Maximum Complex Task Assignment: Towards Tasks Correlation in Spatial Crowdsourcing

Hung Dang, Tuan Nguyen
University of Information Technology
KM20 Hanoi Hwy, Thu Duc, HCMC, Vietnam
{hungdang@aep., tuanna@}uit.edu.vn

Hien To
University of Southern California
SL 300, Los Angeles, CA 90089-0781, USA
hto@usc.edu

ABSTRACT

Recently, spatial crowdsourcing has gained emerging interest from both research community and industries. Most of current spatial crowdsourcing frameworks assume independent and atomic tasks. However, there could be some cases that one needs to crowdsource a spatial complex task consisting of some spatial sub-tasks (i.e., tasks related to a specific location). The spatial complex task's assignment requires assignments of all of its sub-tasks. Currently available frameworks are inapplicable to such kind of tasks. In this paper, we formally define the *Maximum Complex Task Assignment* (MCTA) problem, an optimization problem that schedules a maximum number of complex tasks to a given set of workers subject to constraints of workers and tasks, and propose a solution for it. We also propose another alternative that minimizes the travel cost of the assignment while the number of complex tasks assigned is still maximum. Subsequently, we perform various experiments using both real and synthetic datasets to investigate and verify the usability of our proposed approach.

Categories and Subject Descriptors

K.4.2 [**Computers and society**]: Social Issues - Assistive techonologies for persons with disabilities; H.5.2 [**Information interfaces**]: User Interfaces - Input devices and strategies

General Terms

Crowdsourcing, Complex task

Keywords

Spatial Crowdsourcing, Man-Machine Interaction, Complex Task, Task Correlation

1. INTRODUCTION

iiWAS 2013, 2-4 December, Vienna, Austria

Crowdsourcing has emerged as an efficient framework to obtain tasks in a large scale [6]. By utilizing a large contribution of the crowd, crowdsourcing enables the acquisition of needed services or products without any concerns on employees or suppliers.

The recent emergence of multi-functional mobile devices enables the concept of *spatial crowdsourcing*. Such a concept was defined in [7] as a process of crowdsourcing a set of spatial tasks to a set of people that are equipped with mobile devices. A person is required to physically travel to the location to perform a spatial task. Spatial crowdsourcing makes it possible for a requester to post a spatial task to a spatial crowdsourcing server (SCS) and wait for a worker to complete it instead of performing it by himself.

Professional works are often complicated and could be divided into smaller and atomic sub-tasks. Those sub-tasks are coordinated such that their results can be integrated to give solution for the original complicated work. However, only a few studies have addressed the problem of task correlation in crowdsourcing. Moreover, almost no literature has focused on such a problem in spatial crowdsourcing.

In this paper, we bring up the problem of task correlation in spatial crowdsourcing. The spatial complex task comprises of some spatial sub-tasks and accomplishing all of its sub-tasks is required for its completion. Subsequently, we define *Maximum Complex Task Assignment* (MCTA) problem and propose a solution for it. Finally, we perform various experiments to prove the practicability of our proposed approach.

The remainder of the paper is structured as follows. In Section 2, we discuss literature relevant to our work. Next, we formally define our approach towards working with complex task in spatial crowdsourcing in Section 3. Assignment solutions are presented in Section 4. Section 5 reports our experimental evaluation for the proposed solution based on both real and synthetic data sets. Finally, in Section 6 we conclude our study and broach up the future works.

2. RELATED WORK

Crowdsourcing: Research community has paid a great attention to crowdsourcing. Many large scale crowdsourcing services have been successfully developed. Among those, well knonwn services include MTurk[1], CrowdFlower[2] and oDesk[3]. These platforms enable users (referred to as re-

[1]Amazon mechanical turk. http://www.mturk.com/.
[2]Crowdflower. http://www.crowdflower.com/.
[3]oDesk. http://www.odesk.com

questers) to issue some crowdsourced tasks and other users (referred to as workers) to solve those tasks either voluntarily or for some specific reward. Moreover, crowdsourcing have recently found itself in various fields of computer science such as social games [5], search relevance [11, 2], and especially database design and query processing [12, 13].

Spatial Crowdsourcing: A special class of crowdsourcing is spatial crowdsourcing. Only a few studies have addressed this concept. A crowdsourcing model for answering location-based queries on top of Twitter was designed in [3]. In [7], Kazemi et al. introduced GeoCrowd, a spatial crowdsourcing framework for maximizing the number of assigned tasks. a comprehensive exploration and thorough discernment of labor dynamics in mobile task markets were provided in [10]. CrowdSC, which is a spatial crowdsourcing framework aiming to build smart cities by utilizing citizen participation, was proposed in [1]. However, none of the studies mentioned above effectively support answering complex queries which requires tasks' correlation.

Task Correlation: MTurk and other similar crowdsourcing markets have not yet widely and efficiently supported correlated tasks. A general-purposes framework [9] for crowdsourcing services with capability of achieving complex, interdependent tasks using micro-task markets has been introduced. Recently, Kittur et al. presented a system named Crowdweaver [8] which is dedicated to a visual management of complex crowd work. Though all of the above studies addressed the task correlation problem in crowdsourcing, none of them focused on spatial complex tasks.

3. SPATIAL COMPLEX TASK MODEL

Professional works are often too complex to be treated as one piece. A better and also realistic way to deal with those works is to break them into smaller and less complicated sub-tasks. In such a divide-and-conquer strategy, those subtasks need to be jointly completed to ensure the completion of the complex task. That is, if one of those sub-tasks fails to be completed, the complex task cannot be fulfilled. Thus, we introduce in this paper the concept of *spatial complex task* to reflect one kind of those professional works.

For the sake of readability, we will define in this section key terminologies of our proposed work.

DEFINITION 1 (SPATIAL COMPLEX TASK). *A spatial complex task is denoted by a form $< T, T_s, T_e >$, where T is a set of tuples of form $< q, l >$ in which a query q is to be answered at location l, and T_s and T_e are the issued and expired time of the complex task, respectively.*

A complex task is considered to be completed if and only if query q in every tuple $< q, l > \in T$ is answered. Moreover, a query q can only be answered by a person if the person physically presents at location l. Consider a situation in which one wants to obtain pictures of ten specific buildings and he requires pictures of all of those buildings; none of them is allowed to be missed, otherwise the capturing of all other buildings becomes useless. We consider capturing those ten photos as a spatial complex task comprising of ten sub-tasks, each of which is to take a picture of a particular building. One that takes the picture is required to travel to the precise location of the building. The complex task is considered completed only when all of the ten pictures, each of a particular building, are captured.

From this point, we will use sub-task and task interchangeably to denote a a tuple $< q, l > \in T$.

Similar to [7], we also consider *worker* (denoted by w) as a carrier of a mobile device who is willing to perform spatial task (sub-task) voluntarily. The worker sends a *task inquiry* to the SCS once he is online (i.e., ready to accept tasks). *Task inquiry* is defined in [7] as a request that w sends to the SCS when once he is online. The inquiry contains information on location of w, l, and two constraints: Spatial region R and maximum number of tasks (denoted by $MaxT$) that w is willing to accept. A worker w_i only accepts spatial tasks that are inside R_i and reject any tasks outside the region.

Our optimal goal here is to maximize the number of assigned complex tasks. Because of the complex task's constraint on its completion, it would be profitless if some subtasks of a complex task are assigned while some other remain unassigned. That is, all sub-tasks of a complex task are mutually either assigned or left unassigned. We now formally define the *Maximum Complex Task Assignment*.

DEFINITION 2 (MCTA). *The maximum complex task assignment (MCTA) is the procedure of assigning complex tasks issued during some interval of time to workers that are available at the assignment time in such a way that the number of complex tasks assigned is maximized.*

Note that assigning a complex task is equivalent to assigning its sub-tasks. That is, what we actually assign to workers are sub-tasks of the complex tasks. However, maximizing the number of complex tasks assigned does not necessary mean maximizing the number of sub-tasks assigned due to the fact that each complex task has difference number of sub-tasks.

Without loss of generality, we assume that workers are trusted. That is, if a worker is assigned to a task, he will complete it successfully. Based on such an assumption, a task is to be performed by one worker.

4. MAXIMUM COMPLEX TASK ASSIGNMENT PROTOCOL

4.1 Greedy Strategy

The maximum flow problem is an optimization in which the optimal goal is to find a maximum flow subject to constraints of a given flow network. MCTA problem is also an optimization to maximize the number of complex tasks assigned based on workers' constraints (spatial region R and maxT and complex tasks' requirement on their completion (all sub-tasks of a complex task must be assigned to make the complex task assigned). Because of the similarity between maximum flow and MCTA problems, the idea is that we can solve the MCTA problem by transforming into maximum flow problem. Our intuition for such the transformation is to represent complex tasks, sub-tasks and workers as vertices in a flow network. In such a representation, a complex task or a sub-task is considered to be assigned when there is a flow saturating its representative vertex and the subtask is considered to be assigned to a worker when there is a flow passing through an edge connecting the vertices denoting the sub-task and the worker. Moreover, we need to ensure that maximum flow for the derived flow network will be equivalent to maximum complex tasks assigned.

Based on the following theorem, we can solve MCTA problem by transforming it to maximum flow problem.

THEOREM 1. *The maximum complex task assignment problem is reducible to the maximum flow problem.*

PROOF. Each complex task assignment includes a set of complex tasks $CT = \{ct_1, ct_2, ct_3, ...\}$ in which each complex task is a set of sub-tasks $ct_i = \{st_{i1}, st_{i2}, st_{i3}, ...\}$, a set of all sub-tasks $ST = \{st_{11}, st_{12}, ..., st_{21}, st_{22}, ..., st_{31}, st_{32}, ...\}$ and a set of available workers $W = \{w_1, w_2, w_3, ...\}$. By reducing MCTA problem to maximum flow problem, we obtain a flow network denoted by a graph $G = \{V, E\}$ where V is a set of vertices and E is a set of edges.

The set V contains $|CT| + |ST| + |W| + 2$ vertices. The first vertex represents *source* vertex of the flow networke. The next $|CT|$ vertices denote the set of complex tasks CT. That is, every complex task ct_i is represented by a vertex v_i (referred to as *complex task* vertex). The next $|ST|$ vertices represent the set of all sub-tasks ST. Intuitively, vertices representing sub-tasks of one complex task are clustered together. Specifically, a sub-task st_{ij} is mapped to a vertex $v_{|CT|+\sum_{x=1}^{i-1}|ct_x|+j}$ (if i>1) or a vertex $v_{|CT|+j}$ (if i=1) (referred to as $sub-task$ vertex). There are also $|W|$ vertices denoting the set of worker W. In detail, each worker w_k maps to a vertex $v_{|CT|+|ST|+k}$ (referred to as *worker* vertex). Finally, we add a *sink* vertex to represent the sink vertex of the flow network.

The set E contains $|CT| + |ST| + |W| + n$ edges. The first $|CT|$ edges connect *source* and vertices representing complex tasks. Let Max_ST be a maximum number of sub-tasks per complex task. We set both *demand* and *capacity* (i.e., a lower bound and upper bound of the flow passing through the edge, respectively) of each of those to Max_ST. The reason is that as we want to treat all complex tasks equally disregarding their numbers of sub-tasks, we need to ensure that flows saturating *complex task* vertices must be equal to each other (and equal to Max_ST in this case). The other $|ST|$ edges connect vertices mapped from CT to vertices representing their sub-tasks. We set capacity of these edges to 1 as assignment of a complex task requires all of its sub-tasks are assigned. Since a worker w_k only accepts tasks inside R_k. Thus for every worker w_k, we add edges connecting all vertices representing sub-tasks that lie in R_k to $v_{|CT|+|ST|+k}$ to. We set the capacities of those edges to 1 since one sub-task is assigned to only one worker. The last $|W|$ edges connect vertices representing workers to the *sink* vertex. Because each worker w_k only accepts at most $maxT_k$, capacity of an edge connecting each vertex $v_{|CT|+|ST|+k}$ to *sink* is set to $maxT_k$.

Note that the inflow of a *complex task* vertex is ensured to be Max_ST due to the constraints on *capacity* and *demand* of the edge connecting *source* to it. Moreover, its outflow is equal to its number of sub-tasks. Thus, in order to balance the inflow and outflow of a *complex task* vertex, we need to assign to each *complex task* vertex a *node supply* (i.e, the difference between outflow and inflow of the node) equal to the difference between the number of its sub-tasks and Max_ST.

With the flow network $G = \{V, E\}$ created as described above, finding its maximum flow is equivalent to assigning the maximum number of complex tasks. □

Figures 1 and 2 better illustrate the transformation. In Figure 1, there are three workers (w_1, w_2, w_3) and three complex tasks (denoted by sets of triangles, squares and circles). Each worker has a specific spatial region (represented by a rectangle). The transformation from MCTA to maximum flow problem generates a graph represented in Figure 2. As we can see, the maximum number of sub-tasks per complex task is 5, thus we set $Max\_ST = 5$. Consequently, both demands and capacities of all edges connecting *source* and *complex task* vertices (i.e., v_1, v_2, v_3) are set to 5. Moreover, since ct_1 has only 4 sub-tasks, we set a *node supply* of its corresponding vertex (v_1) to -1. Similarly, *node supply* of vertex v_2 representing ct_2 is set to -2. As spatial region of worker w_1 covers 4 sub-tasks (st_{11}, st_{21}, st_{31}, st_{22}), there are 4 edges connecting vertices mapped from those sub-tasks (v_4, v_8, v_9, v_{11}) to vertex representing w_1 (depicted by v_{16}). And since w_2 (represented by v_{17}) specifies his maximum number of accepted tasks as $MaxT_2 = 5$, the edge connecting vertex v_{17} to the *sink* vertex has a capacity of 5. Besides, as st_{12} is not covered by any worker's spatial region, there is no edge connecting its corresponding vertex (v_5) to any *worker* vertices (v_{16}, v_{17}, v_{18}). On the other hand, since st_{23} (denoted by v_{10}) is covered by both w_2 and w_3, there are 2 edges connecting v_{10} to v_{17} and v_{18}.

After the transformation, we could use any maximum-flow related algorithm to solve the MCTA problem. Among algorithms designed to compute the maximum flow in a flow network, EdmondsKarp is very well known. The algorithm was introduced in [4]. The algorithm solves the maximum flow problem by repeatedly sending the flow from *source* to *sink* provided that there is an edge with available capacity allowing such the flow to pass through.

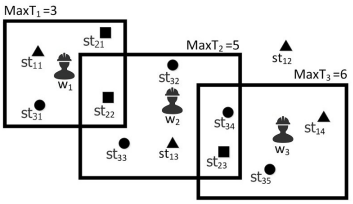

Figure 1: Example of complex task

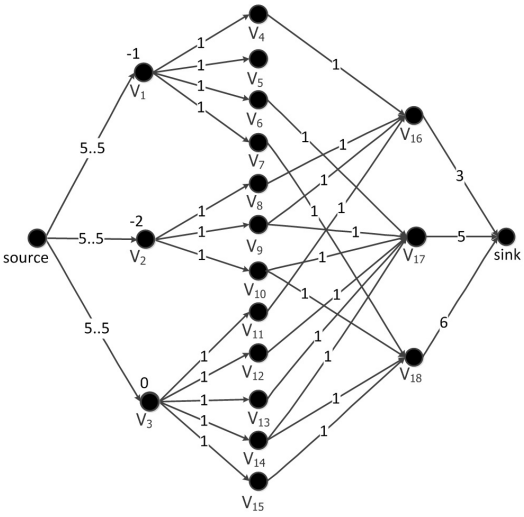

Figure 2: Example of MCTA - Maximum flow Transformation

4.2 Least Travel Cost (LTC) Strategy

Besides satisfying the constraints of workers and tasks, we also take the travel cost into consideration. Our optimal goal is still maximizing the number of complex tasks assigned. However, we give higher priorities to tasks that are in closer distance to the workers. In this paper, the travel cost of a task assignment is computed as a Euclidean distance between them. We assign priorities to the closer spatial tasks by associating to every edge connecting a vertex representing a worker w and a vertex representing a sub-task st a cost which is equal to the distance between the two (i.e., $d(w, st)$).

Let TA be a set of tuples $< w, st >$ in which a sub-task st is assigned to a worker w. The travel cost of the overall assignment is computed as follows:

$$\sum_{<w,st>\in TA} d(w, st)$$

.

The goal of LTC strategy is to maximize the number of assigned complex tasks (i.e., $|TA|$) while minimizing the travel cost (i.e., $\sum_{<w,st>\in TA} d(w, st)$). We could solve the maximum complex task minimum travel cost problem by reducing it to maximum flow minimum cost problem based on the following theorem.

THEOREM 2. *The maximum complex task minimum travel cost assignment problem is reducible to the maximum flow minimum cost problem.*

PROOF. Let $G_i = (V, E)$ be a network flow constructed as described in the proof of Theorem 1. Let's also denote a set of all sub-tasks as $ST = \{st_{11}, st_{12}, ..., st_{21}, st_{22}, ..., st_{31}, st_{32}, ...\}$ and a set of available workers as $W = \{w_1, w_2, w_3, ...\}$. Let $V_w \subset V$ be a set of vertices mapped from W and $V_{st} \subset V$ be a set of vertices mapped from ST. For every edge $(u, v) \in E$ such that $u \in V_{st}$ and $v \in V_w$, we associate with (u, v) a cost of $d(w, st)$ where w is a worker denoted by vertex v and st is a sub-task represented by vertex u. Moreover, we set costs of all other edges in E to 0. With the graph G_i constructed in the proof of Theorem 1 and modified as described above, finding its maximum flow with minimum cost is equivalent to assigning the maximum number of spatial complex task with the minimum travel cost. □

One of the well known techniques to solve the maximum flow minimum cost problem in a given flow network is to first find the maximum flow that could be feasibly sent through the flow network using Edmonds-Karp algorithm and then minimize the cost of the flow by applying linear programming.

5. EXPERIMENTAL EVALUATION

In this section, we discuss the methodology of our experimental evaluation and report average results of our various experiments. We use both real-world (REAL) and synthetic (SYN) data sets to evaluate usability of our proposed work.

5.1 Methodology

We performed three sets of experiments. In the first set of experiments whose purpose is to evaluate the scalability of the proposed approaches, we fixed the number of workers at 1,500 while varying the complex tasks from 1,000 to 4,000. In the second set of experiments which is intended to investigate the impact of number of workers on the overall assignment, we fixed the number of complex tasks at 10,000 while varying the number of workers from 250 to 4,000. In the last set of experiments which is to examine the necessity of considering travel costs in assigning spatial complex tasks, we measured the reduction in travel cost earned by LTC in comparison to the Greedy approach. Note that both approaches result to the same number of spatial complex tasks assigned.

For REAL data set, we use data that was recently published by Yelp[4]. Yelp is a social network which is dedicated for local business directory services and review site. Yelp data includes 11,537 businesses, 43,873 users and 229,907 reviews. For this data set, we consider Yelp users as workers, businesses as spatial tasks, sets of businesses with the same category as spatial complex tasks, and reviewing a business as accepting a spatial task. The spatial region of user w is set to the minimum rectangle that covers locations of businesses that user had reviewed. The maximum number of accepted tasks would be equal to the number of businesses that worker has reviewed.

For SYN data set, we randomly generate workers and complex tasks on the same area captured by the Yelp data set with uniform distribution. In SYN data set, MaxT of a worker varies from 1 to 20 and his spatial region varies from 2 to 4 percents of the entire area; a complex task may consists of 1 to 20 sub-tasks.

In all experiments, we set the duration of a complex task (T_e - T_s) as 20 days. Finally, we ran 50 cases for every set of the experiments and report the average results.

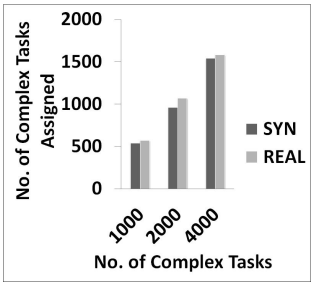

Figure 3: Scalability - No. of Complex Tasks

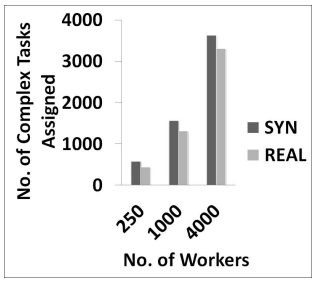

Figure 4: Impact of number of workers

5.2 Experimental Results

Scalability: The first set of experiment shows the increase in overall assignment when the number of complex tasks grows (Figure 3). However, a large number of complex tasks (up to 65%) are left unassigned. This is partially because of worker's constraints on spatial region and MaxT. Another reason lies in the correlation of sub-tasks of a complex task. That is, if only one of several sub-tasks of a complex cannot be assigned for a certain reason (for example, it is not covered by any worker's spatial region), other sub-tasks and the complex task itself will never be assigned.

Effect of number of workers: Figure 4 shows an increase in assignment as the number of available workers increases from 250 to 4,000. It is obvious that as the number of worker raises, the number of tasks could be assigned increases, and thus the overall assignment grows. However, similar to the first experimental set, a great deal of com-

[4]www.yelp.com/dataset_challenge

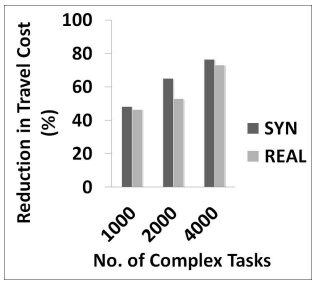

Figure 5: Reduction in travel cost

plex tasks remains unassigned even in a case that worker resource's capability (total MaxT of the workers) exceeds the number of sub-tasks. This once again confirms the cruciality of the spatial constraint in spatial crowdsourcing.

Travel Cost: Figure 5 reports the reduction in travel cost when LTC is applied instead of the Greedy strategy. As we can see, LTC outperforms Greedy strategy in term of the travel cost (up to 75%) while the number of complex tasks assigned in both the strategies is the same.

Moreover, for both data sets, the reduction in travel cost becomes more significant when the number of complex tasks increases. The reason is that as more tasks issued, the area covered by those tasks is likely to increase. Without a consideration on travel cost, a task could be assigned to a worker very far from it, which increases travel cost. The experiment confirm an observation we discussed earlier that travel cost is worth considering in spatial crowdsourcing.

6. CONCLUSION & FUTURE WORK

In this paper, we addressed the problem of task correlation in spatial crowdsourcing by introducing the concept of *spatial complex task*. A spatial complex task comprise of some spatial sub-tasks. Those sub-tasks are correlated and all of them are required to be mutually accomplished in order to ensure the completion of the complex tasks. We formally defined MCTA problem and proposed a solution for it. With our approach towards supporting crowdsourcing spatial complex tasks, spatial crowdsourcing could become more applicable to real-world market-place. Different sets of experiment were conducted utilizing both REAL and SYN data sets. In our experiments, we proved the usability as well as the scalability of our proposed approach. Moreover, the experiments also suggests the cruciality of considering travel cost in the spatial complex task assignment (LTC improves the travel cost by up to 75% in comparison with Greedy strategy).

As future work, we aim to relax the assumption that workers are self-motivated in performing spatial tasks. We plan to extend our work to reward-based spatial crowdsourcing.

7. REFERENCES

[1] K. Benouaret, R. Valliyur-Ramalingam, and F. Charoy. Crowdsc: Building smart cities with large scale citizen participation. Technical report, Feb. 2013.

[2] A. Bozzon, M. Brambilla, and S. Ceri. Answering search queries with crowdsearcher. In *Proceedings of the 21st international conference on World Wide Web*, WWW '12, pages 1009–1018, New York, NY, USA, 2012. ACM.

[3] M. F. Bulut, Y. S. Yilmaz, and M. Demirbas. Crowdsourcing location-based queries. In *Ninth Annual IEEE International Conference on Pervasive Computing and Communications*, pages 513–518, Seattle, WA, USA, 21-25 March 2011. PerCom 2011, IEEE. PerCom 2011.

[4] J. Edmonds and R. M. Karp. Theoretical improvements in algorithmic efficiency for network flow problems. *Journal of the ACM*, 1972.

[5] I. Guy, A. Perer, T. Daniel, O. Greenshpan, and I. Turbahn. Guess who?: enriching the social graph through a crowdsourcing game. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '11, pages 1373–1382, New York, NY, USA, 2011. ACM.

[6] J. Howe. The rise of crowdsourcing. *Wired Magazine*, 2006.

[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*, SIGSPATIAL '12, pages 189–198, New York, NY, USA, 2012. ACM.

[8] A. Kittur, S. Khamkar, P. André, and R. Kraut. Crowdweaver: visually managing complex crowd work. In *Proceedings of the ACM 2012 conference on Computer Supported Cooperative Work*, pages 1033–1036, New York, NY, USA, 2012. ACM.

[9] A. Kittur, B. Smus, S. Khamkar, and R. E. Kraut. Crowdforge: crowdsourcing complex work. In *Proceedings of the 24th annual ACM symposium on User interface software and technology*, pages 43–52, New York, NY, USA, 2011. ACM.

[10] M. Musthag and D. Ganesan. Labor dynamics in a mobile micro-task market. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '13, pages 641–650, New York, NY, USA, 2013. ACM.

[11] A. Parameswaran, A. D. Sarma, H. Garcia-Molina, N. Polyzotis, and J. Widom. Human-assisted graph search: it's okay to ask questions. *Proc. VLDB Endow.*, 4(5):267–278, Feb. 2011.

[12] A. G. Parameswaran, H. Garcia-Molina, H. Park, N. Polyzotis, A. Ramesh, and J. Widom. Crowdscreen: algorithms for filtering data with humans. In *Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data*, SIGMOD '12, pages 361–372, New York, NY, USA, 2012. ACM.

[13] Z. Zhao, W. Ng, and Z. Zhang. Crowdseed: query processing on microblogs. In *Proceedings of the 16th International Conference on Extending Database Technology*, EDBT '13, pages 729–732, New York, NY, USA, 2013. ACM.