ISSN 2278-3091

International Journal of Advanced Trends in Computer Science and Engineering, Vol.2 , No.1, Pages : 75-80 (2013)
Special Issue of ICACSE 2013 - Held on 7-8 January, 2013 in Lords Institute of Engineering and Technology, Hyderabad

A novel approach on Applications, Research Challenges and Mining for Mobile Crowdsensing

Sumayya Afreen[1], Khutaija Abid[2] , Dr V S Giridhar Akula[3]
[1]M tech Student, JNTU, India, sumayyaafreen21@yahoo.com
[2]M Tech Student, JNTU, India,khutaija.abid@gmail.com
[3]Principal, LIET, India, akulagiri2002@yahoo.com

Abstract - A new phase of the information society – The Internet of things in which the web will not only link computers but potentially every object created by mankind. An emerging category of devices at the edge of the Internet are consumer centric mobile sensing and computing devices, such as smart phones, music players, and in-vehicle sensing devices. These devices will fuel the evolution of the Internet of Things as they feed sensor data to the Internet at a societal scale. In this paper, we will examine a category of applications that we term *mobile crowdsensing*, where individuals with sensing and computing devices collectively share information to measure and map phenomena of common interest. . We have implemented our Context-Aware Real-time Open Mobile Miner (CAROMM) to facilitate data collection from mobile users for crowdsensing applications. We will present a brief overview of mobile crowdsensing, its applications, research challenges, and evaluate CARROM frame work.

Key words: cloud, cluster, crowdsensing, sensors.

1. INTRODUCTION

The integration of sensing and embedded everyday computing devices at the edge of the Internet will result in the evolution of an *embedded Internet* or the *Internet of Things*.

An emerging category of edge devices that are fueling this evolution are consumer centric mobile sensing and computing devices. These include devices such as smart phones (iPhone, Google Nexus), music players (iPods), sensor embedded gaming systems (Wii, XboX Kinect), and in-vehicle sensing devices (GPS, OBD-II). They have become extremely popular recently and are potentially important sources of sensor data. They are typically equipped with various sensing facility and wireless capabilities and are connected to the Internet. As an example, a sample list of mobile devices and their corresponding sensing capabilities are provided in Table I.

We observe from Table I that these mobile devices can be used to measure various individual and *community* phenomena. Individual phenomena are those pertaining to a particular device owner, such as movement patterns (e.g. running, walking, climbing stairs), modes of transportation (e.g. biking, driving, taking a bus, riding the subway), and activities (e.g. using an ATM, visiting a specific store, having a conversation, listening to music , and making coffee). Community phenomena are those pertaining to the aggregate of surroundings and not limited to particular individuals. These include pollution (air/noise) levels in a neighborhood, real- time traffic patterns, and pot holes on roads, road closures, and transit timings. Such large scale, community phenomena monitoring is possible when a community of individuals share the sensor data they collect towards a common goal, usually with certain processing involved.

Type of community sensing is popularly called *participatory* sensing or *opportunistic* sensing. Participatory sensing is defined as the kind of sensing where individuals are actively involved in contributing sensor data (e.g. taking a picture, reporting a road closure). On the other hand, opportunistic sensing is where the sensing is more autonomous and user involvement is minimal (e.g. continuous location sampling). We consider that, in terms of the level of user involvement, community sensing spans a wide spectrum, with participatory sensing and opportunistic sensing at the two ends of the spectrum. We therefore coin the term *mobile crowdsensing (MCS) to* refer to a broad range of community sensing paradigms. Fig 1 shows us the different outputs of sensor apps used on phone which include temperature, humidity, pressure, carbon monoxide, etc. The rest of this paper is organized as follows. Section 2 discusses mobile crowdsensing applications. Section 3 discusses Research challenges. Section 4 explains on the move mining for mobile crowdsensing which explains CAROMM frame work.

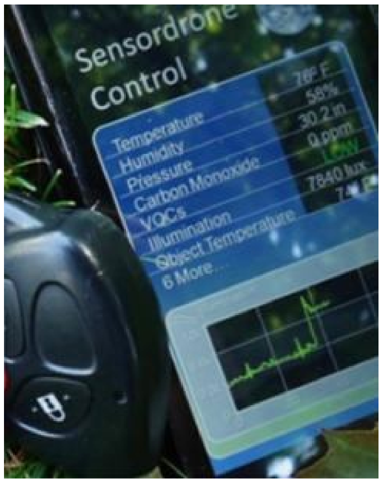

Fig 1: Mobile showing output of different sensors

Table I: Sensors on various mobile sensing devices

Device	Inertial	Compass	GPS	Microphone	Camera	Proximity	Light
iPhone	✓	✓	✓	✓	✓	✓	✓
Nexus S	✓	✓	✓	✓	✓	✓	✓
Nokia 6210	✓	✓	✓	✓	✓		✓
iPod Touch	✓			✓	✓		✓
Garmin ForeRunner 410	✓	✓	✓				

2. MOBILE CROWDSENSING APPLICATIONS

In this section, we will briefly discuss existing mobile crowdsensing applications, we classify MCS applications into three different categories based on the type of phenomenon being measured or mapped as discussed in MCS: Current state and future challenges [1]. These include (i) *Environmental*, (ii) *Infrastructure*, and (iii) *Social Applications.*

In environmental MCS applications, the phenomena being measured are those of the natural environment. Examples of such applications include measuring pollution levels in a city, water levels in creeks, and monitoring wildlife habitats. Such applications enable the mapping of various large scale environmental phenomena by involving the common man. An example prototype deployment for pollution monitoring is Common Sense [2]. Common Sense uses specialized handheld air quality sensing devices that communicate with mobile phones (using Bluetooth) to measure various air pollutants (e.g. CO_2, NO_x). Another example is *Creek Watch* developed by IBM Almaden Research Center. It monitors water levels and quality in creeks by aggregating reports from individuals, such as pictures taken at various locations along the creek, or text messages about the amount of trash.

Another important category is *infrastructure* applications that involve the measurement of large scale phenomena related to public infrastructure. Examples include measuring traffic congestion, road conditions, parking availability, outages of public works (e.g. malfunctioning fire hydrants, broken traffic lights), and real-time transit tracking. Early MCS deployments measured traffic congestion levels in cities, examples of which include MIT's CarTel [3]. CarTel utilizes specialized devices installed in cars to measure the locations and speeds of the cars and transmit the measured values using public Wi-Fi hotspots to a central server. Another MCS example is ParkNet [4], an application that detects available parking spots in cities using ultrasonic sensing devices installed on cars combined with smart phones.

Finally, another category is *social* applications. As an example, individuals can share their exercise data (e.g. how much time one exercises in a day) and compare their exercise levels with the rest of the community. They can use this comparison to help improve their daily exercise routines. Example deployments include BikeNet [5]. In BikeNet, individual's measure location and bike route quality (e.g. CO_2 content on route, bumpiness of ride) and aggregate the data to obtain "most" bike able routes.

3. RESEARCH CHALLENGES

Localized Analytics

Fig 2 depicts Research challenges as functional components. Mobile devices are equipped with various kinds of sensors such as GPS, accelerometer, microphone and camera. The OS allows applications to access the sensors and extract raw sensing data from them. However, depending on the nature of the raw data and the needs of applications, the physical readings from sensors may not be suitable for the direct consumption of applications. Many times, some *local analytics* perform certain primitive processing on the raw data on the device. They produce some intermediate results which are sent to the backend for further processing and consumption. The main challenge in local analytics is finding heuristics and designing algorithms to achieve the desired function.

Resource Limitations: Energy, Bandwidth, and Computation

Resource constraints in traditional sensor networks have been well studied. However, MCS applications introduce many new aspects for this challenge

First, the set of devices that are collecting sensor data are highly dynamic in availability and capabilities. Due to this highly dynamic nature, modeling and predicting the energy, bandwidth requirements to accomplish a particular task becomes much more difficult than in traditional sensor networks. Second, when there are a large number of available devices with diverse sensing capabilities, identifying and scheduling sensing and communication jobs among them under resource constraints becomes more difficult as well.

Different types of data can be used for the same purpose, but with different quality and resource consumption trade- offs. How to leverage their differences to improve the quality while minimizing resource consumption is a new problem. For example, location data can be provided using GPS, WiFi, and GSM, with decreasing levels of accuracy. Compared to WiFi and GSM, continuous GPS location sampling drains the battery much more quickly.

The existence of multiple concurrent applications that require data of different types also complicates resource allocation. A mobile device can be sampling various sensors (e.g. GPS, accelerometer, air quality) on behalf of different applications.

ISSN 2278-3091

International Journal of Advanced Trends in Computer Science and Engineering, Vol.2 , No.1, Pages : 75-80 (2013)
Special Issue of ICACSE 2013 - Held on 7-8 January, 2013 in Lords Institute of Engineering and Technology, Hyderabad

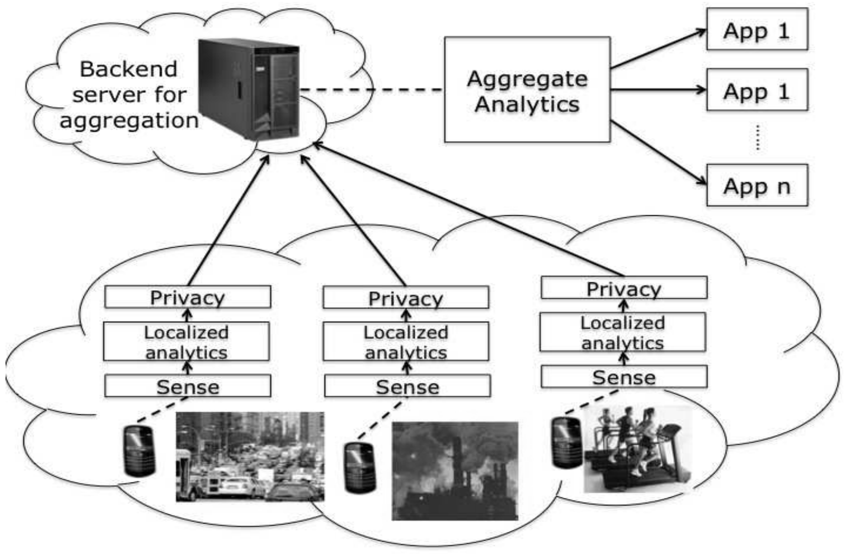

Fig 2: Typical functioning of MCS applications. Raw sensing data is collected on devices and local analytics process it to produce consumable data for applications. After privacy preservation, the data is sent to the backend and aggregate analytics will further process it for different applications.

Privacy

An important aspect of MCS applications is that they potentially collect sensitive sensor data pertaining to individuals. For example, GPS sensor readings can be utilized to infer private information about the individual, such as the routes they take in daily commutes, their home and work locations, and so on. On the other hand, these GPS sensor measurements (from daily commutes) shared within a larger community can be used to obtain traffic congestion levels in a given city. Thus, it is important to preserve the privacy of an individual, but at the same time enable MCS applications.

Aggregate Analytics

The local analytics running on mobile devices give data about the local area to applications. For many applications, they may also need to run *aggregate analytics* at the backend. These analytics detect pat- terns in the sensor data from large number of mobile devices. These patterns occur in certain spatial scope and within some temporal duration; they signify the features and characteristics of the physical or social environment that are interested by the user.

For example in public works maintenance. Citizens can report problems in public facilities, such as broken water pipes, dysfunctional traffic lights. By looking at the counts of complaints, the maintenance personnel can infer to a certain degree the scope and severity of the incident, and use that information to help prioritize and schedule their repair resources.

4. CONTEXT-AWARE REAL-TIME OPEN MOBILE MINER (CAROMM)

Personal sensing combined with social networks has given rise to 'mobile social networks' where sensed user context information is shared with the user's social network. To the best of our knowledge, using social networks/media themselves as a source of information for community sensing is an emerging focus in this area. To this regard, we propose a framework for mobile crowdsensing, Context- Aware Real- time Open Mobile Miner (CAROMM) used in On the move mining [6] , to facilitate sensor data collection from mobile users and correlate this real- time information with social media data from both Twitter (http://twitter.com/) and Facebook (http://facebook.com).

An integral part of a mobile crowdsensing framework such as CAROMM is sensing and sending of information. There are several key factors that need to be considered and addressed in order for mobile crowdsensing to be effective. Firstly, it is imperative that the *data collection* process from mobile device is cost- efficient for both the device performing the sensing, as well as the networks that need to scale for large volumes of users sending sensed data. Secondly, mobile crowdsensing needs to have infrastructure to receive, manage and analyze large volumes of real-time data streams using the pay-per-use cloud computing platforms. Thirdly, sensing using mobile devices requires participation from the user and willingness to allow collection of sensor data, and hence use preferences and privacy-preserving operations for mobile crowdsensing need to considered. In this context, there need to be incentives in place to facilitate such large-scale mobile crowdsensing. For example, an incentive of reduced data transfer costs for supporting citizen surveillance operations and so on need to be considered. The focus of this section is on the data collection dimension as it is the first step for mobile crowdsensing. We propose and develop a mobile data mining driven approach for highly scalable and cost-efficient data collection for mobile crowdsensing. We implement our proposed CAROMM system to leverage cloud technologies to enable collation and

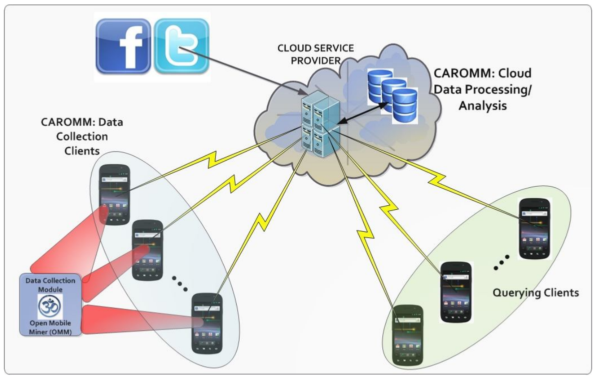

Fig 3 : The CAROMM Framework

processing of huge amounts of real-time data generated by mobile phone sensors, as well as correlation of information feeds from social media to specifically support real-time queries pertaining to locations of interest. The CAROMM system forms the basis for evaluating the feasibility and validity of mobile analytics as an effective mechanism for supporting large-scale mobile crowdsensing.

Caromm Framework

Nowadays, everyone carries a mobile device with them. Most of these mobile devices come with increasingly sophisticated list of sensors that are able to capture various context information pertaining to the user and their environment. Prevalence and wide uptake of social media such as Facebook (http://www.facebook.com/) and Twitter (http://twitter.com/) and photo sharing sites such as Flickr (http://www.flickr.com/) have shown that users are willing to share information. Mobile crowdsensing aims to leverage these phenomena with a view to delivering real-time sensory information to a range of applications such as location-based service delivery; location- based social networking and citizen surveillance.

We propose and develop our CAROMM system to support the scenario where people use a mobile application to upload sensory data to the cloud. The sensory data could have mixed input such as multimedia/videos, twitter/social media streams, text, activities, location, temperature, time, device orientation, speed, and movement of the device. An application on the cloud processes this mixed data to operate as a real-time location information service. Thus, CAROMM supports the provision of collated information to users who request for real-time information about specific locations of interest. The real-time information delivered includes aggregation of sensor feeds from mobile users in that location pertaining to physical phenomena such as light levels, temperature, estimates of crowd intensity, as well videos/photos (and the context in which those photos were taken such as day, night), and recent social media posts.

In this section, we present an overview of the Context Aware Real-time Open Mobile Miner (CAROMM) framework for enabling mobile crowdsensing applications. CAROMM has several features: (i) capture different types of stream data from mobile devices, (ii) process, manage and analyze this data along with the relevant contextual information associated with them (e.g. associate light-intensity levels with pictures/videos, and social media information pertaining to locations of interest), and (iii) facilitate real-time queries from mobile users on the collected (and analyzed) data.

Fig 3 shows the Context Aware Real-time Open Mobile Miner (CAROMM) framework. The framework consists of three main modules- a Data Collection Client and a Querying Client residing on the mobile devices, and a Data Processing Module residing on the cloud. The Data collection Module captures sensory data, performs local continuous real-time stream mining on the data and uploads analyzed information to the Data Processing Module in the cloud where further analysis, management, and fusion of the incoming multiple streams needs to be performed. To intelligently send only analyzed information from each device, we use resource-aware clustering on the sensory data to identify significant changes in the situation. The Querying Client on mobile devices sends user queries to the Data Processing Module and receives and display the results obtained. The Data Processing Module consists of Social Media Data Collection and Query Processing. This module aggregates information obtained from all sources (mobile devices and social media) to provide contextual information in response to the user queries obtained from the Querying Client. The focus of this section is on leveraging mobile data mining for mobile crowdsensing.

Caromm Data Collection Module

A. Data Collection Module Architecture

Here we describe the architecture of the data collection module within the CAROMM framework. The data collection module of CAROMM addresses the challenges in collecting, processing/analyzing and uploading data sensed from user environments. Fig 4 depicts the architecture of Data collection module. The data collection module has five main components, namely, Interface Controller, Data Analysis-Cluster engine, Data Collection Manager, Cloud Upload Manager, and Sensor and Media Manager. The data collection module runs on the mobile device interfacing with sensors available on the device. The proposed approach does not require any additional hardware sensors other than sensors available on the mobile device. *Interface Controller*: This component is the graphical user component presented to the user. *Data Analysis-Cluster Engine*: The data analysis-cluster engine is the core component of the

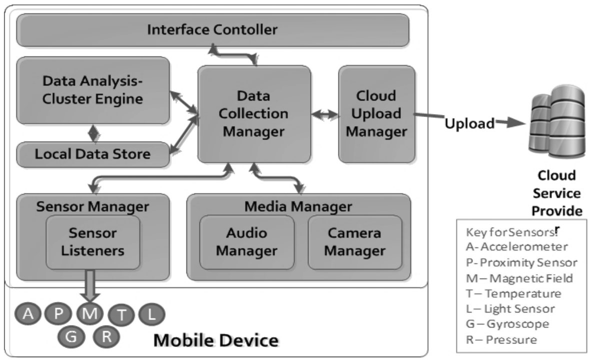

Fig 4: CAROMM-Data Collection Module Architecture

proposed data collection module. It handles all processing and analysis of data. The analysis engine performs continual mining over the sensed data. For continual data mining, we use the generic open source toolkit for mobile data mining (OMM) [7]. We have used the Light Weight Cluster (LWC) algorithm as shown in Fig 5, implemented in OMM toolkit to perform clustering over sensed data.

Data Collection Manager: The data collection manager acts as a coordinator between the components of the data collection module. It instantiates the various sensors on the phone using the sensor manager. It forwards sensed data to the analysis engine for on-the-move mining and clustered datasets from the analysis engine to the cloud manager for further processing. *Cloud Storage Manager*: The cloud storage manager is responsible for uploading data to the cloud using the mobile device's network connection. *Sensor and Media Manager*: The sensor manager is responsible for interfacing with the device's sensors. It periodically queries the device's sensors for data and passes it to the data collection manager. The algorithms implemented in the proposed data collection module are presented in pseudo code format Fig 6.

```
   Begin LWC
2  Data items arrive in sequence with a data rate.
3  The algorithm starts by considering the first point as a center.
4  Compare any new data item with the centers to find the distance.
5  If the distance for all the centers is greater than a threshold, the
   new item is considered as a new center;
   else increase the weight for the center that has the shortest
   distance between the data item and the center by 1 and let the
   new center equals the weighted average.
6  Repeat 4 and 5.
7  If the number of centers = k (according to the available memory)
   then create a new centers    vector.
8  Repeat 4, 5, 6, and 7.
9  If memory is full then re-cluster (incrementally integrate the
   clusters)
   End
```

Fig 5: LWC Algorithm [8]

```
  Begin dataCollection
2 Instantiate sensors
3 Repeat 4,5,6,7 every x seconds
4       sense data
5       dataAnalysis(SensedData)
6       if data to upload:
7               upload SensedData to Cloud
  End
------------------------------------------------------------------
  Begin dataAnalysis(SensedData)
2    if no cluster exist:
3       create new cluster for SensedData
4    else:
5       changeDetect(SensedData)
6       if new cluster created
7               return SensedData
     End
------------------------------------------------------------------
   Begin changeDetect(SensedData)
2  Iterate through existing list of cluster
3  For each cluster get their attributes
4  Compute distance between SensedData and existing clusters
5  If distance < threshold:
6    //member of existing cluster
7        Get the closest cluster for the SensedData
8    Increase the weight of the existing cluster
9    Return closest cluster
10 else:
11   // new cluster
12   if current number of clusters < maximum cluster size
13     Create a new cluster for the SensedData with attributes
14   else:
15     Re-cluster (incrementally integrate the clusters)
   End
```

Fig 6: Data Collection Module Algorithms

B. *Cost Models for Mobile Data Stream Mining*

Here we develop cost models for two data collection approaches for mobile crowdsensing:

1) Model 1: all processing in the cloud: In this mode, the mobile devices sense context data periodically and upload to the cloud. No processing is done on the device.

2) Model 2: local mobile data analytics on-board the device: In this mode, each mobile device performs continuous sensing and local data stream mining on the collected sensor data and only mined data is uploaded to the cloud. This aims to reduce costs related to energy usage and data transmission.

We develop two cost models: a data transmission cost model and an energy usage cost model. These cost models aim to compare the cost related to the above two data collection approaches. Therefore, in these models we do not consider the cost associated with mining social media data. Social media data is only mined on the cloud.

Data Transmission Cost Model: We evaluate the cost of Data transmission in terms of consumed bandwidth for any time period *t*. The cost of data transfer is directly proportional to the amount of data transferred between the mobile device (M) and the cloud (C).

$$Cost_{dt} \propto \textit{total data transferred from M to C} \qquad (1)$$

Therefore, the data transmission cost $Cost_{dt}$ can be represented as follows:

$$Cost_{dt} = x * \textit{total data transferred from M to C} \quad (2)$$

where x is a constant.

- Model 1: All processing in the cloud: amount of data transferred *total data transferred from M to C* is high as all sensed data is uploaded.
- Model 2: Local processing on the mobile should result in lower *total data transferred from M to C*, and hence, lower data transmission cost.

Let, $Cost_{dtraw}$ be the data transmission cost for Model 1 (i.e., using raw data), and $Cost_{dtclust}$ be the data transmission cost for Model 2, (i.e., using on-device clustering). Then, assuming that for similar devices the constant x is same in the case of both raw and clustering approaches, the savings on the data transmission cost for mobile data mining can be evaluated as

$$\textit{Bandwidth Gain} = \frac{Cost_{dtraw}}{Cost_{dtclust}}$$

$$= \frac{(\textit{total data transferred from M to C})^{raw}}{(\textit{total data transferred from M to C})^{clust}} \qquad (3)$$

Energy Usage Cost Model: We model the cost of energy usage in terms of battery drain for any time period *t*. Cost incurred due to energy drain is composed of drain due to sensing, drain due to processing/mining in the device and drain due to data transfer.

$$Cost_{ed} = Cost_{edS} + Cost_{edPr} + Cost_{edDt} \quad (4)$$

\

$Cost_{edS}$ represents energy drain due to sensing. It is directly proportional to the frequency of sensing. Energy expended due to sensing is the same in both modes of operation and can therefore be discounted.

$$Cost_{edS} \propto freq.\ of\ sensing \quad (5)$$

$$Cost_{edS} = a * freq.\ of\ sensing \quad (6)$$

where a is a constant.
Using the same bandwidth, larger amount of data transfer requires more time and hence results in more energy drain. Similarly, more frequent data transfer requires more energy. These relationships can be expressed as follows:

$$Cost_{edDt} \propto total\ data\ transferred\ from\ M\ to\ C \quad (7)$$

$$Cost_{edDt} \propto num.\ of\ data\ transfers\ from\ M\ to\ C \quad (8)$$

$$Cost_{edDt} = y * total\ data\ transferred\ from\ M\ to\ C * num.\ of\ data\ transfers\ from\ M\ to\ C \quad (9)$$

where y is a constant. From 4, 6 and 9, the energy drain cost can be represented as follows:

$$Cost_{ed} = a * freq.\ of\ sensing + Cost_{edPr} + y * total\ data\ transferred\ from\ M\ to\ C * num.\ of\ data\ transfers\ from\ M\ to\ C \quad (10)$$

- Model 1: All processing on the cloud: In this mode, there is no processing on the mobile, therefore, $Cost_{edPr} * 0$. Therefore, the energy drain cost consists of only the data transfer cost and the sensing cost. In this instance, the cost can be represented as:

$$Cost\text{-}raw_{ed} = a * freq.\ of\ sensing + y * total\ data\ transferred\ from\ M\ to\ C * num.\ of\ data\ transfers\ from\ M\ to\ C \quad (11)$$

- Model 2: Local processing on the mobile. In this case, energy drain cost due to mobile data mining $Cost_{edPr}$ becomes significant. Typically, energy drain due to processing is directly proportional to the amount of data being processed. Energy drain may also be affected by the clustering algorithm used, however, we do not consider this in these cost models.

$$Cost\text{-}clust_{edPr} \propto total\ size\ of\ data\ accumulated\ on\ M \quad (12)$$

$$Cost\text{-}clust_{edPr} = z * total\ size\ of\ data\ accumulated\ on\ M \quad (13)$$

where z is a constant. From 10 and 13, the energy drain cost for clustering can be computed as:

$$Cost\text{-}clust_{ed} = a * freq.\ of\ sensing + z * total\ size\ of\ data\ accumulated\ on\ M + y * total\ data\ transferred\ from\ M\ to\ C * num.\ of\ data\ transfers\ from\ M\ to\ C \quad (14)$$

The *total size of data accumulated on M* depends on the *freq. of sensing and* caching mechanisms used. This variable may not vary much in the two models. Similarly, for fair comparison, *freq. of sensing* would be the same in the two models. Therefore, significant reduction of *total size of data accumulated on M and num. of data transfers from M to C* in any given time period t will result in reduction of energy drain cost. Performing mobile data mining aims to reduce these variables.

For energy usage evaluation of the CAROMM data collection model, we use the ratio of energy usage cost for raw approach versus the clustering approach. This is evaluated as:

$$Energy\ Gain = \frac{Cost\text{-}raw_{ed}}{Cost\text{-}clust_{ed}} \quad (15)$$

In addition to the cost models, we also develop data accuracy model for evaluating CAROMM data collection model. Let sf be the sensing frequency and uf be the data upload frequency. With raw approach, all sensed data are uploaded, i.e., $sf = uf$. If sf is high enough, it results in high accuracy as sensing and upload of data are real-time. However, this results in higher data transmission cost $Cost\text{-}raw_{dt}$ and higher energy usage cost Cost-raw_{ed}. For the same sf, the role of mobile data mining is to reduce the upload frequency uf such that there is no significant reduction in accuracy. In the raw approach, if uf is reduced significantly, it will decrease $Cost_{dt}$ and $Cost_{ed}$, but it may also reduce the data accuracy as the uploaded data is no longer current. This is especially applicable in cases of frequent changes in sensed data. With the use of clustering, the uf is affected only by changes in the sensed values, and therefore, may result in higher accuracy as all major changes are detected and reflected.

REFRENCES

[1] R. Ganti, F. Ye, and H. Lei, “Mobile crowdsensing: Current state and future challenges,” *IEEE Communications Magazine*, vol. 49, no. 11, pp. 32–39, 2011.

[2] P. Dutta *et al.*, “Demo abstract: Common sense: Participa- tory urban sensing using a network of handheld air quality monitors,” in *Proc. of ACM SenSys*, 2009, pp. 349–350.

[3] B. Hull *et al.*, “Cartel: a distributed mobile sensor comput- ing system,” in *Proc. of SenSys*, 2006, pp. 125–138.

[4] S. Mathur *et al.*, “Parknet: Drive-by sensing of road-side parking statistics,” in *Proc. of ACM MobiSys*, 2010, pp. 123-136.

[5] S. B. Eisenman *et al.*, “The bikenet mobile sensing system for cyclist experience mapping,” in *Proc. of SenSys*, November 2007.

[6] P. Jayaraman, S Krishnaswamy and A Zaslavsky, “Using On-the-move Mining for Mobile crowdsensing,” *IEEE 13th International conference on Mobile data Management*, pp. 115–124, 2012.

[7] S. Krishnaswamy, M. Gaber, M. Harbach, C. Hugues, A. Sinha, B. Gillick, P. Haghighi, and A. Zaslavsky, “Open mobile miner: a toolkit for mobile data stream mining,” in *Proceedings of ACM KDD?09*, 2009, pp. 109–114.