TASKer: Behavioral Insights via Campus-based Experimental Mobile Crowd-sourcing

Thivya Kandappu[†], Nikita Jaiman[†], Randy Tandriansyah[†], Archan Misra[†], Shih-Fen Cheng[†], Cen Chen[†], Hoong Chuin Lau[†], Deepthi Chander[∗], Koustuv Dasgupta[∗]
† School of Information Systems, Singapore Management University
∗ Xerox Research Centre, India
{thivyak@, nikitaj@, rtdaratan@, archanm@, sfcheng@, cenchen.2012@phdis., hclau@}smu.edu.sg,
{deepthi.chander, koustuv.dasgupta}@xerox.com

ABSTRACT
While mobile crowd-sourcing has become a game-changer for many urban operations, such as last mile logistics and municipal monitoring, we believe that the design of such crowd-sourcing strategies must better accommodate the real-world behavioral preferences and characteristics of users. To provide a real-world testbed to study the impact of novel mobile crowd-sourcing strategies, we have designed, developed and experimented with a real-world mobile crowd-tasking platform on the SMU campus, called *TA$Ker*. We enhanced the *TA$Ker* platform to support several new features (e.g., task bundling, differential pricing and cheating analytics) and experimentally investigated these features via a two-month deployment of *TA$Ker*, involving 900 real users on the SMU campus who performed over 30,000 tasks. Our studies (i) show the benefits of bundling tasks as a combined package, (ii) reveal the effectiveness of differential pricing strategies and (iii) illustrate key aspects of cheating (false reporting) behavior observed among workers.

ACM Classification Keywords
H.5.m. Information Interfaces and Presentation (e.g. HCI): Miscellaneous.

Author Keywords
crowd-sourcing, context-aware, empirical study; user behaviour

INTRODUCTION
Mobile crowd-sourcing, involving the use of a volunteer workforce to perform location-specific micro-tasks, has become a powerful paradigm for a variety of urban services, especially in densely-crowded cities such as Singapore. Such a participatory model of task execution significantly reduces operational cost and response latency, and is used for services such as *last-mile logistics* (e-commerce package delivery services such as Amazon Flex[1] in the US and Courex & RocketUncle in Singapore), *retail auditing* (checking in-store product placement and stock levels) and *municipal monitoring* (reporting problems related to garbage, potholes, broken streetlights via applications such as the OneService™App in Singapore).

[1]http://flex.amazon.com/

In spite of its promise, mobile crowd-sourcing faces several operational challenges including: (i) the well-known problem of worker skew (e.g., Musthag & Ganesan [14] showed that 80% of tasks are performed by a small minority of workers who experience significantly longer detours); (ii) the problem of workers often having low productivity (by often spending longer than expected travel time to complete tasks) and (iii) the problem of veracity or *truth discovery* (as crowd-sourced reports can be biased or fraudulent and must thus be carefully corroborated). A variety of *computational* or algorithmic techniques have been suggested to tackle these challenges—e.g., to introduce "task bundling" as a way to amortize the travel detour overhead over multiple collocated tasks (initially pointed out in [14], and recently demonstrated in [11]), or by employing aggregation-based techniques on user-reported data to eliminate fraudulent reports and outliers (e.g., [10]). However, for large-scale adoption of mobile crowd-sourcing, we believe that such algorithmic designs must accommodate the *real-world human behavioral choices, preferences and artefacts* of the workers. For example, most models of differential task pricing in crowd-sourcing employ elaborate game-theoretic solutions that involve workers behaving as rational, non-cooperating agents, whereas the non-rationality of human choices is well documented.

To specifically study, derive and leverage upon such human behavioral insights, we have designed and deployed *TA$Ker*, an *experimental* campus-scale crowd-sourcing platform (with a client-side mobile App for both Android and iOS devices) on the Singapore Management University (SMU) campus. All tasks in *TA$Ker* presently involve some form of reporting of the *state of on-campus* resources, such as "length of queue in the food court", "availability of a soda brand in a particular vending machine" and "cleanliness level of a specific toilet". The deployment of an initial, limited version of *TA$Ker* (with 80 participants over a trial period of 4 weeks) was previously described in [6], and helped validate a key insight namely, that a centrally-coordinated, trajectory-aware model of crowd-sourcing [2], that recommended tasks to users so as to minimize the detour from their expected future trajectory, was more effective than the traditional pull-based model

DOI: http://dx.doi.org/10.1145/2971648.2971690

(where workers simply select from the corpus of available tasks in a completely de-centralized fashion).

In this paper, we report on the insights gained from a *significantly-expanded* deployment of *TA$Ker* (specifically *TA$Kerv3*, version 3 of the *TA$Ker* App[2]), with specific focus on key aspects of the human behavioral response to different parameters of mobile crowd-sourcing. More specifically, we use a significantly larger study (involving 900 student users, who performed 30,000+ tasks over a 2 month reporting period) to study three key new aspects of crowd-sourcing operation:

- *Bundling*—this refers to task aggregation, where workers must perform all tasks in a set to obtain payment. Our studies help us to understand the worker response to the tradeoff in aggregated vs. unit pricing of tasks, and to understand whether and how bundling helps improve the productivity of choices made by the workers.
- *Pricing*—whereby the prices for tasks at different campus locations were set differentially based on different strategies. The studies reveal how worker choices (especially the tradeoff between higher rewards vs. higher travel detours) affect the overall spatial skew in the task completion rate.
- *Cheating*—this refers to the tendency of workers to report on the location-specific tasks without actually visiting the location. Our studies help establish additional behavioral and temporal factors that belie the simplistic notion of credibility/trustworthiness being purely an intrinsic property of an individual worker.

TA$Ker itself leverages upon SMU's LiveLabs testbed infrastructure [12], which provides Wi-Fi based location tracking of all Wi-Fi enabled mobile devices on the SMU campus. To help support this expanded set of behavioral studies, *TA$Ker* was also significantly enhanced to better predict the movement trajectory prediction of workers on the campus. We shall also describe these enhancements, and explain how *TA$Ker* was able to maintain the superiority of the centrally-coordinated task recommendation approach (over the alternative pull-based approach) even in the face of non-negligible errors in the underlying location data and characteristic randomness in the movement of students on the SMU campus.

Key contributions: Our paper uses an updated version of *TA$Ker*, containing support for (i) tasks with different (start, end) intervals, (ii) tasks specified either individually or as fixed-size bundles, and (iii) differential task pricing. Using this newer version, we make the following contributions:

- *Effectiveness of Differential Pricing & Bundling Strategies:* We experimentally study both novel task pricing and bundling strategies. For pricing, we first show that flat pricing of tasks does indeed result in spatial skews for completed tasks, mirroring the skew in the spatial distribution of students across the campus. We then demonstrate that a simple *inverse density* based pricing strategy, where task rewards were defined to be inversely proportional to the popularity (number of occupants) of different university locations, not only effectively counters such spatial skews (we get a 5-fold reduction in the variance of the task completion rate) but also significantly increases the fair sharing of rewards *among workers*.
  For bundling, we show that workers prefer bundled tasks (i.e., a set of tasks that must all be performed to earn the payment specified for the task bundle), completing bundled tasks with an average completion rate that is $\sim$ 19% higher than atomic tasks, even though both types of tasks are almost equally offered. Moreover, this preference for bundled tasks is manifested even though the per-task reward (total bundle reward divided by the number of tasks in the bundle) is 75% of that for an individual atomic task. This preference can be explained by observing that task bundling improved worker *productivity*, with a worker earning 77% ($0.20) more per minute of additional detour, compared to individual tasks.
- *Cheating Characteristics:* We empirically demonstrate that the tendency to cheat (i.e., generate false crowd-tasking reports without visiting the specified task location) has both intrinsic (individual personality-based) and contextual (based on properties of the tasks) factors. In particular, (a) workers tended to generate a larger proportion of false reports for the latter tasks executed in a bundle (as the potential loss of rewards gets larger with every task performed accurately) and (b) tighter time windows for tasks lead to disproportionately more cheating (80% of cheating occurred on tasks with execution time windows less than 90 minutes, even though these tasks constituted only 50% of the overall task pool). Accordingly, truth discovery mechanisms in crowd-sourcing need to be modified to take into account such dependence on the task performance sequence and associated execution time windows.
- *Practical Trajectory-Aware Recommendation for Time-Constrained Tasks:* By deploying the *TA$Ker* platform, we also show how its trajectory-aware task recommendation engine can accurately predict worker trajectories (even though the location tracking system is not fine-grained) via the estimation of key *reference locations*. We demonstrate that such a practical recommendation strategy is *effective*, even when task execution time windows were limited to 30 minutes: it can assign tasks to users with an average of only 3 mins detour, compared to alternative strategies that generate detours of 6 mins.

We emphasize again that *TA$Ker* is an active, long-running experimental platform, involving thousands of student participants. While it allows us to study novel crowd-sourcing strategies only at campus-scale, we anticipate that the behavioral insights will help improve the design of future city-scale mobile crowd-sourcing services.

TA$KER DEPLOYMENT & STUDY DETAILS

Before diving into the details of the experimental studies, we first outline the key relevant characteristics of *TA$Ker*, a campus-based mobile crowd-tasking platform where consenting student workers use the *TA$Ker* mobile application to retrieve and perform a wide variety of on-campus, location-based, *reporting* tasks. As a quick summary, the SMU cam-

[2]For the rest of this paper, *TA$Ker* will refer to the *TA$Kerv3* version of the platform

pus consists of 4 five-storey academic buildings and a library, connected via an underground concourse, and is used by a student population of roughly 9000 students affiliated with 6 schools.

Figure 1 illustrates the overall functional architecture of *TA$Ker*. As described in [6], in addition to the *TA$Ker* mobile App, the *TA$Ker* backend system includes the following key components: (a) *Route Predictor*, that predicts an individual user's movement trajectory based on her historical movement traces; (b) *Task Recommender*, that suggests tasks (from the overall pool of available tasks) which best match (i.e., minimize the additional detour overhead) the predicted trajectory of an individual worker; (c) *Task Management Portal*, that allows *TA$Ker* administrators to create, modify and monitor the set of available tasks (as well as set task prices or specify task bundles), (d) *Database & Results Validator*, which stores the responses received from the users and validates the integrity of the responses by comparing the location of the specified task and the locations visited by the worker, and (e) *Results Analyzer*, which analyzes the contents of executed tasks to provide deeper understanding of the behavioral interaction by crowd-workers with the *TA$Ker* App.

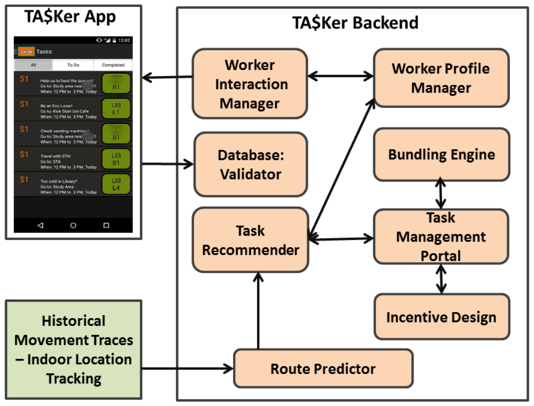


Figure 1. *TA$Ker* framework - architecture

Task Types & Execution Windows

As mentioned before, *TA$Ker* is primarily used as a means to crowd-source reports on the status of various locations and events on campus. Each such task is associated with a *specific validity period* and a time window–i.e., an interval (T_s, T_e), such that the task is required to be executed after time T_s and prior to time T_e. For the study period considered in this paper, tasks were assigned to one of three different 3-hour time windows: (a) morning (9am, 12noon); (b) noon (12noon, 3pm), and (c) afternoon (3pm, 6pm). These three time windows were roughly aligned to the 3 distinct undergraduate course times, and thus can be expected to correspond to the schedules of individual students. A task validity period can span any duration more than 15 minutes but less than 3 hours within the corresponding time window.

To provide an accurate report, an individual worker is thus required to visit the task's specified location within the stipulated task validity period (T_s, T_e). Tasks defined over a specific period would *expire* once the subsequent time window starts (and thus no longer be visible on the *TA$Ker* mobile App, irrespective of whether the worker had accepted the task assignment or not).

Task Types: Reflecting the wide variety of tasks, and the natural modality for reporting such tasks, *TA$Ker* associates each task with one of four "task types": (a) *Discrete Valued Multiple Choice*, where the user has to select an option from a predefined set of values. For example, *Do the green bins near "Pick and Bite" need to be emptied? Yes—No*; (b) *Counting based*, where the user has to provide an input by selecting from a pre-specified set of numerical values. For example, *How many people are queuing at the Western food stall* ; (c) *Picture based*, where the user has to upload task-relevant images. For example, a task to *check the level of crowdedness at the student recreational area* would require the worker to take a picture of that public space; (d) *Free text based*, where the user provides free-form textual responses. For example, *Tell us the price of Toni and Guy shampoo at Watsons*.

Experiment Study Details

To support the studies performed in this paper, *TA$Ker* was deployed over a 8 week period, September 23 - November 20, 2015 (with tasks being available only on working weekdays). During this period, a total of 900 students opted to participate in the study (which had been previously approved by SMU's Institutional Review Board (IRB)). The users were divided randomly into two equal sized-groups: (a) the "push" group were able to view only a smaller set of tasks, selected by the Task Recommender component, that were best aligned to their predicted movement over the corresponding 3 hour window, whereas (b) the "pull" group were able to view all available tasks and had to select an appropriate subset from this entire list. (The only exception was a 2 week period, explained shortly, when differential pricing was activated.)

Moreover, to ensure fairness (and to avoid one undesirable possibility of students skipping classes to perform available tasks and earn money), each worker (in both the "push" and "pull" groups) could only execute *Max* tasks in any 3 hour window (*Max* was set to 3 for all the weeks except week 5 & 6 where it was increased to 6).

As part of the study, we trialled 3 different experiments: (a) *Exp 1: Incentive mechanisms*, which studied two different differential location-dependent pricing strategies for tasks (based on visitor density and historical task acceptance rates), (b) *Exp 2: Task Bundling*, where users were offered the chance to perform individual vs. bundled tasks, and thus a chance to amortize their mobility cost, but with lower per-task reward, and (c) *Exp 3: Preference based recommendation*, which aims to improve the experience of the participants by recommending the tasks not only in their predicted trajectory but also as per their preferred types of tasks (e.g., multiple choice question, free-text based, counting based and photo taking tasks). The details of the conducted experiments are listed in Table. 1, and required us to make the following careful changes to the *TA$Ker* server's functionality:

- During the 2-week period when we experimented with differential pricing, we deliberately stopped generating proac-

Table 1. Summary of experimental study.

Week	Experiment	Max - Task allowance per timewindow	No. of registered users	No. of active users	No. of responses received
1	Flat pricing	3	170	85	472
2	Flat Pricing	3	290	156	1320
3	Incentive - Algo.I	3	613	338	3633
4	Incentive - Algo.II	3	779	453	7068
5	Bundling	6	817	293	5355
6	Bundling	6	835	278	7083
7	User preference recommendation	3	850	254	3507
8	User preference recommendation	3	900	224	3831

tive recommendations for the push group. This is due to the fact that our recommendation engine aims to optimize the total earnings of all the users; hence, if activated, it would preferentially recommend only the higher priced tasks to the users of the push group, effectively biasing the trials.

- During week 5 & 6 (the period of the "bundling studies"), we tweaked the recommendation engine to preferentially pick bundled tasks (in spite of lower per-task reward) for recommendations by adjusting the total detour cost for visiting all locations within the task bundle, so as to increase the reward earned per-task-per-minute of detour. Throughout the trial, the bundles we offered comprised tasks belonging to the same floor level of a building. The size of the bundle is fixed (4 tasks) and the per-task reward value is lower than those of the atomic ones. Each user was allowed to perform 6 tasks (either a combination of 1 bundle and 2 atomic tasks or just 6 atomic tasks) in a time window.
- During weeks 7 & 8 (the study of "user-preference based recommendations"), we first computed task type preference scores (per worker) for each of the 4 task *types*, by dividing the number of tasks completed for the type by the total number of tasks completed by the worker. The *Task Recommender* was then modified to weigh each task incentive according to its task type preference score (thereby preferentially recommending task types that the user prefers).

MODELING ROUTINE DAILY TRIPS OF STUDENTS

We applied a state-of-the-art route prediction algorithm described in [6], on location traces of all the participating students, to predict their expected trajectories in a time window. The location data is collected through the LiveLabs indoor location service [7], which uses server-side Wi-Fi fingerprinting techniques to track the on-campus location of all persons (i.e., their mobile device), as long as their Wi-Fi interface is enabled. Due to well-known limitations of such server-side location tracking, the location service currently offers medium-grained granularity (errors are typically $\pm 6-8$ meters) and latency (the period between successive location updates from individual devices is around 2 - 4 minutes).

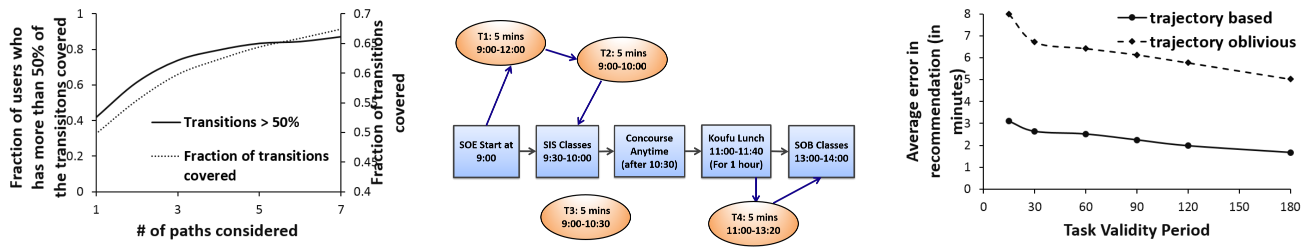
The route prediction algorithm first transforms the raw data to routes. It extracts *reference locations*, in which users stays more in a given time segment (in this paper we consider 30 minute time segments); it then formulates the movement pattern of a user in a given time window as a *transition graph* and finds the best probable k routes (represented as a sequence of reference locations), based on the past traces of movement trajectory of the worker. The real walk paths are generated by applying standard shortest path algorithm for all connected reference location pairs. In Fig. 2(a), we plot how k value affects the number of transitions covered per user basis (in primary Y-axis, we plot the fraction of users who have more than 50% of the transition coverage for various number of paths considered). The same figure also depicts how the k value affects the transitions covered in total for the whole system (in the secondary Y-axis we plot the fraction of total transition coverage of the system). The value of k in our experiments is set to 5, as the top 5 routes can explain more than 50% of all transitions for 85% of users.

TRAJECTORY-AWARE RECOMMENDATION ENGINE

As mentioned previously, to support the experimental studies, half of the workers were allocated to the "push" category, where the *TA\$Ker* backend generates task recommendations to individual workers, based on a worker's predicted "routine" routes (using techniques described in ([2] and [3]). There are three important categories of information that need to be sent to the recommendation engine: (1) the spatial layout of the campus (modeled as a network of location nodes), (2) a set of workers (each with a detour time limit, predicted trajectories and preferences), and (3) a set of tasks (each with a location, validity time-window, execution time, and reward).

Mathematically speaking, the recommendation problem is formulated as an integer linear programming (ILP) model, which is essentially a variant of constrained routing problem. In this model, each task must be recommended to either η workers, or not at all if it's not possible. The objective is to maximize normalized expected reward from all workers, considering task-specific and worker-specific constraints under route uncertainties. To solve the problem scalably, we implemented a Lagrangian relaxation (LR) heuristic that obtains an approximate solution but with significantly lower computational time (minutes instead of hours). *TA\$Ker's Task Recommender* component was, however, enhanced to tackle the novel feature of task-specific 'validity time windows'.

Incorporate task validity time window: A task validity time-window is defined as the period during which the task must be performed. To accommodate such task validity time-windows, changes are made to the routing sub-problems of the LR heuristic developed in prior work. Now, in each worker-route level routing sub-problem, worker's routines



(a) Prediction accuracy (b) A sample user schedule for a day (c) Error in recommendations

Figure 2. Figures depicting (a) prediction accuracy of the route generator, (b) An example scenario of a user who has different schedules throughout the day and some tasks to perform and (c) error in recommending tasks (measured in minutes)

and tasks have to be spatio-temporally connected across the planning horizon. For example, assume a worker has a routine route that starts from School of Economics (SOE) at 9 am and then heads to School of Information Systems (SIS) before 9:30 am because of the class there from 9:30 am to 10:00 am. SOE and SIS are considered as two routine nodes, constrained by time-windows as well. Hence, he can perform the tasks T1 and T2 which lie in between the two routine nodes SOE and SIS. Fig. 2(b) depicts a better illustration of such routine schedules of a worker. The rectangles represent the routine nodes and availability of the user at those locations while the ovals denote various tasks, their validity time period and time taken to perform tasks. Thus, this worker can only deviate and perform the tasks during his free time between these two routine nodes. A task can only be considered if the worker is able to arrive at its location no later than its expiration time (T_e). The extra detour is redefined as the extra time caused by servicing the tasks.

Evaluating the performance of the recommendation engine: It is natural to ask: *given the inherent location errors and movement uncertainties of workers on the campus, can our recommendation strategy generate useful task recommendations?* To establish this efficacy, we compute the accuracy of two distinct recommendation strategies: (a) Our proposed strategy where the recommendations take into account each worker's individual predicted trajectories, vs. (b) A *Trajectory-oblivious* strategy, where each worker is first assigned one of 5 "representative trajectories" (each of which traverses all the floors of one of the 5 campus buildings), and the centralized recommendation algorithm is then executed on these synthetic trajectories. The recommendation error is then computed in terms of the "detour distance" to a task's location, with this detour being defined as the *minimum distance to the task location, from all reference locations that the worker actually visits during the task's validity period.* For example, assume that the recommended task T1 (with location l_1) is valid during (10:00am, 10:30am) and the observed series of stay locations during this period is $\{X,Y,Z\}$. The minimum needed detour is then computed as $\min\{d(X,l_1),d(Y,l_1),d(Z,l_1)\}$.

Fig. 2(c) plots the average error (detour overheads across all users) vs. task validity window for these two strategies. We see that, in practice, the error (average detour overhead) is approx. 2.5 times lower, compared to the Trajectory-oblivious approach. More specifically, for validity windows of 15 minutes, our recommendation error is less than 3 minutes (equivalent to travel times between floors of the same building), while the Trajectory-oblivious approach has an error of over 8 minutes (equivalent to the time needed to visit a location three buildings away). We also can see that in general, as the task validity period increases, the error in recommendation decreases, as the additional slackness in time enables us to better predict a worker's precise trajectory.

PRIMARY OBSERVATIONS FROM STUDY

Before focusing on each of the 3 key specific behavioral aspects of mobile crowd-sourcing that we studied, we present some high-level observations on the behavior of participants during the study.

More than 30,000 tasks were completed during the 8 week study period. (See Table 1 for the number of tasks completed, on a weekly basis; active users are defined as those who have completed at least one task during the corresponding week.) To provide data for corroboration, each task could be performed by up to 3 distinct workers.

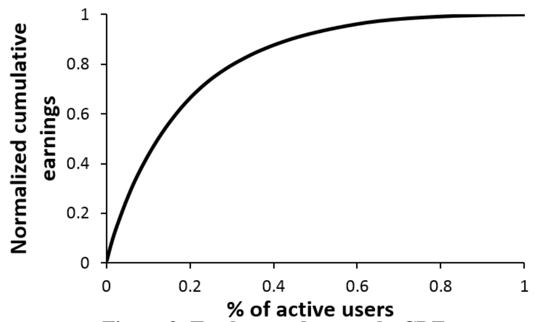


Figure 3. Total rewards earned - CDF.

Demographics

The pool of active users are mainly contributed by: (a) gender - female (55%), (b) school - the university's School of Business (38%) and (c) freshmen - new undergraduate students (47.8%). We also observed the persistence of the *super-agent* phenomenon – a relatively small core-group of users who generate a disproportionately large fraction of task responses. Fig. 3 shows that 25% of active agents are responsible for 80% of total earnings or total tasks done on *TA$Ker*.

Situational Factors

Most popular location: Our campus library building includes many prime locations where many student activities are usually carried out (and thus exhibit highest occupancy counts), which explains why most number of the tasks (35%) were completed in this building.

Most productive time window: The largest share (36%) of completed tasks occurred during the second time window (12pm – 3pm). Considering the fact that the first and last time windows are the peak-hours for lectures, it's not surprising that the second time window accounts for more tasks completed. This result is consistent with the findings in [1], which reported that crowd-workers preferred to perform tasks outside their core business hours.

Favourite type of the tasks: Interestingly, the "multiple-choice question" type was the most popular task category among the participants (56% of the completed tasks belong to this category – as a proportion of tasks posted), while picture-based tasks were least popular (only 4.5% of completed tasks belong to this category – as a proportion of total tasks). One explanation for this might be the task execution time: the average task execution time for picture-taking tasks is three times longer compared to multiple-choice tasks.

Nature of task acceptance: Push class users accept the tasks well ahead in time (around 70 minutes prior to the task expiration) and perform them within 16 minutes, while the pull class users behave more opportunistically by accepting tasks only 36 minutes prior to the task expiration time and perform them within 6 minutes (the differences are statistically significant: $p < 0.001$).

DIFFERENTIAL PRICING & ITS EFFECTS

We first study the impact of variable task pricing (weeks 3 & 4), comparing the observed outcomes against our initial flat pricing strategy (weeks 1 & 2). Note that all tasks during these 4 weeks were atomic–i.e., there were no bundled tasks.

Designing Incentive Mechanisms

While a variety of differential task pricing mechanisms have been proposed, we focus on two relatively simple, easy-to-implement differential pricing approaches, both of which are dependent primarily on the *location* attribute of tasks: (a) The **Density-based** (*popularity of location*) approach prices tasks in inverse proportion to the relative popularity (crowdedness) of individual task locations, whereas (b) The **History** (*task acceptance rate*) based approach computes the cumulative acceptance rate of tasks in a given location to date, and prices tasks in inverse proportion to this acceptance rate. Roughly speaking, both approaches seek to even out the spatial skews observed in task acceptance/completion, by preferentially increasing the rewards offered in less-popular locations (either a smaller set of visiting workers or places where workers tend to perform tasks less).

To ensure a fair comparison among different approaches, all task rewards are constrained to lie within a range defined by two fixed parameters: (1) C_{min}: the minimum reward for any task, and (2) C_{max}: the maximum permitted reward for any task. Variable task rewards are then defined by an additive "incentive" component W via the relationship Price $= C_{min} + W$, where the tunable incentive component satisfies the relationship: $(0 \leq W \leq (C_{max} - C_{min}))$.

Density Based

This approach ties the incentive to the popularity (count of visitors) of the location. For each task, the price is set by first finding, from historical location traces, the average number of unique workers (during the specified 3-hour time window) in that particular location. This measure of a location's intrinsic popularity is then normalized by the worker count over all possible locations across the entire campus. We adopt a reverse-density based pricing mechanism, with the aim of increasing the likelihood that workers will undertake additional detours to visit less-frequent places and perform tasks with greater rewards. Specifically, the variable incentive W (described above) is set as follows:

$$W = \frac{(O_{max} - U_l) * (C_{max} - C_{min})}{O_{max}}, \tag{1}$$

where O_{max} represents the maximum occupancy (visitor count) across all locations and U_l defines the expected number of *TA$Ker* workers at location l.

Historical Task Acceptance Based

This mechanism determines incentives based on the historical task acceptance rate at a particular task location. Instead of focusing on the overall workers' visit count, it factors in the intrinsic differences in task acceptance propensity across different locations. For example, certain locations (e.g., in the underground concourse) may be purely transit passages (which workers simply rush through on the way to their next class) or allow less distractions (e.g., the library); such locations may have a lower task acceptance rate, even if the number of visitors is high. For this approach, we first find out (from historical data) the completion rate of the tasks in that particular location, and then normalize this by the maximum task acceptance rate observed across all campus locations. Specifically, we adjust the incentive W by:

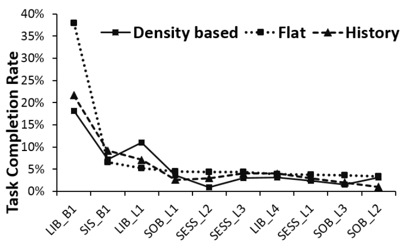
$$W = \frac{(p_{max} - r_l) * (C_{max} - C_{min})}{p_{max}}, \tag{2}$$

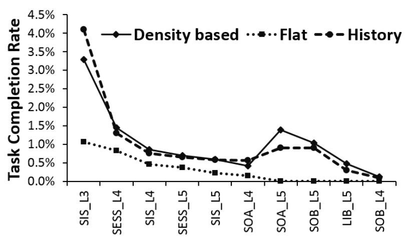
where p_{max} represents the maximum task completion rate (across all locations), and r_l defines the completion rate at task location l.

Note: We calculate the Spearman correlation coefficient for the rankings of the locations (for each time window) via these 2 (density-based and historical) approaches, and then averaged over the 15 (5 days, 3 windows/day) distinct coefficients. The average $\rho = 0.202$, with a two-tailed $p = 0.002$, meaning that these two strategies do rank the campus locations *very differently* (the difference is statistically significant).

Experimental Results

In this section, we evaluate these two differential pricing mechanisms vs. the usual 'uniform pricing' model (no location dependence) on two aspects of fairness: (1) skewness of task completion rate among various campus locations and (2) fairness of rewards earned by the workers. The data we obtained from weeks 1 and 2 were used as baseline (to observe both visitors' count and task completion rates under uniform


(a) Most Popular Locations


(b) Least Popular Locations

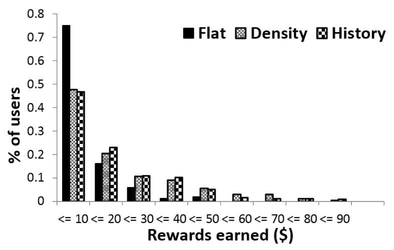

(c) Pricing vs. Worker Reward Distribution

Figure 4. Task completion rate in (a) most and (b) least popular locations, and reward skew among workers

pricing), whereas weeks 3 and 4 were used to implement the Density and Historical based approaches, respectively.

Smoothing out task completion skews: We first studied the distribution of task completion rate among all campus locations during weeks 1 and 2 (flat pricing) and observed it to be highly skewed (fairness value computed as 0.2) towards most popular/crowded locations. Figure 4(a) and 4(b) depict the task completion rates of the ten most and least crowded locations, for the three different pricing approaches. We can see that both the location density and completion rate approaches help to reduce the skew in task completion rates (i.e., make the rates more uniform) for both more-popular/visited and less-popular/uninhabited locations. Moreover, we compute the overall *fairness index* for completion rates (across all locations) using Jain's fairness index [5]: the fairness index for flat, density-based and history-based approaches were 0.2, 0.36 and 0.35, respectively. Additionally, while the overall variation in task completion rates was 0.01 under flat pricing, it reduced to 0.002 (a 5 fold reduction) under the density-based pricing approach.

Fairer rewards among workers: Figure 4(c) plots the histogram of total rewards earned per worker, when flat, density-based and task-acceptance-based incentive mechanisms were respectively employed. We see that the variable pricing schemes significantly also improve the fairness measure *across workers*, as it reduces the number of workers earning less than $10, and in general improving the *kurtosis* of the reward distribution curve. The computed fairness index for the users for flat, density-based and history-based pricing are 0.25, 0.48 and 0.49 respectively.

BUNDLING TASKS

We next study the implications of *task bundling*–i.e., offering workers a set of tasks that must all be successfully completed to receive payment. Task bundling appears to be an increasingly important element of real-world mobile crowdsourcing, where it is often economically unviable to offer the high per-task rewards needed to induce worker participation. Instead, aggregated tasks offer workers the incentive of overall higher rewards (thus increasing their willingness) at the benefit of lower detours, while offering task owners the promise of lower per-task payouts.

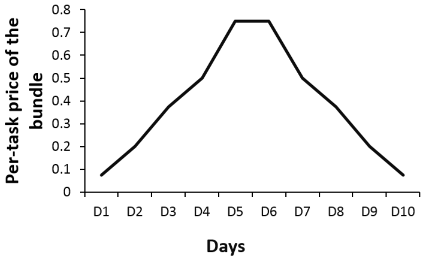
To study the effect of such bundling, we offered a corpus of bundled, as well as atomic, tasks during weeks 5 & 6. For our studies, (a) each bundle had 4 tasks, (b) the price per unit task in a bundle was varied (along with the price for individual tasks to make sure that increased per task price of a bundle does not influence a user's choice) in a sawtooth pattern (see Figure 5(a)), reaching a maximum of $0.75 per task, on days 5 & 6 of the 10-day study period, and (c) all tasks in any bundle were constrained to be on the same floor of the same building.

Note: Due to the tight schedule of the students, we assume that they can spare around 30 minutes in each time window to perform tasks. Hence, we choose the number of tasks in a bundle to be 4, allowing students to choose 2 more atomic tasks – resulting in 30 minutes task performance time (we assume each task would take at most 5 minutes to reach the location and perform).

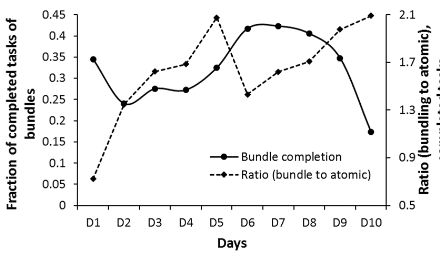
Preference for Bundles: In Figure 5(b), in the primary Y-axis, we plot the variation in the proportion of bundle-based tasks (as a fraction of the overall tasks) completed per day over the 10-day period. We see that this proportion closely tracks the variation in the price-per-bundled task over the same period. We also plot (in Figure 5(b), secondary Y-axis) the ratio between completed bundle tasks vs. atomic tasks (both normalized over number of tasks posted in each category). We observe that except the day 1 (first data point of the secondary Y-axis) bundles are always preferred, yielding a ratio greater than 1. Our analysis show that the average daily variation in task completion rate between bundles and atomic tasks is 19%, implying that bundled-tasks are favoured over atomic ones during the 2 week period.

Detour Overhead: This quantifies the additional cost in travel that is incurred by the user while performing a certain task. To measure the detour, we first need to identify the neighboring stay locations (both prior to and after the task performance) in which he stays for a significant amount of time (in our case, more than 4 minutes) to calculate the additional time needed to deviate from the shortest path between these two locations. If Z is the task location, we analyze the location traces to find the reference (stay) points before (say X) and after visiting (say Y) the task location Z. The detour time is then $(t_{X,Z} + t_{Z,Y}) - t_{X,Y}$, where $t_{x,y}$ denotes the travel time between locations x and y.

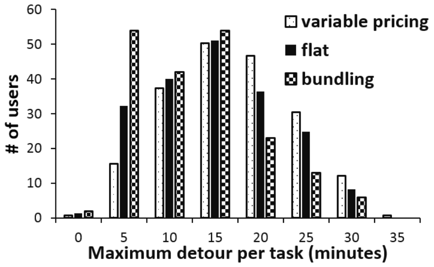
In Fig. 5(c), we show the histogram of maximum detour incurred by a user during the flat pricing study (week 1 & 2) vs. bundling study (week 6 & 7) vs. the variable pricing study (week 3 & 4). We can see that bundling *promotes detour efficiency*: 80% of the workers incurred less than 15 minutes of total detour during this period. In contrast, by increasing



(a) Bundle Pricing Variation



(b) Ratio (Bunding to Individual) - Completion Rate

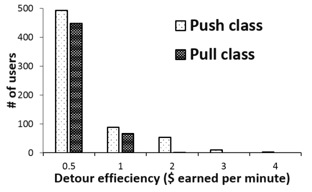


(c) Max. detour for Task Performance

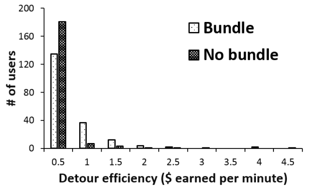
Figure 5. Bundled Tasks: (a) reward variation, (b) ratio of completed tasks vs. time and (c) maximum detour incurred while performing a task

the preference for out-of-the-way tasks, variable pricing increases the detour overhead: 70% of the users incurred more than 15 minutes of worst-case detour during week 3 & 4.

Productivity/Detour Efficiency: To measure a worker's productivity, we define *detour efficiency* as the amount of rewards he earns per minute of task-related detour. In Fig. 6(a) we plot the productivity of the users in *push* and *pull* classes. Overall (across the entire study), we found that *push* class users had higher detour efficiency ($0.38 per minute), which a *t-test* confirmed to be significantly higher ($p < 0.001$) than the detour efficiency of users in the *pull* class ($0.15 per minute). This re-confirmed our starting premise: *centrally-coordinated, trajectory-aware recommendations are more efficient than the current purely de-centralized approach*. Moreover, aggregated across all users, we found that this detour efficiency increased further for bundled tasks ($0.46 per minute of detour) compared to the detour efficiency of individual tasks ($0.26 per minute) offered in weeks 1&2 (we depicted this in Fig. 6(b)). Additionally, by regressing the detour overhead on the reward value (across all users), we observe (please refer to Table. 2) that a unit increment ($1) in the reward value would generate an increase of 3.6 mins in the detour undertaken.



(a) Push vs. Pull users



(b) Bundle vs. Individual tasks

Figure 6. User detour efficiency (a) push vs. pull class and (b)offering bundles vs. atomic tasks

Table 2. Regression Table.

Coefficients	Estimate	Std. Error	t value	$pr(>\mid t \mid)$
Rewards	3.60	0.08	43.37	< 2e-16 ***
Constant	2.24	0.10	21.55	< 2e-16 ***
Signif. codes:	0 ***	0.001 **	0.01 *	

USER CHEATING BEHAVIOUR

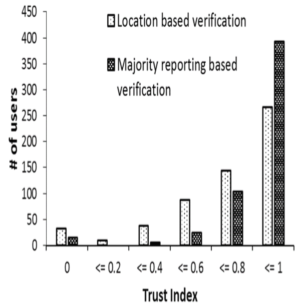
Understanding the behavioral factors behind the generation of incorrect reports is an important part of a platform that uses a de-centralized pool of workers. We now use the results of our experimental studies to analyze two distinct dimensions of such "cheating": (a) using our unique continuous indoor location tracking capability to identify instances where a worker performs a reporting task without actually visiting the location, and (b) where a worker's reports conflicts with that reported by a majority of other peer workers.

Per-user "Trust Index"

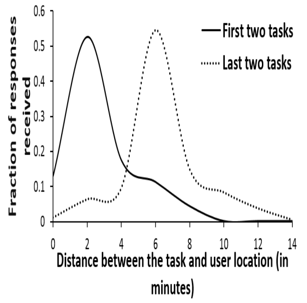
We compute a trust index in two ways: using a novel location-based technique and the classical corroboration-based methods used in most truth discovery analysis.

Location based: Whenever a worker executes a task (i.e., submits a report), we compute the distance $D_{loc}(i, j)$ between the location associated with task j and the closest reported location of user i (the worker who performed the task), within the task validity period. Given the lack of fine-grained location information, we then map each such completed task into a binary *per-task* trust metric that is '0' if the user i was not present in the building associated with task j, '1' otherwise. For any given time period, the final *Trust Index* for user i, denoted by $Trust(i)$, and computed as $Trust(i) = T^i_{nc}/T^i_{tot}$, where, T^i_{nc} is the number of tasks performed while located in the same building as the task location, and T^i_{tot} denotes the total number of tasks performed by worker i.

Corroboration-based: This approach uses the fact that many of our tasks were assigned to multiple workers simultaneously. In particular, nearly 33.5% of the responses have been submitted by 3 people. For such tasks, we declare an executed task to be un-trustworthy if its value lies well outside the range reported by the other workers.



(a) Trust Index



(b) Bundle Sequence

Figure 7. Cheating: (a) Trust index distribution (b) Sequence in bundling task impacts cheating

Fig. 7(a) shows a histogram depicting the distribution of trust index values among all workers, over the eight-week study period, for both location based verification and the corroboration-based techniques. Through the location based verification we observe that approximately 71% of users emerged out to be trustworthy, with $Trust(.)$ scores of 60% and above. We also find out that approximately 5.5% of the users have cheated in almost *every task they submitted* (an "intrinsic property"). In contrast, the corroboration-based approach is more optimistic: 92% of the users turn out to be trustworthy with the Trust(.) score of 60% or higher. Moreover, on careful analysis, we observed (as expected) a strong correlation between these values on a per-task basis: approx. 84% of tasks instances reported as cheating by the majority-voting approach also turned out to be generated by users who were not located at the task location. *However, our observations suggest that truth discovery via corroboration-based approaches may not be adequate, as users can generate acceptable reports even without visiting the specified task location.*

Demographic Influences: We additionally discovered some interesting demographic influences on such cheating behavior, namely: (a) male workers had a higher tendency to cheat (submit reports without visiting the task location) than female workers, and (b) first-year (freshmen) students had a disproportionately higher rate of cheating (38% of cheating instances, normalized by the differences in participation rates across students from all years of study).

Impact of Task Characteristics

We also studied whether such cheating behavior was influenced by other task-related parameters, such as the time validity for task execution, the type of task etc.

Tightness of task validity window: We observed that 80% of the cheated tasks have task execution window (the (start, end) duration) less than 90 minutes. In general, we suspect that the larger the time window, the higher the chance for the worker to actually visit that location, and thus the lower the likelihood of generating false reports.

Bundle tasks sequence: For bundled tasks, we also see that a *users' tendency to cheat depends on the sequence of execution inside the bundle*. Possibly reflecting the larger loss in rewards if they fail to complete just the last (or the last two) tasks, workers generate a larger proportion of false reports for the latter tasks executed in the bundle (see Fig. 7(b)). We find that when the users perform the first two tasks of a bundle, the majority of the responses ($> 65\%$) received were within 2 minutes distance from the task location. However, for the latter part of the bundle (3rd & 4th tasks), the majority of the responses ($> 80\%$) submitted were 6 minutes away from the task's building (i.e., submitted from a different building).

Discrete Valued Multiple choice: We observed that the multiple choice based tasks had the highest incidence of cheating behavior, accounting for approx. 60% of the total cheating instances. Counting based tasks came next, constituting 26.4 % of the total tasks cheated upon. In contrast, tasks such as "photo uploading", which require considerably higher independent effort from the worker, are seen to have lower likelihood of cheating.

DISCUSSION

In this section, we first provide a high-level background (i.e., context) of the deployment of our mobile crowd-sourcing application at SMU campus, and then describe how the same platform can be leveraged to build a smart campus.

Contextualizing Our Study

The SMU campus consists of 4 distinct academic buildings, 1 library, plus 1 administrative building (this is rarely utilized by students). Each of the 4 academic buildings is 5 storeys in height, with a per-floor area varying between 1500-2500 m^2 and consists principally of faculty offices, research labs, teaching classrooms and various group-study rooms. The library is 4 floors in height, with a per-floor area of approx. 2750 m^2. The library building is adjacent to the campus food court, and also contains additional food & beverage outlets– as such, it is the heart of the campus, and is the most heavily trafficked building. In Table 3, we tabulate the average daily occupancy count and fraction of tasks completed, for each building of the campus. During the trial period we posted 4 different types of tasks – the number of posted and completed tasks and average task execution time for each type is tabulated in Table 4.

Table 3. Occupancy and completion rate - building wise.

Buildings	Average Daily occupancy count	Fraction of tasks completed (normalized)
Information Systems	4185	26.23%
Economics	5105	11.75%
Library	7831	35.19%
Accountancy	4870	11.50%
Business	6334	15.32%

Table 4. Task type statistics breakdown.

Task type	Tasks posted	Tasks completed	Execution time (in secs)
Multiple-choice	20777	19110	16
Counting	18937	8732	38
Text-based	18028	2903	62
Photo	1808	135	57

We note our study was performed on a single indoor urban campus, and that additional deployments may be needed to study whether these insights can be generalized to city-scale deployments. However, some of our campus-level characteristics are quite similar to observations made in prior city-scale crowdsourcing work. For example, [4] showed that there was a spatial skew in task completion rates in city-scale crowdsourcing; this is similar to the skews that we observed on our campus.

Smart Campus

Besides providing insights into worker behavior, the *TA$Ker* deployment also helps us to target the partial development of a *smart campus*, where a volunteer student population is used to perform continuous sensing of campus resources. To illustrate the possibilities, we focus on two specific tasks–that report on the cleanliness of all the (a) restrooms and (b) rubbish bins.

With current tasks in *TA$Ker* providing reports on the status of various on-campus resources, we show that *TA$Ker* can be successfully used to crowd-source *live reports* on the status of campus facilities: (1) restrooms in all the five buildings (around 30) via a binary-valued task (clean/not clean), (2) rubbish bins (around 50) via a binary-valued task (empty/full), (3) vending machines (around 10) via a binary-valued task (stock available or not), (4) corridors (events and happenings) via free-form text and (5) the lawns on the campus (freshness). We are successfully able to collect reports on cleanliness of the restrooms and rubbish bins, every 15 and 30 minutes, respectively.

RELATED WORK

There are already several notable systems that have demonstrated values of mobile crowd-sourcing. Systems such as mCrowd [22] , Twitch crowdsourcing [19] and Slide-to-X [18] provide a platform to enable various crowd-sourcing tasks. The most visible examples such as Uber and Grab, provide a platform for willing "part-time drivers" to offer rides to passengers. Several other applications enable smart citizen monitoring by using crowd-sourcing for, e.g., detecting potholes [9], mapping noise [15] and pollution in urban areas [16], monitoring road traffic [17, 13], etc. Several instances of paid mobile crowd-sourcing start-ups have also emerged commercially including FieldAgent[3], GigWalk[4], and NeighborFavor[5]. They pay workers a few dollars for price checks, product placement checks in stores, location aware surveys, and so on. *TA$Ker* focuses on two core problems of all these systems, (1) assigning or recommending location-specific tasks to workers, and (2) exploring the relationship between worker behavior (accept/reject a task, falsely submitting a response, etc.) and attributes of the tasks (rewards, location of the task, etc.)

Task Recommendation: Workers on existing mobile crowd-sourcing service platforms have to browse through list of tasks, which is usually sorted by proximity. As the number of available tasks increases, the job of manually selecting such tasks smartly becomes more challenging. Some researchers [14] believe that such mental burden might be the reason why *super-agent phenomenon* emerges.

The *TA$Ker* platform [6] is meant to allow real-world experimentation on various issues that might be of interest to both practitioners and researchers. There are other similar efforts: for example, Kim [8] has recently created a proximity-based delivery task recommendation engine that alerts workers of suitable tasks if workers are close to a package center.

Worker Behavior: From system operator's perspective, it is important to study the relationship between worker behavior and task attributes; however, there are only very limited academic studies on this topic. Alt et al. [1] discovered a variety of worker preferences, including preference for performing tasks before and after business hours or involving relatively simple chores (e.g., taking pictures). More recently, Thebault-Spieker [4] conducted studies on the relationship between task pricing and location, at city-scale, and showed that workers preferred to perform tasks with lower detours and that were outside economically-disadvantaged areas.

Task Specification: Significant recent work has applied *algorithmic* approaches to improve the efficiency of mobile crowd-sourcing. For example, the CCS framework [21] for mobile crowd-sensing applies the principles of compressive sensing to discover correlations among different sensed data, whereas the CCS-TA framework [20] applies Bayesian inferencing to select a smaller set of tasks (& task locations) from which the data for missing tasks can be interpolated; both approaches reduce the amount of explicitly collected crowd-sensed data. Our focus has thus far been on matching workers to the set of available tasks; in the future, the approaches above be integrated to jointly perform task selection and task recommendation in a trajectory-aware fashion.

CONCLUSIONS AND FUTURE WORK

To provide better understanding of how workers perform mobile crowd-sourcing tasks, we have built and deployed *TA$Ker*, an experimental crowd-sourcing platform (front-ended by a mobile App), on the SMU campus. The studies reported here involved a pool of 900 student volunteers, who performed over 30,000 total tasks over a deployment period of 8 weeks. On the systems side, we showed that *TA$Ker* could indeed be effectively deployed on a campus, and that a trajectory-aware recommendation strategy can outperform the current pull-based approach, notwithstanding the real-world limitations of errors in location tracking and trajectory prediction.

Our experimental studies revealed three insights about human behavioral responses. First, we showed the importance of task bundling for real-world crowd-sourcing: workers prefer bundled tasks over atomic ones despite the lower per-task reward, and in fact, improve their productivity (money earned per unit of additional detour) by 77% by performing such bundled tasks. Second, we found that a simple inverse-density differential pricing model to be adequate in significantly improving the fairness in task completion rates across different campus locations (fairness improving by a factor of 1.25 compared to flat pricing schemes). Finally, we established that the tendency to generate incorrect reports (without visiting the task location) is affected by both intrinsic and *contextual* factors. Workers tended to cheat on the latter-performed tasks in a bundle; smaller execution windows for tasks led to more cheating (80% of the cheating occurred on tasks with validity windows less than 90 minutes).

In ongoing work, we continue to use *TA$Ker* to explore other facets of mobile crowd-sourcing, including individual vs. group incentive strategies (permitting workers to band together to execute larger task bundles) and understanding the price elasticity of task bundles.

Acknowledgement

This research is supported by the National Research Foundation, Prime Minister's Office, Singapore under its International Research Centres in Singapore Funding Initiative. The authors gratefully acknowledge the support of the LiveLabs research centre for the indoor mobility dataset.

[3]http://www.fieldagent.net

[4]http://www.gigwalk.com

[5]http://www.favordelivery.com

REFERENCES

1. Florian Alt, Alireza Sahami Shirazi, Albrecht Schmidt, Urs Kramer, and Zahid Nawaz. 2010. Location-based Crowdsourcing: Extending Crowdsourcing to the Real World. In *Proceedings of the 6th Nordic Conference on Human-Computer Interaction: Extending Boundaries (NordiCHI '10)*.

2. Cen Chen, Shih-Fen Cheng, Aldy Gunawan, Archan Misra, Koustuv Dasgupta, and Deepthi Chander. 2014. TRACCS: Trajectory-Aware Coordinated Urban Crowd-Sourcing. In *2nd AAAI Conference on Human Computation and Crowdsourcing*. 30–40.

3. Cen Chen, Shih-Fen Cheng, Hoong Chuin Lau, and Archan Misra. 2015. Towards city-scale mobile crowdsourcing: Task recommendations under trajectory uncertainties. In *Twenty-Fourth International Joint Conference on Artificial Intelligence*.

4. Brent Hecht Jacob Thebault-Spieker, Loren Terveen. 2015. Avoiding the South Side and the Suburbs: The Geography of Mobile Crowdsourcing Markets *(CSCW '15)*.

5. R. Jain, D.M. Chiu, and W Hawe. 1984. A Quantitative Measure of Fairness and Discrimination for Resource Allocation in Shared Computer Systems. In *Research Report TR-301*.

6. Thivya Kandappu, Archan Misra, Shih-Fen Cheng, Nikita Jaiman, Randy Tandriyansiyah, Cen Chen, Hoong Chuin Lau, Deepthi Chander, and Koustav Dasgupta. 2016. Campus-Scale Mobile Crowd-Tasking Deployment & Behavioral Insights. In *The 19th ACM Conference on Computer-Supported Cooperative Work and Social Computing*.

7. Azeem J. Khan, Vikash Ranjan, Trung-Tuan Luong, Rajesh Krishna Balan, and Archan Misra. 2013. Experiences with performance tradeoffs in practical, continuous indoor localization.. In *IEEE WOWMOM*.

8. Yongsung Kim. 2015. Libero: On-the-go crowdsourcing for package delivery. In *The 33rd Annual ACM Conference on Human Factors in Computing Systems*. ACM, 121–126.

9. A. Mednis, G. Strazdins, R. Zviedris, G. Kanonirs, and L. Selavo. 2011. Real-time Pothole Detection Using Android Smartphones with Accelerometers. In *Proceedings of DCOSS*.

10. Chenglin Miao, Wenjun Jiang, Lu Su, Yaliang Li, Suxin Guo, Zhan Qin, Houping Xiao, Jing Gao, and Kui Ren. 2015. Cloud-Enabled Privacy-Preserving Truth Discovery in Crowd Sensing Systems. In *Proceedings of the 13th ACM Conference on Embedded Networked Sensor Systems, SenSys 2015, Seoul, South Korea, November 1-4, 2015*. 183–196.

11. Microsoft. 2015. Optimizing Task Recommendations in Context-Aware Mobile Crowdsourcing. Accessed March 31 2016. (2015). `http://patents.justia.com/patent/20150317582`

12. Archan Misra and Rajesh Krishna Balan. 2013. LiveLabs: Initial Reflections on Building a Large-scale Mobile Behavioral Experimentation Testbed. *SIGMOBILE Mobile Computing and Communications Review* 17, 4 (2013), 47–59.

13. P. Mohan, V. N. Padmanabhan, and R. Ramjee. 2008. Nericell: Rich Monitoring of Road Traffic Conditions Using Mobile Smartphones. In *Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems (SenSys)*.

14. Mohamed Musthag and Deepak Ganesan. 2013. Labor dynamics in a mobile micro-task market. In *SIGCHI Conference on Human Factors in Computing Systems*. 641–650.

15. R. K. Rana, C. T. Chou, S. S. Kanhere, N. Bulusu, and W. Hu. 2010. Earphone: An end-to-end Participatory Urban Noise Mapping System. In *Proceedings of IPSN*.

16. M. Stevens and E. DHondt. 2010. Crowdsourcing of Pollution Data Using Smartphones. In *Proceedings of the Workshop on Ubiquitous Crowdsourcing*.

17. A. Thiagarajan, L. Ravindranath, K. LaCurts, S. Madden, H. Balakrishnan, S. Toledo, and J. Eriksson. 2009. Vtrack: Accurate, Energy-aware Road Traffic Delay Estimation Using Mobile Phones. In *Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems (SenSys)*.

18. K. N. Troung, T. Shihipar, and D. J. Wigdor. 2014. Slide to X: Unlocking the Potential of Smartphone Unlocking. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI)*.

19. R. Vaish, K. Wyngarden, J. Chen, B. Cheung, and M. S. Bernstein. 2014. Twitch Crowdsourcing: Crowd Contributions in Short Bursts of Time. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI)*.

20. Leye Wang, Daqing Zhang, Animesh Pathak, Chao Chen, Haoyi Xiong, Dingqi Yang, and Yasha Wang. 2015. CCS-TA: Quality-guaranteed Online Task Allocation in Compressive Crowdsensing. In *Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp '15)*.

21. Liwen Xu, Xiaohong Hao, Nicholas D. Lane, Xin Liu, and Thomas Moscibroda. 2015. More with Less: Lowering User Burden in Mobile Crowdsourcing Through Compressive Sensing. In *Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp '15)*.

22. T. Yan, M. Marzilli, R. Holmes, D. Ganesan, and M. Corner. 2009. mCrowd: A Platform for Mobile Crowdsourcing. In *Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems (SenSys)*.