Flag-Verify-Fix: Adaptive Spatial Crowdsourcing leveraging Location-based Social Networks *

Umair ul Hassan
Insight Centre of Data Analytics
National University of Ireland
Galway, Ireland
umair.ulhassan@insight-centre.org

Edward Curry
Insight Centre of Data Analytics
University College Dublin
Dublin, Ireland
edward.curry@insight-centre.org

ABSTRACT

This paper introduces the *flag-verify-fix* pattern that employs spatial crowdsourcing for city maintenance. The patterns motivates the need for appropriate assignment of dynamically arriving spatial tasks to a pool for workers on the ground. The assignment is aimed at maximizing the coverage of tasks spread over spatial locations; however, the coverage depends of willingness of workers to perform tasks assigned to them. We introduce the *maximum coverage assignment* problem that formulates two design issues of dynamic assignment. The *quantity issue* determines the number of worker required for a task and *selection issue* determines the set of workers. We propose an adaptive algorithm that uses *location diversity* based on a location-based social network to address the quantity issue and employs *Thompson sampling* for selecting the workers by learning their willingness. We evaluate the performance of the proposed algorithm in terms of coverage and number of assignments using real world datasets. The results show that our proposed algorithm achieves 30%-50% more coverage than the baseline algorithms, while requiring less workers per task.

Categories and Subject Descriptors

H.2.8 [**Database Management**]: Database Applications—*Spatial databases and GIS*

General Terms

Design, Algorithms

Keywords

Spatial crowdsourcing, location diversity, multi-armed bandit

1. INTRODUCTION

Spatial crowdsourcing (SC) has emerged as a new paradigm of asking crowd workers to perform tasks associated with physical locations [7, 6]. For instance, an air quality monitoring system might require samples from different locations in a city to continuously determine the level of pollution. Recent studies have established that the location plays a significant influence in workers' willingness to perform a task [3, 8, 9]. The majority of existing crowdsourcing platforms have implemented the *worker selected tasks* (WST) interaction mechanism, where workers actively visit the platform to choose tasks to perform [7, 5, 9]. Recent proposals have investigated the *server assigned tasks* (SAT) interaction mechanism; where the platform algorithmically assigns tasks to the workers [7, 6]. SAT-based spatial crowdsourcing aims to improve the coverage of tasks by using optimization techniques, as motivate by the following real-world scenario.

A city's administration has deployed a spatial crowdsourcing system for volunteered city maintenance [12]. Citizens submit reports from time-to-time indicating problems in their area. The system has a pool of crowd volunteers who can be contacted through mobile phones to verify and fix the problem; however, broadcasting tasks of all volunteers is both expensive and interruptive. It is desirable to a select subset of volunteers such that it is likely that one of them will be willing to perform the task. The above scenario outlines a *flag-verify-fix* (FVF) interaction pattern between a spatial crowdsourcing platform and volunteering crowd workers. Specifically, the verify and fix tasks necessitate a dynamic assignment algorithm to maximize the coverage of reported problems.

The design of dynamic assignment algorithms in SAT-based crowdsourcing is guided by two observations. 1) *The spatial and temporal context influence the workers' decision to perform tasks.* It is useful to exploit the spatial context of tasks and workers in design of the assignment algorithms. Recent studies have established that tasks in densely populated areas are more likely to be performed as opposed to sparsely populated areas [3]. Additionally, the socio-economic status of a location also affects the worker willingness [9]. 2) *Mobile workers may not always perform the tasks assigned to them.* Since workers differ in their willingness to perform tasks, it is important to learn from their observed behavior and improve assignment decisions over time [6]. We introduce the *maximum coverage assignment* (MCA) problem that formalizes the above described issues and propose an adaptive assignment algorithm. The main contributions of this paper are:

- We propose a heuristic, based on *location diversity*, to address the *quantity issue* in MCA problem. The intuition behind this heuristic is that highly visited locations are likely be positioned in dense areas; therefore, they require less assigned workers. We use an *location-based social network* to measure the location diversity of spatial tasks.
- We propose a *combinatorial multi-armed bandit* approach to address the *selection issue* in MCA problem and employ

*This work has been supported in part by the Seventh EU Framework Programme (FP7) from ICT grant agreement No. 619660 (WATERNOMICS) and the Science Foundation Ireland (SFI) under grant No. SFI/12/RC/2289.

SIGSPATIAL'15, November 03 - 06, 2015, Bellevue, WA, USA

ACM 978-1-4503-3967-4/15/11 ...$15.00.
DOI: http://dx.doi.org/10.1145/2820783.2820870.

Thompson sampling for estimating worker willingness over time. The willingness estimates are used to choose workers for the task such that its is likely the at least one of the workers will perform the task.

- We evaluate the performance of the proposed algorithm against baseline algorithm under varying conditions. For this purpose, we follow an agent-based simulation methodology using real-world data form location-based social network. The results establish that our algorithm achieves higher coverage with less workers assigned to a task.

2. PROBLEM DEFINITION

Let $\mathcal{W}$ be the set of m workers volunteering for verify and fix tasks that are generated dynamically as problems are reported in flag step at various locations. For each task t, with associated location l_t, a set of workers $W_t \subset \mathcal{W}$ must be assigned such that at least one of the workers performs the task. The objective of the system is to maximize the number of tasks successfully performed; however, success depends on the utility that workers gain from performing tasks. If worker i is assigned to task t then her utility of performing the task is defined $U_{i,t} = \vec{\beta}_{i,t}\vec{z}_{i,t} + \varepsilon_{i,t}$. The vector $\vec{\beta}_{i,t}$ contains values of the worker specific co-efficients that codify her preferences and vector $\vec{z}_{i,t}$ quantifies the factors that are known to influence worker's decision. The variable $\varepsilon_{i,t}$ quantifies the effects of unknown factors. Let the variable $y_{i,t}$ denote i'th worker's decision to perform the task t, which is takes value 1 if $U_{i,t} > 0$ and 0 otherwise. We model the utility of a worker according to the *mixed logit* model [10]; hence, the probability that the worker i performs the task t is

$$p_{i,t} = \Pr[y_{i,t} = 1] = \frac{e^{\vec{\beta}_{i,t}\vec{z}_{i,t}}}{1 + e^{\vec{\beta}_{i,t}\vec{z}_{i,t}}} \tag{1}$$

For the sake of understanding, lets assume that the decisions of workers are revealed beforehand for the task t as a vector $\vec{Y}_t = (y_{1,t}, ..., y_{m,t})$ by an oracle. Then the MCA problem for task t can be modeled by following simple integer program:

$$\begin{aligned} \vec{A}_t &= \underset{\vec{A}}{\text{argmax}}\ \vec{Y}_t \cdot \vec{A} \\ \text{s.t. } \vec{\mathbf{1}} \cdot \vec{A} &\leq= 1 \\ \vec{A} &\in \{0,1\}^m \end{aligned} \tag{2}$$

where $\vec{\mathbf{1}}$ represents the m-dimensional vector of ones. In real-world applications, the values of $y_{i,t}$ as well as the probability $p_{i,t}$ are not known beforehand. The assignment decision must be made under uncertainty; therefore, the $p_{i,t}$ has to be estimated for each task t. The process is further complicated due to the fact that the values of $y_{i,t}$ are only observed for the assigned workers. In short, an algorithm designed for MCA problem must address two issues. Firstly, how many workers should be assigned to a task? Secondly, which workers should be selected for the task?

3. ADAPTIVE ASSIGNMENT ALGORITHM

3.1 Location Diversity

The first design issue requires estimation of the number of workers required for the task. Generally assignment problems, in spatial crowdsourcing, assume that the number of workers for a task is predefined or governed by constraints [7, 5, 6]. In the MCA problem, the dynamic calculation of the number of workers for a task further complicates the problem. To address this issue, we rely of the spatial characteristics of task location. Given that the location of a task t is l_t and maximum allowance of workers for a task is K, the objective is to define a function $g : l_t \rightarrow K_t$ such that $K_t \in [1, K]$. As discussed earlier, the location of a task is one of the major factor that determine the willingness of workers. One heuristic that can be exploited for this purpose is the distribution of workers visiting a location, also known as *location diversity* [4]. A task is more likely to be completed if it is located dense area; where as, the task in sparse area might require more workers.

We use the notion of *location entropy* to represent the location diversity [7]. A location has high entropy when many workers visit that location with equal proportions; conversely, a location have low entropy if limited number of workers visit the location. The underlying intuition is that the a location with high entropy will require less workers to ensure coverage. Let O_l be the number of visits to location l and $O_{w,l}$ be the number of visits made by worker w to a location l. The probability that a random visit to location l is also a visit by worker w is $P_l(w) = O_{w,l}/O_l$ for positive O_l, which is the fractions of visits to location l made by worker w. We formally define the *location entropy* of a location as $LE(l) = -\sum_{w \in W_l} P_l(w) \cdot \log P_l(w)$, where W_l is the set of workers who have visited the location l. Note that the $LE(l) = 0$ when $O_l = 0$. We define the function g as follows:

$$g(l) = \left\lceil K \cdot \left(1 - \frac{LE(l)}{MaxLE}\right)\right\rceil \tag{3}$$

3.2 Thompson Sampling

Given that K_t is the number of workers to be assigned to the task t, the second design issue entails the selection of willing workers. Since the willingness of a worker $p_{i,t}$ is unknown, the algorithm must balance the exploration-exploitation trade-off between choosing willing worker and learning about other workers. We formulate this selection problem according to the *combinatorial multi-armed bandit* (CMAB) framework [2]. The CMAB framework is used to study online learning problems where a subset of actions are chosen from available options and the outcomes are observed only for the chosen actions. It formalizes the joint learning and optimization with limited feedback.

Assuming that the willingness of workers is independent of each other and relatively stable over time. We propose a Bayesian approach to learning the worker's willingness over time, known as *Thompson sampling* [1]. We maintain two counters for each worker: T_i is the number of tasks assigned and S_i is the number of tasks performed. The worker selection is primarily governed by an approximation of the expected willingness of workers using *Thompson sampling*. The expected willingness of worker i is quantified with variable θ_i that is sampled from the Beta distribution. The sampling distribution for θ_i is adjusted using two prior parameters (i.e α_0 and β_0) and two worker specific parameters (i.e. S_i and $(T_i - S_i)$). The set of worker is selected based on the approximated willingness of all workers. The counters of selected workers are updated over time based on tasks they perform.

3.3 The DynTS Algorithm

We propose an adaptive assignment algorithm that proceeds sequentially for dynamically arriving tasks in SAT-based spatial crowdsourcing. Algorithm 1 details the proposed algorithm that requires four parameters: the set of workers $\mathcal{W}$, the prior parameters of Beta distribution α_0 and β_0, and the maximum allowance of worker for a task K. The algorithm starts by initializing the assigned tasks T_i and performed tasks S_i counters for all workers. The main algorithm proceeds in a loop which dequeues a task

Algorithm 1 The DynTS algorithm

Require: $\mathcal{W}, \alpha_0, \beta_0, K$
$m \leftarrow |\mathcal{W}|$
for $i \leftarrow 1$ to m **do**
  $S_i \leftarrow 0$
  $T_i \leftarrow 0$
end for
for $t \leftarrow 1$ to ∞ **do**
  Wait for next task $< l_t, desc_t >$
  $K_t \leftarrow \lceil K \cdot (1 - LE(l_t)/MaxLE) \rceil$ {Quantitiy Issue}
  for $i \leftarrow 1$ to m **do**
    $\theta_i \leftarrow Beta(\alpha_0 + S_i, \beta_0 + T_i - S_i)$
  end for
  $\vec{\theta} \leftarrow (\theta_1, ..., \theta_m)$
  Choose $\vec{A}_t = (a_1, ..., a_m)$ that solves Equation (4)
  $W_t \leftarrow \{w \mid a_{i(w)} > 0, w \in \mathcal{W}\}$ {Selection Issue}
  Assign the workers in W_t
  Observe assignment outcomes $y_{i(w),t}$ for workers in W_t
  for all $w \in W_t$ **do**
    $S_{i(w)} \leftarrow S_{i(w)} + y_{i(w),t}$
    $T_{i(w)} \leftarrow T_{i(w)} + 1$
  end for
end for

Table 1: LSBN datasets

Property	New York	Tokyo
Check-ins	227,428	573,703
Users	1,083	2,293
Spots	38,333	61,858

Table 2: Compared algorithms

	W_t		
K_t	*Random*	*Distance*	*CMAB*
$K_t = K$	FixRAN	FixDIST	FixTS
$K_t = g(l_t)$	DynRAN	DynDIST	DynTS

from the task queue. For the task t, the algorithm queries a *location-based social network* (LBSN) to quantify the location entropy $LE(l_t)$ and maximum entropy $MaxLE$ over all locations. The number of workers K_t required to ensure spatial coverage is calculated using Equation (3). The expected willingness of all worker is estimated as the vector $\vec{\theta}$ by sampling from the adjusted Beta distributions of individual workers. Then, the algorithm chooses the assignment vector $\vec{A}_t$ by solving the integer program:

$$\vec{A}_t = \underset{\vec{A}}{\text{argmax}}\ \vec{\theta} \cdot \vec{A} \quad (4)$$
$$\text{s.t. } \vec{\mathbf{1}} \cdot \vec{A} \leq K_t$$
$$\vec{A} \in \{0,1\}^m$$

The algorithm assigns a set of workers $w \in W_t$ to the task t; such that, $a_{i(w)}$ is positive and $i(w)$ is the index of worker $w \in \mathcal{W}$. Equation (4) can solved in be polynomial time, with respect to number of workers, by sorting and selected the top-K_t workers. After assignment, the process waits while workers respond by performing the task. After a fixed amount of time the algorithm observes the outcome variables $y_{j',t}$ for all $j' \in W_t$. Then the counters for workers in W_t are updated accordingly.

4. EVALUATION METHODOLOGY

We follow an agent-based simulation methodology for evaluating the performance of our algorithm under varying conditions. For this purpose we simulate following agents:

LBSN Agent. This agent simulates an LBSN and exposes functions that return the entropy for locations. We used two real-world dataset extracted from FourSquare, an well known social network, to initialize the LBSN agent [11]. The datasets contains voluntarily reported visits to various locations, by the residents of New York and Tokyo cities from 12 April 2012 to 16 February 2013. Table 1 lists the properties of two datasets: New York Dataset and Tokyo Dataset. A *spot* is a geographical location and a *check-in* represents the visitor relationship between a user and a spot. The users are considered as workers and spots as task locations.

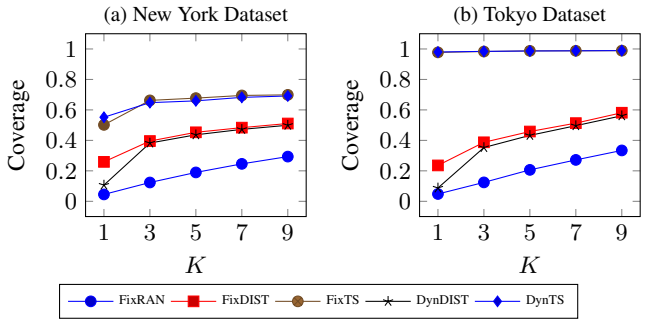

Figure 1: Comparison of assignment algorithms.

Requester Agent. This agent generates n *flag* reports that arrive dynamically with inter-arrival rate defined by parameter of λ hours. The location of each flag report is a randomly sampled from the locations in the LBSN datasets. The requester agent generates a verify task for each flag report and adds it to the queue. During the simulation the assignment process takes the verify task from the head of the queue for assignment. The requester agent exposes a function that returns the task at the head of the queue.

Worker Agent. This agent simulates the worker model as described in Section 2. Recent studies have established that the distance from task location and the socio-economic status of the task location are the major factors influencing worker's decision in spatial crowdsourcing [9]. We define $\vec{z}_{i,t} = (d_{i,t}, e_t)$ and $\vec{\beta}_{i,t} = (\beta_{d_i}, \beta_{e_i})$ for worker decision model. The distance variable $d_{i,t}$ is calculated using the last location of the worker i according to LBSN data. The socio-economic status variable e_t of a task location is approximated using the popularity of the location i.e. number of unique people who visit a location. The co-efficients for the worker i are sampled from parameterized Normal distributions i.e. $\beta_{d_i} \sim \mathcal{N}(\mu_d, \sigma_d)$ and $\beta_{e_i} \sim \mathcal{N}(\mu_e, \sigma_e)$. The random effects variables were sampled from a standard Normal distribution i.e. $\varepsilon_{i,t} \sim \mathcal{N}(0,1)$. A worker agent is initialized using check-ins of a user in the LBSN dataset and traverses the locations of check-ins during simulation. The agent exposes a function that returns a binary response indicating outcomes of an assigned task.

Platform Agent. This agent define the simulation environment and implements an assignment algorithm as specified by the parameters. The platform agent initializes an LBSN agent, a requester agent, m worker agents, and an assignment algorithm. During simulation, the assignment algorithm is executed by iteratively taking tasks from requester agent and querying worker agents.

5. RESULTS

We performed a set of experiments to compare the performance of algorithms listed in Table 2. We used two metrics that evaluate the performance of algorithms. *Coverage* is the primary metric that measures ratio of tasks performed by at least one worker after n tasks are processed. *Notifications* is the secondary metric that measures the overhead due to assignment of multiple workers to a task. We ran each algorithm 10 times under same setting and report the average results. The default parameter values in each experiment were set as $n = 1000$, $m = 50$, $K = 3$, $\alpha_0 = 1$, $\beta_0 = 1$, $\mu_d = -2$, $\sigma_d = 1$, $\mu_e = 1$, $\sigma_e = 0.33$ and $\lambda = 2$.

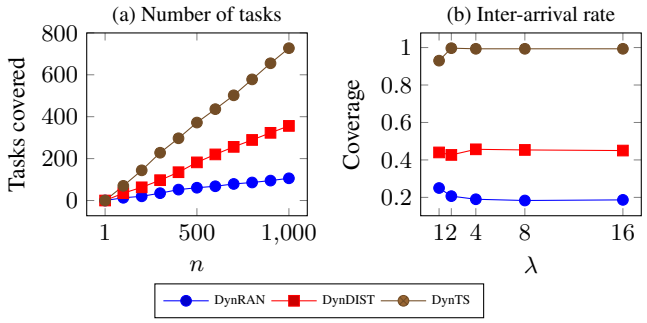

Figure 2: Comparison of assignment policies over time for the New York Dataset

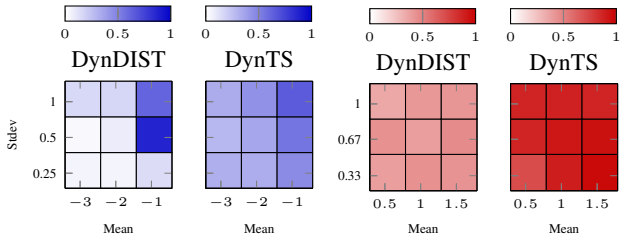

Figure 3: Coverage against co-efficients of worker model for New York Dataset, $\mathcal{N}(\mu_d, \sigma_d)$ in blue and $\mathcal{N}(\mu_e, \sigma_e)$ in red.

Comparison of Coverage. First experiment evaluated performance of algorithms by varying the $K = \{3, 5, 7, 9\}$, as shown in Figure 1. Clearly, the FixTS and DynTS algorithms perform consistently better than other algorithms. This highlights that the adaptive approach of learning worker willingness is indeed better than the non-adaptive approaches. The distance based non-adaptive approach does reach more than 50% coverage for more than 5 assignments per task. By comparison, the algorithm based on Thompson sampling achieves more than 95% coverage with only 2 assignments on Tokyo Dataset. This is achieved without the knowledge of distance between task and worker locations. The results also suggest that reasonably small values of K suffice for achieving high coverage. We also compared the relative number of workers per task between FixTS and DynTS algorithms. The DynTS algorithm required between 10%-15% less workers per task while achieving coverage similar to FixTS. Figure 2a shows the comparison of dynamic assignment algorithms over time with $K = 3$. All algorithms follow a linear pattern of coverage as the more tasks arrive and their relative performance follows same pattern over time.

Task Arrival Rate. Second experiment evaluated the effects of task arrival rates $\lambda = \{1, 2, 4, 8, 16\}$ on the performance of dynamic algorithms. Figure 2b shows the comparison of DynRAN, DynDIST and DynTS algorithms with varying mean inter-arrival rate of tasks. The changes in inter-arrival rate does not have direct effect of relative performance of algorithms. This suggests that the distance heuristic and the estimated willingness are not dependent on the frequency of task arrivals. We do suspect that if the new tasks are allowed to be assigned before the consents from previous tasks are observed then this behavior might change. We keep this investigation as part for the future work.

Worker Distribution. Third experiment evaluated the effects of variations in distribution parameters for sampling β_{d_i} and β_{e_i} co-efficients. Figure 3 shows that the DynDIST and DynTS algorithms area sensitive to the variations in β_{d_i} co-efficients and insensitive to the variations in β_{e_i} . Both algorithms perform particularly well when the β_{d_i} co-efficient are small and diverse. The DynTS algorithms performs better generally better meaning that the *Thompson sampling* is effective in adapting the assignment process for diverse and relatively willing workers.

6. CONCLUSION

In this paper, we introduce the *flag-verify-fix* interaction pattern that employs spatial crowdsourcing for city maintenance . The patterns highlights the need of a *server assigned tasks* method for the coverage of verify and fix tasks. We formalize the *maximum coverage assignment* problem and propose a novel *combinatorial multi-armed bandit* approach to address it. We propose a dynamic assignment algorithm the uses *location diversity* and *Thompson sampling* to dynamically select a set of workers for a task. Empirical evaluation suggests that the proposed algorithm perform better than the baseline non-adaptive algorithm. We plan to extend this work with dynamic assignment based on worker travel trajectories.

7. REFERENCES

[1] O. Chapelle and L. Li. An empirical evaluation of thompson sampling. In *Advances in Neural Information Processing Systems*, pages 2249–2257, 2011.

[2] W. Chen, Y. Wang, and Y. Yuan. Combinatorial multi-armed bandit: General framework and applications. In *Proceedings of the 30th International Conference on Machine Learning*, pages 151–159, 2013.

[3] Y. Chon, N. D. Lane, Y. Kim, F. Zhao, and H. Cha. Understanding the coverage and scalability of place-centric crowdsensing. In *Proceedings of the 2013 ACM international joint conference on Pervasive and ubiquitous computing*, pages 3–12. ACM, 2013.

[4] J. Cranshaw, E. Toch, J. Hong, A. Kittur, and N. Sadeh. Bridging the gap between physical location and online social networks. In *Proceedings of the 12th ACM international conference on Ubiquitous computing*, pages 119–128. ACM, 2010.

[5] D. Deng, C. Shahabi, and U. Demiryurek. Maximizing the number of worker's self-selected tasks in spatial crowdsourcing. In *Proceedings of the 21st ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pages 314–323. ACM, 2013.

[6] U. U. Hassan and E. Curry. A Multi-armed Bandit Approach to Online Spatial Task Assignment. In *Proceedings of the 11th IEEE International Conference onUbiquitous Intelligence and Computing*. IEEE, 2014.

[7] L. Kazemi and C. Shahabi. Geocrowd: enabling query answering with spatial crowdsourcing. In *Proceedings of the 20th International Conference on Advances in Geographic Information Systems*, pages 189–198. ACM, 2012.

[8] R. Teodoro, P. Ozturk, M. Naaman, W. Mason, and J. Lindqvist. The motivations and experiences of the on-demand mobile workforce. In *Proceedings of the 17th ACM conference on Computer supported cooperative work & social computing*, pages 236–247. ACM, 2014.

[9] J. Thebault-Spieker, L. Terveen, and B. Hecht. Avoiding the south side and the suburbs: The geography of mobile crowdsourcing markets. In *Proceedings of the 18th ACM conference on Computer supported cooperative work & social computing*. ACM, 2015.

[10] K. E. Train. *Discrete choice methods with simulation*. Cambridge university press, 2009.

[11] D. Yang, D. Zhang, V. W. Zheng, and Z. Yu. Modeling user activity preference by leveraging user spatial temporal characteristics in lbsns. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 45(1):129–142, 2015.

[12] F. Zambonelli. Pervasive urban crowdsourcing: Visions and challenges. In *IEEE International Conference on Pervasive Computing and Communications Workshops*, pages 578–583. IEEE, 2011.