Energy Efficient and Accuracy Aware (E2A2) Location Services via Crowdsourcing

Yun Huang, Anthony Tomasic, Yufei An, Charles Garrod and Aaron Steinfeld
School of Computer Science
Carnegie Mellon University
Email: yunhuang@cs.cmu.edu, tomasic@andrew.cmu.edu, yufeia@andrew.cmu.edu, charlie@cs.cmu.edu, steinfeld@cmu.edu

***Abstract*—Many mobile applications rely on location information gained from location services on mobile devices. However, continuously tracking the device location with high accuracy drains the battery quickly. Furthermore, sensing the same location can be redundant when multiple devices are co-located. In this paper, we develop a crowdsourcing-based location service, E2A2 (energy efficient and accuracy aware), which places colocated devices into groups, and uses group location to represent individual device location. The E2A2 location service aims to reduce individual device battery consumption associated with location services while simultaneously maintaining high location accuracy for each device. Our experimental results from a prototype system show the effectiveness of our proposed solution with different mobility patterns. We also present results on the impact of different system parameters and the number of users in a group. Compared to running GPS location services on individual devices separately, our E2A2 service saves on average 33% battery consumption rate when 4 devices are co-located at walking speed and 26% battery consumption rate when 4 devices are colocated on the same bus while meeting the same accuracy requirements.**

I. Introduction

With the rising popularity of mobile phones and the proliferation of mobile applications, mobile devices are becoming increasingly used for location-based sensing. Continuous location tracing also enables novel functionality and societal value, e.g. detecting or predicting depression in the user [1]. However, these applications have high requirements for the accuracy of device location data and conforming to these requirements is expensive with respect to battery consumption.

Prior research has been conducted on maximizing the battery performance for mobile sensing at different system levels, e.g. hardware [16], [7], [3], network [5], [11], or application [15], [23]. Inspired by the crowdsourcing philosophy, we propose an inter-device approach to conserve battery consumption. The idea is motivated by the fact that people exhibit daily and weekly patterns [18], [6], [8] and that devices often co-locate [2] as a result of the users' social network or community structures. For example, Loccachino [14] allows people to share their location with friends and colleagues. Apple's iCloud service enables parents to track the location of their children as they walk home. Tiramisu [22] enables bus riders to trace their bus trips, which generates real-time bus arrival information for fellow riders. The idea of sharing the cost of sensing across co-located devices when the apps running on these devices are measuring the same environmental signal (e.g. location information, air quality, etc.) is appealing.

In this paper, we develop the location service E2A2 (Energy Efficient and Accuracy Aware) in which locations are crowdsourced to reduce the energy cost while maintaining location accuracy. E2A2's architecture is based on a central repository and negotiation hub for mobile devices as part of a generic location service. More specifically, E2A2 realizes the following features. (1) Based on a device's location and its mobility, E2A2 determines if the device's location can be used by another device. In this case, the other device can turn off its location service and save battery power. (2) The system partitions a collection of devices into groups, and suggests when group members should contribute location information, how to compute a group's location and when they can use the group location as their own location. (3) Depending on the accuracy requirements specified by the mobile application, the system provides the applications with device location. Although we focus on location data, our solution is applicable to other crowdsourced sensing platforms and mobile sensor data, such as ultraviolet light, carbon dioxide levels, etc. [9][13].

E2A2's key idea is that a collection of devices can be partitioned into groups and a "central" location of the group can be used as every group member's location without violating accuracy constraints imposed by applications. The tradeoff between using group location to save battery consumption and maintaining high accuracy of individual location information introduces many research challenges. Group location should be defined appropriately, so that individual group member's location will not result in a biased location estimate for other group members. Group location management should request group members' location updates as frequently as possible, while maintaining location accuracy. Given diverse mobility patterns, adaptive grouping policies need to be devised, such that grouping structures do not change unnecessarily, resulting in frequent requests to update locations and high battery consumption.

To address these challenges, we first formulate the problem of accuracy-constrained location crowdsourcing to reduce energy consumption and prove that the problem is NP-hard (Section III). We then design E2A2 system and devise a heuristic algorithm to reduce battery consumption when location accuracy is specified by mobile applications. In particular, we elaborate on how to handle device location update and location requests from mobile apps by defining system parameters and applying these parameters in E2A2 system components (Section IV). We implement a prototype location sharing system for Android phones, and evaluate the experimental results of this

978-1-4799-0428-0/13/$31.00 ©2013 IEEE

prototype operating in a variety of scenarios. We demonstrate the efficacy of our approach, and show how different system and environmental factors impact the effectiveness of the solution (Section V). We further present a discussion and show that applications which rely on close proximity of devices will benefit from E2A2s location service by saving battery power while retaining satisfactory location accuracy (Section VI). Before presenting our work, we first review related work in Section II.

II. Related Work

Existing approaches for reducing the energy cost of location services on a mobile device can be classified into two categories: (1) using alternative, lower-power measurements rather than high-power sensors, and (2) avoiding unnecessary location measurements when the mobile device is stationary. For example, in the first category, EnTrack [10] builds a model of position, velocity, direction and uses the accelerometer to detect when the device is moving. CTrack [20] uses cell-tower fingerprints to localize and track mobile devices. Because these alternative measurement techniques are usually less accurate than high-power, high-accuracy sensors on the mobile device, these techniques typically sacrifice accuracy to lower energy cost. In the second category, the system determines when device movement is unlikely to occur. For instance, *ACE* [17] is an infrastructure focused on optimizing the use of sensors on a single mobile device by continuously tracking and building models of the users' activity. Greenstein [4] focuses on controlling access to fixed sensors, which cannot be used when the device is in motion.

None of the above systems share location information among co-located mobile devices. Another solution, CoMon [13], does share measurement information to reduce energy consumption. However, the design limits sharing to devices that are within direct communication range of each other. Second, CoMon establishes only pair-wise sharing arrangements, which limits the savings of any mobile device to a factor of 2. In E2A2 we use a client-server architecture that, in the best case, provides savings of a function of n, where n is the number of devices in one group (see our analysis in Section IV-B).

There have been research efforts on how to proactively detect proximity between devices [12], [21]. In this paper, we focus on how to group co-located devices based on devices' location updates, how to manage group membership and group location.

III. Problem Formulation

In this section we formalize the problem of using group locations to represent the locations of individual mobile devices as an integer program and then prove that, in general, determining optimal groups of devices is NP-Hard. In our model, we include the following entities: mobile devices, location information of mobile devices, mobile groups and location requests from mobile applications.

Let j be the mobile devices, ($|j| < U$). We partition the devices to W groups, $|W| <= |U|$, as illustrated in Figure 1. Within each group, there must be at least one device i, ($|i| < W$) updating its location, such that its location can be used to represent other group member j's location, and it is possible i is j. The battery cost of turning on i's GPS and updating its location can be AC_i. Applications running on mobile node j may request device locations $d_{j,t}$ at time t, and its request has accuracy requirements. Namely, the error of using i's location to represent j's location, i.e. $error_{i,j,t}$, can not be larger than required value. In this case, group members' updates must be frequent enough to ensure the group location remains effective and meets application requirements.

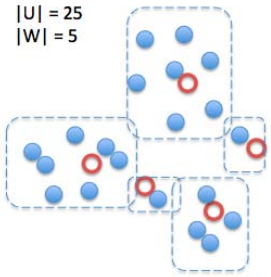

Fig. 1. Five co-located groups. At least one of the group members updates its location to ensure the group location remains current.

Given this model, we can formulate the problem of minimizing the battery consumption of the mobile devices while meeting all the application accuracy requirements as an integer program, where the objective function is:

$$\underset{f_{i,t},\, x_{i,j,t}}{arg\ min} \quad \int_0^T (\sum_i f_{i,t} AC_i \quad + \sum_j \sum_i d_{j,t} error_{i,j,t} x_{i,j,t}) dt \tag{1}$$

where, $f_{i,t}$ equals to 1, indicating i turns on its location service and updates its location; $x_{i,j,t}$ indicating i represents j's location at time t.

To prove that finding optimal groups is NP-Hard, we show that finding optimal groups is at least as hard as a well-known NP-Hard problem, the Uncapacitated Facility Location Problem [19]. We then constrain the time of the crowdsourcing energy problem defined above by assuming f, x and $error$ do not change over time, then the objective function can be:

$$\underset{f_i x_{i,j}}{arg\ min} \quad \begin{array}{c} \sum_i f_i AC_i \\ + \sum_j \sum_i d_j error_{i,j} x_{i,j} \end{array} \tag{2}$$

Function (2) is subjective to the following: $f_i \in \{0, 1\}$, i either turns on or turns off its location service; $x_{i,j} \in \{0, 1\}$, i's location can either represent j or not; for each $j \in |U|$, $\sum_{i \in W} x_{i,j} = 1$, there must be at least one node i from a group can represent its location; and $x_{i,j} <= f_i$, i.e. if i's location is not turned on, its location can't be used to represent others' location.

Given the above, our simplified objective function can be interpreted as an Uncapacitated Facility Location Problem [19], which is NP-Hard. Therefore, in this paper, we will present heuristics solutions to address the problem.

IV. System Design

To address the above grouping problem, we develop E2A2 (Energy Efficient and Accuracy Aware) system. As shown in Figure 2, there are two key components in the E2A2, i.e. *Co-location & Grouping* and *Group Location Management*. The *Co-location & Grouping* component handles device's GPS location updates, finds when multiple devices co-locate, partitions them into groups, and determines when previously co-located mobile devices may be no longer co-located. The component creates new groups as necessary. It also calculates the group location after receiving location updates from group members. The *Co-location & Grouping*'s output is fed into the *Group Location Management* component, such that when the *Group Location Management* receives requests from mobile applications concerning a device's location with accuracy requirements, it can check group membership of the requested device and decides whether to use its group location to represent the device location, if appropriate. In this case, the GPS sensor of the device is left off, thereby saving battery power. Otherwise, it sends requests to devices for GPS location updates, and new GPS location updates will be handled by the *Co-location & Grouping* component as presented above.

In the remainder of this section, we will describe each component's workflow in detail. More specifically, we will introduce a set of system parameters and elaborate on how they are used in key steps of these workflows. We will also discuss the effectiveness of such design on battery consumption.

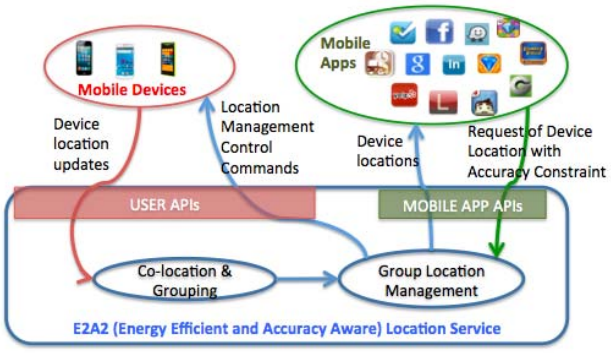

Fig. 2. E2A2 System Model

A. Co-location & Grouping

Figure 3 presents a high-level work flow of the *Co-location & Grouping* component. Briefly speaking, upon receiving a location update from a device, it first checks if this device is a new device in the system. If the device already exists in the system, then it checks if the device's previous group is still valid (*Step CG1*). If so, it checks if the device can keep his group membership given the new location update (*Step CG2*). If not, it checks if the device can join any other existing group (*Step CG3*). Either the device joins an existing group, forms a new group or remains in the existing group, the device's new location is used to calculate and update the the device's group location (*Step CG4*). If the device is new or the device's previous group is not valid anymore, it will eventually go to (*Step CG3*), and follows the previous steps until a new group location is updated (*Step CG4*).

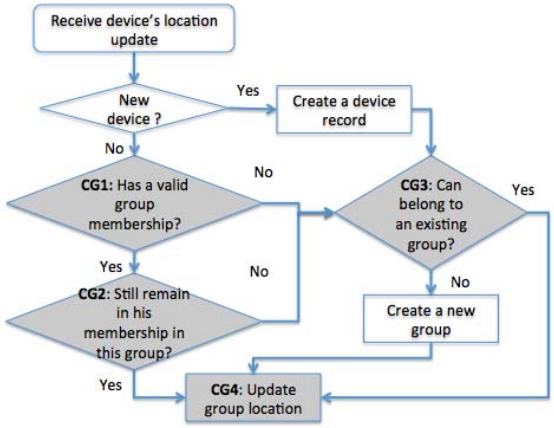

Fig. 3. Upon receiving a device's location update, the *Co-location & Grouping* component groups the device and updates the group's location.

Step CG1: To validate an existing group, we define a system parameter T_{group}. Our rationale is as follows. If none of the group members turns on GPS nor sends location updates, nor the server receives any location information within a period of time, the group location error will be considered infinite and group structure becomes unknown. Thus, if too much time (more than T_{group} seconds) has elapsed, the group is invalidated.

Step CG2: If less than T_{group} seconds have passed, the system checks if the device still keeps its group membership by calculating the distance between the new location and the group location. We define a second system parameter $GroupRadius$ to bound the distance between a new device's location and its group location. If the distance is less than $GroupRadius$ meters, the device remains in the group, i.e. the device is considered colocated to other group members. Otherwise, the device is no longer considered part of this group and, again, may become a member of another group. If the device location passes both the time and distance test, the group's existing location is updated to reflect the new information (in *Step CG4*).

Step CG3: Figure 4 shows an example how a device can form a new group or join an existing group. For example, the E2A2 client library on mobile device A uses the device's location service to detect its location at time $t1$ as point A_{t1}. The library returns this information to the mobile application and sends it to the E2A2 server. In this example, it is the first device so E2A2 forms a new group. With $GroupRadius$, other nodes that fall into the circle which centers on A_{t1} will be considered in the same group. For example, if device B sends its location B_{t1}, and B is within A's grouping area, and A_{t1} lies within B's location error circle, then B can join the existing group of A.

Step CG4: A group location GL is defined as the center location of its group members. As shown in Figure 4, after B joins A's group, GL_{t1} is then updated to be the center of these two devices. When A moves to A_{t2} at time $t2$ and updates its location, the new group location is GL_{t2}. At time $t3$ when B's application sends a query to the E2A2 service asking for B's device location, if B hasn't updated its recent location yet and

the B's estimated location at $t3$ is within the measurement error bounds, i.e. $distance(GL_{t2}, B_{t3}) < Error_{B,t3}$, E2A2 will send GL_{t2} back to B's application, because B can use this location as an approximate measurement of its own location.

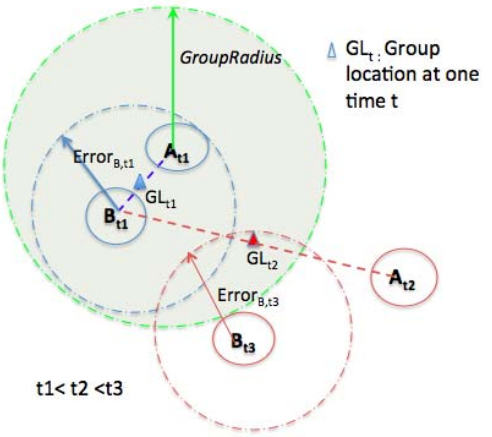

Fig. 4. Grouping and Group Location (GL)

B. Group Location Management

Figure 5 presents a high-level workflow of the *Group Location Management* component. Upon receiving an application's request for one device's location with accuracy constraint, the *Group Location Management* component tries to find out if it can use group location to serve this request. Depending on who updates the last group location (the requested device itself or other group members), the component will use different criteria to make the decision.

More specifically, E2A2 first checks if the device exists in the system. If it does, then E2A2 checks if the device has a valid group membership using the same function introduced in *Step CG1*. If it has a valid group membership, E2A2 finds out who lastly updates the group location. We treat the update from the requested device itself and other group members differently using two system parameters, i.e. T_{self} and T_{others}. The parameter T_{self} is used when the last updated location of a group is from the requested device. Basically, in step *GLM1*, if the location request time is less than T_{self} of its last update, then we regard the current group location as an acceptable proxy for the devices location. Otherwise, we will request the device to update its location. The parameter T_{others} time limit is used when the last updated location of a group is from a device other than the requested one. In step *GLM2*, if the location request is less than T_{others} of others' last update, then we regard the current group location is acceptable. Otherwise, E2A2 will request the device to update its location. E2A2 will also request the device to update its location when the device does not exist in the system, or when its group membership is not valid.

C. Analysis of E2A2 Design

The above workflows and system parameters are the key E2A2 system design. In this section, we give a case analysis on how such design impacts battery consumption. Assume one device forms a group itself and assume requests for its

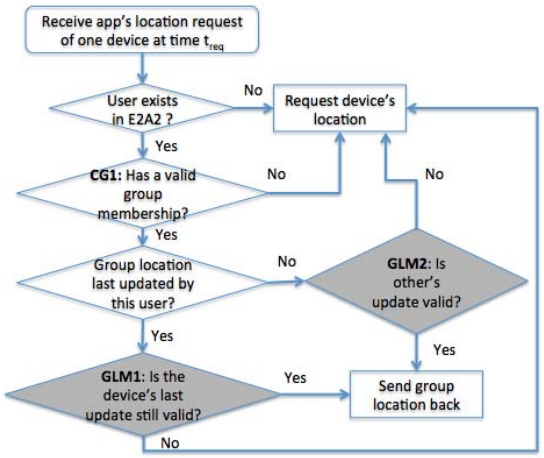

Fig. 5. A high level workflow of the *Group Location Management* component

location is very frequent and evenly distributed over time,e.g. every T_{req} seconds. Within T_{group} seconds, the device has to update its location every T_{self} seconds. However, if it belongs to a group of n devices, and no device leaves the group and coordination continues, then the device will update every $n * T_{others}$ seconds. Thus, as long as $n * T_{others} > T_{self}$, each device will save battery from using the group location. With n increasing, the savings will be linearly increasing as well. The ratio of savings will be $n * T_{others}/T_{self}$, if T_{group} is a multiple of T_{others} and T_{self}.

The above analysis is only applicable to battery savings. Because the group location is calculated by group members' updates and bound by $GroupRadius$, there is no such a linear relationship between the system parameters and location error.

V. Experimental Framework and Results

To evaluate our solution, we developed a prototype location sharing system for Android phones. Our experimental approach includes a combination of recorded traces, an execution of the system based on the recorded traces, and a simulation of battery consumption. This experimental design provides both realistic system tests and a thorough exploration of the parameter space in a simulation. The Android app we developed collects location traces and remaining battery on the device. In our measurements, the only application running was the E2A2 location client, and we run experiments on two different phone models - HTC Nexus One (Android v2.3.6) and Samsung Galaxy Nexus (Android v4.2.2).

When modeling energy consumption on a phone, many factors are involved: e.g. GPS on/off, network package size, frequency of sending the messages, etc. However, because our experiments can potentially experience an order of magnitude difference in total energy consumption across the system, we simplify the battery consumption model to measuring the drain on the battery as a result of running the location service on the mobile device when user moves (either staying stationary or walking or riding a bus). More specifically, in the simulation, it takes the device about S seconds to finish the following: turning on GPS sensor, acquiring GPS location data and sending the location to E2A2 server; and such a process

consumes the device Er times of the battery consumption when without turning on the GPS. To determine the value of S and Er, we ran experiments on two Android phones and other location-based apps, e.g. Tiramisu [22]. We find receiving good GPS location usually takes several seconds, though when device is in downtown area, it can take much longer time up to several minutes. In our model, we set S=5 seconds. Different phones definite will have different Er values. Our experiments on two phone models show that the battery consumption rate is about 10% per hour when the device keeps GPS on, and leaving the phone stand by will consume about 1% battery per hour. Thus we set Er= 10 in our experiments.

We measured performance of battery usage and accuracy for three scenarios: stationary, walking, and taking a bus. These scenarios cover a range of typical activities for mobile phone use. We defined different parameters to demonstrate the performance of the system in the different scenarios (Table I). For simplicity, applications request for device location every 2 seconds (T_{req}) in our experiments.

TABLE I. PARAMETER VALUES FOR EXPERIMENTS.

Parameter	Stationary	Walking	Bus
T_{req} (s)	2	2	2
T_{self} (s)	40	40	15
T_{others} (s)	30	30	10
T_{group} (s)	60	60	30
$GroupRadius$ (meters)	25	25	100

A. Basic Results of Different Mobility Patterns

1) Walking in Star Shape: Our first real-world scenario, a $Star$ formation, explores a simplified case where users converge and then depart to their respective starting points. This would be similar to four people all getting coffee or food from the same source around the same time. In this scenario, four people carrying one mobile device each starts at roughly equal distances from an intersection. At a designated point in time they walk towards the center of the $Star$, meet at the center, stay stationary for a while (e.g. around 2 mins), then reverse their paths to travel back to the edges of the $Star$. Figure 6 illustrates the 18-min device trajectories on a map for this $Star$ pattern.

We gathered location traces of this pattern by having four researchers walk in the Star pattern. The location traces are then used to drive various simulations of the E2A2 implementation. Locations were sampled every second during the trace. Figure 7 is a record of the change in grouping as the devices move. The x-axis is time in seconds, the y-axis is the number of groups. Initially, all devices are in separate groups. As the participants approach the center of the star the groups merge, eventually into a single group containing all four devices. As the devices return to the edge of the star, the devices leave the single group.

Figure 8 shows that, as expected, battery consumption is the lowest when four devices are grouped together. The first column represents the average battery usage rate when at the beginning of the experiment, four devices make their own groups separately. The forth column in the center shows that all 4 devices colocate in the center of the $Star$ and form only one group for 106 seconds (from second 529 to second 635 as indicated on the x-axis). The distribution of the columns on time matches the grouping results in Figure 7. When more devices share locations in a group, the total battery consumption of the system declines because the group locations is used as a proxy for individual device's location, and more devices save battery from turning on their GPS sensors.

When people co-locate into groups, we expect the location errors to become larger because the group location is used instead of individual device location. However, Figure 9 shows that when the devices are stationary in the center (the forth column), the location error temporarily becomes smaller. The stationary, colocated, group of users causes this drop in error. We expect when devices are in fast movement, the error will be larger, which is presented in the following Bus scenario.

Fig. 6. 18-min Walking - A map of $Star$ pattern

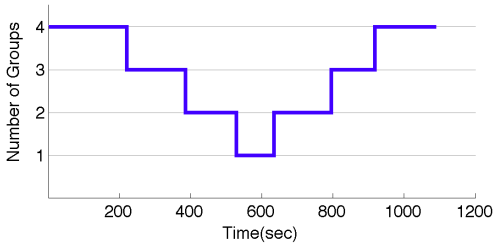

Fig. 7. Change in Grouping of 4 devices - $Star$

2) Riding a Bus Together: The Bus scenario examined people boarding the same bus along a sequence of stops. Four researchers boarded the same bus at four successive bus stops. This is a common mobility pattern for mobile app users who use transit. Figure 10 show a map of the scenario, where within a 2.5-min bus ride, four devices share the same ride for 1 minute and 20 seconds. The results are shown in Figure 11 and Figure 12. Similar to the results of $Star$, E2A2 trades off location accuracy with battery consumption. Because the precision of civilian GPS on a mobile phone can vary from 20 meters to 100 meters, the location error as a result of grouping varying from 12.92 meters to 47.807 meters (in Figure 12) should be acceptable to many apps.

B. Impact of Different System Parameters and Factors

The above experimental results demonstrate that under different mobility patterns E2A2 balances the tradeoff between battery consumption and location errors. We now present how various system parameters and factors impact E2A2's performance.

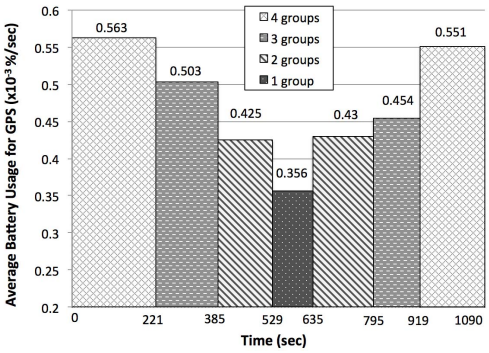

Fig. 8. Battery Consumption - $Star$

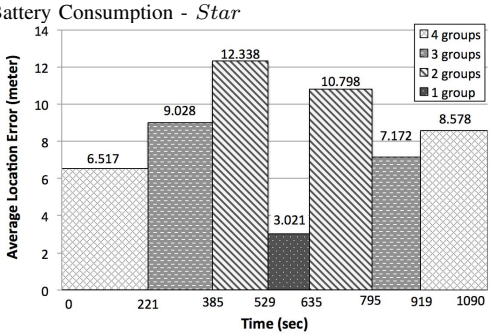

Fig. 9. Location Error - $Star$

1) Impact of Timeout Parameters and Group Radius: As previously discussed, E2A2 uses three system parameters T_{group}, T_{self} and T_{others} to manage grouping and group locations. To evaluate the impact of these parameters on system performance, we examined two metrics: accuracy score and battery score. To calculate accuracy score, whenever the server sends back a location to the device, the device will calculate the error distance (in meters) between the group location and the actual location (read from the trace file). At the end of the simulation, an error of 200m or worse results in a score of 0 and an error of 0 results in a score of 1, an error between 0 and 200m will be calculated by (200 - error)/200. Thus, larger accuracy score indicates smaller error. Battery score is calculated as (the total time of trace - the total time turned on the GPS during the trace) divided by the total time of trace. This is to compare E2A2 battery consumption with when the GPS is turned on all the time. Thus larger battery score means less battery consumption.

Figure 13 and 14 were generated using data from the $Star$ scenario, where T_{group} = 60 seconds. Clearly, given a fixed $GroupRadius$, small Timeout thresholds improve location accuracy but increase battery consumption.

2) Varying App's Requirements on Location Accuracy and Battery Consumption: One of the goals of E2A2 is to accommodate variation in application requirements for location data. The results presented in the above were generated with a consistent accuracy requirement level. For this analysis, we varied application requirements for location accuracy, and re-evaluated system performance. Because different location service providers, e.g. GPS, WiFi, and Cell Tower, can provide three levels of location accuracy, we simulate applications running on the devices request one of these three accuracy

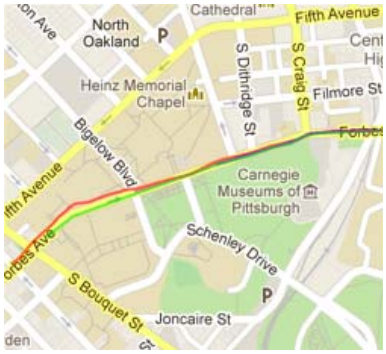

Fig. 10. 2.5-min Bus Riding - Bus

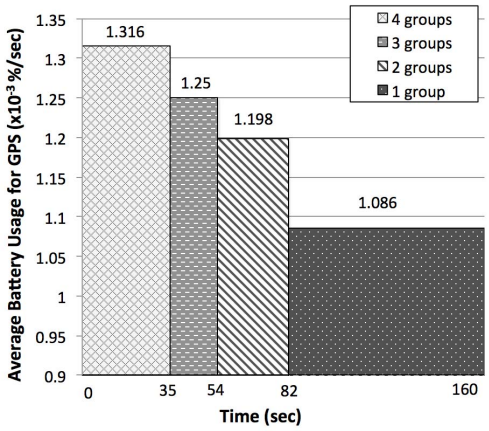

Fig. 11. Battery Consumption - Bus

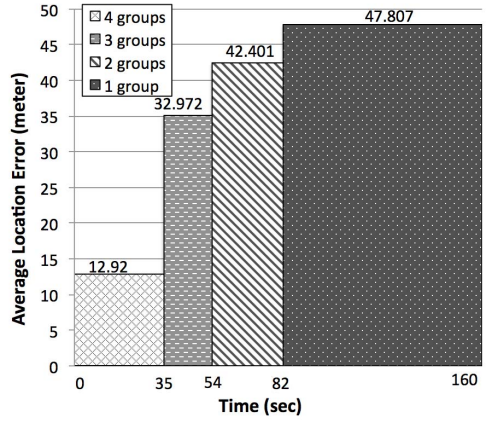

Fig. 12. Location Error - Bus

levels. A higher accuracy level consequently maps to a smaller T_{self} value. Figure 15 and Figure 16 are based on the same $Star$ walking traces however with varying app's requirements on location accuracy. The results are comparable to the earlier results for uniform application requirements on location accuracy.

3) Group Size on Location Accuracy: We also measured four researchers walking in the same direction and remaining co-located for five minutes. We then measured the E2A2 average error of this scenario. We then successively removed

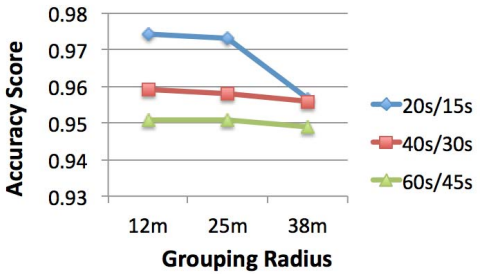

Fig. 13. Accuracy score (T_{self} / T_{others} value set is shown in the legend).

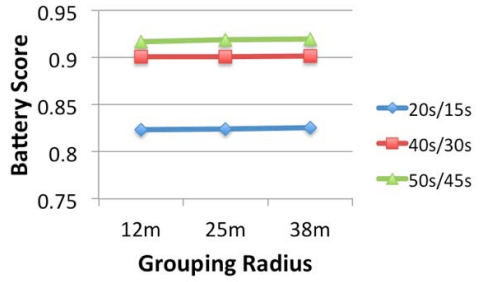

Fig. 14. Battery score (T_{self} / T_{others} value set is shown in the legend).

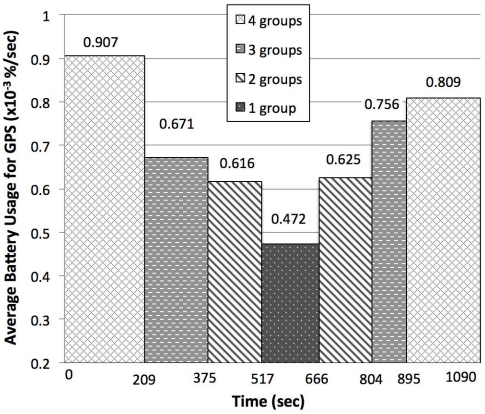

Fig. 15. Battery consumption under varying App's Requirements on accuracy

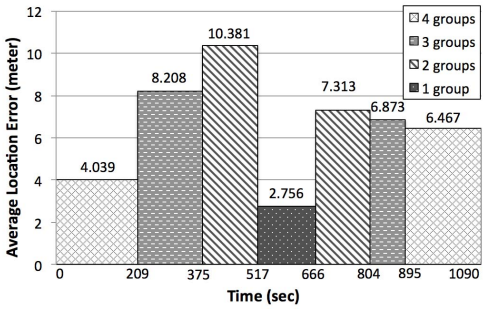

Fig. 16. Location Error under varying App's requirements on accuracy

research traces and reran the simulation, measuring again the average error. Figure 17 shows that when devices co-locate during the entire period of the experiment, i.e. a 5-min co-walking, because they only forms one co-located group, increasing group size (with more devices in the experiments), the location error will be reduced. This is because more devices in one group cause more apps requesting their group location to be updated more frequently, tracking location of co-located devices more closely.

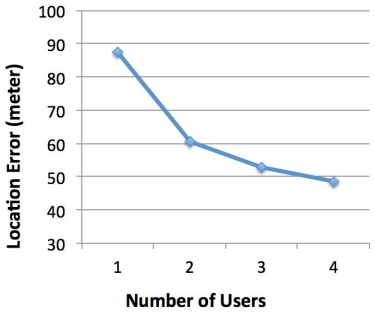

Fig. 17. Location error is reduced when more users co-locate in one group.

VI. Discussion of Transit Results

Besides the questions of accuracy and energy savings, the use of E2A2 raises the question of overall impact. If people are not colocated frequently, the savings from the system will be small. To ground the impact of the system in a real world application, we acquired passanger load data for one bus from the Port Authority of Allegheny County (PAAC), the transit operator for Pittsburgh, PA. The data was gathered on Octber 16th, 2009, a Friday, using automated passenger counters, implemented as infrared sensors mounted on the entrances and exits of the bus. The bus was used on three separate fixed transit routes during that day. The data is recorded with a timestamp of each event of the counter recording entrances and exits to the bus.

We used this log of data to construct a histogram of the passenger load of the bus (Table II). By load, we mean the number of passengers on the bus at a particular point in time. In the left column, the bus load is listed. The next column, $Colocation$, is the total number of seconds the bus operated with the given load. The next column, $Battery\ Hour$ is the number of seconds that all batteries would be operating, if every phone had its GPS chip on during the transit ride. This column is simply a product of the $Load$ and the $Colocation$ columns. The column labeled $E2A2$ is the percentage savings estimated by linear interpolation for the level of load based on the analysis in Section IV-C. The basis of the interpolation is the row for a 4 person load (in italics), which is the saving ratio of battery consumption by using E2A2 service compared to only using GPS sensor when 4 people co-locate on a bus ride. Using this data, we linearly scale performance based on the load. In addition, we set the performance for a load of 1 to 0, because E2A2 doesn't help in this situation. The column labeled $Savings$ is the total seconds of battery time saved by using E2A2 for this bus for one day. This column is the product of the E2A2 column and the battery column.

The data shows that E2A2 provides more and more savings as more devices co-located, but that the size of groups of devices colocate is bound, in this application at least, at thirteen. The total savings is 90,370.97 seconds, or more than 25 hours. Since PAAC runs approximately 800 buses a day, deployment of E2A2 to every bus rider would result

in 20,082.8 hours of battery savings per day. Of course, the savings are only for GPS usage, but the GPS chip is one of the most significant sources of battery drain. Deploying E2A2 to every rider is a challenging task, because not every rider owns a smart phone. However, smart phone usage is expected to reach near 100% penetration in a few years. Delivery of E2A2 can be done either through the operating system or as part of a transit information system application [22].

TABLE II. BATTERY HOUR SAVINGS BASED ON THE HISTOGRAM OF BUS LOAD FOR ONE DAY

Load	Colocation (secs)	Battery Hour (secs)	E2A2 (%)	Savings (secs)
13	469	6097	87.0675	5308.51
12	1740	20880	80.37	16781.26
11	1007	11077	73.6725	8160.70
10	5143	51430	66.975	34445.24
9	1317	11853	60.2775	7144.6921
8	949	7592	53.58	4067.79
7	682	4774	46.8825	2238.17
6	641	3846	40.185	1545.51
5	3165	15825	33.4875	5299.40
4	3165	12660	***26.79***	3391.61
3	1204	3612	20.0925	725.74
2	4712	9424	13.395	1262.34
1	7892	7892	***0***	0
Total	32086	166962	602.775	90370.97

VII. SUMMARY

Mobile applications require location information at varying levels of accuracy and frequency, resulting in a varying degree of battery consumption. We develop a crowdsourcing-based E2A2 location service designed to limit battery drain while preserving location accuracy. E2A2 collects individual device location information, detects co-location situations, and enables devices to utilize nearby device locations as their own. This allows GPS sensors to be left off or sampled less frequently to save battery consumption yet still maintain a tolerable level of location accuracy. We implement a prototype for Android phones to test E2A2 services and demonstrate the efficacy of our approach. We also examine how user mobility and system parameters and environment features impacted system performance. Furthermore, we discuss on the impact of the system in a real world application using transit load data. We believe this technology can enable a new class of applications where continuous location sensing is required. Privacy issue of location tracking is beyond the scope of this paper and will be researched in our future work.

VIII. ACKNOWLEDGEMENTS

This work was supported by the Rehabilitation Engineering Research Center on Accessible Public Transportation (RERC-APT), which is funded by grant number H133E080019 from the United States Department of Education through the National Institute on Disability and Rehabilitation Research. Additional resources for this work were provided by Google and by the Traffic21 initiative at Carnegie Mellon. We acknowledge Michael Quinn, Weber Schulz and John Balser Mathews for their assistance to the project, and we thank William Eddy for preprocessing the transit load data.

REFERENCES

[1] S. K. Chennuru, P.W. Chen, J. Zhu, and Y. Zhang. Mobile lifelogger - recording, indexing, and understanding a mobile user's life. In *MobiCASE*, 2010.

[2] E. Cho, S. A. Myers, and J. Leskovec. Friendship and mobility: user movement in location-based social networks. In *Proceedings of the 17th ACM SIGKDD*, 2011.

[3] G. Dhiman and T. Pusukuri, K.K.and Rosing. Analysis of dynamic voltage scaling for system level energy management. In *Proceedings of the conference on Power aware computing and systems*, 2008.

[4] B. Greenstein and B. Longstaff. Followme: enhancing mobile applications with open infrastructure sensing. In *Proceedings of the 12th Workshop on Mobile Computing Systems and Applications*, 2011.

[5] J. Hare, D. Agrawal, A. Mishra, S. Banerjee, and A. Akella. A network-assisted system for energy efficiency in mobile devices. In *Communication Systems and Networks (COMSNETS)*, 2011.

[6] E. Herder and P. Siehndel. Daily and weekly patterns in human mobility. In *UMAP Workshops*, 2012.

[7] H. Huang, K.G. Shin, C. Lefurgy, and T. Keller. Improving energy efficiency by making dram less randomly accessed. In *Proceedings of the 2005 international symposium on Low power electronics and design*, 2005.

[8] C. Kang, S. Gao, X. Lin, Y. Xiao, Y. Yuan, Y.and Liu, and X. Ma. Analyzing and geo-visualizing individual human mobility patterns using mobile call records. In *Geoinformatics*, 2010.

[9] S. Kim and E. Paulos. inair: measuring and visualizing indoor air quality. In *Proceedings of the 11th international conference on Ubiquitous computing*, 2009.

[10] M. B. Kjaergaard, J. Langdal, T. Godsk, and T. Toftkjaer. Entracked: energy-efficient robust position tracking for mobile devices. In *MobiSys*. ACM, 2009.

[11] V. Kolios, P.and Friderikos and K. Papadaki. A practical approach to energy efficient communications in mobile wireless networks. *Mob. Netw. Appl.*, 17:267–280, 2012.

[12] A. Küpper and G. Treu. Efficient proximity and separation detection among mobile targets for supporting location-based community services. *SIGMOBILE Mob. Comput. Commun. Rev.*, 10(3):1–12, July 2006.

[13] Y. Lee, Y. Ju, C. Min, S. Kang, I. Hwang, and J. Song. Comon: cooperative ambience monitoring platform with continuity and benefit awareness. In *Proceedings of MobiSys*, 2012.

[14] Locaccino. http://locaccino.org.

[15] A. P. Miettinen and V. Hirvisalo. Energy-efficient parallel software for mobile hand-held devices. In *Proceedings of the First USENIX conference on Hot topics in parallelism*, 2009.

[16] A. W. Min, R. Wang, J. Tsai, M. A. Ergin, and T. Tai. Improving energy efficiency for mobile platforms by exploiting low-power sleep states. In *Proceedings of the 9th conference on Computing Frontiers*, 2012.

[17] S. Nath. Ace: Exploiting correlation for energy-efficient and continuous context sensing. In *Proceedings of MobiSys*, New York, NY, USA, 2012. ACM.

[18] S. Phithakkitnukoon, T. Horanont, G. Di Lorenzo, R. Shibasaki, and C. Ratti. Activity-aware map: identifying human daily activity pattern using mobile phone data. In *Proceedings of the First international conference on Human behavior understanding*, 2010.

[19] D.B. Shmoys, E. Tardos, and K. Aardal. Approximation algorithms for facility location problems abstract. In *Proceedings of the twenty-ninth annual ACM symposium on Theory of computing*. ACM, 1997.

[20] A. Thiagarajan, L. Ravindranath, H. Balakrishnan, S. Madden, and L. Girod. Accurate, low-energy trajectory mapping for mobile devices. In *Proceedings of NSDI*, 2011.

[21] P. S. Tikamdas and A. El Nahas. Direction-based proximity detection algorithm for location-based services. In *WOCN*, 2009.

[22] Tiramisu. http://www.tiramisutransit.com.

[23] K. N. Truong, J. A. Kientz, T. Sohn, A. Rosenzweig, A. Fonville, and T. Smith. The design and evaluation of a task-centered battery interface. In *Proceedings of Ubicomp*, 2010.