Transit-based Task Assignment in Spatial Crowdsourcing

Gummidi, Bhuvan; Pedersen, Torben Bach; Xie, Xike

Published in:
SSDBM 2020

DOI (link to publication from Publisher):
10.1145/3400903.3400929

Publication date:
2020

Document Version
Publisher's PDF, also known as Version of record

Transit-based Task Assignment in Spatial Crowdsourcing

Srinivasa Raghavendra Bhuvan Gummidi
Aalborg University
gumuhb@cs.aau.dk

Torben Bach Pedersen
Aalborg University
tbp@cs.aau.dk

Xike Xie
University of Science and Technology of China
xkxie@ustc.edu.cn

ABSTRACT

Worker movement information can help the spatial crowdsourcing platform to identify the right time to assign a task to a worker for successful completion of the task. However, the majority of the current assignment strategies do not consider worker movement information. This paper aims to utilize the worker movement information via transits in an online task assignment setting. The idea is to harness the waiting periods at different transit stops in a worker transit route (WTR) for performing the tasks. Given the limited availability of workers' waiting periods at transit stops, task deadlines and workers' preference of performing tasks with higher rewards, we define the Transit-based Task Assignment (TTA) problem. The objective of the TTA problem is to maximize the average worker rewards for motivating workers, considering the fixed worker transit models. We solve the TTA problem by considering three variants, step-by-step, from offline to batch-based online versions. The first variant is the offline version of the TTA, which can be reduced to a maximum bipartite matching problem, and be leveraged for the second variant. The second variant is the batch-based online version of the TTA, for which, we propose dividing each batch into an offline version of the TTA problem, along with additional credibility constraints to ensure a certain level of worker response quality. The third variant is the extension of the batch-based online version of the TTA (Flexible-TTA) that relaxes the strict nature of the WTR model and assumes that a task with higher reward than a worker-defined threshold value will convince the worker to stay longer at the transit stop. Through our extensive evaluation, we observe that the algorithm solving the *Flexible-TTA problem* outperforms the algorithms proposed to solve other variants of the TTA problems, by 55% in terms of the number of assigned tasks, and by at least 35% in terms of average reward for the worker. With respect to the baseline (online task assignment) algorithm, the algorithm solving the *Flexible-TTA problem* results in three times higher reward and at least three times faster runtime.

CCS CONCEPTS

• **Information systems** → **Location based services**; **Geographic information systems**.

KEYWORDS

Spatial Crowdsourcing, Task Assignment, Transit

SSDBM 2020, July 7–9, 2020, Vienna, Austria

ACM ISBN 978-1-4503-8814-6/20/07.
https://doi.org/10.1145/3400903.3400929

ACM Reference Format:
Srinivasa Raghavendra Bhuvan Gummidi, Torben Bach Pedersen, and Xike Xie. 2020. Transit-based Task Assignment in Spatial Crowdsourcing. In *32nd International Conference on Scientific and Statistical Database Management (SSDBM 2020), July 7–9, 2020, Vienna, Austria.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3400903.3400929

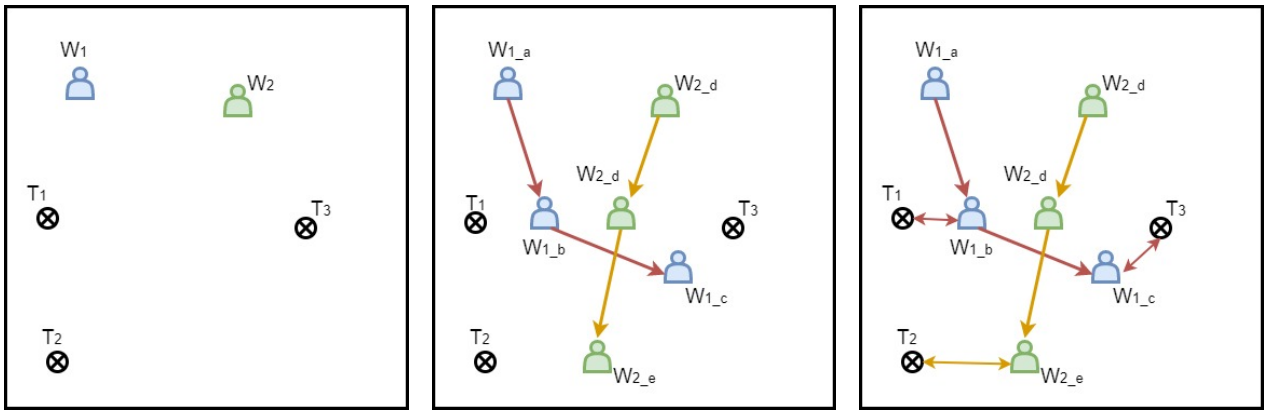
1 INTRODUCTION

Most of the existing spatial crowdsourcing (SC) assignment strategies do not consider the movement of workers in the spatio-temporal dimensions and assign tasks based on workers' current/ reported locations [8]. Assigning tasks to a worker at the right time is critical to the success of an SC application. Workers' movement information help us identify the right time for assigning a task to the worker. For example, consider the transit movement information of a worker. Intuitively, a worker following her daily transit route can perform tasks at the transit stops in the route, namely the origin stop, the intermediate stops (if any), and the destination.

This paper aims to target a new group of workers for spatial crowdsourcing, namely passengers in public transport, by harnessing their waiting periods at different transit stops in a regular worker transit route to offer an alternative strategy for the online task assignment in SC. Given the constraints of transport, the worker will try to strictly adhere to the transit route without any deviation or delay. Consequently, a task can only be assigned at a transit stop if the travel for performing the task does not affect the transit route of the worker, i.e., the worker will not miss the bus at the transit stop or be late to the destination. Among all the reachable tasks near a transit stop, the worker would like to choose the task with maximum reward to maximize her earnings. We assume that the SC-server allows the individual workers to register their transit routes or upload their daily travel data in exchange for better maximization of their reward calculation from the SC-server.

Example 1.1. Consider the example in Fig.1. There are two workers W_1 and W_2 and three noise data collection tasks T_1, T_2, *and* T_3 (See Fig. 1a). The transit routes of W_1 and W_2 before and after assignment can be seen in Fig. 1b & 1c, respectively. The worker travels to the assigned task from the transit stop and returns to the transit stop after recording the noise levels at the task location with her smartphone. For instance, worker W_1 travels from W_{1_b} to T_1 and returns back to W_{1_b} after performing the task to continue with the transit route.

To summarize, there is a need to develop new algorithms to solve the Transit-based task assignment (TTA) problem for harnessing the waiting periods at different transit stops in a worker transit route. Moreover, workers' preferences should be considered for improving the number of successful assignments. To improve the



(a) (b) (c)

Figure 1: Transit-based Task Assignment example.

quality of the task responses, we consider the worker credibility scores and employ a minimum worker credibility threshold constraint on tasks. Additionally, to maximize the workers' reward, we also model the case of flexible transit route which relaxes the strict nature of the worker transit model and assumes that a task with higher reward than a worker-defined threshold value will convince the worker to stay longer at the transit stop.. In this case, we assume that the routes are flexible, and she is willing to spend more time at a transit stop if the reward is high enough.

The TTA problem is resolved by considering different input models, offline and batch-based online versions. The offline version of the TTA can be reduced to a maximum bipartite matching problem, and be leveraged for the batch-based online input version. The batch-based online version of the TTA is solved by dividing each batch into an offline version of the TTA problem, along with additional credibility constraints to ensure a certain level of worker response quality. Furthermore, the batch-based online version of the TTA is extended (Flexible-TTA) to facilitate relaxation of the strict nature of the WTR model with an assumption that a task with a higher reward than a worker-defined threshold value will convince the worker to stay longer at the transit stop. We study the three versions of the TTA problem and propose algorithms to solve them.

The main contributions offered in this paper are:

- We present algorithms based on the Server-Assigned Task mode[9], to improve the task assignments by exploiting the workers' transit route information.
- We formulate the Transit-based Task Assignment (TTA) problem, that aims to maximize the average net reward received by a set of workers, considering transit stop time and deadline constraints.
- We prove that the offline version of TTA problem is reducible to the Maximum-weighted Bipartite Matching problem.
- We further study the online batch-based versions of the TTA using the offline version and propose algorithms to solve the problem.
- Additionally, we propose the *Credible Transit-based Task Assignment* algorithm to harness the worker credibility information to ensure a desired level of quality in the responses.
- We formulate the Flexible-TTA problem, that extends the TTA problem by allowing changes to the worker routes based on their threshold reward and maximum travel time constraints, and propose an algorithm to solve it.
- We test the applicability of the proposed algorithms through an extensive experimental evaluation based on the simulated worker transit routes and tasks in Aalborg, Denmark.

The remainder of the paper is organized as follows. In Section 2, we discuss a set of preliminaries in the context of transit-based SC. In Section 3, we formally define the offline and online batch-based versions of the *TTA* problem and explain our assignment algorithms for solving the batch-based TTA problem. Additionally, in the same section, we describe the *Credible Transit-based Task Assignment* algorithm in detail. Thereafter, in Section 4 we formally define the offline and online batch-based versions of the *Flexible-TTA* problem and explain our assignment algorithm for solving the batch-based Flexible-TTA problem. Section 5 presents the experimental results. In Section 6, we review the related work. Finally, in Section 7 we conclude and discuss the future directions of this study.

2 PRELIMINARIES

This section introduces some basic concepts that will be used throughout this paper. First, we define the worker with their worker transit routes.

Definition 2.1. (Worker): A worker, denoted by w, is a person willing to perform an assigned task by travelling to the task's location. Worker w has a transit stop set WTS that contains worker transit stops wts, belonging to the transit route she follows every day, threshold reward $thresRew$ representing the expected compensation for not following the fixed transit route wr, $strtTime$ represents the start of the transit trip, $maxTrvlTime$ represents the total time the worker is willing to spend on her daily route wr, $servRate$ represents the service price charged by the worker w

per hour, and *credibility* represents the worker credibility score. A worker is defined as:

$$w = < WTS, strtTime, thresRew, maxTrvlTime, servRate, credibility >$$
$$WTS = \{wts_o, ..., wts_i, ..., wts_d\}$$

, where wts_o represents the origin transit stop, wts_d represents the destination transit stop, and wts_i represents an intermediate transit stop of the worker transit route.

We define the credibility of the worker as the probability of worker performing an assigned task correctly. The credibility of the worker is defined similarly as the reputation score in [10]. The credibility of a worker can be determined based on the historical information of workers' answers stored in the SC-server. We assume that the worker credibility scores are stored and maintained at the SC-server.

Furthermore, we defined the worker route as a series of sequential worker transit stops. As mentioned earlier, our intuition is that the worker can perform tasks during their waiting period at the worker transit stops. Worker transit stops are associated with the real-world public transportation stop at a certain geographical location. Accordingly, we define worker transit stop as:

Definition 2.2. (Worker Transit Stop): A worker transit stop, denoted by wts, is at location l, and has $arrivalTime$ and $departure-Time$, that represent the arrival and departure timings at the transit stop of the worker w. $assignedTask$ represents the task, if assigned to the transit stop. The worker transit stop is defined as:

$$wts = < l, arrivalTime, departureTime, w, assignedTask >$$

We modeled the arrival time at the origin transit stop as the $strtTime$ of the worker w, and the departure time at the destination transit stop as the $strtTime + maxTrvlTime$ of the worker w.
We define a spatial task as a task associated with a geographical location. The spatial task definition is inspired by [9].

Definition 2.3. (Spatial task): A *spatial task*, denoted by t, contains a query q to be answered at location l. The query is asked at time $issueTime$ and will expire at time $expiryTime$. The task takes $taskDuration$ time to complete. The task will be guaranteed to result in a correct response, if a worker with at least $minWorkerCred$ credibility is assigned to the task.

$$t = < q, l, issueTime, expiryTime, taskDuration, minWorkerCred >$$

The worker has to visit location l to perform the task during the time interval between issue time and expiry time. Note that the worker has to visit the task location at least $taskDuration$ minutes before the $expiryTime$. For simplicity, we assume that the worker can complete the task with a single response. The task can still be assigned to the worker with less credibility than $minWorkerCred$. However, then the quality of the response is not guaranteed.

In [1], it is mentioned that tasks with "extrinsic incentives" like monetary reward would attract more workers, and affects the speed of accomplishing the task. We offer monetary rewards to workers for accomplishing the task. However, instead of a fixed reward per task, we define the reward associated with spatial tasks in proportion to the time spent by the worker to perform the task. We assume that there will be a base reward for performing the task. In addition, we assume that the workers expect a fixed hourly payment rate as compensation for the time spent on performing the task. The time spent is calculated based on the time taken to reach the task location from the worker transit stop, the time taken to do the task, and the time taken to return to the transit stop.

Definition 2.4. (Reward): The worker w will receive reward $r(w, t)$ after the completion of task t at transit stop $w.wts$. We assume that the reward r is affinely dependent on the distance between the transit stop's location $w.wts.l$ and the task's location $t.l$, and the task duration $t.taskDuration$.

$$r(w, t) = w.servRate * (2 * dist(w.wst.l, t.l)/walkingSpeed + t.taskDuration) + c$$

, where c is the fixed reward for all the tasks, for example 10 Kroners, $serviceRate$ is the fixed hourly compensation rate charged by the worker w, for example 60 Kroners per hour, and $walkingSpeed$ is the average walking speed of workers (m/s). For simplicity, we have assumed all the workers to have the same service rate.

Definition 2.5. (Distance from transit stop to task):
$dist(w.wst.l, t.l)$ denotes the distance that a worker w at a transit stop $w.wts$ needs to travel to reach t. Generally speaking, the distance from a transit stop to task denotes the walking distance from the worker transit stop to the task location.

For simplicity, we assume that a worker would perform at most a single task at every transit stop. Intuitively, a worker w cannot accept all the tasks without considering the additional travel time. Therefore, $maxTrvlTime$ is used to limit the amount of time a worker will spend on completing the transit route. The travel time is calculated based on the transit network. The transit info can be reliably extracted from the public transport web services and *Google Maps*. After the worker makes her task inquiry, the SC-server would then try to assign the tasks according to the worker and update her route.

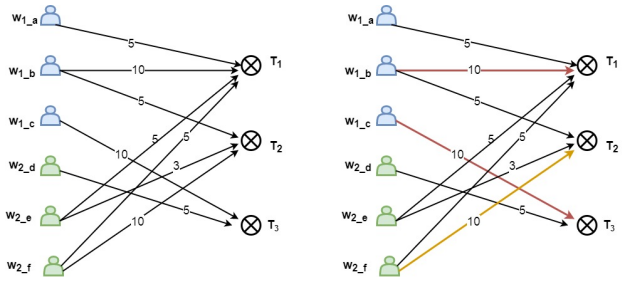


Figure 2: Reduction of Example 1 to MWBM problem

3 TRANSIT-BASED TASK ASSIGNMENT

3.1 Problem Definition

Given the different constraints, the objective of the SC-server is to maximize the net reward for the individual workers, received through performing assigned tasks. We consider two different input models for the TTA problem: offline and batch-based. In the offline model, all the tasks and the workers along with their transit stops

will be known beforehand to the SC-server. However, in the batch-based input model, the SC-server performs the task assignment for every incoming batch at regular time intervals. In the batch-based input model, the unassigned tasks along with the worker transit stops will be added to the next batch. For every new batch of workers and tasks, the SC-server tries to assign the newly available transit stops and tasks along with the unassigned and available, transit stops and tasks from the previous batch. Furthermore, in the TTA problem, **the worker transit route is considered fixed and will not be changed.** Considering these objectives & the input models, we define the offline TTA problem as:

Definition 3.1. (Offline Transit-based Task Assignment Problem): Assume an input set of tasks T and a set of workers W along with their set of worker transit stops WTS. The offline Transit-based Task Assignment (Offline-TTA) problem is an optimization problem with the objective to maximize the sum of the new rewards received by the individual workers. The Offline-TTA problem finds an optimal Transit Stop Task Assignment set, denoted by $TA(WST, T)$, with average net reward of r_{TA}, is a set of 3-tuples of the form $< wts, t, r >$, where wts is assigned to t with an associated reward r, given the following constraints are satisfied:

- Transit Stop Time Constraint: The time required to complete the task, i.e., travel time from the transit stop to the task location and returning to the transit stop and the *taskDuration* is less than the time spent at the transit stop, i.e.,
$$(2 * distance(wts.l, t.l)/walkingSpeed) + t.taskDuration < (wts.depatureTime - wts.arrivalTime)$$
- Task Deadline Constraint: The worker's arrival time at the transit stop should be at least $t.taskDuration$ before the task deadline , i.e.,
$$t.expiryTime \geq wts.arrivalTime + t.taskDuration$$

With the following theorem, we can solve the offline-TTA problem by reducing it to the maximum weighted bipartite matching problem.

Theorem 3.2. *The offline-TTA problem is reducible to the maximum weighted bipartite matching(MWBM) problem.*

Proof. We prove the theorem for a set of workers W with an associated set of worker transit stops $WTS = \{wts_1, wts_2, ..\}$ and a set of tasks $T = \{t_1, t_2, ..\}$. Let $G = (V, E)$ be an undirected graph whose vertices can be partitioned as $V = WTS \cup T$, where each transit stop wts_i maps to a vertex in WTS and each task t_j maps to a vertex in T. If the *task deadline* and *transit stop time constraints* are satisfied between wts_i and t_i, then there is an edge $e_i, j \in E$ connecting the vertex wts_i in WTS and vertex t_i in T. As every edge $e_i, j \in E$ has one end in W and another end in T, the graph G is a bipartite graph. We set the edge weight for every edge $e_i, j \in E$ to the reward r associated with task t and the worker transit stop wts. Since, a worker can perform only one task at each transit stop, $< wts_i, t_j >$ is a valid match only if both wts_i and t_j appear in at most one edge in E. Finally, the offline-TTA problem is solved by finding the maximum matching in weighted bipartite graph G. □

Fig. 2a depicts the bipartite graph G with weights for the example mentioned in Section 1. The left side set consists of the workers' transit points as nodes and the right side set contains the tasks as nodes. Fig 2b depicts the maximum weighted bipartite graph with maximum weighted edges highlighted.

Table 1: Example 2: Tasks and associated transit stops

Task	Issue Time	Expiry Time	Transit Stop	Reward	Worker Credibility Threshold
t_1	8:01 AM	5:00 PM	wts_1	20	0.7
t_2	8:15 AM	5:00 PM	wts_1	25	0.7
t_3	8:30 AM	5:00 PM	-	10	0.6
t_4	9:00 AM	5:00 PM	wts_3	20	0.6

Table 2: Example 2: Transit Stops

Stop	Arrival	Departure	Credibility	Threshold Reward
wts_1	8:00 AM	8:20 AM	0.6	30
wts_2	8:40 AM	9:00 AM	0.6	30
wts_3	9:20 AM	9:40 AM	0.6	30

Definition 3.3. (Online Batch-based Transit-based Task Assignment problem): Assume a set of batches B, with each incoming batch $b \in B$ comprising a set of unassigned tasks T_b and available workers W_b along with their transit routes WTS_b. The Online Batch-based Transit-based Task Assignment (Batch-TTA) problem is an optimization problem with the goal of finding an offline Transit Stop Task Assignment set $TA(WTS_b, T_b)$ that maximizes the net rewards(r_{TA}) received by the individual workers $w \in W_b$ at their worker transit stops $wts \in WTS_b$ by performing assigned tasks $t \in T_b$ for each incoming batch b.

We propose two algorithms to solve the Batch-TTA problem, namely, the Maximum Weighted Bipartite Matching-based(MWBM) and the Minimum Distance based Direct Assignment (DA) algorithms. The proposed algorithms follow a locally optimal assignment strategy as the SC-server has minimum knowledge of the availability of new workers and tasks in the subsequent batches [9]. The algorithms will be explained through the help of the following running example:

Example 2: Consider the scenario in Fig. 3a, where the worker w follows a transit route with three transit stops (wts_1, wts_2, wts_3) (The schedule, credibility scores and threshold reward values are mentioned in Table 2). There are four tasks sent to the SC-server (details are in Table 1). It can noticed that at stop wts_1, two tasks (t_1, t_2) satisfy the travel stop time constraint. However, task t_1 is issued before task t_2. Similarly, it can be noticed that none of the tasks satisfy the travel stop time constraint at transit stop wts_2. In the following subsections, we will observe how different task assignment algorithms would result in different assignments.

3.2 Maximum Weighted Bipartite Matching (MWBM) Algorithm

In MWBM algorithm, we solve the *Batch-based TTA* problem by dividing it into individual *offline-TTA* problems for each incoming batch of available workers and unassigned tasks. Thus, we can solve the batch-based TTA by solving the individual *offline-TTA* problems for each incoming batch. (See Algorithm 1). According to *Theorem* 3.2, the *offline-TTA* problem can be reduced to a maximum weighted bipartite matching problem. Therefore, we can employ

Algorithm 1: Max. Weighted Bipartite Matching

Input: An incoming batch b consisting of a non-emptyset of workers W_b with associated set of worker transit stops WTS_b with $thresRew$ as the minimum Threshold reward for flexible transits and $maxTrvlTime$ as maximum travel time, a set of tasks T_b. $TA(WTS, T)$ is the optimal set of assignments before the batch b. c is the fixed reward per task.

Output: The Optimal set of assignments $TA(WTS \cup WTS_b, T \cup T_b)$ with the average Reward r_{TA} and minimum travel distance $minDist$ for batch b

$TA(WTS \cup WTS_b, T \cup T_b) \leftarrow NULL$;
$TA(WTS_b, T_b) \leftarrow NULL$;
$WeightedGraph\ G \leftarrow NULL$;
foreach $t \in T_b$ **do**
  $G.addVertex(t)$;
foreach $wts \in WTS_b$ **do**
  $G.addVertex(wts)$;
  foreach $t \in T_b$ **do**
    $maxDistAtStop \leftarrow walkingSpeed * (wts.departureTime - wts.ArrivalTime - t.taskDuration)/2$;
    if $dist(wst.l, t.l) \leq maxDistAtStop \wedge t.expiryTime > wts.arrivalTime$ **then**
      $edgeWeight \leftarrow r(w, t)$;
      $G.addEdge(wts, t, edgeWeight)$;
$TA(WTS_b, T_b) \leftarrow MWBM.getMatching(G, WTS_b, T_b)$;
$TA(WTS \cup WTS_b, T \cup T_b) \leftarrow TA(WTS, T) \bigcup TA(WTS_b, T_b)$;
$removeAssignedAndExpiredTasks(T \cup T_b)$;
$removeAssignedWorkerStps(WTS \cup WTS_b)$;
return $TA(WTS \cup WTS_b, T \cup T_b)$

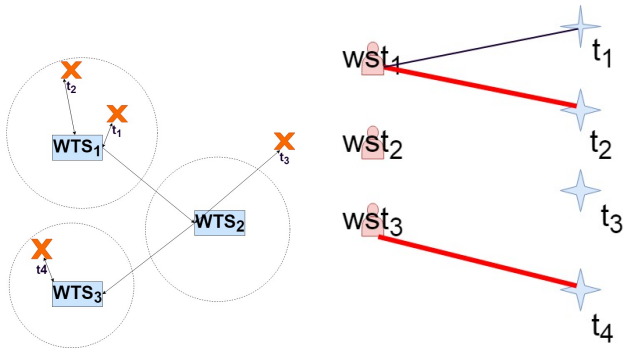
the maximum weight bipartite matching algorithm [11] to solve the *offline-TTA* problem. *MWBM* tries to solve the *batch-based TTA* by setting the task reward as the edge weight.

MWBM constructs the bipartite graph G and establishes edges between worker transit stop vertexes and tasks vertexes if they satisfy the task deadline and transit stop time constraints. Therefore for each worker transit stop, it has to search for all the tasks that satisfy the transit stop time and task deadline constraints to create all the potential edges. Consequently, the time complexity of this algorithm is directly dependent on the number of vertices n ($|WTS_b|+|T_b|$), the number of edges m and the batches p along with the time complexity of the maximum weight bipartite macthing algorithm, i.e., $O(n(m + n \log n) * p)$ [11].

Consider *Example* 2, MWBM constructs the bipartite graph with all the potential edges between the worker transit stops and the tasks. MWBM tries to find the edge with maximum weight for assigning tasks to transit stops (See Fig. 3b, where red edge represents assignment). For worker transit stop wts_1, t_2 is preferred over t_1 as t_2 offers higher reward.

3.3 Minimum Distance based Direct Assignment (DA) Algorithm

Generally, workers are more likely to accept tasks that are closer to them. However, the MWBM algorithm above does not try to prioritize tasks that are closer to the worker transit stop of the worker. Furthermore, the bipartite graph has to be constructed for every incoming batch of workers and tasks. The construction of the bipartite graph has the time complexity $O(t * (1 + w))$, where t and w represent the number of tasks and workers, respectively. The constructed bipartite graph size increases over time, with the addition of new incoming tasks and workers to the previously unassigned tasks and workers. Consequently, the construction of the bipartite graph and the subsequent matching for every incoming batch progressively becomes more time-consuming. Since the transit-based online task assignments need to be performed in real-time and there is a need to prioritize tasks that are closer, we propose a direct assignment-based algorithm (DA). Similar to the MWBM algorithm, DA algorithm also involves solving the batch-based TTA by breaking it into individual offline-TTA problems for each incoming batch. (See Algorithm 2). DA algorithm tries to achieve the objective of batch-TTA by maximizing the rewards received by



(a) Example 2 (b) MWBM Solution

Figure 3

Algorithm 2: Min. Distance-based Assignment

Input: Same as Algorithm 1
Output: Same as Algorithm 1

$TA(WTS \cup WTS_b, T \cup T_b) \leftarrow NULL$;
$TA(WTS_b, T_b) \leftarrow NULL$;
$maxReward \leftarrow 0$;
$minDist \leftarrow \infty$;
foreach $wts \in WTS_b$ **do**
  $maxDistAtStop \leftarrow walkingSpeed * (wts.departureTime - wts.ArrivalTime - t.taskDuration)/2$;
  $feasibleTask \leftarrow NULL$;
  foreach $t \in T_b$ **do**
    if $dist(wts.l, t.l) \leq maxDistAtStop \wedge t.expiryTime > wts.arrivalTime$ **then**
      if $minDist > dist(wts.l, t.l) \wedge maxReward < r$ **then**
        $minDist \leftarrow dist(wts.l, t.l)$;
        $maxReward \leftarrow r(w, t)$;
        $feasibleTask \leftarrow t$;
  $TA(WTS_b, T_b) \leftarrow TA(WTS_b, T_b) \bigcup TA(wts, feasibleTask)$;
Lines 14 – 16 *from MWBM Algorithm* 1 **return** $TA(WTS \cup WTS_b, T \cup T_b)$

workers and simultaneously tries to minimize the travel distance for the workers, with respect to tasks assigned.

The direct assignment algorithm tries to find the best feasible task at each worker transit stop that provides the maximum reward and involves minimum travel distance to the task. Therefore, it has to search for all the tasks that satisfy the distance and task deadline constraints before assigning the task to the transit stop. Consequently, the time complexity of DA algorithm is directly dependent on the number of worker transit stops (n), tasks (m) and the batches (p), i.e., $O(n * m * p)$.

Consider *Example* 2, the DA algorithm assigns the nearest task with greater reward to the transit stop (See Table. 3). For stop wts_1, t_1 is preferred over t_2 as t_1 offers better reward-to-distance ratio.

3.4 Credible Transit-based Task Assignment Algorithm (CTA)

In the previous algorithms, we have studied transit-based task assignment without ensuring a desired level of quality in the task responses from the workers. It was observed that workers might knowingly or unknowingly provide wrong answers to the queries associated with the spatial tasks [10]. To ensure a desired level of quality for the worker responses, we propose the *Credible Transit-based Task Assignment* algorithm. The algorithm solves the extended *Batch-TTA* problem (Definition 3.3) that includes the minimum worker credibility threshold (MWCT) constraint for the spatial task as the probability of the task being performed correctly. In other words, a spatial task can be assigned to the worker if and only if the worker's credibility is greater than or equal to the minimum worker credibility threshold of the spatial task.

The *CTA* algorithm is an extended DA algorithm for solving the updated Batch-based TTA problem with the minimum worker credibility threshold constraint (as defined below).

Definition 3.4. (Minimum Worker Credibility Threshold Constraint): The worker's credibility score should be atleast the task's *MWCT*.

$$w.credibility \geq t.minWorkerCred \quad (1)$$

The CTA algorithm tries to find the best feasible task at each worker transit stop that satisfies all the constraints, including the *MWCT* constraint and involves maximum reward and minimum travel distance to the task. Therefore, it has to search for all the tasks that satisfy the credibility, distance, and task deadline constraints before a task is assigned to a worker. Consequently, the time complexity of the CTA algorithm is directly dependent on the number of worker transit stops (n), tasks (m) and the batches (p), i.e., $O(n * m * p)$.

Consider *Example* 2, the CTA algorithm assigns the nearest task that satisfies the constraints with greater reward to the transit stop (See Table. 4). Only one assignment wts_3, t_4 satisfies the credibility constraint, i.e., worker credibility is at least equal to the *MWCT*.

4 FLEXIBLE TRANSIT-BASED TASK ASSIGNMENT

4.1 Problem Definition

In the previous section, we have studied task assignment for fixed transit routes. However, it was observed that workers would modify their route for performing tasks, if a lucrative incentive is offered. In our context, we assume that if a task with higher reward than a certain threshold value will convince the worker to stay longer at the transit stop, and perform the task, despite the travel stop time constraint. Considering this, we propose the Flexible TTA problem, where **the worker transit route is no longer considered fixed and can be changed.** A worker w can search for a task t with a reward greater than the threshold, that will result in prolonging the stay at the worker transit stop wts, i.e., the *departureTime* at wts will be changed. Consequently, the arrival and departure times of next transit stops in the route will be changed. We assume that a flexible transit stop task assignment can happen at a transit stop wts if and only if the worker cannot find a task satisfying the travel

Algorithm 3: Credible Transit-based Task Assignment (CTA)

Input: Same as Algorithm 1
Output: Same as Algorithm 1
8 $AlgorithmSameAsDAexceptLine9ReplacedByBelowPsuedocode$
9 **if** $dist(wts.l, t.l) \leq maxDistAtStop \wedge t.expiryTime > wts.arrivalTime \wedge t.minWorkerCred \leq w.credibility$ **then**
10 **return** $TA(WTS \cup WTS_b, T \cup T_b)$

Algorithm 4: Flexible Direct Assignment

Input: Same as Algorithm 1, except $FTA(WTS, T)$ is the optimal set of assignments instead of $TA(WST, T)$
Output: Same as Algorithm 1, except with $FTA(WTS \cup WTS_b, T \cup T_b)$
8 *Same as DA Algorithm 2, except lines 9 – 13 replaced by below psuedocode*
9 **if** $dist(wts, t) \leq maxDistAtStop \wedge t.expiryTime > wts.arrivalTime$ **then**
10 $Lines\ 10-13\ from\ DA\ Algorithm\ 2$
11 **else if** $w.threshold \geq r$ **then**
12 $wts.departureTime \leftarrow wts.arrivalTime + (2 * dist(wts, t)/WalkingSpeed)$;
13 $ReconstructRoutes(wts, w.destination, wts.departureTime)$;
14 **if** $ReconstructedRouteDestinationTime < startTime + w.maxTravelTime$ **then**
15 $Lines\ 10-13\ from\ DA\ Algorithm\ 2$
16 **return** $FTA(WTS \cup WTS_b, T \cup T_b)$

Table 3: Eg. 2 DA Solution

Stop	Task Assigned	Reward
wts_1	t_1	20
wts_2	None	0
wts_3	t_4	20

Table 4: Eg. 2 CTA Solution

Stop	Task Assigned	Reward
wts_1	None	0
wts_2	None	0
wts_3	t_4	20

Table 5: Eg. 2 Flexible-DA Solution

Stop	Task Assigned	Reward	New Arrival
wts_1	t_1	20	NA
wts_2	t_3	35	NA
wts_3	t_4	20	9:45 AM

stop time constraint at the worker transit stop. Considering these constraints, we define the offline Flexible-TTA problem as:

Definition 4.1. (Offline Flexible Transit-based Task Assignment problem): Assume an input set of spatial tasks T and a set of workers W along with their set of worker transit stops WTS. The offline Transit-based Task Assignment (Offline Flexible-TTA) problem is an optimization problem with an objective to maximize the net rewards of individual workers. Offline Flexible-TTA finds an optimal Flexible Transit Stop Task Assignment set, denoted by $FTA(WTS, T)$ with net reward r_{FTA}, is a set of 3-tuples of the form $< wts, t, r >$, where wts is assigned to t with an associated reward r, given the following constraints are satisfied:

- Threshold Reward Constraint : The reward r should be greater than the worker w's $thresRew$, i.e., $r > w.thresRew$
- Task Deadline Constraint: The task deadline should be later than the worker's arrival time at the transit stop, i.e., $t.expiryTime > wts.arrivalTime$
- Travel Time Constraint: The new reconstructed transit route with the updated $wts.departureTime$ at transit stop wts should allow the worker w to reach the destination before $w.startTime + w.maxTrvlTime = destThresholdTime$, i.e.,

$$destThresholdTime > wts.arrivalTime + t.taskDuration + (2 * dist(wts.l, t.l)/walkingSpeed)$$

Definition 4.2. (Batch-based Flexible Transit-based Task Assignment Problem): Assume a set of batches B, with each incoming batch $b \in B$ comprising a set of unassigned tasks T_b and available workers W_b along with their transit routes WTS_b. The Batch-based Flexible Transit-based Task Assignment (Batch-based Flexible-TTA) problem is an optimization problem with the goal of finding an optimal Flexible Transit Stop Task Assignment $FTA(WTS_b, T_b)$ that maximizes the net rewards (r_{FTA}) received by the individual workers $w \in W_b$ at their worker transit stops $wts \in WTS_b$ by performing assigned task $t \in T_b$ for every batch b.

4.2 Flexible Transit Route-based Direct Assignment Algorithm

We propose an extended DA algorithm to solve the Batch-based Flexible-TTA problem. In this *Flexible Transit Route-based Direct Assignment* algorithm (Flexible-DA), we consider the flexible transit route scenario, where the worker transit routes can be changed; for example, the worker might arrive late to the final destination. We assume that the worker will change her transit route if she fails to find a task that satisfies the travel stop time constraint at the transit stop and if a task with a threshold reward $thresRew$ is present in the vicinity of the transit stop, resulting in a delay in the departure time at the transit stop. Flexible Transit Route-based assignments involve validation of the new departure timings at worker transit stops and to check whether the worker can reach the final destination before exhausting the maximum travel time $maxTrvlTime$ after performing a potential task. Therefore, it utilizes a transit routing service (for example, Rejseplanen (rejseplanen.dk) to perform these checks, involving a REST-based API call to the routing service. Given, the high cost of the REST-based API (each request costs 0.7 seconds) calls, we do not extend the MWBM algorithm as it involves the construction of the complete bipartite graph resulting in a massive amount of requests to the routing service. Instead, we extend the DA algorithm to include the FTA assignments.

In Flexible-DA algorithm, we solve the Batch-based Flexible-TTA problem, similar to the Batch-TTA problem by breaking into individual *Offline Flexible-TTA* problems for every incoming batch (See Algorithm 4). This algorithm tries to achieve the batch-based Flexible-TTA to maximize the rewards received by workers and simultaneously seeks to reduce the travel distance to the tasks by the workers. Additionally, whenever a worker transit stop cannot find a task adhering to the fixed route, the worker expands the search space by looking for tasks beyond the transit stop time constraint which satisfies her threshold reward value. If a task with higher reward than the threshold is found, then Flexible-DA reconstructs the route with the new delayed departure time from the worker transit stop. If the reconstructed route reaches the final destination before exhausting the $maxTrvlTime$, then the matching will be valid, and the task t is assigned to the stop wts.

Consider *Example* 2, the Flexible-DA algorithm assigns the nearest task with greater reward to the transit stop (See Table. 5). Furthermore, if a transit stop has no tasks satisfying its transit stop time constraint, the Flexible-DA algorithm searches further than the transit stop time constraint for tasks offering reward above the threshold. In this case, at transit stop wts_2, no tasks are within the vicinity. Consequently, the worker transit stop will be assigned to task t_3, that offers more than the threshold reward expected by the worker, i.e., 35. However, due to travelling longer than expected, the worker will miss the transit that departs at 9:00 AM, and will have to take the next bus to reach the next transit stop wts_3, reflected by the new arrival time 9:40 AM in Table 5.

5 EXPERIMENTAL EVALUATION

5.1 Experiment Setup

Workers Transit Routes DataSet: It is hard to find real datasets that reflect the workers and their transit movement information in the real world. Consequently, we have used realistic datasets for generating workers ranging from 1K to 25K in number. The worker transit routes are generated by identifying the residential and commercial zones in the city of Aalborg, Denmark. Two periods were set for workers going from homes in residential zones to workplaces in commercial zones in the morning "7:00 to 11:00" and workers returning from work to home in the evening "16:00 to 21:00". The resulting set contained the origin, destination and start time information for each worker. We constructed the worker transit routes by using that information to create a REST API call to the Rejseplanen Public Transport Routing Service, Denmark. The routing service returns the transit details from the given origin and destination transit stops, that includes the arrival and departure at each transit stop, excluding the destination. At the destination, only the arrival is returned. The workers are generated with uniform values of $thresRew$ and $maxTrvlTime$ parameters. The workers

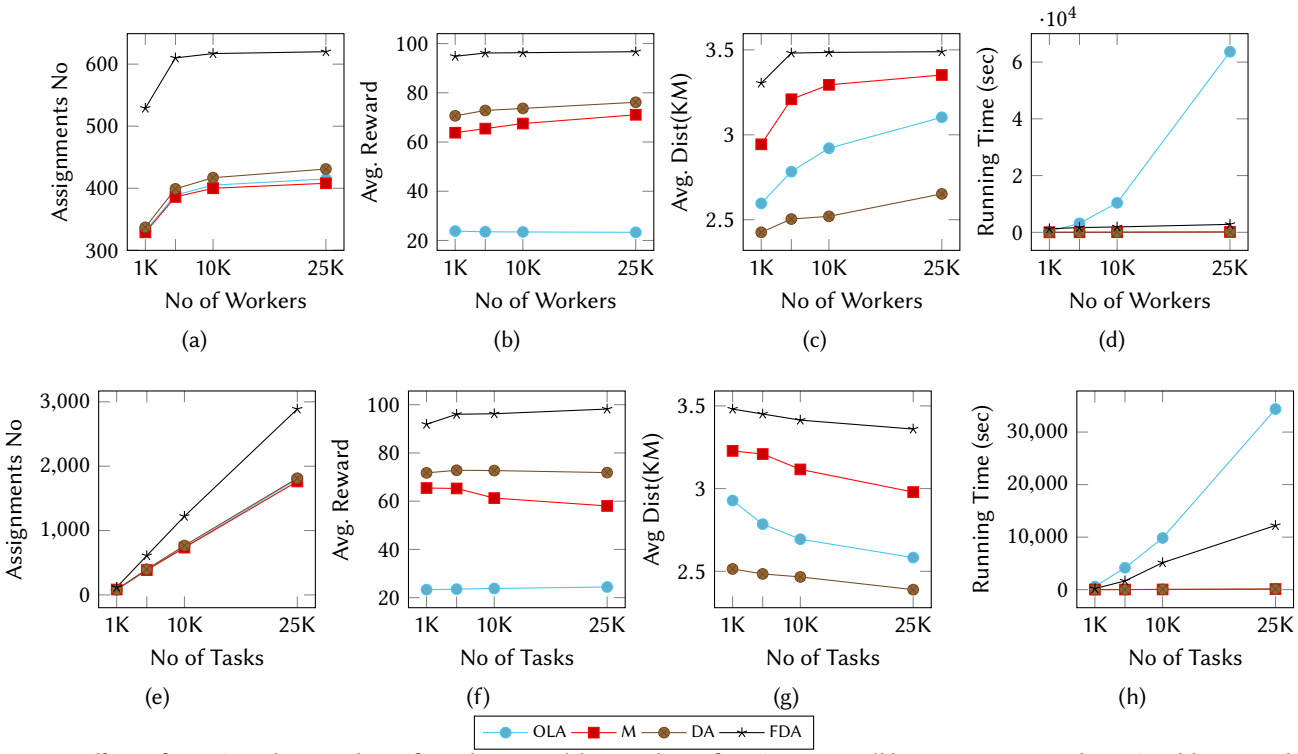


Figure 4: Effect of varying the number of workers on :(a). Number of Assignments.(b) Average Reward received by a worker (c). Average Distance travelled by a worker (KM) (d). Average Running Time (seconds). Effect of varying the number of tasks on :(e). Number of Assignments.(f) Average Reward received by a worker (g). Average Distance travelled by a worker (KM) (h). Average Running Time (seconds).

credibility parameters are generated with random values between 0.2 to 0.9, due to the unavailability of workers' historical information. As the credibility values are dependedent on the correctness of the worker responses and independent of other factors like geographical area, etc., we believe that an uniform distribution would reflect the real-world scenario.

Synthetic Dataset for Tasks: For the synthetic dataset, we have generated a varied number of tasks from 1K to 25K. The tasks are distributed randomly in the geographical extent of the Aalborg City, Denmark. The tasks are generated with varying values of *issueTime* between "7:00 to 21:00". The default *expiryTime* is set as twenty four hours from the *issueTime*. The tasks are generated with equal values of minimum worker credibility threshold parameters.

Algorithms: We have conducted our evaluation based on the assignment algorithms presented in this paper. First, we have evaluated the fixed transit route algorithms like *MWBM*, *DA*, *CTA* and the flexible transit route algorithm *Flexible-DA*. Our algorithms are based on the batch-based input model.

Baseline Algorithm: As a baseline, we consider the online input model, where the worker/ task arrives dynamically to the system. The SC-server will not have any prior information about the WTRs in the online input model. We assume that the worker would notify the SC-server whenever she is available to perform a task. In our transit-based context, the worker will notify the SC-server whenever she reaches a transit stop. The SC-server tries to assign a task once a worker becomes available immediately. The baseline online transit-based task algorithm is denoted by *OLA* (See Algorithm 5).

With the arrival of each new task or worker, the search space of the OLA algorithm grows and simultaneously shrinks with every assignment. Consequently, the time complexity of the OLA algorithm is directly dependent on the number of worker transit stops (n), and tasks (m), i.e., $O(n * (m^2) + (n^2) * m)$.

Configuration and Measures: We compare the different assignment algorithms based on the following measures: Number of Assigned tasks to the workers, Average travel distance for the assigned task, Average reward per worker, and the running time. We vary the number of workers from 1K to 25K and the number of tasks from 1K to 25K to evaluate the scalability of our algorithms. To simulate a real application scenario, we simulate a batch of workers and tasks every hour. The batches contain varying sizes of workers and tasks as they are randomly generated during different periods. For example, 25K worker dataset has a maximum batch size of 3184, a minimum of 2437, a mean of 2777 and a median size of 2600. Similarly, 25K tasks dataset has a maximum batch size of 381, a minimum of 328, a mean of 357, and a median size of 358. The CTA algorithm is evaluated by varying the number of workers (1K to 25K), the number of tasks (1K to 25K), and the minimum worker credibility threshold constraint (0.5 to 0.9). Similarly, we evaluate the effect of varying the *maxTrvlTime* parameter of

Algorithm 5: Online Transit-based Task Assignment (OLA)

Input: An incoming task t or a recently available worker w at transit stop wts along with available workers W at worker transit stops WTS, and a set of previously unassigned tasks T. . c is the fixed reward per task.
Output: Task Assignment $< w, wts, t >$ with reward $maxReward$ and minimum travel distance $minDist$

```
1  maxReward ← 0;
2  minDist ← ∞;
3  removeExpiredTasks(T);
4  removeUnavailbleWorkers(W);
5  if New Task Arrival t then
6      T ← T ⋃ {t};
7      foreach wts ∈ WTS do
8          maxDistAtStop ← walkingSpeed ∗ (wts.departureTime −
             wts.ArrivalTime − t.taskDuration)/2;
9          if dist(wts.l, t.l) ≤ maxDistAtStop ∧
             t.expiryTime > wts.arrivalTime then
10             minDist ← dist(wts.l, t.l);
11             maxReward ← r(w, t);
12             Assignment < w, wst, t >;
13             T ← T \ {t};
14             W ← W \ {w};
15             WTS ← WTS \ {wts};
16 else if New Worker Arrival w at wts then
17     W ← W ⋃ {w};
18     WTS ← WTS ⋃ {wts};
19     maxDistAtStop ← walkingSpeed ∗ (wts.departureTime −
         wts.ArrivalTime − t.taskDuration)/2;
20     foreach t ∈ T do
21         if dist(wts.l, t.l) ≤ maxDistAtStop ∧
             t.expiryTime > wts.arrivalTime then
22             minDist ← dist(wts.l, t.l);
23             maxReward ← r(w, t);
24             Assignment < w, wst, t >;
25             T ← T \ {t};
26             W ← W \ {w};
27             WTS ← WTS \ {wts};
28 return Assignment < w, wst, t >
```

the workers(from 1 to 3 hours) to evaluate its effect on the above-mentioned different measures. Furthermore, we evaluate the effect of varying the threshold reward parameter of workers to determine its effect on the above-mentioned measures. The default values for the scalability experiments are depicted in bold in the *Experiment Parameters* Table 6. All algorithms were implemented in Java utilizing Postgresql with PostGIS and pgrouting extensions and Jgrapht library[12]. All experiments were conducted on the Windows 8.1 OS with Intel Core i7-5600 CPU@ 2.60G HZ and 12 GB memory.

5.2 Scalability with the size of workers data set

In this set of experiments, we evaluated the scalability of our proposed transit-based assignment algorithms by varying the number of workers from 1K to 25K. Figure 4a illustrates the effect of the varying the number of workers on the number of assigned tasks directly. With respect to the number of assigned tasks, the *Flexible-DA* has outperformed the other algorithms. When compared to the *OLA, MWBM, and DA* algorithms, the number of worker transit stop assignments has increased more than 55%. The reason is that *Flexible-DA* supports flexible transit routes, which can bypass the transit stop time constraint in cases where are no feasible tasks at the transit stops. *DA* has performed marginally better than the OLA and MWBM algorithms.

Table 6: Experiment Parameters

Parameters	Value Range
Number of workers (Transit stops)	1K (2419),**5K (12303)**, 10K (24407),25K (60941)
Number of Tasks	1K,**5K**,10K,25K
Maximum Travel Time of worker	1,**2**,3
Worker credibility	Randomly generated between 0.2 - 0.9
Minimum worker credibility threshold of task	0.5,0.6,**0.7**,0.8,0.9
Worker's Threshold Reward	70,**80**,90
Average Walking Speed	1.46 m/s [6]

With regard to the average reward received by the worker, *Flexible-DA* performs better than the other algorithms (See Fig. 4b). The average reward received by the worker in the *Flexible-DA* algorithm has increased by 35% when compared to *DA*, and around 50% when compared to *MWBM*. *Flexible-DA* results in three times higher reward than the baseline algorithm *OLA*. The reason is that in the case of *Flexible-DA*, the SC-server has the knowledge about the worker's future spatiotemporal movement that facilitates the worker to extend the stay at different transit stops of her route. Furthermore, *MWBM and DA* result in nearly three times higher reward than the baseline algorithm *OLA*. *Flexible-DA* performs better due to the availability of tasks with rewards higher than the threshold, that satisfies the maximum travel time constraint. *DA* delivers 15% better average reward than *MWBM*. With respect to the average distance travelled by the worker, *DA* outperforms *MWBM* by 20%, *OLA* by 30% and *Flexible-DA* by 50% (See Fig. 4c). The reason is that *DA* gives more priority to tasks that are closer to the transit stop. Furthermore, with regard to the running time, the baseline algorithm *OLA* has the worst performance among the other algorithms. Due to the increase of search space with every new arrival of worker/ task, a substantial increase of running time is observed for the *OLA* algorithm (See Fig. 4d). *Flexible-DA* has the worst performance among the proposed algorithms, due to the REST API calls for validation and route reconstruction activities. The fixed-route based algorithms (*MWBM, and DA*) is at least 40-times faster than the baseline *OLA*. The proposed algorithms (*MWBM, and DA*) have almost similar performance as the input size increases.

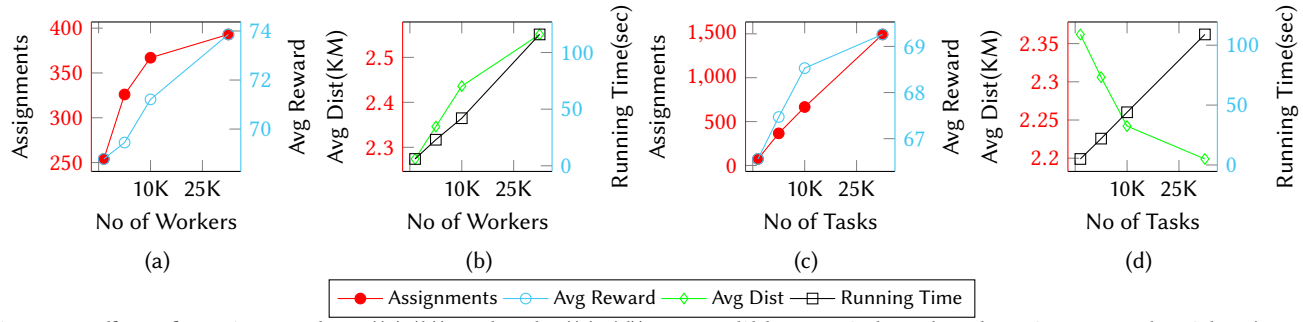


Figure 5: Effect of varying workers ((a),(b)) and tasks ((c), (d)) on *Credible Transit-based Task Assignment* Algorithm (*CTA*).

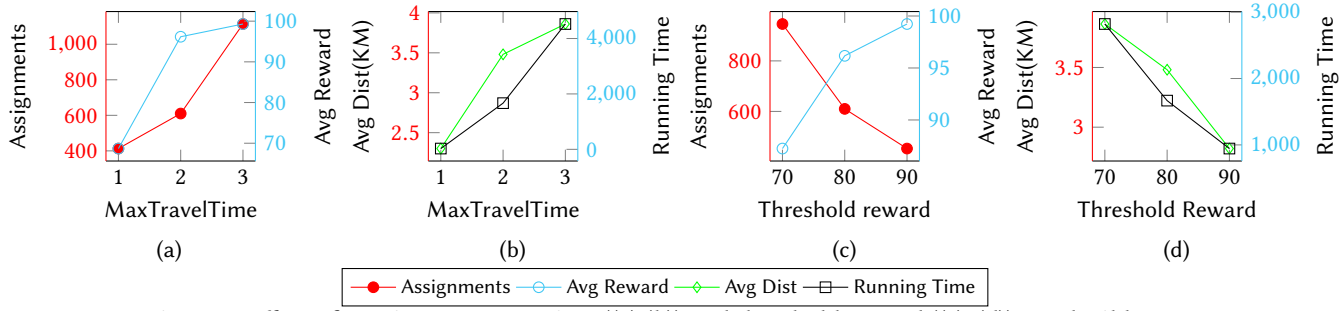


Figure 6: Effect of varying maxTraveTime ((a),(b)) and threshold reward ((c), (d)) on *Flexible-DA*.

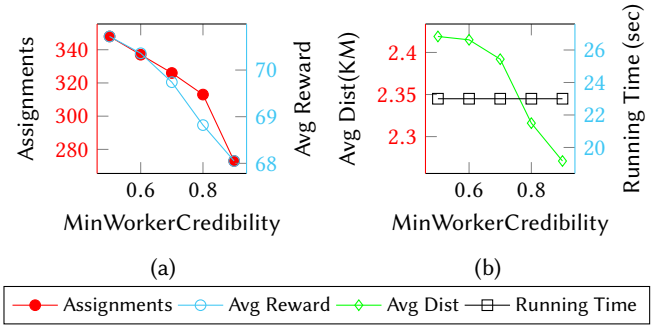


Figure 7: Effect of varying minimum worker credibility threshold ((a),(b)) on *Credible Transit-based Task Assignment Algorithm*.

5.3 Scalability with the size of Tasks data set

In this set of experiments, we evaluated the scalability of our proposed algorithms by varying the tasks from 1K to 25K. Figure 4e illustrates the effect of varying tasks on the number of assigned tasks directly. Regarding the number of assigned tasks, the *Flexible-DA* has outperformed the other algorithms, and the increase is more evident as the number of tasks increases. When compared to *OLA, MWBM, and DA* algorithms, the number of worker transit stop assignments has increased more than 60%. As *Flexible-DA* supports flexible transit routes, the workers can travel further by delaying their departure time to gain more tasks.

Regarding the average reward received by the worker, *Flexible-DA* has resulted in four times higher reward than the baseline algorithm *OLA*, 35% better than *MWBM* algorithm, and around 20% better than *DA* algorithm (See Fig. 4f). *MWBM, and DA* result in around three times higher reward than the baseline algorithm *OLA*. With respect to the average distance travelled by the worker, *DA* outperforms *Flexible-DA* by 50%, *OLA* by 20%, and *MWBM* by 30% (See Fig. 4g). The reason is that *DA* gives more priority to tasks that are closer to the transit stop. Furthermore, with regard to the running time, the baseline algorithm *OLA* has the worst performance among the other algorithms. Due to the increase of search space with every new arrival of worker/ task, a substantial increase of running time is observed for the *OLA* algorithm (See Fig. 4h). *Flexible-DA* has the worst performance among the proposed algorithms, due to the REST API calls for validation and route reconstruction activities with each call costing 0.7 seconds. The other proposed algorithms (*MWBM, and DA*) have almost similar performance as the input size increases.

5.4 Effect of varying the number of workers and the number of tasks on CTA

In this set of experiments, we evaluated the *Credible Transit-based Task Assignment Algorithm (CTA)* by varying the number of workers and the number of tasks. Fig. 5a shows the effect of varying the number of workers on the number of assignments and the average reward received by the worker. As the evaluation of CTA is based on additional parameters like worker credibility and minimum

worker credibility threshold of the task, it cannot be directly compared with the other methods proposed in this paper. Instead, this sub-section focuses exclusively on the special aspects of the CTA algorithm. An upward trend can be observed regarding the number of assignments, and the average reward as the number of workers increases from 1K to 25K. The upward trend can be attributed to the increased availability of workers with higher credibility for task assignment. However, it can be noticed that the jump in the number of assignments when the workers increase from 5k to 10K is sharper than when the workers increase from 10k to 25K. The reason is that the majority of the new workers with credibility higher than the minimum worker credibility threshold (0.7) are not close to the tasks, thereby failing the distance and deadline constraints.

Similarly, Fig. 5b shows the effect of varying the number of workers on the average distance travelled by the worker for performing the assigned tasks and the running time. It can be observed that the average travel distance for a worker and the running time increases gradually as the number of workers increases. The average travel distance for a worker increases due to different reasons like the increase in the number of tasks assigned to a single worker thus, adding extra travel distance, and the increase in the assignment of tasks that are located relatively far from the transit stop.

Fig. 5c shows the effect of varying the number of tasks on the number of assignments and the average reward received by the worker. It can be observed that as the number of tasks increases, the number of assignments, and the average reward increases gradually. The reason is that more tasks within the close proximity of transit stops satisfying deadline constraint are available for assignment. The gradual increase of the number of assignments can be attributed to the uniform distribution for tasks generation.

Similarly, Fig. 5d shows the effect of varying the number of tasks on the average distance travelled by the worker for performing the assigned tasks and the running time. It can be observed that the average travel distance per worker decreases as the number of tasks increases; in contrast, the running time increases. The average travel distance for a worker decreases due to the availability of more tasks in the near vicinity of the workers' transit stops satisfying the different constraints.

5.5 Effect of varying maxTravelTime, threshold reward on Flexible-DA

In this set of experiments, we evaluated the *Flexible-DA* by varying the maxTravelTime and the threshold reward values of the worker. We considered only the *Flexible-DA* as the other algorithms are not impacted by the maxTravelTime and threshold reward values. It can be observed from Fig. 6a and Fig. 6b, that as the maxTravelTime value increases, the number of assignments, average travel distance, average reward and the running time increases. The reason is that as the maxTravelTime increases as more tasks become eligible for the workers to perform in the *Flexible-DA* algorithm.

Similarly, in Fig. 6c and Fig. 6d, it can be observed that the increase in the threshold reward value of the worker, results in a decrease in the number of assignments, the average travel distance and the average running speed. The reason is that as the threshold reward increases, fewer tasks will be eligible for the flexible transit route-based task assignments. However, the average reward value increases as the threshold reward increases as the priority will be given to tasks with higher rewards.

5.6 Effect of varying minimum worker credibility threshold value on *CTA*

In this set of experiments, we evaluated the *CTA* by varying the minimum worker credibility threshold (MWCT) values of the task. We considered only the *CTA* as the *MWCT* values do not impact the other algorithms. It can be observed from Fig. 7a and Fig. 7b, that as the *MWCT* value increases, the number of assignments, the average travel distance, and the average reward showcases a downward trend. Given that the number of worker and the number of tasks are fixed at 5K, the number of potential assignments with the satisfied credibility constraint will be reduced as the *MWCT* value increases. For example, when the *MWCT* value is increased to 0.6 from 0.5, the workers with credibility values ranging between $[0.5, 0.6)$ becomes ineligible to perform tasks owing to the credibility constraint. The running time is not impacted by the increase of *MWCT* value.

5.7 Summary

We found out that the Flexible-DA outclasses the fixed transit-based algorithms in the measure of the number of assigned tasks and the average reward. However, the *Flexible-DA* is highly time-consuming than the fixed transit-based algorithms. With respect to the baseline online task assignment algorithm (*OLA*), *Flexible-DA* results in three times higher reward and at least three times faster runtime. Similarly, *DA* outclasses the other fixed transit-based algorithm (*MWBM*) in the measure of the average travel distance. With regard to the credibility-based assignment algorithm, the measures of assigned tasks, the average reward for an assigned task, and the running time increases gradually as the number of workers and the number of tasks increase. With regard to the average travel distance, *CTA* results in a gradual increase as the number of workers increases, and a gradual decrease as the number of tasks increases. Furthermore, we noticed a downward trend for all the measures for *CTA* as the minimum worker credibility threshold values increase.

6 RELATED WORK

Spatial Crowdsourcing (SC) [8] harnesses the potential of a crowd to perform real-world spatial tasks that are not supported by conventional crowdsourcing techniques. Typically, the workers in SC move to the tasks' locations to perform tasks. The SC-server supports two types of task publishing modes [9]: Server-Assigned Task (SAT) publishing mode and Worker Selected Task (WST) publishing mode. Due to the level of control exerted by the SC-server in the SAT publishing mode, research gained momentum in SC literature related to SAT publishing mode [5, 7, 14]. The SAT publishing mode, in the context of assignment of workers to tasks, involves the problem of SC-server choosing the workers for the tasks. Typically, the workers and tasks arrive dynamically to the SC-server, thereby leading to uncertainty in the task assignment process. This scenario is termed as online task assignment problem [9, 14–16]. Typically, these assignment problems aim to achieve optimization

goals like maximizing the number of tasks assigned[9, 14], minimizing the cost incurred by the server[16], improving the quality of task responses[3] or goals benefitting the workers like maximizing the reward received by the worker[2].

To reduce the uncertainty in the online task assignment scenario, SC-server can exploit the workers' movement information to identify the workers' arrival order [8]. There are some existing works [2, 4, 13], that try to harness the workers' movement information. For instance, [4] tries to solve a single-worker-multiple-tasks type of task scheduling problem. They assume that there are a set of available tasks that have to be incorporated into the worker's route based on her budget on a detour. They combine the two objectives of minimize detour and maximize reward by using the skyline queries (finding the set of non-dominated paths). They prove the problem to be NP-hard by reducing it to the Travelling Salesman Problem and proposes exact and heuristics-based approximate solutions for solving it.However, they do not consider the dynamic nature of workers and tasks arrival. Furthermore, they do not consider some temporal aspects like time required to perform tasks and maximum travel time that a worker can afford. We try to address the dynamic nature by proposing a task assignment problem with a batch-based input model of workers' transit routes and tasks' arrival. Additionally, we also consider the time required to perform tasks and the maximum travel time of the worker.

7 CONCLUSIONS AND FUTURE WORK

In this paper, we proposed a task assignment model that exploits the workers' transit information to offer an alternative strategy for the online spatial task assignment in SC. This paper modeled the potential task assignment opportunities in a fixed worker transit route that is followed strictly. Additionally, we modeled a flexible transit route scenario with an assumption that the worker would be willing to delay her trip if a task offers more than a threshold reward. We defined the three variants of the transit-based task assignment (TTA) problem, *offline-TTA, Batch-TTA, and Flexible-TTA*, with a goal to maximize worker rewards considering the fixed and the flexible worker transit models, respectively.

We prove that the offline version of the TTA problem can be reduced to a maximum weighted bipartite matching problem. We utilize the offline version of the defined problem to solve the online batch-based versions of them. Two algorithms are proposed for solving the Batch-TTA problem; namely, *MWBM* and *DA*. Additionally, to ensure a certain level of worker response quality, we proposed a *CTA* algorithm considering the worker credibility information for task assignment. *CTA* assigns tasks to workers that satisfy the minimum worker credibility threshold constraint. Furthermore, for the Batch-based Flexible-TTA problem, we proposed an extended version of the *DA*, *Flexible-DA* considering the flexible worker transit model. We compared our proposed algorithms to a baseline algorithm, *OLA* that models the online assignment without considering the routing information. Through our extensive evaluation, we observed that the *Flexible-DA* outperforms the other proposed algorithms by 55% in terms of the number of assigned tasks, and at least 35% in terms of average reward for the worker. With respect to the baseline algorithm *OLA*, *Flexible-DA* algorithm results in four times the higher reward and *DA, and MWBM* algorithms result in nearly three times the higher reward. *DA* outperforms the other algorithms by at least 20% in terms of average travel distance. With respect to the running time, the fixed-route based algorithms (*MWBM, and DA*) is at least 40-times faster than the baseline *OLA*. The *Flexible-DA* algorithm is slower than the fixed-route based algorithms due to the REST API calls and route reconstruction activities. However, *Flexible-DA* is at least three times faster than the baseline *OLA* with respect to runtime.

There are several promising directions for future work. First, we need to consider promising worker movement models to identify more real-world cases to improve the chances for the tasks to be assigned. Second, we need to consider new worker movement models that can relax the immutability concept of task assignments, i.e., workers could have the choice to exchange their assigned task for a newly available task with a higher reward. Furthermore, we need to compare our assignment model against a model where the workers can bid for the tasks to validate the time-effectiveness of bidding versus server-assignment in a moving worker scenario.

ACKNOWLEDGMENTS

This research has been funded in part by the European Commission through the Erasmus Mundus Joint Doctorate *Information Technologies for Business Intelligence - Doctoral College* (IT4BI-DC). Xike Xie is supported by the CAS Pioneer Hundred Talents Program, Fundamental Research Funds for the Central Universities, National Natural Science Foundation of China (No. 61772492), and Natural Science Foundation of Jiangsu Province (No. BK20171240).

REFERENCES

[1] Mohammad Allahbakhsh, Boualem Benatallah, Aleksandar Ignjatovic, Hamid Reza Motahari-Nezhad, Elisa Bertino, and Schahram Dustdar. 2013. Quality control in crowdsourcing systems: Issues and directions. *IEEE Internet Computing* 17 (2013), 76–81.
[2] Cen Chen, Shih-Fen Cheng, Aldy Gunawan, Archan Misra, Koustuv Dasgupta, and Deepthi Chander. 2014. Traccs: A framework for trajectory-aware coordinated urban crowd-sourcing. In *HCOMP*. 30–40.
[3] Peng Cheng, Xiang Lian, Lei Chen, and Cyrus Shahabi. 2017. Prediction-based task assignment in spatial crowdsourcing. In *ICDE*. IEEE, 997–1008.
[4] Camila F. Costa and Mario A. Nascimento. 2018. In-route Task Selection in Crowdsourcing. In *GIS*. 524–527.
[5] Dingxiong Deng, Cyrus Shahabi, and Linhong Zhu. 2015. Task matching and scheduling for multiple workers in spatial crowdsourcing. In *GIS*. 21.
[6] Kay Fitzpatrick, Marcus A. Brewer, and Shawn Turner. 2006. Another Look at Pedestrian Walking Speed. *TRR* 1982, 1 (2006), 21–29.
[7] Srinivasa Raghavendra Bhuvan Gummidi, Torben Bach Pedersen, Xike Xie, and Esteban Zimányi. 2019. Push-Based Spatial Crowdsourcing for Enriching Semantic Tags in OpenStreetMap. In *GIS*. 532–535.
[8] Srinivasa Raghavendra Bhuvan Gummidi, Xike Xie, and Torben Bach Pedersen. 2019. A Survey of Spatial Crowdsourcing. *TODS* 44, 2 (2019), 8.
[9] Leyla Kazemi and Cyrus Shahabi. 2012. Geocrowd: enabling query answering with spatial crowdsourcing. In *GIS*. 189–198.
[10] Leyla Kazemi, Cyrus Shahabi, and Lei Chen. 2013. Geotrucrowd: trustworthy query answering with spatial crowdsourcing. In *GIS*. 314–323.
[11] Kurt Mehlhorn and Stefan Näher. 1999. *LEDA: A Platform for Combinatorial and Geometric Computing*. Cambridge University Press. http://www.mpi-sb.mpg.de/%7Emehlhorn/LEDAbook.html
[12] Dimitrios Michail, Joris Kinable, Barak Naveh, and John V Sichi. 2019. JGraphT–A Java library for graph data structures and algorithms. *arXiv preprint arXiv:1904.08355* (2019).
[13] Dezhi Sun, Ke Xu, Hao Cheng, Yuanyuan Zhang, Tianshu Song, Rui Liu, and Yi Xu. 2018. Online delivery route recommendation in spatial crowdsourcing. *WWW* (2018), 1–22.
[14] Hien To, Cyrus Shahabi, and Leyla Kazemi. 2015. A server-assigned spatial crowdsourcing framework. *ACM TSAS* 1, 1 (2015), 2.
[15] Yongxin Tong, Libin Wang, Zimu Zhou, Bolin Ding, Lei Chen, Jieping Ye, and Ke Xu. 2017. Flexible online task assignment in real-time spatial data. *PVLDB* 10, 11 (2017), 1334–1345.
[16] Luan Tran, Hien To, Liyue Fan, and Cyrus Shahabi. 2017. A Real-Time Framework for Task Assignment in Hyperlocal Spatial Crowdsourcing. *TOIST* (2017), 37.