Computers, Materials & Continua
DOI:10.32604/cmc.2022.023716
Article

Tech Science Press

Enhancing Task Assignment in Crowdsensing Systems Based on Sensing Intervals and Location

Rasha Sleem[1], Nagham Mekky[1], Shaker El-Sappagh[2,3], Louai Alarabi[4,*], Noha A. Hikal[1] and Mohammed Elmogy[1]

[1]Faculty of Computers and Information, Mansoura University, Mansoura, 35516, Egypt
[2]Faculty of Computers and Artificial Intelligence, Benha University, Banha, 13518, Egypt
[3]Faculty of Computer Science and Engineering, Galala University, Suez, 435611, Egypt
[4]College of Computer and Information Systems, Umm Al-Qura University, Mecca, 715, Saudi Arabia
*Corresponding Author: Louai Alarabi. Email: lmarabi@uqu.edu.sa
Received: 18 September 2021; Accepted: 23 November 2021

Abstract: The popularity of mobile devices with sensors is captivating the attention of researchers to modern techniques, such as the internet of things (IoT) and mobile crowdsensing (MCS). The core concept behind MCS is to use the power of mobile sensors to accomplish a difficult task collaboratively, with each mobile user completing much simpler micro-tasks. This paper discusses the task assignment problem in mobile crowdsensing, which is dependent on sensing time and path planning with the constraints of participant travel distance budgets and sensing time intervals. The goal is to minimize aggregate sensing time for mobile users, which reduces energy consumption to encourage more participants to engage in sensing activities and maximize total task quality. This paper introduces a two-phase task assignment framework called location time-based algorithm (LTBA). LTBA is a framework that enhances task assignment in MCS, whereas assigning tasks requires overlapping time intervals between tasks and mobile users' tasks and the location of tasks and mobile users' paths. The process of assigning the nearest task to the mobile user's current path depends on the ant colony optimization algorithm (ACO) and Euclidean distance. LTBA combines two algorithms: (1) greedy online allocation algorithm and (2) bio-inspired travel-distance-balance-based algorithm (B-DBA). The greedy algorithm was sensing time interval-based and worked on reducing the overall sensing time of the mobile user. B-DBA was location-based and worked on maximizing total task quality. The results demonstrate that the average task quality is 0.8158, 0.7093, and 0.7733 for LTBA, B-DBA, and greedy, respectively. The sensing time was reduced to 644, 1782, and 685 time units for LTBA, B-DBA, and greedy, respectively. Combining the algorithms improves task assignment in MCS for both total task quality and sensing time. The results demonstrate that combining the two algorithms in LTBA is the best performance for total task quality and total sensing time, and the greedy algorithm follows it then B-DBA.

Keywords: Mobile crowdsensing; online task assignment; participatory sensing; path planning; sensing time intervals; ant colony optimization

1 Introduction

The popularity of mobile devices that consists of internal sensors (camera, temperature, microphone, GPS, Wi-Fi/3G/4G interfaces, accelerometer, etc.) and external sensors (Google glass, healthcare sensors, wearable sensors, etc.) is captivating the attention of researchers for the internet of things (IoT) [1,2] and mobile crowdsensing (MCS) [3,4] techniques. MCS is a novel paradigm of crowdsourcing that is a methodology in which many people use their mobile sensors for sensing, collecting, sharing data, and extracting information to analyze [5].

Comparing MCS with traditional sensor networks, one of the significant benefits is the active participation of workers in the collection and sharing of sensing data. This is attributed to that MCS is a powerful paradigm because of has extraordinary sensing requirements [5–8]. The sensing requirements, communication capabilities, and computation allow smartphones to execute more complicated tasks. There are many applications examples of MCS include intelligent transportation [9], air quality monitoring [10], road traffic monitoring [11], surface monitoring of road pavements [12], noise level sensing, location-based/mobile/spatial crowdsourcing, smart cities, etc.

Participatory sensing and opportunistic sensing are the two paradigms of MCS. Active engagement of mobile users is not required in opportunistic sensing [13], and mobile users typically walk along predetermined pathways or follow movement patterns driven by their everyday schedules, tastes, or desires. Mobile devices may be designed to capture sensing data when their owners travel automatically. Mobile users are expected to travel to certain task sites to complete those activities in participatory sensing [14]. The motions of such users are planned around delegated sensing tasks, and their participation is voluntary and regulated.

Mobile crowdsensing had many challenges for research, such as maximizing data and tasks quality, minimizing sensing cost, privacy guarantee, the storage and energy limitation of mobile node resources, and task assignment. To overcome these challenges, this work focuses on the tasks assignment considering various constraints, such as spatial coverage travel budget, energy consumption, travel cost, sensing time intervals, and sensing task quality [15]. To maximize the sensed data quality, the scenario allocated sensing tasks to a suitable number of participants in a certain area with minimum sensing time.

MCS process consists of: (1) requesters, who send tasks, (2) workers, who have sensor-rich devices, and (3) platform, which receives sensing tasks from requesters and allocates them to workers then sends data back to requesters. The relationship between workers and tasks is critical to the success of MCS applications. The sensing tasks are an essential part of the sensing process.

Selecting tasks for a mobile user might be difficult because different task assignment schemes have varying assignment criteria. Therefore the online task assignment problem for MCS is described in this work. There are many challenges for allocating tasks for a mobile user:

■ Challenge 1: There is a large number of sensing tasks for a large number of participants, and a single mobile user may participate in multiple sensing tasks. The sensing tasks need data from various sensors at various times.

■ Challenge 2: Encourage more people to engage in sensing activities.
■ Challenge 3: Determine whether the task has the smallest increasing aggregate sensing time assigned to a mobile user.
■ Challenge 4: Locating the nearest task for the mobile user whose total travel budget is less than the worker's total travel distance.

To address these challenges, there are two mechanisms. First, reducing the total sensing time of mobile users reduces energy consumption to encourage more participants to engage in sensing activities. Second, determining the location of both mobile users and tasks then choosing the nearest task to the mobile user's path. This work presents a framework location time-based algorithm (LTBA). The framework works on reducing total sensing time based on overlapping in sensing time intervals and assigning the nearest task to the path of the mobile user. Reducing total sensing time and increasing task quality incentivizes mobile users to participate in mobile crowdsensing applications.

LTBA enhances performance metrics for total tasks quality and aggregate sensing time. LTBA combines two algorithms: (1) greedy online allocation algorithm [16,17] and (2) bio-inspired travel-distance-balance-based algorithm (B-DBA) [18]. B-DBA was location-based and worked on maximizing total task quality. The greedy algorithm was sensing time interval-based and worked on reducing the overall sensing time of the mobile user. Combining the algorithms improves task assignment in MCS for both total task quality and sensing time of both mobile users and tasks.

The scenario has a sensing task set T and a mobile worker set M that arrive in sequence at the service platform. Their arrival obeys a Poisson distribution without knowing what will happen next. The service platform must reply to each incoming request, and delegate tasks in the tasks set T to users in the users set M. The smartphone's sensing capabilities can be fully utilized. Because of location privacy for mobile users in mobile crowdsensing, the cases focused on where they are anonymous, and no single mobile user's contribution history is tracked.

The main contributions of this paper are summarized as follows:

■ This work formulates and enhances the problem of sensing tasks assignment in MCS by a tradeoff between maximizing total tasks quality and minimizing aggregate sensing time.
■ Considering two critical attributes: sensing time of mobile user and location of both tasks and mobile user.
■ For improving tasks assignment, this work proposes a combination of two algorithms: (1) greedy online allocation algorithm and (2) B-DBA.
■ Overlapping or covering in the sensing time interval, in this scenario, the service platform divides the available tasks into two task groups based on the complete or partial intersection in time intervals of tasks and mobile user's task pool.
■ Determining the nearest task. The service platform chooses the nearest task in the available task set to the mobile user's path without exceeding budget constraints for increasing task quality. Selecting tasks depends on the ant colony optimization algorithm (ACO) [19] and Euclidean distance.
■ Both synthetic and real-world data are used. The synthesis data set is used to generate information on tasks and mobile users. The real data set is for the location of both tasks and mobile users.

The rest of the paper is structured as follows. Section 2 introduces previous work. Section 3 formulates the task assignment problem and designs the system models. Section 4 discusses in detail the combination of two task assignment algorithms. Section 5 discusses the results' simulation for success assessment. Finally, Section 6 concludes the work and represents the future direction.

2 Related Work

In this section, some previous work that studied task assignment problems based on two attributes: sensing time and location of both sensing tasks and mobile users are addressed. MCS employs smartphones to collect data on a much bigger scale than is possible with traditional methods. Selecting tasks for a mobile user might be difficult because different task assignment schemes have varying assignment criteria and objectives. For example, Zhao et al. [16,17] introduced two allocation algorithms, approximation algorithm for offline model and greedy algorithm for the online model. Allocating tasks with mobile users was depended on time overlapping between tasks' sensing intervals that needed the same sensing services from mobile users. They focused on the energy consumption of smartphones, fairness in assigning tasks, and task arrivals that were dynamic and unexpected. They did not work on heterogeneous sensing tasks or with flexible times' starting and ending and concentrated on reducing smartphone sensing time, but no energy savings have been reported.

Gong et al. [18,20] designed four algorithms to assign workers based on the location of tasks and the current location of workers to increase task quality. Whereas quality/progress-based algorithm (QPA) chose the work with the highest task quality increment to trip cost ratio, task density-based (TDA) tended to direct users to high task-density regions. At the same time, travel-distance-balance-based (DBA) considered journey distance balance information. Bio-inspired travel-distance-balance-based (B-DBA), the final algorithm, combines the travel-distance-balance-aware measure in DBA with a bio-inspired search to further boost job assignment performance. One type of task was assigned to workers.

Wang et al. [21] used spatiotemporal correlations to address heterogeneous crowdsensing tasks. They involved all sharing the same resources but had different spatiotemporal granularities. They used assignment stages such as intra- and inter-task. To enhance the time efficiency, they developed a decomposition-and-combination framework. They aimed to improve data quality while reducing the total incentive budget.

Xia et al. [22] employed geoindistinguishability to make workers' location privacy more secure and minimize workers' travel distance. So, they designed two probability-based distance comparison mechanisms, the worker-based distance comparison mechanism (WDCM) and the task-based distance comparison mechanism (TDCM). Yin et al. [23] introduced task assignment in crowdsourcing platforms by allocating one task to multiple workers using a many-to-one matching approach. They focused on budgetary limits, quality requirements of tasks, and heterogeneity in sensing tasks and workers. Tao et al. [24] worked on the task assignment that depended on the location of both workers and tasks to maximize data quality and increase worker profit. They provided a genetic algorithm (GA) and a detective algorithm (DA).

Miao et al. [25,26] assigned crowdsourcing task allocation dependent on participant position. The aim is to use a quality-aware online task assignment (QAOTA) algorithm to improve overall efficiency for location-based tasks. Gad-ElRab et al. [27] classified tasks based on sensors that required providing interval tree structure in interval tree-based task scheduling method (ITBTS) and determined the overlapping between sensing time instances to minimize the energy consumption and time for the sensing process. Lai et al. [28] focused on the sensitive duration of each task and the capabilities of participants. The objective was to increase the number of tasks that were completed. For allocating tasks to participants, that article used a greedy heuristic algorithm. Xiao et al. [29] studied mobile social networks (MSNs) using MCS to solve the makespan-sensitive task assignment. The goal for different types of sensing tasks was minimizing the average or the largest makespan. Average makespan sensitive online task assignment (AOTA) algorithm and the largest makespan sensitive online task assignment (LOTA) algorithm.

Wu et al. [30] evaluated the extrinsic and intrinsic ability of mobile users for allocating tasks. A modified Thompson sampling worker selection (MTS-WS) algorithm was context-aware that was designed to evaluate a worker's service quality. Huang et al. [31] outlined task assignments based on the performing task time. They illustrated a task assigning technique called the optimized allocation scheme of time-dependent tasks (OPAT) to increase each mobile user's sensing capacity.

The previous studies focused on the sensitive duration of tasks, sending tasks, receiving results, context or location. Tab. 1 summarizes previous literature on task assignment problems in MCS. This paper addresses the task assignment, total sensing time, and path planning problem. The requests of mobile crowdsensing tasks determine the time interval for sensing the required data in specific regions. For the mobile user's tour, he moves from his starting point to his destination. In contrast, this work differs from them in terms of problem definition. It redefines the task assignment problem by working with task sensing time interval and location for both tasks and mobile users' constraints. Combining the two metrics improves the task assignment process in MCS, increasing the total task quality and reducing the overall aggregate sensing time.

Table 1: A summary of some of the current related work

Study	Objective	Used methods	Parameters
Zhao et al. [16] and [17]	Tradeoff between energy consumption of smartphones and fairness in assigning tasks.	Approximation (offline) and greedy (online).	Sensing time interval.
Gong et al. [18] and [20]	Increasing task quality.	QPA, TDA, DBA and B-DBA	Location of both worker and task.
Wang et al. [21]	Improving data quality and reducing the total incentive budget.	Region oriented search (rosear), worker oriented search (wosear) and worker-region oriented search (wrposear).	Real-time and location.
Xia et al. [22]	Location privacy and minimizing workers travel distance.	WDCM and TDCM.	Location of workers
Yin et al. [23]	Maximizing the total quality of platform.	Greedy.	Number of tasks and workers.
Tao et al. [24]	Maximizing data quality and increasing worker profit.	Genetic algorithm (GA) and detective algorithm (DA).	Location of both workers and tasks.

(Continued)

Table 1: Continued

Study	Objective	Used methods	Parameters
Miao et al. [25] and [26]	Quality maximization.	Polynomial-time (online) and approximation algorithm (offline).	Location.
Gad-elrab and Alsharkawy [27]	Minimizing the energy consumption and time for sensing process.	ITBTS.	Sensing time instance.
Lai et al. [28]	Increasing the number of tasks that were completed.	Greedy heuristic algorithm.	Sensitive duration of tasks.
Xiao et al. [29]	Minimizing the average makespan or minimizing the largest makespan.	AOTA and LOTA.	Time of sending tasks and receiving results.
Wu et al. [30]	Selecting mobile users with high-quality.	MTS-WS.	Context.
Huang et al. [31]	Maximizing the capacity of each mobile user's sensing.	OPAT.	The performing task time.

3 The Proposed Method

This section discusses the system model and constructs the problem of online task assignment based on two constraints: (1) task sensing time interval and (2) location for both task and mobile user (the starting location and predetermined destination). The next subsections illustrate the system model, task model, mobile user model, and algorithm design. Fig. 1 illustrates the proposed framework. The proposed framework is explained in detail in Subsection 3.4. The proposed framework consists of three phases. First, the preprocessing phase sets available tasks into task set T and mobile users into worker set M. Second, overlapping or covering in time interval phase divides the available tasks into two task groups based on a complete or partial intersection in time intervals of tasks and worker's task pool. Third, determining the nearest task phase chooses the nearest task in the available task set T to mobile user m_i path.

3.1 System Model

In the MCS sensing process, a mobile user mi has a tour from his starting point to his destination. The requests of mobile crowdsensing tasks determine the time interval for the sensing required data in specific regions. A mobile user mi is allocated with sensing tasks in mi path without exceeding mi travel budget. In a real-time system, requesters' mobile users and tasks arrive at the service platform online sequentially. The service platform must reply to each incoming request without knowing what will happen next. Because of location privacy for mobile users in MCS, this work focuses on cases where users are anonymous, and no single mobile user's contribution history is tracked. The assignment is dependent on the tasks that are still available on the service platform that have arrived and have not yet expired. The algorithm regards tasks that demand the same sensing service from mobile users to be homogeneous. Tab. 2 shows a list of notations that are used in this work.

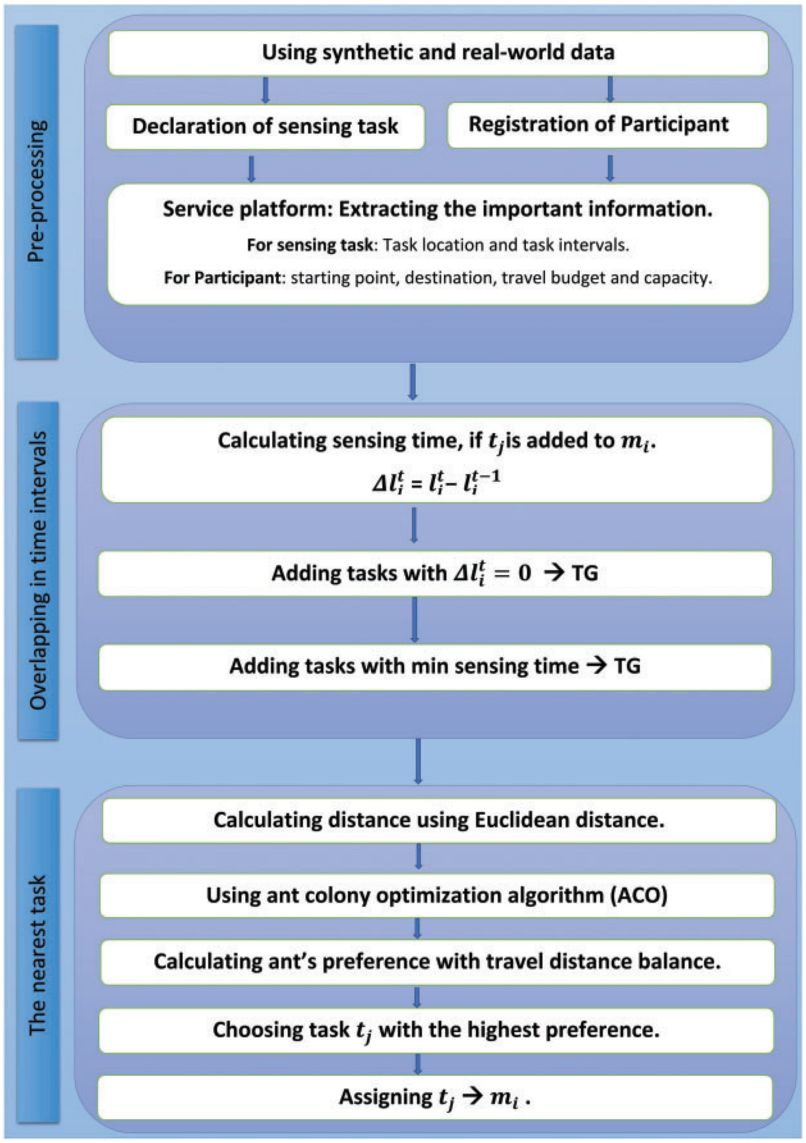

Pre-processing
Using synthetic and real-world data
Declaration of sensing task
Registration of Participant
Service platform: Extracting the important information.
For sensing task: Task location and task intervals.
For Participant: starting point, destination, travel budget and capacity.
Overlapping in time intervals
Calculating sensing time, if t_j is added to m_i.
$\Delta l_i^t = l_i^t - l_i^{t-1}$
Adding tasks with $\Delta l_i^t = 0$ → TG
Adding tasks with min sensing time → TG
The nearest task
Calculating distance using Euclidean distance.
Using ant colony optimization algorithm (ACO)
Calculating ant's preference with travel distance balance.
Choosing task t_j with the highest preference.
Assigning $t_j \rightarrow m_i$.

Figure 1: The framework of location time-based algorithm (LTBA)

Table 2: The basic notations in this work

Notation	Description	Notation	Description
m_i, M	Mobile worker i and worker set.	$\mid P_{m_i} \mid$	The number of tasks in mobile user's task pool.
t_j , T	Task j and tasks' requests set.	dist(loc_{ik} , loc_j)	The distance between current mobile user location loc_{ik} and task loc_j.

(Continued)

Table 2: Continued

Notation	Description	Notation	Description
T_{ID}	The composition of task id and task number.	$\mathrm{T(m_{ikj})}$	Mobile user travel cost budget constraint.
T_{Loc}	It represents task t_j location latitude and longitude.	$\mathrm{P_{mi}}$	worker's task pool.
s_j	The start of task t_j sensing time interval.	*length* ($\mathrm{P_{mi}}$)	Path length for user m_i.
e_j	The end of task t_j sensing time interval.	Δl_i	It represents the increasing in sensing time after allocating task t_j to m_i.
p_j	It denotes the probability, that a randomly selected mobile user would provide correct data for task t_j.	$\tau_{\mathrm{tail,j}}$	The local pheromone on the ant path from the current path tail and task t_j.
l_i	The aggregate sensing time of mobile user m_i.	τ_0	Global pheromone.
$\mathrm{A}p_j(k)$	The function used to calculate task's t_j quality.	λ	The evaporation decreasing rate on the path between task t_j and the current path tail.
K	The number of mobile users that executes task t_j.	$\mathrm{ed_{ij}}$	The distance between t_j and m_i tail.
$\mathrm{F}p_j(k)$	The auxiliary function that smoothes quality function.	(lat,lon)	Latitude and longitude of location.
$\mathrm{U_{ID}}$	It represents both mobile user number and id.	M_i	The capacity for user m_i of sensing tasks that he can execute.
$\mathrm{ini_{Loc}}$	It is the current location of user m_i.	$\mathrm{dest_i}$	It is the predetermined destination of user m_i.
B_i	The maximum travelling distance budget for user m_i.	θ	The travel budget ratio of mobile user m_i

3.2 Task Model

Requesters send sensing tasks to the service platform that are published dynamically upon their arrival. The sensing tasks set are denoted $T = \{t_1, t_2, \ldots.., t_n\}$ every task has attributes such as $t_j = \{T_{\mathrm{ID}}, T_{\mathrm{Loc}}, s_j, e_j, p_j\}$, $\forall\ t_i \in T$, where T_{ID} is the composition of task id and the task number, T_{Loc} represents longitude and latitude of task location, s_j is the start of sensing time interval for t_j, e_j is the end of sensing time interval and p_j denotes the probability, that a randomly selected mobile user would provide correct data for task t_j. For each task t_j, the increment of aggregate sensing time of mobile user m_i is checked if task t_j is allocated to it as [16]:

$$\min\left(\sum_{i=0}^{n} l_i\right) \tag{1}$$

where l_i is the aggregate sensing time of mobile user m_i. The number of mobile users who perform a task denoted by k affects the task quality function $Ap_j\ (k)$ in [18]. Because the reliability of each individual mobile user is unknown, the outcome is assumed using the majority voting rule:

■ for odd values of k:

$$Ap_j(k) \ = \sum_{r=\frac{k+1}{2}}^{k} \binom{k}{r} p_j^r (1-p_j)^{k-r} \tag{2}$$

■ else,

$$Ap_j(k) = \sum_{r=\frac{k}{2}+1}^{k} \binom{k}{r} p_j^r (1-p_j)^{k-r} \ + \frac{1}{2} \left(\begin{array}{c} k \\ k \\ 2 \end{array} \right) p_j^{\frac{k}{2}} (1-p_j)^{\frac{k}{2}} \tag{3}$$

where k is the number of mobile users that perform task t_j and a non-negative integer. p_j is more than 0.5. This is due to the fact that for tasks with p_j less than 0.5, the mobile users' results are reversed, then correct answers are obtained with a probability greater than 0.5. The task quality increases with odd k and stays the same otherwise. In [18], an auxiliary function Fpj(k) is designed to smooth quality function.

■ for zero and odd values of k:

$$Fp_j(k) = A_{p_j}(k) \tag{4}$$

■ else

$$Fp_j(k) = \frac{Ap_j(k-1) + Ap_j(k+1)}{2} \tag{5}$$

3.3 Mobile User Model

The mobile user intended to travel from his or her current location to a destination. The service platform assigns tasks to mobile users' tours to perform without exceeding their travel budget. Every mobile user sends his information to the service platform. Mobile users set is denoted as $M = \{m_1, m_2, \ldots., m_m\}$. Every user has attributes such as $m_i = \{U_{ID}, ini_{Loc}, dest_i, B_i, M_i\}$, where U_{ID} represents both mobile user number and id, ini_{Loc} is the current location of user m_i, $dest_i$ is the predetermined destination of user m_i, B_i represents traveling distance budget. M_i is the maximum number of sensing tasks that he can execute. The mobile user is now at ini_{Loc} and plans to go to $dest_i$. However, completing tasks can require him to take a detour, resulting in extra travel costs. The overall travel cost does not surpass B_i. Any function that measures the cost of traveling from one place to another can be used to calculate the travel cost. In this work, Euclidean distance is used.

■ Mobile user sensing time constraint:

$$\Delta l_{ij} = 0 \text{ or } \min(\Delta l_{ij}) \tag{6}$$

■ Mobile user capacity constraint:

$$|\ P_{m_i}\ | \le M_i \tag{7}$$

■ Mobile user travel cost budget constraint:

$$T(m_{ikj}) = \sum_{r=1}^{|\mathrm{Pm_i}|} dist(loc_{ik}, loc_j) \leq B_i \tag{8}$$

■

$$B_i, = \mathrm{dist}(\mathrm{ini_{Loc}}, \mathrm{dest_i}) * \theta \tag{9}$$

where $\Delta l_{\mathrm{ij}} = l_{\mathrm{ij}} - l_{\mathrm{i}}$. l_{i} is the sensing time of mobile user $\mathrm{m_i}$. l_{ij} is the sensing time of $\mathrm{m_i}$ after assigning task $\mathrm{t_j}$. $\Delta l_{\mathrm{ij}} = 0$ means that the sensing interval of task $\mathrm{t_j}$ is within a task interval in P_{mi}. P_{mi} is a task pool that contains mobile user tasks. $|P_{mi}|$ is the number of tasks in a mobile user's task pool. $dist(loc_{ik}, loc_j)$ denotes the distance between current mobile user location loc_{ik} and task location loc_j. $\mathrm{dist}(\mathrm{ini_{Loc}}, \mathrm{dest_i})$ is the travel distance between the starting location to the destination of mobile user $\mathrm{m_i}$. θ denotes travel budget ratio.

3.4 Algorithm Design

The sensing tasks and mobile users arrive at the service platform upon arrival rate. At each time instance, the service platform schedules available tasks in task set T and available mobile user in worker set M based on the sensing time interval of each task and location of both tasks and workers. LTBA algorithm consists of three phases:

1) **Preprocessing phase**: In this phase, the service platform sets available tasks into task set T and mobile users into worker set M. To explain this phase in detail, at each time instance of arrival rate, the service platform adds the arrival tasks to task set T and the arrival mobile user to worker set M, then extracts tasks' attributes (i.e., location $\mathrm{T_{Loc}}$, start sensing time $\mathrm{s_j}$, end sensing time $\mathrm{e_j}$ and task correctness p_j) and mobile user's attributes (i.e., current location $\mathrm{ini_{Loc}}$, destination $\mathrm{dest_i}$, path length, sensing time l_i, distance budget B_i and capacity M_i). The path length is the total travel distance of the mobile user from starting location to the current location, passing through all tasks in the worker's task pool $\mathrm{P_{m_i}}$. Sensing time for $\mathrm{m_i}$ is the sum of sensing time for tasks in the worker's task pool $\mathrm{P_{m_i}}$. Capacity M_i is the maximum number of tasks that user $\mathrm{m_i}$ can execute.
2) **Overlapping or covering in time interval phase**: In this phase, the service platform divides the available tasks into two task groups based on the complete or partial intersection in time intervals of tasks and worker's task pool. To explain this phase in detail, mi in worker set Δlij is determined for each mobile user, representing the increasing sensing time for $\mathrm{m_i}$ after allocating task $\mathrm{t_j}$ to him, where $\Delta l_{\mathrm{ij}} = l_{\mathrm{ij}} - l_{\mathrm{i}}$.

For example, a mobile user $\mathrm{m_i}$ has task $\mathrm{t_j}$ [2,5] in task pool $\mathrm{P_{m_i}}$. Task set contains $\mathrm{t_x}$ [3,4], $\mathrm{t_y}$ [4,6], and $\mathrm{t_z}$ [6,8]. Each task has to sense interval $I = [\mathrm{s_j}, \mathrm{e_j}]$. The sensing time for $\mathrm{m_i}$, $l_{\mathrm{ij}} = l\{[2,5]\} = 3$. There are some cases to select the best task,

1. In the case of adding $\mathrm{t_x}$ to $\mathrm{P_{m_i}}$, $\mathrm{s_x} > \mathrm{s_j}$ and $\mathrm{e_x} < \mathrm{e_j}$, so $\mathrm{t_x}$ is covered by $\mathrm{t_j}$. $l_{\mathrm{ijx}} = l\{[2,5] \cup [3,4]\} = l\{[2,5]\} = 3$ and $\Delta l_{\mathrm{ijx}} = l_{\mathrm{ijx}} - l_{\mathrm{ij}}$, $\Delta l_{\mathrm{ijx}} = 0$.
2. In the case of adding $\mathrm{t_y}$ is $\mathrm{P_{m_i}}$, so, $l_{\mathrm{ijy}} = l\{[2,5] \cup [4,6]\} = l\{[2,6]\} = 4$ and $\Delta l_{\mathrm{ijy}} = 1$. There is overlapping between $\mathrm{t_j}$ and $\mathrm{t_y}$.
3. In the case of adding $\mathrm{t_z}$ to $\mathrm{P_{m_i}}$, so, $l_{\mathrm{ijz}} = l\{[2,5] \cup [6,8]\} = 3 + 2 = 5$ and $\Delta l_{\mathrm{ijz}} = 2$.

From the previous example, the delta sensing Δlij and the sensing time lij are smaller in cases 1 and 2. In case1, tx is covered by tj. In case 2, there is an overlapping between tj and ty. So, the chance of choosing these tasks for the next phase is greater. Then dividing available task set into two groups, one has tasks with smaller sensing time and the other with greater sensing time. In this phase, the algorithm uses two rules [16]. "Rule 1: assigning a task to a smartphone if it can be fulfilled by other tasks already assigned to that smartphone. Rule 2: assign a task to the smartphone with the shortest aggregate sensing time if the task were to be assigned.

3) **Determining the nearest task phase**: In this phase, the service platform chooses the nearest task in the available task set T to mobile user m_i path. To explain this phase in detail, use the task group TGi with smaller sensing time from the previous phase to find the nearest mobile user m_i that exceeds the mobile user's budget. The algorithm uses B-DBA [18]. Selecting the nearest tasks process depends on the ACO algorithm and Euclidean distance [32].

For ACO, intelligent ants begin their moves to search for food randomly. When an ant finds food, it returns to the colony and leaves a trail of pheromones. Other ants use that route in search of food if they detect the pheromone. The pheromone fades with time, preventing the algorithm from reaching a local optimum solution. The basic idea of B-DBA was dependent on the bio-inspired search and travel distance. B-DBA works as follows: it consists of rounds where ($round > 1$). Each round has a group of ants that move from the user's current position and all available tasks to the destination. In the first round, the initial global pheromone τ_0 is the same for each task location. Then measure preferences of the available tasks using Eq. (10) for each ant to select the next task's location in the user's tour.

$$pre(t_j) = \tau^{\varepsilon}_{tail,t_j}.\frac{\Delta Fp_j \cdot (B_i - length(p_i) - dist(p_{tail}, t_j) - ed_{ij})^{\alpha}}{dist(p_{tail}, t_j)^{\gamma}} \tag{10}$$

Euclidean distance between two locations:

$$dist(tail, t) = \sqrt{(lat_{tail} - lat_t)^2 + (lon_{tail} - lon_t)^2} \tag{11}$$

where $\tau_{\text{tail,tj}}$ denotes local pheromone between the current user's location and task tj. ε, γ and α are weighting parameters. $length(p_i)$ is path length of user m_i. $dist(p_{tail}, t_j)$ denotes distance between current user's location and task t_j. ed_{ij} denotes distance between user's destination and task t_j. lat_{tail} *and* lon_{tail} are latitude and longitude of current user's location. lat_t *and* lon_t are latitude and longitude of task's location.

The chance of selecting task tj as the next task depends on its probability and performance value. The highest preference value of task tj gives a high probability to tj to be chosen as the next task. The pheromone on the path between task tj and the current path tail is reduced by evaporation rate λ, when selecting tj as the next task. For updating local pheromone:

$$\tau_{\text{tail,tj}} = (1 - \lambda).\tau_{\text{tail,j}} + \lambda.\tau_0. \tag{12}$$

After determining all ants' pathways for one round, select the two highest increments of task quality of two ants. The best two ant's global pheromones of task locations are increased by τ_0 for the best ant and $\tau_0/2$ for the second-best ant, on the path that they follow, then a new round starts after all local pheromones have been modified to match their global pheromones.

Algorithm 1 illustrates the steps of the LTBA scenario. New tasks from available tasks are assigned to mobile users at every arrival without exceeding the budget and capacity constraints. Line 2 initializes, sensing time, delta, and creates task group TG_i set. Lines 3 and 4 calculate sensing time by CalcSensingTime() function at Algorithm 2 and delta sensing time for each task tj, if it is added to the worker's task pool. Lines from 5 to 9 check if task covered by any tasks in worker's pool, then add tj to TG_i else add sensing time and delta sensing to subsets Δ, TL. Line 12 checks for overlapping and append those tasks with minimum delta and sensing time to TG_i. Line 13 calls TG_i to TheClosestTask () function.

CalcSensingTime () at Algorithm 2 checks the length of the worker's pool to determine sensing time then return it. TheClosestTask () function at Algorithm 3 is used to find the nearest task tj in the mobile user m_i path. Line 2 determines the global pheromone for the tasks in TG_i. Using iteration of rounds to select the nearest task. Line 4 uses a global pheromone as a local pheromone. Line 5 makes iterations for a group of ants to select the best next task. Lines from 6 to 11 calculate preference for each task, and the task with the maximum preference is selected, then update the local pheromone of the chosen task. At the end of ant iterations, choose the best two ants, update the global pheromone of tasks chosen, and start a new round iteration. Line 16 adds the nearest task tj to the worker's pool P_{m_i}.

Algorithm 1: Location time-based algorithm (LTBA)

```
Input: Mobile user $m_i$, Sensing time $l_i$ Budget $B_i$ and available Taskset $T = 1, 2, \ldots, N$.
Output: tasks $t_j$ in worker's task pool $P_{mi}$.
1: for $t = 1, 2, \ldots, N$ do
2:     $l_{ij} \leftarrow l_i$; $\Delta l_{ij} \leftarrow 0$; $TG_i \leftarrow \Phi$;
3:     $l_{ij} \leftarrow$ CalcSensingTime()
4:     $\Delta l_{ij} = l_{ij} - l_i$;
5:     if $\Delta l_{ij} = 0$ then
6:         $TG_i \leftarrow t_j$;
7:     else
8:         $\Delta \leftarrow \Delta l_{ij}$;
9:         $TL \leftarrow l_{ij}$;
10:    end if
11: end for
12: $TG_i \leftarrow$ min $\Delta$ and min$TL$; Eq. (1)
13: TheClosestTask($TG_i$);
```

Algorithm 2: Procedure for CalcSensingTime() function.

```
Input: Sensing time $l_i$, interval of $p_j$.
Output: $l_{ij}$.
1: function CalcSensingTime ()
       Check the length of mobile user tasks' pool Eq. (7).
2:     if length($P_{m_i}$) = 0 then
3:         $l_{ij} = e_j - s_j$;
4:     else
5:         determining the intersection between
6:         interval of $t_j$ and worker's pool.
7:         $l_{ij} = \max(e_j) - \min(s_j)$
8:     end if
9: return $l_{ij}$
```

Algorithm 3: Procedure for TheClosestTask () function.

```
Input: TG_i.
Output: choose the Closest t_j to the path of m_i.
1: function TheClosestTask (TG_i)
2:         calculate global pheromone $\tau_0$
3:         for $round = 1:30$ do
4:           $\tau_{\text{tail,j}} \leftarrow \tau_0$
5:           for $ant = 1:30$ do
6:               Calculate preference for ant
7:               to find next task using Eq. (10).
8:               $pre(t_j) = \tau^{\varepsilon}_{tail,t_j} \cdot \frac{\Delta Fp_j \cdot (B_i - length(t_i) - dist(t_{tail}, t_j) - e_{ij})^{\alpha}}{dist(t_{tail}, t_j)^{\gamma}}$
9:               choose next task with max preference
10:                then update local pheromone for the newly
11:                  chosen task Eq. (12)
12:                end for
13:              choose the best two ants then update
14:              global pheromone of tasks' chosen
15:          end for
16:        $P_{m_i} \leftarrow t_j$
17: end function
```

4 Experimental Results

In this section, the performance is measured using two metrics: total task quality and aggregate sensing time. This work aims to reduce total sensing time that reduces energy consumption, and increase the quality of tasks metrics to encourage more participants and tasks requesters to engage in sensing activities. The default parameter settings are discussed in the next subsection. The B-DBA and greedy online allocation algorithms are used as the performance comparison baseline with LTBA. The used dataset and simulation settings are illustrated in the next Subsections 4.1 and 4.2.

4.1 Dataset

This paper uses both synthetic and real-world datasets. The synthesis data set is used to generate information on tasks and mobile users. The real data set is for the location of both tasks and mobile users. Gowalla [33] is the real data set, a location-based social networking service where users check in to report their whereabouts. It contains user-id, check-in time, latitude, longitude, and location id. The location of tasks and mobile users is selected spatially uniformly distributed from 5000 data set records.

4.2 Evaluation Metrics

The algorithms were implemented using MATLAB R2018a and running on a laptop with 16 GB memory and Intel Core i7-8550U CPU. Both synthesis and real datasets are used. Calculating the average of total tasks quality performance by:

$$\sum_{j=1}^{n} A_{p_j}(k)/n \qquad (13)$$

where n is the number of tasks, and aggregate sensing time by:

$$\sum_{i=1}^{M}\sum_{j=1}^{|\mathrm{Pm_i}|} l_j \tag{14}$$

The default parameters for sensing tasks are as follows: the arrival rate, location, the end of sensing task intervals, and the probability of task correctness. Poisson distribution is used to determine the arrival rate for tasks that are spatially uniformly distributed. The task arrival rate is set to 2.0 by default. The task-lasting time intervals follow a normal distribution with default mean and standard deviation values of 100 and 30, respectively. The probability of a task correctness $\mathrm{p_j}$ follows a normal distribution with 0.8 and 0.15 as the default mean and standard deviation, respectively.

The default parameters for mobile users are as follows: the arrival rate, the current location, the travel budget ratio, and capacity. For the arrival rate, Poisson distribution is used and set to 1.0 by default. The starting location and predetermined destination are spatially uniformly distributed. The shortest trip distance between mobile users' starting location and destination is denoted as travel budget ratio ($\geq$1) that follows a normal distribution with default 1.5 and 0.6 values for mean and standard deviation, respectively. Capacity is set to 3.

The default setting for LTBA and DBA parameters, γ is 1, weighting parameters ε is 3, evaporation rate λ is 0.9. Furthermore, the *round* number and the *ant* population size are set to 30 and 30, respectively. Euclidean distance is used to measure the distance between tasks' location and the mobile user's current location. The default parameter settings are displayed in Tab. 3.

Table 3: The parameters settings

Parameters	Values
Task arrival rate	2.0
Mobile user arrival rate	1.0
Task lasting time intervals	N (100, 30)
Task correctness probability ($\mathrm{p_j}$)	N (0.8, 0.15)
θ	N (1.5, 0.6)
ε	3
γ	1
α	1
Round	30
Ant	30
M_i	3

4.3 Parameters Analysis

These parameters are determined by trial and error to achieve the best performance for the tradeoff between higher task quality and lower sensing time in LTBA than in B-DBA and greedy. The travel budget of mobile users is computed as $\mathrm{Bi} = \mathrm{dist}(\mathrm{ini_{Loc}}, \mathrm{dest_i})^* \,\theta$. θ follows $\mathrm{N}(\mu_\theta, \sigma_\theta)$ and the probability of a task correctness $\mathrm{p_j}$ follows $\mathrm{N}(\mu_{\mathrm{pj}}, \sigma_{\mathrm{pj}})$ as [18] and [25].

Effects of the *ant* population size on optimum solution, execution time, and pheromone buildup [34]. The performance of the algorithm is not improved by having a large *ant* population size. As a result, a limited number is recommended. In general, to restrict the worst-case complexity, the values of *round* and *ant* are small. Capacity M_i is set to 3, which largely restricts the worst-case complexity.

5 Simulation Results

There are several scenarios for parameters settings that influence the tasks' quality and sensing time. The simulation results for parameters of different algorithms are presented. Varying parameter settings, while the default values are used for other parameters.

5.1 The Impact of γ

The impact of γ on tasks' quality in Fig. 2a and sensing time Fig. 2c. γ is tested in the range [1,5], increasing by 1. The quality of The total task is higher for both LTBA and B-DBA at γ equal 2. However, sensing time is less at γ equal to 1. The tasks' quality and sensing time are constant in the greedy online allocation algorithm. The expected value of γ is 1 or 2. To determine γ, the difference in values is calculated. The difference in the quality value at γ of 1 and 2 is very small, but the difference in the sensing time at γ 1 and 2 is large. γ is equal to 1 for the optimal value for both task quality and sensing time.

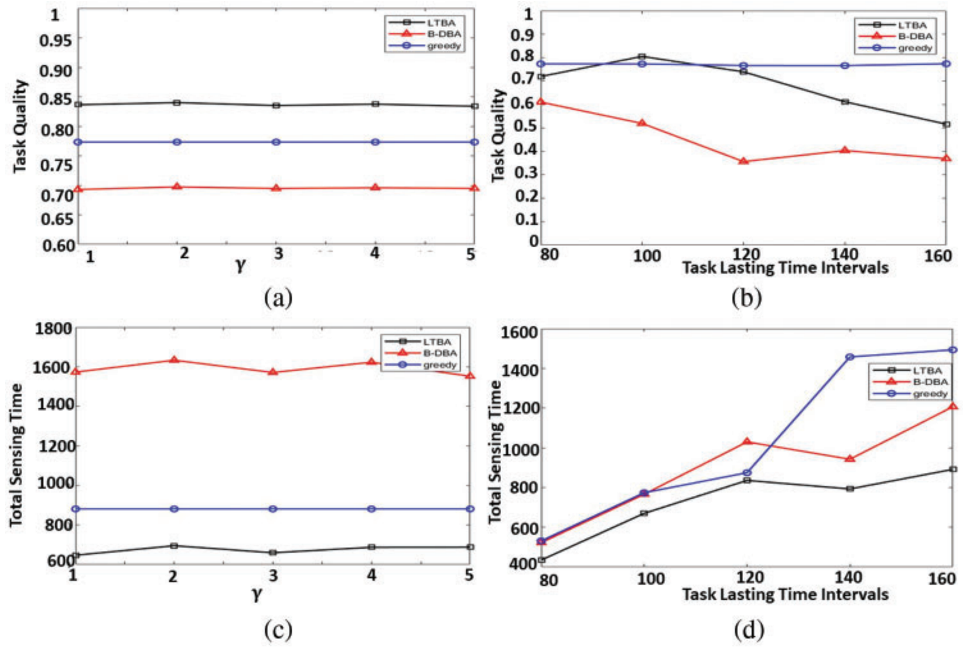


Figure 2: y and task lasting time intervals settings *vs.* total tasks quality and overall sensing time. (a) The impact of y on tasks' quality. (b) The impact of the task-lasting time interval on tasks' quality. (c) The impact of y on sensing time. (d) The impact of the task-lasting time interval on sensing time

5.2 The Impact of Sensing Time Intervals

In this test, the simulations are to see how the lasting task interval impacts different algorithms. The task-lasting time interval follows a normal distribution with a default mean equals to 30. Figs. 2b and 2d illustrate the average tasks quality and total sensing time *vs.* standard deviation value that is changed from 80 to 160. The task-lasting time interval is set to be 100 to achieve the best performance for both quality and sensing time for LTBA sensing time.

5.3 The Impact of Task Arrival Rate

In this test, we are varying the task arrival rate from 1.0 to 5.0. Figs. 3a and 3c present the performance of average task quality and overall sensing time under different settings *vs.* varying task arrival rates. In LTBA, the peak for the average task quality is at an arrival rate of 4.0, and the next is at an arrival rate of 2.0. However, the minimum value of the sensing time is at a task arrival rate of 3.0, and the next is at 2.0. In B-DBA and greedy, the peak is at 2.0. Therefore, the task arrival rate is set to be 2.0 to achieve the optimal solution for increasing the average task quality and decreasing the sensing time for LTBA.

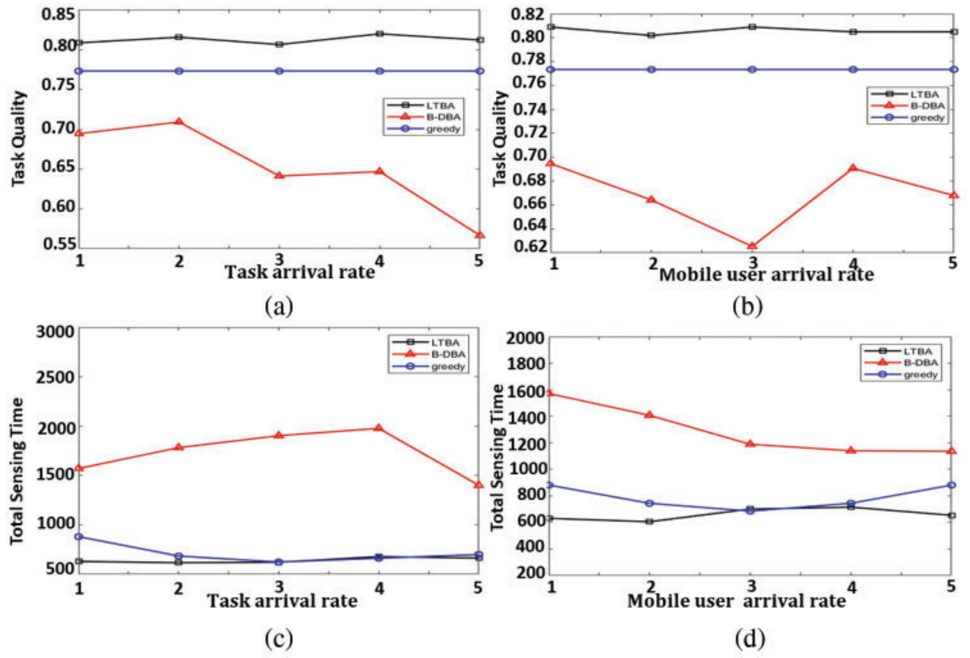


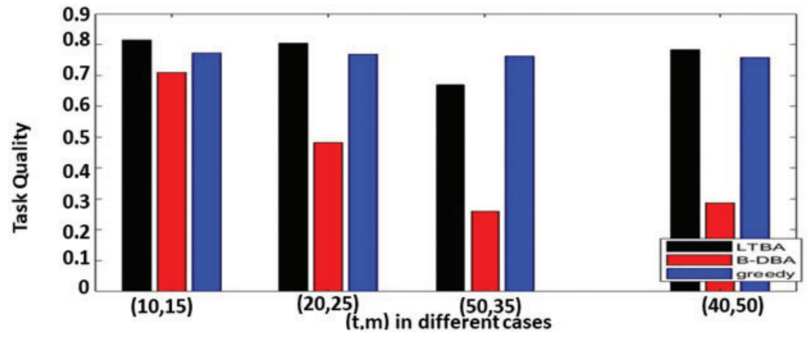
Figure 3: The arrival rate settings for both tasks and mobile users *vs.* total tasks quality and overall

5.4 The Impact of Mobile User Arrival Rate

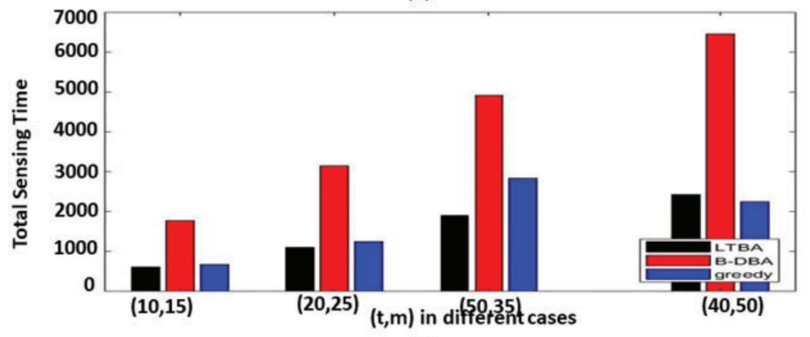
In this test, we are varying the mobile user arrival rate from 1.0 to 5.0. Figs. 3b and 3d illustrate the performance of average task quality and overall sensing time under different settings *vs.* varying mobile user arrival rates. The peak of the average task quality is at the mobile user arrival rate of 1.0 for LTBA, B-DBA, and greedy. The minimum value of the sensing time is at a mobile user arrival rate of 1.0 for LTBA and greedy but for B-DBA is at 5.0. Therefore, the mobile user arrival rate is set to be 1.0 to achieve the optimal solution for increasing the average task quality and decreasing the sensing time for LTBA.

5.5 *Different Cases of Tasks and Mobile Users*

In Figs. 4a and 3b, different cases of tasks and mobile users are tested. In the case of increasing tasks and mobile users, the total task quality decreases, and the overall sensing time increases. LTBA is the best total task quality and the least sensing time, and the greedy algorithm follows it. The B-DBA algorithm performs the worst if the number of tasks is greater than the number of mobile users. LTBA is better than sensing time but smaller than the greedy algorithm in total task quality.



(a)



(b)

Figure 4: The different cases of tasks and mobile users *vs.* (a) total tasks quality and (b) overall sensing time

5.6 *Discussion*

The previous observation motivates us to investigate the problem of assignment sensing tasks for minimizing sensing time while maximizing total task quality based on sensing time and path planning attributes. However, there are still a number of major challenges to be resolved. Striking a good balance between maximizing overall job quality and minimizing sensing time is extremely important.

The requesters determine the time interval for each sensing task to collect the required sensing data and the area of interest. In the proposed method, the covered phase is used because the optimal solution remains optimal after assigning tasks covered by the optimal solution tasks. The optimal aggregate sensing time remains constant.

This work uses the D-BDA, which consists of a bio-inspired search algorithm and a travel-distance-balance-based Algorithm (BDA). ACO is easy to integrate with other methods, and it excels at solving challenging optimization problems. The idea of BDA is based on whether there are two possible tasks to consider: 1 and 2. The work quality increment is the same for both tasks, as is the extra trip distance. The distance between task 1 and the user destination, on the other hand, is substantially greater than the distance between task 2 and the user destination. As a result, if task 1 is chosen, a far larger travel distance budget must be set aside for the user to arrive at his or her destination finally. If task 1 is chosen, the trip distance budget for visiting other task sites will be considerably reduced. We should act based on this observation.

Using the default parameter settings in Tab. 3 to calculate both total task quality and sensing time for LTBA, B-DBA, and greedy. Tab. 4 shows the results. The results demonstrate that combining the two algorithms in LTBA is the best performance for total task quality and total sensing time, and the greedy algorithm follows it then B-DBA. Combining the algorithms improves task assignment in MCS for both total task quality and sensing time.

Table 4: The results of the compared algorithms

Algorithm	LTBA	B-DBA	Greedy	
Average task quality	0.8158	0.7093	0.7733	
Sensing time	644	1782	685	Time units

The computational complexity of function that calculates sensing time for all available sensing tasks is O(n). The function that determines the closest task to the mobile user path has O($r\rho$ log n) where r is the *round* number and ρ is size of *ant* population. The computational complexity of the available mobile user set for LTBA is O(n log n).

The performance metric for task assignment is based on task quality. D-BDA was compared with quality-aware online task assignment QAOTA [25], which improved the overall task quality of location-based. QAOTA has the lowest performance and the highest complexity. So, LTBA is the best performance for total task quality and total sensing time, and the greedy algorithm follows it, then B-DBA then QAOTA.

6 Conclusion

This work presents LTBA that enhances the task assignment in MCS, whereas assigning tasks is restricted by time intervals and location of both tasks and mobile users. LTBA aims to achieve the best performance for the tradeoff between higher task quality and lower sensing time. LTBA combines two algorithms: (1) B-DBA, which was location-based and focused on increasing total task quality, and (2) the greedy online allocation algorithm, which focused on minimizing overall sensing time for mobile users. Minimizing the aggregate sensing time of the mobile user is based on the overlapping between sensing time intervals. The process of assigning the nearest task to the mobile user's current path depends on the ACO algorithm and Euclidean distance. Using a combination of the two algorithms to improve task assignment in MCS enhances task quality and reduces overall sensing time for mobile users. Under various settings, LTBA outperforms the compared algorithms (B-DBA and greedy). The future study will investigate ways to increase the performance of the proposed algorithm as well as investigate their performance in other scenarios, such as discussing the privacy of mobile workers' locations. The heterogeneous crowdsensing tasks will be the scope of the next work to maximize the quality of the collected data.

Funding Statement: The authors received no specific funding for this study.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] R. J. Hassan, S. R. M. Zeebaree, S. Y. Ameen, S. F. Kak, M. A. M. Sadeeq *et al.*, "State of art survey for IoT effects on smart city technology: Challenges, opportunities, and solutions," *Asian Journal of Research in Computer Science*, vol. 8, no. 3, pp. 32–48, 2021.

[2] A. Zanella, N. Bui, A. Castellani, L. Vangelista and M. Zorzi, "Internet of things for smart cities," *IEEE Internet of Things Journal*, vol. 1, no. 1, pp. 22–32, 2014.

[3] X. Zhang, Z. Yang, W. Sun, Y. Liu, K. Xing *et al.*, "Incentives for mobile crowdsensing: A survey," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 1, pp. 54–67, 2015.

[4] A. Capponi, C. Fiandrino, B. Kantarci, L. Foschini, D. Kliazovich *et al.*, "A survey on mobile crowdsensing systems: Challenges, solutions, and opportunities," *IEEE Communications Surveys and Tutorials*, vol. 21, no. 3, pp. 2419–2465, 2019.

[5] J. Wang, Y. Wang, D. Zhang, Q. Lv and C. Chen, "Crowd-powered sensing and actuation in smart cities: Current issues and future directions," *IEEE Wireless Communications*, vol. 26, no. 2, pp. 86–92, 2019.

[6] B. Guo, Z. Wang, Z. Yu, Y. Wang, N. Y. Yen *et al.*, "Mobile crowdsensing and computing: The review of an emerging human-powered sensing paradigm," *ACM Computing Surveys (CSUR)*, vol. 48, no. 1, pp. 1–31, 2015.

[7] J. An, X. Gui, Z. Wang, J. Yang and X. He, "A crowdsourcing assignment model based on mobile crowd sensing in the Internet of Things," *IEEE Internet of Things Journal*, vol. 2, no. 5, pp. 358–369, 2015.

[8] R. K. Ganti, F. Ye and H. Lei, "Mobile crowdsensing: Current state and future challenges," *IEEE Communications Magazine*, vol. 49, no. 11, pp. 32–39, 2011.

[9] K. Ali, D. Al-Yaseen, A. Ejaz, T. Javed and H. S. Hassanein, "Crowdits: Crowdsourcing in intelligent transportation systems," in *2012 IEEE Wireless Communications and Networking Conf. (WCNC)*, Paris, France, pp. 3307–3311, 2012.

[10] Y. Zheng, F. Liu and H. -P. Hsieh, "U-air: When urban air quality inference meets big data," in *Proc. of the 19th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining*, Chicago, USA, pp. 1436–1444, 2013.

[11] X. Wang, J. Zhang, X. Tian, X. Gan, Y. Guan *et al.*, "Crowdsensing-based consensus incident report for road traffic acquisition," *IEEE Transactions on Intelligent Transportation Systems*, vol. 19, no. 8, pp. 2536–2547, 2017.

[12] F. Abbondati, S. A. Biancardo, R. Veropalumbo and G. Dell'Acqua, "Surface monitoring of road pavements using mobile crowdsensing technology," *Measurement*, vol. 171, pp. 108763, 2021.

[13] X. Wang, W. Wu and D. Qi, "Mobility-aware participant recruitment for vehicle-based mobile crowdsensing," *IEEE Transactions on Vehicular Technology*, vol. 67, no. 5, pp. 4415–4426, 2017.

[14] T. Hu, M. Xiao, C. Hu, G. Gao and B. Wang, "A QoS-sensitive task assignment algorithm for mobile crowdsensing," *Pervasive and Mobile Computing*, vol. 41, pp. 333–342, 2017.

[15] W. Gong, B. Zhang and C. Li, "Task assignment in mobile crowdsensing: Present and future directions," *IEEE Network*, vol. 32, no. 4, pp. 100–107, 2018.

[16] J. Peng, Y. Zhu, Q. Zhao, H. Zhu, J. Cao *et al.*, "Fair energy-efficient sensing task allocation in participatory sensing with smartphones," *the Computer Journal*, vol. 60, no. 6, pp. 850–865, 2017.

[17] Q. Zhao, Y. Zhu, H. Zhu, J. Cao, G. Xue *et al.*, "Fair energy-efficient sensing task allocation in participatory sensing with smartphones," in *IEEE INFOCOM 2014-IEEE Conf. on Computer Communications*, Toronto, Ontario, Canada, pp. 1366–1374, 2014.

[18] W. Gong, B. Zhang and C. Li, "Location-based online task assignment and path planning for mobile crowdsensing," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 2, pp. 1772–1783, 2018.

[19] S. Konatowski and P. Pawłowski, “Ant colony optimization algorithm for UAV path planning,” in *2018 14th Int. Conf. on Advanced Trends in Radioelecrtronics, Telecommunications and Computer Engineering (TCSET)*, Lviv-Slavske, Ukraine, pp. 177–182, 2018.

[20] W. Gong, B. Zhang and C. Li, “Location-based online task scheduling in mobile crowdsensing,” in *GLOBECOM 2017–2017 IEEE Global Communications Conf.*, Singapore, Singapore, Asia, pp. 1–6, 2017.

[21] L. Wang, Z. Yu, D. Zhang, B. Guo and C. H. Liu, “Heterogeneous multi-task assignment in mobile crowdsensing using spatiotemporal correlation,” *IEEE Transactions on Mobile Computing*, vol. 18, no. 1, pp. 84–97, 2018.

[22] Y. Xia, B. Zhao, S. Tang and H. -T. Wu, “Repot: Real-time and privacy-preserving online task assignment for mobile crowdsensing,” *Transactions on Emerging Telecommunications Technologies*, vol. 32, no. 5, pp. e4035, 2021.

[23] X. Yin, Y. Chen and B. Li, “Task assignment with guaranteed quality for crowdsourcing platforms,” in *2017 IEEE/ACM 25th Int. Symp on Quality of Service (IWQoS)*, Vilanova i la Geltrú, Spain, Spain, pp. 1–10, 2017.

[24] X. Tao and W. Song, “Location-dependent task allocation for mobile crowdsensing with clustering effect,” *IEEE Internet of Things Journal*, vol. 6, no. 1, pp. 1029–1045, 2018.

[25] Y. Kang, X. Miao, K. Liu, L. Chen and Y. Liu, “Quality-aware online task assignment in mobile crowdsourcing,” in *2015 IEEE 12th Int. Conf. on Mobile Ad Hoc and Sensor Systems*, Dallas, TX, USA, pp. 127–135, 2015.

[26] X. Miao, Y. Kang, Q. Ma, K. Liu and L. Chen, “Quality-aware online task assignment in mobile crowdsourcing,” *ACM Transactions on Sensor Networks (TOSN)*, vol. 16, no. 3, pp. 1–21, 2020.

[27] A. A. A. Gad-ElRab and A. S. Alsharkawy, “Interval tree-based task scheduling method for mobile crowd sensing systems,” *Journal of Communications Software and Systems*, vol. 14, no. 1, pp. 51–59, 2018.

[28] C. Lai and X. Zhang, “Duration-sensitive task allocation for mobile crowd sensing,” *IEEE Systems Journal*, vol. 14, no. 3, pp. 4430–4441, 2020.

[29] M. Xiao, J. Wu, L. Huang, R. Cheng and Y. Wang, “Online task assignment for crowdsensing in predictable mobile social networks,” *IEEE Transactions on Mobile Computing*, vol. 16, no. 8, pp. 2306–2320, 2017.

[30] Y. Wu, F. Li, L. Ma, Y. Xie, T. Li *et al.*, “A Context-aware multiarmed bandit incentive mechanism for mobile crowd sensing systems,” *IEEE Internet Things Journal*, vol. 6, no. 5, pp. 7648–7658, 2019.

[31] Y. Huang, H. Chen, G. Ma, K. Lin, Z. Ni *et al.*, “OPAT: Optimized allocation of time dependent tasks for mobile crowdsensing,” *IEEE Transactions on Industrial Informatics*, pp. 1, 2021.

[32] X. Chen, Y. Kong, X. Fang and Q. Wu, “A fast two-stage ACO algorithm for robotic path planning,” *Neural Computing and Applications*, vol. 22, no. 2, pp. 313–319, 2013.

[33] SNAP: Network datasets: Gowalla. 2009. [Online]. Available: https://snap.stanford.edu/data/loc-gowalla.html.

[34] M. M. Alobaedy, A. A. Khalaf and I. D. Muraina, “Analysis of the number of ants in ant colony system algorithm,” in *2017 5th Int. Conf. on Information and Communication Technology (ICoIC7)*, Melaka, Malaysia, pp. 1–5, 2017.