Dynamic Task Assignment in Spatial Crowdsourcing

Yongxin Tong[1], Zimu Zhou[2]
[1] BDBC and SKLSDE Lab, Beihang University, Beijing, China
[2] TIK, ETH Zurich, Zurich, Switzerland
[1]yxtong@buaa.edu.cn, [2] zzhou@tik.ee.ethz.ch

Abstract

Spatial crowdsourcing is a crowdsourcing paradigm featured with spatiotemporal information of tasks and workers. It has been widely adopted in mobile computing applications and urban services such as citizen sensing, P2P ride-sharing and Online-To-Offline services. One fundamental and unique issue in spatial crowdsourcing is dynamic task assignment (DTA), where tasks and workers appear dynamically and need to be assigned under spatiotemporal constraints. In this paper, we aim to provide a brief overview on the basics and frontiers of DTA research. We define the generic DTA problem and introduce the evaluation metrics to its solutions. Then we review mainstream solutions to the DTA problem. Finally we point out open questions and opportunities in DTA research.

1 Introduction

Crowdsourcing is a computing paradigm where humans actively participate in the procedure of computing, especially for the tasks that are intrinsically easier for humans than for computers. There has been active research on crowdsourcing [3, 11, 16, 22, 23] using web-based crowdsourcing platforms such as Amazon Mechanical Turks (AMT) and oDesk. The development of mobile Internet and sharing economy has triggered the shift from web-based crowdsourcing to spatial crowdsourcing (*a.k.a* mobile crowdsourcing) [4], where *(i)* each worker is considered as a mobile computing unit to complete tasks using their mobile devices [18] and *(ii)* spatial information such as location, mobility and the associated contexts plays a crucial role. Applications of spatial crowdsourcing have deeply penetrated into everyday life. Some of the most representative applications include real-time taxi-calling services (*e.g.*, Uber and DiDi), product placement checking services in supermarkets (*e.g.*, Gigwalk and TaskRabbit) on-wheel meal-ordering services (*e.g.*, GrubHub and Instacart), and citizen sensing services (*e.g.*, Waze and OpenStreetMap).

As with web-based crowdsourcing, a central issue in spatial crowdsourcing is task assignment, which aims to assign tasks to suitable workers such that the total weighted value of the assigned pairs of tasks and workers is maximized or the total moving distance of the workers is minimized [13, 26, 25, 28, 19, 27, 29, 30]. Different from task assignment in web-based crowdsourcing, the unique spatiotemporal dynamics in spatial crowdsourcing calls for new designs in task assignment theories and methods. Particularly, the tasks and works in spatial crowdsourcing may appear dynamically and task assignment needs to be performed immediately or in a short period, *a.k.a.* dynamic task assignment (DTA).

The DTA problem is challenging because *(i)* assignments are made under incomplete information; *(ii)* assignments usually cannot be revoked; and *(iii)* assignments need to be performed computationally efficient to meet the real-time requirements on large datasets. We formulate the generic DTA problem in Sec. 2 and review representative solutions to the DTA problem in Sec. 3. We finally point out open questions and opportunities for future research on DTA in Sec. 4.

2 Dynamic Task Assignment Problem

2.1 Problem Statement

For a spatial crowdsourcing platform ("platform" for short), the generic dynamic task assignment problem can be formulated based on the following definitions.

Definition 1 (Task): A task, denoted by $t =< \boldsymbol{l}_t, a_t, d_t, c_t >$, at the location $\boldsymbol{l}_t$ in the 2D space is posted on the platform at time a_t and is either allocated to a worker who arrives on the platform before the response deadline d_t or cannot be allocated thereafter. No more than c_t worker are required to perform the task.

Definition 2 (Worker): A worker, denoted by $w =< \boldsymbol{l}_w, a_w, d_w, c_w >$, arrives at the platform with an initial location $\boldsymbol{l}_w$ in the 2D space at time a_w and either performs a task which arrives at the platform before its response deadline d_w or does not conduct any task. Once a worker finishes a task, s/he can be viewed as a new worker if s/he is willing to be assigned other tasks. A worker is able to perform c_w tasks at most.

Definition 3 (Constraint Function): A constraint function $f_c(t, w)$ is used to indicate whether t can be assigned to w. Generally speaking, the constraint function is related to some spatiotemporal requirements, such as whether t is in the service range of w, or whether w can arrive at the position of t before its deadline.

Definition 4 (Utility Function): A utility function $f_u(t, w)$ is used to measure the utility of assigning t to w. It can be the payoff of the task or the payoff times the probability that t can be finished successfully.

Definition 5 (Distance Function): A moving cost function $f_d(t, w)$ is used to measure the cost of w if s/he moves to the location of t to perform the task. In practice the distance function can be the Euclidean distance or road network distance between t and w.

Definition 6 (DTA Problem): Assume a set of tasks T, a set of workers W, a constraint function $f_c(\cdot, \cdot)$, a utility function $f_u(\cdot, \cdot)$ and a distance function $f_d(\cdot, \cdot)$ on a spatial crowdsourcing platform. Suppose initially there is no task or worker on the platform. Workers and tasks then arrive dynamically at any time. The DTA problem is to find an assignment M among the tasks and the workers for different objectives, which can be either maximizing the total utility $U = \sum_{(t,w)\in M} f_u(t, w)$ or minimizing the total moving cost $C = \sum_{(t,w)\in M} f_d(t, w)$ of the assignment pairs, such that the following constraints are satisfied:

- Spatiotemporal constraint: $\forall (t, w) \in M, f_c(t, w) = 1$, which means that t can be assigned to w.
- Invariable constraint: once a task t is assigned to a worker w, the allocation of (t, w) cannot be changed.

As opposed to static task assignment, where the spatiotemporal information of all the workers and tasks is known, the DTA problem needs to make an effective assignment with partial information about the workers and tasks. We illustrate the DTA problem for maximizing the total utility using the following example.

Example 1: Suppose we have five tasks t_1-t_5 and three workers w_1-w_3 on a spatial crowdsourcing platform, whose initial locations are shown in a 2D space in Fig. 1. Each worker has a spatial restricted activity range, indicating that the worker can only conduct tasks that locate within the range, which is shown as a dotted circle in Fig. 1. Each user also has a capacity (*i.e.*, c_w), which is the maximum number of tasks that can be assigned to him/her. In this example, the capacity of each worker is 2 and the capacity of each task is 1. Fig. 2 presents the utility values for each pair of task and worker, which is marked on the edge between the worker and the task. In the static scenario, the total utility of the optimal task assignment is 10 (marked in red in Fig. 2). However, in the dynamic scenario, the offline solutions are not always applicable since both the workers and the tasks arrive dynamically at the platform. This is the main challenge for dynamic task assignment.

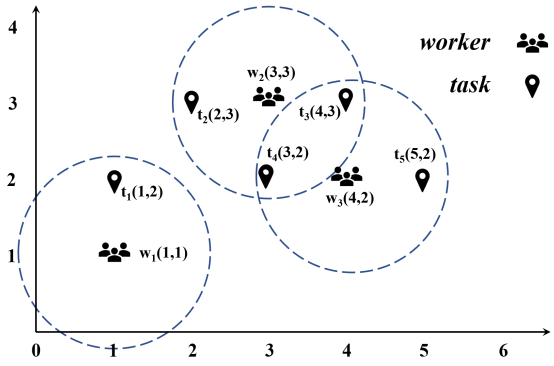

Figure 1: An instance of dynamic task assignment.

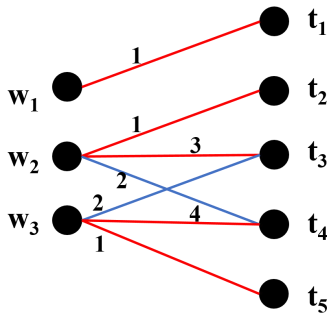

Figure 2: The utility between workers and tasks.

2.2 Evaluation Metric for DTA Algorithms

The solutions to the DTA problem are usually online algorithms [2].Different from traditional approximation algorithms for which approximation ratios are utilized to measure the approximation quality, for online algorithms, *competitive ratios (CR)* are used to evaluate their performance. In particular, the competitive ratio measures how good an online algorithm is compared with the optimal result of the offline model where all the information is provided. Based on different assumptions on the arrival order of the tasks and workers, typical online models include the adversarial model, random order model and i.i.d model. Take the goal of maximizing total utility as examples. The corresponding competitive ratios of the three types of online models are defined as follows.

Definition 7 (CR in the Adversarial Model [17]): The competitive ratio in the the adversarial model of a specific online algorithm for the DTA problem is the following minimum ratio between the result of the online algorithm and the optimal result over all possible arrival orders of the tasks and the workers,

$$CR_A = min_{\forall G(T,W,U) \text{ and } \forall v \in V} \frac{\text{Performance of } M}{\text{Performance of } OPT} \tag{1}$$

where $G(T, W, U)$ is an arbitrary input of tasks, workers and their utilities, V is the set of all possible input orders, and v is one order in V.

Definition 8 (CR in the Random Order Model [17]): The competitive ratio in the the random order model of a specific online algorithm for the DTA problem is the following ratio,

$$CR_{RO} = min_{\forall G(T,W,U)} \frac{E[\text{Performance of } M]}{\text{Performance of } OPT} \tag{2}$$

where $G(T, W, U)$ is an arbitrary input of tasks, workers and their utilities, $\frac{E[MaxSum(M)]}{MaxSum(OPT)}$ is the expectation of the ratio of the total utility produced by the online algorithm and the optimal total utility of the offline scenario over all possible arrival orders.

Definition 9 (CR in the i.i.d Model [10]): The competitive ratio in the i.i.d model of a specific online algorithm for the DTA problem is the minimum ratio of the result of the online algorithm over the optimal result under all possible arrival orders generated by the spatiotemporal distributions of the tasks and the workers $\mathcal{D}_{\mathcal{R}}$ and $\mathcal{D}_{\mathcal{W}}$,

$$CR_{i.i.d} = min_{\forall G(T,W,U) \text{ and } \forall v \in V \text{ follows } \mathcal{D}_{\mathcal{R}} \text{ and } \mathcal{D}_{\mathcal{W}}} \frac{\text{Performance of } M}{\text{Performance of } OPT}$$

where $G(T, W, U)$ is an arbitrary input of tasks, workers and their utilities, V is the set of all possible input orders of tasks and the workers, and v is one order in V.

3 Dynamic Task Assignment Algorithms

Solutions to the dynamic task assignment problem roughly fall into two modes: batch mode and real-time mode. Batch mode periodically processes a set of workers and tasks that arrive within a specific time interval. Real-time mode makes an assignment immediately when a worker or a task appears on the platform. Both modes are able to handle dynamically arrived workers and tasks. However, only the real-time mode is suited for stringent real-time requirement, *i.e.*, tasks should be assigned immediately upon arrival.

3.1 Batch Mode

The basic idea of the batch mode is to periodically make an assignment in the static scenario, *i.e.*, both workers and tasks have already appeared on the platform. All the existing solutions in the batch mode [13, 14, 21, 20, 6] aim to maximize the total utility. According to the methods to conduct the static assignment, the batch mode can be further categorized as *maxmimum flow based* methods [13, 14, 21] and *greedy based* methods [20, 6].

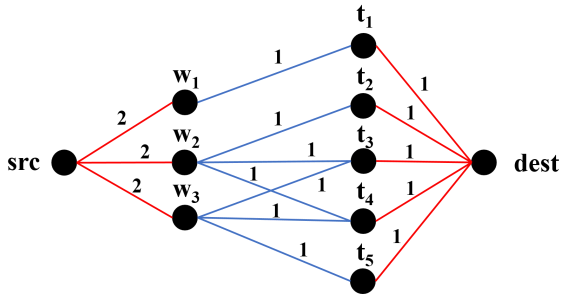

Figure 3: The procedure of reducing DTA to the maximum flow problem.

Maximum Flow Based Methods. Kazemi *et al.* [13] are the first to use a batch mode solution to the DTA problem in spatial crowdsourcing. Since they aim to maximize the number of performed tasks (*i.e.*, the utility between each worker and task is 1), the basic idea is to reduce the instance of DTA into an instance of maximum flow problem. Fig. 3 illustrates the procedure of the reduction. First, the capacity of the edge (src, w_i) is c_{w_i} because each worker can only perform c_{w_i} tasks at most. Second, since workers can only perform tasks that are in their regions (*e.g.*, w_1 can only perform t_1), the vertex mapped from w_i can transfer flow to only some of the vertices mapped from those tasks (*e.g.*, the edge between w_1 and t_1). The capacity of the edge between workers and tasks is 1 because a worker would not repeatedly perform the same task. Last, the capacity of the edge $(t_j, dest)$ is c_{t_j} because a task can only have at most c_{t_j} assigned workers. By reducing to the maximum flow problem, any algorithm that computes the maximum flow in the network can be used to solve the instance, *e.g.*, Ford-Fulkerson algorithm [15]. Finally, the assignment between workers and tasks can be induced through the flow and capacity between w_i and t_j in the instance of maximum flow problem. Consequently, to solve the DTA problem we repeat this step for every batch.

Multiple heuristics techniques have been proposed to optimize solutions in the batch mode. Hient *et al.* [21] introduce a Least Location Entropy Priority (LLEP) strategy to seek a global optimal by considering future coming workers. They use entropy of a location to measure the total number of workers in that location as well as the relative proportion of their future visits to that location. A higher priority is given to tasks located in areas with smaller location entropy, because those tasks have a lower chance of being completed by other workers. A Nearest Neighbor Priority (NNP) Strategy is also proposed to minimize the total travel distance of workers.

Greedy Based Methods. Greedy is a straightforward batch mode solution to the DTA problem. Hien *et al.* [20] propose to always select the worker who has the maximum number of unperformed tasks. Cheng *et al.* [6] design a greedy strategy to always select the pair of worker and task with the maximum utility. The benefit of

the greedy methods is that they are usually efficient and a few techniques can help improve the effectiveness of the methods. Some successful optimization techniques include prediction the arrivals of tasks and workers [6], divide-and-conquer [6], etc.

Summary. A comprehensive experimental comparison among batch-based solutions can be found in [5]. LLEP [21] is more effective but less efficient due to the time complexity of maximum flow problem and NNP [21] is more efficient. Note that assignment algorithms in the batch mode are online algorithms and it is possible to analyze their theoretical guarantees under partial information, *i.e.*, competitive ratio. However, all the methods in the batch mode [13, 14, 21, 20, 6] can only guarantee their effectiveness within a batch but there is no guarantee on the global performance. It remains open whether the batch mode is competitive under the adversarial model, random order model and i.i.d model.

3.2 Real-time Mode

Solutions in the real-time mode to the DTA problem decide the assignment once a worker or a task appears on the platform and are thus more challenging than those in the batch mode. Existing solutions in the real-time mode vary in objective goals. Popular optimization objectives include minimizing the total travel distance between workers and tasks [12, 1, 17, 25] such that the average waiting time of tasks (*e.g.*, passengers in taxi dispatching services [31]) is minimized, and maximizing the total utility between workers and tasks [26, 19, 28, 7, 8].

Minimizing the Total Distance. Greedy [12] can be a naive method to immunize the total distance. It matches each new arrival request to its currently nearest unmatched worker. The competitive ratio of Greedy is $O(2^n - 1)$ under adversarial model [12]. In [12], the authors also propose another method, called Permutation, to further improve the competitive ratio to $O(2n-1)$. The basic idea of Permutation is to make an assignment for each task according to the result of the offline minimum weighted matching (*e.g.*, Hungarian algorithm). Since a deterministic method may easily obtain a worse competitive ratio under the adversarial model, other researchers [1, 17] utilize the Hierarchically Separated Tree (HST) [9] to design a randomized algorithm such that a log-scale competitive ratio can be obtained. Specifically, they first embed the metric space into an HST. Then, they use HST-Greedy [17] and HST-Reassignment [1] to achieve the ratio of $O(\log^3 n)$ and $O(\log^2 n)$. However, these studies [12, 1] mainly focus on analyzing the worst-case competitive ratios of the proposed online algorithms, while [25] studies the performance of these algorithms in practice (*i.e.*, Random Order Model). Particularly, they observe a surprising result that the simple and efficient greedy algorithm, which has been considered as the worst due to its exponential worst-case competitive ratio, is significantly more effective than other algorithms. They further show that the competitive ratio of the worst case of the Greedy algorithm is actually a constant of 3.195 in the average-case analysis.

Maximizing the Total Utility. In order to maximize the total utility between workers and tasks, Tong *et al.* [26] propose a Hungarian-based method called TGOA with competitive ratio $1/4$ under random order model. They also propose a greedy-based method called TGOA-Greedy with competitive ratio $1/8$ under the same model. Both [26] and [19] propose a threshold-based method to maximize the utility in bipartite matching [26] and trichromatic matching [19]. A threshold of utility is sampled beforehand and the method arbitrarily choose an assignment only if the utility between the worker and the task is above the threshold. In practice, prediction is also used to improve the effectiveness and efficiency of the real-time methods [28, 24, 7, 8].

Summary. A comprehensive experimental comparison among solutions in the real-time mode which minimize the total distance can be found in [25]. From large-scale evaluations, a surprising result is observed that the simple greedy algorithm may be still competitive in the random order model, which is more practical than the adversarial model. Nevertheless, comprehensive evaluations among solutions in the real-time mode in the goal of maximizing the total utility are still missing.

We summarize existing solutions in the real-time mode in Table 1. Constant competitive ratio is usually achievable under the random order model and the i.i.d model. The competitive ratio under i.i.d model is usually

Table 1: Comparisons of existing real-time solutions to the dynamic task assignment problem.

Objective	Method	Analysis Model	Competitive Ratio
Minimize Distance	Greedy [12]	Adversarial	$O(2^n - 1)$
	Permutation [12]	Adversarial	$O(2n - 1)$
	HST-Greedy [17]	Adversarial	$O(\log^3 n)$
	HST-Reassignment [1]	Adversarial	$O(\log^2 n)$
	Greedy [25]	Random order	3.195 in worst case
Maximize Utility	TGOA [26]	Random Order	0.25
	TGOA-Greedy [26]	Random Order	0.125
	Basic-Threshold [19]	Random Order	$1/(3e \ln(U_{max} + 1))$
	POLAR-OP [28]	i.i.d	0.47
	ADAP [7]	i.i.d	$0.5 - \epsilon$
	NADAP [8]	i.i.d	0.295

higher, because prediction is usually helpful to improve the effectiveness of method in practice [28, 7, 8].

4 Conclusion

In this article, we formulate the generic Dynamic Task Assignment (DTA) problem for spatial crowdsourcing and briefly review the state-of-the-art solutions to the DTA problem. As an emerging research topic, DTA is far from mature. We list some of the open questions below. One interesting open problem is whether Greedy can achieve constant competitive ratio under the random order model for the DTA problem when minimizing total distance [25]. Another open issue is whether existing spatial indexes, which support moving object queries, can be extended to support the online data processing in spatial crowdsourcing. Finally, well-defined benchmarks to test and compare different spatial crowdsourcing data processing techniques are still missing. We envision this paper to not only raise awareness of DTA in the database community but also invite the database researchers to advance this promising area.

References

[1] N. Bansal, N. Buchbinder, A. Gupta, and J. Naor. A randomized $o(log^2 k)$-competitive algorithm for metric bipartite matching. *Algorithmica*, 68(2):390–403, 2014.

[2] A. Borodin and R. El-Yaniv. *Online computation and competitive analysis*. Cambridge University Press, 2005.

[3] L. Chen, D. Lee, and T. Milo. Data-driven crowdsourcing: Management, mining, and applications. In *31st IEEE International Conference on Data Engineering*, ICDE '15, pages 1527–1529, 2015.

[4] L. Chen and C. Shahabi. Spatial crowdsourcing: Challenges and opportunities. *IEEE Data Engineering Bulletin*, 39(4):14–25, 2016.

[5] P. Cheng, X. Jian, and L. Chen. An experimental evaluation of task assignment in spatial crowdsourcing. *Proceedings of the VLDB Endowment*, 11(11):1428–1440, 2018.

[6] P. Cheng, X. Lian, L. Chen, and C. Shahabi. Prediction-based task assignment in spatial crowdsourcing. In *33rd IEEE International Conference on Data Engineering*, ICDE '17, pages 997–1008, 2017.

[7] J. P. Dickerson, K. A. Sankararaman, A. Srinivasan, and P. Xu. Allocation problems in ride-sharing platforms: Online matching with offline reusable resources. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*, AAAI '18, pages 1007–1014, 2018.

[8] J. P. Dickerson, K. A. Sankararaman, A. Srinivasan, and P. Xu. Assigning tasks to workers based on historical data: Online task assignment with two-sided arrivals. In *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems*, AAMAS '18, pages 318–326, 2018.

[9] J. Fakcharoenphol, S. Rao, and K. Talwar. A tight bound on approximating arbitrary metrics by tree metrics. In *Proceedings of the 35th Annual ACM Symposium on Theory of Computing*, STOC '13, pages 448–455, 2003.

[10] J. Feldman, A. Mehta, V. Mirrokni, and S. Muthukrishnan. Online stochastic matching: Beating 1-1/e. In *FOCS 2009*.

[11] H. Garcia-Molina, M. Joglekar, A. Marcus, A. G. Parameswaran, and V. Verroios. Challenges in data crowdsourcing. *IEEE Transactions on Knowledge and Data Engineering*, 28(4):901–911, 2016.

[12] B. Kalyanasundaram and K. Pruhs. Online weighted matching. *Journal of Algorithms*, 14(3):478–488, 1993.

[13] 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, 2012.

[14] L. Kazemi, C. Shahabi, and L. Chen. Geotrucrowd: trustworthy query answering with spatial crowdsourcing. In *Proceedings of the 21st International Conference on Advances in Geographic Information Systems*, SIGSPATIAL '13, pages 304–313, 2013.

[15] J. Kleinberg and E. Tardos. *Algorithm design*. Pearson Education India, 2006.

[16] G. Li, J. Wang, Y. Zheng, and M. Franklin. Crowdsourced data management: A survey. *IEEE Transactions on Knowledge and Data Engineering*, 28(9):2296–2319, 2016.

[17] A. Meyerson, A. Nanavati, and L. Poplawski. Randomized online algorithms for minimum metric bipartite matching. In *Proceedings of the Seventeenth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '06, 2006.

[18] M. Musthag and D. Ganesan. Labor dynamics in a mobile micro-task market. In *2013 ACM SIGCHI Conference on Human Factors in Computing Systems*, CHI '13, 2013.

[19] T. Song, Y. Tong, L. Wang, J. She, B. Yao, L. Chen, and K. Xu. Trichromatic online matching in real-time spatial crowdsourcing. In *33rd IEEE International Conference on Data Engineering*, ICDE '17, pages 1009–1020, 2017.

[20] H. To, L. Fan, L. Tran, and C. Shahabi. Real-time task assignment in hyperlocal spatial crowdsourcing under budget constraints. In *2016 IEEE International Conference on Pervasive Computing and Communications*, PerCom '16, pages 1–8, 2016.

[21] H. To, C. Shahabi, and L. Kazemi. A server-assigned spatial crowdsourcing framework. *ACM Trans. Spatial Algorithms and Systems*, 1(1):2:1–2:28, 2015.

[22] Y. Tong, L. Chen, and C. Shahabi. Spatial crowdsourcing: Challenges, techniques, and applications. *Proceedings of the VLDB Endowment*, 10(12):1988–1991, 2017.

[23] Y. Tong, L. Chen, Z. Zhou, H. V. Jagadish, L. Shou, and W. Lv. SLADE: A smart large-scale task decomposer in crowdsourcing. *IEEE Transactions on Knowledge and Data Engineering*, 30(8):1588–1601, 2018.

[24] Y. Tong, Y. Chen, Z. Zhou, L. Chen, J. Wang, Q. Yang, J. Ye, and W. Lv. The simpler the better: A unified approach to predicting original taxi demands based on large-scale online platforms. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '17, pages 1653–1662, 2017.

[25] Y. Tong, J. She, B. Ding, L. Chen, T. Wo, and K. Xu. Online minimum matching in real-time spatial data: experiments and analysis. *Proceedings of the VLDB Endowment*, 9(12):1053–1064, 2016.

[26] Y. Tong, J. She, B. Ding, L. Wang, and L. Chen. Online mobile micro-task allocation in spatial crowdsourcing. In *32nd IEEE International Conference on Data Engineering*, ICDE '16, pages 49–60, 2016.

[27] Y. Tong, L. Wang, Z. Zhou, L. Chen, B. Du, and J. Ye. Dynamic pricing in spatial crowdsourcing: A matching-based approach. In *Proceedings of the 2018 International Conference on Management of Data*, SIGMOD '18, pages 773–788, 2018.

[28] Y. Tong, L. Wang, Z. Zhou, B. Ding, L. Chen, J. Ye, and K. Xu. Flexible online task assignment in real-time spatial data. *Proceedings of the VLDB Endowment*, 10(11):1334–1345, 2017.

[29] Y. Tong, Y. Zeng, Z. Zhou, L. Chen, J. Ye, and K. Xu. A unified approach to route planning for shared mobility. *Proceedings of the VLDB Endowment*, 11(11):1633–1646, 2018.

[30] Y. Zeng, Y. Tong, L. Chen, and Z. Zhou. Latency-oriented task completion via spatial crowdsourcing. In *34rd IEEE International Conference on Data Engineering*, ICDE '18, pages 317–328, 2018.

[31] L. Zhang, T. Hu, Y. Min, G. Wu, J. Zhang, P. Feng, P. Gong, and J. Ye. A taxi order dispatch model based on combinatorial optimization. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '18, pages 2151–2159, 2017.