the high delay penalty scenario, three instances of app "U" and

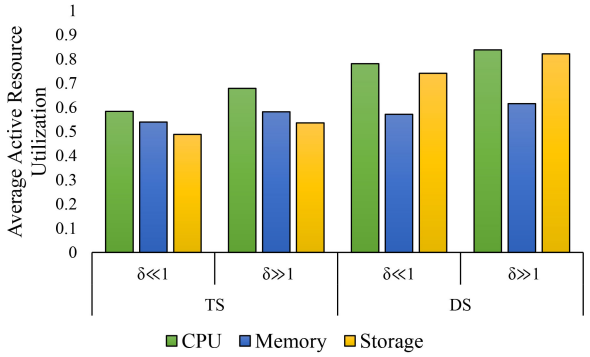
app "X" are created while only two instances of fog app "W" are created. High CPU and memory demands of emerging fog app "V" prevents the replication of the app to avoid significant increase in TFPC. As observed when TSs are deployed in the network, radio CSs are often used to host instances of emerging fog apps to ensure that the total volume of traffic in the network topology is minimized. This because of the high user density associated with radio CSs in the system setup. For example, App "V" is placed in network node 14 because the node does not require the activation of an additional CPU component to support the CPU intensive demands of emerging fog app "V".

Furthermore, since app "U" and app "W" have higher data rate per user relative to other emerging fog apps as illustrated in Table VII, replication of these emerging fog apps can significantly reduce the total volume of traffic and congestion experienced in the network. Additionally, app "X" has non-intensive CPU and memory demands; therefore, the replication of app "X" does not lead to significant increase in the TFPC but can lead to additional reduction in the network traffic. Relative to the results obtained when low delay penalty is considered under a similar setup, the creation of replicas of some emerging fog apps when the DSs are deployed under the high delay scenario leads to about 8% rise in the TFPC as seen in Fig. 14. A comparison of TNPC under both scenarios shows about 1% decrease due to reduced traffic on the network topology as illustrated in Fig. 14. Under DS architecture, the creation of replicas of some emerging fog apps ("U", "W" and "X") under the high delay penalty increased the number of active compute resource components and the average active utilization of resource components in the fog network relative to the low delay penalty scenario as shown in Fig. 15a and Fig. 15b respectively.

Comparison of TS and DS architectures under the high delay penalty scenario expectedly shows that the adoption of the disaggregation concept enabled notable (about 14%) reduction in TFPC as shown in Fig. 14. This is because server resource components are independently and proportionally utilized. A marginal fall in TNPC is also observed as a result of the revised server architecture. This is because the DS architecture encouraged the creation of more distributed replicas of most emerging fog apps relative to when the TS architecture is adopted in compute nodes within the fog network. Compared to the placement obtained under TS architecture where replication of emerging fog app "U" is discouraged because of the app's high compute footprint, replicas of emerging fog app "U" are created when the DS architecture is deployed. Note that app "U" has moderate CPU demand and high memory demand; hence, proportional usage of resource components when DS architecture is employed promotes the independent activation of new memory components to support replicas of app "U", while the moderate CPU demands of the app's replicas are aggregated with the CPU demand of other applications into active CPU components. However, replication of app "V" is still discouraged because of its high CPU and memory demands. Therefore, proportional usage of resource components does not enable sufficient benefits to promote replication of an emerging fog app that is CPU and memory intensive.



(a) Number of active components across fog sites



(b) Average utilization of active components across fog sites

Fig. 15. Resource components utilization across fog sites under EE placement scenarios.

Relative to the deployment of TS under the high delay penalty scenario, Fig. 13 shows that the average and the maximum round trip time are higher for some emerging fog apps when DSs are employed. Thus, the number of hops between users of such emerging fog apps and the instances of the app that is assigned to them increased because more network nodes are traversed.

E. Energy Efficient Placement of Delay Sensitive Fog App

A network with trivial queuing delay penalty ($\delta \ll 1$) is adopted under this scenario. In contrast with the previous scenario, emerging fog app "X" is sensitive to end to end delay experienced in the network (i.e., $\tau \leq 20ms$) under this scenario. Other emerging fog apps remain moderately sensitive to end-to-end delay in the network as in previous subsections. Under this scenario, EE placement of both traditional and emerging fog apps is also evaluated when both TS and DS architectures are deployed in the fog computing nodes placed in the fog network.

Multiple instances of the delay sensitive emerging fog app are provisioned at all radio CS in the network topology as shown in Fig. 9 irrespective of the server architecture adopted in fog nodes. This ensures that the delay threshold of the fog app "X" is satisfied for users that are directly attached to a radio CS. On the other hand, users of emerging fog app "X" which do not have direct access to a radio CS are out rightly rejected. Hence, local computation capacity is required to support such users. This led to additional fog compute power consumption as observed in Fig. 16. It also implies higher CAPEX

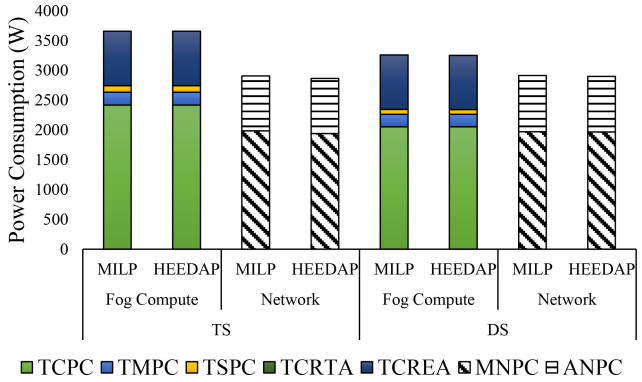


Fig. 16. Power consumption following energy efficient placement of delay sensitive fog app.

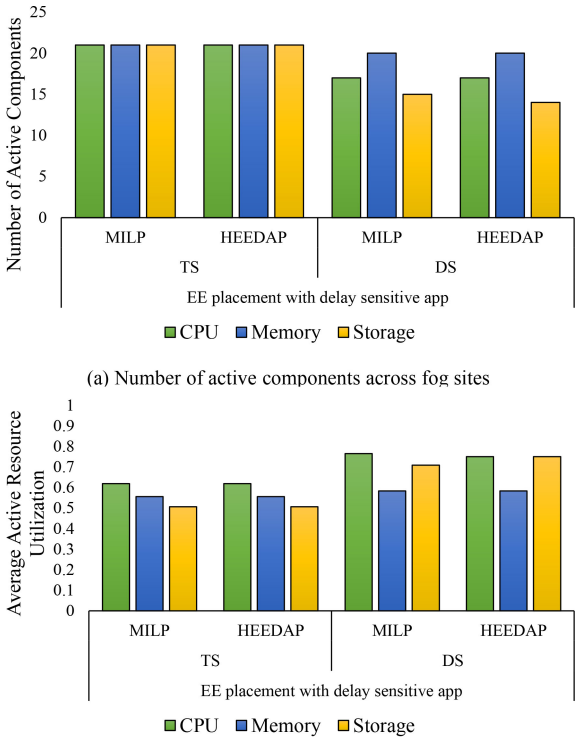


Fig. 17. Resource components utilization across fog sites in the network following energy efficient placement of a delay sensitive fog app.

and OPEX. It is expected that a similar placement strategy will be implement if users of emerging fog apps are associated with enterprise office. Other emerging fog apps, which are moderately sensitive to delay, are placed in the fog network as shown in Fig. 9 to achieve optimal energy efficiency in the fog network as reported in Section V-C. Fig. 16 shows that the TNPC increases marginally when DS architecture is deployed in fog sites to replace TS architecture as observed previously.

Furthermore, the TFPC and the number of active components is lower when DSs are used to replace TSs in the distributed fog network as shown in Fig. 16 and Fig. 17a, respectively. By reducing the number of active resource components while provisioning the same number of emerging fog apps as when TS architecture is adopted, the adoption of DS architecture in the fog network can increase the amount of spare capacity available in the fog network. This spare capacity can support both highly sensitive and moderately sensitive emerging fog apps at the network edge without additional CAPEX. As observed in Section V-C, Fig. 18 shows that the average and maximum delay experienced by distributed users of emerging fog apps, which are moderately sensitive to end to end delay, increases when DS architecture is adopted. For instance, the illustration in Fig. 9 shows that each instance of moderately sensitive emerging fog apps created are provisioned in radio CSs, which are far from most of the distributed users of each fog app. However, the performance of such applications does not degrade because they have greater delay-tolerance. It is important to note that the delay experienced by accepted users of the delay sensitive app is extremely low and insignificant as expected of today's 5G mobile networks and future 6G mobile networks.

Under this scenario, the HEEDAP algorithm also effectively mimicked the performance of the MILP model when TS or DS architecture is adopted in the fog network as depleted in Fig. 9. Pre-processing of input applications, which is performed in the initial steps of the HEEDAP algorithm, simplified the placement or rejection of delay sensitive fog apps in the presence or absence of in situ computing capacity at the source of each request for such apps. Therefore, the HEEDAP algorithm effectively mimicked the MILP model by provisioning some instances of delay sensitive emerging fog apps at radio CSs to serves users at such location. Users of delay sensitive emerging fog apps located at network nodes without local computing capacity are rejected. Thus, the corresponding cost of rejection (i.e., TCREA) as represented by fog computing power consumption can be seen in Fig. 16. Additionally, CAPEX and OPEX will also be incurred to setup an in-situ fog computing node for each user cluster of the rejected fog app.

When TS architecture is deployed in the fog network, the resulting placement of emerging fog apps by the HEEDAP algorithm is an exact replica of the placement obtained by solving the MILP model. Consequently, the same TFPC is achieved by both the MILP model and HEEDAP algorithm under the TS architecture as shown in Fig. 16. As shown in Fig. 17a, the HEEDAP algorithm obtained the same number of active resource component as those obtained by solving the MILP model. Likewise, HEEDAP also replicated the average utilization of active components across fog computing sites that obtained by solving the MILP model as show in Fig. 17b. The TNPC obtained by the HEEDAP algorithm is also comparable to the same value obtained by solving the MILP model under a similar scenario. Similarly, the average and maximum RTT to distributed users of emerging fog apps obtained by HEEDAP is comparable to those obtained by the solving the MILP model as shown in Fig. 18.

However, when the DS architecture is employed in the fog network, the resulting placement of emerging fog apps by the HEEDAP algorithm is not an exact replica of the placement obtained by solving the MILP. The TFPC obtained with the HEEDAP algorithms is about 2% higher than the TFPC obtained by solving the MILP model under this scenario as shown in Fig. 16. The adoption of homogeneous compute

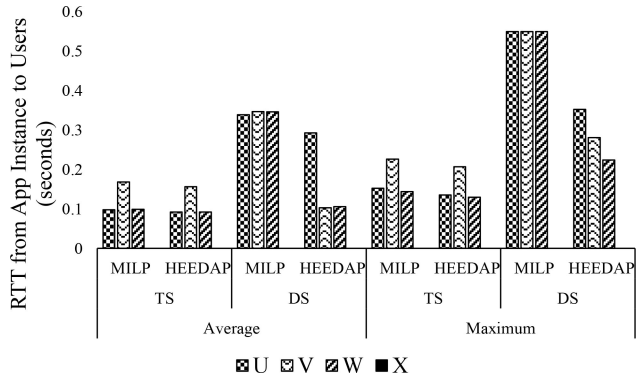


Fig. 18. Round trip delay experienced by users of emerging fog app following energy efficient placement of a delay sensitive fog app.

resources across the fog network is responsible for the comparable TFPC obtained. Therefore, the number of active resource components and the average utilization of these active components across fog computing sites as obtained by the HEEDAP algorithm is comparable to those obtained by solving the MILP model. Difference in the placement of emerging fog apps is responsible for the changes in the average and maximum RTT experienced by users as shown in Fig. 18. Apps "V" and "W" are placed in COs in the metro ring by the HEEDAP algorithm; hence, the average and maximum RTT experience by the distributed users of these fog apps is reduced compared to results obtained by solving the MILP model under the same setup.

VI. Conclusion

In this paper, we evaluated the energy efficient placement of delay sensitive emerging fog applications in the presence of mission critical traditional fog applications in a shared distributed fog network that employs TS and DS architecture across selected fog computing sites at the network edge. Compared to the non-federated fog computing layer, federation of selected fog computing sites over the metro-access network leads to significant reductions in TFPC. Relative to the use of the TS architecture in the fog network that is built over a network with low delay penalty, the adoption of DSs enabled up to 18% reduction in TFPC. This because disaggregation enabled proportional usage of compute resources at each fog computing site and consequently improved the energy efficiency of the fog network. However, this is achieved at the expense of marginal increase in TNPC and somewhat higher response time when DS architecture is adopted. Setting up a fog network with high delay penalty increased the TFPC when either TS or DS architecture are employed in fog computing sites. This was done to minimize the congestion experienced on the network by reducing the network traffic. Consequently, the TNPC is also reduced. But the TFPC of the fog network that employed DS architecture was 14% lower than that of the fog network that adopted TS architecture when a fog network with high delay penalty is considered. Our result also showed that COs and radio CSs are important edge locations for supporting interactive applications demanded by geo-distributed end-users when energy efficiency is an important design criterion, and such applications are moderately sensitive to the end-to-end delay experienced on the network. Otherwise, instances of such fog apps, which are more sensitive to delay, must be provisioned in the nearest network node that satisfies a predefined (and acceptable) delay threshold to distributed users. We also proposed a polynomial time heuristic, HEEDAP, which leverages a centralized orchestration and management framework, for a network of distributed fog computing nodes. The heuristic can rapidly provision both mission critical traditional applications and (delay sensitive) emerging fog applications in a fog network when possible. The HEEDAP algorithm achieves comparable results as those reported when the MILP model was solved under similar evaluation scenarios. On most occasions, the HEEDAP algorithm achieves the same application placement pattern, compute and network energy efficiencies as the exact results obtained by solving the MILP model. Occasional difference between the results obtained via by the HEEDAP algorithm and by solving the MILP model are marginal. For example, the difference between the TFPC obtained with the HEEDAP and that obtained by solving the MILP model is not greater than 2% in all scenarios considered. A limitation of the work in this paper is that delay in the fog network is modelled using M/M/1 queueing system which implicitly adopts the Poisson traffic model with exponential inter-arrival times. Adoption of the open queuing network can lead to more accurate modelling of the fog network. Future work will consider scenarios where mission critical traditional apps can be placed dynamically in contrast to the static placement considered in this paper. Furthermore, inter-workload communication may also be introduced between applications. The HEEDAP algorithm can also be extended or revised to investigate various placement strategies for emerging fog apps. Finally, artificial intelligence and machine learning techniques can also be adopted to further validate and enhance the proposed concepts and policy in this paper.

Acknowledgment

The first author would like to acknowledge his Ph.D. scholarship awarded by the Petroleum Technology Trust Fund (PTDF), Nigeria.

References

[1] "Cloud Vision 2020: The Future of the Cloud." LogicMonitor Inc. 2017. [Online]. Available: https://www.logicmonitor.com/wp-content/uploads/2017/12/LogicMonitor-Cloud-2020-The-Future-of-the-Cloud.pdf (Accessed: Jun. 17, 2019).

[2] "IDC Forecasts Worldwide Technology Spending on the Internet of Things to Reach $1.2 Trillion in 2022." DC. 2018. [Online]. Available: https://www.idc.com/getdoc.jsp?containerId=prUS43994118 (Accessed: Jun. 17, 2019).

[3] "Internet of Things." Cisco. 2016. [Online]. Available: www.cisco.com/go/iot (Accessed: Jun. 17, 2019).

[4] M. Satyanarayanan, P. Bahl, R. Caceres, and N. Davies, "The case for VM-based cloudlets in mobile computing," *IEEE Pervasive Comput.*, vol. 8, no. 4, pp. 14–23, Oct.–Dec. 2009.

[5] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli, "Fog computing and its role in the Internet of Things," in *Proc. 1st Ed. MCC Workshop Mobile Cloud Comput.*, 2012, pp. 13–16.

[6] F. Bonomi, R. Milito, P. Natarajan, and J. Zhu, "Fog computing: A platform for Internet of Things and analytics," in *Big Data and Internet of Things: A Roadmap for Smart Environments* (Studies in Computational Intelligence). F. Bonomi, R. Milito, P. Natarajan, J. Zhu, N. Bessis, and C. Dobre, Eds. Cham, Switzerland: Springer Int., 2014.

[7] M. Aazam, S. Zeadally, and K. A. Harras, "Fog computing architecture, evaluation, and future research directions," *IEEE Commun. Mag.*, vol. 56, no. 5, pp. 46–52, May 2018.
[8] Y. Yang, "FA2ST: Fog as a service technology," in *Proc. IEEE 41st Annu. Comput. Softw. Appl. Conf. (COMPSAC)*, vol. 7, 2017, p. 708.
[9] W. Li *et al.*, "System modelling and performance evaluation of a three-tier Cloud of Things," *Future Gener. Comput. Syst.*, vol. 70, pp. 104–125, May 2017.
[10] J. Taylor, "Facebook's data center infrastructure: Open compute, disaggregated rack, and beyond," in *Proc. Opt. Fiber Commun. Conf. Exhibit. (OFC)*, 2015, p. 1.
[11] A. D. Papaioannou, R. Nejabati, and D. Simeonidou, "The benefits of a disaggregated data centre: A resource allocation approach," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, 2016, pp. 1–7.
[12] B. Abali, R. J. Eickemeyer, H. Franke, C.-S. Li, and M. Taubenblatt, "Disaggregated and optically interconnected memory: When will it be cost effective?" 2015, *arXiv:1503.01416*.
[13] H. M. M. Ali, T. E. H. El-Gorashi, A. Q. Lawey, and J. M. H. Elmirghani, "Future energy efficient data centers with disaggregated servers," *J. Lightw. Technol.*, vol. 35, no. 24, pp. 5361–5380, Dec. 15, 2017.
[14] R. Lin, Y. Cheng, M. De Andrade, L. Wosinska, and J. Chen, "Disaggregated data centers: Challenges and trade-offs," *IEEE Commun. Mag.*, vol. 58, no. 2, pp. 20–26, Feb. 2020.
[15] O. O. Ajibola, T. E. H. El-Gorashi, and J. M. H. Elmirghani, "Disaggregation for improved efficiency in fog computing era," in *Proc. 21st Int. Conf. Transparent Opt. Netw. (ICTON)*, 2019, pp. 1–7.
[16] H. A. Alharbi, T. E. H. Elgorashi, and J. M. H. Elmirghani, "Energy efficient virtual machines placement over cloud-fog network architecture," *IEEE Access*, vol. 8, pp. 94697–94718, 2020.
[17] I. S. B. M. Isa, T. E. H. El-Gorashi, M. O. I. Musa, and J. M. H. Elmirghani, "Energy efficient fog-based healthcare monitoring infrastructure," *IEEE Access*, vol. 8, pp. 197828–197852, 2020.
[18] B. A. Yosuf, M. Musa, T. Elgorashi, and J. Elmirghani, "Energy efficient distributed processing for IoT," *IEEE Access*, vol. 8, pp. 161080–161108, 2020.
[19] "Fog Computing and the Internet of Things: Extend the Cloud to Where the Things Are What You Will Learn." Cisco. 2015. [Online]. Available: https://www.cisco.com/c/dam/en_us/solutions/trends/iot/docs/computing-overview.pdf (Accessed: Feb. 26, 2019).
[20] "Application Hosting on Catalyst 9000 Series of Switches—Cisco DevNet." Cisco. [Online]. Available: https://developer.cisco.com/docs/app-hosting/#!application-hosting-in-the-enterprise/what-is-application-hosting (Accessed: Apr. 21, 2020).
[21] K. David, J. Elmirghani, H. Haas, and X.-H. You, "Defining 6G: Challenges and opportunities [from the guest editors]," *IEEE Veh. Technol. Mag.*, vol. 14, no. 3, pp. 14–16, Sep. 2019.
[22] A. N. Al-Quzweeni, A. Q. Lawey, T. E. H. Elgorashi, and J. M. H. Elmirghani, "Optimized energy aware 5G network function virtualization," *IEEE Access*, vol. 7, pp. 44939–44958, 2019.
[23] N. Chen, Y. Yang, T. Zhang, M.-T. Zhou, X. Luo, and J. K. Zao, "Fog as a service technology," *IEEE Commun. Mag.*, vol. 56, no. 11, pp. 95–101, Nov. 2018.
[24] M. Chiang and T. Zhang, "Fog and IoT: An overview of research opportunities," *IEEE Internet Things J.*, vol. 3, no. 6, pp. 854–864, Dec. 2016.
[25] L. Cominardi *et al.*, "Opportunities and challenges of joint edge and fog orchestration," in *Proc. IEEE Wireless Commun. Netw. Conf. Workshops (WCNCW)*, 2018, pp. 344–349.
[26] 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.
[27] K. Katrinis *et al.*, "Rack-scale disaggregated cloud data centers: The dReDBox project vision," in *Proc. Des. Autom. Test Eur. Conf. Exhibit. (DATE)*, 2016, pp. 690–695.
[28] C.-S. Li, H. Franke, C. Parris, B. Abali, M. Kesavan, and V. Chang, "Composable architecture for rack scale big data computing," *Future Gener. Comput. Syst.*, vol. 67, pp. 180–193, Feb. 2017.
[29] G. Kandiraju, H. Franke, M. D. Williams, M. Steinder, and S. M. Black, "Software defined infrastructures," *IBM J. Res. Develop.*, vol. 58, nos. 2–3, pp. 1–13, Mar. 2014.
[30] O. O. Ajibola, T. E. H. El-Gorashi, and J. M. H. Elmirghani, "Energy efficient placement of workloads in composable data center networks," *J. Lightw. Technol.*, vol. 39, no. 10, pp. 3037–3063, May 15, 2021.
[31] X. Dong, T. El-Gorashi, and J. M. H. Elmirghani, "Green IP over WDM networks with data centers," *J. Lightw. Technol.*, vol. 29, no. 12, pp. 1861–1880, Jun. 15, 2011.
[32] X. Dong, T. El-Gorashi, and J. M. H. Elmirghani, "IP over WDM networks employing renewable energy sources," *J. Lightw. Technol.*, vol. 29, no. 1, pp. 3–14, Jan. 1, 2011.
[33] X. Dong, T. E. H. El-Gorashi, and J. M. H. Elmirghani, "On the energy efficiency of physical topology design for IP over WDM networks," *J. Lightw. Technol.*, vol. 30, no. 12, pp. 1931–1942, Jun. 15, 2012.
[34] A. Q. Lawey, T. E. H. El-Gorashi, and J. M. H. Elmirghani, "Distributed energy efficient clouds over core networks," *J. Lightw. Technol.*, vol. 32, no. 7, pp. 1261–1281, Apr. 1, 2014.
[35] A. Q. Lawey, T. E. H. El-Gorashi, and J. M. H. Elmirghani, "BitTorrent content distribution in optical networks," *J. Lightw. Technol.*, vol. 32, no. 21, pp. 4209–4225, Nov. 1, 2014.
[36] L. Nonde, T. E. H. El-Gorashi, and J. M. H. Elmirghani, "Energy efficient virtual network embedding for cloud networks," *J. Lightw. Technol.*, vol. 33, no. 9, pp. 1828–1849, May 1, 2015.
[37] K. Hong, D. Lillethun, U. Ramachandran, B. Ottenwälder, and B. Koldehofe, "Mobile fog: A programming model for large-scale applications on the Internet of Things," in *Proc. 2nd ACM SIGCOMM Workshop Mobile Cloud Comput.*, 2013, pp. 15–20.
[38] A. Yousefpour, G. Ishigaki, and J. P. Jue, "Fog computing: Towards minimizing delay in the Internet of Things," in *Proc. IEEE Int. Conf. Edge Comput. (EDGE)*, 2017, pp. 17–24.
[39] A. Yousefpour, G. Ishigaki, R. Gour, and J. P. Jue, "On reducing IoT service delay via fog offloading," *IEEE Internet Things J.*, vol. 5, no. 2, pp. 998–1010, Apr. 2018.
[40] S. Sarkar, S. Chatterjee, and S. Misra, "Assessment of the suitability of fog computing in the context of Internet of Things," *IEEE Trans. Cloud Comput.*, vol. 6, no. 1, pp. 46–59, Jan.–Mar. 2018.
[41] 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.
[42] O. O. Ajibola, T. E. H. El-Gorashi, and J. M. H. Elmirghani, "Network topologies for composable data centers," *IEEE Access*, vol. 9, pp. 120955–120984, 2021.
[43] J. Zheng and H. T. Mouftah, "Virtual topology design," in *Proc. Optical WDM Netw.*, 2004, pp. 89–130.
[44] R. Li, D. Zhou, and D. Du, "Satisfiability and integer programming as complementary tools," in *Proc. Asia South Pac. Des. Autom. Conf. (ASP-DAC)*, 2004, pp. 880–883.
[45] "ARC3—Advanced Research Computing." University of Leeds. [Online]. Available: http://arc.leeds.ac.uk/systems/arc3/ (Accessed: Sep. 11, 2018).
[46] "NFX series network services platform," Data Sheet, Juniper Netw., Sunnyvale, CA, USA. Accessed: Mar. 28, 2019. [Online]. Available: https://www.juniper.net/assets/uk/en/local/pdf/datasheets/1000563-en.pdf
[47] "MX series 5G universal routing platforms product description," Data Sheet, Juniper Netw., Sunnyvale, CA, USA. Accessed: Mar. 28, 2019. [Online]. Available: https://www.juniper.net/assets/uk/en/local/pdf/datasheets/1000597-en.pdf
[48] "Cisco ME 4600 Series Optical Network Terminal Data Sheet—Cisco." Cisco. 2017. [Online]. Available: https://www.cisco.com/c/en/us/products/collateral/switches/me-4600-series-multiservice-optical-access-platform/datasheet-c78-730446.html (Accessed: Mar. 28, 2019).

Opeyemi O. Ajibola (Member, IEEE) received the B.Sc. degree (High Hons.) in Electrical and Electronic Engineering from Eastern Mediterranean University, Famagusta, North Cyprus, in 2011, the M.Sc. degree (with distinction) in Digital Communications Networks from University of Leeds, Leeds, UK in 2015 and the PhD degree in Energy Efficient Composable Data Centers from the University of Leeds, Leeds, UK, in 2021. He is currently a Network Architecture Engineer with Ultracell Networks Limited, Leeds, UK and a Visiting Research Fellow in the School of Electronic and Electrical Engineering, University of Leeds, Leeds, UK. From 2012 to 2013, he was a Wireless Solution Sales Engineer with Huawei Technologies, Abuja, Nigeria. He was a Graduate Assistant and Lecturer at Federal University Oye-Ekiti, Ekiti State, Nigeria, from 2014 to 2015 and from 2016 to 2021 respectively. His research interests include composable data center infrastructures, energy efficient cloud and edge (fog) data centers and communication networks, and the Internet of Things.

Taisir E. H. El-Gorashi received the B.S. degree (first-class Hons.) in Electrical and Electronic Engineering from the University of Khartoum, Khartoum, Sudan, in 2004, the M.Sc. degree (with distinction) in Photonic and Communication Systems from the University of Wales, Swansea, UK, in 2005, and the PhD degree in Optical Networking from the University of Leeds, Leeds, UK, in 2010. She is currently a Lecturer in optical networks in the School of Electronic and Electrical Engineering, University of Leeds. Previously, she held a Postdoctoral Research post at the University of Leeds (2010–2014), where she focused on the energy efficiency of optical networks investigating the use of renewable energy in core networks, green IP over WDM networks with datacenters, energy efficient physical topology design, energy efficiency of content distribution networks, distributed cloud computing, network virtualization and big data. In 2012, she was a BT Research Fellow, where she developed energy efficient hybrid wireless-optical broadband access networks and explored the dynamics of TV viewing behavior and program popularity. The energy efficiency techniques developed during her postdoctoral research contributed 3 out of the 8 carefully chosen core network energy efficiency improvement measures recommended by the GreenTouch consortium for every operator network worldwide. Her work led to several invited talks at GreenTouch, Bell Labs, Optical Network Design and Modelling conference, Optical Fiber Communications conference, International Conference on Computer Communications, EU Future Internet Assembly, IEEE Sustainable ICT Summit and IEEE 5G World Forum and collaboration with Nokia and Huawei.

Jaafar M. H. Elmirghani (Fellow, IEEE) is Fellow of the IET, Fellow of the Institute of Physics and is the Director of the Institute of Communication and Power Networks and Professor of Communication Networks and Systems within the School of Electronic and Electrical Engineering, University of Leeds, UK. He joined Leeds in 2007 having been full professor and chair in Optical Communications at the University of Wales Swansea 2000–2007.

He received the BSc in Electrical Engineering, First Class Honours from the University of Khartoum in 1989 and was awarded all 4 prizes in the department for academic distinction. He received the PhD in the synchronization of optical systems and optical receiver design from the University of Huddersfield UK in 1994 and the DSc in Communication Systems and Networks from University of Leeds, UK, in 2012. He co-authored Photonic Switching Technology: Systems and Networks, (Wiley) and has published over 550 papers.

He was Chairman of the IEEE UK and RI Communications Chapter and was Chairman of IEEE Comsoc Transmission Access and Optical Systems Committee and Chairman of IEEE Comsoc Signal Processing and Communication Electronics (SPCE) Committee. He was a member of IEEE ComSoc Technical Activities Council' (TAC), is an editor of IEEE Communications Magazine and is and has been on the technical program committee of 41 IEEE ICC/GLOBECOM conferences between 1995 and 2020 including 19 times as Symposium Chair. He was founding Chair of the Advanced Signal Processing for Communication Symposium which started at IEEE GLOBECOM'99 and has continued since at every ICC and GLOBECOM. Prof. Elmirghani was also founding Chair of the first IEEE ICC/GLOBECOM optical symposium at GLOBECOM'00, the Future Photonic Network Technologies, Architectures and Protocols Symposium. He chaired this Symposium, which continues to date. He was the founding chair of the first Green Track at ICC/GLOBECOM at GLOBECOM 2011, and is Chair of the IEEE Sustainable ICT Initiative, a pan IEEE Societies Initiative responsible for Green ICT activities across IEEE, 2012-present. He has given over 90 invited and keynote talks over the past 15 years.

He received the IEEE Communications Society 2005 Hal Sobol award for exemplary service to meetings and conferences, the IEEE Communications Society 2005 Chapter Achievement award, the University of Wales Swansea inaugural 'Outstanding Research Achievement Award', 2006, the IEEE Communications Society Signal Processing and Communication Electronics outstanding service award, 2009, a best paper award at IEEE ICC'2013, the IEEE Comsoc Transmission Access and Optical Systems outstanding Service award 2015 in recognition of "Leadership and Contributions to the Area of Green Communications", the GreenTouch 1000x award in 2015 for "pioneering research contributions to the field of energy efficiency in telecommunications", the IET 2016 Premium Award for best paper in IET Optoelectronics, shared the 2016 Edison Award in the collective disruption category with a team of 6 from GreenTouch for their joint work on the GreenMeter, the IEEE Communications Society Transmission, Access and Optical Systems technical committee 2020 Outstanding Technical Achievement Award for outstanding contributions to the "energy efficiency of optical communication systems and networks". He was named among the top 2% of scientists in the world by citations in 2019 in Elsevier Scopus, Stanford University database which includes the top 2% of scientists in 22 scientific disciplines and 176 sub-domains. He was elected Fellow of IEEE for "Contributions to Energy-Efficient Communications," 2021.

He is currently an Area Editor of IEEE Journal on Selected Areas in Communications series on Machine Learning for Communications, an editor of IEEE Journal of Lightwave Technology, IET Optoelectronics and Journal of Optical Communications, and was editor of IEEE Communications Surveys and Tutorials and IEEE Journal on Selected Areas in Communications series on Green Communications and Networking. He was Co-Chair of the GreenTouch Wired, Core and Access Networks Working Group, an adviser to the Commonwealth Scholarship Commission, member of the Royal Society International Joint Projects Panel and member of the Engineering and Physical Sciences Research Council (EPSRC) College.

He has been awarded in excess of £30 million in grants to date from EPSRC, the EU and industry and has held prestigious fellowships funded by the Royal Society and by BT. He was an IEEE Comsoc Distinguished Lecturer 2013–2016. He was PI of the £6m EPSRC Intelligent Energy Aware Networks (INTERNET) Programme Grant, 2010–2016 and is currently PI of the EPSRC £6.6m Terabit Bidirectional Multi-user Optical Wireless System (TOWS) for 6G LiFi, 2019–2024. He leads a number of research projects and has research interests in communication networks, wireless and optical communication systems.