Fostering ParticipAction in Smart Cities: A Geo-Social Crowdsensing Platform

Giuseppe Cardone, Luca Foschini, Paolo Bellavista, and Antonio Corradi, University of Bologna
Cristian Borcea, Manoop Talasila, and Reza Curtmola, New Jersey Institute of Technology

ABSTRACT

This article investigates how and to what extent the power of collective although imprecise intelligence can be employed in smart cities. The main visionary goal is to automate the organization of spontaneous and impromptu collaborations of large groups of people participating in collective actions (i.e., *participAct*), such as in the notable case of urban crowdsensing. In a crowdsensing environment, people or their mobile devices act as both sensors that collect urban data and actuators that take actions in the city, possibly upon request. Managing the crowdsensing process is a challenging task spanning several socio-technical issues: from the characterization of the regions under control to the quantification of the sensing density needed to obtain a certain accuracy; from the evaluation of a good balance between sensing accuracy and resource usage (number of people involved, network bandwidth, battery usage, etc.) to the selection of good incentives for people to participAct (monetary, social, etc.). To tackle these problems, this article proposes a crowdsensing platform with three main original technical aspects: an innovative geo-social model to profile users along different variables, such as time, location, social interaction, service usage, and human activities; a matching algorithm to autonomously choose people to involve in participActions and to quantify the performance of their sensing; and a new Android-based platform to collect sensing data from smart phones, automatically or with user help, and to deliver sensing/actuation tasks to users.

INTRODUCTION

Recent advances in wireless communications, smart devices, and social computing applications are enabling new urban sensing and management opportunities. Smart phones and mobile platforms available on the consumer market already permit accurate tracing of world-related information and (physical) activities of citizens by taking advantage of people willing to collaborate toward a continuous data harvesting process, called *crowdsensing*. At the same time, being relatively new, this physical world perspective still requires much work to orchestrate and manage crowdsensing processes effectively. From a social perspective, there is the need to identify people willing to participate in urban sensing tasks and to find good incentives for participation, not only monetary rewards but also social ones (e.g., cleaner and safer cities). Once identified, the participants have to be kept in the crowdsensing loop, thus fostering people's *participAction*, which involves active participation in sensing campaigns. From a more technical perspective, one of the main challenges is finding a good balance between system scalability and sensing accuracy for city-wide deployment environments. In such a new socio-technical system, the types of resources are very different, spanning from computing ones (network bandwidth, memory, CPU, etc.) to humans (number of people involved, human attention, personal skills to contribute, etc.). Thus, it is impossible to fully control them.

At the current stage, although a few seminal works have started to consider how to facilitate the delivery of crowdsensing tasks and the collection of their results [1–3], much work is still to be done to characterize and manage urban crowdsensing due to several open issues. First of all, human behavior is inherently difficult to predict and calls for novel approaches based on probabilistic models. In addition, to design meaningful models, we need to run extensive experiments to collect large datasets by involving a high number of socio-technical resources for a long time. Finally, while some efforts have started to appear in the crowdsourcing research community to mathematically model and quantify socio-technical resources and the performance of accomplished crowdsourcing tasks (completion time, ratio, etc.) in fixed Internet settings [4, 5], to the best of our knowledge this relevant effort is still missing in the mobile crowdsensing area, which is becoming of greater and greater importance with tremendous opportunities offered by people carrying today's smart phones.

The article addresses all the above issues and truly enriches the whole crowdsensing management cycle by proposing novel technical solutions that exhibit several original characteristics. First, it proposes a novel geo-social model that statistically quantifies and describes, in a compact and easy-to-use way, how socio-technical resource availability varies in space and time,

0163-6804/13/$25.00 © 2013 IEEE

such as population density and region vitality. The core idea is to build time-variant resource maps that could be used as a starting point for the design of crowdsensing participActions. Second, it studies and benchmarks different matching algorithms aimed to find, according to specific urban crowdsensing goals geo-localized in the smart city, the "best" set of people to include in the collective participAction. Here the technical challenge is to find, for the specific geo-socially modeled region, the good dimensioning of number/profile of involved people and sensing accuracy. Third, it presents a new crowdsensing platform that consists of an Android mobile app to ease crowdsensing tasks' delivery/execution and upload the collected results, and an infrastructure-side server to manage crowdsensing tasks and associated incentives. We have validated the proposed solution over a large set of data collected for two months over a population of 44 people; we present technical results about the accuracy of the employed matching algorithms, and the results of a survey to assess user satisfaction with our crowdsensing mobile app and the received incentives. The collected results show that, with relatively low socio-technical resource overhead, it is possible not only to achieve good sensing accuracy, but also to minimize monetary incentives necessary to drive user participation.

Background and State of the Art

ParticipAction services typically cross-cut and work at the intersection of two main research areas: modeling/managing socially interactive Smart City systems and crowdsourcing techniques toward collective intelligence in urban sensing. Without claiming completeness, due to space limitations, this section briefly overviews the current state of the art in these very active fields.

Smart City Systems

Recent disruptive technologies — such as location-based services, Internet of Things, and multimodal interfaces — are the basis of new distributed services that in the near future will use cities as development and deployment platforms. Worldwide, large cities are extremely important in social development: 50 percent of the world population lives in cities, the top 100 urban centers account for 25 percent of the global gross domestic product, and by 2050 the urban population will be almost 6.4 billion people; socially, the tightly knit collaborative community of cities promotes free flow of ideas, leading to exponentially greater innovation. Since cities account for more than 75 percent of the global energy consumption and are responsible for 80 percent of the total greenhouse gases, they have to be considered the central focus for driving worldwide sustainability. The high population density and the very large number of different interconnected issues make effective city management a challenging task, but at the same time future smart cities provide countless opportunities to foster human collaboration by keeping them in the loop.

Several significant government and industrial research efforts are currently underway and confirm the impact that smart technologies may have on society in the near future. The European Digital Agenda has funded many projects, such as European Digital Cities,[1] InfoCities,[2] IntelCity roadmap,[3] and EUROCITIES,[4] to promote smart urban services. The EU has also funded FuturICT,[5] a long-lasting 10-year initiative that seeks to "understand and manage complex, global, socially interactive systems, with a focus on sustainability and resilience." Around the world, the government of South Korea is building the Songdo Business District, a green low-carbon area that aims at becoming the first full-scale realization of a smart city.[6]

Also many corporate initiatives are on the way. Among them, a significant one is the IBM Smarter Planet project that promotes the deployment of several "smarter systems" to improve social progress,[7] from smart grids and traffic managers to water management and cheaper/safer healthcare. A similar effort led by Intel aims at using London as a testbed where users and the existing city infrastructure are exploited to improve city efficiency, for instance, to manage traffic flows, predict extreme weather conditions, and monitor water supplies.[8]

Mobile Crowdsourcing Models and Tools for Collaborative Sensing

One way to collect sensing data across large cities in a scalable manner is to exploit the full potential of crowdsensing: while crowdsourcing aims to leverage collective intelligence to solve complex problems by splitting them in smaller tasks executed by the crowd, crowdsensing splits the responsibility of harvesting information (typically urban monitoring) to the crowd. Crowdsensing is an important technological enabler for smart cities that has attracted several research efforts aimed at improving sensing quality on mobile devices, promoting user participation, and validating collected data [6, 7]. Compared to infrastructure-based sensing, crowdsensing has several advantages, but also brings in some additional challenges. Unlike an infrastructure-based sensing solution, crowdsensing can potentially be cheaper as it does not require the deployment of expensive fixed infrastructure. Ultimately, the trade-offs are akin to the trade-offs that exist when using an ad hoc network instead of a fixed infrastructure network: it is easier to deploy and can be used in areas where deploying a fixed infrastructure is difficult/impossible, but introduces additional complexity and challenges. In general, mobile devices used for crowdsensing and infrastructure-based sensing are complementary technology that can cooperate to enable sensing in smart cities [8].

Crowdsensing has similarities with traditional crowdsourcing platforms, such as Amazon Mechanical Turk (AMT), which act as mediators between task clients, usually called workers, and providers: the introduction of such platforms has driven specific research efforts that aim at optimizing crowdsourcing resources [9]. For example, Bernstein *et al.* propose a statistical model and an algorithm to pre-recruit and pay workers

One way to collect sensing data across large cities in a scalable manner is to exploit the full potential of crowdsensing: while crowdsourcing aims to leverage collective intelligence to solve complex problems by splitting them in smaller tasks executed by the crowd, crowdsensing splits the responsibility of harvesting information to the crowd.

[1] *http://www.digital-cities.eu/*

[2] *http://www.infocities.eu/*

[3] *http://intelcities.iti.gr/intelcities*

[4] *http://www.eurocities.eu/*

[5] *http://www.futurict.eu*

[6] *http://www.songdo.com*

[7] *http://www.ibm.com/smarterplanet/us/en/overview/ideas/*

[8] *http://www.ucl.ac.uk/news/news-articles/May2012/240512-*

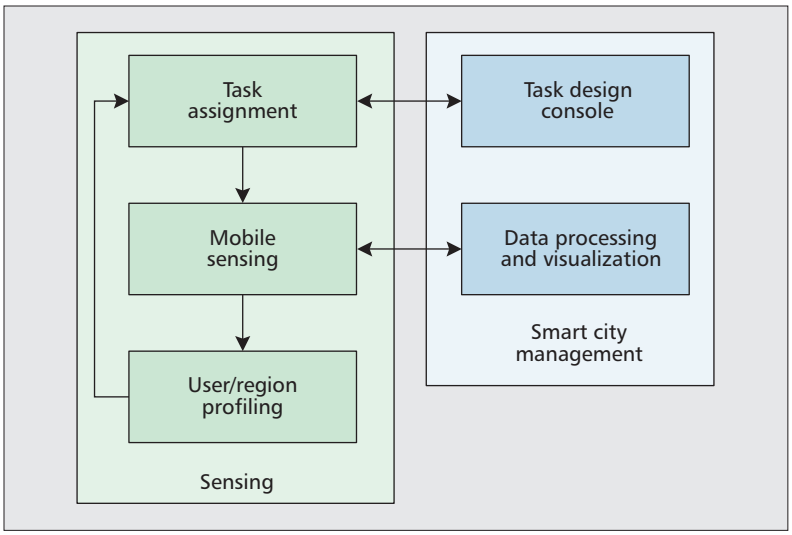

Figure 1. *The feedback loop of our mobile sensing for smart cities: users sense the surrounding environment; McSense profiles their performance and helps design/assign new tasks, which feed new sensing activities.*

to minimize task execution time on AMT [4]. Another important efficiency goal is to minimize the number of workers to get a sufficiently reliable result; for instance, CrowdSense aims at sampling subsets of workers and weighting their contributions to efficiently obtain an output that approximates the opinion of the whole crowd [5].

These crowdsourcing models, tools, and platforms are typically not suitable for mobile crowds because they are unable to exploit the sensors available onboard mobile devices and do not include context-aware mechanisms, which are necessary for effective mobile sensing (e.g., task replication to mitigate client mobility or impossibility to connect). Therefore, mobile crowdsensing is recently emerging as a new research area to propose original solutions that enhance and extend traditional crowdsourcing platforms with the goal of facilitating the management, delivery, and execution of potentially global massive sensing tasks to mobile clients. In the following, we briefly report a selection of these projects, from the first seminal works in crowdsensing to the research activities that are closer to our work.

PEIR is an application that exploits mobile phones to evaluate if users have been exposed to airborne pollution, enables data sharing to encourage community participation, and estimates the impact of individual user/community behaviors on the surrounding environment [1]. mCrowd is an iPhone app that enables users to post and work on sensor-related tasks, such as requesting photos of a specific location, asking to tag photos, and monitoring traffic [2]. Medusa is the project closest to our effort: it uses a high-level domain-specific programming language, called MedScript, to define sensing tasks and workflows that are promoted with monetary incentives to encourage user participation [3]. By using MedScript it is possible to organize incentives, sensing tasks, and processing tasks in high-level workflows, while the underlying Medusa framework hides the resulting complexities and takes care of task coordination, worker management, incentive assignment, and result collection. PEIR, mCrowd, and Medusa are important milestones of crowdsensing, but still do not include any specific model to ease the decisions of mobile sensing task assignment to people and to quantify urban sensing particip**Actions**. Our platform aims to fill this gap, as detailed in the following.

McSense: A Geo-Social Mobile Crowdsensing Platform for Smart Cities

In its simplest formulation, crowdsensing is a two-step process: to assign sensing tasks to users and to wait for results after assignment completion. A more refined approach, taken by McSense — our mobile crowdsensing platform — is to exploit information about potential workers and their mobile execution context (processing power of their phones, battery level, people's geographical location, etc.) to better and more effectively tailor the task assignment process. In particular, McSense puts task description, task assignment, and mobile sensing in a closed loop that allows more efficient and effective usage of all socio-technical resources involved.

Figure 1 shows the McSense sensing lifecycle, which consists of three main modules: mobile sensing, user/region profiling, and task assignment. The Smart City Management system interacts with the sensing modules to provide a full-fledged framework for data sensing including functions to create and describe new sensing tasks and to analyze sensed data. *Mobile sensing* is a comprehensive term used to describe all types of sensing activities carried out by users via their mobile devices, such as recording noise pollution and taking pictures. *Data processing* analyzes the output of mobile sensing to reap the relevant data for the active sensing task. *User profiling* processes the collected data and aims to profile user particip**Action** by drawing accurate estimations about their potential involvement in future mobile sensing tasks; moreover, to avoid long processing operations and improve user experience, McSense includes region profiling to cache already evaluated user profiles, stored by geo-localized regions, thus also exploiting the physical locality principle (higher probabilities of similar profiles in the same region). *Task assignment* automatically assigns tasks to users as a function of their characteristics derived by the profiling module. The *task design console* enables the creation and publication of new crowdsensing tasks and includes tools and support components to assist the design of these tasks. *Data processing and visualization* provides active mapping features and exploits over-imposed informative visualization layers to show statistics about completed sensing tasks and collected data.

Before providing more details about these core components, we briefly introduce and define more formally three core McSense entities: *worker*, *task*, and *task app*. *Worker* is the user running the McSense app and specific sens-

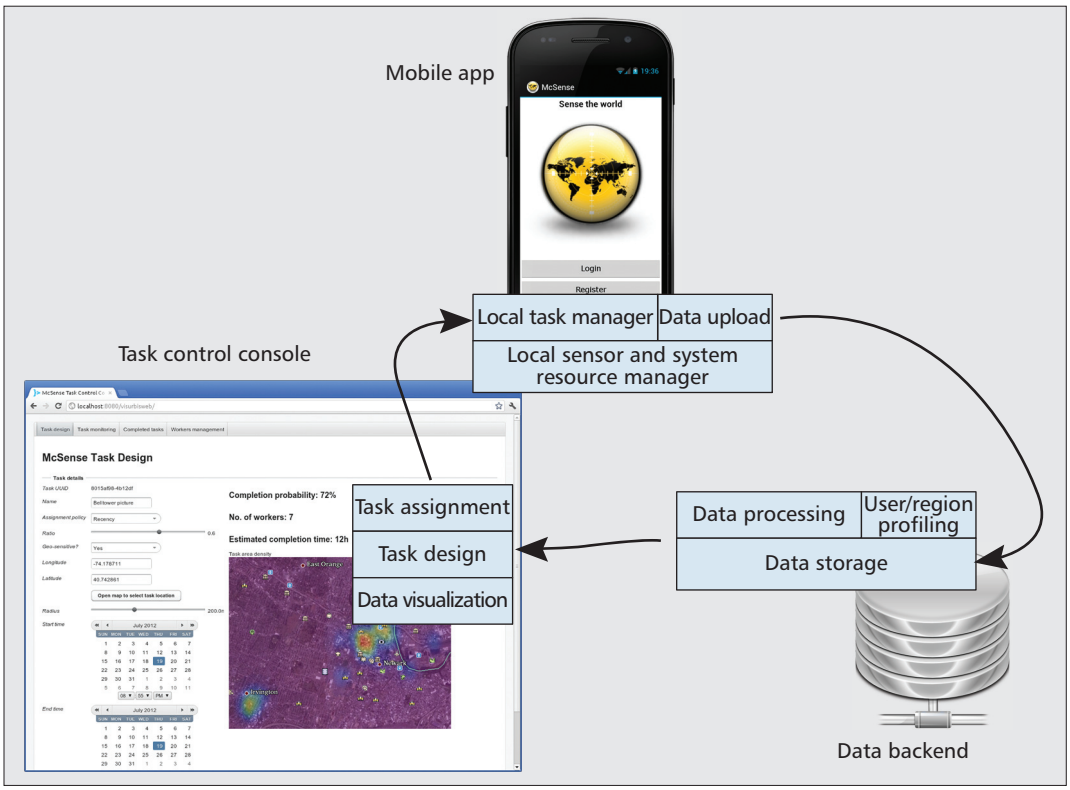

Figure 2. *The McSense distributed architecture.*

A more refined approach, taken by McSense, is to exploit information about potential workers and their mobile execution context (processing power of their phones, battery level, people's geographical location, etc.) to better and more effectively tailor the task assignment process.

ing tasks on her smartphone; in particular, in McSense, we assume that worker particip Action is incentivized with a monetary compensation reported to users with the request to participate in a crowdsensing action. However, we could think of that monetary compensation as virtual currency, which could be translated into other types of incentives in the future (e.g., increased social recognition/visibility and city/task-related awards). Tasks are geo-dependent units of sensing work and are either long-running, such as recording location via GPS or noise-level via a microphone for six hours, or one-shot, such as taking a picture of a specific location. For task localization, we adopt a simplified model that associates each task with three properties: location, area, and duration. Location is the set of physical GPS coordinates; area is a circle-like region determined by a radius, and a task is successfully executed if the worker executes it within the area; and duration is the time a task remains valid and waiting for possible completion before its deadline. Finally, to ease the sensing action for workers, we provide each sensing task with a ready-to-use and intuitive mobile app, called a task app, which includes everything needed to complete the required task. For instance, for a geo-localized photo, the McSense task app interacts directly with GPS and WiFi localization functions and only requires the user to take a snapshot by pressing a single button.

The McSense Distributed Architecture

The McSense architecture includes three main distributed components that map the three phases of data collection and management into the mobile app, the data backend, and the task control console, as shown in Fig. 2. The *McSense mobile app* acts as an active stub deployed on the worker smartphone: it receives task offers, allows users to accept them, and provides the tools to complete them, possibly with a very simple graphical user interface (GUI) interaction, by accepting the corresponding task app that seamlessly configures all needed sensors available onboard. The task apps report their data to the McSense mobile app. In addition, the McSense mobile app collects data useful to profile users, devices, and, eventually, regions. When tasks are completed, it also uploads sensed data and local profiling results to the data backend.

The *McSense data backend* is the infrastructure component that receives data from the McSense mobile app, and stores and analyzes sensed data to evaluate task-related statistics, such as the number of workers receiving it, the number of tasks successfully completed, and the task completion time. The data backend uses all the available data to profile technical and social dimensions of users. Technical dimensions are system-level resources, such as number and type of sensors available on the smart phone, type of available network interfaces, and available battery. Social ones focus on geo-social behavior of workers in terms of location, such as visited areas and current position, and of social relationships, such as collocation and social ties. Worker and region profiles, incrementally refined as McSense collects more and more data about workers crossing an area, are very useful to tailor future task assignment and make it more effective.

The last component of the distributed architecture of McSense is the *task control console*,

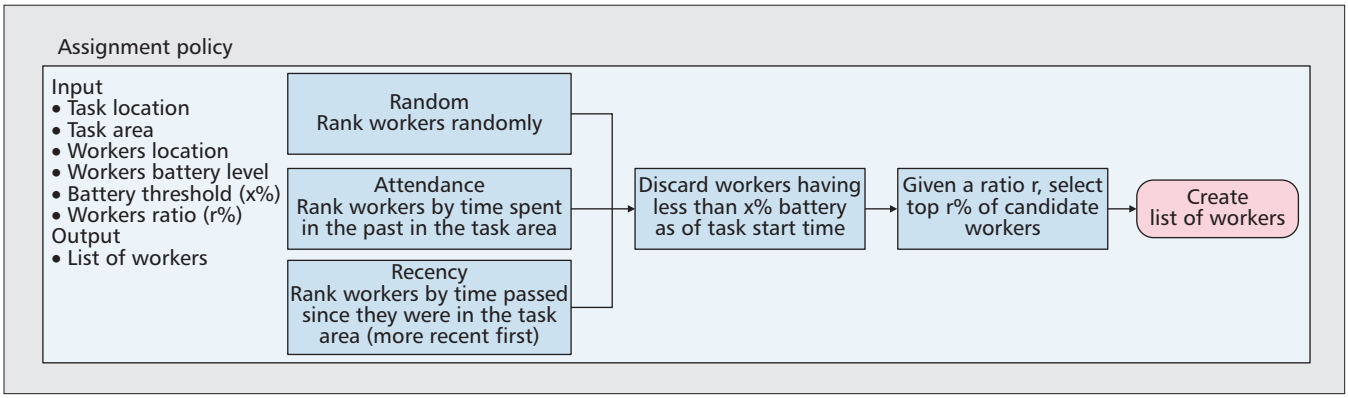

Figure 3. *McSense assignment and task execution emulation.*

used by city managers, who are the human operators who create and assign tasks. The console is a web application, and, apart from data visualization, it offers two main functions: task design and task assignment. The task design component gives immediate feedback to the operators by estimating expected performance for a geo-localized sensing task. In particular, by using statistics/profiles stored in the data backend and task-related data (location, area, and duration), the task design component evaluates the time required and the number of workers needed to complete the task with a desired probability. The task assignment component goal, instead, is to define the optimal set of workers to carry out the sensing task with the targeted objectives and success level. For example, assigning a task to a user typically spending much time in an area increases its probability of success, whereas assigning it to a user that has been there recently tends to decrease the time needed for task completion.

Task Assignment Policies

McSense deals with a continuously changing landscape: tasks are dynamically described and assigned, while workers roam free, following paths that unpredictably change dynamically, making it difficult to predict the best task-worker assignment schema. McSense evaluates the sensing task performance in terms of three main goal parameters: success ratio (i.e., ratio of successfully completed tasks over created ones), completion time (i.e., the interval between task start time and its successful conclusion), and number of required workers (i.e., number of workers to activate for task execution). At sensing task creation time, the city manager can either rely on default goal parameter values proposed by McSense or configure them depending on the experience stemming from previous task execution runs and specific application requirements. For instance, for a mission-critical sensing task, she will choose higher completion probability with higher number of workers, whereas a Smart City municipality with a low budget may be interested in minimizing the number of workers to reduce the associated incentive costs. Let us stress that these parameters also represent variables that are mutually dependent and deeply related to the assignment algorithm used to select workers.

McSense provides three different policies that, taking as inputs task properties (e.g., location, area, and duration) and user/region profiles, assign tasks to workers (Fig. 3): random policy, attendance policy, and recency policy. The *random policy* is not context aware and selects the set of workers to employ as a random group of available people in the whole city. The *attendance policy* exploits knowledge about the time previously spent by people in the task area; based on that indicator, it chooses and ranks potentially good workers. The *recency policy*, instead, favors and selects as workers the people who have more recently (with respect to the creation time of the sensing task) traversed the sensing task area. For each policy, McSense calculates ranked lists of candidate workers. In addition, all the implemented policies do not consider workers whose battery level is below a certain threshold, called *battery threshold*, at task starting time because we assume that these workers will be unlikely to run the sensing task to avoid battery exhaustion. The last input parameter, called *workers ratio*, is used to decide the percentage of candidate workers that will receive the task assignment, expressed as a percentage value in the [0.0, 1.0] range.

Given the above inputs and assignment policies, our McSense task assignment component evaluates sensing task performance (success ratio, completion time, and number of required workers). Moreover, because completed tasks are typically much fewer than all possible crowdsensing tasks of potential interest (e.g., some of them could relate to scarcely traversed areas), the task assignment component also runs prediction algorithms to decide how to effectively self-manage its behavior based on the sensing and profiling data harvested so far. The primary goal is to exploit the already collected sensing and profiling information to reasonably calculate accurate performance forecasts and estimations about potential future tasks in a relatively lightweight way. The prediction process consists of two phases and exploits the collected real-world dataset by dividing it temporally into two parts. The first phase considers the first part of the dataset as the past history and builds user/region profiles, such as ranked candidate worker lists. The second phase, instead, as better detailed in the following, creates synthetic ("virtual") future tasks and assigns them to candidate

workers selected according to the prediction algorithms; then it evaluates the performance of the predicted situations. Task designers take advantage of those forecasts, evaluated offline by our prediction process, to get fast feedback online about the expected performance of the sensing tasks they are defining.

With a closer view to technical details, McSense task prediction stochastically generates virtual tasks with different locations, areas, and durations. It emulates their execution based on the user profile stored in the data backend, in particular based on location traces: a virtual task is considered to be successfully completed if the location trace of a user comes in its range. To make the model more realistic, we also assume that workers whose device battery level is very low (e.g., less than 20 percent) will never execute any task, while if the battery level is high (e.g., 80 percent or more), they will always execute any task they can; the probability of executing a task increases linearly between 20 and 80 percent. In particular, given a task, its duration, and the set of workers to whom it has been assigned, the emulator looks for a worker within the task area by iterating worker position records in the task duration period. When it finds one, it stochastically evaluates whether the worker will be able to complete the task, and then updates the statistics about the policy under prediction by moving to the next worker location record. In the future, additional user profile parameters can be added, such as task completion rate or quality of data provided. We run the predicted situations for each assignment policy implemented in McSense; city managers can exploit these additional data to compare possible assignment policies and to choose the one that better suits their needs, whether it has a high chance of successful completion, minimizes the completion time, or minimizes the number of workers involved.

Implementation Insights and Experimental Results

We developed a prototype of the McSense platform to assess its architecture and the validity of its assumptions. The McSense mobile app has been implemented as an Android app that runs on the smart phones of available workers. The McSense data backend interface that receives data from the mobile app is a Java servlet application that uses PostgreSQL as a database for persistent data storage optimized with PostGIS to support fast geographic object storage and indexing. Finally, the Task control console is an Ajax web application based on Apache Spring at the server side.

As part of our experiment, the McSense mobile app was installed by 44 people who often visit the New Jersey Institute of Technology (NJIT) campus in Newark and decided to participate as potential workers. The experiment ran for two months, from February 2012 to April 2012. During that period, McSense executed many tasks that collected the following data: location, accelerometer readings, medium access control (MAC) addresses of nearby Bluetooth devices, BSSID, capabilities and signal power of WiFi hotspots, application usage and associated bandwidth usage, battery level, and photos. During the two-month study, we posted daily sensing tasks requests to all users; workers competed to get them, without enforcing any specific assignment policy, such as in the random policy.

The collected data from real scenarios of urban sensing have been used to quantitatively evaluate the emulated performance of our McSense assignment policies. In particular, we assigned tasks randomly placed on a 4 km × 4 km area centered on the NJIT campus. Tasks had an area with radius in the [100 m, 500 m] range and duration between one and seven days; according to our prediction process, we used the first part of the dataset (data sensed in the first month) to evaluate user/region profiles, while virtual tasks were temporally placed in the second month. In all our experiments we consider a worst case scenario in which tasks are randomly generated in the whole target 4 km × 4 km area. Figure 4 reports a limited selection of the many experimental results collected.[9] Fixing an area with radius 400 m and task duration equal to three days, Fig. 4a shows how the selected candidate workers ratio influences the success ratio of executed tasks: when the workers ratio is 1, the random policy has the highest success ratio, but employs all available workers (Fig. 4c). Attendance and recency policies (with slightly better performance results by recency) exhibited better results when the workers ratio is low because they carefully target task assignment only to users who have higher probability to accept executing the assignments. In addition, we collected analogous experimental results by ignoring battery level: as shown in Fig. 4b, those predictions overestimate the success ratio because they are based on the assumption that tasks will be executed by workers located within the task area independent of their battery levels, which typically does not hold in real-world scenarios. That result confirms the importance of using technical profiles to forecast workers and device behavior to improve task assignment performance.

Comparing Figs. 4a and 4c, we remark that the recency and attendance policies allow for very good performance even when using fairly low numbers of workers. Moreover, they allow careful control of the number of workers to involve in the particip Action, even for high worker ratio values: indeed, these two policies always use less than four workers because they are able to filter and keep in only good candidates; in other words, their ranked lists of workers are much shorter than the one for the random policy.

Finally, Fig. 4d shows the impact of task radius on completion time: as the radius increases, the time for completion decreases, because there is a larger number of workers who are likely to execute the task. However, the radius is more relevant for the random policy, mainly because the attendance and recency policies already have good performance for small radius values.

At the end of our experimental campaign, we ran a post-study survey about users' satisfaction in using McSense, which also allowed us to learn

McSense deals with a continuously changing landscape: tasks are dynamically described and assigned, while workers roam free, following paths that unpredictably change dynamically, making it difficult to predict the best task-worker assignment schema.

[9] *Additional information, tools, and experimental results about the McSense project are available at http://lia.deis.unibo.it/Research/McSense/.*

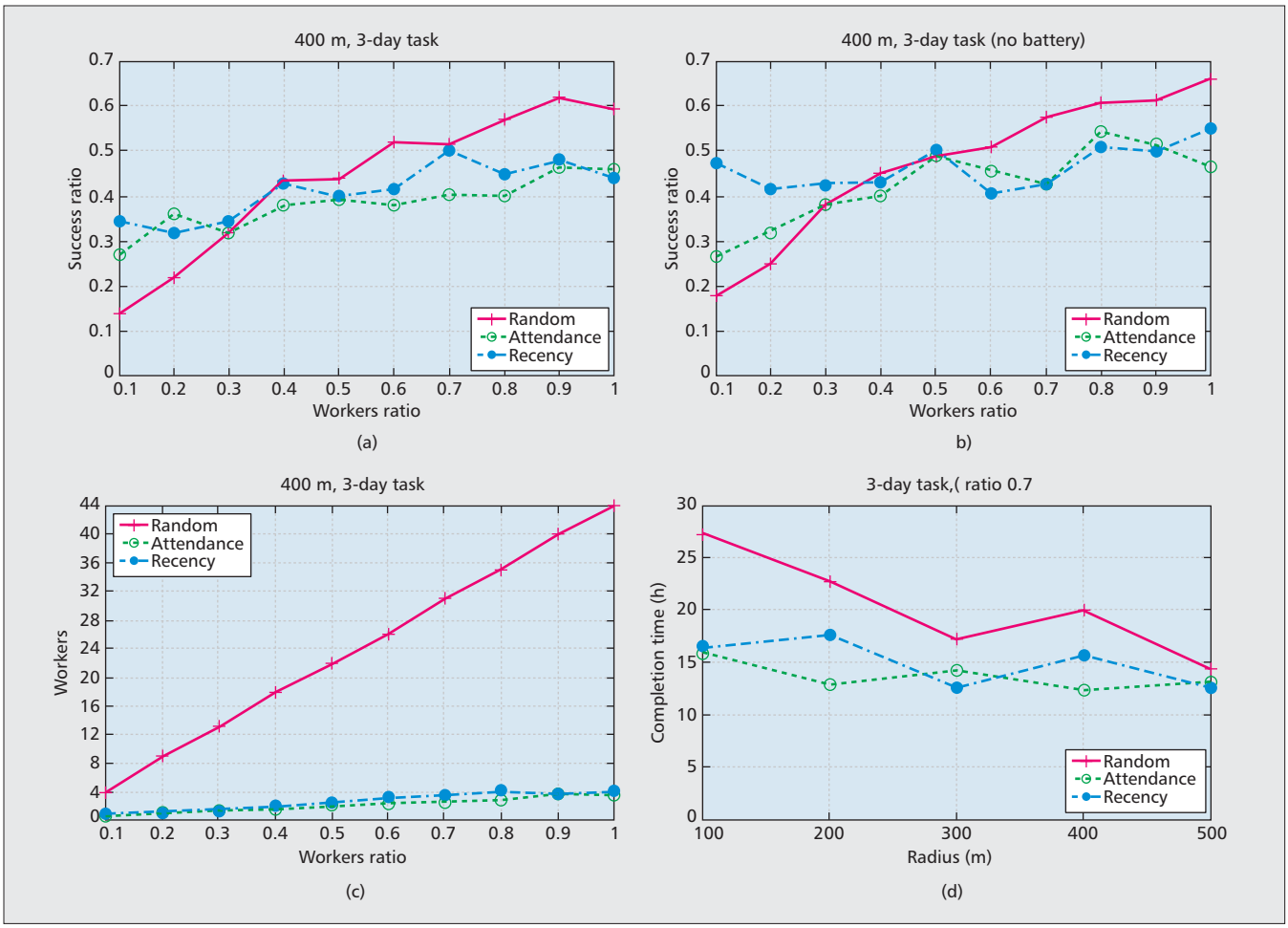

Figure 4. *Performance of different task assignment policies with regard to different task properties.*

the following lessons to be carefully considered for further refinement of our solution and prototype.

First, by observing user behavior and considering task completion vs. monetary incentive, McSense could perform a more intelligent task assignment that finds the right balance between budget and data quality. In our study, users prefer long-term automatic tasks (e.g., collect GPS and accelerometer readings for a day) to short-term manual tasks (e.g., take a photo at a given location), but once tasks were accepted, the completion rate was higher for manual tasks: that was caused in large part by certain power-hungry sensing applications. At the same time, while some users decided to abort the tasks when the power dropped under a certain level, we observed that many were willing to recharge their phone during the day due to the monetary incentive. Similarly, many users seem willing to trade off privacy (e.g., location privacy) for money. *Second, fault tolerance mechanisms are needed to cope with poor quality data or not receiving data at all from users who accepted a task.* Assigning more tasks than necessary (i.e., using a high workers ratio) could be a potential solution, although budget constraints could create impediments. In terms of the quality of the data collected in our study, we learned two interesting facts:

- As expected, a non-negligible percentage of users attempted to "fool" the system by providing fake data (i.e., for photo tasks).
- Users were tempted to accept high-priced short-term tasks with short duration, but quite often they were not able to complete them.

Third, user and region profiling can help with task decomposition decisions. For example, the decision on whether to split long-term tasks into several shorter tasks given to many users or keep the longer tasks and assign them to a few users is important. Based on our results, the completion rate is higher for longer tasks, but mobility traces and other parameters should also be considered when making decomposition decisions. *Fourth, the more the places are frequented by users, the better the task acceptance probability.* In fact, our results show that users are more willing to take tasks located at highly frequented points of interest on the campus, and user earnings increase with the amount of time spent on campus.

Conclusions

This article describes McSense, a full-featured geo-social crowdsensing platform for smart cities. Its distributed architecture and data analysis capabilities make it a flexible and reliable framework for leveraging sensing crowds in city-

wide deployment environments with expected high density of workers. The reported experimental results show that McSense assignment policies allow the preferred performance tradeoff to easily be tuned depending on specific task properties.

The encouraging results already obtained are stimulating further research efforts to extend McSense capabilities on both mobile and infrastructure sides. On the mobile side, we are developing collaborative sensing features in the McSense mobile app to enable ad hoc local interactions among collocated workers to reduce redundant data directly at the source. On the server side, we are enhancing the management capabilities of the McSense control console by providing more refined statistical profiles of workers, in particular to leverage social networking information (e.g., Facebook post activities) to foster the formation of worker groups in order to execute collaborative tasks that require multiple concurrent workers.

Acknowledgments

This research was supported in part by the National Science Foundation under Grant No. CNS 0831753, and also by CIRI, Center for ICT technology transfer of the University of Bologna.

References

[1] M. Mun *et al.*, "PEIR, the Personal Environmental Impact Report, as a Platform for Participatory Sensing Systems Research," *Proc. 7th Int'l. Conf. Mobile Systems, Applications, and Services*, 2009, pp. 55–68.
[2] T. Yan *et al.*, "mCrowd: A Platform for Mobile Crowdsourcing," *Proc. 7th ACM Conf. Embedded Networked Sensor Systems*, 2009, pp. 347–48.
[3] M.-R. Ra *et al.*, "Medusa: A Programming Framework for Crowd-Sensing Applications," *Proc. 10th Int'l. Conf. Mobile Systems, Applications, and Services*, 2012, pp. 337–50.
[4] M. S. Bernstein *et al.*, "Analytic Methods for Optimizing Realtime Crowdsourcing," *Collective Intelligence*, 2012.
[5] S. Ertekin, H. Hirsh, and C. Rudin, "Learning to Predict the Wisdom of Crowds," *Collective Intelligence 2012*, 2012.
[6] R. K. Ganti, Y. Fan, and L. Hui, "Mobile Crowdsensing: Current State and Future Challenges," *IEEE Commun. Mag.*, vol. 49, no. 11, 2011, pp. 32–39.
[7] M. Talasila, R. Curtmola, and C. Borcea, "Improving Location Reliability in Crowd Sensed Data with Minimal Efforts," *WMNC '13: Proc. 6th Joint IFIP/IEEE Wireless and Mobile Net. Conf.*, 2013.
[8] G. Cardone, A. Corradi, and L. Foschini, "Cross-Network Opportunistic Collection of Urgent Data in Wireless Sensor Networks," *Computer J.*, vol. 54, no. 12, Nov. 2011, pp. 1949–62.
[9] P. G. Ipeirotis, "Analyzing the Amazon Mechanical Turk Marketplace," *XRDS: Crossroads — The ACM Magazine for Students*, vol. 17, no. 2, 2010, pp. 16–21.

Biographies

Giuseppe Cardone [StM] (giuseppe.cardone@unibo.it) graduated from the University of Bologna, Italy, in computer science engineering in 2009. He is now a Ph.D. student in computer science engineering at the same university. His interests include performance and scalability issues of distributed systems, wireless sensor network management and integration, urban and mobile sensing, and power-aware middleware solutions.

Luca Foschini [M] (luca.foschini@unibo.it) graduated from the University of Bologna, Italy, where he received a Ph.D. degree in computer engineering in 2007. He is now an assistant professor of computer engineering at the University of Bologna. His interests include distributed systems and solutions for system and service management, management of cloud computing, context-aware session control and adaptive mobile services, and mobile crowdsensing.

Cristian Borcea [M] (borcea@njit.edu) is an associate professor in the New Jersey Institute of Technology's Department of Computer Science. His research interests include mobile computing, middleware for ubiquitous networked systems, vehicular networks, and sensor networks. He received his Ph.D. in computer science from Rutgers University. He is a member of ACM and Usenix.

Paolo Bellavista [SM] (paolo.bellavista@unibo.it) is an associate professor at the University of Bologna, Italy. His research activities span from mobile agent-based middleware and pervasive wireless computing to location/context-aware services and vehicular sensor networks, from big data adaptive stream processing to adaptive multimedia. He serves on the Editorial Boards of *IEEE TC*, *IEEE TNSM*, *IEEE TSC*, *Elsevier PMC*, and *Springer JNSM*.

Antonio Corradi [M] (antonio.corradi@unibo.it) graduated from the University of Bologna, Italy, and received an M.S. in electrical engineering from Cornell University, Ithaca, New York. He is a full professor of computer engineering at the University of Bologna. His research interests include distributed and parallel systems and solutions, middleware for pervasive and heterogeneous computing, infrastructure support for context-aware multimodal services, network management, and mobile agent platforms. He is a member of the ACM and the Italian Association for Computing (AICA).

Manoop Talasila [S] (mt57@njit.edu) graduated from Western Kentucky University, Bowling Green, where he received an M.S. degree in computer science in 2006. He is now a Ph.D. candidate in the Department of Computer Science at New Jersey Institute of Technology, Newark. His research interests include mobile computing, mobile crowdsensing, and secure location authentication of smart phones.

Reza Curtmola [M] (reza.curtmola@njit.edu) is an assistant professor in the Department of Computer Science at the New Jersey Institute of Technology. He holds a Ph.D. in computer science from The Johns Hopkins University. His research focuses on the security of cloud services, applied cryptography, and security aspects of wireless networks. He is the recipient of the NSF CAREER award. He is a member of the ACM.

The McSense distributed architecture and data analysis capabilities make it a flexible and reliable framework for leveraging sensing crowds in city-wide deployment environments with expected high density of workers.