2018 Third ACM/IEEE Symposium on Edge Computing

Scalability and Performance Evaluation of Edge Cloud Systems for Latency Constrained Applications

Sumit Maheshwari, Dipankar Raychaudhuri and Ivan Seskar
WINLAB, Rutgers University, North Brunswick, NJ, USA
{*sumitm, ray, seskar*}*@winlab.rutgers.edu*

Francesco Bronzino
Inria, Paris, France
francesco.bronzino@inria.fr

***Abstract*—This paper presents an analysis of the scalability and performance of an edge cloud system designed to support latency-sensitive applications. A system model for geographically dispersed edge clouds is developed by considering an urban area such as Chicago and co-locating edge computing clusters with known Wi-Fi access point locations. The model also allows for provisioning of network bandwidth and processing resources with specified parameters in both edge and the cloud. The model can then be used to determine application response time (sum of network delay, compute queuing and compute processing time), as a function of offered load for different values of edge and core compute resources, and network bandwidth parameters. Numerical results are given for the city-scale scenario under consideration to show key system-level trade-offs between edge cloud and conventional cloud computing. Alternative strategies for routing service requests to edge vs. core cloud clusters are discussed and evaluated. Key conclusions from the study are: (a) the core cloud-only system outperforms the edge-only system having low inter-edge bandwidth, (b) a distributed edge cloud selection scheme can approach the global optimal assignment when the edge has sufficient compute resources and high inter-edge bandwidth, and (c) adding capacity to an existing edge network without increasing the inter-edge bandwidth contributes to network-wide congestion and can reduce system capacity.**

***Keywords*-Cloud Computing, Mobile Edge Cloud, Fog Computing, Real-time Applications, Augmented Reality, System Modeling**

I. Introduction

Edge clouds promise to meet the stringent latency requirements of emerging classes of real time applications such as augmented reality (AR) [1] and virtual reality (VR) [2] by bringing compute, storage and networking resources closer to user devices [3], [4]. Edge compute resources which are strategically placed near the users in the access network do not incur the irreducible propagation delays associated with offloading of compute intensive tasks to a distant data center. In addition, the use of edge computing can also lower wide-area backhaul costs associated with carrying user data back and forth from the central cloud. AR and VR applications enable users to view and interact with virtual objects in real time, hence requiring fast end-to-end delivery of compute services such as image analytics and video rendering. Previous studies [5]–[8] have shown that latency associated with AR or gaming applications can be reduced by migrating some of the delay-sensitive tasks computing tasks to local servers, while maintaining global state in the core cloud.

While edge clouds have significant potential for improved system-level performance, there are some important trade-offs between edge and core clouds that need to be considered. Specifically, core clouds implemented as large-scale data centers [9] have the important advantage of service aggregation from large numbers of users, thus making the traffic volume predictable. Further, service requests entering a large data center can be handled in a close to optimal manner via centralized routing and load balancing [10] algorithms. In contrast, edge clouds are intrinsically local and have a smaller scale and are thus subject to significantly larger fluctuations in offered traffic due to factors such as correlated events and user mobility. In addition, we note that edge computing systems by definition are distributed across multiple edge networks and hence are associated with considerable heterogeneity in bandwidth and compute resources. Moreover, the data center model of centralized control of resources is not applicable to a distributed system [11], [12] implemented across multiple edge network domains, possibly involving a multiplicity of service providers.

A general technology solution for edge clouds will thus require suitable distributed control algorithms and associated control plane protocols necessary for realization. The unique nature of the distributed edge cloud system poses key design challenges such as specification of a control plane for distributed edge, distributed or centralized resource assignment strategies, traffic load balancing, orchestration of computing functions and related network routing of data, mobility management techniques and so on. In order to address these challenges, a simulation based system model is the foundation for understanding performance and evaluating alternative strategies for any of the above design issues.

This paper presents an analysis of the scalability and performance of a general hybrid edge cloud system which supports latency-sensitive applications. The goal is to provide a better understanding of key system design parameters such as the proportion of resources in local cloud vs. data center, fronthaul and backhaul network bandwidth, relative

Research supported under NSF Future Internet Architecture - Next Phase (FIA-NP) Award CNS-134529

978-1-5386-9445-9/18/$31.00 ©2018 IEEE
DOI 10.1109/SEC.2018.00028

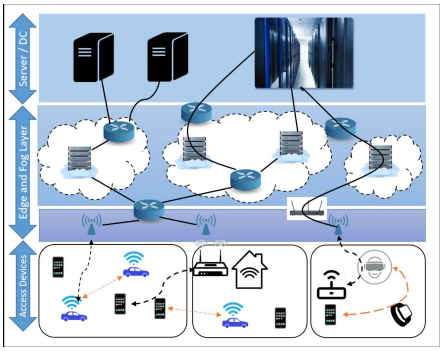

Figure 1. General Multi-tier Edge-cloud Network Architecture

latency/distance of core and edge clouds, and determine their impact on system level metrics such as average response time and service goodput. Using the model described here, we seek answers to the following questions: (a) How much load can an edge cloud network support without affecting the performance of an application; (b) How does the value of the application delay-constraint affects the capacity of the system; (c) What is the impact of offered load and resource distribution on goodput; (d) Under what circumstances can the core cloud perform better than an edge network and vice-versa; and (e) What is the impact of inter-edge (fronthaul) and edge-to-core (backhaul) network bandwidth on system capacity?

We use a simulation model to study a city scale general multi-tier network as shown in Fig. 1 containing both edge and central cloud servers. The model is used to obtain system capacity and response time for an augmented reality application while analyzing the impact of key parameters resource distribution and fronthaul/backhaul bandwidth. A general optimization framework for the distributed system is proposed and compared with distributed algorithm approaches. The rest of paper is organized as follows. Section II demonstrates the augmented reality application with two use-cases and discusses the need of edge clouds to fulfill their low-latency requirements. Section III details the system model with an emphasis on system design, and performance model to analyze edge clouds using a city scale network including models for application, compute and latency. A baseline distributed resource allocation approach for selecting an edge cloud for an AR application is also detailed in Section III. Section IV presents the performance evaluation of the baseline approach. Section V proposes and evaluates a capacity enhancement heuristic (ECON) for real-time applications. Numerical results to compare ECON and the baseline are given in Section VI. Section VII provides related work in the field and finally, Section VIII concludes the paper.

II. Augmented Reality and Edge Clouds

Augmented reality is gaining popularity in numerous fields such as healthcare, visualization, entertainment and education. Most of the commercially available AR devices like Atheer AiR [13], Microsoft Hololens [14] and Google Glass [15] have limited power, storage and on-chip computation capabilities for example currently Hololens has storage ∼64 GB and RAM ∼2GB. In turn, these devices often rely upon offloading storage as well as compute to an architecturally centralized cloud server while ensuring application response time.

The Quality of Experience (QoE) perceived by a user running an AR application using cloud services is a complex combination of network bandwidth, network traffic and compute capabilities of the cloud. First, the bandwidth from end–user to a cloud data center is the minimum bandwidth available across all the hops in the network path, which could be significant when cloud is located far from the user. Second, the network traffic depends upon the network load and congestion, and varies for each individual local network. Edge cloud computing (denoted as ”edge” in the following discussions) promises to alleviate the shortcomings of the cloud server by bringing computation, networking and storage closer to the user and providing fast response, context awareness and mobility support [16]. Therefore, edge computing can be viewed as having the same centralized cloud resources scattered at the mobile network edge and accessed through fast Wi-Fi or 5G access networks. This approach has the potential to provide tightly bounded service response time thereby creating a geographically distributed heterogeneous computing and communication system.

Edge computing does not replace but complements the cloud infrastructure as edge clouds are resource limited in terms of bandwidth and compute. The multifaceted edge system therefore must be studied in conjunction with the existing core cloud for different user requirements, application types, edge assignments and QoS constraints. Thus, for a resource constrained system it is required to allocate resources per request while taking system capacity into consideration. This leads to a nonlinear optimization problem [17] due to multiple factors affecting the capacity including but not limited to network bandwidth, resource availability and application type. In order to understand the capacity constraints of a hybrid edge cloud system for a latency sensitive application, we first, analyze the system taking the AR application as an example and later generalize to other applications.

A. Use Case Scenario

Figure 2(a) shows the process flow of our implementation of a demo AR application using Microsoft Hololens. A client sends a continuous video stream to the edge server which processes the information based upon application type and returns output to the client. The video stream (30 fps) is processed by OpenCV [18] 3.3 running on Intel i7 CPU 980, 3.33GHz and 15GB RAM taking ∼20 ms time for processing each frame. The edge server is connected to the

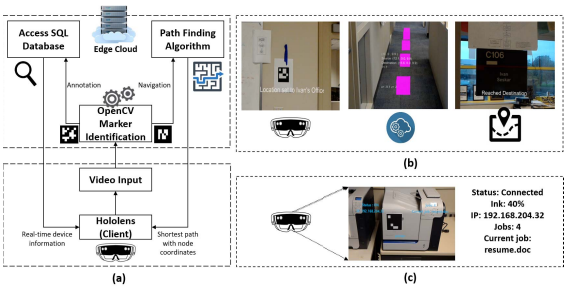

Figure 2. AR Use-case Scenario Set-up: (a) AR Application Flow (b) Smart Meeting Application using Indoor Navigation and (c) Annotation based Assistance

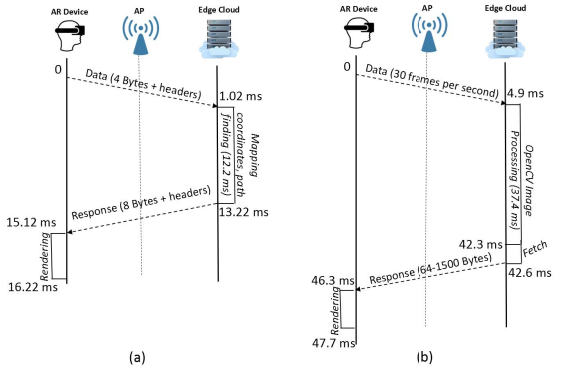

Figure 3. Timing Diagram for the AR Applications: (a) Smart Meeting Application using Indoor Navigation and (b) Annotation based Assistance.

client in two hops: (i) edge to first hop router (bandwidth: 932 Mbps) and (ii) router to Hololens (bandwidth: 54 Mbps). The following use-cases are evaluated.

Smart Navigation. A user enters a building. The edge in the building has her contextual information from calender entries and GPS. As shown in Fig. 2(b) the user is navigated to meet a person in the building using a set of cubes appearing on the device as she moves. Achievable latency is critical here because real-time activities of the user can be disrupted by late arrival of AR information.

Annotation based assistance. In this scenario, a user looks at an object having a set marker through Hololens with an intention to get supplementary information about the object. In Fig. 2(c), user looks at the printer and the status, ink level, number and current jobs are annotated on the user's display.

B. Application Flow Timing Diagram

Figures 3(a) and (b) show (not to the scale) timing diagrams of a packet flow in the system for smart meeting and annotation based assistance application respectively. The network delay in both the cases is kept below 10 ms by deploying edge cloud services a single hop away from the AP. In both the scenarios, the processing delay, path finding in the navigation and OpenCV image processing in the annotation application, can be a major bottleneck. The following techniques are used in our implementation to lower the total response time as compared to the traditional core cloud based services: (i) reduction of network latency via higher bandwidth and closer edge cloud service; (ii) passing minimum processed information to the client such as end-to-end coordinates (8 Bytes) per query in case of the navigation and 64-1500 Bytes per frame processed for the annotation application, and (iii) offloading multiple tasks to the edge cloud to minimize local processing at the UE. The AR implementation serves as a guide to the parameters used in the system model described in the next section, which assumes a low-latency requirement (< 50 ms) to run AR applications with acceptable subjective quality [8].

Using our deployed AR applications, this section confirms that: (a) the total application latency can be brought down by reducing the number of hops and increasing available access bandwidth, and (b) although edge cloud lowers the network latency, application processing latency contributes significantly to the total latency for AR applications.

III. System Model

A. System Design

The system diagram of the hybrid edge cloud under consideration is shown in Fig. 4. Each AP is equipped with an edge cloud with a configurable compute resource capacity. In general, a compute resource represents a machine or a group of machines (cluster) also known as cloud or edge rack. A rack has limited capacity to support users for their computational requirements. For instance, an AR application requires computation to process video/image stream and receive their response back from the server. The edge rack in our design has maximum five processors each having 3.33 GIPS processing speed. The central cloud server is placed at Salem, Oregon (OR; location chosen to relate with commercially available central clouds) which again has a configurable capacity. The compute capacity is defined as the number of servers available at the edge cloud and/or at the central cloud. The inter-edge bandwidth is varied from 1 Gbps to 100 Gbps and AP-Cloud bandwidth from 10 Gbps to 500 Gbps. The special case of unconstrained inter-edge and AP-cloud bandwidth is also considered. The central controller has the capability to collect network and compute parameters from all the edge clouds and the core cloud. The system design parameters are listed in Table I.

In this study, the total amount of compute available at the edge clouds and core cloud is assumed to be fixed. This assumption holding the compute cost constant allows us to fairly analyze the impact of varying other key system parameters such as % of edge servers or core/edge bandwidth. In our simulation, we increase the resource density of already deployed edge clouds by removing and redistributing compute resources from the central cloud thereby keeping the overall compute resources for the whole system unchanged.

We use Chicago, the third most populous city in US, as a test-case considering locations of 11,00 WiFi APs [19] as shown in Fig. 5. The number of hops from Chicago to

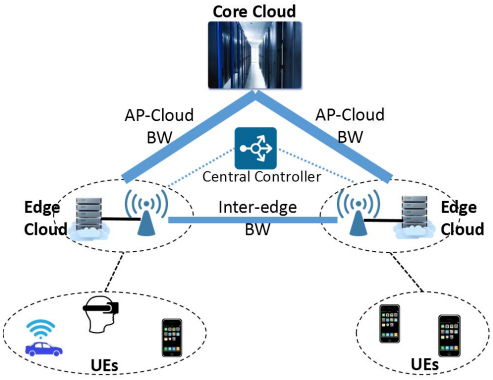

Figure 4. Hybrid Edge Cloud System Diagram

Table I
SYSTEM DESIGN PARAMETERS

Parameter	Value/Range
AR Bit Rate	42.48 or 66.24 Mbps
AP-Cloud Bandwidth	10–500 Gbps
Inter-edge Bandwidth	1-100 Gbps
Core Cloud Resources	0, 20, 40, 60 or 100%
Edge Cloud Resources	0, 20, 40, 60 or 100%
Core Cluster	0-5500 servers
Edge Clusters	0-5500 servers
AR Latency Requirements	50-100 ms

OR varies from 10 to 20 (including switches) and takes around 5-6 hops to reach the cloud server gateway whereas the average latency in US ranges from 13 ms to 106 ms [20] based on a simple ping test of 64 bytes packet from various locations. The mean local delay in Oregon is as low as 13 ms. It is to be noted that the AR application's bit rate increases rapidly with resolution for instance a 160x120 pixels video needs around 1.7 Mbps whereas a 640x480 pixels video requires 27 Mbps continuous uplink bandwidth (assuming 30 fps, 24 bit per pixel) which goes up to 432 Mbps for 1920x1080 video. For annotation based assistance, assuming each frame is processed for information, relevant data is queried from the database and sent to the user, the required downlink bandwidth varies from 54–600 Mbps. The response from the server is sent to the UE as multiple packets (100–1500 Bytes) per frame processed. The uplink bandwidth is assumed to be from 27–300 Mbps as listed in Table II. For the simulations in this paper, we used 1280x720 and 1026x576 video size chosen randomly for each user and maintained throughout. Note that the uplink bandwidth requirement for an AR application is more than the download bandwidth due to its uplink video/downlink processed information characteristic which is quite different from most web traffic today. We model the network based on the type of application and its latency requirement.

We run an AR application at the UE which sends a video stream to the server while server computes the contextual information and sends back the output to the user. The

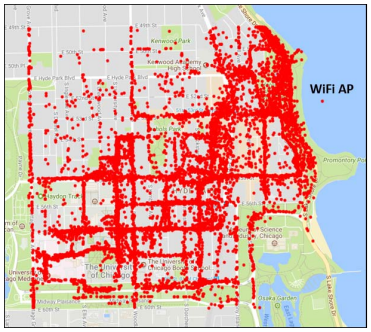

Figure 5. Wi-Fi APs Placement in Chicago City

application is annotation-based assistance using AR wherein a user gets information about surrounding annotated on his AR device as described in Section II. Annotation-based assistance can be used in various application scenarios. For example, a policeman looks at a license plate of a car while driving and the information about the owner gets displayed on the device. The license plate can also be run against a list of stolen car and can be immediately reported to the policeman.

Table II
SIMULATION PARAMETERS

Parameter	Value
Area	5.18 km^2
Number of APs	1.1K
Number of Users	55K
Distribution of Users	Random
Bandwidth (Uplink)	27, 150 and 300 Mbps
Bandwidth (downlink)	54, 300 and 600 Mbps
Packet Size	1500 Bytes
Edge Resources (baseline)	5 Machines
α	2
β	1
γ	0.1
δ	1
ρ	0.9
w	0.5
p	10

B. Performance Model

In this section, we describe system modeling aimed at evaluating user performance and system capacity as a function of key design parameters. A multi-tier edge-cloud system as shown in Fig. 4 can be divided into user, network (data and control) and computation plane. Our system design is a hierarchical composition of compute and network elements. The computation at edge or cloud is similar in functionality but different in terms of resources availability as the core cloud has a single big pool of shared resources while each edge cloud has limited resources closer to the user. The following discussion presents application, compute and latency modeling.

1) Application: In our model, the application is defined by a four tuple $< V, G, S, L >$ where V denotes the computational task per unit time ranging from $[1, n], n \in \mathbb{Z}^+$. Each AR application requires these tasks to be completed within a specified real-time threshold latency in order to be useful to the AR application. In case a task is not completed within the application latency threshold, the goodput of system goes down. G denotes the geolocation of the UE. A city is considered to be a collection of G_i blocks (assume as cells of a cellular network), $i \in [1, N]$ where N is the total number of geographical blocks. For simplicity, we divide the geographical area into square G_i's. Analyzing the users served by each block provides us meaningful information if we need to upgrade the capacity of an edge cloud in the block. Binary $S \in \{0, 1\}$ denotes the availability of the edge cloud in the geographical area G of a user. Unavailability of an edge cloud may mean that there is no physical edge cloud present or the edge cloud of that region has run out of capacity in which case, a neighboring edge cloud can be chosen or the user can be routed to the central cloud. For delay-tolerant applications, routing a user to the central cloud frees resources at the edge to serve latency sensitive applications. Finally, $L \in (0, d_{\max})$ represents the maximum tolerable latency for the said application.

2) Compute: The delay due to computation is modeled using a multi-server queuing model. The edge cloud is like a mini data center where tasks arrive from geographically distributed users, processed by the available resources in the edge cloud and depart. Therefore, as the number of transactions in the system increase when the system load rises these tasks are queued till they are processed. This scenario can be best represented by employing an M/M/C queuing model [21]. Each edge or central cloud processes multiple service requests in a work-conserving FCFS queue with assumed infinite buffers. The overall latency is dependent on the arrival rate λ, service rate μ and the number of servers c. It can be noted that as the system computation power is constant, increasing capacity at the edge will mean removing equivalent resources from the central cloud implying a rise in queuing delay at the cloud. As the system load increases, the arrival rate, λ, rises thereby increasing the total computing latency per task V as $d_{comp} = 1/(c\mu - \lambda)$ where $\mu = f/K$, f being the rated speed in instructions per second and K is number of instructions required per task.

For a given set of static users, the system load is proportional to the number of active users and the rate of application requests per second. In our model, we assume 55K users and Load=1 is defined as 10% of the the users are running the application. Load=10 implies that all 55K users are running the AR application 100% of the time. In general, average time spent by a task in the server is the sum of transmission delay, queuing delay and processing delay, which is calculated using the M/M/c queuing model as given below in Eq. (1-3).

$$d_{node} = W + \frac{1}{\mu} + t_{tx} = P_Q * \frac{\rho}{\lambda(1-\rho)} + \frac{1}{\mu} + t_{tx} \quad (1)$$

$$P_Q = \frac{(c\rho)^c}{c!}\frac{1}{1-\rho}p_0 \quad (2)$$

$$p_0 = \left[\sum_{k=0}^{c}\frac{(c\rho)^k}{k!} + \frac{(c\rho)^c}{c!}\frac{1}{1-\rho}\right]^{-1} \quad (3)$$

Here, d_{node} is the total time spent by a task V at the edge cloud or the core cloud, W is the wait time in the queue, P_Q is the queuing probability, ρ is the server utilization, c are number of servers at each edge or total server at the cloud, p_0 is the initial probability, and t_{tx} is the average transmission time for a task at an edge as noted in [22] given by $t_{tx} = (N * r)\sum_{j=1}^{\infty} j(1-\Phi)^{(j-1)}\Phi$, where Φ is the non-outage probability of a link implying available bandwidth for a task, r are the number of tasks per user per second and N is the total number of users in the system. In view of shared bandwidth on inter-edge links, t_{tx} can be simplified as b_{link}/r_{users} where b_{link} is the total bandwidth of a link and r_{users} are number of total tasks run by all the users at an edge. For large c, to avoid involved calculations in Eq. (2), we split cloud computing resources into set of uniform clusters where a selected cluster is one serving the lowest number of concurrent tasks.

3) Latency: The overall latency of an application has several components including irreducible propagation delay, the transmission delay, routing node delays and the cloud processing time. For a core cloud server, which carries aggregated traffic, there is also a Software Defined Networking (SDN) switching latency. As the number of users increase in a geographical region, the bandwidth is shared among them costing more transmission delay. For a cloud only model when there are no edge servers, the total cloud latency can be stated as:

$$L_{cloud} = (\alpha+\delta)*D_{\min(UE,APs)}+(\beta+\gamma)*D_{AP-cloud}+d_{node} \quad (4)$$

Eq. 4 shows that a closest AP is chosen to route a user to the cloud. Here, α and δ are the proportionality constants for uplink and downlink bandwidth from UE to AP link respectively, and β and γ are the similar factors for AP to cloud uplink and downlink bandwidth respectively. $D_{\min(UE,APs)}$ is distance from UE to nearest AP and $D_{AP-cloud}$ is the distance from AP to the central cloud. It is noted that the uplink bandwidth usage for the AR application is much higher than that of the downlink as mentioned earlier. When resources are available at the edge, the total edge latency can be represented as:

$$L_{edge} = (\alpha + \delta) * D_{\min(UE,APs)} + d_{node} + d_s \quad (5)$$

In Eq. 5, $d_s \geq 0$ is the control plane switching latency from an edge at AP to another AP's edge in case of unavailable

resources which is assumed to be between 1–5ms. The response time for an application is the sum of transmission delay, propagation delay, switching delay (if any), queuing delay and computation delays in both the cases.

A core cloud-only system is defined as one with no edge cloud available. The edge-only system does not have any core cloud and if the load exceeds the available computational resources, a request is queued until it is processed. We also consider hybrids of core and edge based on the percentage parameter that splits computing resources between the two.

C. Edge Selection for an AR Application

Edge selection in a system for a given traffic load can be achieved using multiple approaches depending upon whether the system has centralized or distributed control. The network routing information that is available to all the routers can be used to deliver the service request to the nearest edge cloud — the edge cloud then independently decide to serve the request based upon resource availability or can route the user to the central cloud. A queuing model (M/M/c) is used to predict the estimated service time for a request apart from networking delays (control plane), propagation delays and transmission delays (available bandwidth). This approach works well for scenarios with evenly distributed users and network resources. However, this simple nearest edge cloud routing strategy does not work well when the user distribution is not geographically uniform ascertained by our simulation showing only 10% improvement in the average system response time as compared to a cloud-only system.

An alternative distributed approach improves upon simple anycast by having routers maintain compute resource availability states of neighboring edge clouds. This may involve the use of overlay protocols to exchange cloud state in a distributed manner [23], [24]. A user is routed to the nearest edge first which makes one of the following decisions: (i) serve the request, (ii) route to a neighboring edge with available resources, or (iii) route to the central cloud. The decision at the edge is based upon application requirement and traffic load. For an AR application, the decision metric selects the closest edge to the UE which can serve the UE in $L_{edge} \leq d_{\max}$. The algorithm for this approach is as detailed below.

D. Baseline Approach

Algorithm 1 shows the pseudo-code for the baseline edge cloud selection approach adopted in our study. The algorithm is invoked whenever the default edge cloud is unable to serve the user's demand (line: 2). It then scans the states of neighboring edges to find the best edge which can serve the user within the specified latency threshold. This approach relies upon shared resource and bandwidth information among neighbors. The list of neighbors is defined as p closest edge clouds from the current edge location. For finite p the order of state update messages to be exchanged is $\sim N * p^2$ where N is the number of edge clouds, and is thus an acceptable overhead for small to moderate values of p.

Algorithm 1: Finding neighboring edge with available resources for an AR application

```
1 function AvailableNeighbor (a, b);
  Input  : Neighbor resource and bandwidth s_i and b_i
  Output: TorF
2 Condition: TotalDelay_Edge ≥ delay_th
3 while(NeighborEdge)
4 if TotalDelay_NeighborEdge_i ≤ delay_th then
5 |   return TRUE;
6 else
7 |   return FALSE;
8 end
```

This section detailed our system and performance model. A baseline algorithm which scans the states of neighboring edge clouds to find the best edge which can serve the user within the specified latency threshold is developed. Next section evaluates the performance of baseline algorithm.

IV. Performance Evaluation of Baseline System

In this section we discuss the capacity of different edge cloud systems with respect to traffic load, resource distribution and inter-edge bandwidth. Consider a system with following compute resources: (i) core cloud only, (ii) edge cloud only, and (iii) core cloud plus edge cloud, where in each case, the total amount of resources are same. Major system parameters used in the simulation are summarized in Table II.

A. Impact of Network Bandwidth Parameters

Figure 6 shows the average response time for core cloud only and edge only networks for different system load when there is no limit on inter-edge and edge-cloud bandwidth. As there is no bandwidth limitation, the queuing delay dominates and crosses the 50 ms response time threshold after the system load is more than 60% for edge only system without bandwidth constraints. In the case when the core cloud has infinite capacity we observed that the network latency affects the total application response time.

Figure 7 illustrates the impact of constraint bandwidth AP-cloud system on the average response time. Here, the total bandwidth limit is set between edge network and the core cloud cluster. For a 500 Gbps AP-cloud bandwidth, for given system, the average response time compares with that of an unconstrained bandwidth case while for 50 Gbps case, it rises exponentially as the load increases. In case of lower bandwidth cases like 10 Gbps and 25 Gbps, the system is unable to handle higher load. As a bandwidth-constrained

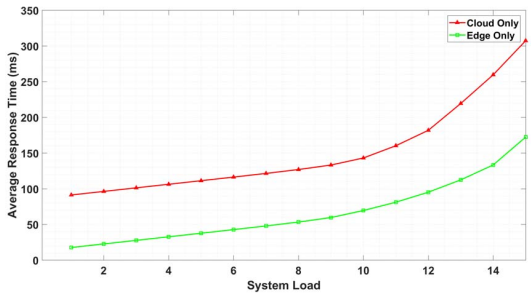

Figure 6. AR Application Average Response Time for Core Cloud only and Edge only Networks with Increasing System Load

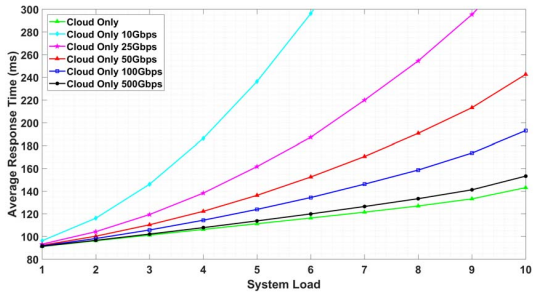

Figure 7. AR Application Average Response Time for Core Cloud only System with Increasing System Load and Different Uplink Bandwidth

cloud system cannot compete with an edge-only system in terms of response time, further discussions in this paper will assume a bandwidth-unconstrained cloud.

Figure 8(a) plots the average response time for the core cloud as well as edge only system with different inter-edge bandwidth. On one hand, the extreme fronthaul bandwidth of 100 Gbps edge-only compares with the unconstrained bandwidth edge-only system and therefore all the edge resources are utilized. On the other hand, after the system fills up at Load=7, core cloud only system outperforms the edge only system with 1 Gbps inter-edge bandwidth. The reason is that for the baseline case, when an edge fills up the capacity, it routes the request to a neighboring edge utilizing inter-edge bandwidth. As the finite inter-edge bandwidth is split between multiple application flows, the propagation delay and queuing delay rise which in turn increases the average response time for higher load. In the baseline approach, the edge decides whether to send the request to a neighboring edge or to the central cloud. For 1 Gbps inter-edge bandwidth, the average response time for Load=1 is as low as 30 ms while for Load=10 case, it rises to 170 ms as the bandwidth exhausts and queuing delay rises. A delay more than 100 ms is unsuitable for most of the AR applications. As the bandwidth doubles, for Load=10 case, the average response time is $\sim$120 ms. Increasing bandwidth lowers the average response time for a completely loaded system but beyond 10 Gbps there is no significant advantage

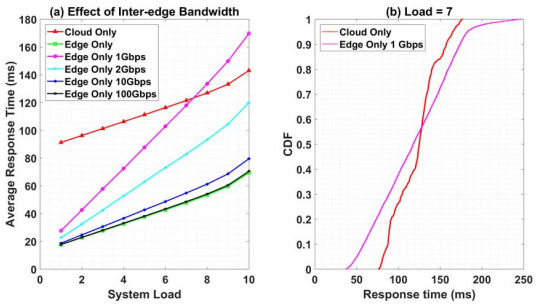

Figure 8. Average Response Time Comparison for Core Cloud and Edge Only System, with Different Load and Inter-edge Bandwidth for Baseline

visible for the baseline case as there are still significant queuing delays for a loaded edge at an AP (or neighboring AP). After a load point, there is no dip in response time irrespective of how good the fronthaul connectivity between edges is. In this case, there is a crossover around Load=7 so we compare the CDF of core cloud only and edge-only with the 1 Gbps case in Fig. 8(b). A linear rise in response time can be observed for the static load case implying that the inter-edge bandwidth of 1 Gbps is insufficient to run such a heavily loaded system.

B. Impact of Resource Distribution

In this subsection, we analyze the impact of the compute resource distribution between the core cloud and edge cloud on the average response time. There are a total of 5.5K processors each having 3.33 GIPS speed, available as compute resources which are equivalent to 1.1K full edge racks. Figure 9 shows the baseline latency performance for a core cloud-only system, edge-only system and cloud-edge (CE) system for the simulation parameters listed in Table II. CE80-20 implies that 80% compute resources are available at the cloud and 20% are placed at the edge near the APs and so on. The inter-edge bandwidth has no limitation in this case. As expected, the edge only system outperforms irrespective of load. As the resources are moved from central cloud to the edge, the response time CDF moves towards the left close to the edge-only system. When the CE system does not find resources available at the neighboring edge using Algorithm 1, the request is routed to the core cloud. Therefore in each of these cases, except for the edge-only case, a few requests are bound to have response time as close as core cloud-only case. As expected, increasing resources at the edge brings response time down in the case of unconstrained bandwidth. Next we consider more realistic scenarios with constrained bandwidth.

Figures 10(a) and (b) compare average response time in CE28 and CE82 for the baseline with respect to inter-edge bandwidth and load respectively. Response times for inter-edge bandwidth of 10, 50 and 100 Gbps are close to each other for all the load cases for both scenarios. This

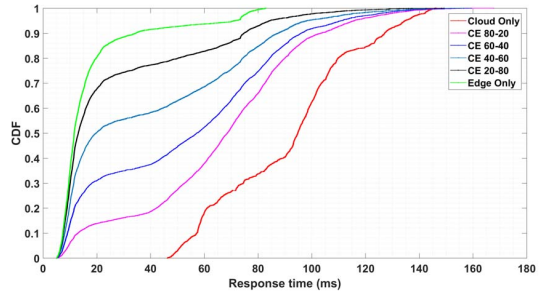

Figure 9. Response Time CDF for Different Resource Distribution for Baseline without Inter-edge Bandwidth Limit

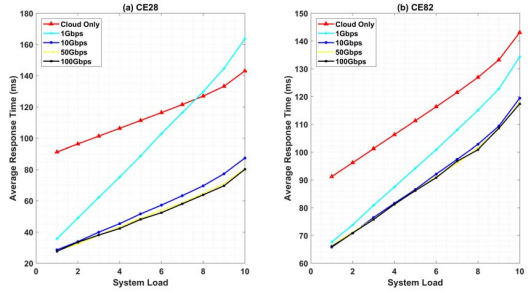

Figure 10. Average Response Time for Edge Cloud System for Different Load, Resource Distribution and Inter-edge Bandwidth for Baseline

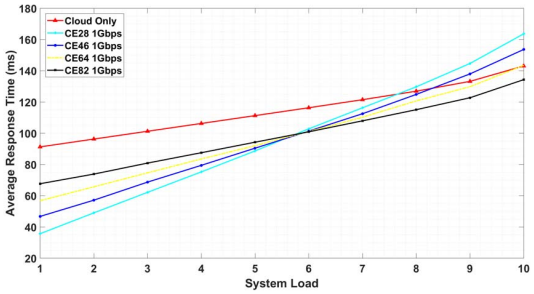

Figure 11. Average Response Time for Edge Cloud System with Different Load and Resource Distribution for Baseline. Inter-edge Bandwidth=1Gbps.

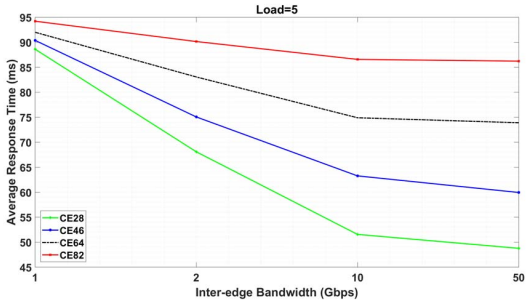

Figure 12. Average Response Time for Edge Cloud System with Different Resource Distribution and Inter-edge Bandwidth for Baseline. Load=5.

implies that increasing inter-edge bandwidth indefinitely cannot improve the system performance when using the simple scheme of filling neighboring edge resources. Figure 10(a) also highlights the fact that when edge resources are higher than the core cloud for a low inter-edge bandwidth, beyond a load point, the core cloud-only system performs better. This means that for a highly loaded system, if fast edge connectivity is unavailable, it is better to use the core cloud.

C. Impact of AR Application Traffic Parameters

Figure 11 establishes the fact that inter-edge bandwidth plays a crucial role in the system. For the CE28 case, when the cloud-edge resource distribution is 20%-80% and inter-edge bandwidth is 1 Gbps, average response time increases at a faster rate than that of the CE82 case. The reason is that in the baseline scenario for CE28, an edge might be able to find a neighbor with available capacity but the connectivity is not sufficient to reach to that neighbor. In the case of lower or no edge resources, the core cloud is immediately favored and therefore performs better than the edge cloud scenario as can be observed from the crossover point at Load=8 case.

One more point of interest in Fig. 11 is between Load=5 and Load=6 where all the CE cases intersect. Figure 12 shows the average response time with different inter-edge bandwidth and resource distribution for baseline when Load=5. Here, for the CE82 case, increasing inter-edge bandwidth does not boost the system performance as compared to the CE28 case because for the low edge resources case, increasing inter-edge bandwidth cannot decrease the processing delays at the edge. For a system with high edge resources, a higher inter-edge bandwidth is therefore needed to maintain AR performance.

Similarly, for the Load=6 case, Fig. 13 plots average response time vs. resource distribution for different inter-edge bandwidths. Again, for a 50 Gbps inter-edge bandwidth system, a faster drop in the average response time can be observed for the CE28 case when 80% resources are at the edge. For a 1 Gbps inter-edge bandwidth system, the average response time is slightly higher for the CE28 system than for the CE46 system.

Using our designed system and performance model, we make following observations for the baseline scenario: (a) for unconstrained compute resources, the edge cloud continues to perform better than the core cloud due to its vicinity to the users (lower network latency), (b) increasing core network bandwidth beyond a load point does not lower the total application latency as the compute latency takes over, (c) for higher system load, the propagation delay and queuing delay rise because finite inter-edge bandwidth is divided among multiple application flows, (d) indefinitely increasing fronthaul edge cloud connectivity does not improve the response time after a load level, and (e) for lower

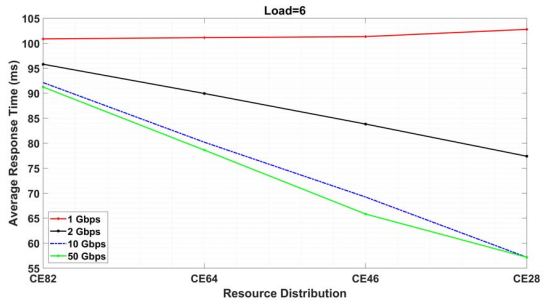

Figure 13. Average Response Time for Edge Cloud System with Different Resource Distribution and Inter-edge Bandwidth for Baseline. Load=6.

inter-edge bandwidth case, distributing more resources at the edge clouds only worsens the application performance.

V. ECON: Enhanced Capacity Edge Cloud Network

The baseline approach considered in Section IV relies on distributed control to select the best available neighboring edge cloud which might be sub-optimal in terms of overall system capacity. A more general approach is to select an edge cloud based upon global information about network and compute resources available at a logically centralized point such as an SDN controller. The idea is to use the complete network view before assigning an application/user to an edge cloud or deciding to route it to the core cloud. We call this approach Enhanced Capacity Edge Cloud Network (ECON). This section describes the ECON method and compares its performance with the baseline method.

Definition 1: An edge or cloud is *"usable"* for a request *i* if the latency L_i^a for the user running an application *a* is below the latency threshold for given application L_{Th}^a i.e. $L_i^a \leq L_{Th}^a$. Here, L_i^a is simply equal to L_{cloud} or L_{edge} with different d_{node} and d_s.

A *"usable"* server is best for a user request in terms of service quality whereas the overall system capacity might not be optimal with this assignment. For example consider a user's application latency threshold 110 ms which may be assigned to an edge server serving request within 30 ms. This assignment will hamper performance of another needy user who required 35 ms latency but cannot be accommodated due to unavailable resources at the edge.

Definition 2: *"delay-constraint (%)"* of an edge-cloud system is defined as the number of requests out of hundred served below the application response time threshold, L_{Th}^a. For a specific value of L_{Th}^a, the delay-constraint can also be interpreted as system capacity. For instance, a delay-constraint of 10% for a 15 ms threshold implies that system can accommodate only 10% of the total requests and 90% requests will only consume resources to lower the goodput. This means for 90% of the requests, the assigned edge resources are *"not usable"*.

Percentage delay-constraint, $C = \frac{n_{Th}}{N} * 100$, where n_{Th} are requests served within threshold response time and N are the total number of requests in the system. A system with high C for a threshold is required to run latency sensitive applications.

A. "Usable" Edge-Cloud Optimization

Assigning requests to a *"usable"* server is similar to capacity optimization of an edge-cloud system for given compute as well network resources and application delay fulfillment. This problem is equivalent to the maximum cardinal bin packing and hence is NP-hard [25], [26]. We can model the global optimization to maximize usable server *s* for *N* requests, where each request *i* is assigned to the server *s*, as:

$$\max_{s} \sum_{n \in N} I_{\{s_n > 0\}} \tag{6}$$

subject to:

$$L_i^a(s) \leq L_{Th}^a, \forall s_n > 0, n \in N \tag{7}$$

$I_{\{s_n>0\}}$ being the indicator function with values 1 or 0 depending upon if such a server is available or not for a given request which means if it can serve the request in application response time threshold. Mapping users to *"usable"* server is NP-hard problem as explained earlier thus requiring an alternative approach.

The total average processing delay, d_{comp}, at the cloud or edge, comprise of a waiting delay in the queue and a processing delay associated to the type of application. At each node, there is a transmission time, t_{tx} associated with each task V, adding which to d_{comp} provides total time, d_{node}, spent at a server. Therefore, for such a system, we can formulate Eq. (6) as minimizing d_{node} of the system for all the users, while compromising on the optimality, instead of a *"usable"* server problem as follows:

$$P_1 : \min \sum_{i=1}^{M} (\sum_{j=1}^{N} d_{proc}^{j,i} + d_{tx}^{i} + d_s^i) \tag{8}$$

subject to:

$$L_j^a \leq L_{Th}^a, \forall j \in N \tag{9}$$

$$b_{\min}^{i,up} \leq b_i^a \leq b_{\max}^{i,up}, \forall i \in M \tag{10}$$

$$b_{\min}^{i,down} \leq b_i^a \leq b_{\max}^{i,down}, \forall i \in M \tag{11}$$

$$\sum_{i=1}^{M} c_i \leq C \tag{12}$$

Equation 8 defines the optimization problem with Eq. (9) as delay constraint, Eq. (10) and Eq. (11) as bandwidth constraints for uplink and downlink each user application and, Eq. (12) as capacity constraint of each node respectively. As explained earlier, b_i^a can be computed as b_i / r_{edge}. Again,

the problem is similar to maximum cardinality bin packing problem and is NP-hard. Therefore, to find the *"usable"* server, we need to fix a user to a nearby edge and find the Pareto optimal edge for the next user sequentially satisfying the application latency constraint. This can be done by omitting the switching delay. Therefore, the problem can be simplified as (with same constraints as above) follows assuming d^i_{tx} constraint is satisfied by bandwidth splitting for each request.

$$P_2 : \min \sum_{i=1}^{M} \sum_{j=1}^{N} d^{j,i}_{proc} \tag{13}$$

Equation 13 establishes that for a latency sensitive AR application, finding the *"usable"* server for a user means we need to place the task to a server which is *nearby* to the user in strict network sense having low load, latency and high available bandwidth. The delay minimization objective function fills up the edge resources before routing a task to the central cloud. The latency and bandwidth of chosen server are estimated using the exponential moving average: $x_p * w_x + (1 - w_x)\overline{x}$, with w_x as weight factor for x, x_p is the previous value, $\overline{x}$ is the previous average and x is latency or bandwidth parameter. We call this approach ECON and results are compared with the baseline in next section.

VI. ECON vs. Baseline

A. Resource Distribution and Inter-edge Bandwidth

ECON relies upon filling up the edge resources before routing to the central cloud. Figures 14(a) and (b) compare average response time for CE28 and CE82 cases when the inter-edge bandwidth is 1 Gbps. For an edge-favored CE28 scenario in Fig. 14(a), ECON and baseline have similar performance because finding an available resource in ECON is equivalent to finding a neighbor in the baseline which has high probability when edge resources are 80%. When the resources are cloud-favored i.e. CE82 in Fig. 14(b), for a lightly loaded system, ECON performs better as it is able to find the resources anywhere in the network without additional queuing delays at the edge. For a highly loaded system, finding an available edge is more expensive than routing the request to the cloud itself and therefore baseline outperforms ECON in case Load>5.

B. Application Delay Constraints

Figure 15 presents the delay-constraints for unlimited fronthaul bandwidth edge-cloud system for CE82 case when Load=1. As application latency threshold increases, delay-constraint rises meaning if an application has a latency threshold of 100ms, about 60% requests can be fulfilled by the cloud-only system whereas the edge-only system will be able to fulfill all the requests. As shown in the plot, without inter-edge bandwidth limits, ECON performs better

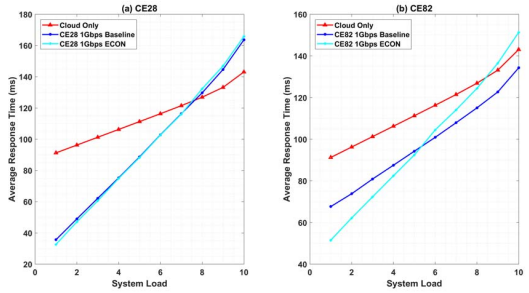

Figure 14. Average Response Time Comparison for ECON and Baseline, for Different Load and 1 Gbps Inter-edge Bandwidth

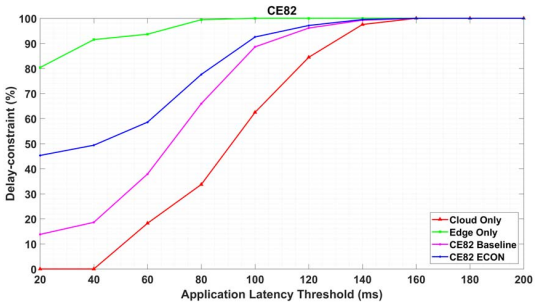

Figure 15. Impact of Application Latency Threshold on Delay-constraint Percentage for ECON and Baseline without Inter-edge Bandwidth Constraints

than the baseline as it fills up maximum edge resources before routing any request to central cloud.

Figures 16(a) and (b) compare a 1 Gbps edge-favored (CE28) system with Load=1 and Load=10. For a lightly loaded system when the edge cloud has more resources, ECON and baseline have similar performance as both of these schemes are able to find an available resource at the edge and 1 Gbps bandwidth is sufficient to route the request to a neighboring edge. In the case of a heavy load scenario, both of these schemes again have similar performance but the core cloud-only system is able to serve more requests than any of these schemes when the application latency threshold is more than 140 ms. This study shows that for elastic applications such as email, a cloud-only system is sufficient and can even perform better when compared to an edge-cloud system with low bandwidth. Also, for the low bandwidth scenario, routing to the cloud is more helpful in improving application latency performance than maximizing usage of edge clouds as illustrated by Fig. 16(b) as baseline outperforms ECON when application latency threshold is more than 100 ms.

Figures 17(a) and (b) show the difference between ECON and baseline delay-constraint performance for Load=1 and Load=10 for CE82 case. For a lightly loaded system, and lower available inter-edge bandwidth, ECON is able to fill up edge clouds before routing to the cloud and therefore

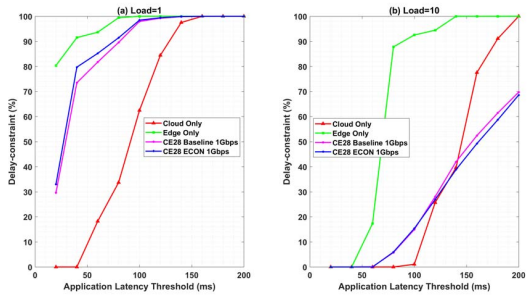

Figure 16. Edge Cloud System Capacity for Different Load and Edge Favored Resources (Inter-edge BW=1 Gbps)

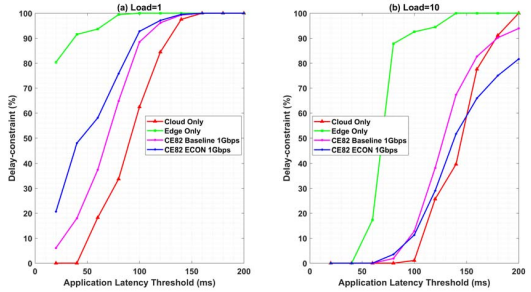

Figure 17. Edge Cloud System Capacity for Different Load and Cloud Favored Resources (Inter-edge BW=1 Gbps)

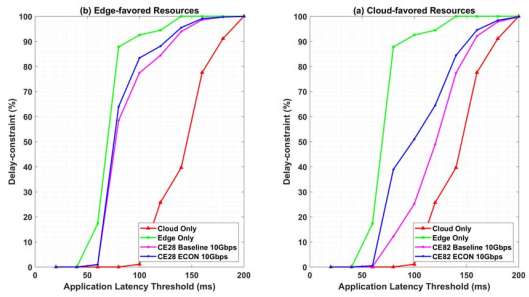

Figure 18. ECON and Baseline Comparison for Edge and Cloud Favored Resources (Inter-edge BW=10 Gbps)

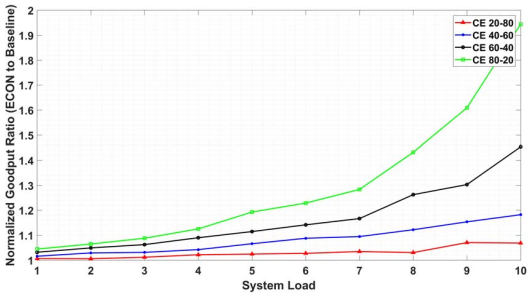

Figure 19. Impact of Load on Goodput Ratio of ECON and Baseline in an Edge Cloud System for Real-time Applications

performs better than baseline. When the load is higher, when ECON tries to fill up all the edge resources which are only 20% here, with 1 Gbps inter-edge bandwidth connectivity, it introduces more transmission delays and therefore the baseline outperforms. In this specific case, the cloud-only system overtakes first ECON and later the baseline case when the application can withstand higher latencies.

1) Edge-favored vs. Cloud-favored: Figures 18(a) and (b) compare edge and cloud favored resources respectively when inter-edge bandwidth is 10 Gbps. Figure 18(a) shows that for an edge-favored case when most of the resources are available at the edge, a baseline neighbor selection scheme performs equally well as ECON which selects the best of all edge resources for the request. For the cloud favored resource case shown in Fig. 18(b), ECON performs better than baseline as each of the edges has sufficient bandwidth to reach a far away available edge resource. Therefore, when sufficient bandwidth is available, it is better to choose an edge even if there are fewer resources available as the queuing time at an edge can be compensated by faster request transfers. On the other hand, if the inter-edge bandwidth is low, instead of trying to maximize edge resource utilization, it is good to send the request to the cloud if the application can withstand the resulting delay.

2) Goodput: As discussed earlier, AR applications are delay sensitive and discard packets which arrive late. Goodput is defined as the number of useful (on time) bits per second delivered to UEs running the AR application. Therefore, even when the system throughput is high, the goodput could remain low due to high proportion of late arrivals. The capacity improvement can be studied by analyzing a geographic block, $G_i's$ level of goodput using our simulation tool. If goodput is lowest in a block, this is an indicative of a need to augment additional edge resources to the serving edge. Figure 19 shows the normalized goodput ratio of ECON and baseline for different resource distribution and load. For an unconstrained inter-edge bandwidth system, the goodput ratio of a cloud-favored system is more than that of an edge-favored one as ECON tries to find the best available edge resource as compared to the neighbor selection baseline scheme. In a cloud-favored system, the edge has minimal resources and therefore each edge requires sufficient bandwidth to transfer requests to other edges which may be far away. The edge-favored system cannot be significantly improved with ECON as there are ample neighboring edges available from the baseline and therefore finding a more optimal edge tends to increase the network delay. Also, as the system load increases, there is a rise in the queuing delay at the edge server and therefore the system performance is similar for ECON as well as baseline in this case.

This section compared baseline scenario with a global edge assignment approach called ECON. We found that: (a) for an edge-favored resource system, ECON and baseline

have similar application response time performance, (b) for a cloud-favored resources and lightly loaded system, ECON performs better than the baseline, (c) maximizing edge clouds usage for lower inter-edge bandwidth hampers the average system response time, and (d) for elastic applications such as email, a cloud-only system is sufficient and can even perform better as compared to an edge-cloud system with low bandwidth.

VII. Related Work

Edge cloud solutions have been proposed for a number of emerging scenarios including Internet of Things (IoT) [27], Cloud of Things (CoT) [28]–[31], health analytics [32] and autonomous driving [33], [34]. The term cloud is generically used to describe a remotely located on-demand computing and networking system along with its typical storage functionality. Architectures such as Mobile Edge Cloud (MEC) [17], [25], fog [35] and edge [36] computing bring these resources close to the user to support faster networking and ultra-low latency applications.

Serving IoT devices using edge clouds is proposed in [37]–[39] with or without virtualization techniques to provide local compute offload, nearby storage, and networking. Real-time applications such as autonomous driving, traffic monitoring/reporting, and online multi-player 3D gaming have also been considered, [8], [40]–[42]. Applications of ICN (Information Centric Networking) have been proposed in [43] as a means to reduce network complexity through named services and content. A three-tier cloud of things (CoT) system is modeled in [44] which identifies edge cloud is a key design element for time-constraint applications. Attempts are also made to provide hierarchical models of edge clouds thereby enabling aggregation capabilities similar to data center networks [45]. Understanding network topology is a critical step in analyzing a cloud or edge network mainly due to effect of routing on latency and throughput. Attempts have been made to characterize the network using geographical properties in [46] using data of autonomous system (ASes) and their relationships, to create a network topology for realistic analysis.

Motivated by faster compute and connectivity needs of newer AR/VR applications, an edge-centric computing is described in [47]. A QoS-aware global optimal edge placement approach is described in [48]. An energy efficient resource allocation strategy is proposed in [49] considering link layer parameters. A small cell based multi-level cloud system is simulated in [50]. Existing literature either relies on a central controller for an optimal edge placement or the use of new network hierarchy to realize improvements in system performance [51], [52]. Studies aimed at determining the overall capacity of a edge cloud system to support multiple applications using a city-scale network are lacking in the existing literature. To the best of our knowledge, this is one of the early attempts to characterize such a hybrid system with respect to edge-cloud resource distribution, inter-edge bandwidth, AP-cloud bandwidth and system load.

VIII. Conclusion

This paper provides a framework for modeling and analyzing capacity of a city-scale hybrid edge cloud system intended to serve augmented reality application with service time constraints. A baseline distributed decision scheme is compared with a centralized decision (ECON) approach for various system load, edge-cloud resource distribution, inter-edge bandwidth and edge-core bandwidth parameters. The results show that a core cloud only system outperforms the edge-only system when inter-edge fronthaul bandwidth is low. The system analysis results provide guidance for selecting right balance between edge and core cloud resources given a specified application delay constraint. We have shown that for the case with higher inter-edge bandwidth and edge computing resources, a distributed edge selection achieves performance close to centralized optimization, whereas with ample core cloud resources and no bandwidth constraints, ECON provides a lower average response time. Our study shows that adding capacity to an existing edge resource without increasing internetwork bandwidth may actually increase network-wide congestion and can result in reduced system capacity. Future work includes evaluating alternative application profiles with task splitting and compute prediction, analyzing the impact of mobility on the system capacity and edge placement using the city-scale edge cloud testbeds such as COSMOS [53].

Acknowledgement

We thank the anonymous reviewers and Prof. Ellen Zegura (shepherd for our submission) for valuable comments which have helped us to significantly improve the paper. This research is supported under NSF Future Internet Architecture - Next Phase (FIA-NP) Award CNS-134529.

References

[1] Azuma, Ronald T. "A survey of augmented reality." Presence: Teleoperators and Virtual Environments 6.4 (1997): 355-385.

[2] Rheingold, Howard. Virtual reality: exploring the brave new technologies. Simon and Schuster Adult Publishing Group, 1991.

[3] Chandra, Abhishek, Jon Weissman, and Benjamin Heintz. "Decentralized edge clouds." IEEE Internet Computing 17.5 (2013): 70-73.

[4] Greenberg, Albert, et al. "VL2: a scalable and flexible data center network." ACM SIGCOMM computer communication review. Vol. 39. No. 4. ACM, 2009.

[5] Caudill, Jason G. "The growth of m-learning and the growth of mobile computing: Parallel developments." The International Review of Research in Open and Distributed Learning 8.2 (2007).

[6] Jain, Puneet, Justin Manweiler, and Romit Roy Choudhury. "Overlay: Practical mobile augmented reality." Proceedings of the 13th Annual International Conference on Mobile Systems, Applications, and Services. ACM, 2015.

[7] Jacobs, Marco C., and Mark A. Livingston. "Managing latency in complex augmented reality systems." Proceedings of the 1997 symposium on Interactive 3D graphics. ACM, 1997.

[8] Zhang, Wuyang, et al. "Towards efficient edge cloud augmentation for virtual reality MMOGs." Proceedings of the Second ACM/IEEE Symposium on Edge Computing. ACM, 2017.

[9] Armbrust, Michael, et al. "A view of cloud computing." Communications of the ACM 53.4 (2010): 50-58.

[10] Chana, Inderveer, and Nidhi Jain Kansal. "Cloud load balancing techniques: A step towards green computing." International Journal of Computer Science Issues (IJCSI) 9.1 (2012): 238.

[11] Simoens, Pieter, et al. "Service-Centric Networking for Distributed Heterogeneous Clouds." IEEE Communications Magazine (2017).

[12] Wang, Shiqiang, et al. "Dynamic service migration in mobile edge-clouds." IFIP Networking Conference (IFIP Networking), 2015. IEEE, 2015.

[13] Atheer AiR. (n.d.). Retrieved April 03, 2018, from https://atheerair.com/.

[14] Microsoft. (n.d.). Detail of light relfecting on lens of HoloLens. Retrieved April 03, 2018, from https://www.microsoft.com/en-us/hololens.

[15] Glass Explorer Edition. (n.d.). Retrieved April 03, 2018, from https://developers.google.com/glass/.

[16] Roman, Rodrigo, Javier Lopez, and Masahiro Mambo. "Mobile edge computing, fog et al.: A survey and analysis of security threats and challenges." Future Generation Computer Systems 78 (2018): 680-698.

[17] Reiter, Andreas, Bernd Prnster, and Thomas Zefferer. "Hybrid mobile edge computing: Unleashing the full potential of edge computing in mobile device use cases." Proceedings of the 17th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing. IEEE Press, 2017.

[18] OpenCV library. (n.d.). Retrieved April 04, 2018, from https://opencv.org/.

[19] WiGLE.net. WiGLE: Wireless Network Mapping. WiGLE: Wireless Network Mapping, www.wigle.net/.

[20] CloudPing.info. CloudPing.info, www.cloudping.info/.

[21] Rodrigues, Tiago Gama, et al. "Hybrid method for minimizing service delay in edge cloud computing through VM migration and transmission power control." IEEE Transactions on Computers 66.5 (2017): 810-819.

[22] Liu, Juan, et al. "Delay-optimal computation task scheduling for mobile-edge computing systems." Information Theory (ISIT), 2016 IEEE International Symposium on. IEEE, 2016.

[23] Tootoonchian, Amin, and Yashar Ganjali. "Hyperflow: A distributed control plane for openflow." Proceedings of the 2010 internet network management conference on Research on enterprise networking. 2010.

[24] Koponen, Teemu, et al. "Onix: A distributed control platform for large-scale production networks." OSDI. Vol. 10. 2010.

[25] Chen, Xu, et al. "Efficient multi-user computation offloading for mobile-edge cloud computing." IEEE/ACM Transactions on Networking 24.5 (2016): 2795-2808.

[26] Muritiba, Albert E. Fernandes, et al. "Algorithms for the bin packing problem with conflicts." Informs Journal on computing 22.3 (2010): 401-415.

[27] Satyanarayanan, Mahadev, et al. "Edge analytics in the internet of things." IEEE Pervasive Computing 14.2 (2015): 24-31.

[28] Bonomi, Flavio, et al. "Fog computing and its role in the internet of things." Proceedings of the first edition of the MCC workshop on Mobile cloud computing. ACM, 2012.

[29] Aazam, Mohammad, et al. "Cloud of Things: Integrating Internet of Things and cloud computing and the issues involved." Applied Sciences and Technology (IBCAST), 2014 11th International Bhurban Conference on. IEEE, 2014.

[30] Biswas, Abdur Rahim, and Raffaele Giaffreda. "IoT and cloud convergence: Opportunities and challenges." Internet of Things (WF-IoT), 2014 IEEE World Forum on. IEEE, 2014.

[31] Celesti, Antonio, et al. "Characterizing cloud federation in IoT." Advanced Information Networking and Applications Workshops (WAINA), 2016 30th International Conference on. IEEE, 2016.

[32] Kuo, Alex Mu-Hsing. "Opportunities and challenges of cloud computing to improve health care services." Journal of medical Internet research 13.3 (2011).

[33] Zhu, Quanwen, et al. "3d lidar point cloud based intersection recognition for autonomous driving." Intelligent Vehicles Symposium (IV), 2012 IEEE. IEEE, 2012.

[34] Kumar, Swarun, Shyamnath Gollakota, and Dina Katabi. "A cloud-assisted design for autonomous driving." Proceedings of the first edition of the MCC workshop on Mobile cloud computing. ACM, 2012.

[35] Vaquero, Luis M., and Luis Rodero-Merino. "Finding your way in the fog: Towards a comprehensive definition of fog computing." ACM SIGCOMM Computer Communication Review 44.5 (2014): 27-32.

[36] Shi, Weisong, and Schahram Dustdar. "The promise of edge computing." Computer 49.5 (2016): 78-81.

[37] Pan, Jianli, and James McElhannon. "Future edge cloud and edge computing for internet of things applications." IEEE Internet of Things Journal 5.1 (2018): 439-449.

[38] Morabito, Roberto, et al. "Consolidate IoT Edge Computing with Lightweight Virtualization." IEEE Network 32.1 (2018): 102-111.

[39] Pinto, Sandro, et al. "IIoTEED: An Enhanced, Trusted Execution Environment for Industrial IoT Edge Devices." IEEE Internet Computing 21.1 (2017): 40-47.

[40] Sasaki, Kengo, et al. "Vehicle control system coordinated between cloud and mobile edge computing." Society of Instrument and Control Engineers of Japan (SICE), 2016 55th Annual Conference of the. IEEE, 2016.

[41] Zeydan, Engin, et al. "Big data caching for networking: Moving from cloud to edge." IEEE Communications Magazine 54.9 (2016): 36-42.

[42] Aujla, Gagangeet Singh, et al. "Optimal decision making for big data processing at edge-cloud environment: An sdn perspective." IEEE Transactions on Industrial Informatics 14.2 (2018): 778-789.

[43] Ravindran, Ravishankar, et al. "Towards software defined icn based edge-cloud services." Cloud Networking (CloudNet), 2013 IEEE 2nd International Conference on. IEEE, 2013.

[44] Li, Wei, et al. "System modelling and performance evaluation of a three-tier Cloud of Things." Future Generation Computer Systems 70 (2017): 104-125.

[45] Tong, Liang, Yong Li, and Wei Gao. "A hierarchical edge cloud architecture for mobile computing." Computer Communications, IEEE INFOCOM 2016-The 35th Annual IEEE International Conference on. IEEE, 2016.

[46] Hu, Yi, et al. "GeoTopo: A PoP-level Topology Generator for Evaluation of Future Internet Architectures." Network Protocols (ICNP), 2015 IEEE 23rd International Conference on. IEEE, 2015. ogi

[47] Garcia Lopez, Pedro, et al. "Edge-centric computing: Vision and challenges." ACM SIGCOMM Computer Communication Review 45.5 (2015): 37-42.

[48] Zhang, Wuyang, et al. "Segue: Quality of service aware edge cloud service migration." Cloud Computing Technology and Science (CloudCom), 2016 IEEE International Conference on. IEEE, 2016.

[49] You, Changsheng, et al. "Energy-efficient resource allocation for mobile-edge computation offloading." IEEE Transactions on Wireless Communications 16.3 (2017): 1397-1411.

[50] Ateya, Abdelhamied A., et al. "Multilevel cloud based Tactile Internet system." Advanced Communication Technology (ICACT), 2017 19th International Conference on. IEEE, 2017.

[51] Choy, Sharon, et al. "A hybrid edge-cloud architecture for reducing on-demand gaming latency." Multimedia systems 20.5 (2014): 503-519.

[52] Bjrkqvist, Mathias, et al. "Minimizing retrieval latency for content cloud." INFOCOM, 2011 Proceedings IEEE. IEEE, 2011.

[53] Cosmos-lab.org. (2018). COSMOS Project. [online] Available at: http://www.cosmos-lab.org [Accessed 30 Aug. 2018].