Postprint: Mobile Crowdsensing Task Allocation Method Based on Data Redundancy Control

Authors: He Xingyu, Zhao Dan, Yang Guisong, Jin Ziri, Yangkailong Qin, Wang Qipei

Date: 2022-04-07T15:01:57Z

Abstract

In mobile crowdsensing systems, tasks exhibit spatiotemporal coverage overlap, which may lead to redundant data collection and consequently cause data redundancy issues. To address this, we propose a task allocation method that can simultaneously control both intra-task and inter-task data redundancy. The method first proposes a trajectory sequence prediction model based on Long Short-Term Memory (LSTM) neural networks to perform fine-grained spatiotemporal unit-based trajectory sequence prediction for task participants. Subsequently, based on the trajectory prediction results, an optimization model for minimizing data redundancy is proposed. By minimizing the data redundancy degree of spatiotemporal units, the data redundancy problem within individual tasks is controlled, and by maximizing the reuse of sensing data from individual task participants across spatiotemporal units, the data redundancy arising from spatiotemporal coverage overlap among multiple tasks is controlled. Experimental results demonstrate that the proposed task allocation method can effectively reduce both intra-task and inter-task data redundancy.

Full Text

Preamble

Task Allocation Method Based on Data Redundant Control in Mobile Crowd Sensing

He Xingyu^{a,b}, Zhao Dan^{a}, Yang Guisong^{a†}, Jin Ziri^{b}, Qin Yangkailong^{b}, Wang Qipei^{b}

^{a}School of Optical-Electrical & Computer Engineering, ^{b}College of Communication & Art Design, University of Shanghai for Science & Technology, Shanghai 200093, China

Abstract: Due to the overlap of time and space coverage between tasks in mobile crowd sensing systems, repeated data collection may occur and cause data redundancy problems. To address this issue, we propose a task allocation method that can simultaneously control data redundancy within and between tasks. The method first introduces a trajectory sequence prediction model based on Long Short-Term Memory (LSTM) neural networks to predict the trajectory sequences of task participants within subdivided spatiotemporal units. Based on the trajectory prediction results, we then propose an optimization model that minimizes data redundancy. This model controls data redundancy within a single task by minimizing the data redundancy metric in each spatiotemporal unit, and controls data redundancy caused by spatiotemporal coverage overlap between multiple tasks by maximizing the reuse of sensing data from individual participants across spatiotemporal units. Experimental results demonstrate that the proposed task allocation method can effectively reduce data redundancy both within and between tasks.

Keywords: mobile crowd sensing; data redundancy; trajectory sequence prediction; optimization model

0 Introduction

Mobile Crowd Sensing (MCS) has emerged as a novel IoT sensing paradigm that leverages the sensing capabilities of mobile devices, attracting widespread attention from both academia and industry. Compared with traditional static sensor networks, MCS utilizes sensors embedded in mobile devices and the mobility of participants to perceive the surrounding environment, achieving broader spatiotemporal coverage without incurring substantial costs or time investments. MCS has found extensive applications in environmental monitoring, urban management, and other scenarios. Sensing quality and cost represent crucial performance metrics in mobile crowd sensing, where sensing quality is primarily measured by spatiotemporal coverage and cost control generally manifests in managing participant compensation.

To balance sensing quality and cost, some studies aim to achieve full coverage or maximize coverage range under budget constraints, while others seek to minimize sensing costs while meeting coverage requirements. Further cost control efforts have begun addressing data redundancy in task allocation. Previous research indicates that once tasks achieve certain coverage levels, adding more participants yields marginal improvements in sensing results while increasing costs unnecessarily.

Existing task allocation strategies primarily focus on data redundancy within individual tasks. Some studies analyze spatiotemporal correlations between data to enable high-precision data inference, thereby avoiding redundancy and reducing sensing costs. Others consider participant uncertainty and uncontrollability, analyzing mobility patterns to develop reasonable task allocation methods. Certain approaches minimize redundant data collection among multiple participants

serving the same task to reduce data redundancy. To ensure adequate data collection while preventing excessive redundancy, some works define redundancy factors to compute data quality or set maximum thresholds for sample collection per task, though these methods overlook inter-task redundancy. Other research proposes graph similarity models for fine-grained image selection to address redundancy in uploaded images among participants. Some approaches introduce task redundancy factors into matching functions to penalize redundant task assignments, thereby avoiding intra-task redundancy and saving budgets to complete more tasks. Within the compressive sensing domain, some methods exploit implicit correlations among sensing data to reduce redundancy while selecting appropriate user groups to guarantee spatiotemporal coverage of sensing grids. However, these methods assume fixed sensing regions for participants, whereas in MCS scenarios, participants are mobile and their mobility significantly impacts sensing quality.

Beyond intra-task redundancy, inter-task redundancy also presents a critical challenge. Specifically, when two tasks require data from the same spatiotemporal unit and different participants are assigned to collect this data separately, task-level redundancy emerges. Although existing MCS task allocation methods analyze inter-task correlations, their objectives typically focus on cost minimization. Some studies consider spatiotemporal inclusion relationships between tasks to minimize incentive costs, while others define correlation metrics between tasks to reduce participant costs. However, these approaches ignore the impact of inter-task correlations on redundancy control, particularly the data redundancy caused by spatiotemporal coverage overlap among multiple tasks. Moreover, reducing inter-task data redundancy is extremely challenging as it requires considering different spatiotemporal granularities across tasks while balancing data quality and incentive payment differences.

To achieve data redundancy control during task allocation and address these challenges, we must predict participants' longer-term mobility trajectory sequences. Existing studies typically employ deep learning methods, Markov models, or probability models based on historical information for trajectory prediction. However, conventional RNNs suffer from limitations in addressing long-term dependencies and contextual generalization for long sequence prediction, Markov model complexity increases rapidly with sequence length, and probability-based or statistical analysis methods exhibit low accuracy. In contrast, Long Short-Term Memory (LSTM) neural networks are better suited for effectively handling gradient vanishing and explosion problems in long sequence training. Therefore, we utilize LSTM neural networks for long trajectory sequence prediction. While typical LSTM-based trajectory prediction models use short-term historical sequences as input to obtain future trajectories, we enhance prediction accuracy by analyzing participants' long-term trajectory information, constructing a long-term visit probability matrix as model input, and employing an encoder-decoder framework for continuous trajectory sequence prediction.

In summary, this paper proposes a task allocation method that controls both

intra-task and inter-task data redundancy while meeting sensing quality constraints, leveraging LSTM neural networks to analyze participants' future mobility trajectories. Our main contributions are:

a) To enable more accurate prediction of participants' future trajectories, we analyze long-term visit probabilities, define a spatiotemporally relevant visit probability matrix, and propose an LSTM-based participant mobility trajectory sequence prediction model that improves prediction accuracy.

b) To control costs, we propose a data redundancy minimization optimization model based on participant mobility trajectories. This model minimizes spatiotemporal unit data redundancy through fine-grained time slot partitioning to control intra-task redundancy, while simultaneously analyzing spatiotemporal coverage overlap among multiple sensing tasks to maximize reuse of participant sensing data across spatiotemporal units, thereby reducing inter-task redundancy and lowering platform costs.

1 System Model and Problem Definition

In our system, a day is divided into K equal-length time slots, with all slots forming set $\mathcal{T} = \{T_1, T_2, ..., T_K\}$, where the k-th slot is denoted as T_k. The MCS activity area is partitioned into G grids, with all regions forming set $\mathcal{R} = \{R_1, R_2, ..., R_G\}$, where the g-th partition is denoted as R_g. Grid partitioning is easy to implement and highly scalable, allowing adjustment of grid width and length to achieve different granularity control.

The system publishes a series of crowd sensing tasks, with M tasks forming task set $\mathcal{S} = \{s_1, s_2, ..., s_M\}$. For any task s_i, there exist required time range $\mathcal{T}_{s_i} \subseteq \mathcal{T}$ and space range $\mathcal{R}_{s_i} \subseteq \mathcal{R}$. Here, $\mathcal{T}_{s_i}$ represents the time slots that task s_i needs to sense, being any specified slots from system partition $\mathcal{T}$, i.e., $\mathcal{T}_{s_i} = \{T_{p_{s_i,1}}, ..., T_{p_{s_i,t}}, ..., T_{p_{s_i,|\mathcal{T}_{s_i}|}}\}$, where $|\mathcal{T}_{s_i}|$ denotes the number of time slots required by task s_i. $\mathcal{R}_{s_i}$ represents the partitions that task s_i needs to sense, being any specified regions from system partition $\mathcal{R}$, i.e., $\mathcal{R}_{s_i} = \{R_{q_{s_i,1}}, ..., R_{q_{s_i,r}}, ..., R_{q_{s_i,|\mathcal{R}_{s_i}|}}\}$, where $|\mathcal{R}_{s_i}|$ is the number of partitions required by task s_i.

Assume the system has N participants, with participant set $\mathcal{W} = \{w_1, w_2, ..., w_N\}$. For any participant w_j, $w_j \in \mathcal{W}$ can access task details in the sensing system and participate at any time.

Each task s_i has a budget B_{s_i} for task allocation compensation and requires participants to be within the sensing time range $\mathcal{T}_{s_i}$ and space range $\mathcal{R}_{s_i}$ to provide valid sensing data. To facilitate representing task completion, we define a "time-partition pair" as a binary group formed by any time slot $T_{p_{s_i}}$ and any partition $R_{q_{s_i}}$ in task s_i. The set of time-partition pairs contained in task s_i is denoted as $\mathcal{L}_{s_i} = \{(T_{p_{s_i}}, R_{q_{s_i}}) | T_{p_{s_i}} \in \mathcal{T}_{s_i}, R_{q_{s_i}} \in \mathcal{R}_{s_i}\}$. To ensure the quality of returned sensing results, at least L sensing data reports must be collected in

each time-partition pair.

To achieve balanced data distribution across a task's time-partition pairs and reduce data redundancy, we further divide each time slot into finer granularities by uniformly partitioning each slot into L equal-length sub-slots. For example, dividing slot $T_{p_{s_i}}$ into $T_{p_{s_i}}^{(1)}, T_{p_{s_i}}^{(2)}, ..., T_{p_{s_i}}^{(L)}$, where $T_{p_{s_i}}^{(\tau)}$ represents the τ-th sub-slot in time slot $T_{p_{s_i}}$. We name the binary group of subdivided sub-slots and partitions as "spatiotemporal units," so one time-partition pair contains L spatiotemporal units. For instance, a time-partition pair $(T_{p_{s_i}}, R_{q_{s_i}})$ in task s_i can be represented after fine-grained division as $(T_{p_{s_i}}^{(\tau)}, R_{q_{s_i}})$, where $T_{p_{s_i}}^{(\tau)}$ is the τ-th sub-slot divided from time slot $T_{p_{s_i}}$. Therefore, within each task, to achieve more balanced data collection, we hope that data collected in each time-partition pair is evenly distributed across its divided spatiotemporal units.

Additionally, for each participant w_j, due to device and other constraints, each participant can participate in at most ω spatiotemporal units of data collection work in the system.

As shown in Figure 1, assuming task s_1 has time range $\mathcal{T}_{s_1} = \{T_1, T_2, T_3\}$ and space range $\mathcal{R}_{s_1} = \{R_1, R_2, R_3, R_4\}$, the set of time-partition pairs is $\mathcal{L}_{s_1} = \{(T_1, R_1), (T_2, R_2), (T_3, R_3)\}$, illustrated by green and blue boxes. Task s_2 has time range $\mathcal{T}_{s_2} = \{T_2, T_3, T_4\}$ and space range $\mathcal{R}_{s_2} = \{R_2, R_3, R_4\}$, with time-partition pairs $\mathcal{L}_{s_2} = \{(T_2, R_2), (T_3, R_3), (T_4, R_4)\}$, shown by orange and blue boxes. Taking time slot T_3 as an example, to achieve balanced data distribution within T_3, we further divide T_3 into three sub-slots $T_3^{(1)}$, $T_3^{(2)}$, and $T_3^{(3)}$. Our task allocation method addresses each spatiotemporal unit contained in time-partition pairs to ensure collected data is evenly distributed across fine-grained units rather than concentrating in the same unit, thereby controlling data redundancy in each spatiotemporal unit.

Moreover, tasks s_1 and s_2 share identical time-partition pairs (T_2, R_2) and (T_3, R_3), shown as blue boxes in the figure. In this case, spatiotemporal units $(T_3^{(1)}, R_3)$, $(T_3^{(2)}, R_3)$, and $(T_3^{(3)}, R_3)$ each require only one data report to satisfy data collection needs for both tasks. Ignoring the spatiotemporal overlap between tasks s_1 and s_2 and independently assigning data collection for these three spatiotemporal units to different participants would not only generate duplicate data collection but also incur duplicate costs. Our approach minimizes inter-task data redundancy by maximizing the reuse of data collected in each spatiotemporal unit across multiple tasks during task allocation.

2 Task Participant Trajectory Prediction Model

To allocate suitable participants to tasks, we must predict participant mobility trajectories across different spatiotemporal units. Therefore, we design an LSTM-based participant mobility trajectory sequence prediction model. Rather than directly using statistical models based on historical records to derive par-

ticipant visitation states for specific spatiotemporal units, our LSTM approach enables more accurate trajectory prediction.

As shown in Figure 2, the prediction model comprises encoder and decoder components. The encoder processes input data while the decoder handles output data, with each component containing L LSTM units. The visit probability matrix represents the probability of participant w_j visiting each partition in each sub-slot of time slot T_k during historical statistical periods, denoted as probability matrix P_j^k. Element $P_j^k(\tau, g)$ indicates the probability of participant w_j visiting partition R_g in sub-slot $T_k^{(\tau)}$. For example, in past seven-day historical records, if a participant visited partition R_1 twice, partition R_2 once, and partition R_3 four times in sub-slot $T_k^{(1)}$, then $P_j^k(1,1) = 2/7$, $P_j^k(1,2) = 1/7$, and $P_j^k(1,3) = 4/7$.

If we denote the first row of matrix P_j^k as $P_j^k(1,:)$, the τ-th row as $P_j^k(\tau,:)$, and the L-th row as $P_j^k(L,:)$, then the input for L time steps can be represented as $I_N = \{P_j^k(1,:), ..., P_j^k(\tau,:), ..., P_j^k(L,:)\}$. The L elements correspond to inputs for L units in the encoder. For representation convenience, we denote the input for the τ-th unit as time step τ input. To accelerate convergence without significantly degrading performance, we first embed each input into a d-dimensional embedding vector through an embedding layer. The embedding vector for time step τ input $P_j^k(\tau,:)$ is e_τ, as shown in equation (2). We then use this embedding vector as input for the unit at this time step to obtain the hidden state h_τ at time step τ, as shown in equation (3).

Equation (2) uses W_{enc} as the embedding weight with ReLU activation, while equation (3) shows h_τ as the LSTM function of current input embedding e_τ and previous hidden state $h_{\tau-1}$, with W_{enc} as learnable parameters shared across all input time steps. Specifically, the current time step' s hidden state derives from the previous hidden state and current input.

The context vector formed by input time steps 1 through L is represented as $H = f_{context}(h_1, h_2, ..., h_L)$, comprising implicit features from all input time steps. During the prediction output stage, the hidden state h'_τ at the τ-th output unit is computed as $h'_\tau = f_{LSTM}(y_{\tau-1}, h'_{\tau-1}, H; W_{dec})$, where $y_{\tau-1}$ is the previous output, $h'_{\tau-1}$ is the previous hidden state, H is the context vector from all input time steps, and W_{dec} represents learnable weight parameters in the decoder.

The output at time step τ is $y_\tau = \text{softmax}(h'_\tau)$. From this, we derive the predicted trajectory sequence for participant w_j in time slot T_k as $Tr_{raw}^{pre,j,k} = \{R_{w_j,T_k}^{(1)}, R_{w_j,T_k}^{(2)}, ..., R_{w_j,T_k}^{(\tau)}, ..., R_{w_j,T_k}^{(L)}\}$, where $R_{w_j,T_k}^{(\tau)}$ represents the partition predicted for participant w_j in sub-slot $T_k^{(\tau)}$.

3.1 Task Allocation Optimization Model

We define a visit indicator $Y_{w_j}^{(\tau)}(T_k, R_g)$ for participant w_j regarding spatiotemporal unit $(T_k^{(\tau)}, R_g)$. If the model predicts that participant w_j will visit partition R_g in sub-slot $T_k^{(\tau)}$, then $Y_{w_j}^{(\tau)}(T_k, R_g) = 1$; otherwise, it is 0. Similarly, for each task, we use binary decision variable $X_{i,j}$ to indicate whether task s_i is assigned to participant w_j.

Since participants frequently revisit historically accessed locations, we analyze features affecting prediction accuracy. We define a spatiotemporally relevant visit probability matrix as input for the prediction model, corresponding to $P_j^k(1, :)$ through $P_j^k(L, :)$ in the encoder portion of Figure 2.

The expected number of data samples collected by task s_i in spatiotemporal unit $(T_{p_{s_i}}^{(\tau)}, R_{q_{s_i}})$ is calculated as shown in equation (8), and the total number of data samples collected by task s_i in time-partition pair $(T_{p_{s_i}}, R_{q_{s_i}})$ is calculated as shown in equation (9), where $C(\cdot)$ denotes a counting function.

For individual tasks, we define the data redundancy of a time-partition pair as the ratio of collected data quantity to covered spatiotemporal units, calculated as:

$$dr_{s_i}(T_{p_{s_i}}, R_{q_{s_i}}) = \frac{C_{s_i}(T_{p_{s_i}}, R_{q_{s_i}})}{\sum_{\tau=1}^{L} cov_{s_i}(T_{p_{s_i}}^{(\tau)}, R_{q_{s_i}})}$$

The calculation premise is that the time-partition pair has already met the task' s minimum sensing requirement. The coverage indicator $cov_{s_i}(T_{p_{s_i}}^{(\tau)}, R_{q_{s_i}})$ in equation (10) is computed as:

$$cov_{s_i}(T_{p_{s_i}}^{(\tau)}, R_{q_{s_i}}) = \begin{cases} 1 & \text{if } C_{s_i}(T_{p_{s_i}}^{(\tau)}, R_{q_{s_i}}) \geq 1 \\ 0 & \text{otherwise} \end{cases}$$

Multiple tasks in the system may exhibit spatiotemporal coverage overlap, meaning multiple tasks contain identical time-partition pairs. To reduce data redundancy caused by such overlap, we maximize the reuse of data sensed by participant w_j in the same spatiotemporal unit across multiple tasks. The reuse degree is defined as the number of tasks in which the participant participates in this spatiotemporal unit, calculated as:

$$overlap_{w_j}(T_k^{(\tau)}, R_g) = \sum_{i=1}^{M} X_{i,j} \cdot Y_{w_j}^{(\tau)}(T_k, R_g)$$

The compensation for participant w_j in a single spatiotemporal unit $(T_k^{(\tau)}, R_g)$ is defined as the sum of fixed unit compensation and discounted compensation for overlapping tasks, with total compensation determined by data reuse degree:

$$pay_{w_j}(T_k^{(\tau)}, R_g) = u \cdot [1 + (\gamma - 1) \cdot overlap_{w_j}(T_k^{(\tau)}, R_g)]$$

where u is unit compensation and γ is a discount factor parameter ($0 \leq \gamma \leq 1$). Participants receive higher compensation when their data is utilized by more tasks. The compensation paid by task s_i to participant w_j in time-partition pair $(T_{p_{s_i}}, R_{q_{s_i}})$ is:

$$pay_{w_j}^{s_i}(T_{p_{s_i}}^{(\tau)}, R_{q_{s_i}}) = \frac{pay_{w_j}(T_{p_{s_i}}^{(\tau)}, R_{q_{s_i}})}{overlap_{w_j}(T_{p_{s_i}}^{(\tau)}, R_{q_{s_i}})}$$

The total compensation expenditure for task s_i across all time-partition pairs is:

$$pay_{s_i} = \sum_{p_{s_i}=1}^{|\mathcal{T}_{s_i}|} \sum_{q_{s_i}=1}^{|\mathcal{R}_{s_i}|} \sum_{\tau=1}^{L} \sum_{j=1}^{N} pay_{w_j}^{s_i}(T_{p_{s_i}}^{(\tau)}, R_{q_{s_i}})$$

Our first optimization objective minimizes intra-task data redundancy by evenly distributing collected data across spatiotemporal units in each time-partition pair. The second objective maximizes data reuse for each participant in the same spatiotemporal unit, thereby reducing inter-task redundancy. The optimization objectives are:

Objective 1 (Minimize intra-task redundancy):

$$\min \sum_{i=1}^{M} \sum_{p_{s_i}=1}^{|\mathcal{T}_{s_i}|} \sum_{q_{s_i}=1}^{|\mathcal{R}_{s_i}|} dr_{s_i}(T_{p_{s_i}}, R_{q_{s_i}})$$

Objective 2 (Maximize inter-task reuse):

$$\max \sum_{j=1}^{N} \sum_{k=1}^{K} \sum_{\tau=1}^{L} \sum_{g=1}^{G} overlap_{w_j}(T_k^{(\tau)}, R_g)$$

Subject to: 1. Budget constraint: $pay_{s_i} \leq B_{s_i}$ for each task s_i 2. Quality constraint: $C_{s_i}(T_{p_{s_i}}, R_{q_{s_i}}) \geq L$ for each time-partition pair 3. Participant capacity: $\sum_{i=1}^{M} \sum_{p_{s_i}=1}^{|\mathcal{T}_{s_i}|} \sum_{q_{s_i}=1}^{|\mathcal{R}_{s_i}|} \sum_{\tau=1}^{L} X_{i,j} \cdot Y_{w_j}^{(\tau)}(T_{p_{s_i}}, R_{q_{s_i}}) \leq \omega$ for each participant w_j

3.2 Task Allocation Optimization Model Solution

The problem we address is the optimization problem shown in equation (16). We employ a genetic algorithm due to its fast execution and strong applicability. However, considering the fine-grained spatiotemporal units, our problem has a large solution space where individuals in the initial population may be far from optimal solutions. To achieve good results at low cost, we propose a hybrid genetic algorithm that combines greedy algorithms with genetic algorithms. Below we detail our proposed method.

a) Allocation Matrix Representation. Our problem involves task allocation for redundancy reduction, represented using a matrix structure of binary decision variables. Matrix rows and columns correspond to M tasks and N participants respectively, forming an $M \times N$ allocation matrix X where element $X_{i,j} \in \{0, 1\}$. When element $X_{i,j} = 1$, task s_i is assigned to participant w_j; otherwise, it is not assigned. Figure 3 illustrates an example allocation matrix.

b) Population Initialization. The initial population is a set of chromosomes at the search beginning, significantly affecting algorithm performance. Randomly generated chromosomes may not always satisfy problem constraints. To ensure population diversity, we introduce a greedy operator to improve individuals that violate constraints, as shown in Algorithm 1.

c) Fitness Function. Our goal is to minimize data redundancy in task allocation—lower redundancy yields better fitness. Therefore, individual X_k' s fitness relates to data redundancy:

$$fitness(X_k) = \frac{1}{DR(X_k)}$$

where $DR(X_k)$ represents the data redundancy produced by allocation scheme X_k, comprising intra-task and inter-task redundancy:

$$DR(X_k) = DR_1(X_k) + DR_2(X_k)$$

The intra-task redundancy $DR_1(X_k)$ and inter-task redundancy $DR_2(X_k)$ are calculated as:

$$DR_1(X_k) = \sum_{i=1}^{M} \sum_{p_{s_i}=1}^{|\mathcal{T}_{s_i}|} \sum_{q_{s_i}=1}^{|\mathcal{R}_{s_i}|} dr_{s_i}(T_{p_{s_i}}, R_{q_{s_i}})$$

$$DR_2(X_k) = \sum_{k=1}^{K} \sum_{\tau=1}^{L} \sum_{g=1}^{G} \max_{w_j \in \mathcal{W}} overlap_{w_j}(T_k^{(\tau)}, R_g)$$

d) Selection. The selection operator passes higher-fitness individuals to the next generation while eliminating lower-fitness ones. Since low-fitness individuals may still contain good genes, we use roulette wheel selection to determine which individuals to preserve.

e) Crossover. We perform partially matched crossover operations row-wise on the matrix. We randomly select two individuals as parents, set crossover points, and exchange rows at these points to generate two new individuals. If new individuals violate participant capacity constraints, we reset crossover points until equation (17-3) is satisfied.

f) Mutation. To avoid local optima and accelerate convergence, we randomly select mutation elements in the matrix, flipping "1" to "0" and vice versa, while

verifying that mutated individuals satisfy minimum task requirements (equation (17-2)).

g) Termination. The algorithm terminates when iteration count reaches the maximum evolutionary generation.

Algorithm 2 presents our task allocation method execution process.

4 Experimental Analysis

For MCS task allocation, this paper comprehensively considers task spatiotemporal attributes and sensing quality requirements, participants' future trajectory sequences, and other factors. We design a task allocation algorithm based on genetic algorithms to minimize intra-task data redundancy and maximize inter-task sensing data reuse, thereby reducing sensing costs. To evaluate our method, we first compare prediction accuracy in Python, then verify task allocation algorithm efficiency. Table 1 shows main parameter settings.

4.1 Comparison Algorithms

To evaluate our algorithm, we compare against two baseline algorithms (MTPS and CAPR) under varying task quantities.

Fine-grained Multi-task Allocation (MTPS): This centralized algorithm optimizes allocation through an iterative greedy process based on utility functions under total budget constraints to maximize sensing quality. MTPS designs reasonable incentive functions and performs fine-grained period partitioning, representing a typical method for controlling intra-task data redundancy.

Conflict-Aware Participant Recruitment (CAPR): This method considers correlations between tasks and participants (including positive and negative correlations) for participant recruitment, proposing a three-stage heuristic mechanism to reduce participant costs and maximize platform utility. Although CAPR does not directly control inter-task redundancy, its correlation-based approach can indirectly reduce inter-task redundancy.

For CAPR' s correlation function between any two tasks s_i and s_i', we measure spatiotemporal coverage overlap rate as shown in equation (23). Since we ignore participant reputation factors, we set reputation values to constant 1 in CAPR.

4.2 Evaluation Metrics

To verify effectiveness, we first analyze prediction accuracy. Then we evaluate the task allocation algorithm using three metrics: task execution rate, data redundancy rate, and sensing cost, analyzing how task quantity variations affect these indicators.

a) Task Execution Rate: Defined as the ratio of successfully allocated tasks (meeting minimum sensing quality) to total system tasks, ranging from 0 to 1.

b) Data Redundancy Rate: Defined as the difference between total collected data and effective data (spatiotemporal units covered by collected data), with values in [0,1].

c) Average Sensing Cost: Defined as the ratio of total platform compensation to allocated tasks, representing average cost per task.

4.3.1 Prediction Method Evaluation

To evaluate our trajectory prediction method, we compare against statistical analysis-based prediction, Markov model-based methods, and short-term history LSTM methods. Literature [20] uses statistical models to derive participant passage probabilities, [23] employs semi-Markov processes, and [25] uses recent historical trajectory sequences as input. In experiments, we divide a day into $K = 12$ time slots and analyze accuracy for different subdivisions $L = 2, 4, 6, 12$.

Figure 4 shows prediction accuracy across different L values. Our method outperforms the other three approaches. Markov and statistical methods show lower accuracy, demonstrating LSTM' s self-learning capability improves prediction. The short-term LSTM method is slightly less accurate than ours because we incorporate long-term mobility probability information for better generalization. Our method achieves highest accuracy (~86.4%) at $L = 4$, with all methods decreasing as L increases because smaller prediction intervals increase trajectory variability and reduce pattern clarity.

4.3.2 Task Allocation Method Evaluation

To demonstrate task quantity impact, we fix participant count at 15 with maximum participation limit of 30 spatiotemporal units. We first compare convergence, then evaluate our algorithm against CAPR and MTPS across task execution rate, redundancy rate, and average cost, repeating each experiment 10 times and averaging results.

Table 2 shows iteration counts. To illustrate convergence performance, we compare our algorithm with an uninitialized version. As task quantity increases, both algorithms require more iterations, but the uninitialized version needs significantly more, demonstrating that our initialization algorithm (Algorithm 1) accelerates solution speed.

Figure 5 shows task quantity impact on execution rate. Our algorithm achieves higher execution rates than CAPR and MTPS. MTPS performs poorly because while it reduces intra-task redundancy through periodic allocation, it ignores inter-task spatiotemporal overlap, causing excessive duplicate assignments and high costs that prevent selecting enough participants to meet minimum thresholds. All three methods show decreasing execution rates as task quantity increases because participants reach their spatiotemporal unit limits. Our algorithm and CAPR perform similarly when task count is below 50 because low inter-task overlap reduces our reuse maximization advantage. Above 50 tasks,

our algorithm outperforms CAPR as increased overlap enables better data reuse in overlapping regions. CAPR' s pre-allocation at task start times may become suboptimal for subsequent tasks due to negative correlations or participant unavailability.

Figure 6 shows task quantity impact on redundancy rate. Our algorithm achieves lower redundancy than CAPR and MTPS because we simultaneously control intra-task redundancy through fine-grained partitioning and maximize data reuse across tasks, while MTPS only addresses intra-task redundancy and CAPR only partially reduces inter-task redundancy through correlation. As task quantity increases, our algorithm and CAPR show slowly increasing redundancy rates while MTPS increases rapidly (reaching 30% vs our 22% at high task counts) due to unchecked inter-task overlap. CAPR' s redundancy increases more slowly than ours, possibly because it already generates more intra-task redundancy at low task counts, and increased task overlap doesn' t significantly worsen its performance.

Figure 7 shows task quantity impact on average cost. Below 40 tasks, all three methods have similar costs. As task quantity increases, our algorithm and CAPR costs decrease because greater spatiotemporal overlap enables higher data reuse and lower compensation payments. CAPR achieves slightly lower costs than ours at low task counts due to higher execution rates. MTPS costs remain stable because without inter-task redundancy control, budget constraints prevent completing more tasks as task quantity increases.

5 Conclusion

Spatiotemporal coverage overlap among multiple tasks in mobile crowd sensing systems may cause duplicate data collection and redundancy problems. This paper proposes a task allocation method that reduces both intra-task and inter-task data redundancy. We first design a participant mobility trajectory prediction method, then propose a genetic algorithm-based task allocation approach considering temporal and spatial overlap among tasks. Simulation results demonstrate the proposed method effectively reduces data redundancy within and between tasks. Future work should consider additional factors affecting participant behavior and availability, and explore new optimization methods and theoretical foundations.

References

[1] Hettiachchi D, Kostakos V, Goncalves J. A survey on task assignment in mobile crowdsensing with clustering effect [J]. ACM Computing Surveys, 2022, 55 (3): 1-35.

[2] Seid S, Zennaro M, Libse M, et al. Mobile crowdsensing based road surface monitoring using smartphone vibration sensor and lorawan [C]// Proc of the 1st Workshop on Experiences with the Design and Implementation of Frugal

Smart Objects. New York: ACM Press, 2020: 13-18.

[3] Bock F, Martino S D, Origlia A. Smart parking: using a crowd of taxis to sense on-street parking space availability [J]. IEEE Trans on Intelligent Transportation Systems, 2020, 21 (2): 496–508.

[4] Fang Wenfeng, Zhou Zhaorong, Sun Sanshan. Research on task assignment for mobile crowd sensing [J]. Application Research of Computers, 2018, 35 (11): 3206-3212.

[5] Ko H, Pack S, Leung V. Coverage-guaranteed and energy-efficient participant selection strategy in mobile crowdsensing [J]. IEEE Internet of Things Journal, 2019, 6 (2): 3202-3211.

[6] Yang Jing, Fu Lei, Yang Boran, et al. Participant service quality aware data collecting mechanism with high coverage for mobile crowdsensing [J]. IEEE Access, 2020 (8): 10628-10639.

[7] Xiao Mingjun, Gao Guoju, Wu Jie, et al. Privacy-preserving user recruitment protocol for mobile crowdsensing [J]. IEEE/ACM Trans on Networking, 2020, 28 (2): 519-532.

[8] Hu Qin, Wang Shengling, Cheng Xiuzhen, et al. Cost-efficient mobile crowdsensing with spatial-temporal awareness [J]. IEEE Trans on Mobile Computing, 2021, 20 (3): 928-938.

[9] Song Shiwei, Liu Zhidan, Li Zhenjiang, et al. Coverage-oriented task assignment for mobile crowdsensing [J]. IEEE Internet of Things Journal, 2020, 7 (8): 7407-7418.

[10] Zhou Siwang, He Yan, Xiang Shuzhen, et al. Region-based compressive networked storage with lazy encodings [J]. IEEE Trans on Parallel and Distributed Systems, 2019, 30 (6): 1390–1402.

[11] Liu Wenbin, Wang Leye, Wang En, et al. Reinforcement learning-based cell selection in sparse mobile crowdsensing [J]. Computer Networks, 2019, 161 (9): 102–114.

[12] Liu Wenbin, Yang Yongjian, Wang En, et al. User recruitment for enhancing data inference accuracy in sparse mobile crowdsensing [J]. IEEE Internet of Things Journal, 2020, 7 (3): 1802-1814.

[13] Liu Wenbin, Yang Yongjian, Wang En, et al. Prediction based user selection in time-sensitive mobile crowdsensing [C]// Proc of the 14th Annual IEEE International Conference on Sensing, Communication, and Networking. Piscataway, NJ: IEEE Press, 2017: 1-9.

[14] Tao Xi, Song Wei. Location-dependent task allocation for mobile crowdsourcing [J]. IEEE Internet of Things Journal, 2019, 6 (1): 1029-1045.

[15] Wang Jiangtao, Wang Yasha, Zhang Daqing, et al. Fine-grained multitask allocation for participatory sensing with a shared budget [J]. IEEE Internet of

Things Journal, 2016, 3 (6): 1395-1405.

[16] Zhou Tongqing, Xiao Bin, Cai Zhiping, et al. A utility model for photo selection in mobile crowdsensing [J]. IEEE Trans on Mobile Computing, 2021, 20 (1): 48-62.

[17] Gendy M, Al-Kabbany A, Badran E. Maximizing clearance rate of budget-constrained auctions in participatory mobile crowdsensing [J]. IEEE Access, 2020 (8): 113585-113600.

[18] Xia Xingyou, Zhou Yan, Li Jie, et al. Quality-aware sparse data collection in MEC-enhanced mobile crowdsensing systems [J]. IEEE Trans on Computational Social Systems, 2019, 6 (5): 1051-1062.

[19] Nguyen T N, Zeadally S. Mobile crowd-sensing applications: data redundancies, challenges, and solutions [J]. ACM Trans on Internet Technology, 2022, 22 (2), 1-15.

[20] Wang Liang, Yu Zhiwen, Zhang Daqing, et al. Heterogeneous multi-task assignment in mobile crowdsensing using spatiotemporal correlation [J]. IEEE Trans on Mobile Computing, 2019, 18 (1): 84-97.

[21] Zhang Lichen, Ding Yu, Wang Xiaoming, et al. Conflict-aware participant recruitment for mobile crowdsensing [J]. IEEE Trans on Computational Social Systems, 2020, 7 (1): 192-204.

[22] Yang Wenjie, Sun Guodong, Ding Xingjian, et al. Budget-feasible user recruitment in mobile crowdsensing with user mobility prediction [C]// Proc of the 37th International Performance Computing and Communications Conference. Piscataway, NJ: IEEE Press, 2018: 1-10.

[23] Yang Yongjian, Liu Wenbin, Wang En, et al. A prediction-based user selection framework for heterogeneous mobile crowdsensing [J]. IEEE Trans on Mobile Computing, 2019, 18 (11): 2460-2473.

[24] Wang Jiangtao, Wang Feng, Wang Yasha, et al. HyTasker: hybrid task allocation in mobile crowd sensing [J]. IEEE Trans on Mobile Computing, 2020, 19 (3): 598-611.

[25] Zhu Xiaoyu, Luo Yueyi, Liu Anfeng, et al. A deep learning-based mobile crowdsensing scheme by predicting vehicle mobility [J]. IEEE Trans on Intelligent Transportation Systems, 2021, 22(7): 4648-4659.

Note: Figure translations are in progress. See original paper for figures.