Research Article

eCOTS: Efficient and Cooperative Task Sharing for Large-Scale Smart City Sensing Application

Qingyu Li and Panlong Yang

College of Communication Engineering, PLA University of Science and Technology, Nanjing 210007, China

Correspondence should be addressed to Panlong Yang: panlongyang@gmail.com

Received 5 July 2013; Revised 23 October 2013; Accepted 3 November 2013; Published 12 January 2014

Academic Editor: Yunhuai Liu

Copyright © 2014 Q. Li and P. Yang. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

With the pervasive use of mobile devices and increasingly computational ability, more concrete and deeper collaborations among mobile users are becoming possible and needed. However, most of the studies fail to consider load balancing requirement among mobile users. When tasks are unevenly distributed, the processing time as well as energy consumption will be extremely high on some devices, which will inevitably counterweight the benefits from incentive mechanism and task scheduling scheme. In this work, we propose eCOTS (Efficient and Cooperative Task Sharing for Large-scale Smart City Sensing Application). We leverage the “balls and bins” theory for task assignment, where $d$ mobile users in contact range are investigated, and select the least loaded one among the $d$ users. It has been proved that such simple case can effectively reduce the largest queueing length from $\Theta(\log n / \log \log n)$ to $\Theta(\log \log n / \log d)$. Simulation and real-trace driven studies have shown that, eCOTS can effectively improve the balancing effects in typical network scenarios, even the energy level and computational capability are diverse. In simulation study, eCOTS can reduce the gap between the maximum and minimum queueing lengths up to $5\times$ and over $2\times$ in real trace data evaluations.

1. Introduction

Recent years have witnessed a significant rise in the usage of smart phone as well as the associated applications. With the increasing number of mobile devices in network, people would share their data with mobile devices and make further collaborations with each other. Recently, crowdsourcing with participatory sensing schemes enables more unconscious and volunteering collaborations among sensing information shareholders. Actually, in such mobile computing environment, deeper collaborations are needed. Mobile users with different number of tasks will share and reassign tasks among them, taking advantages of computing and energy diversity. For example, users with low battery level will offload their tasks onto the contacted users in communication ranges with higher battery level. Even further, tasks should be assigned to nodes with higher computing ability.

However, previous works [1–5] fail to achieve load balancing among users. Under these schemes, the queueing length may be extremely high in some particular nodes, which will inevitably lead to exhausted energy and long delay. The root reason is that these works are all focusing on the data sharing efficiency instead of the allocation equilibrium among diverse users. Moreover, computational capability and remaining energy levels are also vitally important for task reallocation.

Traditional load balancing schemes cannot be applied to mobile social network directly, because, in mobile social network, the information collection and task assignment should be distributed. Centralized schemes will suffer from overwhelmingly communication overhead and time delay. Even though, the distributed features are applied such as some of the distributed algorithms, the transitory intercontact time, and dynamic queueing length [6, 7] making this task assignment fail to balance the tasks among users properly.

Actually, a simple and effective method is called for balancing the tasks among mobile users, which is critically important for smart phone based large-scale urban sensing applications. Unfortunately, balancing the tasks allocation among mobile users is not trivial. First, task balancing among mobile users needs accurate buffering information for each user, which will cost significant amount of bandwidth. Second, multiple task-assignment users and unpredictable task completion time will incur dynamic buffer-length,
which makes the balanced queue length difficult to achieve. Third, it is difficult to measure the intercontact time in mobile social network, which makes it difficult to evaluate the task completion time. Also, the remaining energy level and computational diversity among users should be considered for fair and balanced task allocation.

In this work, we propose eCOTS (Efficient and Cooperative Task Sharing for Large-scale Smart City Sensing Application). We address the above three challenges with two important techniques. First, we use the “two choices” scheme in “balls and bins” theory [8] as fundamental mechanism for task offloading, where queuing lengths of d users are compared, and users with the least value are selected. Such scheme is simple but effective. The root reason is that random walk also plays the role of random selection. Second, we leverage the random walk behavior for random choice and consider the energy level as well as computational capability for task reassignment. We find that the considerations for energy consumption and computational capability effectively offset the uncertainty in mobility without adding much complexity. The root reason is that energy consumption is also an implicit factor for intercontact time. Further, the computational capability also dominates the task allocation preference as well as inter contact frequency. We verified the effectiveness of the proposed techniques in our extensive experimental study.

The contribution of this work is threefold.

(i) We propose eCOTS, an efficient cooperative task shared among smartphone users for large-scale urban sensing applications. To the best of our knowledge, we are the first systematic study on balanced task assignment in pure distributed environment.

(ii) We incorporate the energy level and computational capability in task-assignment scheme, which could be in appliance with heterogeneous mobile social network. eCOTS can deal with these two cases separately and jointly with satisfiable performance. Also, these concerns are helpful in achieving the robustness in mobile environment.

(iii) We make an extensive experimental study on our proposed scheme. The evaluations include the task balancing effects in different traffic load under different energy levels and computational capabilities. Experimental results based on real trace data show the surprisingly good balancing property even though the variance and diversity between users are very large.

The rest of the paper is organized as follows. We describe our system model in Section 2. Section 3 provides problem formation as well as preliminary analysis about it. After describing our algorithm design in Section 4. We evaluate the performance of eCOTS by simulation in Section 5 and make real trace driven study in Section 6. Finally, we discuss related work in Section 7 and conclude the work in Section 8.

2. Preliminary and System Model

2.1 “Balls and Bins” Theory. The “balls and bins” problem is classic and simple, being widely used in many applications in computer science such as hashing [9, 10], shared memory emulations in DMM (Distributed Memory Machines) [11, 12], and load balancing with limited information. The basic problem is very simple. Given n nodes are to be thrown into n bins, where each ball is chosen to each bin uniformly and independently, the focus is the maximum loaded bin, that is, the largest number of balls in all the bins; is approximately

\[ \frac{\log n}{\log \log n} \]  

(1)

If the balls are thrown sequentially and each ball is placed in the least loaded bins of \( d \geq 2 \), the maximum load is

\[ \frac{\log \log n}{\log d} + \Theta \]  

(2)

with high probability. We call this method “\( d \)-choice paradigm.”

When the number of balls \( m \) is larger than \( n \), especially \( m \gg n \log n \), the maximum load of random choice is

\[ \frac{m}{n} + \sqrt{\frac{m \log n}{n}} \]  

(3)

while for the “\( d \)-choice paradigm,” the maximum load is

\[ \frac{m}{n} + \frac{\log \log n}{\log d} \]  

(4)

Notably, there is also a very interesting and beneficial property in “\( d \)-choice paradigm.” This property motivates us to take this method into task assignment and balancing scheme where increasing the number of tasks will not increase the variance of the queue length.

The “\( d \)-choice paradigm” has been widely studied and extended for many interesting aspects. For example, many studies [8] focus on the weighted balls and bins, which is practical and useful in real task assignment. For computer system and networks, the parallel execution studies with overhead considerations are very useful and get many concentrations [6, 7].

In our study, the “\( d \)-choice paradigm” is useful but should be tailored for our investigated problem. We will introduce our system model and the investigated problem in the following paragraphs.

2.2. Modeling the Users and Tasks. We consider a mobile social network, where \( n \) users in user set \( U = 1, 2, \ldots, n \), will share their tasks among them. For each task, there is a metric in evaluating its difficulty, which corresponds to the weight of a task. \( 0 \leq w_{ij} \leq 1 \) is the weight of the \( j \)-th task on user \( i \). The weight can be represented as the difficulty for completing a task, which is directly relevant to task completion time as well as energy consumption. Each user \( i \in U \) holds a task set \( T_i = \{T_{i1}, \ldots, T_{in}\} \). For each task from user \( i \), when it is reallocated to another user \( j \), it will be processed at \( j \) without being further forwarded. We do not consider multiforwarding case in this study. We put it to future work and social mapping technology. The tasks in user
i have not been given priority here, and we do not consider the "dequeueing" and "enqueueing" technologies for task priority because in social task offloading network there should be very few urgent tasks for reassignment. The tasks with higher priority should be processed on local device.

The tasks getting from other users are listed in queue $Q_i = \{q_{i1}, q_{i2}, \ldots, q_{im}\}$, and the queuing length is $||Q_i||$.

2.3. Modeling the Energy Level and Computational Capability. Each user $i$ has an energy level $E_i$. For each processing of the reallocated task, a corresponding amount of energy is consumed. In our model, it is proportional to task weight. More difficult tasks will cost more energy and we set the energy-cost model linear here. For example, the energy cost for task $T_{ij}$ is $E_{ij} = \alpha w_{ij}$, where $\alpha$ is a scaling factor and depends on the user device. We assume energy consumption model among users is identical. The root reason is that we are focusing on the load balancing scheme in network. Also, we discuss the tasks and devices with different weights, which can be easily and reasonably extended to energy level. Combining the resources allocation and task reassignment will be left for future work.

Another fact is that the execution time is inversely proportional to the computational capability, which will accordingly affects the queuing length. In our model, the task queue is processed according to users’ capability; that is, the queuing length $||Q||$ is inversely proportional to the computational capability.

2.4. Modeling the Mobile Social Network. Each user performs random walk in a finite area. There are totally $n$ users randomly roaming in an area, where a communication range $R$ is set for each mobile user. When two nodes are in the communication range of each other, the tasks can be reallocated. As the data transmission rate is comparatively high with the task information, we assume the task reallocation period is sufficient when two nodes are roaming in communication range. We also assume that the inherent parameters in each user, for example, the energy level, queuing length, and the computational capabilities, can also be exchanged during the intercontact time between users. The basic scheme is shown in Figure 1. In this picture, we can see our model of task offloading and reassignment more clearly. When User 1 is doing the allocation, he or she is considered as the center of his or her communication circle. For example, there are in total 5 users within the communication range. User 1 just picks randomly two of them (User 2 and User 3) and deliver the task to the less loaded one (User 2). More extensively, User 1 can also reassign tasks to other users according to different metrics.

3. Problem Formation

3.1. Basic Problem. We investigate the task of offloading and balancing scheme for mobile social network. In particular, balancing among users means an even distribution of the tasks. The goal is to minimize the sum of queueing length difference between each one and the average value, which can be given by

$$\min \sum_{i \in U} |Q_i - E[Q_i]|,$$

where $E[\cdot]$ is the average value for random variables.

3.2. Evaluating the Gap between Maximum and the Average. It can be formulated by minimizing the gap between the average value and the maximum queueing length achievable with high probability. This evaluation has two advantages. First, it is simple. Instead of computing the gap between average value and all the queues, the maximum queueing length is investigated, which saves large amount of communication
Input: the number of users, \( n \); the number of time slots, \( m \);
communication range, \( r \);
location edge (users’ locations can’t go beyond the edge), \( \text{edge} \);
number of choice, \( d \);

Output: queue length: \( c_1, c_2, \ldots, c_n \)
queue length (considering task weight and users’ ability): \( w_1, w_2, \ldots, w_n \)

(1) Initialize \( \{c_1, c_2, \ldots, c_n\} = 0 \), \( \{w_1, w_2, \ldots, w_n\} = 0 \), \( \text{count} = 0 \)
(2) while \( \text{count} < m \) do
(3) randomly generate users’ locations: \((X_i, Y_i)\), \(0 < X < \text{edge} \), \(0 < y < \text{edge} \), \(1 \leq i \leq n \)
(4) for every user \( j \), \(1 \leq j \leq n \);
(5) if \((X_i - X_j)^2 + (Y_i - Y_j)^2 < r^2\) then
(6) record that \((X_i, Y_i)\) is in communication range of user \( j \);
(7) end if
(8) if number of users in range \( \geq d \) then
(9) randomly choose \( d \) of them
(10) \( s \) = the one with shortest length among the \( d \)-choice
(11) else if \( 0 < \) the number of users in range \( < d \) then
(12) \( s \) = the one with shortest length in range
(13) else if nobody is in range then
(14) continue allocation
(15) end if
(16) \( c_s = c_s + 1 \)
(17) \( w_s = w_s + \text{taskweight(count)/ability(s)} \)
(18) \( \text{count} = \text{count} + 1 \)
(19) end while
(20) return \( c_1, c_2, \ldots, c_n; w_1, w_2, \ldots, w_n \)

Algorithm 1: eCOTS algorithm for large-scale smart city sensing application.

4. Algorithm Design

4.1. eCOTS Overview. Our algorithm is leveraging two basic characteristics in random based paradigm with theoretical guarantees. One is the 2-choice paradigm, where investigating 2 choices is nearly optimal in practice. The other is the randomness in mobile sensor network, especially the random walk behavior. The stationary distribution over the visiting frequency and diversity among users provides sufficient choices as well as randomness for task reassignment. In summary, our basic idea is simple. There are basically two parts in our algorithm. First, the algorithm will select the users in communication range according to the number of users in range. Second, according to the "\( d \)-choice paradigm," the users are selected for task reassignment according to their queuing length. Notably, when the energy level and computational capability are considered, the problem is similar

overhead in distributed mobile networks. Without loss of
generality, such evaluation is also reasonable. Second, it is
also an important metric in evaluating the worst case between
the average cases. For example, we need to know the task
finishing time in average and the worst. Also, in batched task
offloading case, the task finishing time depends on the latest
one, which corresponds to the longest queuing length. Thus,
such evaluation can be given by

\[
\min \max_{i \in U} \| Q_i \| - E_{i \in U} [\| Q_i \|].
\] (6)

3.3. Allocate with Weighted Tasks and Different Computational
Capability. We investigate the allocation scheme \( A \) where
each task is given different weights. Similarly, the evaluation
can be represented by

\[
\min \max_{i \in U} \sum_{k=1}^{\| Q_i \|} w(q_{ik}) - E_{i \in U} [W(Q_i)],
\] (7)

where \( W(Q_i) = \sum_{k=1}^{\| Q_i \|} w(q_{ik}) \).

When the nodes are given different computational capa-
bilities, the representation can be given by

\[
\min \max_{i, j \in U} \sum_{k=1}^{\| Q_i \|} \frac{w(q_{ik})}{c_i} - E_{i \in U} [\bar{W}(Q_i)],
\] (8)

where \( \bar{W}(Q_i) = \sum_{k=1}^{\| Q_i \|} w(q_{ik})c_i \).
to weighted bin case in “balls and bins” theory. However, considering energy level brings negative impact. Such property is different from traditional weighted bin case. While we consider the computational capability case, it differs from weighted bin theory, because, for the traditional weighted bin problem, selecting the higher weight bin with higher probability is not favorable. However, in our concern, as the execution time is inversely proportional to the computational capability, selecting users in higher capability with higher probability accordingly is favorable and reasonable. In the following subsection, we will describe our algorithm eCOTS in detail.

4.2. Algorithm Description on eCOTS. According to the above derivation, we present the algorithm description on task allocation in mobile case as Algorithm 1. The input parameters are the number of users $n$, the number of time slots $m$, the communication range $r$, the location edge, edge (users’ location cannot go beyond the edge), and the choice number $d$. The amount of tasks a person has accepted is recorded as queue length $c$, but considering that tasks are not identical and users’ ability is in diversity, we provide the task length $w$. Here, we do our allocation work according to the apparent queue lengths of users.

At the beginning of our Algorithm 1, both queue lengths and the allocation time slot are initialized with 0. And then, when the allocation time slot arrives, we first randomly generate users’ locations $(X_i, Y_i)$; after that, for every single user $j$, we check whether the other users are in communication range and record the in-ones. If the number of users in range is no less than $d$, for example, 2 in 2-choice paradigm, we arbitrarily choose $d$ users out of them in advance instead of delivering the task to a random person. After comparison, we give the task to the one who has the shortest length among the $d$ selected users. When the number of users in range is smaller than $d$, we can also choose the shortest queue among them and allocate the task to the chosen one. Further, if there is nobody in communication range, we just continue the task allocation in the next time slot. After choosing the right person $s$ to deliver the task, let the queue length of $s$ add 1; $c_s = c_s + 1$.

Considering that every task has its own difficulty and each person’s ability to deal with the task is in diversity, the task length is totally different from what we can see; it is the actual complexity of every task queue. We measure this

Figure 2: Choice performance in the communication range.
with $w$; in addition, $w$ should be added by the corresponding value of person $s$ with ability $(s)$ processing the current task with task weight $(count)$. When all the users have finished their allocation at time $count$, we can continue our task allocation with above steps until the allocation time slot $count = m$.

5. Simulation Results

In this section, we conduct extensive simulations to evaluate eCOTS algorithm. The overall methodology and simulation settings are introduced firstly. Afterward, we evaluate the performance of eCOTS in terms of different parameter settings. Table 1 summarizes the parameters frequently used in this paper.

5.1. Implementation Methodology and Simulation Settings

In this simulation, we emulate a realistic and meaningful scenario for mobile social network as follows. For each time slot $i$, $n$ users randomly walked in a square region of $100 \times 100$ m. As a result, for user $k$, the location $(X_u, Y_u)$ is randomly generated, where $0 < X < 100, 0 < Y < 100, 1 \leq u \leq n$. Users in the communication range $r$ can contact with each other, switching basic information and reassigning the tasks. In each time slot, each user will share only one task; thus, $n$ users allocate their respective tasks (in total $n$) in each time slot. For each task, the person who launches the task allocation is considered as the center point of the communication circle. We calculate the distance with every other user to find the candidates. Among the candidates, we pick two of them randomly and compare the queue length

| Parameter                      | Setting       |
|-------------------------------|---------------|
| Total number of users $n$     | 100           |
| Communication range $r$       | 5, 10, 20 m   |
| Time slot number $I$          | 30            |
| Task weight                   | {1–5}, {1–50}, {1–500} |
| Users’ computational ability  | {1–5}         |

Table 1: Parameter settings.

Figure 3: 2-choice performance with different task weights, comparing queue lengths only.
between them, delivering the current task to the one with shorter queue length. In more practical case, the weight of tasks, the diversity of users’ ability, and energy consumption are also taken into consideration, and we set up the various situations for our experimental evaluations.

5.2. Performance Evaluation of eCOTS. We compare eCOTS with the simple random delivering approach based on four metrics: (1) the communication range; (2) tasks with different difficulties; (3) the users owning diverse abilities; (4) energy consumption. We evaluate the impacts of four factors on the performance of eCOTS as follows. Additionally, we consider the most complicated situation.

5.2.1. Impact of Communication Range. We need to compare eCOTS with the initial random allocation in different communication ranges. Here we choose 100 users and 30 time slots. To show the results more obviously, we experiment identical case, where each task has the unit weight (i.e., 1) and each user has the same ability to process the task.

As shown in Figure 2(a), when the communication range is set to 5 meters, eCOTS and random allocation do not have much difference. The main reason is that communication range is too small to find enough candidates. For example, if we find only one person in each search, eCOTS has nothing different with the random allocation. Moreover, if nobody can be found in communication range, there is no allocation among users then. With the increased communication range, we can discover obviously that eCOTS outperforms random allocation. Let us see Figure 2(b), in which the communication range is set to 10 meters, while random allocation brings about the maximum queue length with the users with 38 and the minimum with 17, the maximum load is twofold high over the minimum load. By contrast, eCOTS algorithm narrows the difference into 4 and the load of every user approach to the balance level (total task amount/user), 30, which is a favorable property. Moreover, as shown in Figure 2(c), the allocation results of eCOTS is better than random allocation too.

Further, we also perform the comparison between eCOTS and optimal case (reassign tasks to the lowest loaded users within the communication range) in Figure 4. Since the full line (eCOTS) almost coincides with the dotted one (optimal case), we have no doubt to believe that eCOTS algorithm works perfectly. With the eCOTS algorithm, no person bears too much work to do while some other users have little task to process.

5.2.2. Impact of Task Weight. We need to evaluate the impact of task weight on the eCOTS performance. In this experiment, we make a slight modification on the previous settings, where every task has its own weight. As we clearly know that different tasks have different difficulties, we should make the more difficult tasks owning far more weight than the easy one. The person we choose to deliver the task actually will not just add its task length by 1 but also the given task weight.

To demonstrate the better performance of eCOTS, we let the task weight vary from {1–5, 1–50, 1–500}. If we have access to know about the task length of everyone, we can have the picture as in Figure 5. It shows comparison between eCOTS and random allocation in task length. Comparing with Figure 3, the case that users cannot know the task load of others, only knowing the number of tasks. The difference between the maximum load and minimum load has been narrowed significantly. As shown in Figure 5(a), we find that eCOTS algorithm is almost the same as the optimal case, which indicates that eCOTS algorithm really works in task allocation.

In summary, eCOTS method achieves a much better performance rather than random allocation in terms of task weight.

5.2.3. Impact of Users’ Computational Ability. In social network, users have diversity in ability. An able man always should do more work; we take this factor into our consideration. To quantify different users’ ability, we simply set their ability with 1–5 levels. That is to say, if a person’s ability is 4, the processing speed will be 4 times faster than the basic level, that is, level one. Concretely in our experiments, when a person is picked to handle a task, the task length is added by $1/\text{ability}(j)$, where $j$ is the label of user.

As shown in Figure 6, we first allocate our tasks according to the queue length. While it shows great superiority in queue lengths of users, the performance is not satisfactory when incorporating into the diverse ability. On account of this issue, we assume that everyone’s ability is widely known in our experiments. In Figure 7, the full line represents our eCOTS method and the dotted one is on behalf of the random allocation method. Obviously, our method performs much better than the random allocation method because the distribution of queue length of users is centering at the perfect average point.

5.2.4. The Impact of Energy Consumption. Task execution will cost energy consumption. In task allocation, we must consider the metric of users’ energy. Apparently, we may not give the task to the user with limited energy although this
user may be the one bearing the least load. In other words, we need to consider users’ energy levels in advance and the queue length of every user as well. Each user has full energy 100 at the start of allocation. Every time when we allocate a task, we choose the energetic one (the one who has more energy) from random 2. The chosen user will lose 1 point energy. As shown in Figure 8, eCOTS method (the full line) performs much better than random allocation (the dotted line).

5.2.5. The Impact of Parameter-\(d\) Setting. After we finished all the work above, we begin to study the impact of parameter-\(d\) setting. In the “balls and bins” theory, it is well known that, if we increase the number of \(d\), the performance of \(d\)-choice will be even better. Based on this, we set parameter-\(d\) as \([2, 3, 4]\), and the results of identical case are shown in Figure 9. We can easily find that, although the performance of 4-choice is only a little better than 2-choice, considering that seeking two more users’ information will also cost a lot, 2-choice is good enough for our task allocation in mobile social network.

6. Trace Driven Evaluation

According to the simulation results above, we find that “2-choice” has shown great superiority in the typical scenario of mobile social network, while simulation is obviously not enough to convince our algorithm. As a matter of fact, we need trace-driven performance evaluation to put up the adaptability of “2-choice” in real world.

6.1. Trace Data Processing. The trace we use is Rollernet [14], which is the public data collected with iMotes in the roller tour in Paris. RollerNet was interested in tracking contacts between different mobile users. The data set has been collected in August 20, 2006. According to organizers and police information, about 2,500 people participated in the rollerblading tour. The total duration of the tour was about three hours. During the tour, the iMotes has been deployed to 62 skaters. The experiment successfully recorded contacts between not only all devices they distributed, but also a large...
amount of external devices (cell phones, PDAs). To make sure the trace is authentic and faithful, we only pick the contacts recorded between iMotes to continue our work.

Part of User 1’s original contact file is in Table 2; the first and second columns give the IDs of the devices which the contact is reporting. The third and fourth columns describe, respectively, the first and last time when the address of ID2 was recorded by ID1 or the address of ID1 was recorded by ID2 for this contact. The fifth column enumerates contacts with the same ID1 and ID2 as 1, 2, … . The last column describes the time difference between the beginning of this contact and the end of the previous contact with the same ID1 and ID2.

Back to our model, we do the allocation task in every time slot. Here, we should divide the original time data into a series of time slots. From Table 2, we find that Starting Times have a public basic value 1156000000. For convenience, let Starting Times subtract basic value 1156000000. After that, we change the time unit from second to minute and then sort the meeting time in ascending order, for example, 1156085064 − 1156000000 = 85064, 85064/60 = 1417.

6.2. Algorithm Implementation

6.2.1. One User Allocating Tasks. Let us start our algorithm in this way. First, we should find out the users that User 1
met in the same time slot. Then, in every time slot, if User 1 comes across more than $d$ users, randomly picks $d$ of them and offload the task to the least loaded one. Otherwise, if the number is smaller than $d$, then just give the task to the least loaded one among them. Setting $d$ in different values, we get graphs as Figure 10. In Figure 10(a), when $d = 2$, although the figure is a little undulating, comparing with the performance of the random allocation below, 2-choice has a superiority with no doubt. Moreover, larger $d$ will lead to better task balancing performance. Remarkably, the value of $d$ cannot reach too high because the number of users encountering in a time slot is limited.

6.2.2. All Users Allocating Tasks. In the social mobile network, users reassign their tasks distributively. We should take all users into consideration as Figure 11(a). We prefer that users do their allocation in the same time slot. Of course, the longer the contact interval is, the more people will do their allocation in one time slot. We set the interval into 60, 120, and 300 seconds, respectively; the performance of eCOTS ($d = 2$) allocation and random allocation is shown in Figure 11. We find that the queue length of eCOTS is relatively stable, while that of random allocation is highly dynamic. When time interval is 300 seconds, standard deviation is bigger than that of 60 seconds mainly because users will meet more users in a single time slot and the $d$ users are more uncertain. If the time interval is too small to get enough user to choose, we have to give the task to the only one in the current time slot. In our experiments, the queueing length of user with ID23 does not reach the average level most of the time. After checking the original trace data, we find the reason is that the contact records of User 23 are fewer than others’; when $d = 4$; the CDF figure of eCOTS and random allocation is shown in Figure 12. As expected, eCOTS still performs better than random choices. With the increased contact interval, eCOTS performs better. Notably, we also find that, when contact interval increases to 300 s, the performance of eCOTS improves significantly, where minimum length and maximum length are 18 and 35, respectively. While for random choices, although the minimum queueing length also reduces (to 15), the maximum length is not decreased accordingly (still over 45).

7. Related Work

Our work relates to the efficient data transfer scheme over disruption-tolerant networks or opportunistic networks. The intermittent contact between randomly roaming users is useful for data sharing between mobile users. These networks have been studied extensively in a variety of settings, from military [4] to disasters [5]. These settings all assume that the fixed infrastructure is unavailable or costly even highly unreliable. With numerous cheap and distributed working terminals, some expensive works can be accomplished successfully. Further, the communication links are subject to disruptions, and the opportunistic access will bring more chances for data sharing. In summary, our work relates to three research topics, which are crowdsourcing schemes,
random walk, and “balls and bins” theory. As “balls and bins” theory has been introduced in Section 2, in this section, we mainly introduce the related works on crowdsourcing schemes for cooperative task accomplishment and the literatures on random walk studies.

7.1. Crowdsourcing Schemes for Cooperative Task Accomplishment. Many working crowdsourcing systems are concentrating on many researchers and industrial efforts in terms of designing actual platforms like [13, 15, 16]. These working systems also need extensive evaluation with solid results like [1, 2]. Further and more importantly, there are many studies focusing on pricing schemes for effective crowdsourcing incentives [3]. In smart city sensing applications, crowdsourcing paradigms leverage the pervasive human behavior, for example, walking, driving, shopping, and so forth, to provide a large-scale urban sensing network with wider coverage in time and space domain. Moreover, the social relationship, for example, the crowd gathering and migrations, is also important for some specific applications, for example, the flu influence detection, air quality, traffic monitoring, and so forth. The popularity of smartphone also speeds up the crowdsourcing based applications for urban sensing. Recently, the crowdsourcing based sensing applications are exploited to monitor the urban environment [17–20]. More specifically, Mun et al. [19–21] employ the customized and portable sensors on each
participant to monitor the air quality of the city. The constructions of noise map for the smart city are discussed in [17, 22]. Leveraging the cell phone microphones of the participants, these works focus on the implementation of the monitoring system. However, they fail to consider the unreliability and inaccuracy of the observations in the participatory sensing.

7.2. Random Walk. The random walk concept was proposed by Pearson [23]. We are interested in random walks in city scale, where a walker starts from a source node to a destination node and for each step of this travel. Note that, in mobile social network, the social relationship and human behavior dominate the trajectories of the human graph. Although in some specific trajectories, it does follow the random walk character, the mass number of users can form a relatively steady distribution of random walk. The inherent reason is the law of large numbers [24].

Fortunately, random walks can be put into mobile social network for exploring the character and opportunities in data transmissions. For instance, Newman [25] proposes the random-walk betweenness centrality, such kind of metric reasonably defines how often a node in a graph is visited by a random walker between all possible node pairs. Similar to the betweenness evaluation, Noh and Rieger [26] introduce the random-walk closeness centrality metric, which measures how fast a node can successfully get a random-walk message from other mobile nodes, in the random deployed system, such as mobile social networks. These works provide us with a guarantee that there are relatively robust closeness between

\[ \text{Performance of eCOTS (d = 2) and random allocation in different contact intervals.} \]

![Graphs showing performance comparison between eCOTS and random allocation in different contact intervals.](image-url)

Figure 11: Performance of eCOTS (d = 2) and random allocation in different contact intervals.
users even if they are randomly mobile users. And further, the messages can be delivered relatively stable among users with high probability of convergence.

The main focus of this paper is load balancing in distributed crowdsourcing system. Different from previous crowdsourcing based applications, we use a pure distributed computation and communication model, where users need not transmit any messages for centralized computation. In large-scale urban sensing application, such property would be welcome because it will not bring much trouble to the mobile users. Also, difficult tasks can be cooperatively shared by mass number of users, which can fully explore the energy usage among users and provide more reasonable usages on computational capability.

The random walk model in our study is also realistic and applicable to mobile social networks. We use the simulation model as well as real trace data. Both network scenarios have shown the effectiveness of our proposed scheme.

8. Conclusion

In this work, we investigate the task offloading and reassignment problem in mobile social network. In particular, we focus on the load balancing issue for efficient task execution for energy constrained mobile device. We propose eCOTS (Efficient and Cooperative Task Sharing for Large-scale Smart City Sensing Application). Our algorithm leverages the "d-choice paradigm" for "balls and bins" problem. When mobile
users are in communication range, only 2 users are selected and compared for the least loaded checking. Such simple scheme ensures balanced allocation even the energy level and computational capability are highly dynamic.

In future work, we are going to investigate the multilevel forwarding case for efficient and balanced load allocation. Also, trace-driven performance evaluations are needed for more convincing algorithm validation.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

This research is partially supported by NSF China under Grants nos. 61003277, 61232018, 61272487, 61170216, and 61172062, NSF CNS-0832120, NSF CNS-1035894, and China 973 Program under Grants nos. 2009CB320400, 2010CB328100, 2010CB334707, and 2011CB302705.

References

[1] M. S. Bernstein, J. Brandt, R. C. Miller, and D. R. Karger, “Crowds in two seconds: enabling realtime crowd-powered interfaces,” in Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology (UIST ’11), pp. 33–42, October 2011.

[2] M. S. Bernstein, G. Little, R. C. Miller et al., “Soylent: a word processor with a crowd inside,” in Proceedings of the 23rd Annual ACM Symposium on User Interface Software and Technology (UIST ’10), pp. 313–322, October 2010.

[3] P. G. Ipeirotis, F. Provost, and J. Wang, “Quality management on Amazon Mechanical Turk,” in Proceedings of the Human Computation Workshop 2010 (HCOMP ’10), pp. 64–67, July 2010.

[4] P. Marshall, “DARPA progress towards affordable, dense, and content focused tactical EDGE networks,” in Proceedings of the IEEE Military Communications Conference (MILCOM ’08), November 2008.

[5] K. Fall, G. Iannaccone, and J. Kannan, “A disruption-tolerant architecture for secure and efficient disaster response communications,” in Proceedings of the International Conference on Information Systems for Crisis Response and Management (ISCRAM ’10), 2010.

[6] L. Rudolph, M. Sliivkin-Allalouf, and E. Upfal, “A simple load balancing scheme for task allocation in parallel machines,” in Proceedings of the 3rd Annual ACM Symposium on Parallel Algorithms and Architectures, pp. 237–245, 1991.

[7] V. Stemann, “Parallel balanced allocations,” in Proceedings of the 1996 8th Annual ACM Symposium on Parallel Algorithms and Architectures, pp. 261–269, June 1996.

[8] M. Michael, A. W. Richa, and R. Sitaraman, The Power of Two Random Choices: A Survey of Techniques and Results, 2005.

[9] A. Broder and M. Mitzenmacher, “Using multiple hash functions to improve IP lookups,” Tech. Rep., Department of Computer Science, Harvard University, Cambridge, Mass, USA, 2000.

[10] A. Broder and A. Karlin, “Multi-level adaptive hashing,” in Proceedings of the First Annual ACM SIAM Symposium on Discrete Algorithms, 1990.

[11] A. Czumaj, F. Meyer, A. der Heide, and V. Stemann, “Shared memory simulations with triple-logarithmic delay,” in Algorithms, Lecture Notes in Computer Science, pp. 46–59, 1995.

[12] R. M. Karp, M. Luby, and F. Meyer, “Efficient PRAM simulation on a distributed memory machine,” Algorithmica, vol. 16, no. 4-5, pp. 517–542, 1996.

[13] Crowdflower, https://crowdflower.com/.

[14] P. U. Tournoiux, J. Leguay, F. Benbadis, V. Conan, M. D. De Amorim, and J. Whitbeck, “The accordion phenomenon: analysis, characterization, and impact on DTN routing,” in Proceedings of the 28th Conference on Computer Communications (IEEE INFOCOM ’09), pp. 1161–1124, April 2009.

[15] Casting words, https://castingwords.com/.

[16] Crowd spring, http://www.crowdspring.com/.

[17] R. K. Rana, C. T. Chou, S. S. Kanhere, N. Bulusu, and W. Hu, “Ear-phone: an end-to-end participatory urban noise mapping system,” in Proceedings of the 9th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN ’10), pp. 105–116, April 2010.

[18] M. Faulkner, M. Olson, R. Chandy, J. Krause, K. M. Chandy, and A. Krause, “The next big one: detecting earthquakes and other rare events from community-based sensors,” in Proceedings of the 10th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN ’11), pp. 13–24, April 2011.

[19] W. Willett, P. Aoki, N. Kumar, S. Subramanian, and A. Woodruff, “Common sense community: scaffolding mobile sensing and analysis for novice users,” in Pervasive Computing, vol. 6030, pp. 301–318, 2010.

[20] P. Völgyesi, A. Nádas, X. Koutsoukos, and Á. Lédeczi, “Air quality monitoring with SensorMap,” in Proceedings of the International Conference on Information Processing in Sensor Networks (IPSN ’08), pp. 529–530, April 2008.

[21] M. Mun, S. Reddy, K. Shilton et al., “PEIR, the personal environmental impact report, as a platform for participatory sensing systems research,” in Proceedings of the 7th ACM International Conference on Mobile Systems, Applications, and Services (MobiSys ’09), pp. 55–68, June 2009.

[22] N. Maisonneuve, M. Stevens, and B. Ochab, “Participatory noise pollution monitoring using mobile phones,” Information Polity, vol. 15, no. 1, pp. 51–71, 2010.

[23] K. Pearson, “The problem of the random walk,” Nature, vol. 72, no. 1865, p. 294, 1905.

[24] P. K. Sen and J. M. Singer, Large Sample Methods in Statistics, Chapman & Hall, New York, NY, USA, 1993.

[25] M. E. J. Newman, “A measure of betweenness centrality based on random walks,” Social Networks, vol. 27, no. 1, pp. 39–54, 2005.

[26] J. D. Noh and H. Rieger, “Random walks on complex networks,” Physical Review Letters, vol. 92, no. 11, pp. 118701–1, 2004.
