Multi-Task Allocation in Mobile Crowd Sensing With Mobility Prediction

Jinyi Zhang and Xinglin Zhang, *Member, IEEE*

Abstract—Mobile crowd sensing (MCS) is a popular sensing paradigm that leverages the power of massive mobile workers to perform various location-based sensing tasks. To assign workers with suitable tasks, recent research works investigated mobility prediction methods based on probabilistic and statistical models to estimate the worker's moving behavior, based on which the allocation algorithm is designed to match workers with tasks such that workers do not need to deviate from their daily routes and tasks can be completed as many as possible. In this paper, we propose a new multi-task allocation method based on mobility prediction, which differs from the existing works by (1) making use of workers' historical trajectories more comprehensively by using the fuzzy logic system to obtain more accurate mobility prediction and (2) designing a global heuristic searching algorithm to optimize the overall task completion rate based on the mobility prediction result, which jointly considers workers' and tasks' spatiotemporal features. We evaluate the proposed prediction method and task allocation algorithm using two real-world datasets. The experimental results validate the effectiveness of the proposed methods compared against baselines.

Index Terms—Mobile crowd sensing, multi-task allocation, fuzzy control, mobility prediction

1 Introduction

With the rapid development of wireless network technology and the ubiquity of mobile devices, mobile crowd sensing (MCS) has become more and more prevalent in solving various location-based sensing tasks, such as environmental monitoring [1], urban planning [2], and intelligent transportation [3]. By incorporating the recent advances in 5G and IoT technologies, it is expected that more promising application domains of MCS, as introduced in [4], will arise and benefit people's lives.

Generally, an MCS system consists of three characters: the MCS platform, task requesters, and mobile workers. The MCS platform collects sensing tasks from requesters and then harnesses the sensing capability of a large number of workers with mobile devices (e.g., smartphones and wearables equipped with multi-functional sensors) to complete sensing tasks. A prominent feature here is that a worker needs to physically move to the location of interest in order to finish a task, which essentially influences the system performance regarding the overall task completion efficiency. For example, in Fig. 1, task t_1 is associated with a starting time and a required sensing time interval at school, while workers may arrive at school at different times with different available sensing time duration. Only the workers who meet the spatiotemporal requirements of the task can eventually complete the task. Therefore, research efforts have been devoted to developing efficient task allocation schemes to match workers with the most suitable tasks [4], [5].

Albeit in different optimization forms, existing works on task allocation in MCS largely encounter two challenges. First, to motivate normal mobile users to participate in performing sensing tasks and act as workers, it is essential to assign them with tasks located along their daily routes. In this way, workers can naturally pass by the task locations and willingly complete the task given some rewards. Therefore, how to predict workers' trajectories accurately becomes critical for an attractive and sustainable task allocation scheme. Recent works have tried to use probabilistic trajectory models or statistical results based on workers' historical trajectories. For example, Yang *et al.* [6] predict the probability of workers completing tasks through the Markov model. Wang *et al.* [7] assume that the worker mobility follows an inhomogeneous Poisson process. Wang *et al.* [8] directly employ a statistics-based model to predict the probability that one worker will pass by a specific region during a given time interval.

Second, as there are multiple tasks with constraints (e.g., time and budget) competing for a set of suitable workers, the MCS platform usually needs to solve a complex (generally NP-hard) optimization problem in order to find an allocation solution that is not only socially efficient but also practically feasible. Acquainted with the difficulty in finding the global optimal allocation result, recent efforts mostly use the greedy strategy to design allocation schemes, which select the task-worker pair iteratively and locally, despite of different constraints and utility functions [6], [7], [9].

In this work, we aim to propose a new task allocation scheme by accommodating the above two challenges from different perspectives. First, considering the mobility prediction, we notice that the existing probabilistic models have not mined the historical trajectories comprehensively in the scenario of MCS. Typically, these models consider that a worker's appearance at time τ in region l only contributes to the probability computation of this worker's appearance at time interval $[\tau_a, \tau_d]$ that contains τ (i.e., $\tau \in [\tau_a, \tau_d]$).

• *The authors are with the School of Computer Science and Engineering, South China University of Technology, Guangzhou, Guangdong 510641, China. E-mail: zjinyiz@126.com, zhxlinse@gmail.com.*

Manuscript received 18 Dec. 2020; revised 30 Apr. 2021; accepted 7 June 2021. Date of publication 10 June 2021; date of current version 6 Jan. 2023. (Corresponding author: Xinglin Zhang.) Digital Object Identifier no. 10.1109/TMC.2021.3088291

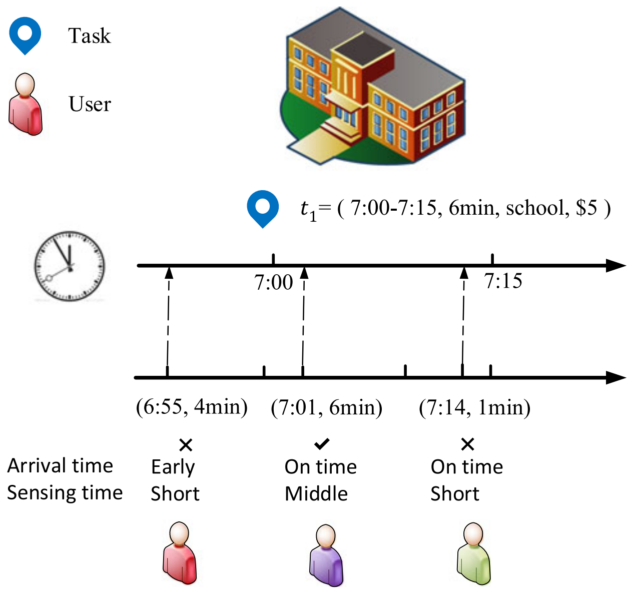


Fig. 1. An example of workers performing a sensing task. (6:55, 4min) means that the worker arrives at 6:55 and spends 4 minutes in sensing.

This is known as a crisp set model, in which an element is either a member of the set or not. However, in the context of predicting the probability of a worker's appearance in a region during a specific time interval $[\tau_a, \tau_d]$, the worker's arrival record $\tau' \notin [\tau_a, \tau_d]$ at adjacent time intervals also increases the belief that the worker may appear during $[\tau_a, \tau_d]$. For example, as shown in Fig. 1, assume that task t_1 requires selected workers to collect sensing data at school l_1 during 7:00-7:15 am and the three workers have historical visit records as depicted. Existing works predict the worker's probability based on the records that he had reached l_1 exactly during 7:00-7:15 am, i.e., workers w_2 and w_3 are counted. However, the worker's records of arriving at l_1 around 7:00-7:15 (e.g., worker w_1 who has an arrival record at 6:55) should also be incorporated to certain extent. Indeed, due to the uncertain factors such as traffic condition and speed variation, the worker's arrival time in a region during his daily routine may fluctuate. In addition, once a worker is allocated to a task with a time interval constraint that largely coincides with his daily routine, it would be easy for the worker to speed up a little bit to perform the task. To portray this key observation, we resort to the fuzzy set theory, a good alternative to handle real-world imprecision, that can characterize a set by a membership function which assigns to each element a grade of membership ranging from zero to one [10]. Further, we establish a **f**uzzy **c**ontrol **s**ystem for **m**obility **p**rediction (FCSMP)), which predicts a worker's probability of appearance at the location of interest during the required time interval for a specified time duration.

Then, based on the aforementioned mobility prediction, we investigate a general multi-task allocation problem to maximize the overall task completion ratio given tasks' different spatiotemporal and budget constraints and workers' availability. We first design a greedy strategy named the **m**ost **l**ikely **f**irst (MLF) algorithm. It allocates a task to a worker who can most likely finish the task. However, considering that greedy strategies tend to be trapped into local optimal solutions for the complex allocation problem, we then propose a **g**reedy-and-**g**enetic enhanced **p**article **s**warm **o**ptimization (GGPSO) algorithm, in which the incorporation of the greedy initialization boosts solution searching and the genetic operations increase the solution diversity for the PSO algorithm. Hence, the proposed GGPSO is able to improve a candidate task allocation solution iteratively and globally with respect to the system utility function.

The main contributions of this paper are briefly summarized as follows:

- We develop a new mobility prediction method, named FCSMP, based on fuzzy control for multi-task allocation in MCS, which is able to make use of workers' historical trajectories more comprehensively and facilitate efficient allocation. To the best of our knowledge, we are the first to consider using fuzzy logic for mobility prediction in MCS.
- We propose two algorithms for the formulated task optimization problem, which is shown to be NP-hard. We design MLF as a greedy strategy, then employ greedy initialization and genetic operations in GGPSO to enhance the optimal solution searching ability of PSO.
- We conduct extensive experiments on two real-world datasets, and the results show that the proposed prediction and allocation approaches outperform the compared schemes.

The remainder of this paper is organized as follows. First, we review related works in Section 2. Then the system model and problem formulation are introduced in Section 3. In Section 4, we introduce the principle of the fuzzy control system and the detailed process of mobility prediction. We demonstrate the two proposed algorithms in Section 5. We conduct experiments on two real-world datasets in Section 6. In Section 7, we discuss the limitations and future directions. Finally, we conclude this paper in Section 8.

2 Related Work

Many task allocation problems have been studied for MCS in recent years. At the beginning, researchers studied the single task allocation problem. For example, In [11], Zhang *et al.* proposed a task assignment framework, which selects a set of workers in each sensing cycle to perform a task based on worker mobility prediction. Recently, some researchers have considered more complex but practical settings of sensing tasks, such as multi-task competition [8], heterogeneous sensing tasks [12], time constraints [13] and participant-side factors [14]. In [8], a multi-task assignment problem was proposed to maximize the spatiotemporal coverage. In [12], Li *et al.* proposed a worker selection problem with heterogeneous sensing tasks and aimed at minimizing the sensing cost. In [13], Li *et al.* proposed a multi-task allocation problem with time constraints, aiming at maximizing the utility of the MCS platform. In [14], Wang *et al.* proposed a novel task allocation framework to optimize overall system utility from the perspective of task organizers and participants. These works select workers to complete MCS tasks during their daily routines. Similar to these works, we also allocate tasks to workers who perform tasks without the need to change their original trajectories.

The multi-task allocation problem in MCS has been studied in recent years and researchers have proposed various multi-task allocation methods from different perspectives. For example, Liu *et al.* [15] proposed a new Minimum Cost Maximum Flow (MCMF) model to solve the multi-task allocation problem efficiently. Zhang *et al.* [16] proposed the greedy approach to solve task assignment problems and proved that it can achieve an approximation ratio of $1/e$ compared to the optimal solution. Cheng *et al.* [17] proposed dynamic programming to maximize the profit of the MCS platform. In addition, some works used evolutionary algorithm to solve the task allocation problem in MCS. Abououf *et al.* [18] designed the genetic algorithm (GA) to allocate a set of workers to a specific task set in order to maximize the QoS of the tasks. Li *at al.* [13] proposed two algorithms based on GA and the experiments showed that the algorithms can maximize the utility of the MCS platform under the workers' and tasks' time constraints. Similarly, Wei *et al.* [19] proposed the genetic based algorithm to optimize task allocation problem. Estrada *et al.* [20] proposed the Particle Swarm Optimization (PSO) technique to maximize the aggregated QoI (Quality of Information)/budget ratio. Similar to these works, we propose an improved evolutionary algorithm to solve the task allocation problem. We design an enhanced PSO algorithm, which combines crossover and mutation operators to increase the solution diversity.

Mobility prediction has been studied in many works on multi-task allocation in MCS. Yang *et al.* [6] predicted the probability of workers completing tasks through the Markov model, which aims at maximizing the probability of completing multiple tasks by selecting a worker set under the budget constraint. Wang *et al.* [21] proposed multi-objective optimization after predicting the worker's trajectory, which aims at maximizing the coverage of the tasks and minimizing the cost of the tasks. They modeled the worker mobility by a statistic model. Wang *et al.* [8] proposed a multi-task assignment algorithm to maximize the spatiotemporal coverage under the constraint of the total budget, and used the Poisson distribution to predict the completion of the task. The Poisson distribution was used in [7] to obtain the probability of workers passing by a certain area in a certain period of time, and an effective task allocation algorithm was proposed to minimize cost and maximize coverage. In [22], [23], the authors also used the Poisson distribution to predict user mobility. Guo *et al.* [24] proposed the problem of multi-task assignment in two cases, one is time-sensitive tasks and the other is delay-tolerant tasks. For the latter, the purpose is to lower the burden of workers by allocating tasks by predicting user mobility. The authors used the statistical result of a worker's location records to model user mobility. Liu *et al.* [9] selected a worker set to perform tasks through the mobility prediction by using Markov model. Lai *et al.* [25] proposed a DSTA model, which aims at maximizing the number of completed tasks with specific sensing duration requirement. The authors modeled the probability of each worker completing each task subject to the requirement of sensing duration based on the exponential distribution.

Different from existing studies, we investigate mobility prediction from a new perspective instead of traditional probabilistic and statistic models, and establish a fuzzy control system for prediction, which can mine worker's trajectories more comprehensively. Then, we propose a heuristic multi-task assignment algorithm based on mobility prediction.

TABLE 1
Main Notations

Symbol	Description
$\mathcal{T} = \{t_1, t_2, \dots, t_n\}$	A set of n sensing Tasks
$\mathcal{W} = \{w_1, w_2, \dots, w_m\}$	A set of m workers
at_i	Task t_i's required arrival time period
ct_i	Task t_i's required sensing time duration
l_i	Task t_i's target location
B_i	Task t_i's recruitment budget
K_j	Worker w_j's maximum workload
c_{ij}	Worker w_j's incentive reward for task t_i
$P(t_i, w_j)$	Probability estimate of w_j performing t_i
p^*_{ijk}	The estimate value of w_j performing t_i for r^j_{ik}
$\mathbb{X}$	The task allocation result
x_{ij}	Binary variable that indicates if w_j is assigned to t_i
$\mathcal{R}$	Workers' historical trajectories
$\mathcal{R}^j$	w_j's historical trajectories
$\mathcal{R}^j_i$	w_j's historical trajectories passing by l_i
r^j_{ik}	the k'th trajectory in R^j_i
$E_i(\mathbb{X})$	Probability of completing t_i given $\mathbb{X}$

3 System Overview and Problem Formulation

In this section, we first introduce the studied MCS framework. Then we formally define the multi-task allocation problem under this framework. Table 1 lists the main notations in this paper.

3.1 System Overview

Fig. 2 shows the MCS framework that sketches the main components for task management and the interaction between different roles. The task requester submits task requests to the MCS platform while the registered worker is willing to perform sensing tasks without disturbing his daily routine. The MCS platform, acting as the central manager, is responsible to maintain the information of requesters and workers (e.g., historical trajectories) and design efficient task allocation schemes to match tasks with appropriate workers. Upon completing the assigned tasks, the worker will submit the sensing data to the platform which in turn will integrate sensing data to generate a sensing report for the corresponding task requester. Note that the main components of interest in this paper are the mobility prediction module and the task allocation module.

- *Mobility Prediction.* This component predicts each worker's mobility using the historical movement trajectory data. Specifically, we establish a fuzzy control system FCSMP which can estimate the completion rate $P(t_i, w_j)$ of worker w_j performing task t_i successfully. The detailed design of FCSMP will be introduced in Section 4.
- *Multi-Task Allocation.* This component selects a subset of task-worker pairs based on the predicted mobility, aiming to optimize the task completion rate subject to requesters' and workers' constraints.

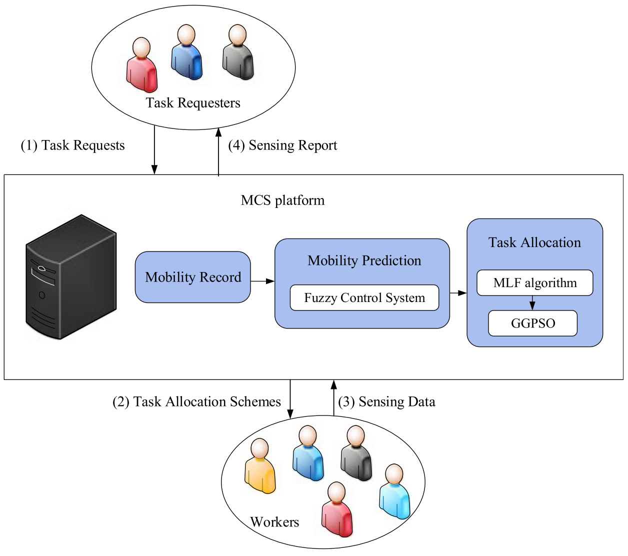


Fig. 2. System framework.

We first design a greedy heuristic algorithm MLF. Then, different from traditional PSO, we propose GGPSO that enhances the optimal solution searching ability. We will demonstrate the proposed algorithms in Section 5.

3.2 Problem Formulation

3.2.1 Preliminary

Before formulating the task allocation problem, we first introduce the definitions of the sensing task and worker in the MCS system.

Definition 1 (Sensing Task). *A sensing task t_i submitted by a requester is associated with a set of attributes denoted by $t_i = \{at_i, ct_i, l_i, B_i\}$, where at_i is the required arrival time period, ct_i is the required sensing time duration for each worker to collect sensing data, l_i is the target location, and B_i is the recruitment budget.*

According to the definition, sensing tasks are heterogeneous in the sense that they have different spatial and temporal requirements. For example, for task $t_i = \{[7{:}00, 7{:}15], 6\text{min}, \text{school}, \$5\}$, it requires workers to collect sensing data for more than $ct_i = 6$ minutes at school during $at_i = [7{:}00, 7{:}15]$. A feasible task allocation result for a task t_i should satisfy all of this task's requirements as illustrated in the definition.

Definition 2 (Worker). *A worker w_j is a mobile user in the MCS system, who is willing to perform sensing tasks along his daily routes subject to the maximum work load constraint K_j.*

As the worker only performs sensing tasks located along his routes and each sensing task has its spatiotemporal constraints, it is essential to predict the worker's probability of completing a task. Therefore, the MCS platform maintains the worker's trajectories, denoted as $\mathcal{R} = \{\mathcal{R}^1, \mathcal{R}^2, \ldots, \mathcal{R}^m\}$, where $\mathcal{R}^j \in \mathcal{R}$ contains worker w_j's historical trajectories. Worker's historical trajectories include the sample time, the sample location and the stay duration. Similar to [9], [26], [27], workers should be rewarded as they consume time, effort and resources (e.g., battery and mobile communication costs). Here we use c_{ij} to denote the incentive reward of worker w_j performing task t_i. For simplicity, we assume that c_{ij} is determined by the system and is attractive to the worker. More details on designing incentive rewards can be found in [28].

3.2.2 Task Completion Estimate

As mentioned above, the MCS platform needs to estimate the probability of each available worker w_j completing a task t_i when designing the allocation scheme. We denote this completion probability value as $P(t_i, w_j)$ and we estimate this value as follows.

Consider a new task t_i with location l_i, we find worker w_j's all historical trajectories passing by l_i, denoted as $\mathcal{R}_i^j$. The kth ($1 \leq k \leq |\mathcal{R}_i^j|$) historical trajectory r_{ik}^j that w_j passed by l_i consists of arrival time at_k and sensing time duration ct_k. For each trajectory $r_{ik}^j \in \mathcal{R}_i^j$, we use the proposed fuzzy logic system FCSMP (as demonstrated in Section 4) to compute the estimated value p_{ijk}^*, which reflects the probability that worker w_j can arrive at l_i according to t_i's requirement. Then, we obtain k such estimate values to compute $P(t_i, w_j)$ as:

$$P(t_i, w_j) = \frac{\sum_{k=1}^{|\mathcal{R}_i^j|} p_{ijk}^*}{|\mathcal{R}^j|}. \tag{1}$$

Note that some workers may have no valid trajectory data for task t_i in the beginning. In this case, we can randomly generate a value for $P(t_i, w_j)$.

3.2.3 Task Allocation Problem

Consider that there are a set of n tasks $\mathcal{T} = \{t_1, t_2, \ldots, t_n\}$ and a set of m workers $\mathcal{W} = \{w_1, w_2, \ldots, w_m\}$ in the MCS system. Since tasks have different spatiotemporal constraints and recruitment budgets and workers only perform tasks following their daily routines, it is difficult to fulfill all tasks in practice. In this paper, we aim to optimize the task completion ratio to satisfy as many tasks as possible.

The optimization problem of designing a **m**ulti-task **allo**cation scheme with **m**obility **p**rediction (MAMP) subject to tasks' and workers' constraints are defined as follows.

Definition 3 (MAMP). *The problem of MAMP is to solve the following optimization problem:*

$$\max \quad \sum_{i=1}^{n} E_i(\mathbb{X}) \tag{2}$$

$$\text{s.t.} \quad \sum_{j=1}^{m} x_{ij} * c_{ij} \leq B_i, \forall t_i \in \mathcal{T} \tag{3}$$

$$\sum_{i=1}^{n} x_{ij} \leq K_j, \forall w_j \in \mathcal{W} \tag{4}$$

where

$$E_i(\mathbb{X}) = \begin{cases} \frac{\sum_{j=1}^{m} P(t_i, w_j) * x_{ij}}{\sum_{j=1}^{m} x_{ij}} & \text{if } \sum_{j=1}^{m} x_{ij} \neq 0 \\ 0 & \text{if } \sum_{j=1}^{m} x_{ij} = 0 \end{cases}, \tag{5}$$

reflects the probability of completing task t_i given the allocation result $\mathbb{X}$, an $n * m$ 0-1 matrix with each entry $x_{ij}(i \in$

$\{1, 2, \ldots, n\}, j \in \{1, 2, \ldots, m\}$) *indicating whether worker* w_j *is assigned to task* t_i.

3.2.4 Problem Analysis

The MAMP problem is a complex optimization problem. Indeed, we show that the MAMP problem is NP-hard by reducing the 0-1 knapsack problem, which is a classic NP-complete problem [29], to an instance of the MAMP problem. First, the 0-1 knapsack problem can be described as follows: Given a set of items $\mathcal{U} = \{u_1, u_2, \ldots, u_n\}$, each item $u_i \in \mathcal{U}$ has weight s_i and value v_i. The decision variable $x_i = 1$ if the item u_i is put in the knapsack. Then the 0-1 knapsack problem aims at maximizing the total value of the items in the knapsack under the constraint of knapsack's capacity S:

$$\max \quad \sum_{i=1}^{n} v_i * x_i \tag{6}$$

$$\text{s.t.} \quad \sum_{i=1}^{n} s_i * x_i \leq S \tag{7}$$

$$x_i \in \{0, 1\}, \; i = 1, 2, \ldots, n \tag{8}$$

Then, we assume that the number of tasks in the MAMP problem is 1 and each worker's maximum workload is 1. We thus can associate the 0-1 knapsack problem and the MAMP problem by mapping the item set U to the set of workers $\mathcal{W}$, the item's weight s_i to the reward c_{ij} for performing the task, the item u_i's value to the contribution of worker w_j performing the task, and the knapsack's capacity to the task's budget B_i. Therefore, the 0-1 knapsack problem is as complex as the simplified MAMP problem, which means that the MAMP problem is NP-hard.

4 Fuzzy Control System

In this section, we elaborate on the design of the fuzzy control system FCSMP for mobility prediction. Different from the traditional probabilistic or statistic models, the fuzzy inference of FCSMP is able to address the stochastic and imprecise nature of the worker's traveling behavior in the scenario of MCS. As shown in Fig. 3, FCSMP takes the worker's trajectory information as input, and then it operates with three components: Fuzzifier, Fuzzy Inference Engine and COG Defuzzifier. Fuzzifier converts crisp values into degrees of matching with linguistic values through membership functions. Fuzzy Inference Engine infers the fuzzy output based on the predefined fuzzy rules. COG Defuzzifier converts the fuzzy output into a crisp value using a typical center of gravity (COG) strategy [30]. The symbols used in FCSMP are listed in Table 2.

4.1 Fuzzifier

In FCSMP, the inputs of arrival time at_k and sensing time duration ct_k are crisp values. Fuzzifier converts crisp values into degrees of matching with linguistic values by using membership functions. This process of conversion is called fuzzification. Before fuzzification, we predefine several linguistic values for the two linguistic variables, i.e., arrival

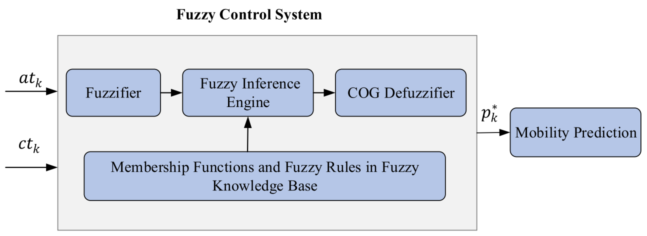


Fig. 3. Fuzzy control system for mobility prediction.

time and sensing time duration. For the arrival time, we use very early (E^+), early (E), on time (OT), late (L) and very late (L^+) as the linguistic values. For the sensing time duration, we use short (S), middle (M) and long (L) as the linguistic values. The linguistic values are quantified by the membership function $\mu(x)$, which reflects the degree of x belonging to a linguistic value. Similar to other membership function decision mechanisms in fuzzy control system applications in the literature (e.g, [10], [31], [32]), the membership functions are defined by the experience and specific application scenarios. As shown in Figs. 4a and 4b, we use the triangle membership function, the trapezoid membership function and the Z-type membership function [33] to describe these linguistic values. Take the curve of the membership function $\mu^L(ct_k)$ in Fig. 4b for illustration. The x-axis represents crisp values of the inputs and the y-axis represents the degree of the input value ct_k belonging to the fuzzy set L. For instance, the sensing time duration $ct_k = 12$ lies in the linguistic value L with a degree of membership 1, denoted as $\mu^L(12) = 1$. Based on these membership functions, Fuzzifier can calculate the degrees of matching with linguistic values for all inputs. For example, if at_k is 6:57 in Fig. 4a, its degree of matching with each linguistic value is expressed as: $\{E^+\text{: } 0,\ E\text{: } 0.6,\ OT\text{: } 0.4,\ L\text{: } 0,\ L^+\text{: } 0\}$. In the same way, if $ct_k = 12$ in Fig. 4b, its degree of matching with each linguistic value is expressed as $\{S\text{: } 0,\ M\text{: } 0,\ L\text{: } 1\}$.

4.2 Fuzzy Inference Engine

Fuzzy Inference Engine infers the fuzzy output from the results of Fuzzifier through fuzzy rules. Fuzzy rules are defined based on linguistic values, expressed as the IF-THEN rule with a condition and a conclusion [34]. In addition to the above defined linguistic values, we also define very unlikely (UL^+), unlikely (UL), likely (L), more likely (ML), most likely (ML^+) as the linguistic values for the output variable p_k. Here, similar to other fuzzy control system applications in the literature (e.g., [10], [31], [35]), we empirically determine the fuzzy rules based on the domain knowledge on MCS. Some rules defined in our fuzzy control system are shown in Table 3.

Next, we show how to infer the fuzzy output through fuzzy rules. The fuzzy inference is composed of three steps: combination, activation and accumulation. In the combination phase, we perform fuzzy set operations (e.g., AND and OR) for the IF part. Specifically, we use Minimum and Maximum functions for AND and OR operators, respectively [36]. Take Rule 2 in Table 3 as an example: IF the arrival time is early (E) AND the sensing time is long (L) THEN the estimated value is more likely (ML). We use the Minimum function for the AND operator in the IF part, denoted as $\min\{\mu^E(at_k), \mu^L(ct_k)\}$.

TABLE 2
Definitions of Linguistic Variables

Linguistic variable	Description	Membership function
arrival time	very early (E^+)	$\mu^{E^+}(at_k)$
	early (E)	$\mu^{E}(at_k)$
	on time (OT)	$\mu^{OT}(at_k)$
	late (L)	$\mu^{L}(at_k)$
	very late (L^+)	$\mu^{L^+}(at_k)$
sensing time	short (S)	$\mu^{S}(ct_k)$
	middle (M)	$\mu^{M}(ct_k)$
	long (L)	$\mu^{L}(ct_k)$
output variable	very unlikely (UL^+)	$\mu^{UL^+}(p_k)$
	unlikely (UL)	$\mu^{UL}(p_k)$
	likely (L)	$\mu^{L}(p_k)$
	more likely (ML)	$\mu^{ML}(p_k)$
	most likely (ML^+)	$\mu^{ML^+}(p_k)$

The activation phase determines how the evaluated result in the IF part is applied to the THEN part of the fuzzy rule. Note that the fuzzy rule reflects the relationship between the input linguistic variables and the output linguistic variables. Given the fuzzy rule, we need to use a method to map the input values of the linguistic variables to the output fuzzy set. For example, IF the arrival time is at_k^* AND the sensing time is ct_k^*, what is the output fuzzy set μ_{Rule2}^{ML}? Here, we use the Mamdani method [37] to compute the output fuzzy set μ_{Rule2}^{ML}. Specifically, the Mamdani method computes μ_{Rule2}^{ML} using the Minimum functions as follows:

$$\mu_{Rule2}^{ML}(p_k) = \min\{\min\{\mu^E(at_k^*), \mu^L(ct_k^*)\}, \mu^{ML}(p_k)\}. \tag{9}$$

The activation process of other fuzzy rules is similar to that of Rule 2. After the activation phase, we obtain several new

TABLE 3
Fuzzy Rules

Rule Index	at_k	ct_k	p_k
Rule 1	E	S	UL^+
Rule 2	E	L	ML
Rule 3	OT	M	ML^+
Rule 4	OT	L	ML^+
Rule 5	L	S	UL^+

fuzzy sets $\mu_{Rule1}^{UL^+}(p_k)$, $\mu_{Rule2}^{ML}(p_k)$, $\mu_{Rule3}^{ML^+}(p_k)$, $\mu_{Rule4}^{ML^+}(p_k)$ and $\mu_{Rule5}^{UL^+}(p_k)$.

The accumulation phase then determines how to combine these new fuzzy sets to obtain the fuzzy output. Specifically, we use the Maximum function in the accumulation phase. So the fuzzy inference results for all rules can be computed as follows:

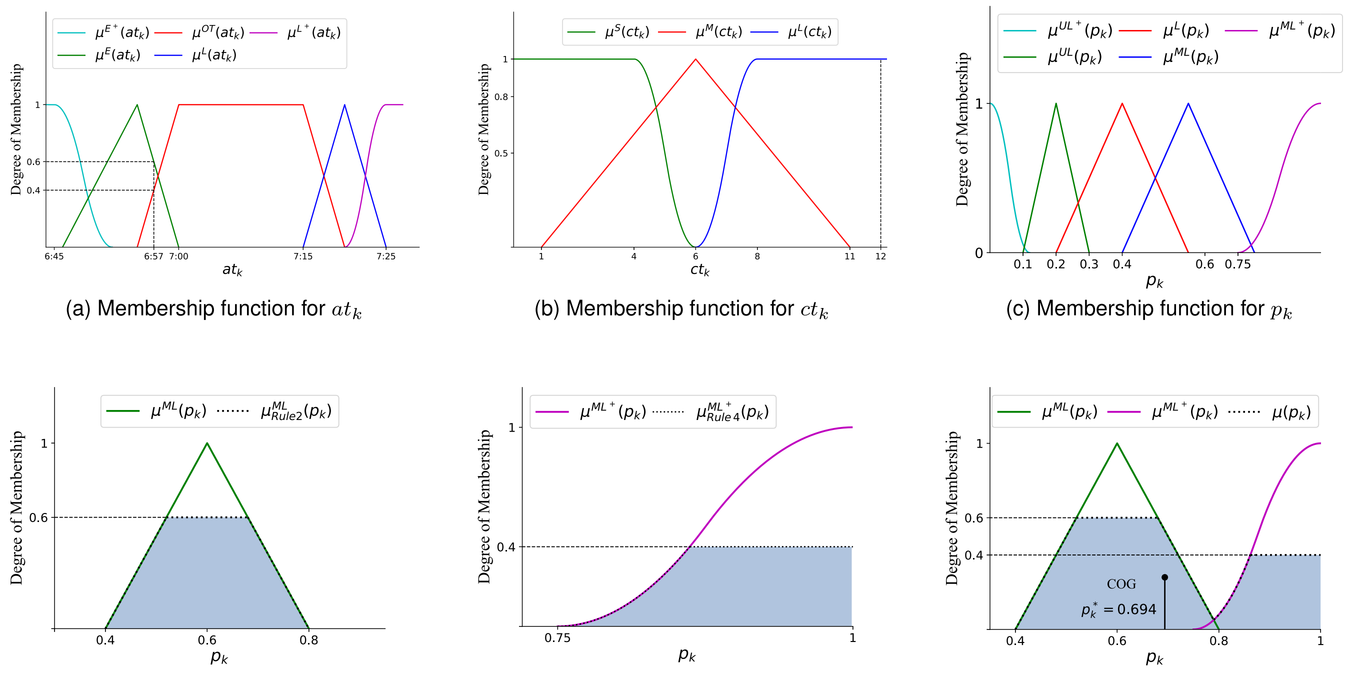
$$\mu(p_k) = \max\{\mu_{Rule1}^{UL^+}(p_k), \mu_{Rule2}^{ML}(p_k), \ldots, \mu_{Rule5}^{UL^+}(p_k)\}. \tag{10}$$

4.3 COG Defuzzifier

The output of Fuzzy Inference Engine is a fuzzy value, which is converted to a crisp value by Defuzzifier. In this paper, we use a typical center of gravity (COG) strategy which returns the center of gravity of the area under the curve for defuzzication [30]. The output of our fuzzy control system p_{ijk}^* can be calculated by using:

$$p_{ijk}^* = \frac{\int p_k \mu(p_k) dp_k}{\int \mu(p_k) dp_k}, \tag{11}$$

Finally, we use a simple example to intuitively illustrate the computation flow of the above components of our FCSMP.



(a) Membership function for at_k (b) Membership function for ct_k (c) Membership function for p_k

(d) Activation phase in Rule 2 (e) Activation phase in Rule 4 (f) Defuzzification operations

Fig. 4. Illustration of FCSMP.

Example. Assume that task $t_i = \{7{:}00\text{-}7{:}15, 6\text{min}, l_1, \$9\}$ and one record in worker w_j's kth historical trajectory is $r^j_{ik} = \{6{:}57, 12\text{min}\}$. We also assume that there are only ve rules as listed in Table 3. Then the fuzzy control system works as follows:

- Fuzzifier takes the two inputs $at_k = 6{:}57$ and $ct_k = 12$ and maps them by using membership functions in Figs. 4a and 4b, respectively. The fuzzification results of at_k and ct_k are degrees of matching with the linguistic values $\{E^+\text{: } 0, E\text{: } 0.6, OT\text{: } 0.4, L\text{: } 0, L^+\text{: } 0\}$ and $\{S\text{: } 0, M\text{: } 0, L\text{: } 1\}$, respectively.
- Fuzzy Inference Engine computes the fuzzy output $\mu(p_k)$ based on fuzzy rules. Since the sensing time duration $ct_k = 12$ lies in the linguistic variable S and M with a degree of membership 0 and the arrival time $at_k = 6:57$ lies in the linguistic variable L with a degree of membership 0, the combination and activation phases are only applied for Rule 2 and Rule 4:

$$\mu^{ML}_{Rule\,2}(p_k) = \min\{0.6, \mu^{ML}(p_k)\} \tag{12}$$

$$\mu^{ML^+}_{Rule\,4}(p_k) = \min\{0.4, \mu^{ML^+}(p_k)\}, \tag{13}$$

  Figs. 4d and 4e display the fuzzy sets $\mu^{ML}_{Rule2}(p_k)$ and $\mu^{ML^+}_{Rule4}(p_k)$, respectively. Then the fuzzy output can be calculated in the accumulation phase using Eq. (10). The curve of the fuzzy output $\mu(p_k)$ is shown in Fig. 4f.
- COG Defuzzifier transforms the fuzzy output into the crisp value 0.694 (as shown in Fig. 4f) through defuzzication. Hence this crisp value is used to estimate $P(t_i, w_j)$.

5 Task Allocation Algorithm

The MAMP problem is a multi-task allocation problem which is NP-hard. It has such a large solution space that performing an exhaustive search is not feasible. In order to increase the task completion rate, we first think of allocating tasks to workers who are most likely to complete them. Therefore, we first design a local greedy strategy MLF. However, MLF may get trapped into local optimal solutions. So we consider using an effective global heuristic algorithm to solve this problem. Specifically, we propose an enhanced PSO algorithm, GGPSO, which incorporates greedy initialization to boost the solution searching and considers genetic operators that can effectively jump out of the local optimal solution to increase the solution diversity.

5.1 Most Likely First Algorithm

The key idea of MLF is to iteratively assign a task to a worker who can most likely complete the task. The pseudo code of our proposed MLF is shown in Algorithm 1. In each iteration, we select the task allocation decision x_{ij} with the maximum $P(t_i, w_j)$, the probability estimate of worker w_j performing task t_i successfully. Then, we assign task t_i to worker w_j if the two constraints are satisfied. Finally, we update the available task set $\mathcal{T}^\#$ and the available worker set $\mathcal{W}^\#$. If the budget of task t_i is exhausted, we will remove t_i from $\mathcal{T}^\#$. Similarly, if the number of assigned tasks of worker w_j reaches his maximum capacity, we will remove w_j from $\mathcal{W}^\#$. The iteration is terminated when all tasks have been assigned or there are no available workers on the MCS platform.

Algorithm 1. MLF Algorithm

```
Input: Task set $\mathcal{T}$; Worker set $\mathcal{W}$; Workers' historical trajectories set $\mathcal{R}$
Output: Task allocation result $\mathbb{X}$
 1:  $\mathbb{X} \leftarrow 0$;
 2:  Initialize available task set $\mathcal{T}^\# \leftarrow T$;
 3:  Initialize available worker set $\mathcal{W}^\# \leftarrow \mathcal{W}$;
 4:  Calculate $P(t_i, w_j)$ for all task-worker pairs according to Eq. (1);
 5:  while $\mathcal{T}^\# \neq \emptyset$ and $\mathcal{W}^\# \neq \emptyset$ do
 6:    Select task allocation decision $x_{ij}$ with the maximum $P(t_i, w_j)$;
 7:    if $\sum_{j'=1}^{m} x_{ij'} * c_{ij'} + c_{ij} \leq B_i$ and $\sum_{i'=1}^{n} x_{i'j} + 1 \leq K_j$ then
 8:      $x_{ij} = 1$;
 9:      Update $\mathcal{T}^\#$ and $\mathcal{W}^\#$;
10:    end if
11:    Update $P(t_i, w_j)$ for all task-worker pairs;
12:  end while
```

5.2 GGPSO Algorithm

Considersing that MLF may fall into local optimal solutions, we design a global heuristic algorithm called greedy-and-genetic enhanced particle swarm optimization algorithm (GGPSO).

PSO is an evolutionary algorithm which simulates predation behavior of birds. A group of birds is looking for food in the forest with one food. All birds have no prior knowledge about the position of the food, but know how far they are from the food. The simplest strategy is to search the area around the bird closest to the food by constantly changing the birds' positions and velocities. We abstract the bird's position as the solution of the MAMP problem and the distance as the fitness function. PSO iteratively tries to find a candidate solution to maximize the system utility function. In order to boost the solution searching process, we initialize the particle swarm based on the output of the MLF algorithm. We also try to maintain the diversity of the solutions during searching, hence we introduce operations including crossover, mutation and selection. The pseudo code of the proposed GGPSO is shown in Algorithm 2.

First of all, we initialize velocity $\mathbb{V}_g$ of each particle g randomly and initialize position $\mathbb{P}_g$ of each particle g using the output of the MLF algorithm, then we can calculate the fitness value of each particle g according to fitness function $U(\mathbb{P}_g)$. Next, we update the best ever position $\mathbb{B}^{local}_g$ of each particle g and the best global position $\mathbb{B}^{global}$. By comparing the fitness value of each particle in the particle swarm, the particle's position with the maximum fitness value is determined as the best global position $\mathbb{B}^{global}$. By comparing the fitness value of particle g in historical iterations, the position with the maximum fitness value is determined as the best local position $\mathbb{B}^{local}_g$. In each iteration h, GGPSO first updates particles' positions through crossover, mutation, and selection operations. Then, it updates $\mathbb{B}^{local}_g$ and $\mathbb{B}^{global}$. Next, GGPSO will update the velocities and positions by Eqs. (18) and (19). Each particle g updates its position based on two extreme values. The first extreme value is the best local position $\mathbb{B}_g^{local}$ found by particle g, while the other extreme value is the best global position $\mathbb{B}^{global}$ found by the particle swarm. $\mathbb{B}_g^{local}$ and $\mathbb{B}^{global}$ will be updated after updating the velocity and position. GGPSO repeats the above operations until the number of iterations reaches the threshold $MaxIteration$. The details of GGPSO are demonstrated in the following.

Algorithm 2. GGPSO Algorithm

```
Input: Task set $\mathcal{T}$; Worker set $\mathcal{W}$; Workers' historical trajectories set $\mathcal{R}$
Output: Task allocation result $\mathbb{X}$
1:  for each particle $g$ do
2:      Initialize velocity $\mathbb{V}_g$ and position $\mathbb{P}_g$ for particle $g$ based on the output of Algorithm 1;
3:      Evaluate the fitness value of $\mathbb{P}_g$ for particle $g$;
4:      Set the best local position $\mathbb{B}_g^{local}$ for particle $g$;
5:  end for
6:  Set the best global $\mathbb{B}^{global}$ of particle swarm;
7:  while $h \leq MaxIteration$ do
8:      Perform crossover, mutation and selection operations to each particle;
9:      Update the best local position $\mathbb{B}_g^{local}$ and the best global position $\mathbb{B}^{global}$;
10:     Update the velocity and position of each particle according to Eq. (18) and Eq. (19);
11:     Evaluate the fitness value for each particle;
12: end while
13: $\mathbb{X} = \mathbb{B}^{global}$;
```

5.2.1 Particle's Position Representation

We use an $n * m$ 0-1 matrix $\mathbb{P}_g$ to represent particle g's position, which represents one task allocation solution for the MAMP problem. For example, the matrix in Fig. 5 shows a task allocation solution with 6 tasks and 8 workers. If task t_i has been assigned to worker w_j, $p_{ij} = 1$; otherwise $p_{ij} = 0$. The solution is feasible as long as the following two constraints are met: t_i's budget B_i and w_j's maximum workload K_j.

5.2.2 Position and Velocity Initialization

We randomly set an initial velocity for each particle g. In general, the initial position is randomly generated. Considering the search space of the MAMP problem is large, we use the result of our proposed MLF algorithm (as sketched in Algorithm 1) to generate the initial position of each particle. Specifically, we first set all particles' positions as the result of the MLF algorithm. Then, we adjust each particle randomly by changing some elements of the matrix.

5.2.3 Crossover

Crossover operation is used to produce new particles. Two new particles are created through crossover operation by exchanging genes (particle's position). For GGPSO, we use column exchange for the crossover operation. Specifically, each particle exchanges the columns on the same column index with the particle randomly selected from the particle swarm with probability p_c. Fig. 5 gives an example of crossover operation, we randomly select the 6th column to exchange the particles' genes.

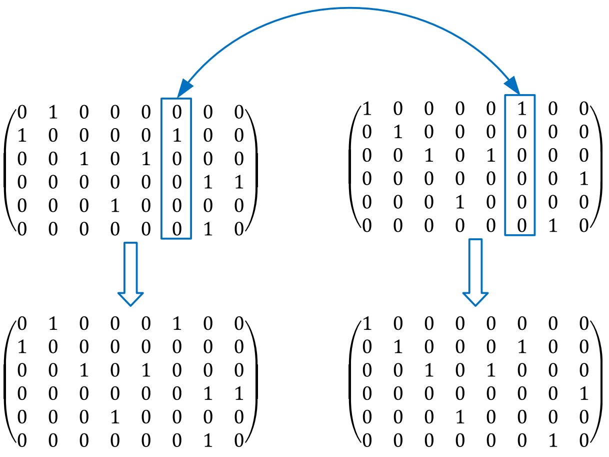


Fig. 5. Illustration of the crossover operation.

5.2.4 Mutation

Mutation operation can increase the particle swarm diversity by randomly changing genes. In GGPSO, we randomly select several columns to change some elements with probability p_m. As shown in Fig. 6, GGPSO selects the first row and changes some of its elements through mutation operation.

5.2.5 Selection

Selection operation retains particles with better fitness values. In GGPSO, we evaluate the fitness values of new particles generated by crossover and mutation operations. Then, we compare the new particle's fitness value with the original particle's fitness value before the mutation and crossover operations. If the fitness value of the new particle is larger, we replace the original particle with the new particle.

5.2.6 Fitness Function With Penalty Mechanism

The traditional PSO algorithm was first proposed for unconstrained continuous optimization problems, while the MAMP problem we proposed is a constrained optimization problem. Therefore, we cannot directly use $E_i(\cdot)$ in Eq. (2) as the fitness function for optimization. Instead, we incorporate the penalty mechanism [38] to solve the MAMP problem. We define the penalty function relating to the level of violation of each constraint as follows:

$$Penalty(\mathbb{P}_g) = \lambda\left(\sum_{i=1}^{n} f_{1i}(\mathbb{P}_g) + \sum_{j=1}^{m} f_{2j}(\mathbb{P}_g)\right), \tag{14}$$

where λ is a constant factor, and the functions $f_{1i}(\mathbb{P}_g)$ and $f_{2j}(\mathbb{P}_g)$ are defined as:

$$f_{1i}(\mathbb{P}_g) = \max\left\{0, \sum_{j=1}^{m} x_{ij} * c_{ij} - B_i\right\} \tag{15}$$

$$f_{2j}(\mathbb{P}_g) = \max\left\{0, \sum_{i=1}^{n} x_{ij} - K_j\right\}. \tag{16}$$

$$\begin{pmatrix} \boxed{0\ 1\ 0\ 0\ 0\ 0\ 0\ 0} \\ 1\ 0\ 0\ 0\ 0\ 1\ 0\ 0 \\ 0\ 0\ 1\ 0\ 1\ 0\ 0\ 0 \\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 1 \\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 0 \\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 0 \end{pmatrix} \Rightarrow \begin{pmatrix} 0\ 0\ 0\ 0\ 1\ 0\ 0\ 1 \\ 1\ 0\ 0\ 0\ 0\ 1\ 0\ 0 \\ 0\ 0\ 1\ 0\ 1\ 0\ 0\ 0 \\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 1 \\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 0 \\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 0 \end{pmatrix}$$

Fig. 6. Illustration of the mutation operation.

Therefore, the fitness function $U(\mathbb{P}_g)$ can be calculated as:

$$U(\mathbb{P}_g) = \sum_{i=1}^{n} E_i(\mathbb{P}_g) - Penalty(\mathbb{P}_g). \tag{17}$$

5.2.7 *Velocity and Position Update*

We adopt the classic position and velocity formulas in GGPSO:

$$\mathbb{V}_g^{h+1} = \omega \mathbb{V}_g^h + c_1 r_1 (\mathbb{B}_g^{local} - \mathbb{P}_g^h) + c_2 r_2 (\mathbb{B}^{global} - \mathbb{P}_g^h) \tag{18}$$

$$\mathbb{P}_g^{h+1} = \mathbb{P}_g^h + \mathbb{V}^{h+1}, \tag{19}$$

where $\mathbb{V}_g^h$ is the velocity of particle g in the hth iteration, $\mathbb{P}_g^h$ is the position of particle g in the hth iteration, $\mathbb{B}_g^{local}$ is the best ever position of g, $\mathbb{B}^{global}$ is the best global position of the particle swarm, r_1 and r_2 are random numbers in [0,1], ω is the inertial weight, and c_1 and c_2 are learning factors.

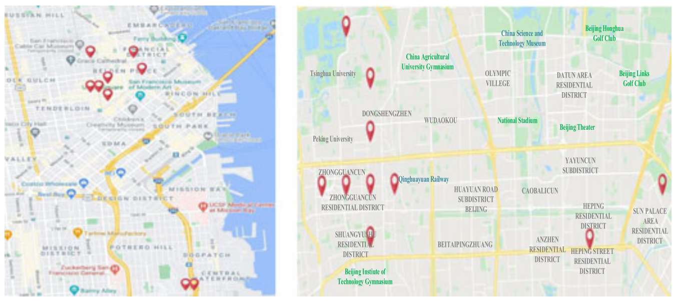
6 Evaluation

In this section, we evaluate the performance of the proposed algorithms using two real-world datasets. We first introduce the datasets and baseline methods. Then, we compare the evaluation results of the proposed algorithms and baselines in terms of mobility prediction and task allocation.

6.1 Datasets

The first dataset we used is San Francisco dataset [39], which contains the trajectories of approximately 500 taxis within one month. Each trajectory contains the following information: taxi id, sample time and sample position including latitude and longitude. In experiments, we filter some locations and taxis having few trajectories, and select 100 active taxis as the available worker set and 10 popular locations as tasks' target locations (Fig. 7a). Due to the lack of the taxi's stay duration at the sample position in the original dataset, we randomly generate the sensing time duration from 1 to 15 minutes. Our experiment uses the data of the first 19 days (training data) for conducting the worker's mobility prediction. Then we evaluate the task completion rate according to the task allocation plan obtained by running the allocation algorithm based on the mobility prediction on the data of the last 12 days (testing data).

The second dataset is GeoLife trajectory dataset [40], [41], which collects 187 users' trajectories in their daily lives. The dataset contains 17,621 trajectories with a total distance of 1,292,951 kilometers and a total duration of 50,176 hours. Each trajectory contains the user id, sample time and sample location. We first filter some locations and users having few trajectories. Then we select 100 active users as available



(a) San Francisco (b) Geolife

Fig. 7. Dataset distribution.

worker set and 10 popular locations (Fig. 7b) as tasks' target locations. We randomly generate the user's stay duration from 1 to 15 minutes as the sensing time duration. Due to the long time span of the geolife trajectory dataset, we divide it into 128 two-week periods of sub-datasets. Then we select the 128 first-week data (training data) for worker's mobility prediction and the 128 second-week data (testing data) to evaluate the task completion rate.

6.2 Parameter settings

The main parameters (as listed in Table 4) are divided into three parts: task-related, worker-related and GGPSO-related. For the tasks, their target locations are randomly distributed to our selected popular locations. Task t_i's required arrival time is randomly generated from 00:00-23:59 and the required sensing time duration is assumed to last from 1 to 12 minutes. The budget for each task obeys a uniform distribution. The average budget of all tasks is denoted as u in Table 4. For the workers, the reward for the worker is randomly set between 1 to 3. The maximum number of tasks they can perform along their daily routes is randomly generated from 1 to 4. For GGPSO, the main parameters include the number of particles G, the number of iterations $MaxIteration$, the crossover probability p_c, the mutation probability p_m , the learning factors c_1 and c_2, and the inertial weight ω. We set $G = 80$, $MaxIteration = 120$, $p_c = 0.9$, $p_m = 0.01$, $c_1 = 1.5$ and $c_2 = 1.5$ following the common settings for these parameters. The value of ω varies with the number of iterations.

6.3 Baselines

To verify the performance of our proposed mobility prediction FCSMP, we implement the following recent mobility prediction models for comparison: the mobility predition method based on statistical probability (MPP) [8] and the mobility prediction method based on Poisson distribution (MPPo) [7].

- *MPP*. This mobility prediction model counts the number of times that worker w_j arrives at location l_i in the time period at_i and stay there for ct_i, with which the probability $P(t_i, w_j)$ of worker w_j performing task t_i is obtained.
- *MPPo*. The model assumes that the worker's mobility behavior follows an inhomogeneous Poisson process. Specifically, it counts the average times that worker w_j reaches task t_i's destination l_i in the arrival time period of each task and stay for no less

TABLE 4
Parameter Settings

Parameters	Settings
n	$20, 40, 60, 80, 100$
m	$20, 40, 60, 80, 100$
u	$U(2,6), U(3,9), U(4,12), U(5,15), U(6,18)$
c_{ij}	Randomly generated from 1 to 3
K_j	Randomly generated from 1 to 4
at_i	Randomly generated from 00:00-23:59
l_i	Randomly generated from 1 to 10
ct_i	Randomly generated from 1 to 12
ω	$0.4 \sim 0.9$
p_c	0.9
p_m	0.01
c_1	1.5
c_2	1.5
λ	1000

than the required sensing time (denoted as $\lambda_{i,j}$). For example, to estimate $\lambda_{i,j}$, task t_i requires workers to arrive at the library from 7:00-7:15 and spend 5 minutes to collect the data. The model counts the average times that worker w_j reaches the library during the same period and stay for no less than 5 minutes in the historical records. Then, the completion probability that worker w_j performs task t_i is predicted as:

$$P(t_i, w_j) = 1 - e^{-\lambda_{i,j}}. \tag{20}$$

In order to demonstrate the efficiency of GGPSO, we provide the following baseline methods for comparison.

- *GA*. We use the traditional genetic algorithm (GA) [42] as a baseline, which simulates the evolution process in the nature world. GA randomly generates initial population and it searches for the optimal solution by using crossover, selection, and mutation operators.
- *PSO*. The second evolutionary algorithm for comparison is the traditional PSO [43]. Compared with GGPSO, PSO does not contain operations such as crossover, mutation and selection. PSO randomly initializes particle swarm positions and updates positions and velocities to find the optimal solution.
- *GMWS*. We implement Group based Multi-task Worker Selection (GMWS) algorithm proposed in [18] for comparison. The GMWS algorithm allocates workers to tasks based on an improved genetic algorithm, which uses the roulette strategy for selection operation. GMWS is adjusted to select the feasible solution by Eq. (2) instead of the task's QoS. Considering the different constraints of the problems, we also adapt the operation constraints of GMWS to satisfy the MAMP problem.
- *GSMS*. The last compared algorithm is Gale-Shapley Matching game Selection (GSMS) algorithm proposed in [44]. GSMS forms stable matching between multiple workers and multiple tasks based on two preference lists for task requesters' and workers' preferences. Due to the difference of the optimization goals, we create the preference list for the worker by ranking the tasks in a descending order of the task's reward c_{ij}. Then the task requester's preference list is defined by ranking the workers in a descending order of the worker's probability of completing task $P(t_i, w_j)$. We also adapt the operation constraints in GSMS to satisfy the MAMP problem.

In each experiment, we randomly generate the workers' cost distribution, workers' maximum workload distribution and task's budget distribution 10 times and take the average values of these 10 times as the experimental result. And we run GGPSO 10 times to get the average values in each case.

6.4 Performance of Mobility Prediction

We first evaluate the prediction accuracy of the mobility prediction models. As the accuracy of mobility prediction has a direct and essential influence on the task completion ratio, we then conduct several experiments under different number of tasks, workers and budgets to further verify the effectiveness of our proposed FCSMP. The corresponding allocation results based on different prediction models are obtained by running GGPSO.

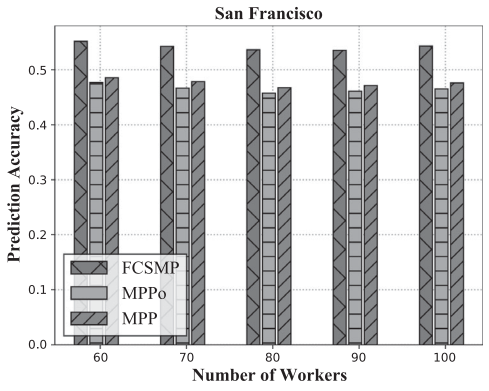
6.4.1 Prediction Accuracy

Fig. 8 depicts the prediction accuracies of FCSMP, MPPo, and MPP on San Francisco and Geolife datasets. As can be seen, given the varying numbers of workers, the proposed FCSMP method outperforms the compared methods in all circumstances. Specifically, the accuracy of FCSMP is on average 7.6 percent higher than MPPo and 6.6 percent higher than MPP on the San Francisco dataset, while on the Geolife dataset, FCSMP outperforms MPPo by 10.9 percent and MPP by 8.9 percent, respectively. The results demonstrate that the proposed FCSMP can make better prediction, and the reason is that FCSMP considers the worker's imprecise arrival time in a region during his daily routine that may fluctuate due to uncertain factors.

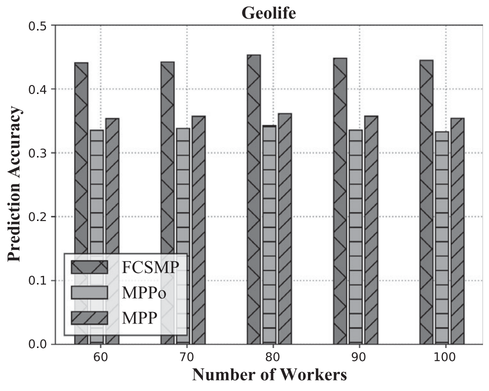
6.4.2 The Influence of the Prediction Method on the Task Allocation Result

Fig. 9 shows the average task completion rates by running GGPSO based on the prediction results of FCSMP, MPP, and MPPo. On the San Francisco dataset, we can see that with the increment of the number of tasks, the task completion ratios of all three methods decrease (Fig. 9a). This is because that the available workers are fixed in this context and thus the total number of tasks these workers can complete keep constant. On average, FCSMP, which has better mobility prediction, outperforms MPP and MPPo. When we vary the number of workers while keeping other parameters fixed, the task completion ratios exhibit increasing trends for the three methods. This is because the maximum number of tasks that workers can complete increases now. Again, the proposed FCSMP based allocation method achieves better performance, compared to MPP and MPPo. Finally, we vary the average budget for each task. The performance trends of all methods are not influenced much with different budgets (Fig. 9c).

The results on the Geolife dataset by varying the number of tasks, the number of workers, and the value of average budgets are shown in Figs. 9d, 9e, and 9f, respectively. As



(a) San Francisco



(b) Geolife

Fig. 8. Performance comparison for different mobility prediction methods.

can be seen, the performance trends are similar to the results on the San Francisco dataset in each scenarios. Also, the proposed FCSMP based allocation method outperforms the compared methods consistently in all settings.

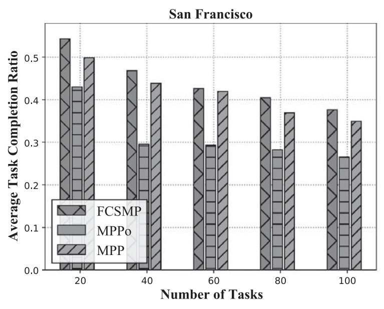
In summary, the mobility prediction result directly influence the final allocation result, and thus the proposed FCSMP based allocation method achieves the best performance compared with the other mobility prediction based methods.

6.5 Performance of Task Allocation

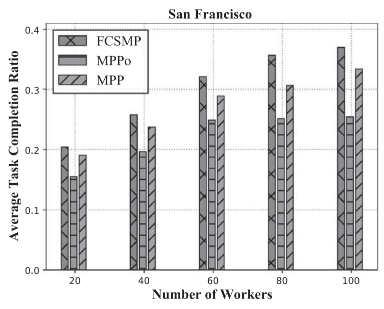
Recall that the MAMP problem is a multi-task allocation problem and its goal is to maximize the average task completion rate. We here examine the performance of the proposed task allocation algoirhtm GGPSO and MLF by comparing with PSO and GA. There are three factors that directly affect the task allocation result: the number of tasks n, the number of workers m and the average task budget u. Therefore, we compare the performance of the algorithms by changing these factors.

6.5.1 The Effect of n

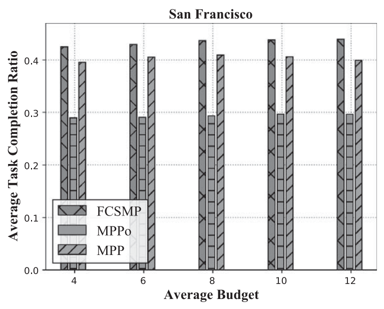
From the curves on the San Francisco dataset in Fig. 10a, we can see that the average task completion ratio decreases with the increasing number of tasks for GGPSO, MLF, and GSMS. This is because the competition for the limited resources (workers) among different tasks becomes more intense when the number of tasks increases. The performance decreasing trends of PSO, GA, and GMWS are relatively mild because the ratios of these two algorithms are much smaller than GGPSO and MLF and thus are less sensitive to the number of tasks. It can also be observed from the figure that the proposed GGPSO outperforms the proposed MLF by 7.3 percent, and outperforms the baselines GA, PSO, GMWS and GSMS by 29.8, 17.2, 22.5 and 13.1 percent higher average task completion ratio, respectively. The performance trends on the Geolife dataset (Fig. 10d) exhibit similar patterns as the San Francisco dataset. On average, the ratios of GGPSO are 8.5, 39.8, 38.3, 35.9 and 16.6 percent higher than MLF, PSO, GA, GMWS and GSMS, respectively. It can be observed from Table 5 that GGPSO can



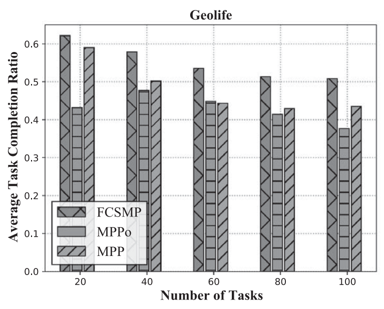
(a) $m = 100, u = 6$



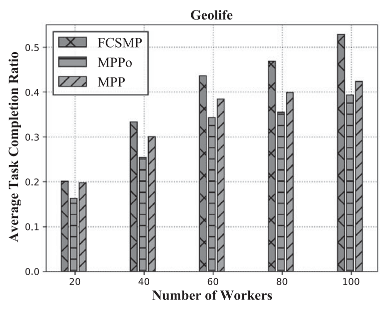
(b) $n = 100, u = 6$



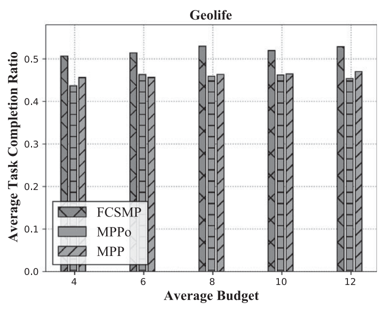
(c) $n = 60, m = 100$



(d) $m = 100, u = 6$



(e) $n = 100, u = 6$



(f) $n = 60, m = 100$

Fig. 9. Allocation results with different mobility prediction methods.

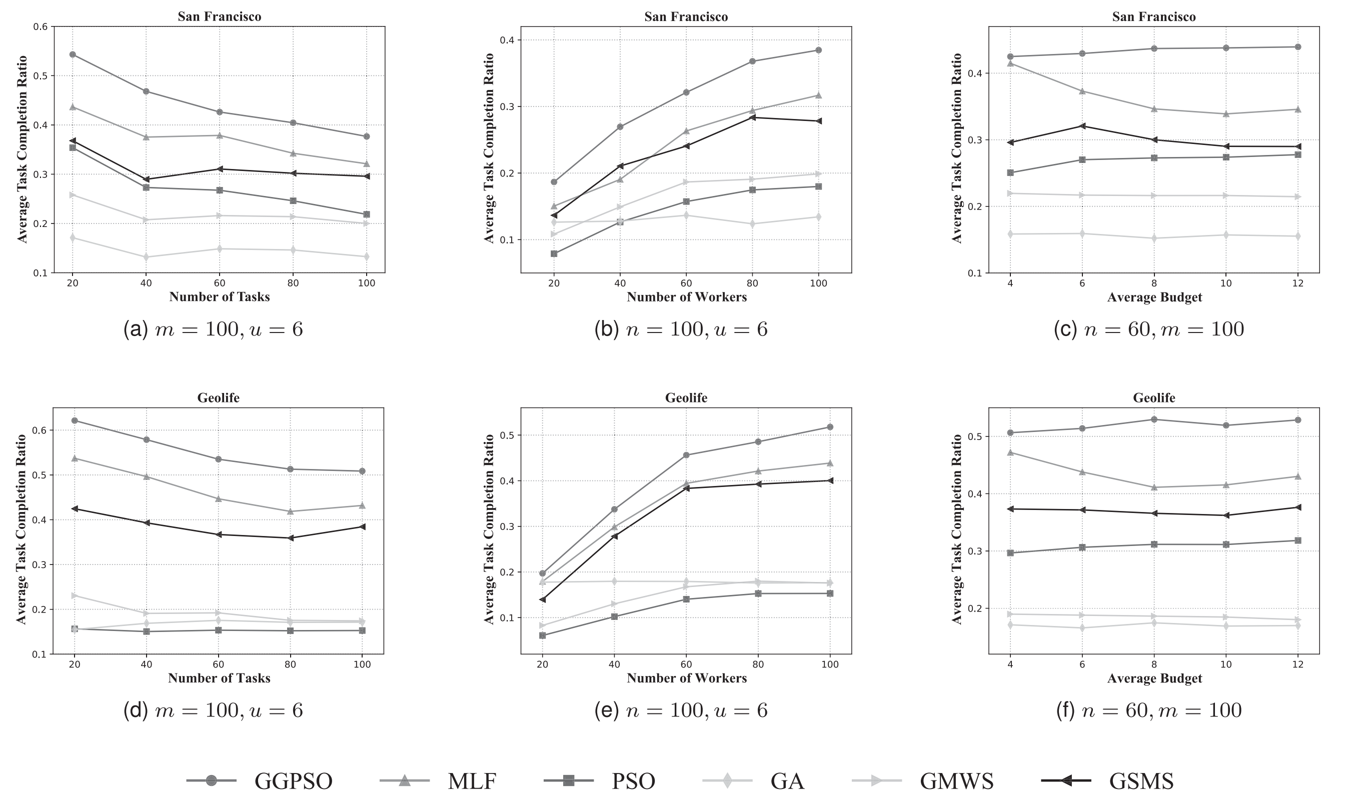


Fig. 10. Average task completion ratio.

achieve fewer average number of tasks allocated per worker compared with MFL on two real-world datasets. This means that GGPSO not only achieves a higher system target value, but also saves the task requester's budget and worker's workload.

6.5.2 The Effect of m

In Fig. 10b, we compare the task completion ratios of different algorithms under different number of workers on the San Francisco dataset. It can be observed that as the number of workers increases, the average task completion ratio increases. On average, GGPSO outperforms MLF by 6.3 percent, GA by 17.6 percent, PSO by 16.5 percent, GMWS by 13.9 percent and GSMS by 7.6 percent. The results indicate that when the worker resources become more abundant, the proposed algorithm can keep its advantage of matching the tasks with the most suitable workers. The results on the Geolife dataset (Fig. 10e) have the similar trends as San Francisco dataset. Numerically, the average ratios of GGPSO are 5.2, 27.7, 22.1, 25.2 and 8.0 percent higher than MLF, PSO, GA, GMWS and GSMS respectively.

TABLE 5
Average Number of Tasks Per Worker

Number of tasks	San Francisco		Geolife	
	GGPSO	MLF	GGPSO	MLF
20	1.18	1.39	1.27	1.41
40	1.45	1.87	1.31	1.72
60	1.62	2.26	1.49	2.16
80	1.69	2.49	1.62	2.46
100	1.87	2.51	1.79	2.46

6.5.3 The Effect of u

Fig. 10c shows the average task completion ratio on the San Francisco dataset under different task budgets. We can see that GGPSO outperforms MLF, PSO, GA, GMWS and GSMS in all settings, with an average improvement of 7.0, 16.5, 27.7, 21.7 and 13.5 percent, respectively. Note that there is no obvious relationship between the average task completion ratio and the increasing budget. The reason is that although the average budget of the task is constantly increasing, the number of workers is limited and the uncertain factors that affect the completion of the task will increase. We can also see that the performance of GGPSO is not obvious when the average budget is small. This is because when the budget is small, the number of workers that can be hired for each task is small. When the number of workers is limited, the performance of GGPSO and MLF is almost the same. Fig. 10f shows the results on the Geolife dataset. It can be seen that the results exhibit similar patterns as those on the San Francisco dataset. On average, GGPSO achieves 8.6, 21.1, 35.0, 33.4 and 15.0 percent higher task completion ratios than MLF, PSO, GA, GMWS and GSMS, respectively.

7 Discussion

In this section, we discuss the limitations of our proposed FCSMP and multi-task allocation model.

- The experimental results show that the proposed FCSMP outperforms baselines in terms of mobility prediction and that mobility prediction affects the overall system performance. However, FCSMP may need to adjust membership functions and fuzzy rules to obtain the best performance for different applications. This limitation can be addressed by investigating neural networks to learn membership functions [45] and rule bases [46], [47] to achieve automatic parameter tuning in the future study.

- The preferences of workers/tasker requesters also play an important role in MCS. Recently, some task allocation problems that focus on preferences of workers/task requesters have been investigated. In [44], Abououf *et al.* utilize the distance that workers are willing to travel to characterize workers' preferences on tasks and utilize workers' QoS to characterize task requesters' preferences on workers. Wu *et al.* [48] assign tasks to workers based on the workers' preferences. The authors characterize workers' preferences by exploiting their implicit feedback (e.g., their task browsing histories or task selection records). Considering that task requesters want to receive high-quality services, we assign the tasks to the workers who can most likely finish the tasks in our work. However, we do not consider the workers' preferences directly. In our future work, we may infer the workers' preferences through their historical task selection/completion records and allocate the tasks to the workers based on their preferences.

8 Conclusion

In this paper, we study a general multi-task allocation problem which allocates tasks with different spatiotemporal requirements to workers to maximize the average task completion rate. In order to allocate tasks to the most suitable workers who do not need to deviate from their daily routes, we propose a prediction method based on the fuzzy control system, FCSMP, to predict worker's mobility. Based on this prediction result, we design heuristic allocation algorithms. Through the experimental results on two real-world datasets, we obtain the following findings. The proposed FCSMP predicts the worker's trajectory more accurately than traditional prediction methods based on statistical models, which means that the fuzzy system is a new promising direction for trajectory prediction in MCS. On the other hand, the proposed GGPSO algorithm achieves the best allocation performance compared with the baselines, indicating that GGPSO accommodates the problem's challenges well and thus can find a superior solution. As a future work, we plan to apply neural networks to learn membership functions and fuzzy rules, so that FCSMP can adjust the parameters adaptively. In addition, it is also helpful to take into account workers' preferences in the multi-task allocation problem to improve workers' satisfaction.

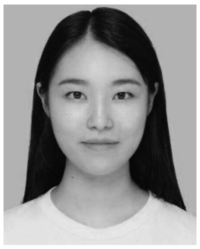
Acknowledgments

This work was supported in part by the National Natural Science Foundations of China under Grants 61632013 and 61872149, in part by the Natural Science Foundations of Guangdong Province for Distinguished Young Scholar under Grant 2018B030306010, in part by the Guangdong Special Support Program under Grant 2017TQ04X482, and in part by the Fundamental Research Funds for the Central Universities.

References

[1] F. Montori, L. Bedogni, and L. Bononi, "A collaborative Internet of Things architecture for smart cities and environmental monitoring," *IEEE Internet Things J.*, vol. 5, no. 2, pp. 592–605, Apr. 2018.
[2] T. J. Matarazzo *et al.*, "Crowdsensing framework for monitoring bridge vibrations using moving smartphones," *Proc. IEEE*, vol. 106, no. 4, pp. 577–593, Apr. 2018.
[3] S. Hu, L. Su, H. Liu, H. Wang, and T. F. Abdelzaher, "SmartRoad: Smartphone-based crowd sensing for traffic regulator detection and identification," *ACM Trans. Sensor Netw.*, vol. 11, no. 4, pp. 1–27, Jul. 2015.
[4] A. Capponi, C. Fiandrino, B. Kantarci, L. Foschini, D. Kliazovich, and P. Bouvry, "A survey on mobile crowdsensing systems: Challenges, solutions, and opportunities," *IEEE Commun. Surveys Tuts.*, vol. 21, no. 3, pp. 2419–2465, Jul.–Sep. 2019.
[5] Y. Liu, L. Kong, and G. Chen, "Data-oriented mobile crowdsensing: A comprehensive survey," *IEEE Commun. Surveys Tuts.*, vol. 21, no. 3, pp. 2849–2885, Jul.–Sep. 2019.
[6] Y. Yang, W. Liu, E. Wang, and J. Wu, "A prediction-based user selection framework for heterogeneous mobile crowdsensing," *IEEE Trans. Mobile Comput.*, vol. 18, no. 11, pp. 2460–2473, Nov. 2019.
[7] J. Wang *et al.*, "Multi-task allocation in mobile crowd sensing with individual task quality assurance," *IEEE Trans. Mobile Comput.*, vol. 17, no. 9, pp. 2101–2113, Sep. 2018.
[8] L. Wang, Z. Yu, D. Zhang, B. Guo, and C. H. Liu, "Heterogeneous multi-task assignment in mobile crowdsensing using spatiotemporal correlation," *IEEE Trans. Mobile Comput.*, vol. 18, no. 1, pp. 84–97, Jan. 2019.
[9] W. Liu, Y. Yang, E. Wang, Z. Han, and X. Wang, "Prediction based user selection in time-sensitive mobile crowdsensing," in *Proc. 14th Annu. IEEE Int. Conf. Sens., Commun., Netw.*, 2017, pp. 1–9.
[10] C. Sonmez, A. Ozgovde, C. Ersoy, C. Sonmez, A. Ozgovde, and C. Ersoy, "Fuzzy workload orchestration for edge computing," *IEEE Trans. Netw. Serv. Manage.*, vol. 16, no. 2, pp. 769–782, Jun. 2019.
[11] D. Zhang, H. Xiong, L. Wang, and G. Chen, "Crowdrecruiter: Selecting participants for piggyback crowdsensing under probabilistic coverage constraint," in *Proc. ACM Int. Joint Conf. Pervasive Ubiquitous Comput.*, 2014, pp. 703–714.
[12] H. Li, T. Li, W. Wang, and Y. Wang, "Dynamic participant selection for large-scale mobile crowd sensing," *IEEE Trans. Mobile Computing*, vol. 18, no. 12, pp. 2842–2855, Dec. 2019.
[13] X. Li and X. Zhang, "Multi-task allocation under time constraints in mobile crowdsensing," *IEEE Trans. Mobile Comput.*, vol. 20, no. 4, pp. 1494–1510, Apr. 2021.
[14] J. Wang, F. Wang, Y. Wang, D. Zhang, B. Lim, and L. Wang, "Allocating heterogeneous tasks in participatory sensing with diverse participant-side factors," *IEEE Trans. Mobile Comput.*, vol. 18, no. 9, pp. 1979–1991, Sep. 2019.
[15] Y. Liu, B. Guo, Y. Wang, W. Wu, Z. Yu, and D. Zhang, "Taskme: Multi-task allocation in mobile crowd sensing," in *Proc. ACM Int. Joint Conf. Pervasive Ubiquitous Comput.*, 2016, pp. 403–414.
[16] X. Zhang, Z. Yang, Y. Liu, and S. Tang, "On reliable task assignment for spatial crowdsourcing," *IEEE Trans. Emerg. Topics Comput.*, vol. 7, no. 1, pp. 174–186, Sep. 2019.
[17] P. Cheng, X. Lian, L. Chen, J. Han, and J. Zhao, "Task assignment on multi-skill oriented spatial crowdsourcing," *IEEE Trans. Knowl. Data Eng.*, vol. 28, no. 8, pp. 2201–2215, Aug. 2016.
[18] M. Abououf, R. Mizouni, S. Singh, H. Otrok, and A. Ouali, "Multi-worker multi-task selection framework in mobile crowd sourcing," *J. Netw. Comput. Appl.*, vol. 130, pp. 52–62, 2019.
[19] X. Wei, Y. Wang, J. Tan, and S. Gao, "Data quality aware task allocation with budget constraint in mobile crowdsensing," *IEEE Access*, vol. 6, pp. 48010–48020, 2018.
[20] R. Estrada, R. Mizouni, H. Otrok, A. Ouali, and J. Bentahar, "A crowd-sensing framework for allocation of time-constrained and location-based tasks," *IEEE Trans. on Serv. Comput.*, vol. 13, no. 5, pp. 769–785, Sep./Oct. 2020.
[21] L. Wang, Z. Yu, Q. Han, B. Guo, and H. Xiong, "Multi-objective optimization based allocation of heterogeneous spatial crowdsourcing tasks," *IEEE Trans. Mobile Comput.*, vol. 17, no. 7, pp. 1637–1650, Jul. 2018.
[22] J. Wang, Y. Wang, D. Zhang, F. Wang, Y. He, and L. Ma, "Psallocator: Multi-task allocation for participatory sensing with sensing capability constraints," in *Proc. ACM Conf. Comput. Supported Cooperative Work. Soc. Comput.*, 2017, pp. 1139–1151.

[23] M. Xiao, J. Wu, L. Huang, Y. Wang, and C. Liu, “Multi-task assignment for crowdsensing in mobile social networks,” in *Proc. IEEE Conf. Comput. Commun.*, 2015, pp. 2227–2235.
[24] B. Guo, L. Yan, W. Wu, Z. Yu, and H. Qi, “ActiveCrowd: A framework for optimized multitask allocation in mobile crowdsensing systems,” *IEEE Trans. Hum.-Mach. Syst.*, vol. 47, no. 3, pp. 392–403, Jun. 2017.
[25] C. Lai and X. Zhang, “Duration-sensitive task allocation for mobile crowd sensing,” *IEEE Syst. J.*, vol. 14, no. 3, pp. 4430–4441, Sep. 2020.
[26] X. Wang, W. Wu, and D. Qi, “Mobility-aware participant recruitment for vehicle-based mobile crowdsensing,” *IEEE Trans. Veh. Technol.*, vol. 67, no. 5, pp. 4415–4426, May 2018.
[27] J. Wang, F. Wang, Y. Wang, L. Wang, Z. Qiu, D. Zhang, B. Guo, and Q. Lv, “HyTasker: Hybrid task allocation in mobile crowd sensing,” *IEEE Trans. Mobile Comput.*, vol. 19, no. 3, pp. 598–611, Mar. 2020.
[28] X. Zhang *et al.*, “Incentives for mobile crowd sensing: A survey,” *IEEE Commun. Surveys Tuts.*, vol. 18, no. 1, pp. 54–67, Jan.–Mar. 2016.
[29] S. Khuri, T. Bäck, and J. Heitkötter, “The zero/one multiple knapsack problem and genetic algorithms,” in *Proc. ACM Symp. Appl. Comput.*, 1994, pp. 188–193.
[30] K. M. Passino and S. Yurkovich, *Fuzzy Control*. Boston, MA, USA: Addison-Wesley, 2001.
[31] F. Basic, A. Aral, and I. Brandic, “Fuzzy handoff control in edge offloading,” in *Proc. IEEE Int. Conf. Fog Comput.*, 2019, pp. 87–96.
[32] S. A. Soleymani *et al.*, “A secure trust model based on fuzzy logic in vehicular ad hoc networks with fog computing,” *IEEE Access*, vol. 5, pp. 15619–15629, 2017.
[33] J. M. Mendel, “Fuzzy logic systems for engineering: A tutorial,” *Proc. IEEE*, vol. 83, no. 3, pp. 345–377, Mar. 1995.
[34] Q. Luo, C. Li, T. H. Luan, and W. Shi, “EdgeVCD: Intelligent algorithm-inspired content distribution in vehicular edge computing network,” *IEEE Internet Things J.*, vol. 7, no. 6, pp. 5562–5579, Jun. 2020.
[35] W.-J. Chang and F.-L. Hsu, “Mamdani and takagi-sugeno fuzzy controller design for ship fin stabilizing systems,” in *Proc. 12th Int. Conf. Fuzzy Syst. Knowl. Discov.*, 2015, pp. 345–350.
[36] R. L. de Mantaras and L. Godo, “From fuzzy logic to fuzzy truth-valued logic for expert systems: A survey,” in *Proc. 2nd IEEE Int. Conf. Fuzzy Syst.*, 1993, pp. 750–755.
[37] R. R. Panda and M. Reza, “Optimization of buffer overflow probability in jackson queueing networks using mamdani fuzzy inferecnce system,” in *Proc. 2nd Int. Conf. Bus. Inf. Manage.*, 2014, pp. 92–98.
[38] G. Liang, P. Liyuan, L. Ruihuan, Z. Fen, L. Jinhui, and W. Xin, “Study on economic operation for micro-grid based on scenario and PSO,” in *Proc. Int. Conf. Power Syst. Technol.*, 2014, pp. 3152–3156.
[39] P. Michal, S. D. Natasa, and G. Matthias, “Crawdad dataset epfl/mobility (v. 2009–02-24),” Feb. 2009. [Online]. Available: https://crawdad.org/epfl/mobility/20090224
[40] Y. Zheng, L. Zhang, X. Xie, and W.-Y. Ma, “Mining interesting locations and travel sequences from GPS trajectories,” in *Proc. 18th Int. Conf. World Wide Web*, 2009, pp. 791–800.
[41] Z. Chen, H. T. Shen, X. Zhou, Y. Zheng, and X. Xie, “Searching trajectories by locations: An efficiency study,” in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, 2010, pp. 255–266.
[42] G. R. Harik, F. G. Lobo, and D. E. Goldberg, “The compact genetic algorithm,” *IEEE Trans. Evol. Comput.*, vol. 3, no. 4, pp. 287–297, Nov. 1999.
[43] J. Kennedy and R. Eberhart, “Particle swarm optimization,” in *Proc. Int. Conf. Neural Netw.*, 1995, pp. 1942–1948.
[44] M. Abououf, S. Singh, H. Otrok, R. Mizouni, and A. Ouali, “Gale-shapley matching game selection-a framework for user satisfaction,” *IEEE Access*, vol. 7, pp. 3694–3703, 2019.
[45] R. Wang and K. Mei, “Analysis of fuzzy membership function generation with unsupervised learning using self-organizing feature map,” in *Proc. Int. Conf. Comput. Intell. Secur.*, 2010, pp. 515–518.
[46] M. Elkano, J. A. Sanz, E. Barrenechea, H. Bustince, and M. Galar, “CFM-BD: A distributed rule induction algorithm for building compact fuzzy models in big data classification problems,” *IEEE Trans. Fuzzy Syst.*, vol. 28, no. 1, pp. 163–177, Jan. 2020.
[47] M. Pratama, W. Pedrycz, and G. I. Webb, “An incremental construction of deep neuro fuzzy system for continual learning of nonstationary data streams,” *IEEE Trans. Fuzzy Syst.*, vol. 28, no. 7, pp. 1315–1328, Jul. 2020.
[48] F. Wu, S. Yang, Z. Zheng, S. Tang, and G. Chen, “Fine grained user profiling for personalized task matching in mobile crowdsensing,” *IEEE Trans. Mobile comput.*, early access, May 02, 2020, doi: 10.1109/TMC.2020.2993963.



Jinyi Zhang received the BE degree in information security from the South China University of Technology in 2020. She is currently working toward the master’s degree at the South China University of Technology. Her research focuses on mobile crowdsensing.

Xinglin Zhang (Member, IEEE) received the BE degree from the School of Software from Sun Yat-sen University in 2010 and the PhD degree from the Department of Computer Science and Engineering, Hong Kong University of Science and Technology, in 2014. He is currently an associate professor with the South China University of Technology. His research interests include mobile crowdsensing, edge computing, and mobile computing. He is a member of the the ACM.