CTOM: Collaborative Task Offloading Mechanism for Mobile Cloudlet Networks

Xiaochen Fan*, Xiangjian He*, Deepak Puthal*, Shiping Chen†, Chaocan Xiang‡, Priyadarsi Nanda*, Xunpeng Rao§
* School of Electrical and Data Engineering, University of Technology Sydney, Sydney, Australia
† Information Engineering Laboratory, CSIRO ICT Centre, Sydney, Australia
‡ Logistic Engineering University, Chongqing, China
‡ PLA Army Engineering University, Nanjing, China
{fanxiaochen33, xiang.chaocan, xunpengrao}@gmail.com {shiping.chen}@csiro.au
{xiangjian.he, deepak.puthal, priyadarsi.nanda}@uts.edu.au

***Abstract*—With the advance in wireless networking technologies and communication infrastructures, mobile cloud computing has emerged as a pervasive paradigm to execute computing tasks for capacity-limited mobile devices. More specifically, at the network edge, resource-rich and trusted cloudlet system is acting as a 'data center in a box' to support compute-intensive mobile applications. The mobile cloudlets can provide in-proximity services by executing the workloads for nearby devices. Nevertheless, load balancing in mobile cloudlet network is of great importance and has a huge impact on response time. Existing methods for cloudlet load balancing basically rely on the user-cooperation or strategic placement. However, above solutions require global task load information from the network, which is costly in communication and computational overhead. To achieve more efficient and low-cost load balancing, we propose 'CTOM', a Collaborative Task Offloading Mechanism for mobile cloudlet network. Our solution is based on balls-and-bins theory and can balance the task load only with limited information. Extensive simulations and mobility trace based evaluation show that, the proposed 'CTOM' is effective and can achieve similar performance to the greedy algorithm with low computing complexity.**

***Keywords*-load balancing; mobile cloudlet network; task allocation**

I. Introduction

In recent years, with the pervasive proliferation of mobile devices and the advances in networking technologies, mobile users are free to enjoy more powerful and functional applications, for example, Augmented Reality, Virtual Reality and Face Recognition [1]. While mobile applications become more demanding in computation and resources, the capacity of smart devices are still constrained. Such that, mobile users are constantly facing with the problems of resource-exhaustion and energy-drain. To tackle this issue, cloud computing has been proposed and pervasively used for processing resource-intensive tasks [2]. However, due to the long distance between the central servers and mobile users, there are some inevitable limitations in cloud computing, such as network latency, signal loss, link noise and transmission delays [3]. To overcome the above issues and provide more accessible resource to mobile users, an alternative cloud computing paradigm is proposed, which is the so called cloudlet [4].

A cloudlet is a trusted, resource-rich cluster of servers that integrated with wireless access points(APs), by which it is accessible and connected to nearby mobile users [5]. By providing seamless computing services with low-latency and high-bandwidth wireless access, cloudlets can execute computation tasks offloaded from mobile users at real-time speeds, thereby significantly improve the performance of cloud computing [4], [6], [7]. Recent studies [8], [9], [10], [11] focused on mobile cloudlets, which utilize the multitude of near-user vehicular cloudlet to carry out more efficient communication and computation.

A key challenge in mobile cloudlet network is how to keep load balancing among all the mobile cloudlets, so that cloudlet resources are fully utilized and tasks could be concurrently processed by multiple servers, thus shortening the average task response time. As vehicle-based cloudlets may randomly travel around various areas with different population density, it is impossible to centrally control the amount of user task offloading to an exact cloudlet. Adding to that the connectivity in mobile cloudlet network is intermittent, how to achieve the load balancing still remains a challenge.

There are some studies address load balancing issues in static cloudlet systems, either by strategic cloudlet placement [12], [5] or cloudlet-oriented task redistribution [6], [13]. However, these methods are not applicable in mobile scenario, where the cloudlets are enhanced with random mobility and the network is intermittently connected. Indeed, it is quite daunting to achieve load balancing among mobile cloudlets as they are purely distributed. Even worse, for each cloudlet, the load information of its neighbors is constantly changing, which implies that the computation of obtaining overall load information would be more costly. Accordingly, two challenges need to be carefully addressed.

First, the load balancing should be achieved through the collaborative task offloading. As the mobility of cloudlets can neither be centrally controlled nor predicted, it is hard to redirect exact amount of task flow from one cloudlet to another. Fortunately, it is impossible for encountering cloudlets to offload tasks collaboratively by sharing their load information.

Second, the balanced task allocation method should be low-cost and light-weight in communication and computation. It is

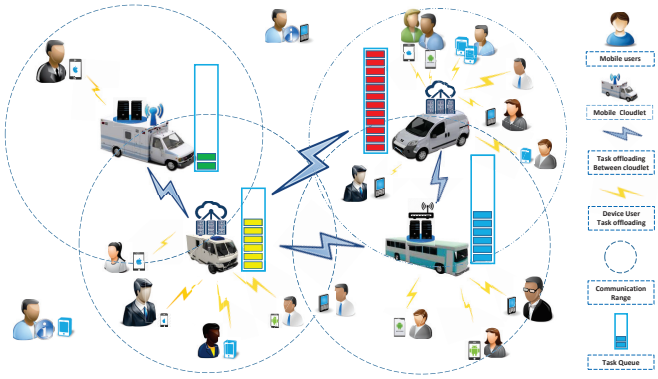


Fig. 1: Task offloading in mobile cloudlet network scenario

impractical to query global load information in a distributed mobile cloudlet network. Even if it can be achieved, the accumulative communication cost from the overall network would be extremely high. Moreover, the out-sync task load information caused by transmission delay may lead to wrong offloading decision to already overloaded cloudlets.

In this paper, to deal with aforementioned challenges, we propose 'CTOM', a Collaborative Task Offloading Mechanism for mobile cloudlet networks. Our method leverages balls-and-bins model to perfectly fit the distributed task allocation scenario in mobile cloudlet networks [14]. Based on the 'two-choice' paradigm, by only querying load information from two random neighbors in each interval, a cloudlet could process a relatively balanced task offloading. Accumulatively, the longest task queue among all mobile cloudlets would be significantly reduced with high probability [15].

We summarize the contributions of this paper as follows.

1) To the best of knowledge, this is the first work on collaborative task offloading mechanism for mobile cloudlet network, where the cloudlets are enhanced with mobility and intermittently connected.
2) Inspired by probability theory derived from balls-and-bins model, we propose an innovative algorithm for balanced task allocation in distributed mobile cloudlet network. By comparing the task load of only two neighbors, a mobile cloudlet can make a valid task offloading decision with low communication cost.
3) We validate the effectiveness of our method in simulation and evaluation with real-world trace dataset. The simulation results show that 'CTOM' achieves exceedingly balanced results in mobile cloudlet task allocation and performs closely to the optimal solution that using global task load information.

The remainder of this paper is as follows. We introduce the system model and the problem formulation in Section II. The algorithm design is introduced in Section III. To validate the proposed 'CTOM' scheme, extensive simulation and evaluations have been done and the results are illustrated in Section IV. Finally, we conclude our paper in Section V.

II. System Model and Problem Formulation

In this section, we present the preliminaries of our system model. After that, we describe the load balancing problem in mobile cloudlet networks.

Network Model We start the network model with a set of mobile cloudlets deployed in a Metropolitan Area Network-s(MAN). We assume there are K mobile cloudlets are integrated with vehicular access points(APs), and we denote these cloudlets by a set of $\mathrm{C} = \{\mathrm{c}_1,\mathrm{c}_2,...,\mathrm{c}_K\}$. It is also assumed that the user's applications are dynamically partitioned into offloadable and executable computing tasks that can be processed at any of the k cloudlets. Also, cloudlets could communicate with each other via WiFi networks. Such that, each cloudlet can either locally process incoming tasks, or transmit current tasks to neighboring cloudlets in the network(as depicted in Fig. 1.

Cloudlet Model According to [6], for each mobile cloudlet $i \in \{1, 2, ..., K\}$, we model it as a $M/M/n$ queue. Each cloudlet i has s_i server(s) with the service rate μ_i. Also, we adopt random walk to model the mobility of cloudlets, as they randomly travel around in the metropolitan area. For any cloudlet i, as the amount of task offloading from nearby user varies constantly, we also adopt the Poisson process from [6] to represent the incoming user tasks, where the mobile users choose the nearest mobile cloudlet to offload tasks and the task arrival rate at cloudlet i is λ_i. Also, to store the arrived tasks pending for execution, each mobile cloudlet holds a FIFO task queue $Q = \{q_1, q_2, .., q_k\}$, where the queueing length is $||Q_i||$.

Communication Model Similar to [6], the mobile cloudlets in our model are also integrated with Wi-Fi Access Points, which provides for one-hop, low-latency and high-bandwidth wireless access opportunities for task offloading . Due to the random mobility, the mobile cloudlet network is intermittently connected. Only when the wireless connection is established the mobile cloudlets can offload tasks to each other. We assume that the geographical distributions that mobile cloudlet follows an independent Homogeneous Poisson Point Process [7]. To better investigate the task offloading in mobile cloudlet networks, we divide the time domain into serial intervals as time slots. During each time slot, a cloudlet i has probability of accessing j cloudlet, which can be calculated as:

$$\Pr\{K = j\} = e^{-\pi R^2 \lambda_c} \frac{(\pi R^2 \lambda_c)^j}{j!}, j = 0, 1, 2, ..., k \qquad (1)$$

where λ_c is the distribution density of mobile cloudlets and R is the inter-contact range between cloudlets. When the distance d_ij between cloudlet i and j is within the inter-contact range R, a communication can be established between them. The inter-meeting time of cloudlets c_i and c_j is denoted as $t_{i,j}$. Based on [16] and [17], the inter-contact time $t_{i,j}$ would follow exponential distribution with pairwise rate α_{ij}, i.e., $f(t) = \frac{1}{\alpha_{i,j}} e^{-\frac{1}{\alpha_{i,j}} \cdot t}, t \geq 0, t \geq 0$. Such that, between any two time stamp t_a and t_b, the encountering probability of cloudlets c_i and c_j is denoted as followed:

$$P_{i,j}(t_a, t_b) = e^{-\frac{1}{\alpha_{i,j}} \cdot t_a} - e^{-\frac{1}{\alpha_{i,j}} \cdot t_b} \qquad (2)$$

According to computing task offloading experiment, for applications such as augmented reality and face recognition, under WiFi connection, the task execution time is approximately $10^{-4} \sim 10^{-2}$ seconds [1]. Adding to that the round-trip time (RTT) of wireless transmission only takes tens to hundreds of milliseconds, we can set time interval reasonably long enough for the inter-contact time (including execution time and RTT), *i.e.*, the task execution results can be sent back to the corresponding mobile users within the same time interval [8].

Task Offloading Model In our model, a 'task' refers to an application phase that involves executable codes and offloadable data to be processed at any mobile cloudlet [7]. Such that, the total number of tasks generated from different user's application fluctuates. We address above considerations by sampling Poisson process to determine the actual number of tasks at cloudlet i. We denote $T_i = \lambda_i$ as the arriving task from users to cloudlet i. We adopt the percent imbalance metric and the statistical moment from [18] to evaluate the overall load balancing of task allocation, *i.e.*, percent imbalance metric η and skewness φ. The above metrics are calculated as follows

$$\eta = \left(\frac{L_{max}}{\bar{L}} - 1\right) \times 100\%, \quad \varphi = \frac{\frac{1}{n}\sum_{i=0}^{n}(L_i - \bar{L})^3}{\left(\frac{1}{n}\sum_{i=0}^{n}(L_i - \bar{L})^2\right)^{3/2}} \tag{3}$$

where L_{max} and $\bar{L}$ are the maximum and average load respectively. The percent imbalance metric measures the severity of load imbalance, while the skewness provides a detailed picture of load distribution [18].

Problem Domain Given a mobile cloudlet network G with a set of cloudlet $\mathrm{C} = \{\mathrm{c}_1,\mathrm{c}_2,...,\mathrm{c}_K\}$, where each cloudlet i holds a FIFO task queue in $Q = \{q_1, ...q_i, ...q_k\}$ to store received tasks. Meanwhile, cloudlet i has n_i servers with service rate μ_i and the task arrival rate at cloudlet i is λ_i. We define the Mobile Cloudlet Load Balancing Problem as follows.

Basic Load Balancing Problem: We investigate how to collaboratively offload tasks in mobile cloudlet networks. Particularly, our goal is to minimize the overall variance of task queue in achieving balanced task distribution, which can be given by

$$Minimize \min \sum_{i \in C} \|Q_i - E[Q]\| \tag{4}$$

Subject to $\mu_i \cdot n_i \geq \lambda_i, i \in C$.

Gap minimization and balance metric evaluation: Minimizing the task load gap between maximum queue and average queue is also worth evaluating. Note that the maximum load $L_{\max}$ and average load $\bar{L}$ both count for the imbalance metric and statistical skewness in 3. The evaluation of task load gap can be given by

$$Minimize \max_{i \in C} \|Q_i\| - E_{i \in C}[Q_i] \tag{5}$$

Subject to $\mu_i \cdot n_i \geq \lambda_i, i \in C$.

Load \ Maximum m Balls, n Bins	Random Allocation	D-choice Allocation
Case: $m = n$	$\frac{\log n}{\log\log n} \cdot (1 + o(1))$	$\frac{\log\log n}{\log d} + \Theta(1)$
Case: $m > n\log n$	$\frac{m}{n} + \Theta(\sqrt{\frac{m\log n}{n}})$	$\frac{\log\log n}{\log d} + \frac{m}{n} + \Theta(1)$
Case: $m < n$	$\Theta(\frac{\log n}{\log(n/m)})$	$\frac{\log\log n}{\log d} + \Theta(1)$

Fig. 2: Theoretical results of maximum load in balls-and-bins problem

III. Our Solution and Algorithm Design

A. Leveraging balls-and-bins based probability theory

The balls-and-bins model is a classic in probability model for randomized allocation process [15]. Suppose that m balls are to be thrown into n bins , with each ball choosing a bin independently and uniformly at random. The question comes with the $maximumload$, i.e., the largest number of balls at any bin. In [14], a partially randomized allocation paradigm called 'd-choice' is proposed. For each allocation, the 'd-choice' places a ball into the least loaded bin of d ($d \geq 2$) bins. As the maximum load varies with different quantities of balls and bins, we conclude the maximum load for random allocation and 'd-choice' allocation in Fig. 2.

B. Algorithm Design

1) Overview: To solve unbalanced task allocation problem, we propose 'CTOM', a collaborative task offloading mechanism for mobile cloudlet networks. Here, the tasks and cloudlets are considered as balls and bins respectively. As the network connectivity is intermittent, the neighbor's load information of each cloudlet also continuously changes. Such that, traditional methods that based on global load information will be costly in communication and computation for our problem.

In designing the algorithm, we adopt 'd-choice' paradigm from balls-and-bins model as '2-choice' method, By applying '2-choice' method, a mobile cloudlet compares the task load of 2 random selected neighbors in its communication range, and offload a computing task to the one with shorter task queue.

There are some basic assumptions in the model for algorithm design. First, we assume that the incoming tasks from mobile users are of the same size, thus the final allocation results can be measured precisely. Second, at each cloudlet i, the arrived tasks are stored at the task queue Q_i. Third, the time interval is long enough for the inter-contact time (including execution time and RTT).

2) Algorithm: As illustrated in Algorithm 1, we present the two-choice based mobile cloudlet collaborative task offloading algorithm, which balances the task distribution and computes the imbalance metrics as well as statistical moments.

First, the algorithm starts with initializing mobile cloudlet's location At the beginning of each time interval, the algorithm will update cloudlet's location and task load. For cloudlet i,

Algorithm 1 Algorithm CTOM

Input:
Mobile Cloudlet **C**, Time Interval **t**, Contact Range R
User Task Flow λ_i, Number of Servers **S**, Service Rate μ_i
Output:
Task Queue **Q**, Imbalance Metric and Statistical Moments

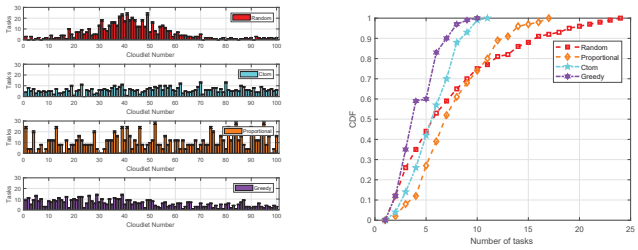
Minimize $\min \sum_{i \in C} \|Q_i - E[Q]\|$ using 'two-choice' method.
Initialize cloudlet's location (**X**,**Y**)
for Interval $i = [1:t]$ **do**
  Update cloudlet's location based on random walk
  Update cloudlet's load information with λ_i, s_i, μ_i
  for Each cloudlet $j = [1:k]$ **do**
    Updating neighboring list $L(j)$ according to Equ.1
      Initialize the offloading task weight W(i)
      Randomly select d neighbors
      $s \leftarrow$ the first selected one of d neighbors
        for $v = 2$ to d **do**
          if $q_s > q_v$ **then** $s \leftarrow v$
          end if
        end for
      if $q_j > q_s$ **then**
        $P \leftarrow 1 - q_s/q_j$
        $q_j \leftarrow l_s + W(i) * P$
        $l_{u(i)} \leftarrow l_{u(i)} - W(i) * P$
      end if
  end for
end for
return **Q**, η, φ

based on Equation 1, the algorithm computes the number of its neighbors in current time interval. And then, by randomly selecting d neighbors, the algorithm begins to iteratively compare their task load in sorting for the least task load. Note that, here we use d for illustrating different algorithm. The proposed 'CTOM' adopt 'two-choice' paradigm for task offloading, which determines 'd' as 2. For greedy algorithm, 'd' equals to the total number of current cloudlet's neighbors. After that, the algorithm will check whether the selected neighbor is appropriate for taking over the current cloudlet's task by further comparing their task load. The proportional algorithm [19] continues to compute the offloading probability based on the proportion of task load between current cloudlet and the selected cloudlet.

At last, the imbalance metric together with statistical moments will be calculated for evaluation use.

IV. Performance Evaluation

The performance evaluation of proposed scheme is twofold. First, we evaluate 'CTOM' in simulated network scenario, where cloudlet encounters are generated from random walk simulations. Second, we apply the proposed algorithm to a real-world trace for further evaluations.



(a) Task allocation result (b) Task load distribution

Fig. 3: Task allocation results under different schemes

A. Simulation Study

1) Basic setups: We run the simulation in a $10km^2$ region, which is of the similar scale of a city's central area. Here, we set the number of mobile cloudlets as 100 and the communication $range$ as 20 metre. The total number of time slot is 600. According to [6], for each cloudlet i, we set the service rate μ_i by sampling normal distribution $\mathcal{N}(2,1) > 0$, and we set the number of its servers by sampling the Poisson distribution with a mean of 2. For arriving tasks at cloudlet i, we set task arrival rate λ_i by sampling the Normal distribution $\mathcal{N}(4,2) > 0$

Under the 'CTOM' scheme, during each time interval, a cloudlet first randomly choose 2 neighbors in its contact range. After querying and comparing their load states, the cloudlet choose offload a task to the one with less task load, where the computing complexity in each time interval is $O(1)$. Similar to [17], we compare performance of proposed scheme with three benchmarks, i.e., random allocation, proportional allocation [14] and greedy allocation. In random allocation, a mobile cloudlet offloads tasks by randomly selecting another mobile cloudlet in its contact range. Conversely, the greedy allocation method queries all load information from its neighbors. then by comparing their task loads to allocate tasks to the optimal cloudlet (with computing complexity of $O(n)$). As for proportional allocation, the chance for task offloading to a randomly selected cloudlet is based on a probability parameter, which is calculated with task load information. The simulation programs are all written in MATLAB codes. We run the programs in a Dell laptop with Intel Core i5 processor and 8 GB RAM. In general, each simulation program is executed for 100 times, and we take the average results as the final performance.

2) Simulation Results: Fig.3a plots the overall task allocation results of mobile cloudlets. As the cloudlet's servers keep processing tasks, the overall allocation shows the remaining tasks at each mobile cloudlet. In random allocation, the task distribution is centralized among a adjacent group of cloudlets(ID 18 to 60), where most of the task loads are more than 10 and up to 24. Meanwhile, the mobile cloudlets at edge area are loaded with much fewer tasks (average less than 5) or even in idle state. This may be due to the density of mobile users at central area is usually high, so that the cloudlets are

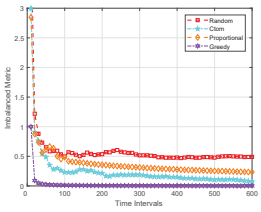


Fig. 4: The values of imbalance metric under different schemes

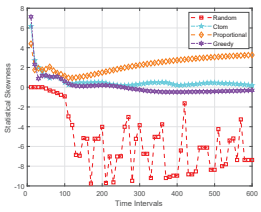


Fig. 5: The values of statistical skewness under different schemes

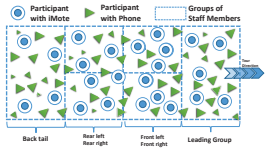


Fig. 6: Diagram of device deployment in roller tour

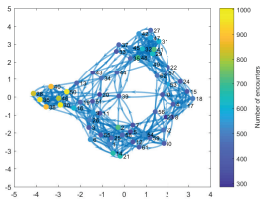


Fig. 7: The node-relation graph in iMote trace dataset

offloaded with more tasks and have more chances to share to task load. Similarly, the task allocation by 'Proportional' method is also extremely unbalanced, where the distribution of overloaded mobile cloudlets(with 25 remaining tasks) is more sparse. In contrast, under 'CTOM' and greedy allocation, mobile cloudlets are allocated with equivalent tasks(mostly around or below 10).

Fig.3b demonstrates the task allocation performance in cumulative distribution. Under random allocation and proportional allocation, about 30% mobile cloudlets are allocated more than 10 tasks, which will drag the overall task response time. Meanwhile, The 'CTOM' performs closely to greedy method in achieved balanced task offloading, where nearly 90% cloudlets are with task load under 10 and 55% cloudlets are offloaded with 5 to 10 tasks. Based on the imbalance metric [18], we further evaluate the task offloading performance. The percent imbalance metric and statistical skewness are calculated as in 3. The lower imbalance metric means the better balance performance in task allocation, *i.e.*, lower ratio of maximum and average task loads. From Fig.4, it is obviously that the greedy algorithm achieves the best performance in imbalance metric, which converges closely to 0. The imbalance metric of proposed 'CTOM' and proportional algorithm converges to 0.1 and 0.25 respectively. The random allocation performs with worst imbalance metric(0.5). Meanwhile, the positive or negative skewness means that, the quantities of mobile cloudlets that have higher or lower task load then average. In Fig.5, we can observe that the greedy allocation and 'CTOM' both achieved best skewness values at about 0, which means that there are few or no cloudlet with unbalanced load. While the proportional method has a skewness of 2, the random allocation's skewness value fluctuates violently in negative values, which means there exist many mobile cloudlets with much lower task load then average.

The above simulation results demonstrate that, the proposed 'CTOM' can achieve balanced and sustainable overall task allocation. As task distribution is more balanced and tasks are processed concurrently, 'CTOM' improves the utilization efficiency of mobile cloudlets and thus shortens the overall task response time.

B. Trace-driven study

We further explore the balanced task allocation in a trace-driven study. The mobility dataset we used is called 'RollerNet' [16], which was collected during a 2500 people participated roller tour in Paris, France.

1) Basic setups: 'RollerNet' includes the traces of opportunist sightings by wireless sensor network nodes called iMotes. The iMotes were distributed to a group of people to collect any opportunistic sighting of other mobile devices (including the other iMotes distributed) by Bluetooth. We drew a sample diagram of iMote deployment as depicted in Fig.6, where totally 62 skaters are equipped with iMotes and were divided into 6 groups at different region of the roller crowd. In this evaluation, we consider iMotes as mobile cloudlets that can remotely execute computing tasks for mobile users. For cloudlet i with service rate μ_i, we assign the service rate by sampling the normal distribution $N(6,2) > 0$. The number of servers at cloudlet i is sampled from Poisson distribution with a mean of 3. The task arrival rate λ_i follows a Normal distribution $0 < N(18,6) < s_i \cdot \mu_i$, where s_i is number of servers at mobile cloudlet i. All the settings are derived according to [6], differently, our evaluation is based on real-world trace dataset for mobility-enhanced cloudlets.

We conduct a twofold pre-processing on 'RollerNet' dataset. First, we unify the timing of user encounter records. By setting a common starting time based on the earliest record, we convert duration of all encounters into serial time slots in minutes. Based on the unified encounter records, we find that the total inter-contact time is $1567 - 1417 = 150$. Such that, we set the total interval for task offloading as 166. Second, we plot a encounter graph to depict the frequency of communications(FoC) among all the iMote skaters in Fig.7. From the FoC Fig.7, we find that the iMote carriers can be roughly divided into three groups based on communication frequency, which are: active group (with 800-1000 contacts), common group (with 500-800 contacts) and passive group (with 300-500 contacts). The above division consist with the formation of iMote skaters: skater association, staff and a set of friends.

2) Evaluation performance: Fig.8 shows the task allocation results based on 'RollerNet' in bar graph. The performance of 'CTOM' is similarly good to the greedy allocation, where most of the mobile cloudlets are offloaded with around 50 tasks. Meanwhile, in random and proportional allocation, the allocation results are unbalanced with task loads fluctuating severely among different cloudlets(up to 80 and down to 10).

Fig.9 illustrates the cumulative distribution of task alloca-

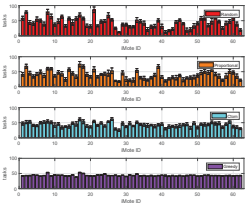


Fig. 8: Task load results in trace-driven evaluation

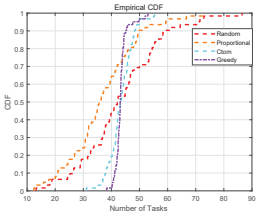


Fig. 9: Load distribution in trace-driven evaluation

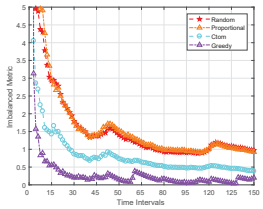


Fig. 10: Imbalance metric values under different schemes

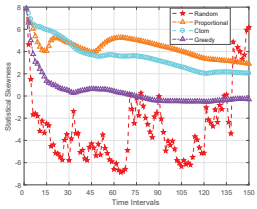


Fig. 11: Statistical skewness values under different schemes

tion. In random allocation, more than 30% mobile cloudlets have more than 50 tasks and about 30% others are with less than 30 tasks, this unbalance would result in longer average task response time. Meanwhile, under 'CTOM' around 95% of cloudlets are allocated with 30-50 tasks, which is equivalent. As the CDF line of greedy algorithm is the most centralised, it means that the task loads at different cloudlet only vary within a small range(around 40 to 50).

We also evaluate the percent imbalance metric and statistical skewness. In Figure.10, still the greedy algorithm achieved the best performance with 0.2 imbalance value, followed by 'CTOM' with converged results of 0.5. Interestingly, random allocation and proportional allocation here have similarly worse imbalance metric at 1, showing that both of them are not applicable enough for trace-driven mobile cloudlet scenario. In Figure.11, random allocation's skewness value fluctuates violently between positive and negative values, which implies that the task loads are continuously unbalanced throughout the process of allocation. While greedy allocation achieves the best performance in skewness with 0, there are overloaded mobile cloudlets under 'CTOM' and proportional method, which are revealed by their statistical skewness values of 2 and 3 respectively. The effectiveness of 'CTOM' is validated with the above simulation and evaluation results. which shows that our method can effectively tame the complex mobility issue and balance the load in mobile cloudlet networks.

V. Conclusion and Future Work

In this paper, we investigated the load balancing problem in mobile cloudlet networks (MCNs). By leveraging balls-and-bins theory, we devise 'CTOM', a collaborative task offloading mechanism. By locally querying limited task load information, the proposed scheme can reduce the longest task queue in allocation process effectively. The simulation and trace-driven evaluation results demonstrate that 'CTOM' performs exceedingly close to the optimal solution in load balancing, with computing complexity reduced from $O(n)$ to $O(1)$ in each allocation interval.

References

[1] M. Satyanarayanan, "The emergence of edge computing," *Computer*, vol. 50, no. 1, pp. 30–39, 2017.

[2] B.-G. Chun, S. Ihm, P. Maniatis, M. Naik, and A. Patti, "Clonecloud: elastic execution between mobile device and cloud," in *Proceedings of the sixth conference on Computer systems*. ACM, 2011, pp. 301–314.

[3] X. Chen, L. Jiao, W. Li, and X. Fu, "Efficient multi-user computation offloading for mobile-edge cloud computing," *IEEE/ACM Transactions on Networking*, vol. 24, no. 5, pp. 2795–2808, 2016.

[4] M. Satyanarayanan, P. Bahl, R. Caceres, and N. Davies, "The case for vm-based cloudlets in mobile computing," *IEEE pervasive Computing*, vol. 8, no. 4, 2009.

[5] Z. Xu, W. Liang, W. Xu, M. Jia, and S. Guo, "Efficient algorithms for capacitated cloudlet placements," *IEEE Transactions on Parallel and Distributed Systems*, pp. 2866–2880, 2016.

[6] M. Jia, W. Liang, Z. Xu, and M. Huang, "Cloudlet load balancing in wireless metropolitan area networks," in *Computer Communications, IEEE INFOCOM 2016-The 35th Annual IEEE International Conference on*, vol. 27, no. 10. IEEE, 2016, pp. 1–9.

[7] Y. Zhang, D. Niyato, and P. Wang, "Offloading in mobile cloudlet systems with intermittent connectivity," *IEEE Transactions on Mobile Computing*, vol. 14, no. 12, pp. 2516–2529, 2015.

[8] Y. Liu, M. J. Lee, and Y. Zheng, "Adaptive multi-resource allocation for cloudlet-based mobile cloud computing system," *IEEE Transactions on Mobile Computing*, vol. 15, no. 10, pp. 2398–2410, 2016.

[9] X. Hou, Y. Li, M. Chen, D. Wu, D. Jin, and S. Chen, "Vehicular fog computing: A viewpoint of vehicles as the infrastructures," *IEEE Transactions on Vehicular Technology*, vol. 65, no. 6, pp. 3860–3873, 2016.

[10] H. Cao and J. Cai, "Distributed multi-user computation offloading for cloudlet based mobile cloud computing: A game-theoretic machine learning approach," *IEEE Transactions on Vehicular Technology*, 2017.

[11] S. Jeong, O. Simeone, and J. Kang, "Mobile edge computing via a uav-mounted cloudlet: Optimization of bit allocation and path planning," *IEEE Transactions on Vehicular Technology*, 2017.

[12] M. Jia, J. Cao, and W. Liang, "Optimal cloudlet placement and user to cloudlet allocation in wireless metropolitan area networks," *IEEE Transactions on Cloud Computing*, 2015.

[13] J. Zhao, K. Yang, X. Wei, Y. Ding, L. Hu, and G. Xu, "A heuristic clustering-based task deployment approach for load balancing using bayes theorem in cloud environment," *IEEE Transactions on Parallel and Distributed Systems*, vol. 27, no. 2, pp. 305–316, 2016.

[14] B. Vöcking, "How asymmetry helps load balancing," *Journal of the ACM (JACM)*, vol. 50, no. 4, pp. 568–589, 2003.

[15] M. Mitzenmacher, "The power of two choices in randomized load balancing," *IEEE Transactions on Parallel and Distributed Systems*, vol. 12, no. 10, pp. 1094–1104, 2001.

[16] P.-U. Tournoux, J. Leguay, F. Benbadis, V. Conan, M. D. De Amorim, and J. Whitbeck, "The accordion phenomenon: Analysis, characterization, and impact on dtn routing," in *INFOCOM 2009, IEEE*. IEEE, 2009, pp. 1116–1124.

[17] Q. Li, P. Yang, X. Fan, S. Tang, C. Xiang, D. Guo, and F. Li, "Taming the big to small: efficient selfish task allocation in mobile crowdsourcing systems," *Concurrency and Computation: Practice and Experience*, 2017.

[18] O. Pearce, T. Gamblin, B. R. De Supinski, M. Schulz, and N. M. Amato, "Quantifying the effectiveness of load balance algorithms," in *Proceedings of the 26th ACM international conference on Supercomputing*. ACM, 2012, pp. 185–194.

[19] P. Berenbrink, T. Friedetzky, L. A. Goldberg, P. W. Goldberg, Z. Hu, and R. Martin, "Distributed selfish load balancing," *SIAM Journal on Computing*, vol. 37, no. 4, pp. 1163–1181, 2007.