IFIP Networking 2019 1570511170

Data Collection and Node Counting by Opportunistic Communication

Tong Li[†], Sylvia T. Kouyoumdjieva[⋆], Gunnar Karlsson[⋆], and Pan Hui[†‡]
[†]Department of Computer Science & Engineering, The Hong Kong University of Science and Technology, Hong Kong
[⋆]Department of Network and Systems Engineering, KTH Royal Institute of Technology, Stockholm, Sweden
[‡]Department of Computer Science, University of Helsinki, Helsinki, Finland
E-mail: t.li@connect.ust.hk, {stkou, gk}@kth.se, panhui@cse.ust.hk

***Abstract*—Ever more powerful mobile devices are nowadays capable of collectively carrying out reasonably demanding computational tasks without offloading the processing to an edge server or a distant cloud-computing service. In this work, we explore such distributed computing and study how it is affected by the mobility as well as the number of nodes that collaborate. We choose distributed counting of a number of nodes in an enclosed area as application. Such application is useful for estimating attendance at events and measuring occupancy for facility management, as needed for monitoring of crowdedness with respect to safety and evacuation, climate control and comfort. For this application, we are interested in determining the time until all nodes know the correct number of nodes in the space where they reside.**

Our study shows the effect of mobility on the distributed process. We find that the process of collecting data opportunistically from a closed set of nodes is well described by empirical laws that we derive. We discuss the results and suggest further work needed to understand opportunistic computation and to develop it as a new model of computation among collectives of mobile nodes.

***Index Terms*—opportunistic computing, distributed computing, data collection, counting**

I. Introduction

Mobile cloud computing and mobile crowd-sensing solutions propose that mobile nodes act as sensors collecting and reporting data, while the actual computation is offloaded to the cloud. The results of the computation are afterward reported back to those nodes that are interested in it. However, the increase in computing power, available storage and communication capabilities of mobile devices enables local processing and storage of data, rather than offloading it to remote servers. This makes a collective of mobile nodes self-sufficient with respect to many computational tasks, and it becomes independent of any infrastructure. There are advantages to this in terms of cost, privacy, and availability.

In this work, we apply the principle of opportunistic networking to the domain of distributed mobile computing. In an opportunistic network, nodes exchange information whenever they come in a direct contact by some device-to-device communication technology [1]. The underlying node mobility contributes to the mixing of nodes, and thus to the spreading of data. We here consider a node to be a person carrying a device, such as a mobile phone, equipped with a wireless communication interface [2]. In the context of distributed computing, each node that participates in the opportunistic network acts as a collector and processor of data, as well as the disseminator of the result of the computation. The computing is often iterative among the nodes.

ISBN 978-3-903176-16-4 ©2019 IFIP

Distributed computing in opportunistic networks is not always straightforward, especially when dealing with open systems with node churn, i.e., nodes that enter and leave the space in which the distributed computations are performed. As an example of distributed computing in open systems with mobile nodes, we have previously evaluated the performance of a distributed node-counting protocol and have shown that the accuracy of the counting measure is strongly dependent on the node density [3]. Moreover, when dealing with open systems, there may be two different definitions of node count: number of nodes present (1) over the lifetime of the system, or (2) over a limited period. In the first case, the count is cumulative across all unique nodes that have ever been in the area. In the latter case, the count also needs to deduct nodes that leave the system during the period for which counting is being performed. As previously demonstrated in [3], the second definition is more challenging, and providing an accurate count is not trivial.

In closed systems, on the contrary, the number of nodes is fixed but unknown and needs to be estimated. A use case is to estimate attendance at an event where churn is low and may be neglected. Thus, the counting measure is well-defined: any distributed computation is finalized when all (interested) nodes in the system have converged to the same computed value. Closed systems often allow for better understanding of distributed processes.

In this work, we study the performance of distributed computing in the opportunistic network in the context of closed systems. We allow nodes to exchange data opportunistically via device-to-device communication and to perform computations locally on the obtained data. We present results for the use-case of aggregate counting as a simple example of a distributed task. The ultimate goal for any distributed task is two-fold: (1) each node, participating in the collective task, can determine by itself the accuracy of its computed result at any given point in time (e.g., that the process has converged), and (2) an outside observer is able to determine the accuracy of the result if requested from a randomly selected computing

node (i.e., if nodes are to report their estimates to a centralized unit). We are predominantly interested in the second goal, and to this end, we address the following questions:

- How does the computation time depend on the mobility and number of nodes?
- How does the counts in the nodes converge towards a unanimous value?
- Can the computation be modeled independently of the mobility?

The remainder of this paper is organized as follows. In Section II, we position our work concerning previous contributions in the field of distributed aggregation. In Section III we outline a use-case of distributed aggregation, namely distributed counting in mobile environments. Section IV presents a model of the counting process for closed systems. Finally, we discuss the results and conclude the study in Section V, and present directions for further work.

II. Related Work

The topic of data dissemination in opportunistic networks is well-studied, both in the context of closed and open systems. In [4], Wang *et al.* discuss the potential of mobile opportunistic networks in the context of time and location sensitive information dissemination. Vatandas *et al.* [5] focus on modeling the characteristics of opportunistic networks to represent data dissemination realistically. In [6], Passarella *et al.* present an analytical model that describes the dependence between the node inter-contact times and the aggregate inter-contact times in mobile opportunistic networks. Hernández-Orallo *et al.* [7] present an analytical model based on population processes to characterize opportunistic data dissemination in the context of 5G networks, and obtain closed-form expressions for determining diffusion time, network coverage, and waiting time. In [8], Kouyoumdjieva and Karlsson study opportunistic data dissemination in the context of mobile data offloading under the assumption of an open system. All of these works, however, allow nodes to only exchange information with each other, without performing any computations or modifications of the shared data. Thus, they only partially utilize the potential of opportunistic networks.

The work presented in this paper falls under the broader area of opportunistic computing. It is a paradigm that considers the use of any available resource in an opportunistic network by utilizing sporadic contacts occurring when two (or more) mobile nodes come in direct communication range [9]. As opposed to opportunistic data dissemination, nodes share and exploit each other's software and hardware resources not only to exchange information but also to execute tasks. One such task is computing an aggregate value across all nodes in the network.

Various gossiping algorithms have been proposed to address the problem of computing an aggregate value, i.e., a sum or an average of a given measure, over a network. Typically nodes exchange messages in rounds, and at each round, only one pair of nodes (chosen at random or pre-defined) can exchange information and perform some computation with the objective to converge to a common value across all network nodes [10]. Gossiping algorithms may be operated in the presence of churn, i.e., nodes joining and leaving the network. In [11] Jelasity *et al.* propose a fully distributed gossip-based aggregation protocol for large dynamic networks. Nodes in the network utilize pairwise interactions to assure quick convergence to the desired aggregate value. In [12] Shi and Srimani propose an online distributed gossiping protocol for mobile networks where every node has only a limited knowledge of its neighbors. As opposed to gossiping algorithms, in opportunistic computing, there may be more than one node pair communicating at a time. Moreover, the list of neighbors is usually not known in advance and there is no sampling of all neighboring nodes before one communication is established with one of them. This is often due to the dynamics of the system which occur faster than what is typically considered in studies on gossiping algorithms; i.e., a typical link duration in an opportunistic network with underlying pedestrian mobility is around 10 s [13].

Mobile or fog computing has also been suggested as a way of bringing computational power closer to end users, especially in the context of the large-scale Internet of Things (IoT) networks [14]. The fog can in principle constitute any devices with sufficient storage, processing power, and energy for carrying out distributed computing tasks. Few works, however, consider expanding the mobile fog to utilize the processing power of end devices, either mobile devices carried by humans or the sensor devices. In [15], Di Pascale *et al.* present a generalized framework for leveraging resources in ultra-dense IoT mesh networks by coupling data communication and processing. The authors map an artificial neural network on top of the IoT network and are thus able to exploit the communication between devices to perform data processing and aggregation while reducing computational latency and improving the energy distribution across the network. However, the framework is currently only evaluated for scenarios in which all nodes in the network are static. Moreover, the authors assumed that the size of the network and the functions of nodes within the network are known in advance.

The concept of mobile crowd sensing provides a way of utilizing resources on mobile devices for data collection and processing [16]. A variant, opportunistic sensing, allows data to be collected in the background without active involvement of the user [17]. Mobile crowd sensing applications are usually client-server-based, with mobile devices sensing and reporting data to a central server (located somewhere in the cloud) that further processes and distributes the data to those users who request access to them. Although peer-to-peer alternatives have been suggested by Jiang *et al.* [18], those solutions only suggest that nodes share information directly with one another. However, data processing is still executed at the server. As opposed to mobile crowd sensing, opportunistic computing allows nodes to act as sensors and processors of the information, as well as to exchange data and computational results with one another. Finally, the application of node counting is surveyed for non-imaging techniques in [19].

III. Distributed Counting via Opportunistic Communication

A. The computing task

To demonstrate the potential of distributed computing via opportunistic communications, we consider a useful yet simple computing task, namely counting a node population in a closed system. The computing task is considered executed when all participating nodes in the system have converged to the correct count. In this work, we assume that all nodes in the system are participating and collaborating for the computation of the task.

We assume N unique nodes (represented by a set $\mathcal{N}$ of node IDs, with cardinality N) that are distributed at random in a given area. At the beginning of the counting process, each node $n_i \in \mathcal{N}$ is only aware of its own presence in the area, thus its state is set to $S_i = \{n_i\}$. The cardinality of the set S_i, $|S_i|$ denotes the current node count in the area as perceived by node n_i. Whenever two nodes, i and j, come in direct communication range, they exchange their state sets, and each node forms the union of its own state set with the received set: $S_i = S_j \leftarrow \{S_i \cup S_j\}$ Thus, the cardinality grows by the number of nodes that each one adds to its set. Eventually the set will include all node identities and the cardinality will then be $N = |S_i|, \forall_i$.

B. Protocol design

The node set can grow without limit when the count of the population is taken over all nodes ever present in the lifetime of the system. Representing sets as bit vectors may then cause issues related to storage space on mobile devices. Furthermore, as we rely on wireless communication for the spreading of data, the set representation should be kept small, possibly fitting into a single beacon message for the wireless channel without the need for fragmentation. To address these issues, we use D-GAP compression [20] for storing the state vector of each node. D-GAP compression can be considered as a specialized variant of run-length encoding which provides a compressed representation of bit vectors in the form of integer vectors (later referred to as D-GAP vectors). Consecutive 0s or 1s that are part of a bit vector are represented by a single integer as part of a D-GAP vector. The first bit of the D-GAP thereby determines if an integer represents a sequence of 0s or 1s. A leading bit of 0 shows that the first integer corresponds to a number of consecutive 0s, followed by a number of consecutive 1s and so on; a leading bit of 1 indicates the opposite behavior. For instance, the D-GAP equivalent of the bit vector 1000 0000 0001 is [**1**] 1 10 1, where with [**1**] we denote the leading bit. We note that D-GAP compression is particularly beneficial when storing vectors that have long sequences of 0s or 1s, which makes it highly appropriate for the use-case of distributed counting.

Let us consider an example of four nodes to show the functional principles of the proposed protocol for distributed opportunistic node counting (Fig. 1). Upon entry into the area, a node inserts itself into its D-GAP state vector. For instance, node 1 in Fig. 1 has inserted itself in position 1 in its state D-GAP vector which is initialized with [**1**] 1. We have further indicated the three empty positions in the D-GAP vector of node 1 for illustration purposes; in reality, a node entering the area has no prior knowledge as to how many nodes are expected to be present, and thus its D-GAP vector will change and increase as the discovery process continues. We note that to preserve privacy, the node ID is determined by a cryptographic hash function, which is calculated over a unique identifier, e.g., the node's MAC address. Each node broadcasts a beacon at regular intervals, which comprises its state D-GAP vector to other nodes in its direct communication range. Upon reception of a beacon from another contact, a node updates its local state D-GAP vector. For instance, in Fig. 1 nodes 2 and 4 have previously exchanged their state vectors, while nodes 3 and 4 are currently in the process of exchanging their state vectors and thus have identical views on the population in the area.

We define three operations on D-GAP vectors: namely `merge`, `consolidate` and `append`, to assist the process of combining two D-GAP vectors. We refer the interested reader to [3] for further detail on these definitions. In a nutshell, the protocol operates as follows. Upon receiving a new vector, a node first checks in a local database whether the received vector carries potentially new information. This check is currently done by looking up the `node-ID` in a local database (a simple table containing the `node-ID` and `updated` fields, where the `updated` field holds the last time when the node received new information from the peer). If the `node-ID` is found, there has been prior communication with this peer and the advertised `updated` field value is compared to the previously registered `updated` field value. If they do not differ, the local state vector stays unchanged. Otherwise, the protocol consecutively executes the operations `merge`, `consolidate` and `append` in order to update the local estimate. Thus, the state vector contains cumulative information of all nodes that have been counted over time. This is the computation taking place in a node with each update of its state set.

C. Mobility scenario

We assume all nodes to follow a random walk (RW) mobility. It is commonly used in computer simulations to test the robustness of mobile network functions concerning user mobility [21]. In RW mobility, the movement of nodes is governed in the following manner. Each node selects a random destination in the simulation area (i.e., a 500 m $\times$ 500 m closed space) and a random speed. The destinations follow a uniform distribution and the speed v follows a Gaussian distribution, $v \sim N(1, \sigma^2)$, where the mean speed is 1 m/s and σ represents the standard deviation (STD). When a node arrives at its destination, it will simultaneously pick a new speed and a new destination. Thus, there is no pausing in our used RW model; all nodes are constantly moving during the simulations. Initially, the nodes are uniformly and randomly distributed in the simulation area. In practice, we generate

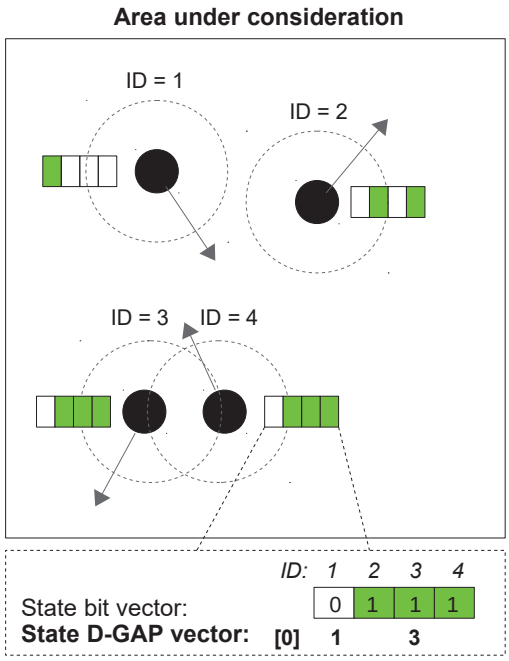


Fig. 1. An example of four mobile nodes executing the protocol for distributed computing in an opportunistic network. Information in each node is represented by a state D-GAP vector; the respective bit vectors are shown for clarity.

the traces of nodes in advance and store them in a trace file containing a snapshot of the positions of all nodes in the system every 0.6 s.

D. Simulation setup

In our simulation scenario, we assume that all nodes carry devices and are participating in collectively gathering data from other nodes. For the simulation, we use an implementation of an opportunistic content distribution system in the OMNeT++ modeling framework MiXiM [22]. Each simulation run is executed in synchronous rounds of 0.6 s corresponding to the granularity of the mobility traces we use. At the beginning of each round, nodes broadcast their state sets of nodes, i.e., their gathered data. To avoid collisions on the wireless medium, the broadcast transmission of each node in each round is distributed uniformly at random from 0 s to 0.5 s. The communication range of each node is set to 10 m.

E. Performance metrics

We use the following performance metrics and definitions to evaluate and monitor the processing of data gathering under opportunistic communications.

- *Number of contacts* (C) and contacts per time epoch (c) (the rate) reflect the dynamism of data dissemination. In our work, one contact for a node refers to that node receiving a message from another node.
- *Fully knowledgeable node* are those nodes that have gathered all node IDs in the system.
- *Accuracy* (Δ) of the system shows the ratio of fully knowledgeable nodes to all nodes in the system. It reflects the likelihood of hearing the correct count if a node is selected at random to report. Let X be the number of fully knowledgeable nodes and N be the total number of nodes in the system. Then, the accuracy is calculated as: $\Delta = X/N$. When the accuracy equals 1, all nodes have gathered all node IDs.
- *Knowledge gain* (G) of each node and for each contact shows how many new IDs that the node learns during a contact. It corresponds to the increase in counting per contact.

IV. Modeling of the Data Collection Process

A. Effects of number of contacts

We first look into how the number of nodes and their mobility affect the contact rate.

First, we depict how the contact rate changes with time in Fig. 2. We remark that a contact here is perceived on the application layer, i.e., when a broadcast message is received by a node. Thus the estimated contact rate may differ from the actual physical contact rate due to interference on the wireless channel. We show the number of contacts in each time bin for two different values of the speed STD, for $\sigma = 0.4$ and $\sigma = 0.1$. Time bins here correspond to the granularity of the mobility trace, i.e., 0.6 s. We notice that the average contact rate does not change dramatically over time, it merely fluctuates periodically. This may be caused by the collision avoidance mechanism in the MAC layer. The average number of contacts in each period is almost constant.

Fig. 3 shows the average number of contacts across different node populations for two different values of the speed STD. As shown in Fig. 3, when there are 300 nodes, the average number of contacts is nearly 12, while it drops to around 2 when the number of nodes is 100. Decreasing the STD value leads to a slight decrease in the average number of contacts, especially when the node population is larger; for sparse populations, i.e., 100 nodes, the average number of contacts is not affected by the speed STD. Hence, in the next step, we try to model the relation between the number of nodes and the average number of contacts in each time bin without consideration for the speed STD.

Table I shows that the power function has the lowest root mean square error (RMSE) of all considered functions. For N nodes, the number of contacts in each time bin, $\hat{c}$, is thus almost proportional to the square of the number of nodes,

$$\hat{c} = 6.856 \cdot 10^{-5} \cdot N^{2.117}. \quad (1)$$

The contact rate depends only on the number of nodes and not significantly on the variability of the speed according to Fig. 3. However, the mixing of nodes could be affected by the spread of the speed distribution. For this reason, we now consider the time for counting N nodes and the convergence towards the full count.

TABLE I
CURVE FITTING WHERE N IS THE NUMBER OF NODES AND $\hat{c}$ IS THE CONTACT RATE (NUMBER OF CONTACTS IN EACH TIME BIN).

Function	Expression	RMSE
Exponential	$\hat{C} = 1.962\exp(5.838 \cdot 10^{-3} \cdot N)$	2.20
Polynomial	$\hat{C} = 0.086 \cdot N - 10.64$	3.390
	$\hat{C} = 1.56 \cdot 10^{-4} \cdot N^2 - 7.59 \cdot 10^{-3} \cdot N + 0.28$	0.438
Power	$\hat{C} = 6.856 \cdot 10^{-5} \cdot N^{2.117}$	0.396
	$\hat{C} = 8.418 \cdot 10^{-5} \cdot N^{2.085} - 0.2278$	0.467

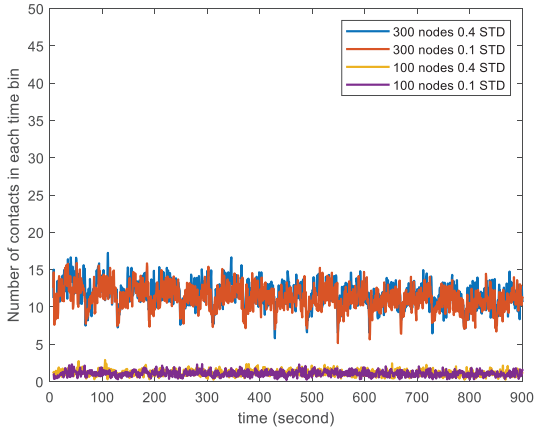


Fig. 2. The average number of contacts in each time bin vs. simulation time. Time bins here correspond to the granularity of the mobility trace, i.e., have a duration of 0.6 s.

B. Effects on accuracy

Since the number of contacts per time unit is independent on the speed STD, we regard the opportunistic computation as being clocked by contacts rather than by time bins.

Fig. 4 shows how the accuracy of the counting changes with the number of contacts. Note that we consider all contacts, regardless of whether they bring new data to the node or not. Specifically, in the 300-node case, after about 12,000 contacts, the first fully knowledgeable node appears, and all nodes have gathered all data after about 26,000 contacts. When the number of nodes is 500, the first fully knowledgeable node appears after about 30,000 contacts, and the accuracy becomes one after about 60,000 contacts.

The increasing number of nodes leads to higher contact rate, as shown above, hence more opportunities to gather all node IDs in a given time. However, there is also a higher number of nodes to be counted. To see the combined effect, we derive the time to count all nodes and consider the accuracy as well.

To evaluate the effect of the number of nodes on the accuracy, we fix the accuracy and measure how the number of needed contacts changes with the number of nodes. We set accuracy to 0.2, 0.4, 0.6 and 0.8 respectively, and show the results in Fig. 5. To achieve the same accuracy, more nodes need more contacts, as expected. The differences caused by speed STD can be observed in the cases of large numbers of nodes.

We model the relation among the number of nodes (N), (total) number of needed contacts (C) and accuracy (Δ). We first set the speed STD to 0.1 and try the exponential, polynomial and power functions as fitted curves for the number of nodes versus the number of contacts under different accuracy scenarios. We find that a power function has the lowest RMSE. Specifically, the expression is

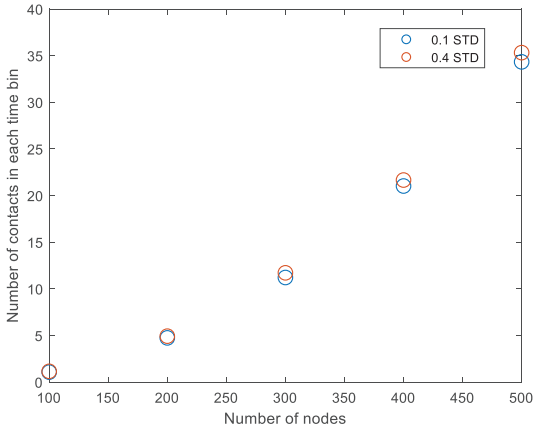


Fig. 3. The average number of contacts in each time bin vs. number of nodes (time bins 0.6 s).

$$C = (0.5478 \cdot \Delta + 0.1389) \cdot N^{1.7652}. \quad (2)$$

Letting Eq. (2) be divided by Eq. (1), we are able to derive the model of the number of time bins, $N_{TB} = C/\hat{c}$,

$$N_{TB} = \frac{(7.9901 \cdot \Delta + 2.0260) \cdot 10^3}{N^{0.3518}}, \quad (3)$$

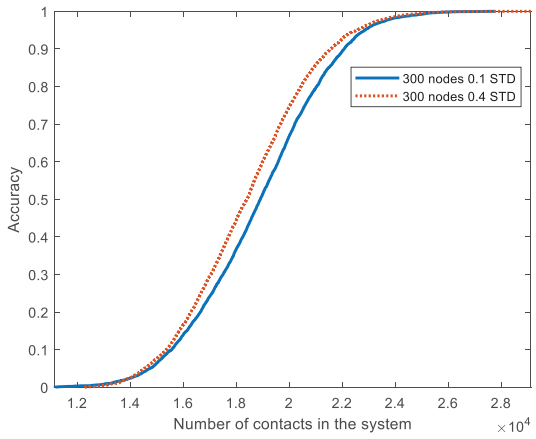
where each time bin is 0.6 seconds (see Section III).

Eq. (3) reveals that the needed time to compute the count to a given accuracy increases linearly with the accuracy while it slowly decreases as a power law with the increasing number of nodes. To exemplify, for 500 nodes (which for the area is one node per 500 square meters), it will take 11 minutes for them to know how many they are.

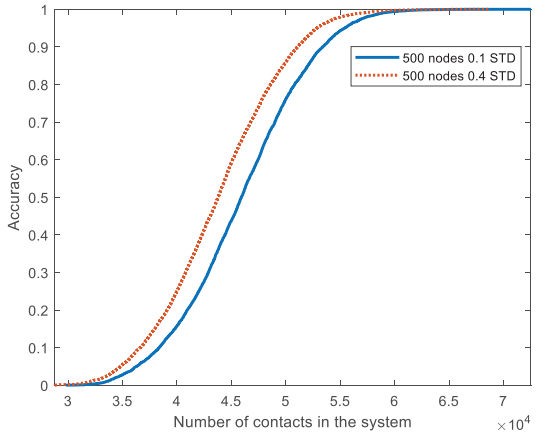
We now go back to the difference in convergence with respect to the speed STD. The accuracy rises more quickly during the middle period of opportunistic communications for higher speed STD, as noted in Fig. 4, but it does not affect the convergence time. We then set the number of nodes to 300 and 500 and show the distribution of count across all nodes in Fig. 6. We take snapshots of the distribution at specific times (clocked by contacts) and use box-plots to depict them. In the box-plots, the candlesticks depict the bottom 10% and the upper 90% of the count values while the boxed area contains the values between 25% and 75%. The horizontal line depicts the median of the distribution.

We note that the upper 90% of the counts increase to the maximum value quickly, but the growth rate of the bottom 10% of the nodes has tiny counts. There is not any discernible influence on the speed STD in these extreme points. For instance, when the number of nodes is 300, after 1.8×10^4 contacts in the system, the median of count values nearly equals to 300, i.e., the maximum value. However, the bottom 10% count values are still lower than 150.

We find that the speed STD affects the counting procedures for the higher number of nodes. The median of counts for 0.4



(a)



(b)

Fig. 4. Accuracy versus the number of contacts in the system: (a) 300 nodes, (b) 500 nodes.

speed STD and 500 nodes is always slightly higher than that for 0.1 speed STD, corresponding to the findings in Fig. 4. The difference is more pronounced for the 25th percentile. We will further analyze the effect of speed STD during the processes of opportunistic communications in Section IV-C below.

C. Effects on knowledge gain

As for the speed STD, we know now that it influences the accuracy during the middle period of the opportunistic computation and that the increasing number of nodes makes this effect more noticeable. Therefore, in this subsection, we use the knowledge gain (G) to model the processes of opportunistic computation.

The knowledge gain reflects how many new node IDs that are gathered during each contact. Hence, it can be used to capture how data are disseminated or collected during the counting processes using opportunistic communications. Similar to Section IV-B, we use contacts to clock the communication procedure and depict how the knowledge gain changes during the processes of communications in Fig. 7.

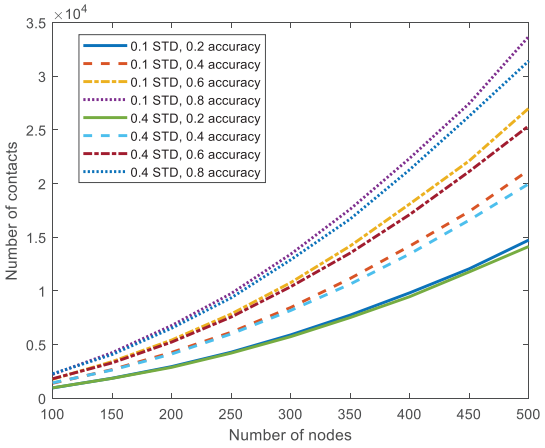


Fig. 5. The number of needed contacts vs. number of nodes

As shown in Fig. 7, although the speed STDs are different, the knowledge gain reduces to 0 almost at the same time for all both values. With the increasing number of nodes, the larger speed STD gives a slightly higher average knowledge gain for each contact during the middle period of the opportunistic communications, which is corresponding to Fig. 5.

TABLE II
FITTING THE CURVES OF KNOWLEDGE GAIN WHERE THE FITTED FUNCTION IS $a_1 \cdot \exp(-(\frac{C-b_1}{c_1})^2)$.

# of nodes	Speed STD	a_1	b_1	c_1	RMSE
500	0.1	8.866	$2.295 \cdot 10^4$	$1.643 \cdot 10^4$	0.237
500	0.4	9.683	$2.150 \cdot 10^4$	$1.496 \cdot 10^4$	0.214
400	0.1	8.595	$1.545 \cdot 10^4$	$1.082 \cdot 10^4$	0.232
400	0.4	9.099	$1.456 \cdot 10^4$	$1.020 \cdot 10^4$	0.245
300	0.1	8.080	9228	6452	0.221
300	0.4	8.460	8870	6140	0.190
200	0.1	7.049	4552	3267	0.116
200	0.4	7.374	4442	3110	0.130
100	0.1	5.305	1471	1070	0.059
100	0.4	5.185	1513	1100	0.104

We set the number of nodes to 500, 400, 300, 200 and 100 respectively, and use a Gaussian function, $a_1 \cdot \exp(-(\frac{C-b_1}{c_1})^2)$, to fit curves to the knowledge gain, where C denotes the number of contacts in the system, a_1 is the maximum knowledge gain during opportunistic communications.

The curve fitting results are shown in Table II. The RMSEs are all lower than 0.25, which demonstrates the correctness of the obtained models. We find that, except for the case of 100 nodes, the maximum knowledge gain for 0.4 speed STD is always higher than the value for 0.1 speed STD. This implies that higher speed STD is able to lead to higher maximum knowledge gain for opportunistic communications.

To fully understand the relationships among maximum knowledge gain, the number of nodes and speed STD, we try a polynomial function, a power function and a logarithmic function to model the maximum G with respect to N and speed STD. The results are illustrated in Table III. The logarithmic function gives the lowest RMSE. It has as the following

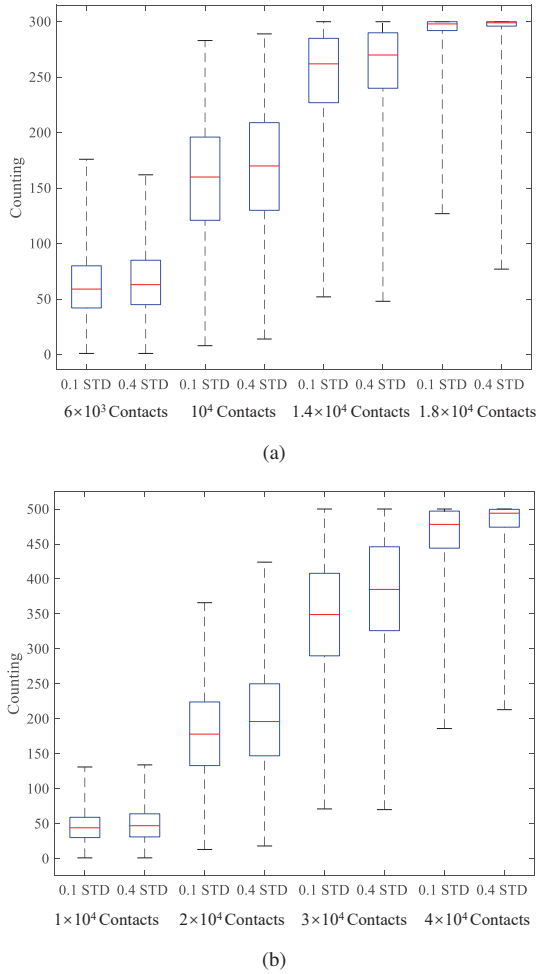


Fig. 6. The distribution of count across all nodes at specific number of contacts: (a) 300 nodes, (b) 500 nodes.

general form,

$$a_1 = \beta \cdot \log_{\alpha}(N) - \beta, \tag{4}$$

where parameter β depends on the speed STD.

TABLE III
MODELLING THE RELATIONSHIP BETWEEN MAXIMUM KNOWLEDGE GAIN (a_1) AND THE NUMBER OF NODES N UNDER DIFFERENT SPEED STDS.

Function	STD	Expression	RMSE
Polynomial	0.1	$a_1 = 0.0087 \cdot N + 4.9790$	0.5410
Power	0.1	$a_1 = 1.3420 \cdot N^{0.3087}$	0.2860
Logarithmic	0.1	$a_1 = 5.349 \cdot \lg(N) - 5.349$	0.1515
Polynomial	0.4	$a_1 = 0.0107 \cdot N + 4.744$	0.5961
Power	0.4	$a_1 = 1.03 \cdot N^{0.364}$	0.3113
Logarithmic	0.4	$a_1 = 5.645 \cdot \lg(N) - 5.645$	0.2444

Similarly, we model the functions of the number of nodes N and the parameters b_1 and c_1 under different speed STDs, respectively. The results are shown in Table IV and Table V.

Since both parameters b_1 and c_1 are highly related to the number of contacts in the system, there is no surprise that the power function has the lowest RMSE. Note that when the speed STD is fixed, the index of the power function is the same

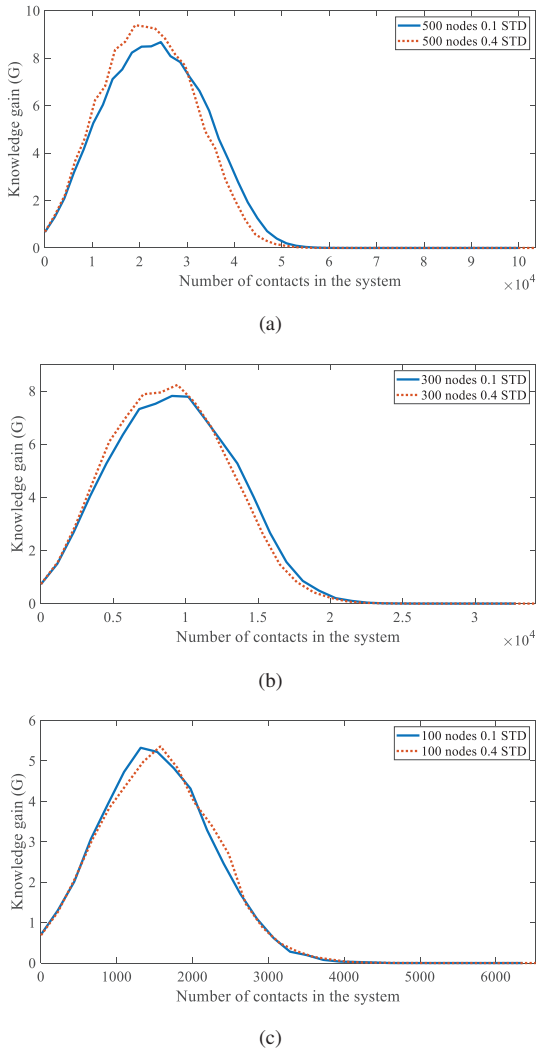


Fig. 7. Knowledge gain (G) vs. number of contacts: (a) 500 nodes, (b) 300 nodes, and (c) 100 nodes. Note the different scales of the x-axes in the three graphs and their effect on visualizing the spread of the knowledge gain distribution.

TABLE IV
MODELLING THE RELATIONSHIP BETWEEN THE PARAMETER b_1 AND THE NUMBER OF NODES N UNDER DIFFERENT SPEED STDS.

Function	STD	Expression	RMSE
Polynomial	0.1	$b_1 = 53.86 \cdot N - 5427$	1603
Power	0.1	$b_1 = 0.3945 \cdot N^{1.7652}$	77.93
Exponential	0.1	$b_1 = 1864 \cdot \exp(5.078 \cdot 10^{-3} \cdot N)$	1346
Polynomial	0.4	$b_1 = 50.09 \cdot N - 4851$	1433
Power	0.4	$b_1 = 0.4895 \cdot N^{1.72}$	107.4
Exponential	0.4	$b_1 = 1841 \cdot \exp(4.971 \cdot 10^{-3} \cdot N)$	1246

for b_1 and c_1, which infers that the index merely depends on speed STD.

To sum up, when speed STD is 0.1, the model of knowledge gain (G) is,

$$G = (5.349 \lg(N) - 5.349) \exp\left(-\left(\frac{C - 0.3945 \cdot N^{1.7652}}{0.2801 \cdot N^{1.7652}}\right)^2\right). \tag{5}$$

TABLE V
MODELLING THE RELATIONSHIP BETWEEN THE PARAMETER c_1 AND THE NUMBER OF NODES N UNDER DIFFERENT SPEED STDS.

Function	Speed STD	Expression	RMSE
Polynomial	0.1	$c_1 = 38.27 \cdot N - 3874$	1237
Power	0.1	$c_1 = 0.2801 \cdot N^{1.7652}$	148.2
Exponential	0.1	$c_1 = 1286 \cdot \exp(5.143 \cdot 10^{-3} \cdot N)$	856.4
Polynomial	0.4	$c_1 = 34.81 \cdot N - 3341$	1010
Power	0.4	$c_1 = 0.3408 \cdot N^{1.72}$	88.86
Exponential	0.4	$c_1 = 1293 \cdot \exp(4.954 \cdot 10^{-3} \cdot N)$	861.1

When speed STD is 0.4, the model of knowledge gain (G) is,

$$G = (5.645 \lg(N) - 5.645) \exp\left(-\left(\frac{C - 0.4895 \cdot N^{1.72}}{0.3408 \cdot N^{1.72}}\right)^2\right), \quad (6)$$

where N is the number of nodes and C represents the number of contacts in the system.

To verify the correctness of our obtained models, we set the number of nodes as 450 and 350 respectively and depict the true values and modeled values in Fig. 8, where the circle marks are the true values, and the dash lines represent our obtained model. The modeled values are very close to the true values.

V. DISCUSSION AND CONCLUSION

In this paper, we explore a distributed computational task for counting the number of nodes that are constrained in an area. We do so by exchanging data opportunistically via device-to-device communications and study how the counting is affected by the number and the mobility of nodes. We have made an exploratory study based on simulation for a closed system to better understand the interaction between the opportunistic communication and the distributed computation of the nodes.

During the modeling, we find that the difference in mobility, i.e., different speed STDs can affect the accuracy of the count during the middle period of the distributed computation, but it does not affect the convergence time. Hence, we further investigate how the convergence of the counts in the nodes progress towards a unanimous value and use the knowledge gain of contacts to model the processes of data collection. According to the derived model, we find that the gain follows a Gaussian distribution over time and a higher speed STD will lead to a slightly higher maximum gain for opportunistic communication, which reflects that the computation is related to the mobility as well.

We now return to the questions we posed in the introduction.

- How does the computation time depend on the mobility and number of nodes? *We have derived an expression for the computation time to a given accuracy. It increases linearly with accuracy while decreases as a power law with the number of nodes.*
- How does the counts in the nodes converge towards a unanimous value? *We use the knowledge gain of the contacts to study the processes of data collection. The*

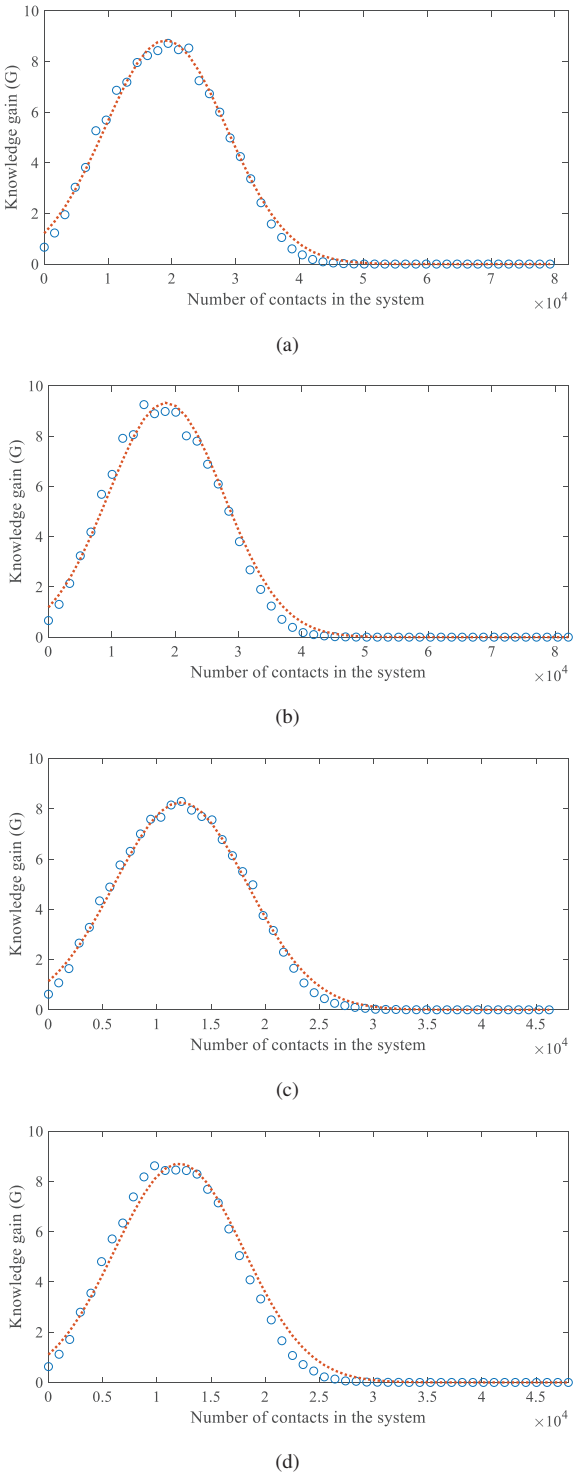


Fig. 8. Verification for the knowledge gain model where circle marks are true values and the dash lines represent the modeling values: (a) 450 nodes 0.1 speed STD, (b) 450 nodes 0.4 speed STD, (c) 350 nodes 0.1 speed STD, and (d) 350 nodes 0.4 speed STD.

convergence progresses according to a Gaussian distribution over time (i.e., contacts). The maximum knowledge gain is proportional to the logarithm of the number of nodes.

- Can the computation be modeled independently of the mobility? *In the underlying communication, there is not any dependence of the contact rate on the standard deviation, σ, of the speed distribution. However, for the application, there is a dependence of the maximum knowledge gain on σ. This is seen in the two mean values of the distribution for STD 0.1 and 0.4, respectively. It does not appear to affect the ultimate convergence time.*

Given this understanding, we will model the process analytically. We especially will study the distribution of the count over the collection of nodes at a given time point. This is now shown in the box-plots in Fig. 6. It relates to the count an outside observer would get if a randomly selected node would be chosen to report on how many nodes reside in the area. It is different than the accuracy we have used here, which is the proportion of nodes that have converged to the true count. Ultimately, we would like to return to our previous work in [3] with a model that captures the dynamics in node population for an open system with node arrivals and departures.

The studied distributed process is contact limited since the computation occurs only when new information is provided in a node (and for counting, the processing is light). Other distributed opportunistic computations might have heavier computation and be less dependent on the exchange of intermediate results with other nodes in the system. We remark that our findings might not be applicable in such a case.

We believe (and will further study) that our results bring insights also to other tasks than counting. The basic process we have laid out consists of collecting data, performing a computation on the data, and sharing the result of the computation. Data in our case are node IDs, but could have been other systems parameters that should collectively be estimated, or the data could be environmental parameters for the space that the nodes jointly sense, collect and compute. This indicates broader applicability of opportunistic computing.

ACKNOWLEDGMENT

KTH gratefully acknowledges support from Peter Wallenberg Foundation (grant no. 2017.0001). HKUST gratefully acknowledges support from projects 26211515, 16214817, and G-HKUST604/16 from the Research Grants Council of Hong Kong, and an HKUST internal grant FP805 for supporting collaboration with KTH.

REFERENCES

[1] T. Li, Z. Xiao, H. M. Georges, Z. Luo, and D. Wang, "Performance analysis of co-and cross-tier device-to-device communication underlaying macro-small cell wireless networks." *KSII Transactions on Internet & Information Systems*, vol. 10, no. 4, 2016.

[2] O. Helgason, S. T. Kouyoumdjieva, L. Pajevic, E. A. Yavuz, and G. Karlsson, "A middleware for opportunistic content distribution," *Computer Networks*, vol. 107, pp. 178 – 193, 2016, mobile Wireless Networks.

[3] P. Danielis, S. T. Kouyoumdjieva, and G. Karlsson, "Urbancount: Mobile crowd counting in urban environments," in *2017 8th IEEE Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON)*, Oct 2017, pp. 640–648.

[4] S. Wang, X. Wang, J. Huang, R. Bie, Z. Tian, and F. Zhao, "The potential of mobile opportunistic networks for data disseminations," *IEEE Transactions on Vehicular Technology*, vol. 65, no. 2, pp. 912–922, Feb 2016.

[5] Z. Vatandas, S. M. Hamm, K. Kuladinithi, U. Killat, A. Timm-Giel, and A. Frster, "Modeling of data dissemination in oppnets," in *2018 IEEE 19th International Symposium on "A World of Wireless, Mobile and Multimedia Networks" (WoWMoM)*, June 2018, pp. 01–04.

[6] A. Passarella and M. Conti, "Characterising aggregate inter-contact times in heterogeneous opportunistic networks," in *International Conference on Research in Networking*. Springer, 2011, pp. 301–313.

[7] E. Hernández-Orallo, M. Murillo-Arcila, J. Cano, C. T. Calafate, J. A. Conejero, and P. Manzoni, "An analytical model based on population processes to characterize data dissemination in 5g opportunistic networks," *IEEE Access*, vol. 6, pp. 1603–1615, 2018.

[8] S. T. Kouyoumdjieva and G. Karlsson, "Energy-aware opportunistic mobile data offloading under full and limited cooperation," *Comput. Commun.*, vol. 84, no. C, pp. 84–95, Jun. 2016.

[9] M. Conti, S. Giordano, M. May, and A. Passarella, "From opportunistic networks to opportunistic computing," *IEEE Communications Magazine*, vol. 48, no. 9, pp. 126–139, Sept 2010.

[10] S. Boyd, A. Ghosh, B. Prabhakar, and D. Shah, "Randomized gossip algorithms," *IEEE Transactions on Information Theory*, vol. 52, no. 6, pp. 2508–2530, June 2006.

[11] M. Jelasity, A. Montresor, and O. Babaoglu, "Gossip-based aggregation in large dynamic networks," *ACM Trans. Comput. Syst.*, vol. 23, no. 3, pp. 219–252, Aug. 2005.

[12] Z. Shi and P. K. Srimani, "An online distributed gossiping protocol for mobile networks," *Journal of Combinatorial Optimization*, vol. 11, no. 1, pp. 87–97, Feb. 2006.

[13] O. Helgason, S. T. Kouyoumdjieva, and G. Karlsson, "Opportunistic communication and human mobility," *IEEE Transactions on Mobile Computing*, vol. 13, no. 7, pp. 1597–1610, July 2014.

[14] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli, "Fog computing and its role in the internet of things," in *Proceedings of the First Edition of the MCC Workshop on Mobile Cloud Computing*, ser. MCC '12. New York, NY, USA: ACM, 2012, pp. 13–16.

[15] E. D. Pascale, I. Macaluso, A. Nag, M. Kelly, and L. Doyle, "The network as a computer: A framework for distributed computing over iot mesh networks," *IEEE Internet of Things Journal*, vol. 5, no. 3, pp. 2107–2119, June 2018.

[16] R. K. Ganti, F. Ye, and H. Lei, "Mobile crowdsensing: current state and future challenges," *IEEE Communications Magazine*, vol. 49, no. 11, pp. 32–39, November 2011.

[17] H. Ma, D. Zhao, and P. Yuan, "Opportunities in mobile crowd sensing," *IEEE Communications Magazine*, vol. 52, no. 8, pp. 29–35, Aug 2014.

[18] C. Jiang, L. Gao, L. Duan, and J. Huang, "Scalable mobile crowdsensing via peer-to-peer data sharing," *IEEE Transactions on Mobile Computing*, vol. 17, no. 4, pp. 898–912, April 2018.

[19] S. T. Kouyoumdjieva, P. Danielis, and G. Karlsson, "Survey of non-image based approaches for counting people," *IEEE Communications Surveys Tutorials*, 2019.

[20] J. Chen and T. Cook, "Using d-gap patterns for index compression," in *Proceedings of the 16th international conference on World Wide Web*. ACM, 2007, pp. 1209–1210.

[21] M. McGuire, "Stationary distributions of random walk mobility models for wireless ad hoc networks," in *Proc. of the 6th ACM International Symposium on Mobile Ad Hoc Networking and Computing*, ser. MobiHoc '05. New York, NY, USA: ACM, 2005, pp. 90–98.

[22] O. R. Helgason and K. V. Jónsson, "Opportunistic networking in omnet++," in *Proc. of the 1st International Conference on Simulation Tools and Techniques for Communications, Networks and Systems & Workshops*, ser. Simutools '08, 2008, pp. 82:1–82:8.