Crowdsourcing with Smartphones

Georgios Chatzimilioudis*, Andreas Konstantinidis*, Christos Laoudias†, Demetrios Zeinalipour-Yazti*
*Dept. of Computer Science †KIOS Research Center
University of Cyprus, P.O. Box 20537, 1678 Nicosia, Cyprus

Abstract—Smartphones can unfold the full potential of crowdsourcing, allowing users to transparently contribute to complex and novel problem solving. We present the intrinsic characteristics of smartphones, a taxonomy that classifies the emerging field of mobile crowdsourcing and three in-house applications that optimize location-based search and similarity services over data generated by a crowd: (i) *SmartTrace*$^+$ enables similarity matching between a given pattern and the trajectories of smartphone users, keeping the target trajectories private; (ii) *Crowdcast* enables location-based interaction by efficiently calculating the k nearest neighbors for each user at all times; (iii) *SmartP2P* optimizes energy, time and recall of search in a mobile social community for objects generated by a crowd. We show how these applications can be deployed on *SmartLab*, a novel cloud of 40+ Android devices deployed at University of Cyprus, providing an open testbed that facilitates research and development of applications on smartphones at a massive scale.

1 Introduction

Crowdsourcing refers to a distributed problem-solving model in which a crowd of undefined size is engaged to solve a complex problem through an open call (see Figure 1). Crowdsourcing has still not fully penetrated the mobile workforce, which will eventually unfold the full potential of this new problem-solving model. This is true due to the smartphones' usage characteristics and unique features. Smartphones are in widespread, everyday use and are always connected. Therefore, they offer a great platform for *extending existing web-based crowdsourcing applications* to a larger contributing crowd, making contribution easier and omnipresent. Furthermore, the multi-sensing capabilities (geo-location, light, movement, audio and visual sensors, among others) of smartphones, provide a new variety of efficient means for opportunistic data collection *enabling new crowdsourcing applications*.

Crowdsourcing applications on smartphones can be classified into extensions of web-based applications or as new applications. The former class expands to users that do not have access to a conventional workstation and adds the dimension of real-time location-based information to the service. Instances of such applications are Gigwalk[1], Jana[2] and the work of Ledlie et al. [1]. The latter class includes applications for crowdsourced traffic monitoring (e.g., Waze[3]) and road traffic delay estimation (*VTrack* [2]); constructing fine-grained noise maps by letting users upload data captured by their smartphone microphone (*Ear-Phone* [3], *NoiseTube* [4]); identifying holes in streets by allowing users to share vibration and location data captured by their smartphone (*PotHole* [5]); location-based games with a purpose to collect geospatial data (*CityExplorer* [6]); leveraging mobile phones for collaborative traffic signal schedule advisory (*SignalGuru* [7]); and real-time fine-grained indoor localization services exploiting the Radio Signal Strength (RSS) of WiFi access points (*Airplace* [8]).

1. Gigwalk Inc., May 2012, http://www.gigwalk.com/
2. Jana, May 2012, http://www.jana.com/
3. Waze Ltd., April 2012, http://www.waze.com/

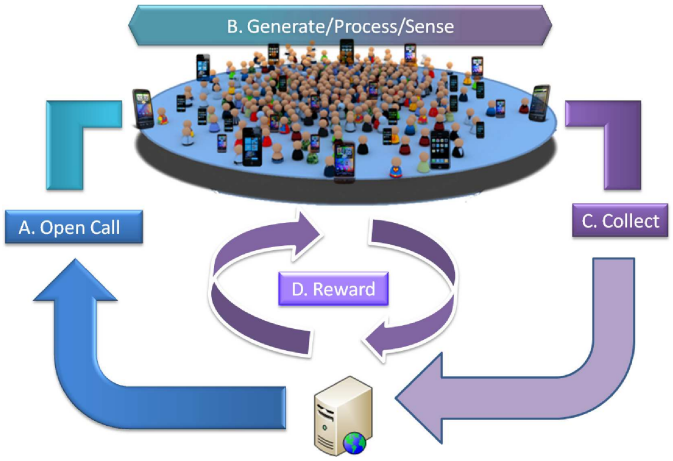


Fig. 1. Crowdsourcing with smartphones: A smartphone crowd is constantly moving and sensing providing large amounts of *opportunistic* data that enables new services and applications.

Another key characteristic of mobile crowdsourcing is whether the crowd's contribution is *participatory* or *opportunistic*. Generally speaking, computations performed by users and user generated data is the input for *participatory* crowdsourcing, while the input for *opportunistic* crowdsourcing is data generated from sensors and computations performed by the crowd's devices automatically — i.e., trajectory matching, positional triangulation. The classical crowdsourcing services on the web are participatory, since they require the active participation of the users. The crowdsourcing tasks of the second category are transparent to the user as they usually run in the background using the sensors to collect readings from the environment.

Further classifications can be adapted from crowdsourcing taxonomies proposed by Geiger et al. [12] and Quinn et al. [13]. Both studies recognize that the value of the input can lie

TABLE 1
Taxonomy of Mobile Crowdsourcing Applications

Applications	web-extend	involvement	data wisdom	contrib. quality	incentives	human skill	sensors	location
Gigwalk.com	✓	participatory	individual	heterogeneous	monetary	labor	camera	✓
Jana.com	✓	participatory	individual	heterogeneous	monetary	visual	×	✓
Crowd Translator [1]	✓	participatory	collective	homogeneous	service	visual	camera	×
Waze.com	×	both	collective	homogeneous	ethical/service	visual	×	✓
CityExplorer [6]	×	participatory	collective	homogeneous	entertainment	visual	camera	✓
VTrack [2]	×	opportunistic	collective	homogeneous	ethical/service	×	×	✓
SignalGuru [7]	×	opportunistic	collective	homogeneous	ethical/service	×	camera	✓
Ear-Phone [3]	×	opportunistic	collective	homogeneous	ethical	×	audio	✓
NoiseTube [4]	×	opportunistic	collective	homogeneous	ethical	×	audio	✓
PotHole [5]	×	opportunistic	collective	homogeneous	ethical	×	vibration	✓
AirPlace [8]	×	opportunistic	collective	homogeneous	service	×	×	✓
SmartTrace [9]	×	opportunistic	collective	homogeneous	service	×	×	✓
Crowdcast [10]	×	opportunistic	collective	homogeneous	service	×	×	✓
SmartP2P [11]	×	opportunistic	collective	homogeneous	service	×	×	✓

either in the *individual* or the *collective* contribution, where "*the crowdsourcing system strives to benefit from each contribution in isolation or from an emerging property resulting from the system of stimuli*", respectively. Furthermore, [12] divides applications regarding the contribution quality, which can be *homogeneous* or *heterogeneous*. In the former, each contribution has the same weight, whereas in the latter, each contribution is evaluated and can be compared to, compete against or complete other contributions. In [13], the incentives used for the crowd are also studied, which can be one or more from: *pay*, *altruism*, *enjoyment*, *reputation*, among others. We choose to make a more distinct division between *monetary*, *ethical*, *entertainment* and exchange of *service* incentives in our taxonomy. Finally, [13] further classifies applications according to the *human skill* that is exploited including visual recognition, language understanding and communication. Notice that human skill is only required in applications with *participatory* contribution.

In Table 1, we present a taxonomy of existing mobile crowdsourcing applications. In column 8, we present the sensors used for each application, since they are the key value-adding feature of smartphones. We then introduce the localization ability of smartphones in a separate column 9 to emphasize that the vast majority of crowdsourcing applications are location-aware.

The location-dependent crowdsourcing applications can further benefit from adding the temporal dimension to location data in order to exploit trajectory-related information. Similarly, they can benefit from inter-relations between location data, e.g., proximity information. It is essential to optimize and extend location-based search and similarity services. We present three smartphone applications: (i) *SmartTrace*$^+$ enables trajectory similarity functionalities without disclosing the user's trajectory; (ii) *Crowdcast* efficiently reports to each user their k-geographically nearest neighbors; and (iii) *SmartP2P* exploits proximity functionalities to optimize location-dependent objectives (energy, time and recall) of search tasks in a mobile social community. We finally present *SmartLab*, our experimental testbed of approximately 40+ Android smartphones deployed at the University of Cyprus, which is used to implement and evaluate smartphone applications at a massive scale.

TABLE 2
Energy profiling of a typical smartphone.

Basic Operation on Smartphone	**Power(mW=mJ/s)**
CPU Minimal use (just OS running)	35mW
CPU Standard use (light processing)	175mW
CPU Peak (heavy processing)	469mW
WiFi Idle (Connected)	34mW
WiFi Localization (avg/minute)	**125mW**
WiFi Peak (Uplink 123Kbps, -58dBm)	400mW
3G Localization (avg/minute)	**300mW**
3G Busy	900mW
GPS On (steady)	**275mW**
OLED Economy Mode	300mW
OLED Full Brightness	676mW

2 Issues and Characteristics of Crowdsourcing With Smartphones

Smartphones feature different Internet connection modalities that provide intermittent connectivity (e.g., WiFi, 2G/3G/4G), as well as peer-to-peer connection capabilities that provide connectivity to nodes in spatial proximity (e.g., Bluetooth, Portable WiFi or the new generation NFC). Notice that each of these connection modalities comes at different energy and data transfer rate characteristics. In particular, smartphones have typically energy-expensive communication mediums with asymmetric upload/download links, both in terms of bandwidth and energy consumption, with the upload link being the weaker link.

In Table 2, a set of real energy measurements from Android smartphones is presented. The values were obtained by running experiments on an HTC Desire smartphone with Android 2.3 and a Qualcomm QSD8250 ARMv7 1GHz processor, and using the benchmarking tools MobiPerf[4] and PowerTutor[5].

A general smartphone crowdsourcing architecture is shown in Figure 1, in which a problem is published to a mobile crowd in the form of an open call for solution. The crowd use their smartphones to contribute to the solution of the problem by generating, processing and/or sensing data of interest, which in turn are collected by the server. This results in a win-win situation where both the open call publisher and the mobile

4. MobiPerf, May 2012, http://www.mobiperf.com/
5. PowerTutor ver.1.4, May 2012, http://powertutor.org/

Fig. 2. ***SmartTrace***$^+$ **[9]:** The *SmartTrace*$^+$ project enables trace similarity search among smartphone users. It answers queries of the form *"Report the users that move similar to* Q*"*, where Q is some query trace. It optimizes such queries with respect to response time and energy consumption on the smartphones, without sharing the personal trajectories with the query processor. (a) Our system model. (b) Screenshots of the SmartTrace$^+$ client for outdoor environments with GPS and indoor environments with RSS signals.

crowd are rewarded.

Classical crowdsourcing applications are developed in a centralized or a decentralized manner. Centralized methods would ship the data generated and collected from the crowd to a server where the answer would be computed. Centralized methods are currently utilized by all social networking sites (such as Twitter, Youtube, Facebook, etc.) Continuously transferring data from the smartphone to the query processor can deplete the smartphone battery, increase user-perceived delays and quickly degrade the network health. In addition, it demands users to disclose their personal data with a central authority. On the other hand, decentralized methods would send the query to the smartphones, where all computations and communications would be performed locally. This approach might also perform poorly in terms of energy consumption if it invokes expensive computation tasks on all participants.

For location-dependent crowdsourcing applications, localization is usually either: i) GPS-only (fine-grained positioning, i.e., a few meters), ii) WiFi-only (semi-fine-grained, i.e., tens of meters), iii) Cellular-only (coarse-grained, i.e., tens to hundreds of meters). The latter two methods, which can be combined, require transmitting Cellular Tower and/or WiFi Received Signal Strength values over the Internet (via WiFi or 3G connection) to the localization server. GPS does not need to communicate any information over the Internet to a localization server. According to the power measurements of Table 2 and the above background information one can rank the methods from best to worse as follows [14], [2]:

- Energy: 1) WiFi-only, 2) 3G-only and GPS-only
- Accuracy: 1) GPS-only, 2) WiFi-only, 3) 3G-only
- Monetary: 1) WiFi and GPS, 2) 3G (assuming free WiFi access and a data plan for 3G).

In the following sections, we present three crowdsourcing application that successfully overcome the aforementioned issues using hybrid architectures.

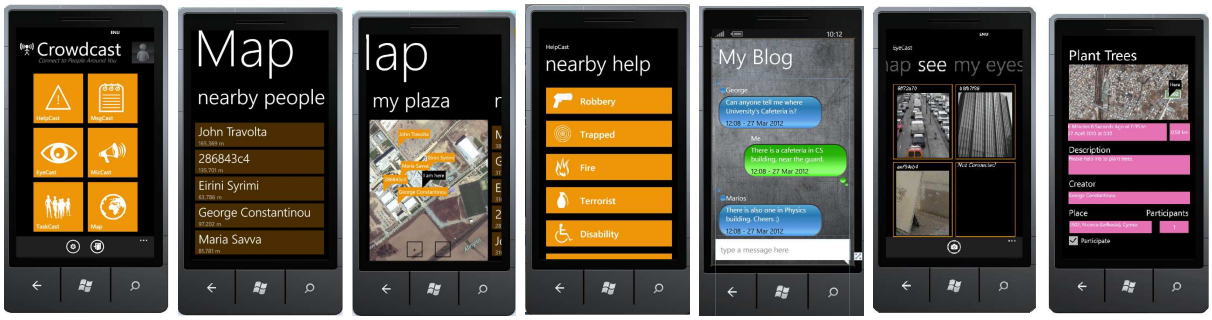
3 SmartTrace$^+$

The crowd of smartphone users can be asked to contribute in identifying mobility patterns or popularity of a given trajectory. Such a contribution can be utilized in large-scale urban and transit planning, transit rider information applications[6], shared ride applications[7,8], social networking applications on smartphones, habitant monitoring and others. Consider a transit authority that plans its bus routes and wants to know whether a specific route is taken by at least k users between 7:00 - 8:00 am. In such a scenario, one is interested in asking a crowd of users, in some target area, to participate with their local trace history through an open call. In particular, the users can opportunistically participate in the resolution of the query for monetary benefit or intellectual satisfaction, without disclosing their traces to the authority.

The *SmartTrace*$^+$ project[9] [9] enables trace similarity search among smartphone users and optimizes queries with respect to response time and energy consumption (see Figure 2). More importantly, *SmartTrace*$^+$ is privacy-aware since it does not share the user trajectories to the authority, rather it only returns matching scores.

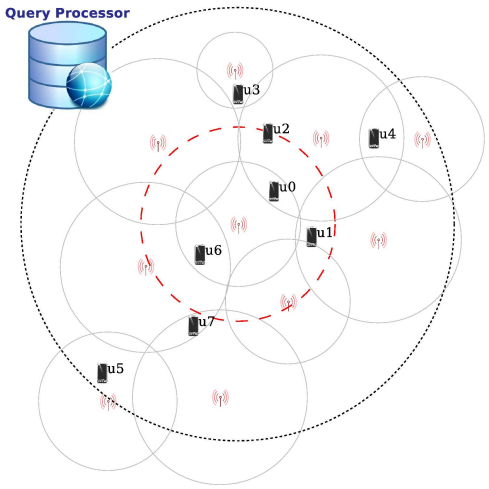
At a high level, our GUI enables the following functions: i) record traces on local storage and plot those on the screen for the outdoor case; ii) configure various logging and querying features; iii) connect to a *SmartTrace*$^+$ server and query the traces stored on other connected nodes; and iv) switch between *online* and *offline* mode to change between experimentation and real operation. We deploy instances of our real prototype system in Android over our *SmartLab* testbed described later in Section 6.

6. Tiramisu Transit project, April 2012, http://www.tiramisutransit.com/
7. Avego, April 2012, http:/www.avego.com/
8. RealyRides, April 2012, http://www.relayrides.com/
9. Software available at: http://smarttrace.cs.ucy.ac.cy/



(a)

(b)



(c)

Fig. 3. ***Crowdcast*[10]:** (a) Screenshots from an example application. *Crowdcast* efficiently connects you to your closest neighbors at all times, regardless of where you are and how far they are. Those neighbors can be shown in a list or on a map. On top of functionality a whole suit of applications have been developed: Helpcast to send out SOS beacons or disseminate natural disaster warnings, Msgcast to post local micro-blogging messages, Eyecast to extend the view on the urban environment using the cameras of one's neighbors, Miccast to post local vocal messages and warnings, Taskcast to post local tasks in your neighborhood as part of local crowdsourcing or organizing a charity event, etc. (b) The server has the overall picture of the user's whereabouts and can compute the k nearest neighbors for each user. (c) The *search space* of the cell is the big circle with the dotted outline. Any user inside this circle is a *kNN* candidate for any user inside the cell. The two nearest neighbors for u_0 are $\{u_1, u_2\}$. Similarly for the other users: $u_1 \rightarrow \{u_0, u_2\}$, $u_2 \rightarrow \{u_3, u_0\}$, $u_3 \rightarrow \{u_2, u_0\}$, $u_4 \rightarrow \{u_2, u_3\}$, $u_6 \rightarrow \{u_0, u_1\}$.

4 CROWDCAST

A highly desirable function for mobile devices is to continuously provide each user with its k geographically nearest neighbors in real-time. Such a service realizes the special operator that solves the *Continuous All k-Nearest Neighbor (CAkNN)* problem efficiently. Such an extended neighborhood "sensing" capability for mobile users enables several novel applications. For example, it would allow somebody to send out SOS beacons to its geographically closest neighbors and save them from a life-threatening situation enhancing public emergency services like *E9-1-1*[10]and *NG9-1-1*[11]. Other examples include applications where users engage in a location-based micro-blogging service that allows users to "follow" or "post-to" their neighborhood while being on the go. This would in effect facilitate the uptake of location-based social networks. Finally, novel social network analysis metrics based on the geographical neighborhood characteristics of a user can be calculated in real-time to enable new applications or services that are suited to the different roles of network users.

The *Crowdcast* framework [10] is proposed to answer such *CAkNN* queries efficiently based on the crowdsourcing of user locations (see Figure 3). The framework is: i) *Stateless* to cope with transient user populations and high mobility patterns; ii) *Parameter-free* to be invariant to parameters that are network-specific (such as cell size, capacity, etc.) and user distribution specific; iii) *Memory-resident*, since the dynamic nature of mobile user makes disk resident processing prohibitive; iv) *Specially designed* for highly *mobile and skewed* distribution environments performing equally well in downtown, suburban, or rural areas; v) *Fast and scalable*, in order to allow massive

10. Federal Communications Commission - Enhanced 911, Jan 2011, http://www.fcc.gov/pshs/services/911-services/enhanced911/

11. Department of Transportation: Intelligent Transportation Systems New Generation 911, Jan 2011, http://www.its.dot.gov/NG911/

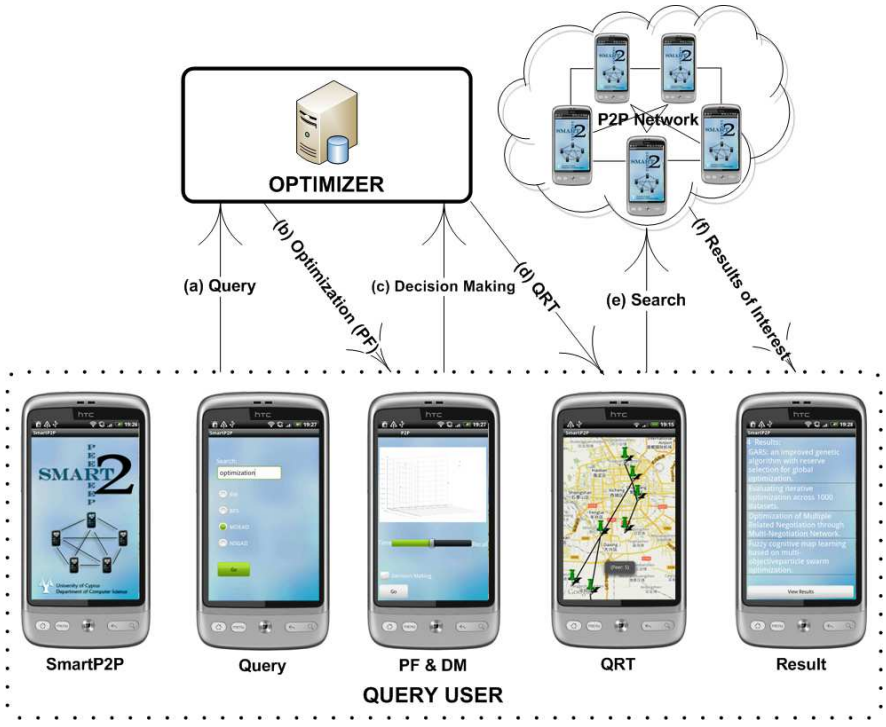


Fig. 4. ***SmartP2P* [11]:** The Framework Workflow and screenshots of the SmartP2P Client Side GUI in Android: (a) A user enters a keyword of interest to issue a query, which is subsequently optimized. (b) The answer is returned back to the user in a graphical format. (c) In the Decision Making process, the user studies all possible options and set its optimization preference with respect to recall and query execution time using a slide-bar. (d) The smartphone software fetches the selected optimized tree from a server and displays it along with respective annotations on a Google Maps interface. (e) Finally, the user searches the peer-to-peer network and (f) obtains a list of the results of interest.

deployment; and vi) *Infrastructure-ready* since it does not require any additional infrastructure or specialized hardware.

The efficiency of *Crowdcast* is mainly achieved due to a novel smart search space sharing technique. *Crowdcast* groups users of the same cell and uses the same search space for each group (*search space sharing*). Note that the search space includes all candidate kNN users that can reside in other nearby or even far-away cells. Using a novel data structure it builds the complete search space in a batch process by iterating over all user locations just once, performing minimal number of comparisons as seen in Figure 3(c).

Our experimental evaluation has shown that the build time is the bottleneck for our *Crowdcast* framework [10]. This drawback has been overcome by introducing a search space sharing technique. *Crowdcast*'s efficiency in search time is independent of k, scales with the number of users in realistic traffic scenarios and outperforms its competitors by at least an order of magnitude.

5 SmartP2P

SmartP2P [11] offers high performance search and data sharing over a crowd of mobile users participating in a smartphone social network. The main contribution of *SmartP2P* is to use location data made available by the crowd to optimize the search process. When a user invokes a search to find an object of interest, e.g., *"Pictures of street artists performing in Manhattan"*, our system favors querying a friend living in lower Manhattan rather than a person living in California as the former would have a higher probability having captured such pictures. Also, if the querying user had two friends both living in lower Manhattan, but with one being in spatial proximity to the querying user during the query (i.e., within a few meters), then he would have made a better choice for answering the query saving energy, network bandwidth, possible network traffic costs and transmission time.

SmartP2P can be used as a recommender system where the mobile social crowd generates instant information for certain places. A querying user can use *SmartP2P* to retrieve accurate and spatially close information about a place of interest (e.g., restaurant, pharmacy, hospital, police station). Furthermore, *SmartP2P* can be utilized for publish/subscribe services, i.e., the crowd (subscribers) shares its interests and preferences about a topic by subscribing to the server (broker) and a query user (publisher) posts and forwards messages to the interested users only.

In *SmartP2P*, with every new query the user first downloads a *Query Routing Tree (QRT)* from a server. This tree is tuned to optimize multiple objectives concurrently during searches in a smartphone P2P network: (i) minimize energy consumption during search; (ii) minimize the query response time in conducting the search; and (iii) maximize the recall rate of the user query. Due to those conflicting objectives, there is not a single routing tree that optimizes all objectives

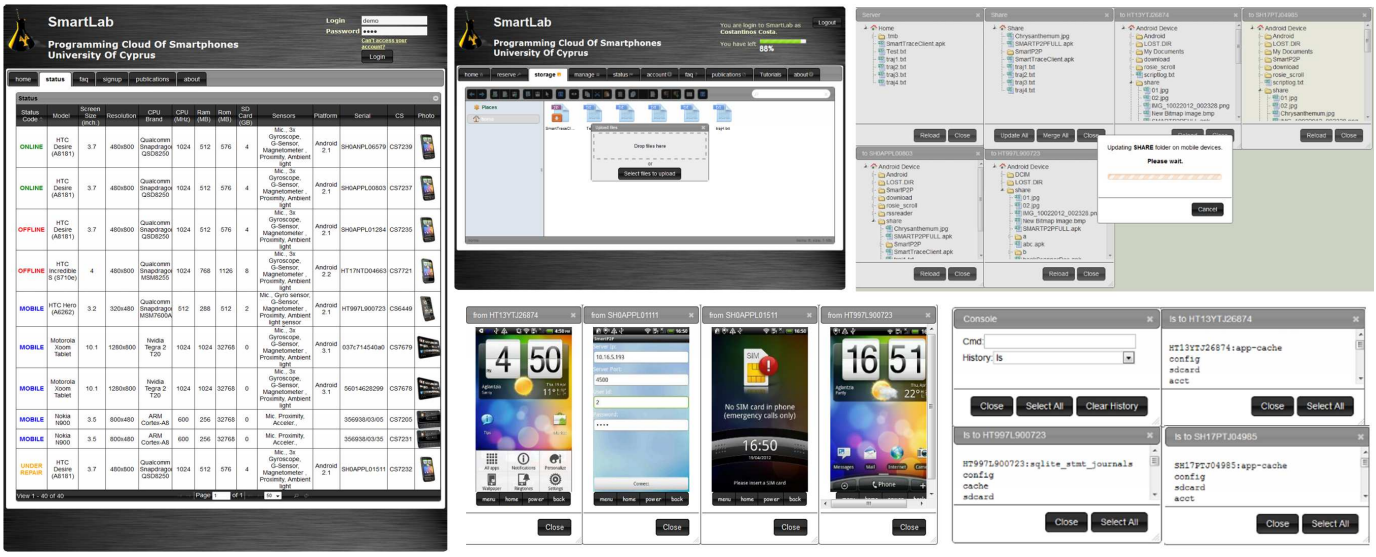


Fig. 5. ***SmartLab* [15]:** An innovative programming cloud of approximately 40+ real Android smartphones, plus numerous emulated devices, deployed at the Department of Computer Science building at the University of Cyprus.

simultaneously. A *Decision Maker (DM)* needs a set of optimal solutions, commonly known as the *Pareto Front (PF)*, in the field of *Multi-Objective Optimization (MOO)*, to choose the final QRT to be used. In addition, *SmartP2P* improves the centralized algorithm in that it does not transmit user's data to a central authority. Users keep their own generated data in-situ for data-disclosure and performance reasons.

The *SmartP2P* framework is composed of three main phases as illustrated in Figure 4: i) the optimization; ii) the decision making; and iii) the P2P search. In the optimization phase, the optimizer can be any MOO evolutionary approach that utilizes the information of the registered crowd to obtain a diverse and high quality set of near-optimal solutions (i.e., PF). These are then forwarded to the decision making phase, in which the user (i.e., DM) picks the final QRT based on instant network requirements and preferences. The selected routing tree is then utilized by the P2P search approach to retrieve the answers.

SmartP2P has been implemented on a prototype system for the Android OS and was tested on our *SmartLab* testbed that will be introduced in Section 6. Experimental evaluation reveals that this framework yields high query recall rates of 95%, with one order of magnitude less time and two orders of magnitude less energy than its competitors.

6 SmartLab

Experimenting with many devices simultaneously is often a tedious process. Therefore, we have implemented the *SmartLab*[12] testbed [15] in order to implement and evaluate smartphone applications at a massive scale.

SmartLab is an innovative programming cloud of approximately 40+ real Android smartphones, plus numerous emulated devices, deployed at the Department of Computer Science building at the University of Cyprus. SmartLab provides an open, permanent testbed for development and testing of smartphone applications via an intuitive web-based interface. Registered users can upload and install Android executables (APKs) on a number of Android smartphones, capture their output, reboot the devices, issue commands and many other exciting features (see Figure 5). *SmartLab* supports four modes of user interaction with the remote devices: i) *Remote Control Terminals (RCT)*, a web-based remote screen terminal for Android developed in-house on Ajax that mimics touchscreen clicks and gestures among other functionalities; ii) *Remote Shells (RS)*, a web-based shell developed in-house on Ajax that enables issuing a wide variety of UNIX commands to the Android devices; iii) *Remote Scripting Environment (RSE)*, which allows users to author Android MonkeyRunner automation scripts (written in python) and upload them to the devices to perform automated tasks; and iv) *Remote Debug Tools (RDT)*, which provide web-based debugging extensions to the Android Debug Bridge (ADB). SmartLab aims to facilitate research in smartphone network programming environments, communication protocols, system design, and crowdsourcing applications.

12. Available at: http://smartlab.cs.ucy.ac.cy/

7 Summary and Future Work

In this paper, we have presented the emerging field of crowdsourcing on smartphones. We expect that crowdsourcing with smartphones will evolve rapidly in the future. Smartphone networks comprise a new computation system that involves the joint efforts of both computers and humans. The unique data generated by the smartphone sensors and the crowd's constant movement, will enable new challenging applications and the solution of harder problems than crowds can currently accomplish. The focus of future efforts in this area lies in the collection of specialized location-related data and the better task assignment to match the particular expertise and interests of the smartphone users. Our experiences with three different crowdsourcing platforms for smartphones have shown that

energy consumption, privacy preservation and application performance might be the building blocks of future applications in this domain.

The overview and taxonomy of the early works on mobile crowdsourcing introduced in this article, present a glimpse of the possible research frontiers that will emerge. Especially extending the location-awareness, which smartphones offer, will give rise to new applications and services that will engage the world's citizens enabling crowdsourced quality of life.

Acknowledgements The last author's startup grant, funded by the University of Cyprus, partially supported this research.

REFERENCES

[1] J. Ledlie, B. Odero, E. Minkov, I. Kiss, and J. Polifroni, "Crowd translator: on building localized speech recognizers through micropayments," *ACM SIGOPS'10 Operating Systems Review*.

[2] A. Thiagarajan, L. Ravindranath, K. LaCurts, S. Madden, H. Balakrishnan, S. Toledo, and J. Eriksson, "Vtrack: accurate, energy-aware road traffic delay estimation using mobile phones," in *7th Conference on Embedded Networked Sensor Systems (SenSys'09)*.

[3] R. K. Rana, C.-T. Chou, S. Kanhere, N. Bulusu, and W. Hu, "Ear-phone: an end-to-end participatory urban noise mapping system," in *9th International Conference on Information Processing in Sensor Networks (IPSN'10)*.

[4] M. Stevens and E. D. Hondt, "Crowdsourcing of pollution data using smartphones," *Ubiquitous Computing (UbiComp'10)*.

[5] J. Eriksson, L. Girod, B. Hull, R. Newton, S. Madden, and H. Balakrishnan, "The pothole patrol: using a mobile sensor network for road surface monitoring," in *6th international conference on Mobile systems (MobiSys'08)*.

[6] S. Matyas, C. Matyas, C. Schlieder, P. Kiefer, H. Mitarai, and M. Kamata, "Designing location-based mobile games with a purpose: collecting geospatial data with cityexplorer," in *International Conference on Advances in Computer Entertainment Technology*, 2008.

[7] E. Koukoumidis, L.-S. Peh, and M. R. Martonosi, "Signalguru: leveraging mobile phones for collaborative traffic signal schedule advisory," in *9th International conference on Mobile Systems, Applications, and Services (MobiSys'11)*.

[8] C. Laoudias, G. Constantinou, M. Constantinides, S. Nicolaou, D. Zeinalipour-Yazti, and C. G. Panayiotou, "The airplace indoor positioning platform for android smartphones," in *13th International Conference on Mobile Data Management (MDM'12)*.

[9] D. Zeinalipour-Yazti, C. Laoudias, C. Costa, M. Vlachos, M. I. Andreou, and D. Gunopulos, "Crowdsourced trajectory similarity with smartphones," *IEEE Transactions on Knowledge and Data Engineering (TKDE'12)*.

[10] G. Chatzimilioudis, D. Zeinalipour-Yazti, W.-C. Lee, and M. D. Dikaiakos, "Continuous all k-nearest neighbor querying in smartphone networks," in *13th International Conference on Mobile Data Management (MDM'12)*.

[11] A. Konstantinidis, D. Zeinalipour-Yazti, P. Andreou, and G. Samaras, "Multi-objective query optimization in smartphone social networks," in *12th International Conference on Mobile Data Management (MDM'11)*.

[12] D. Geiger, M. Rosemann, and E. Fielt, "Crowdsourcing information systems : a systems theory perspective," in *22nd Australasian Conference on Information Systems (ACIS'11)*.

[13] A. J. Quinn and B. B. Bederson, "Human computation: a survey and taxonomy of a growing field," in *Annual Conference on Human Factors in Computing Systems (CHI'11)*.

[14] K. Lin, A. Kansal, D. Lymberopoulos, and F. Zhao, "Energy-accuracy trade-off for continuous mobile device location," in *8th international conference on Mobile Systems, Applications, and Services (MobiSys'10)*.

[15] A. Konstantinidis, C. Costa, G. Larkou, and D. Zeinalipour-Yazti, "Demo: A programming cloud of smartphones," in *10th International Conference on Mobile Systems, Applications, and Services (MobiSys'12)*.