Efficiency of A Domain-Based Fog Computing Architecture

Edmund Chen, Kyle Dunne, Aditya Tyagi

†UC Berkeley EE 122 Communication Systems

Abstract

In this project, we investigate different methods to improve quality of service (QoS) for fog network applications. This emerging technology seeks to further minimize service delay by adding a layer of "fog" devices between the IOT and Cloud layers. We explore different protocols that may be introduced in the fog layer to further reduce IOT service delay, including FIFO, CR, and EDD.

Background

A fog layer is introduced to receive tasks which would otherwise be transmitted to the cloud for the purposes of expediting service delay for IOT applications. This new architecture comprises of a certain number of "fog nodes" being placed at close proximity to the IOT layer, thus the term "edge computing". Currently, a common protocol for the introduction of this layer is as follows. Naturally, we must differentiate certain requests by their respective computational times, which are sorted into "light" and "heavy" requests, whose average processing times are given by z_j and z'_j respectively. Then the average waiting time at fog node j is

$$W_j = c_j z_j + c'_j z'_j$$

where c_j and c'_j are the number of light and heavy tasks in queue at the fog node respectively. We want to model a way to minimize the waiting time of any given task, which can be given by

$$d_i = p_i^I(A_i) + p_i^F(X_{ij}^{IF} + Y_{ij}^{IF} + L_{ij}) + p_i^C(X_{ik}^{IC} + Y_{ik}^{IC} + H_k + X_{ki}^{CI} + Y_{ki}^{CI})$$

where $X_{st}^{LL'}$ denotes propagation delay, $Y_{st}^{LL'}$ denotes transmission delay, for node s in layer L to node t in layer L'. p_i^L denotes the probability the request is evaluated at layer L. i, j, k and I,F,C correspond to nodes and layers for IOT, fog, and cloud respectively. We can see that the transmission delays can be modeled by

$$Y_{ij}^{IF} = \sum_l \frac{\zeta_i}{R_l}$$

for the data amount of the request at question ζ_i and for the link rate R_l between the two nodes in question. A_i, L_{ij}, H_k denote the processing time at the IOT, Fog, and Cloud layers respectively. The processing in the cloud and IOT levels are

$$H_k = f_k g_k + f'_k g'_k \qquad A_i = b_i \cdot a_i + b'_i \cdot a'_i$$

simply the average waiting time, similar to W_j. Furthermore, we can see that the delay in the fog layer can be given by

$$L_{ij}(x) = P_j(\bar{W}_j + X_{ji}^{FI} + Y_{ji}^{FI}) + (1-P_j)\Big[(1-\phi(x))(X_{jj'}^{FF} + Y_{jj'}^{FF} + L_{ij'}(x+1)) + \phi(x)(X_{jk}^{FC} + Y_{jk}^{FC} + \bar{H}_k + X_{ki}^{CI} + Y_{ki}^{CI})\Big]$$

If we take P_j to be the probability that node j in the fog layer accepts such an IOT task, the first term expresses the total time from entering the queue to returning to the original IOT node. In the case that node j cannot take this task, as its W_j is above a certain threshold θ_j, the task is either forwarded to another fog node j' and then the function is recursively run, or it goes to the cloud layer if it has reached a maximum number of forwards. $\phi(x)$ serves to determine this factor, which remains at 0 until the task request at hand has reached the maximum number of forwards.

Problem Formulation

What is the best way to organize the fog layer? The established protocol leaves many options open for further optimization. The goal of our project is to investigate alternative architectures for the fog layer and to run simulations testing these alternative architectures against the established one. Our project's main stipulation is the introduction of a control node at the fog layer, as the incessant propagation between the fog nodes would seem to be a major factor to the increasing delay of each IOT task. To introduce a control node, we pinpoint node n_c as a specific control node in each fog domain. Doing this sacrifices one node to do actual processing, which we denote as "execution nodes". Thus, the revised d_i and L_{ij} is given by

$$L_{ij} = (1-\phi(x))\left[X_{cj}^{FF} + Y_{cj}^{FF} + W_j + X_{ji}^{FI} + Y_{ji}^{FI}\right] + \phi(x)\left[X_{ck}^{FC} + Y_{ck}^{FC} + H_k + X_{k_i}^{CI} + Y_{ki}^{CI}\right]$$

$$d_i = p_i^I(A_i) + p_i^F(X_{ic}^{IF} + Y_{ic}^{IF} + L_{ij}) + p_i^C(X_{ik}^{IC} + Y_{ik}^{IC} + H_k + X_{ki}^{CI} + Y_{ki}^{CI})$$

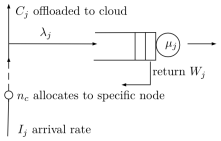


Fig. 1: Queuing Model for the fog domain

Thus we get the model shown for the fog network queuing model. In addition, we will create a normal distribution of processing times as to subvert the simplification of having only "heavy" and "light" requests.

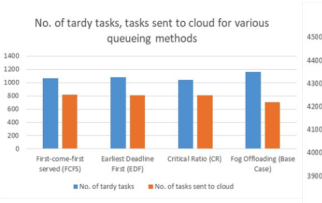
The second optimization we are testing is on the use of different queuing algorithms at the fog nodes. The current model uses FIFO (first in first out). In an attempt to make a more efficient architecture, we introduced a "deadline" to each task. This allows the IoT devices to prioritize more important tasks by giving them earlier deadlines. To accommodate this, we tested our architecture with two additional queuing mechanisms: EDD (earliest-due-deadline), where tasks are executed in order their deadline (from earliest to latest), and CR (critical ratio), where both the deadline and the computational complexity of tasks are considered when deciding which task to evaluate next. We test all these situations using discrete event simulation in python with various parameters.

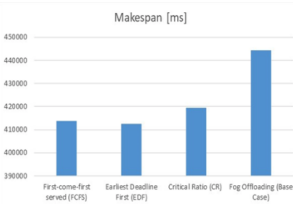
Our model was accomplished through using a discrete event simulation simulation in python, where we could map out each of the respective events that happen with an IOT request. These events included things like "fog request arrived at control node", and others to that effect. Our simulated network comprised of 1 cloud server, 1 fog domain, 10 fog nodes, and evaluated a total of 1500 IOT requests from the IOT layer. In the case of a control node, the 10 fog nodes would be split into 9 execution nodes, and 1 control node. These parameters coupled with the delays we chose were used to obtain our metrics which included "total time to complete tasks", "number of tasks offloaded to cloud", and "number of tardy tasks". It would stand to reason tha the fog layer is farther away from the cloud layer than the IOT layer, so our parameters reflected that similarly. However, parameters such as delay and processing time could be easily changed.

Results

Running the simulation described in the Problem Formulation yields the following results for the 4 types of queuing algorithms

Fog Queue	Makespan[ms]	# of Tardy Tasks	# Tasks to Cloud
FCFS	413698.169ms	1064/1500	814/1500
EDF	412481.24ms	1081/1500	813/1500
CR	419568.97ms	1042/1500	808/1500
Fog Offloading	444406.74ms	1165/1500	703/1500





As shown, the introduction of the index node is better than the original case, fog offloading. Within the queuing policies, it we see that EDF ultimately works best in the sake of time, but works out to have a higher # of tardy tasks. All the queuing policies with an index node hover at around the same offloading rate to the cloud, showing consistently better results than the fog offloading. Each queuing policy aligns nicely with the metric they seek to optimize.

Conclusions

As with the results in our discrete event simulation above, we can see that the introduction of an index node does reduce IOT service delay by a significant amount as compared to the algorithms introduced in [2][1], and the further queuing policies within such an architecture optimize the various minutia of the fog layer as a whole. It might be interesting for future work to simulate multiple domains, consider different task request sizes, or find an algorithm to change θ dynamically.

References

[1] A. Yousefpour G. Ishigaki R. Gour and JP. Jue. "On Reducing IOT Service Delay via Fog Offloading". In: *IEEE* 15 (Apr. 2018), pp. 303–370.

[2] T. Mori Y. Utsunomiya X. Tian and T. Okuda. "Queuing Theoretic Approach to Job Assignment Strategy Considering Various Inter-arrival of Job in Fog Computing". In: *IEEE* (Sept. 2017), pp. 151–156.