applied sciences

MDPI

Article

Isolation-Sensitive Online Task Assignment in Spatial Crowdsourcing with Adaptive Regional Coarsening

Fanyu Meng [1], Xinyu Gao [2] and Yajie Wang [1,*]

[1] School of Innovation and Entrepreneurship, Shenyang Aerospace University, Shenyang 110136, China; mengfanyu@sau.edu.cn
[2] School of Computer Science and Engineering, Shenyang Jianzhu University, Shenyang 110168, China; gaoxinyu4242@gmail.com
* Correspondence: wangyajie@sau.edu.cn

Abstract

Public health emergencies require spatial crowdsourcing platforms to finish urgent tasks while limiting unnecessary movement across regions. Most online task assignment studies focus on profit, travel distance, latency, task coverage, or service quality. However, isolation sensitive scenarios need a different assignment goal. In such scenarios, regional crossings should be directly controlled during worker–task matching. This paper studies an isolation sensitive online task assignment problem in spatial crowdsourcing. The service space is modeled as a regional adjacency graph. The matching objective combines cross-region movement cost, an urgency reward for delayed task completion, and a dummy no-assignment cost for carry-over decisions. To handle dynamic arrivals, a time-sliced online process is used. Unfinished tasks are carried over to later time slots, and the priority of each carried-over task increases with waiting time. Based on this framework, we design two algorithms. OnlineKM serves as the basic priority-aware online matching algorithm. OnlineKM builds a matching problem in each time slot and applies KM-based partial matching with the information currently available. OnlineARC further uses δ-balanced adaptive regional coarsening. OnlineARC merges adjacent regions according to recent supply–demand balance before matching. This step adjusts the regional granularity used for movement cost evaluation and helps keep assignments close to local regions when regional merging is suitable. Experiments are conducted using a real-world task locations dataset from a 2022 COVID-19-related scenario in Changchun, with simulated worker availability and online arrivals. The results show that the proposed methods usually reduce the combined assignment objective value under the tested settings. The service quality and movement control metrics show that OnlineARC reduces the cross-region assignment ratio and average hop distance while maintaining a high task completion rate under the representative setting. OnlineKM improves running efficiency through time-sliced matching, while OnlineARC provides a trade-off between adaptive coarsening cost and locality-aware movement cost evaluation. These results suggest that adaptive regional coarsening can serve as a practical heuristic for locality-aware online task assignment in isolation sensitive spatial crowdsourcing under suitable worker–task distributions.

Keywords: dynamic matching; isolation sensitive task assignment; online task assignment; regional coarsening; region-level movement control; spatial crowdsourcing

Check for updates

Academic Editor: Hyokyung Bahn

Received: 15 May 2026
Revised: 14 June 2026
Accepted: 15 June 2026
Published: 19 June 2026

Copyright: © 2026 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license.

1. Introduction

Public health emergencies often create urgent demand for controlled logistics services. Critical supplies must be delivered quickly, while unnecessary movement across regions should be limited. In this setting, a task assignment method should not only decide whether a worker can complete a task, but also whether the assignment causes avoidable regional crossings. This requirement is important for emergency logistics and other isolation-sensitive services.

Spatial crowdsourcing offers a suitable framework for this problem. In a spatial crowdsourcing system, task requesters publish location-based tasks, and workers report their locations and availability. The platform assigns tasks to workers under spatial and temporal constraints. Spatial crowdsourcing has been widely studied in mobile micro-task allocation, location-based query answering, ride hailing, food delivery, urban sensing, and emergency-related data collection [1–3]. Task assignment is a core problem in these systems because assignment decisions affect task completion, worker cost, platform efficiency, and service quality [4,5].

Previous studies have optimized many objectives, such as platform utility, task coverage, travel distance, reliability, fairness, and worker preference [6–10]. Route- and trajectory-aware studies also considered worker trajectories, pickup and delivery routes, and road network constraints [11–13]. These studies provide useful foundations for spatial crowdsourcing, but they focus on operational objectives that differ from the region-level movement control setting considered here.

This paper studies an isolation-sensitive online task assignment problem in spatial crowdsourcing. The service space is divided into regional nodes. A worker who moves from one region to another produces a hop-based cross-region cost on the regional adjacency graph. Unlike fine-grained distance-based assignment, the proposed problem treats regional crossings as the main operational cost. This design matches emergency scenarios, in which crossing a regional boundary may increase management costs, coordination burdens, or potential contact risk.

To provide an intuitive example, Figure 1 illustrates a simple isolation-sensitive task-matching instance. Workers and tasks are distributed across multiple regions. Each worker can complete at most two tasks. The platform first gives priority to feasible worker–task assignments within the same region. After local matching, the platform considers the remaining unmatched workers and tasks across regions.

Figure 1 shows that local matching can reduce unnecessary regional crossings. Cross-region matching is still allowed, but it is mainly used for the remaining unmatched workers and tasks. This static example becomes more difficult in an online setting. Workers and tasks arrive over time. The platform cannot observe future information when making current assignment decisions.

An offline assignment can be formulated as a weighted bipartite matching problem. Classical KM-based methods can solve this type of matching problem [14,15]. Delivery batching has also been modeled through matching formulations [16]. However, real spatial crowdsourcing platforms usually operate online. Workers and tasks may arrive in different time slots. Some tasks may not be assigned immediately. These unfinished tasks need to be carried over to later time slots. Existing online methods mainly optimized utility, latency, preference, reliability, or completion quality [17–19]. This motivates an online formulation that handles regional movement control and delayed task carryover within a single decision process.

Direct use of KM matching in each time slot can obtain a locally optimal assignment under the current information. However, this direct strategy has two limitations. First, the strategy does not fully use the regional structure of isolation-sensitive assignment.

Second, the platform needs to build and solve a per-slot worker–task matching instance. This process can be expensive when many workers and tasks are available. Static regional grouping can simplify regional organization. However, fixed groups may not match the changing supply–demand distribution. Therefore, the assignment algorithm should adjust the regional granularity according to recent matching patterns.

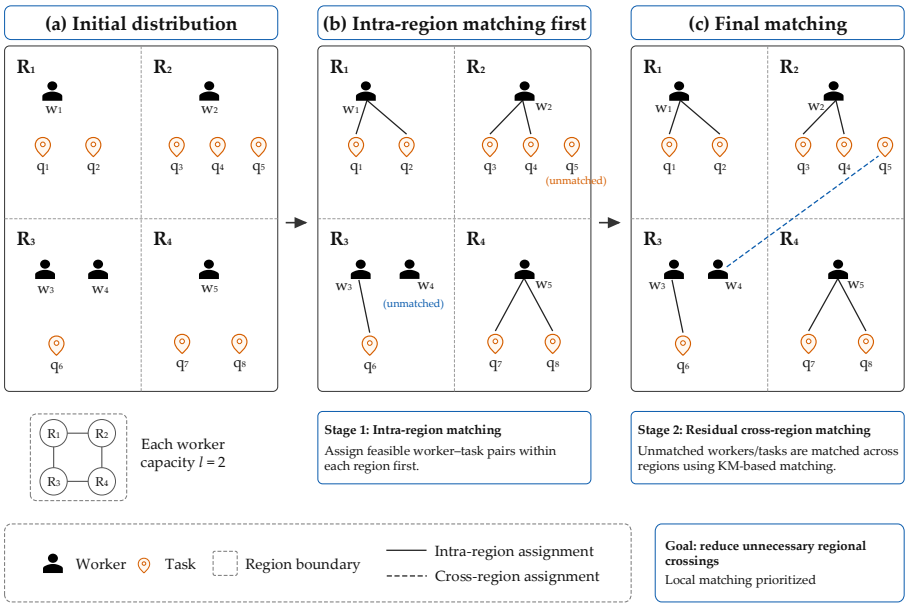


Figure 1. Illustration of an isolation-sensitive task matching instance. The example illustrates the preference for intra-region assignments and the possibility of residual cross-region matching. The blue dashed line in panel (**c**) highlights a residual cross-region assignment from worker w_4 to task q_5 after intra-region matching.

To address these issues, this paper formulates a region-crossing-cost-aware online assignment problem. The formulation includes worker capacity, task carry-over, carry-over urgency, and sequential decision making. Based on this formulation, we design a time-sliced online assignment framework. OnlineKM is proposed as the basic priority-aware online matching algorithm. OnlineKM constructs a local matching problem for each time slot using available workers, newly arriving tasks, and carried-over tasks. OnlineKM then applies KM-based partial matching under the current information. To further use the regional structure, we design OnlineARC as an adaptive extension of OnlineKM. OnlineARC updates the regional graph according to recent supply–demand imbalance before matching, so the movement–cost calculation can reflect short-term regional changes. This update helps keep assignments close to local regions when the recent worker–task distribution supports regional merging.

The main contributions of this paper are summarized as follows.

1. We define an isolation-sensitive online task assignment problem in spatial crowdsourcing. The problem includes regional nodes, worker capacity, online task arrivals, task carry-over, carry-over urgency, and sequential decision-making.
2. We introduce a unified assignment objective. The objective combines cross-region movement cost, an urgency reward for delayed task completion, and a dummy no-assignment cost for carry-over decisions. This design clarifies the trade-off between reducing regional crossings and avoiding excessive task delay.
3. We propose OnlineKM as a basic priority-aware online matching algorithm. OnlineKM performs KM-based partial matching in each time slot. OnlineKM also updates unfinished tasks through a carry-over priority mechanism.

4. We propose OnlineARC as a δ-balanced adaptive regional coarsening algorithm for online task assignment. OnlineARC merges adjacent regions according to the recent supply–demand balance before KM-based partial matching. This design adaptively updates the regional graph and can improve assignment locality when recent worker–task distributions support regional merging.
5. We evaluate the proposed algorithms using a real-world task location dataset from the 2022 COVID-19-related scenario in Changchun. We also use simulated worker sets and online arrival processes in the experiments. The experiments compare the combined assignment objective value, running time, and service quality and movement control metrics under different parameter settings and the representative multi-seed setting.

2. Related Work

2.1. Spatial Crowdsourcing Task Assignment

Spatial crowdsourcing has been widely studied as a paradigm for assigning location-dependent tasks to distributed mobile workers. Early studies, such as GeoCrowd, built an early spatial crowdsourcing framework for answering spatial queries with mobile workers [3]. Deng et al. studied worker-selected task assignment [1]. Tong et al. and Gummidi et al. reviewed spatial crowdsourcing models, assignment modes, optimization objectives, and main challenges [4].

Task assignment in spatial crowdsourcing usually includes two modes. The first mode is the worker-selected-task mode. In this mode, workers choose tasks according to personal preferences or constraints [9,10]. The second mode is the server-assigned-task mode. In this mode, the platform assigns tasks to workers according to system-level objectives [19–21]. This paper focuses on the server-assigned-task mode because isolation-sensitive emergency logistics requires centralized coordination.

Previous studies optimized many assignment objectives; some considered worker preference, destination awareness, fairness, reliability, and task quality [22–24], while others considered coalition formation, group task allocation, social impact, and dependency-aware assignment [25–28]. These studies provide useful foundations for worker–task matching in spatial crowdsourcing. This paper uses these foundations in a different region-level movement control setting.

2.2. Online and Real-Time Task Assignment

Online task assignment is more difficult than offline task assignment. In an online setting, future workers and tasks are unknown when the platform makes current decisions. The platform must assign tasks according to current information. Previous decisions are also difficult to revise. For this reason, real-time online matching has become an important topic in spatial crowdsourcing.

Cheng et al. studied real-time cross-online matching in spatial crowdsourcing [17]. In that problem, workers and tasks arrived dynamically, and the platform matched them under real-time constraints. Tong et al. studied online mobile micro-task allocation [29]. Zeng et al. studied latency-oriented task completion in spatial crowdsourcing [30]. Zhao et al. studied predictive task assignment through a data-driven method [10]. Wang et al. introduced worker churn prediction to improve assignment robustness under worker uncertainty [19]. Other studies examined continuous task collection and pickup and delivery scheduling, such as online trichromatic pickup and delivery scheduling and time-continuous spatial crowdsourcing [13,31].

The above studies improved online spatial crowdsourcing from different aspects, including real-time matching, latency control, prediction, worker uncertainty, and pickup

and delivery scheduling. They provide the online decision background for the time-sliced formulation used in this paper.

2.3. Locality-Aware and Route-Aware Assignment

Spatial crowdsourcing tasks depend on location. Therefore, many studies considered spatial locality, road networks, trajectories, and routes in assignment models. Road network-aware and multi-location task assignment methods considered realistic movement constraints and privacy requirements [32]. Trajectory-aware task coalition assignment used worker trajectories in coalition formation [11]. Destination-aware task assignment decomposed workers according to destinations and improved matching efficiency [20]. Proximity-aware assignment considered the spatial distance between workers and tasks [33].

Route-aware and delivery-oriented studies also provided related ideas. FoodMatch transformed food delivery batching into a matching problem in dynamic road networks [16]. Safe route planning considered traveler safety and privacy during movement [12]. Courier route prediction modeled the couriers' spatial–temporal behaviors [34]. Online pickup and delivery scheduling studied assignment decisions under dynamic arrivals [13].

These studies showed that spatial structure and movement patterns were important for task assignment. However, route-aware optimization is not the same as isolation-sensitive assignment. Most route-aware methods optimized travel distance, travel time, route utility, or delivery efficiency, rather than region-level movement control under online supply–demand imbalance. These studies are complementary to the region-level movement control model considered in this paper.

2.4. Risk-Aware and Isolation-Sensitive Mobility Optimization

Public-health-oriented task assignment requires different objectives from those of ordinary commercial crowdsourcing. In epidemic prevention, emergency logistics, and controlled service scenarios, cross-region movement may increase management cost and potential contact risk. Therefore, a task assignment method should reduce unnecessary regional crossings.

Some studies examined safety-aware or privacy-aware mobility. Safe route planning considered traveler safety and privacy during movement [12]. Privacy-preserving spatial task assignment protected worker locations through uncertainty or cloaked locations [35]. Road network-aware and privacy-preserving assignment methods also considered movement constraints and personalized privacy requirements [32]. These studies showed that spatial crowdsourcing could support safety-aware decision-making.

These studies motivate the region-level movement control view adopted in this study, but the present work focuses on online worker–task assignment rather than route selection or privacy protection.

2.5. Research Gap and Paper Positioning

In summary, previous spatial crowdsourcing studies provide useful foundations for preference-aware, reliability-aware, route-aware, and online task assignment [2,6,11,27]. However, isolation-sensitive emergency scenarios require three elements in one online assignment framework: region-level movement control, online worker–task arrivals, and increasing urgency for carried-over tasks.

This paper focuses on this combined setting. OnlineKM uses a priority-aware time-sliced matching process with explicit task carry-over. OnlineARC extends OnlineKM through adaptive regional coarsening. OnlineARC updates the regional graph used for movement cost evaluation. Therefore, the proposed method is a graph-based combinatorial optimization heuristic for isolation-sensitive online assignment. The proposed method is not a machine-learning model comparison.

3. Problem Formulation

3.1. Regional Graph Model

We consider a spatial crowdsourcing platform that assigns spatial tasks to workers in an isolation-sensitive setting. The service space is divided into a set of regions. These regions can be administrative areas, communities, or grid cells. We use a regional adjacency graph $G = (V, E)$ to describe the regional structure. In this graph, V denotes the set of regional nodes, and E denotes the set of adjacency relationships among regions [28].

Each regional node $v \in V$ is represented as $v = \langle id, v.Q, v.W \rangle$, where id is the region index, $v.Q$ is the set of tasks in the region, and $v.W$ is the set of workers in the region. Each edge $e \in E$ is represented as $e = \langle (v_i, v_j), e.weight \rangle$, where $(v_i | v_j)$ indicates that two regions are adjacent, and $e.weight$ denotes the hop distance between them.

Directly adjacent regions have edge weight 1. We use $d_G(v_i, v_j)$ to denote the shortest path hop distance between regions v_i and v_j. In this paper, $d_G(v_i, v_j)$ measures cross-region movement cost. If a worker and a task are in the same region, the corresponding movement cost is 0. This graph-based cost focuses on regional crossings rather than fine-grained travel distance.

3.2. Online Time-Slot Model

We divide the operating period into equal-length time slots. The time slot sequence is denoted as $T = \{t_1, t_2, \ldots, t_s\}$. At each time slot t_k, the platform observes only current information. The current information includes available workers, newly arriving tasks, and unfinished tasks carried over from previous time slots. The platform cannot observe future workers or future tasks when the current assignment decision is made.

Each worker $w_i \in W$ is represented as $w_i = \langle id_i, v(w_i), t(w_i) \rangle$, where id_i is the worker identifier. The term $v(w_i) \in V$ denotes the worker's region. The term $t(w_i) \in T$ denotes the time slot in which the worker w_i is available.

Each task $q_j \in Q$ is represented as $q_j = \langle id_j, v(q_j), t(q_j), p_j \rangle$, where id_j is the task identifier. The term $v(q_j) \in V$ denotes the task region. The term $t(q_j) \in T$ denotes the task arrival time slot. The term p_j denotes the task priority.

If task q_j is not assigned in the current time slot, the platform carries out the task q_j over to the next time slot. The platform updates the task priority as $p_j \leftarrow p_j + 1$. This update rule describes the increasing urgency of delayed tasks.

3.3. Capacity Constraint

Each worker can complete at most l tasks in one time slot. The parameter l denotes the worker capacity. We use x_{ijk} to denote the assignment decision between worker w_i and task q_j in time slot t_k. If task q_j is assigned to worker w_i in time slot t_k, then $x_{ijk} = 1$. Otherwise, $x_{ijk} = 0$. For every worker $w_i \in W_k$, the capacity constraint is $\sum_{q_j \in Q_k} x_{ijk} \leq l$, where Q_k denotes the task set considered in time slot t_k.

Each task can be assigned to at most one real worker in one time slot. To model the carry-over decision explicitly, we introduce a binary variable y_{jk}. If task q_j is carried over from time slot t_k to t_{k+1}, then $y_{jk} = 1$. Otherwise, $y_{jk} = 0$. The task assignment and carry-over constraint is defined as follows:

$$\sum_{w_i \in W_k} x_{ijk} + y_{jk} = 1, \quad \forall q_j \in Q_k \tag{1}$$

This constraint ensures that each task has exactly one outcome within a time slot. The task is either assigned to one real worker or carried over to the next time slot. Therefore, the model supports at most one assignment for real worker–task pairs. The model does not force a complete real-worker assignment.

3.4. Carry-Over Constraint

Let Q_k^{new} denote the newly arriving tasks in time slot t_k. Let Q_k^{carry} denote the tasks carried over from previous time slots. The task set considered in time slot t_k is $Q_k = Q_k^{new} \cup Q_k^{carry}$. After matching in time slot t_k, the platform removes assigned tasks from the current task set. A task is carried over only when its carry-over variable equals (1). Therefore, the carry-over task set is defined as follows:

$$Q_{k+1}^{carry} = \left\{ q_j \in Q_k \mid y_{jk} = 1 \right\} \tag{2}$$

For each carried-over task, the platform updates the task priority as follows:

$$p_j^{k+1} = p_j^k + 1, \quad \forall q_j \in Q_{k+1}^{carry} \tag{3}$$

This carry-over rule is needed for two reasons. First, the current worker capacity may be insufficient. Second, assigning all tasks immediately may cause excessive cross-region movement. The binary variable y_{jk} makes the carry-over decision explicit in the mathematical formulation.

To illustrate the carry-over rule, Figure 2 presents an example over two consecutive time slots. Unfinished tasks are carried over with increased priority and are reconsidered together with newly arriving tasks.

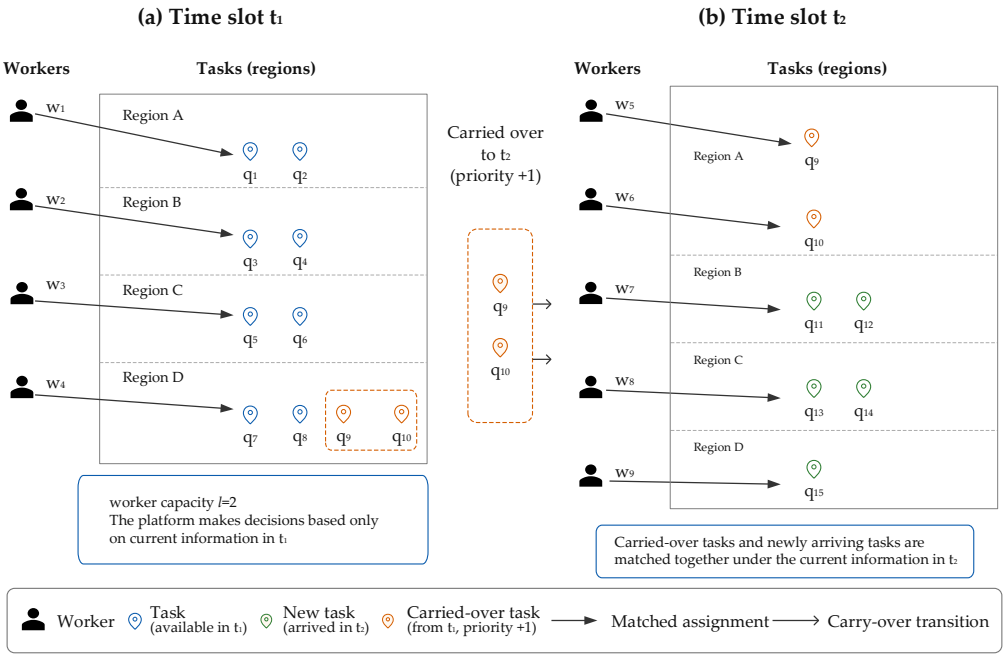


Figure 2. Example of online isolation-sensitive task matching over two consecutive time slots. Unfinished tasks in time slot t_1, highlighted by the orange dashed boxes as q_9 and q_{10}, are carried over to t_2 with increased priority and are matched together with newly arriving tasks under the current online information.

Figure 2 shows that the platform makes sequential decisions without future workers or task information. The carry-over priority mechanism prevents delayed tasks from being ignored in later matching.

3.5. Assignment Cost

For a worker–task pair (w_i, q_j) in time slot t_k, the cross-region movement cost is defined as $c_{ij} = d_G\left(v(w_i), v(q_j)\right)$. The current priority p_j^k represents the urgency accumulated

by task q_j after waiting. In the matching score, this priority is treated as an urgency reward rather than an additional penalty. A higher priority encourages the platform to complete delayed tasks earlier. A larger movement cost discourages the platform from assigning workers across distant regions.

The assignment score for a real worker–task edge is defined as $s_{ijk} = p_j^k - \lambda c_{ij}$. Equivalently, the real-assignment cost can be written as $a_{ijk} = \lambda c_{ij} - p_j^k$.

Thus, maximizing the assignment score is equivalent to minimizing the real-assignment cost. The no-assignment option and the carry-over decision are modeled in the global objective through the dummy carry-over variable.

The movement cost in this formulation is a region-level access cost. It measures the cost of assigning worker w_i from the worker's current region to task q_j's region. It is not a complete multi-stop route cost when the same worker receives multiple tasks within a single time slot.

3.6. Global Objective

The online isolation-sensitive assignment problem aims to construct a sequence of matching decisions $R = \{R_1, R_2, \ldots, R_s\}$. Here, R_k denotes the assignment set in time slot t_k. Each assignment is represented as $\langle q_j, w_i, t_k \rangle$. The objective is to minimize the cumulative assignment cost over all time slots. The assignment result must satisfy worker capacity, task assignment-or-carry-over, and binary decision constraints. The term θy_{jk} represents the dummy no-assignment cost when task q_j is carried over from time slot t_k to the next time slot. In this study, θ is set to 0, which is consistent with the dummy edge setting in the partial bipartite graph. The full optimization model is:

$$\begin{aligned} &\min \sum_{t_k \in T} [\sum_{w_i \in w_k} \textstyle\sum_{q_j \in Q_k} x_{ijk} \left(\lambda d_G \left(v(w_i), v(q_j) \right) - p_j^k \right) + \sum_{q_j \in Q_k} \theta y_{jk}], \\ &s.t. \quad \sum_{q_j \in Q_k} x_{ijk} \le l, \sum_{w_i \in W_k} x_{ijk} + y_{jk} = 1, \forall q_j \in Q_k, \forall t_k \in T, x_{ijk} \in \{0,1\}, y_{jk} \in \{0,1\}.' \end{aligned} \tag{4}$$

Equation (4) defines a combined assignment objective value. It is not a pure movement-cost metric and does not add task priority as a penalty. The first term penalizes cross-region movement for real worker–task assignments. The term $-p_j^k$ acts as an urgency reward for completing delayed tasks. The term θy_{jk} represents the dummy no-assignment cost when a task is carried over. In the experiments, θ is set to 0, and carry-over is mainly controlled through the dummy no-assignment option and the increasing priority of delayed tasks in later time slots. The movement weight parameter λ controls the relative importance of cross-region movement cost and urgency reward. A larger λ makes the assignment more conservative with respect to regional crossings, while a smaller λ gives more relative weight to completing urgent, delayed tasks. Because future information is unavailable, the online solution does not claim global optimality over the full time horizon. Instead, the online solution seeks high-quality sequential decisions. These decisions reduce cross-region movement and prevent excessive task delay.

3.7. Feasibility Condition

The total worker capacity should be sufficient for all tasks in the considered time horizon. This condition is written as $\sum_{t_k \in T} l \mid W_k \mid \ge \mid Q \mid$. This condition ensures that all tasks can eventually be completed. If this condition does not hold, some tasks cannot be completed. In that case, the problem becomes a partial assignment or task-rejection problem. This case is beyond the scope of this paper.

For the last time slot, unfinished tasks need an additional processing rule. Possible rules include horizon extension, virtual workers, or a terminal penalty. This pa-

per assumes that the time horizon and worker capacity are sufficient to meet the task completion requirement.

4. Online Framework and Algorithms

In the online isolation-sensitive task assignment problem, workers and tasks arrive dynamically. The platform cannot use future worker or task information when making current decisions. Therefore, we divide the operating period into time slots. In each time slot, the platform assigns available workers to newly arriving tasks and carried-over tasks.

We propose two algorithms in this framework. OnlineKM performs priority-aware KM-based partial matching at each time slot. OnlineARC extends OnlineKM through δ-balanced adaptive regional coarsening. OnlineARC uses recent supply–demand information to merge adjacent regions before matching. This step updates the regional graph used for movement cost evaluation and helps keep worker–task assignments close to local regions when regional merging is suitable. We use the KM algorithm as the basic bipartite matching component [36]. We also use the regional graph to measure cross-region movement cost. The proposed framework is related to an online spatial crowdsourcing assignment study [17,18,29] and movement-aware assignment studies [13,16,34].

4.1. Priority-Aware OnlineKM Algorithm

Algorithm 1 presents OnlineKM. OnlineKM follows a time-sliced matching process. At each time slot, OnlineKM first builds the available worker set. OnlineKM then combines newly arriving tasks with carried-over tasks. After this step, OnlineKM increases the priority of each carried-over task. Finally, OnlineKM builds a partial bipartite graph and applies KM-based partial matching under the current regional graph.

Algorithm 1: Priority-Aware Online Task Matching Algorithm OnlineKM

Input: Time slot sequence $T = \{t_1, t_2, \ldots, t_s\}$; task set Q; worker set W; regional graph $G(V, E)$; worker capacity l; Movement weight parameter λ.
Output: Online matching result $R(Q, W, G, T)$.

(1) $R \leftarrow \emptyset$; $Q^{carry} \leftarrow \emptyset$
(2) for each time slot $t_k \in T$ do
(3) $\quad W_k \leftarrow \{w \in W \mid w.t = t_k\}$
(4) $\quad Q_k^{new} \leftarrow \{q \in Q \mid q.t = t_k\}$
(5) $\quad Q_k \leftarrow Q_k^{new} \cup Q^{carry}$
(6) $\quad$ for each $q \in Q^{carry}$ do
(7) $\quad\quad q.p \leftarrow q.p + 1$
(8) $\quad$ end for
(9) $\quad B_k \leftarrow$ BuildPartialBipartiteGraph(W_k, Q_k, G, l, λ)
(10) $\quad R_k \leftarrow$ PartialKM(B_k)
(11) $\quad R \leftarrow R \cup R_k$
(12) $\quad Q^{carry} \leftarrow \emptyset$
(13) $\quad$ for each $q \in Q_k$ do
(14) $\quad\quad$ if q is matched to d_q in R_k then
(15) $\quad\quad\quad Q^{carry} \leftarrow Q^{carry} \cup \{q\}$
(16) $\quad\quad$ end if
(17) $\quad$ end for
(18) end for
(19) return R

OnlineKM obtains a locally optimal partial matching result in each time slot based on the current information. The partial matching is implemented by adding dummy no-assignment options to the bipartite graph. If a task is matched to a real, replicated worker node, it is completed in the current time slot. If a task is matched to its dummy no-assignment option, it is treated as unmatched and is carried over to the next time slot. OnlineKM then increases the priority of each carried-over task by one. This priority update gives delayed tasks higher urgency in later matching. Since future workers and future tasks are unknown, OnlineKM does not claim global optimality over the full online horizon.

4.2. Construction of the Priority-Aware Partial Bipartite Graph

Algorithm 2 constructs the weighted partial bipartite graph used by OnlineKM and OnlineARC. Each worker can complete at most l tasks in one time slot. Therefore, the algorithm replicates each worker node l times. This step transforms the capacity-constrained assignment problem into a standard one-to-one matching form. To support carry-over tasks, the algorithm further adds dummy no-assignment options. These dummy options allow the KM-based solver to produce a partial real assignment instead of forcing every task to be assigned to a real worker.

Algorithm 2: Priority-Aware Partial Bipartite Graph Construction

Input: Worker set W_k; task set Q_k; regional graph $G(V, E)$; worker capacity l; Movement weight parameter λ.
Output: Weighted partial bipartite graph B_k.
(1) $B_k \leftarrow \emptyset$; $W_k{}' \leftarrow \emptyset$; $D_k \leftarrow \emptyset$
(2) for each worker $w \in W_k$ do
(3) for r = 1 to l do
(4) create a replicated worker node w^r
(5) $W_k{}' \leftarrow W_k{}' \cup \{w^r\}$
(6) end for
(7) end for
(8) for each task $q \in Q_k$ do
(9) create a dummy no-assignment node d_q
(10) $D_k \leftarrow D_k \cup \{d_q\}$
(11) end for
(12) for each replicated worker node $w^r \in W_k'$ do
(13) for each task $q \in Q_k$ do
(14) $c(w^r, q) \leftarrow d_G(v(w), v(q))$
(15) $score(w^r, q) \leftarrow q.p - \lambda \cdot c(w^r, q)$
(16) add edge $(w^r|q)$ with weight $score(w^r, q)$ to B_k
(17) end for
(18) end for
(19) for each task $q \in Q_k$ do
(20) add edge $(d_q|q)$ with weight θ to B_k
(21) end for
(22) if a square KM matrix is required then
(23) add dummy task nodes with zero-weight edges for unused worker capacity
(24) end if
(25) return B_k

For each replicated worker–task pair, the algorithm computes an edge score. The score combines task priority and cross-region movement cost. A higher score means that the task has a higher urgency or that the worker is closer to the task region. If a minimum-cost KM implementation is used, the score can be transformed into an equivalent cost.

The dummy no-assignment node d_q represents the option of not assigning the task q to any real worker in the current time slot. If q is matched to d_q, the task is treated as unmatched in the current time slot and is added to Q_{k+1}^{carry}. This case corresponds to $y_{jk} = 1$ in the mathematical formulation. If q is matched to a real replicated worker node w^r, the task is completed in the current time slot. This case corresponds to $y_{jk} = 0$ and $x_{ijk} = 1$.

The dummy edge weight θ is used as the no-assignment threshold. In the experiments, θ is set to 0. This setting allows the algorithm to avoid forced long-distance assignments with negative assignment scores. If the KM implementation requires a square cost matrix, dummy task nodes are added to absorb unused worker capacity. Matching a replicated worker node to a dummy task does not represent a real assignment. This match is discarded after the KM step. Therefore, the algorithm supports at most worker capacity and task carry-over within the KM-based matching process.

When one worker receives multiple tasks through replicated worker nodes, each replicated node represents one unit of task capacity. The algorithm computes each task cost from the worker's original region to the task region. Therefore, the capacity model should be interpreted as a region-level assignment model. The capacity model is not a complete multi-stop routing model.

This simplification is acceptable for the intended isolation-sensitive setting for two reasons. First, the main control target is unnecessary regional crossing. The model, therefore, uses hop-based regional movement costs instead of detailed road network travel costs. Second, the platform makes decisions at the time slot level. The exact visiting order may depend on operational details that are unavailable during online matching.

This simplification is mainly suitable for region-level dispatching scenarios. In such scenarios, the platform first controls whether an assignment crosses regional boundaries, and detailed route planning can be handled by a downstream routing module after the assignment decision is made. Therefore, the proposed framework should be interpreted as a region-level assignment model for movement control decisions. The proposed framework is not a complete routing model, pickup and delivery model, or vehicle-routing solution. When the visiting order among assigned tasks becomes the main source of movement cost, a route-aware extension is needed.

This simplification also has a limitation. If one worker receives several tasks in different regions, the true movement cost may depend on the visiting order among those task regions. The current model does not optimize this order. The current model may underestimate the route-level cost in such cases. Future work can extend the framework with route sequencing, pickup and delivery constraints, or vehicle-routing-style optimization.

4.3. Adaptive Regional Coarsening in OnlineARC

OnlineKM directly matches workers and tasks on the current regional graph. However, OnlineKM does not adjust the regional structure according to changing supply–demand distributions. OnlineARC addresses this limitation through δ-balanced adaptive regional coarsening.

Algorithm 3 gives the coarsening procedure. For each region, the algorithm computes worker–task statistics from a recent historical window. The algorithm then checks adjacent

regions. If two adjacent regions have a combined supply–demand imbalance no larger than δ, OnlineARC merges the two regions into one super node.

Algorithm 3: δ-Balanced Regional Coarsening

Input: Regional graph $G(V, E)$; historical matching results R^{hist}; historical window size h; coarsening threshold δ.
Output: Coarsened regional graph $G'(V', E')$.
(1) $V' \leftarrow \emptyset$; $E' \leftarrow \emptyset$
(2) visited[v] $\leftarrow$ FALSE for each $v \in V$
(3) for each regional node $v \in V$ do
(4) if visited[v] = TRUE then continue
(5) compute $v.wn$ and $v.qn$ from R^{hist} within the previous h time slots
(6) group $\leftarrow \{v\}$; visited[v] $\leftarrow$ TRUE
(7) for each adjacent node u of v in G do
(8) if visited[u] = TRUE then continue
(9) compute $u.wn$ and $u.qn$ from R^{hist} within the previous h time slots
(10) if $|(v.wn + u.wn) - (v.qn + u.qn)| \leq \delta$ then
(11) group $\leftarrow$ group $\cup \{u\}$
(12) visited[u] $\leftarrow$ TRUE
(13) end if
(14) end for
(15) create a super node v_g from group
(16) $vg.Q \leftarrow$ union of $v.Q$ for all $v \in$group
(17) $vg.W \leftarrow$ union of $v.W$ for all $v \in$group
(18) $V' \leftarrow V' \cup \{v_g\}$
(19) end for
(20) rebuild edges E' among super nodes according to G
(21) $G' \leftarrow (V', E')$
(22) return G'

After the algorithm creates super nodes, OnlineARC rebuilds the edge set of the coarsened graph. If two super nodes contain at least one adjacent pair of original regions, OnlineARC adds an edge between the two super nodes. The edge weight is the minimum hop distance between any original region in one super node and any original region in the other super node. This rule preserves the shortest cross-region connection between coarsened regions. This rule also ensures that the coarsened graph is consistent with the hop-based movement cost.

The value of δ controls the coarsening strength. A smaller δ produces conservative merging. A larger δ produces more aggressive coarsening. Therefore, δ affects the regional granularity used for movement cost evaluation and the locality of the resulting assignments.

To further clarify this process, Figure 3 illustrates how OnlineARC progressively simplifies the regional graph by merging adjacent regions satisfying the δ-balanced criterion.

Figure 3 shows that regional coarsening reduces the number of active regional nodes in the movement-cost graph before matching. The coarsened graph keeps the adjacency structure among coarsened regions. This design allows OnlineARC to perform priority-aware matching with an adaptive regional cost structure.

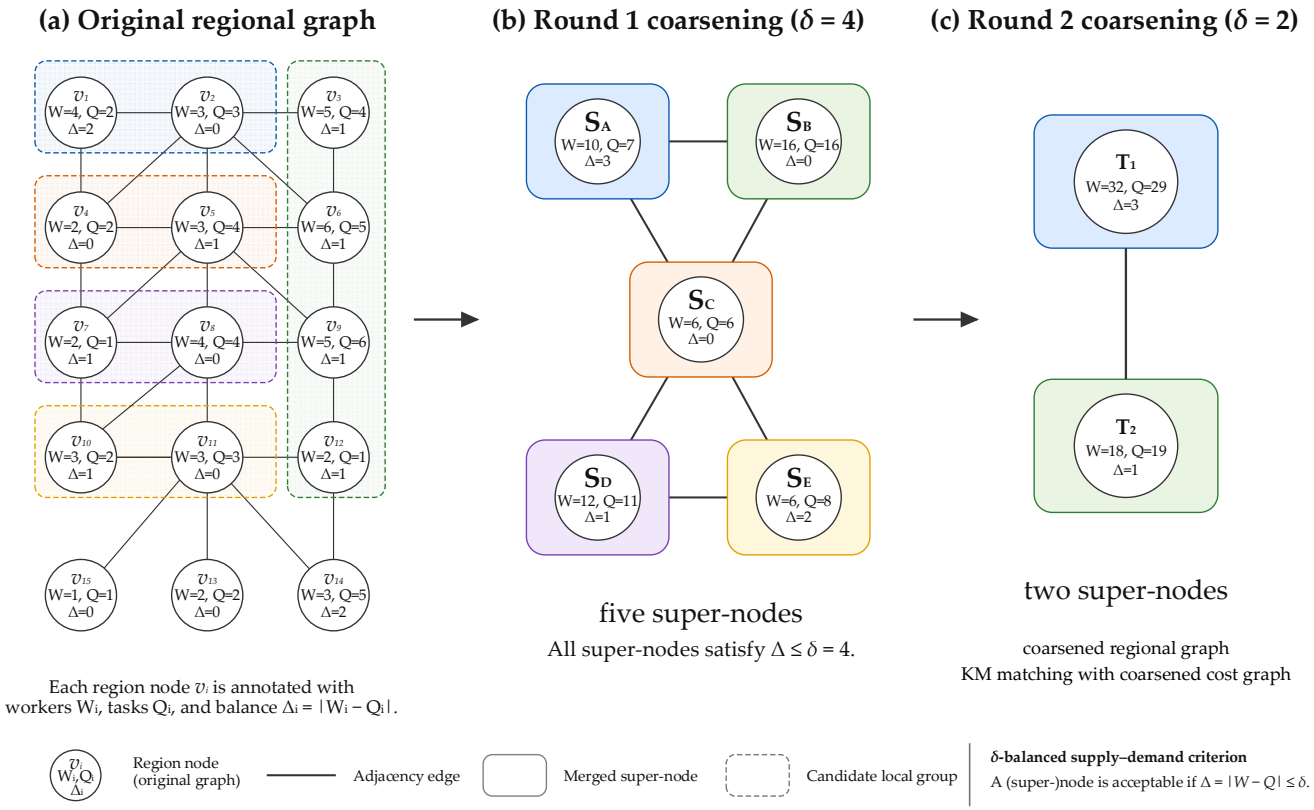


Figure 3. Example of adaptive regional coarsening in OnlineARC. The original regional graph is progressively simplified by merging adjacent nodes that satisfy the δ-balanced supply–demand criterion. The resulting super nodes are used to update the regional graph for subsequent movement cost evaluation in KM-based matching. The colored dashed boxes in panel (**a**) mark different candidate local groups, and the colored filled boxes in panels (**b**,**c**) mark the corresponding merged super-nodes. The colors are used only for visual distinction and do not indicate additional weights or categories.

4.4. OnlineARC Algorithm

Algorithm 4 presents the complete OnlineARC algorithm. OnlineARC follows the same online process as OnlineKM. The main difference is that OnlineARC updates the regional graph before building the bipartite graph. OnlineARC uses recent matching history to coarsen adjacent regions. OnlineARC then performs KM-based matching by using the updated regional graph for movement cost evaluation.

At each time slot, OnlineARC first constructs the current worker set and the current task set. The current task set includes newly arriving tasks and carried-over tasks. OnlineARC then updates the priorities of carried-over tasks. After the priority update, OnlineARC applies δ-balanced regional coarsening by using recent matching history. OnlineARC builds the priority-aware partial bipartite graph by using the updated regional graph for movement cost evaluation and applies KM-based partial matching.

After matching, OnlineARC updates the historical matching record. OnlineARC also carries tasks matched to dummy no-assignment options over to the next time slot. Compared with OnlineKM, OnlineARC adaptively adjusts the regional graph before matching. This adjustment can improve assignment locality when recent worker–task distributions support regional merging.

Algorithm 4: Online Task Matching with Adaptive Regional Coarsening OnlineARC

Input: Time slot sequence $T = \{t_1, t_2, \ldots, t_s\}$; task set Q; worker set W; initial regional graph $G(V, E)$; worker capacity l; movement weight parameter λ; coarsening threshold δ; historical window size h.
Output: Online matching result $R(Q, W, G, T)$; updated regional graph G'.
(1) $R \leftarrow \emptyset$; $R^{hist} \leftarrow \emptyset$; $Q^{carry} \leftarrow \emptyset$
(2) for each time slot $t_k \in T$ do
(3) $W_k \leftarrow \{w \in W \mid w.t = t_k\}$
(4) $Q_k^{new} \leftarrow \{q \in Q \mid q.t = t_k\}$
(5) $Q_k \leftarrow Q_k^{new} \cup Q^{carry}$
(6) for each $q \in Q^{carry}$ do
(7) $q.p \leftarrow q.p + 1$
(8) end for
(9) $G' \leftarrow$ RegionalCoarsening(G, R^{hist}, h, δ)
(10) $B_k \leftarrow$ BuildPartialBipartiteGraph(W_k, Q_k, G', l, λ)
(11) $R_k \leftarrow$ PartialKM(B_k)
(12) $R \leftarrow R \cup R_k$
(13) $R^{hist} \leftarrow$ UpdateHistory(R^{hist}, R_k, h)
(14) $Q^{carry} \leftarrow \emptyset$
(15) for each $q \in Q_k$ do
(16) if q is matched to d_q in R_k then
(17) $Q^{carry} \leftarrow Q^{carry} \cup \{q\}$
(18) end if
(19) end for
(20) $G \leftarrow G'$
(21) end for
(22) return R, G

4.5. Complexity Analysis

Let n_k denote the maximum number of task nodes or replicated worker nodes in time slot t_k. Since KM matching has time complexity $O(n_k^3)$, the total complexity of OnlineKM over s time slots is

$$O\left(\sum_{k=1}^{s} n_k^3\right), \tag{5}$$

and the worst-case complexity is $O(s \cdot n^3)$, where $n = \max_k n_k$.

For OnlineARC, each time slot contains two main operations: regional coarsening and KM-based matching. Let m be the number of regional nodes, and let r be the maximum number of adjacent nodes for a region. The coarsening step has complexity $O(rm)$, or $O(r^2 m)$ when additional neighborhood checking is required. After regional coarsening, OnlineARC builds the partial bipartite graph for the current workers and tasks by using the updated regional graph for movement cost evaluation. Therefore, the KM step still depends on the number of current task nodes and replicated worker nodes. The total complexity of OnlineARC is

$$O\left(\sum_{k=1}^{s} (r^2 m + n_k^3)\right), \tag{6}$$

Compared with OnlineKM, OnlineARC introduces an additional regional coarsening cost. Its practical efficiency depends on whether the updated regional graph improves assignment locality enough to compensate for the graph update overhead. Therefore, OnlineARC is used as a practical adaptive heuristic for isolation-sensitive online task assignment.

4.6. Algorithmic Discussion

OnlineKM focuses on priority-aware per-slot matching. It is simple and stable, but it does not adapt to the regional structure. OnlineARC further uses recent matching history to adjust regional granularity before matching. This strategy should be interpreted as a practical heuristic rather than a universally superior online assignment strategy. The proposed algorithms are empirical online heuristics. They generate slot-level decisions based on currently available workers, tasks, and regional information. They do not provide a competitive ratio, a regret bound, an approximation guarantee, or a worst-case dominance condition. Therefore, the contribution of the proposed algorithms should be understood as an application-oriented empirical improvement rather than a formal worst-case online-optimization guarantee.

OnlineARC is preferable when recent regional supply–demand patterns are stable, when workers and tasks show spatial locality, and when locality improvement is more important than the additional graph update cost. OnlineKM is preferable when strict real-time efficiency is required, when workers and tasks are highly scattered, or when recent historical information is unstable.

Overall, the proposed algorithms combine regional movement control and carry-over urgency under an online isolation-sensitive setting. Different from classical bipartite matching [36], grouped matching strategies [15,20], route-aware assignment [16], and general online spatial crowdsourcing studies, the proposed framework explicitly models task carry-over and region-level movement control within the same online matching process. A detailed discussion of OnlineARC applicability and failure cases is provided in Section 5.14.

5. Experiments

5.1. Dataset, Online Simulation Protocol, and Statistical Validation

The experiments use a real-world task location dataset from Changchun City during the 2022 COVID-19 period. The 19th China Post-Graduate Mathematical Contest in Modeling (2022) provides this dataset. The dataset contains spatial information from nine administrative districts and 17,201 spatial tasks. To evaluate the effect of spatial granularity, the nine districts are divided into grid-based regional nodes with different resolutions: 30×30, 40×40, 50×50, 60×60, and 70×70. These settings generate 900, 1600, 2500, 3600, and 4900 regional nodes, respectively.

Real worker trajectories and real-time worker arrival records are unavailable. Therefore, we generate the worker set according to the number of tasks. We also assign workers and tasks to different time slots through a probabilistic process. For this reason, the experiments should be understood as tests on real task locations with simulated worker availability and online arrivals. The experiments are not a complete real-time platform trace.

The simulation protocol approximates the decision setting of a controlled spatial crowdsourcing platform. The simulation protocol does not reproduce all operational details of a real platform. In real-world emergency logistics or public health service platforms, task requests are spatially distributed. Workers become available over time. The platform must make assignments without future arrivals. The current setting keeps the real spatial distribution of task demand. The current setting simulates the worker side and the online arrival process to reproduce this sequential decision structure.

This modeling choice matches the intended application in three aspects. First, the regional graph reflects a coarse-grained management structure, such as administrative districts, communities, or grid cells. Second, the time slot process reflects periodic platform dispatching under limited current information. Third, the carry-over rule reflects the operational need to reconsider unfinished tasks in later dispatching rounds.

However, the simulation does not capture all real platform conditions. The simulation does not use real worker trajectories, real worker acceptance behavior, road network travel time, order cancellation, individual worker preference, or dynamic service capacity changes. Therefore, the experimental results should be interpreted as controlled simulation evidence based on real task locations. The results should not be interpreted as direct deployment results from a complete spatial crowdsourcing platform.

Although worker availability and online arrivals are simulated, the real task location dataset still keeps the spatial distribution of task demand across administrative districts. This distribution is important for evaluating cross-region movement cost and adaptive regional coarsening.

We set the number of time slots to 10, 20, 30, 40, and 50. Other experimental parameters include the intra-group approximate matching threshold δ_0, the regional coarsening threshold δ, the number of regional nodes $| V |$, the worker–task ratio $| W | / | Q |$, the worker capacity l, and the movement weight parameter λ. Table 1 shows the parameter settings. In each group of experiments, we vary one parameter and keep the other parameters fixed.

Table 1. Experimental parameter settings. Bold values indicate the representative parameter setting: $t = 30$, $\delta_0 = 4$, $\delta = 4$, $| V |= 2500$, $| W | / | Q |= 1.2$, $l = 3$, and $\lambda = 1$. These values are also used as the default fixed values in the parameter-effect experiments unless the corresponding parameter is varied.

Parameter Name	Parameter Value
Number of time slots t	10,20,**30**,40,50
Intra-group approximate match threshold δ_0	0,2,**4**,6,8
Regional coarsening threshold δ	0,2,**4**,6,8
Number of regional nodes $\lvert V \rvert$	900,1600,**2500**,3600,4900
worker–task ratio $\lvert W \rvert / \lvert Q \rvert$	1.0,1.1,**1.2**,1.3,1.4
worker capacity l	1,2,**3**,4,5
Movement weight parameter λ	1

To evaluate the effect of simulation randomness, we further conduct a multi-seed statistical validation under a representative parameter setting. The representative setting is $t = 30$, $\delta_0 = 4$, $\delta = 4$, $| V |= 2500$, $| W | / | Q |= 1.2$, $l = 3$, and $\lambda = 1$. The λ-sensitivity analysis keeps the same representative setting and varies only λ. The service quality and movement control metrics are computed under the same representative setting. The validation is repeated over 30 independent random seeds. In each run, the real task locations are kept fixed. The simulated worker set, worker locations, worker availability slots, and online arrival slots are regenerated under the corresponding random seed. For each seed, all compared algorithms use the same generated online scenario. This paired setting ensures that the statistical comparison reflects algorithmic differences rather than different simulated inputs.

The data generation process can be summarized as follows. First, each real-world task coordinate is mapped to a regional node according to the selected grid resolution. Second, the number of workers is generated according to the worker–task ratio ($|W|/|Q|$). Third, worker regions are uniformly sampled from the regional nodes in the same service space as the task regions. This step generates simulated worker locations while keeping the task-side spatial distribution real. Fourth, task arrival slots are uniformly sampled from the time slot sequence. Worker availability slots are uniformly sampled. Thus, each time slot contains a simulated subset of newly arriving tasks and available workers. Fifth, tasks matched to dummy no-assignment options are carried over to the next time slot. These carried-over tasks are combined with newly arriving tasks at the beginning of the next slot.

Sixth, under each random seed, all compared algorithms use the same generated worker set, task arrival sequence, worker-availability sequence, and carried-over tasks.

The simulation does not impose additional worker-demand spatial correlation beyond the shared regional service space. Uniform sampling of worker regions, task arrival slots, and worker availability slots is used to construct a controlled, reproducible paired comparison. In this setting, all algorithms see the same online inputs for each random seed, so the results primarily reflect algorithmic differences. However, this setting does not fully represent the spatial and temporal heterogeneity of real crowdsourcing platforms. In real platforms, worker locations and availability may be affected by population density, road infrastructure, time-of-day patterns, and local demand intensity. Therefore, the experimental results should be interpreted as controlled semi-real evidence based on real task locations, rather than as direct deployment performance on a full real-world platform.

5.2. Comparison Algorithms

Five algorithms are compared in the experiments. Greedy assigns each task to the worker with the lowest current assignment cost. Greedy is used as a simple, efficiency-oriented baseline [14]. BaseKM [20] constructs a complete worker–task bipartite graph. BaseKM then applies the classical KM algorithm for matching [36]. GroupKM first groups regional nodes according to neighborhood relationships. GroupKM then applies KM to the remaining tasks and workers. GroupKM represents a static region grouping baseline related to grouped matching ideas [15]. OnlineKM performs priority-aware KM-based partial matching in each time slot, considers newly arriving and carried-over tasks, and updates the priorities of tasks matched to dummy no-assignment options. OnlineARC further introduces adaptive regional coarsening before priority-aware KM-based partial matching. OnlineARC uses historical supply–demand information to merge adjacent regions. This design adaptively updates the regional graph and can improve assignment locality when regional merging is useful.

For a fair comparison in the online environment, all offline baseline algorithms (Greedy, BaseKM, and GroupKM) were adapted for the time-sliced process. These baselines were executed independently at each time slot based on the currently available information, and they shared the same task carry-over and priority update mechanism as our proposed OnlineKM and OnlineARC. This setup ensures that the performance gains of the proposed methods are attributable to their internal matching logic rather than to differences in experimental environments.

To examine the contribution of each design component, we add two ablation variants under the representative parameter setting. OnlineKM-NoPriority removes the priority update for carried-over tasks. The variant keeps the same KM-based partial matching and dummy no-assignment mechanism. OnlineARC-Fixed keeps the regional coarsening idea but uses a fixed coarsened regional graph. This variant does not update the graph according to recent historical matching information. These two variants help separate the effects of priority update, KM-based partial matching, and adaptive regional coarsening.

We also include a stronger lookahead reference baseline, Rolling-Horizon KM (RH-KM). RH-KM applies KM-based partial matching over a short rolling horizon with $H = 2$ time slots. RH-KM uses limited future information inside the rolling window. Therefore, RH-KM serves as a stronger reference baseline rather than a strictly online method. This baseline helps estimate how close the proposed online methods are to a short-horizon lookahead strategy.

To evaluate the influence of simulation randomness, Table 2 reports the multi-seed statistical validation under the representative setting. Table 3 reports the component ablation and stronger baseline check under the same setting.

Table 2. Multi-seed statistical validation under the representative parameter setting.

Algorithm	Objective Value, Mean ± Std	Improvement over Best Baseline	Median Paired Objective Reduction vs. Basekm	*p*-Value vs. Best Baseline	Running Time, Mean ± Std
Greedy	8380.83 ± 83.42	-	-	-	430.73 ± 10.15
BaseKM	7573.70 ± 66.49	Best baseline	Reference	-	1121.57 ± 22.55
GroupKM	7904.70 ± 74.75	-	-	-	1041.30 ± 18.75
OnlineKM	7322.57 ± 58.37	3.32%	253	8.59×10^{-7}	760.83 ± 11.37
OnlineARC	7172.47 ± 58.13	5.30%	403	8.61×10^{-7}	1311.57 ± 22.55

Table 3. Component ablation and stronger baseline check under the representative parameter setting.

Method	Priority Update	Adaptive Coarsening	Lookahead	Objective Value, Mean ± Std	Running Time, Mean ± Std	Main Purpose
OnlineKM-NoPriority	No	No	No	7464.21 ± 62.18	748.36 ± 10.94	Effect of priority update
OnlineKM	Yes	No	No	7322.57 ± 58.37	760.83 ± 11.37	Full OnlineKM
OnlineARC-Fixed	Yes	Fixed	No	7246.83 ± 59.72	1086.94 ± 18.63	Effect of fixed coarsening
OnlineARC	Yes	Adaptive	No	7172.47 ± 58.13	1311.57 ± 22.55	Full OnlineARC
RH-KM	Yes	No	$H = 2$	7134.96 ± 56.84	1538.42 ± 29.17	Stronger lookahead baseline

Note: The representative parameter setting is $t = 30$, $\delta_0 = 4$, $\delta = 4$, $|V| = 2500$, $|W|/|Q| = 1.2$, $l = 3$, and $\lambda = 1$. Each result is reported as mean ± standard deviation over 30 independent random seeds. RH-KM uses a rolling horizon of $H = 2$ time slots and is included as a lookahead reference baseline. Lower objective values and lower running times are better.

5.3. Evaluation Metrics

Two primary metrics are used as the main objective and efficiency indicators in this study.

Combined assignment objective value: This metric directly corresponds to the optimization objective in Equation (4). The metric is computed from the real assignment cost and the dummy no-assignment cost. The real assignment cost equals the cross-region movement cost minus the urgency reward of the assigned task. The dummy no-assignment cost represents the carry-over option. Therefore, this metric should not be interpreted as a simple sum of movement cost and urgency reward. A lower value indicates a better combined assignment result under the stated objective.

Running time: This metric records the time required to complete the matching process under different parameter settings. A lower running time indicates higher computational efficiency. This metric also shows whether the online matching process or adaptive regional coarsening introduces additional computational overhead.

To further evaluate the trade-off between isolation control and service quality, we also report service quality and movement control metrics under the representative multi-seed setting. These metrics include task completion rate, average waiting time, maximum waiting time, carry-over events, cross-region assignment ratio, average hop distance, and worker load standard deviation. Completion rate measures the percentage of tasks completed within the finite simulation horizon. Average and maximum waiting times measure the delay between task arrival and completion. Carry-over events count how many times tasks are carried over between consecutive time slots. Cross-region assignment ratio measures the percentage of real assignments with a nonzero regional hop distance. Average hop distance measures the mean regional hop distance of real worker–task assignments. Worker load standard deviation measures the balance of completed task numbers among available workers.

SLA violation rate and region-level risk exposure are not reported because the current dataset does not contain platform-specific SLA thresholds or region-level risk weights. These two indicators are discussed as future evaluation directions.

We report a multi-seed statistical validation under the representative parameter setting. Table 2 reports the objective value and running time as mean ± standard deviation. For the proposed methods, we also report the improvement over the best baseline in terms of the mean objective value. The improvement is computed as follows:

$$\text{Improvement} = \frac{\text{Objective}_{\text{Best baseline}} - \text{Objective}_{\text{method}}}{\text{Objective}_{\text{Best baseline}}} \times 100\% \quad (7)$$

A positive value means that the method obtains a lower mean objective value than the best baseline. Statistical significance is evaluated against the best baseline under the same random seeds. The p-values are computed by paired Wilcoxon signed rank tests. To quantify the magnitude of the paired difference, we also report the median paired objective reduction relative to BaseKM. This effect-size indicator is computed from the 30 seed-level paired objective differences between each proposed method and BaseKM. The running time is measured for the complete matching process of each method in each run.

Note: The representative parameter setting is $t = 30$, $\delta_0 = 4$, $\delta = 4$, $|V| = 2500$, $|W|/|Q| = 1.2$, $l = 3$, and $\lambda = 1$. The improvement is computed relative to BaseKM using the mean objective values. The median paired objective reduction is computed from the 30 seed-level paired objective differences between each proposed method and BaseKM. The p-values are computed by paired Wilcoxon signed-rank tests under the same random seeds. Lower objective values and lower runtimes are better.

Table 2 shows that BaseKM has the lowest mean objective value among the three baselines. Therefore, BaseKM is used as the best baseline for the statistical comparison. OnlineKM reduces the mean objective value by 3.32% compared with BaseKM. OnlineARC reduces the mean objective value by 5.30%. The p-values indicate that these reductions are statistically significant under the representative parameter setting. The median paired objective reduction further quantifies the magnitude of the seed-level paired differences. OnlineKM obtains a median paired objective reduction of 253 over BaseKM, and OnlineARC obtains a median paired objective reduction of 403. These effect-size values show consistent reductions in objective values across paired random seeds. The practical effect should be interpreted as moderate rather than large.

In terms of runtime, Greedy is the fastest method but has the highest objective value. OnlineKM achieves a lower objective value than BaseKM and also uses less runtime. OnlineARC achieves the lowest objective value, but has the highest runtime because of adaptive regional coarsening overhead. This result supports the interpretation of OnlineARC as a locality-aware heuristic. OnlineARC improves the assignment objective under suitable distributions, but it also introduces extra computation.

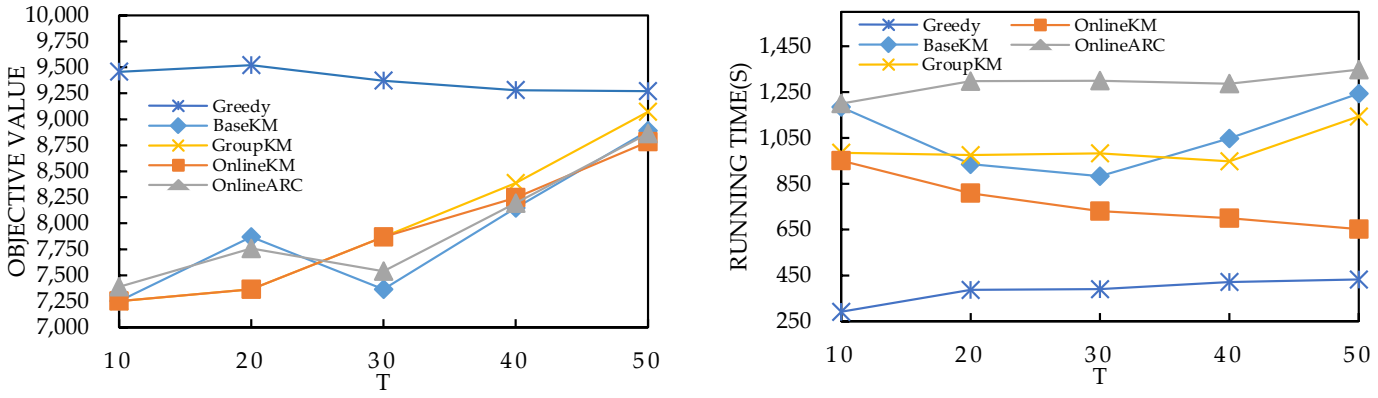
Because the combined assignment objective value depends on λ, objective values should be compared under the same λ. Section 5.11 further reports a sensitivity analysis of λ under the representative setting.

5.4. Effect of the Number of Time Slots

Figure 4 shows the effect of the number of time slots on the combined assignment objective value and running time. As the number of time slots increases, the online assignment process becomes more fine-grained, and more tasks may be temporarily unmatched and carried over to later slots.

OnlineKM and OnlineARC generally achieve lower combined assignment objective values than the static baselines under different time slot settings. Since all compared algorithms share the same carry-over and priority update mechanism in the time-sliced setting, this difference should not be attributed to the priority update rule alone. Instead, the observed differences mainly reflect the matching logic used in each time slot and, for

OnlineARC, the adaptive regional graph updating strategy. Compared with BaseKM, OnlineKM solves smaller per-slot matching instances through time-slice batching. OnlineARC further adjusts the regional graph before matching, which can improve assignment locality when regional merging is suitable. In terms of runtime, OnlineKM is usually faster than BaseKM, while OnlineARC introduces additional coarsening overhead but can benefit from improved assignment locality when regional merging is effective.

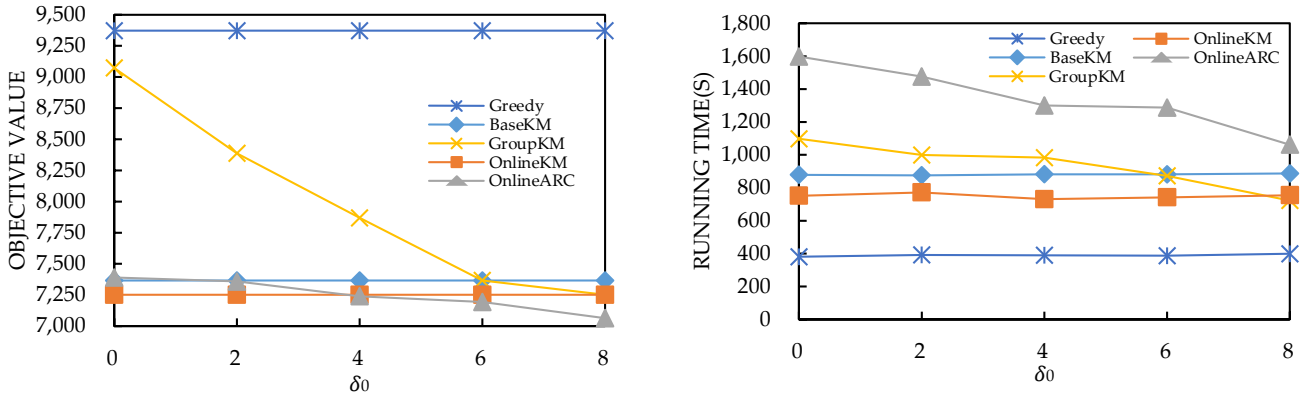


(**a**) (**b**)

Figure 4. Effect of number of time slots on algorithm performance. The varied parameter is t. The fixed parameters are $\delta_0 = 4$, $\delta = 4$, $| V | = 2500$, $| W | / | Q | = 1.2$, $l = 3$, and $\lambda = 1$. (**a**) The impact of the number of time slots on the combined assignment objective value. (**b**) The impact of the number of time slots on running time.

5.5. Effect of the Intra-Group Approximate Matching Threshold δ_0

Figure 5 evaluates the influence of the intra-group approximate matching threshold δ_0. This parameter mainly affects group-based or region-based methods. Therefore, Greedy, BaseKM, and OnlineKM are relatively insensitive to δ_0, while GroupKM and OnlineARC are more directly influenced.



(**a**) (**b**)

Figure 5. Effect of the intra-group approximate matching threshold on algorithm performance. The varied parameter is δ_0. The fixed parameters are $t = 30$, $\delta = 4$, $| V | = 2500$, $| W | / | Q | = 1.2$, $l = 3$, and $\lambda = 1$. (**a**) The impact of δ_0 on the combined assignment objective value. (**b**) Effect of δ_0 on runtime.

As δ_0 increases, more worker–task pairs can be handled within local or approximate groups. This can improve local matching coverage and reduce cross-region movement for group-based methods. This can decrease both the combined assignment objective value and the runtime for group-based methods. However, an overly large threshold may weaken

locality by allowing aggressive regional grouping. Thus, δ_0 controls the trade-off between local matching coverage and regional assignment precision.

5.6. Effect of the Regional Coarsening Threshold δ

Figure 6 reports the influence of the regional coarsening threshold δ. This parameter directly affects OnlineARC because it determines whether adjacent regions can be merged according to their combined supply–demand imbalance.

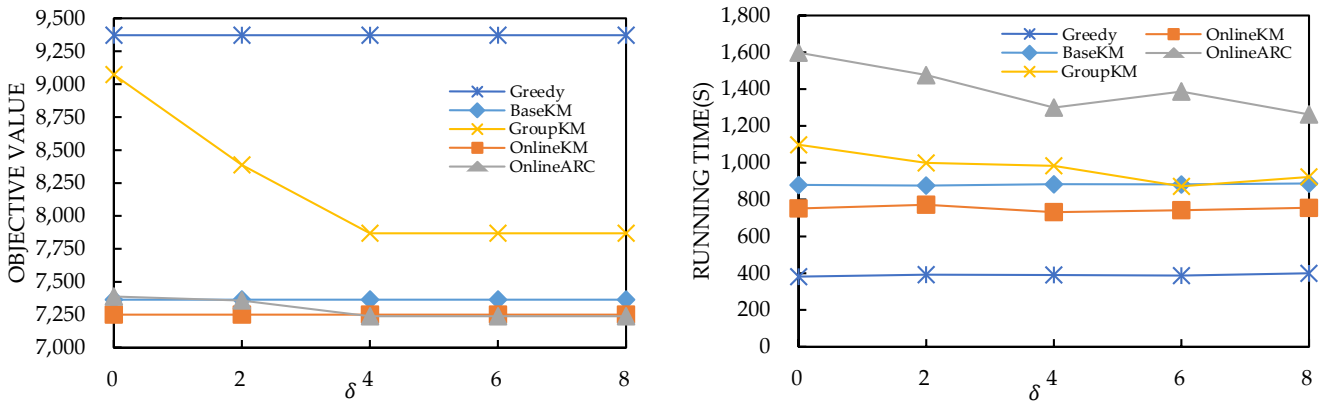


Figure 6. Effect of the regional coarsening threshold on algorithm performance. The varied parameter is δ. The fixed parameters are $t = 30$, $\delta_0 = 4$, $| V | = 2500$, $| W | / | Q | = 1.2$, $l = 3$, and $\lambda = 1$. (**a**) The impact of δ on the combined assignment objective value. (**b**) The impact of δ on runtime.

When δ increases, more adjacent regions satisfy the merging condition. This can improve assignment locality and reduce costly cross-region assignments when the merged regions contain complementary worker–task distributions. It can also reduce the combined assignment objective value when the merged regions contain complementary worker–task distributions. However, the effect is not always monotonic. If δ is too small, few regions are merged, and OnlineARC behaves similarly to OnlineKM; if it is too large, regional coarsening may become too aggressive. Therefore, δ should be selected according to spatial granularity and supply–demand distribution.

A moderate value of δ is more suitable for OnlineARC because it allows adjacent regions with complementary supply–demand patterns to be merged while still preserving useful regional boundaries. If δ is too small, OnlineARC performs little coarsening and behaves similarly to OnlineKM. If δ is too large, the algorithm may merge regions with different worker–task distributions. This aggressive merging can smooth out regional differences and may weaken assignment locality. Therefore, the effect of regional coarsening is conditional rather than monotonic.

5.7. Effect of the Number of Regional Nodes

Figure 7 shows the effect of the number of regional nodes. A larger number of regional nodes corresponds to a finer spatial partition. With finer granularity, workers and tasks are more likely to be located in different regions, which may increase cross-region movement cost.

The results show that the combined assignment objective value of several baseline methods increases as the number of regional nodes grows. OnlineARC alleviates this issue by dynamically merging adjacent regions and recovering a more suitable operational granularity. For runtime, finer regional partitions increase the cost of graph construction and coarsening. Thus, OnlineARC may not always be faster than OnlineKM when

the number of regions is large, reflecting the trade-off between adaptive coarsening and computational overhead.

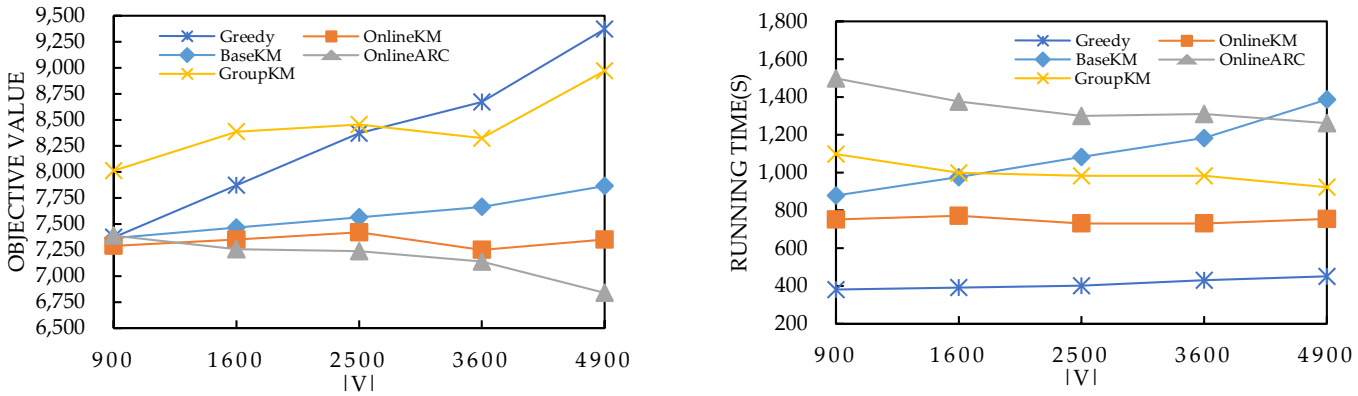


Figure 7. Effect of the number of regional nodes on algorithm performance. The varied parameter is $|V|$. The fixed parameters are $t = 30$, $\delta_0 = 4$, $\delta = 4$, $|W| / |Q| = 1.2$, $l = 3$, and $\lambda = 1$. (**a**) The impact of the number of regional nodes on the combined assignment objective value. (**b**) The impact of the number of regional nodes on runtime.

5.8. Effect of Worker–Task Ratio

Figure 8 evaluates the effect of the worker–task ratio $|W| / |Q|$. As the number of workers increases relative to the number of tasks, the platform has more candidate workers for each task, which generally improves matching flexibility and reduces the combined assignment objective value.

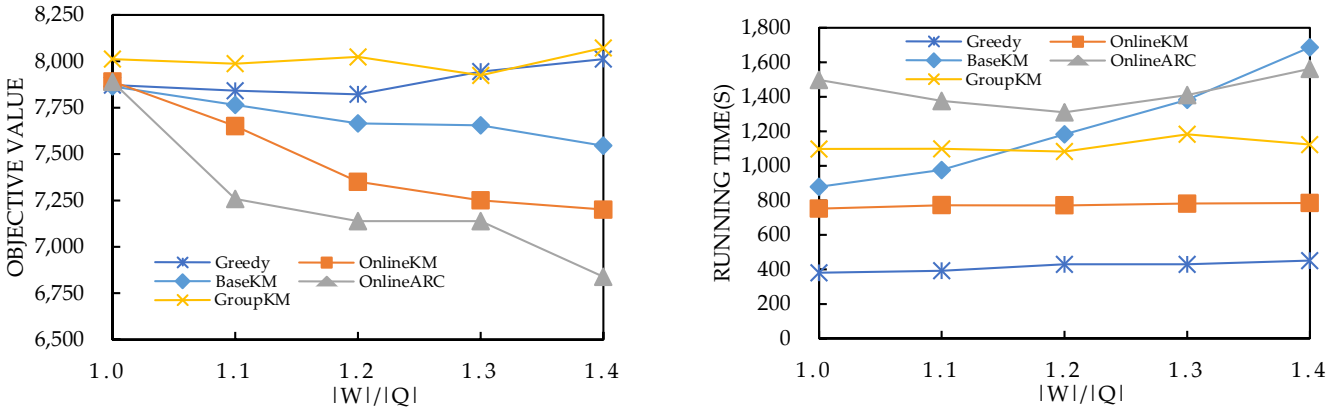


Figure 8. Effect of the worker–task ratio on algorithm performance. The varied parameter is $|W| / |Q|$. The fixed parameters are $t = 30$, $\delta_0 = 4$, $\delta = 4$, $|V| = 2500$, $l = 3$, and $\lambda = 1$. (**a**) Impact of worker–task ratio on the combined assignment objective value. (**b**) The impact of worker–task ratio on running time.

BaseKM, OnlineKM, and OnlineARC generally benefit from a higher worker–task ratio. OnlineARC often achieves greater cost reduction because more available workers increase the probability of local or coarsened-region matching. However, for runtime, a higher worker–task ratio may enlarge the bipartite graph and increase the number of comparisons. Therefore, the efficiency of OnlineARC depends on whether adaptive regional coarsening improves assignment locality enough to offset the graph update overhead.

5.9. Effect of Worker Capacity

Figure 9 shows the effect of worker capacity l. A larger worker capacity means that each worker can complete more tasks in one time slot, which increases assignment flexibility and makes local task completion more likely.

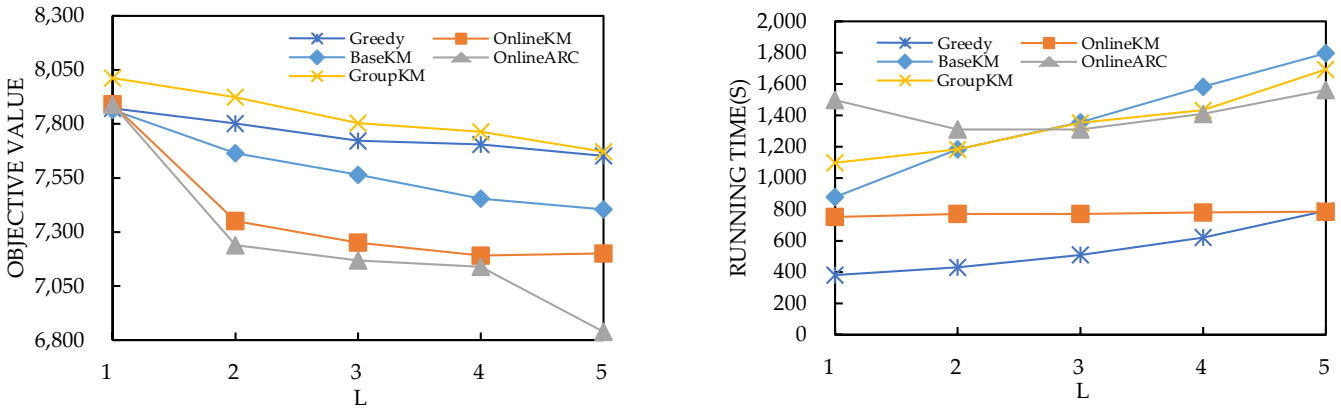


Figure 9. Effect of worker capacity on algorithm performance. The varied parameter is l. The fixed parameters are $t = 30$, $\delta_0 = 4$, $\delta = 4$, $| V | = 2500$, $| W | / | Q | = 1.2$, and $\lambda = 1$. (**a**) The impact of worker capacity on the combined assignment objective value. (**b**) The impact of worker capacity on runtime.

The results show that the combined assignment objective value generally decreases as l increases. This is consistent with the capacity model because higher worker capacity reduces the need to assign additional workers from other regions. However, increasing l also increases the number of replicated worker nodes in the bipartite graph, which may increase the computation required by KM-based matching. OnlineKM reduces this cost through time-slice batching, while OnlineARC further adjusts the regional graph through adaptive regional coarsening.

5.10. Component Ablation and Stronger Baseline Check

The previous experiments compared the algorithms under different parameter settings. To further clarify the contribution of each design component, we conduct an additional ablation and a stronger baseline check under the representative parameter setting. The setting is $t = 30$, $\delta_0 = 4$, $\delta = 4$, $|V| = 2500$, $|W|/|Q| = 1.2$, $l = 3$, and $\lambda = 1$. The same 30-seed protocol and paired online scenarios are used for all methods.

Table 3 reports the results. OnlineKM-NoPriority is used to test the effect of the priority update for carried-over tasks. OnlineKM is the full priority-aware partial matching method without adaptive regional coarsening. OnlineARC-Fixed is used to test whether fixed regional coarsening is sufficient when the regional graph is not updated according to recent matching history. OnlineARC is the full adaptive regional coarsening method. RH-KM is included as a stronger lookahead reference baseline with a rolling horizon of $H = 2$ time slots.

Table 3 separates the effects of the main design components. OnlineKM reduces the mean objective value from 7464.21 to 7322.57 compared with OnlineKM-NoPriority. This result indicates that the priority update helps carry-over tasks receive higher urgency in later matching. OnlineARC-Fixed further reduces the mean objective value to 7246.83. This result suggests that regional coarsening can improve the assignment objective even without historical adaptation. OnlineARC reduces the mean objective value from 7246.83 to 7172.47 compared with OnlineARC-Fixed. This comparison indicates that adaptive coarsening based on recent matching history provides additional benefit over fixed coarsening.

RH-KM achieves the lowest mean objective value, 7134.96, but RH-KM also has the highest runtime. This result is expected because RH-KM uses limited future information within a rolling horizon and solves a larger short-horizon matching problem. Therefore, RH-KM should be interpreted as a stronger lookahead reference rather than a strictly online method. Compared with RH-KM, OnlineARC achieves a close objective value without future information. OnlineARC also requires less runtime than RH-KM. These results support the use of OnlineARC as a practical online heuristic. These results also show the potential benefit of short-horizon lookahead information.

5.11. Sensitivity Analysis of the Movement Weight Parameter λ

The movement weight parameter λ controls the trade-off between regional movement control and urgency reward. To examine whether the algorithmic trend is stable under different trade-off settings, we conduct an additional sensitivity analysis of λ. The representative setting is $t = 30$, $\delta_0 = 4$, $\delta = 4$, $| V | = 2500$, $| W | / | Q | = 1.2$, and $l = 3$. The tested values are $\lambda = \{0.25, 0.5, 1, 2, 4\}$. For each tested λ, all algorithms are rerun under the same 30 random seeds and the same generated online scenarios.

Table 4 reports the results. The objective values should be compared among algorithms under the same λ, because λ changes the scale of the combined assignment objective value.

Table 4. Sensitivity analysis of the movement weight parameter λ under the representative setting.

λ	BaseKM Objective, Mean ± Std	OnlineKM Objective, Mean ± Std	OnlineARC Objective, Mean ± Std	RH-KM Objective, Mean ± Std	OnlineARC Improvement over BaseKM
0.25	6816.42 ± 61.35	6668.25 ± 55.92	6607.38 ± 54.76	6574.91 ± 53.88	3.07%
0.5	7089.56 ± 63.18	6904.73 ± 56.81	6789.62 ± 55.34	6752.44 ± 54.91	4.23%
1	7573.70 ± 66.49	7322.57 ± 58.37	7172.47 ± 58.13	7134.96 ± 56.84	5.30%
2	8462.85 ± 72.64	8103.18 ± 64.72	7876.39 ± 63.58	7827.51 ± 62.49	6.93%
4	10,194.37 ± 86.52	9638.74 ± 78.43	9251.86 ± 75.91	9176.28 ± 74.36	9.24%

Note: The representative setting is $t = 30$, $\delta_0 = 4$, $\delta = 4$, $| V | = 2500$, $| W | / | Q | = 1.2$, and $l = 3$. Each result is reported as mean ± standard deviation over 30 independent random seeds. RH-KM uses a rolling horizon of $H = 2$ time slots and is included as a lookahead reference baseline. Lower objective values are better. The improvement is computed relative to BaseKM under the same λ. Objective values should be compared among algorithms under the same λ, because changing λ changes the scale of the combined assignment objective value.

The results show that the relative performance trend is stable under the tested λ values. OnlineKM and OnlineARC both achieve lower objective values than BaseKM under all tested settings. OnlineARC also achieves lower objective values than OnlineKM. This result indicates that adaptive regional coarsening remains useful under different movement urgency trade-off settings.

The improvement of OnlineARC over BaseKM increases from 3.07% to 9.24% as λ increases. This trend is consistent with the design of OnlineARC. When λ is small, the objective gives less weight to regional movement control, so the benefit of regional coarsening is relatively limited. When λ is large, the objective gives more weight to avoiding regional crossings, so the locality-aware regional coarsening strategy provides a larger benefit. This trend does not imply universal dominance of OnlineARC. The result only shows that OnlineARC remains effective under the tested movement weight settings.

RH-KM achieves the lowest objective value under all tested λ values. This result is expected because RH-KM uses short-horizon future information. OnlineARC remains close to RH-KM without using lookahead information. Therefore, the sensitivity analysis supports the robustness of the proposed online methods under the tested movement weight settings.

5.12. Service Quality and Movement Control Metrics

The previous results mainly evaluated the combined assignment objective value and runtime. To further examine whether the proposed methods provide a better trade-off between isolation control and service quality, we add several service quality and movement control metrics under the representative parameter setting. The setting is $t = 30$, $\delta_0 = 4$, $\delta = 4$, $| V | = 2500$, $| W | / | Q | = 1.2$, $l = 3$, and $\lambda = 1$. The same 30-seed protocol and paired online scenarios are used for all methods.

Table 5 reports the results. The service quality metrics include completion rate, average waiting time, maximum waiting time, carry-over events, and worker load standard deviation. The movement control metrics include cross-region assignment ratio and average hop distance. Lower values are better for waiting time, carry-over events, cross-region ratio, average hop distance, and worker load standard deviation. A higher value is better for completion rate.

Table 5. Service quality and movement control metrics in the representative setting.

Algorithm	Completion Rate (%)	Avg. Waiting Time (Slots)	Max Waiting Time (Slots)	Carry-Over Events	Cross-Region Ratio (%)	Avg. Hop Distance	Worker Load Std
Greedy	99.75 ± 0.14	1.55 ± 0.13	5.57 ± 0.53	2711.13 ± 128.54	34.76 ± 1.25	0.792 ± 0.054	0.934 ± 0.033
BaseKM	99.85 ± 0.11	1.36 ± 0.11	4.82 ± 0.41	2328.59 ± 116.39	30.91 ± 1.03	0.668 ± 0.037	0.867 ± 0.027
GroupKM	99.80 ± 0.11	1.44 ± 0.10	5.21 ± 0.53	2491.47 ± 116.02	28.87 ± 0.90	0.630 ± 0.036	0.949 ± 0.036
OnlineKM	99.93 ± 0.07	1.21 ± 0.09	4.35 ± 0.33	2099.47 ± 90.85	25.18 ± 0.89	0.551 ± 0.033	0.848 ± 0.031
OnlineARC	99.94 ± 0.06	1.26 ± 0.09	4.45 ± 0.45	2160.32 ± 88.66	20.29 ± 0.80	0.425 ± 0.027	0.876 ± 0.033
RH-KM	99.98 ± 0.03	1.14 ± 0.08	4.04 ± 0.28	1943.84 ± 79.18	18.96 ± 0.74	0.403 ± 0.018	0.823 ± 0.023

Note: The representative setting is $t = 30$, $\delta_0 = 4$, $\delta = 4$, $| V | = 2500$, $| W | / | Q | = 1.2$, $l = 3$, and $\lambda = 1$. Values are reported as mean ± standard deviation over 30 independent random seeds. Completion rate is measured within the finite simulation horizon. Cross-region ratio is computed over real worker–task assignments. Average hop distance is computed over real worker–task assignments. RH-KM uses a rolling horizon of $H = 2$ time slots and is included as a lookahead reference baseline.

All methods achieve high completion rates under the representative setting. The completion rates of OnlineKM and OnlineARC are 99.93% and 99.94%, respectively. These results indicate that the proposed online methods do not reduce the assignment objective by sacrificing final task completion.

The waiting-related metrics show different service quality effects. OnlineKM has a lower average waiting time and fewer carry-over events than OnlineARC. This result is reasonable because OnlineKM directly performs priority-aware partial matching on the current regional graph, whereas OnlineARC places greater emphasis on movement control structure through adaptive regional coarsening. The differences are small. OnlineARC keeps the average waiting time at 1.26 time slots and the maximum waiting time at 4.45 time slots.

The movement control metrics show the main advantage of OnlineARC. OnlineARC reduces the cross-region assignment ratio from 25.18% to 20.29% compared with OnlineKM. OnlineARC also reduces the average hop distance from 0.551 to 0.425. These results show that adaptive regional coarsening improves locality-aware assignment and reduces unnecessary regional movement. Compared with BaseKM, OnlineARC also reduces the cross-region assignment ratio and average hop distance clearly.

RH-KM obtains the best values on most additional metrics. This result is expected because RH-KM uses short-horizon future information. OnlineARC remains close to RH-KM in movement control metrics without using lookahead information. Therefore, Table 5 provides additional evidence that OnlineARC improves regional movement control while maintaining acceptable service quality indicators under the tested setting.

5.13. Summary of Experimental Findings

The experimental results show that the proposed online methods usually provide better or competitive performance under the tested settings. Since all compared algorithms use the same carry-over and priority update mechanism in the adapted online environment, the observed differences should be interpreted mainly from matching logic- and regional-structure adaptation. OnlineKM benefits from time-sliced KM-based partial matching based on current information, while OnlineARC further adjusts the regional graph according to recent supply–demand balance.

The multi-seed validation shows statistically significant improvements in objective value over the best baseline under the representative parameter setting, and the median paired objective reductions provide an effect-size description of these differences. The component ablation further separates the sources of improvement. The comparison between OnlineKM-NoPriority and OnlineKM evaluates the contribution of priority update. The comparison between OnlineARC-Fixed and OnlineARC evaluates the contribution of adaptive historical coarsening. The RH-KM result provides a stronger short-horizon lookahead reference and shows the potential benefit of limited future information. This comparison strengthens the empirical evaluation. At the same time, the baseline set is still mainly based on matching heuristics, so the experiments should not be interpreted as a comprehensive comparison with all modern online optimization methods. Broader comparisons with primal-dual online matching, auction-based assignment, and online min-cost flow are left for future work.

The λ -sensitivity analysis further shows that the relative performance trend remains stable under the tested movement weight settings. The reported service quality and movement control metrics show a trade-off between locality control and service quality. OnlineARC reduces the cross-region assignment ratio and average hop distance while keeping a high task completion rate under the representative setting. These results support the movement-control benefit of OnlineARC when regional coarsening provides useful locality information. However, OnlineARC also introduces additional graph update cost, and it may slightly increase waiting-related metrics compared with OnlineKM. Therefore, OnlineARC should be interpreted as improving movement locality with additional computational overhead, rather than as a universally superior method on all service quality indicators.

These results should be interpreted as controlled empirical evidence under the tested settings rather than proof of universal superiority. The multi-seed statistical validation is mainly conducted under the representative parameter setting. Therefore, the statistical results in Tables 2–5 should be interpreted as evidence under this representative setting rather than as full statistical validation over all parameter combinations. The parameter experiments in Figures 4–9 show trend-level behavior when one parameter is varied, and the other parameters are fixed. A full multi-seed validation over all combinations of spatial granularity, worker–task ratio, worker distribution, time slot setting, and coarsening threshold would require a much larger experimental design. Future work should include additional challenging settings, such as low worker–task ratio, high spatial granularity, and non-uniform or clustered worker distributions, to further test the robustness of OnlineARC. Section 5.14 discusses the practical applicability of OnlineARC in more detail.

5.14. Practical Applicability and Failure Cases of OnlineARC

OnlineARC is designed for online scenarios where recent regional supply–demand patterns contain useful spatial information. OnlineARC is more suitable when workers and tasks show spatial locality. OnlineARC is also more suitable when adjacent regions have a supply–demand imbalance and when the historical window captures stable short-term

matching patterns. In such cases, adaptive regional coarsening can update the movement-cost graph and can help the algorithm favor locality-aware assignments.

OnlineARC may be less effective in several cases. First, regional merging may not provide useful locality information when workers and tasks are highly scattered over the service space. In this case, the algorithm may introduce graph update overhead without improving the assignment objective. Second, regional coarsening may not compensate for insufficient worker capacity when most tasks are concentrated in a few regions, but workers are sparse. Third, recent historical matching patterns may be unstable when the time slots are too short. If the time slots are too long, the per-slot matching instance becomes larger. In that case, the cost of graph updating may partly offset the benefit of regional adaptation.

The coarsening threshold δ is also important. A very small δ produces conservative merging, so OnlineARC behaves similarly to OnlineKM. A very large δ may merge heterogeneous regions too aggressively. This can smooth out useful regional boundaries and may reduce assignment precision. Therefore, a moderate δ is preferred when adjacent regions show complementary supply–demand patterns but still preserve meaningful regional boundaries.

Overall, OnlineARC should be used when regional coarsening can improve assignment locality enough to compensate for the graph update overhead. It should not be treated as a universally better strategy for all worker–task distributions.

6. Conclusions

This paper studied isolation-sensitive online task assignment in spatial crowdsourcing. The proposed formulation uses a regional adjacency graph, a time-sliced online process, explicit task carry-over, and a combined assignment objective. This formulation supports region-level movement control decisions in a carry-over-aware online matching process.

OnlineKM and OnlineARC implement this formulation through priority-aware partial matching and adaptive regional coarsening, respectively. OnlineKM performs KM-based partial matching in each time slot. OnlineARC applies δ-balanced adaptive regional coarsening before matching and updates the regional graph used for movement cost evaluation. Experiments based on real task locations and simulated online arrivals show that the proposed methods generally reduce the combined assignment objective value under the tested settings. The multi-seed validation, component ablation, and RH-KM comparison further support the roles of priority update, KM-based partial matching, adaptive regional coarsening, and short-horizon lookahead information.

The results also indicate important boundaries. OnlineARC should be viewed as an adaptive regional coarsening heuristic rather than a universally superior online strategy. OnlineARC is more useful when recent regional supply–demand patterns provide reliable locality information. OnlineARC may be less effective when worker–task distributions are scattered, or graph update overhead dominates.

This study still has limitations. The experiments use real task locations but simulated worker availability and online arrivals. In addition, the proposed algorithms do not provide a competitive ratio, a regret bound, an approximation guarantee, or a worst-case dominance condition. This limitation should be considered when the methods are viewed from a theoretical online optimization perspective. The current evaluation reports the combined assignment objective value, running time, and service quality and movement control metrics. However, the experiments still do not cover all operational dimensions of a real online isolation-sensitive platform. For example, SLA violation rate and region-level risk exposure are not evaluated because the current dataset does not contain platform-specific SLA thresholds or region-level risk weights. Worker capacity is modeled through replicated worker nodes, so the model does not optimize the visiting order among multiple

tasks assigned to the same worker. Future work should evaluate non-uniform worker distributions, clustered worker availability, real worker-arrival logs, and real trajectory records. Future work should also extend the model to route-level and multi-objective optimization, broader service quality metrics, and broader movement weight settings in real-world platform scenarios.

Author Contributions: Conceptualization, F.M. and Y.W.; methodology, F.M.; software, F.M.; validation, F.M., X.G. and Y.W.; formal analysis, F.M.; investigation, X.G.; resources, Y.W.; data curation, X.G.; writing—original draft preparation, F.M.; writing—review and editing, F.M., X.G. and Y.W.; visualization, F.M.; supervision, Y.W.; project administration, Y.W.; funding acquisition, Y.W. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by Department of Education of Liaoning Province, grant number LJ212510153014.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: A supplementary reproducibility package is available at https://github.com/gao4242/IsolationSensitive-SC-Reproducibility.git (accessed on 16 March 2026). The package includes the original task-related data files, processed text files from the original implementation, selected original C++ source files, representative configuration files, a fixed 30-seed protocol, and example scripts. The package supports inspection of the data source, parameter settings, seed protocol, scenario-generation workflow, and metric-calculation workflow. Full regeneration of all reported figures and tables requires the complete internal implementation and full per-seed assignment logs. These materials are not publicly released at this stage because they are part of an ongoing internal research codebase and are subject to data-use and laboratory management restrictions. Within these constraints, the manuscript and repository provide the dataset source, regional partition settings, experimental parameters, seed protocol, and statistical validation design.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Deng, D.; Shahabi, C.; Demiryurek, U. Maximizing the number of worker's self-selected tasks in spatial crowdsourcing. In Proceedings of the 21st ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems, Orlando, FL, USA, 5–8 November 2013; pp. 324–333.
2. Tong, Y.; Zhou, Z.; Zeng, Y.; Chen, L.; Shahabi, C. Spatial crowdsourcing: A survey. *VLDB J.* **2020**, *29*, 217–250. [CrossRef]
3. Kazemi, L.; Shahabi, C. Geocrowd: Enabling query answering with spatial crowdsourcing. In Proceedings of the 20th International Conference on Advances in Geographic Information Systems, Redondo Beach, CA, USA, 6–9 November 2012; pp. 189–198.
4. Gummidi, S.R.B.; Xie, X.; Pedersen, T.B. A survey of spatial crowdsourcing. *ACM Trans. Database Syst.* **2019**, *44*, 8. [CrossRef]
5. Zhao, Y.; Chen, X.; Deng, L.; Kieu, T.; Guo, C.; Yang, B.; Zheng, K.; Jensen, C.S. Outlier detection for streaming task assignment in crowdsourcing. In Proceedings of the ACM Web Conference 2022, Lyon, France, 25–29 April 2022; pp. 1933–1943.
6. Zhu, G.; Li, Y.; Du, S.; Xu, J.; Ding, S.; Xu, M. Aggregative Online Task Assignment in Spatial Crowdsourcing: An Auction-aware Approach. *IEEE Trans. Mob. Comput.* **2025**, *25*, 3998–4012. [CrossRef]
7. Huang, Y.; Liu, Y.; Zhou, X.; Ren, T.; Yang, Z.; Li, K.; Li, K. Optimizing Dynamic Task Assignment in Spatial Crowdsourcing: Bilateral Preference-aware Approaches. *IEEE Trans. Mob. Comput.* **2025**, *25*, 1613–1627. [CrossRef]
8. Zeng, X.; Lin, J.; Deng, L.; Fang, Y.; Zhao, Y.; Zheng, K. Optimizing Multi-Center Collaboration for Task Assignment in Spatial Crowdsourcing. In Proceedings of the 2025 IEEE 41st International Conference on Data Engineering (ICDE), Rotterdam, The Netherlands, 12–16 May 2025; pp. 2838–2851.
9. Zhao, Y.; Xia, J.; Liu, G.; Su, H.; Lian, D.; Shang, S.; Zheng, K. Preference-aware task assignment in spatial crowdsourcing. In Proceedings of the AAAI Conference on Artificial Intelligence, Honolulu, HI, USA, 27 January–1 February 2019; pp. 2629–2636.
10. Zhao, Y.; Zheng, K.; Cui, Y.; Su, H.; Zhu, F.; Zhou, X. Predictive task assignment in spatial crowdsourcing: A data-driven approach. In Proceedings of the 2020 IEEE 36th International Conference on Data Engineering (ICDE), Dallas, TX, USA, 20–24 April 2020; pp. 13–24.
11. Xie, Y.; Wu, F.; Zhou, X.; Luo, W.; Yin, Y.; Zimmermann, R.; Li, K.; Li, K. Trajectory-aware task coalition assignment in spatial crowdsourcing. *IEEE Trans. Knowl. Data Eng.* **2023**, *36*, 7201–7216. [CrossRef]

12. Islam, F.T.; Hashem, T.; Shahriyar, R. A privacy-enhanced and personalized safe route planner with crowdsourced data and computation. In Proceedings of the 2021 IEEE 37th International Conference on Data Engineering (ICDE), Chania, Greece, 19–22 April 2021; pp. 229–240.
13. Zheng, B.; Huang, C.; Jensen, C.S.; Chen, L.; Hung, N.Q.V.; Liu, G.; Li, G.; Zheng, K. Online trichromatic pickup and delivery scheduling in spatial crowdsourcing. In Proceedings of the 2020 IEEE 36th International Conference on Data Engineering (ICDE), Dallas, TX, USA, 20–24 April 2020; pp. 973–984.
14. Zhu, H.; Zhou, M.; Alkins, R. Group role assignment via a Kuhn–Munkres algorithm-based solution. *IEEE Trans. Syst. Man Cybern. Part A Syst. Hum.* **2011**, *42*, 739–750. [CrossRef]
15. Yuan, Z.-W.; Zhang, H. Research on application of Kuhn-Munkres algorithm in emergency resources dispatch problem. In Proceedings of the 2012 9th International Conference on Fuzzy Systems and Knowledge Discovery, Chongqing, China, 29–31 May 2012; pp. 2774–2777.
16. Joshi, M.; Singh, A.; Ranu, S.; Bagchi, A.; Karia, P.; Kala, P. FoodMatch: Batching and matching for food delivery in dynamic road networks. *ACM Trans. Spat. Algorithms Syst.* **2022**, *8*, 6. [CrossRef]
17. Cheng, Y.; Li, B.; Zhou, X.; Yuan, Y.; Wang, G.; Chen, L. Real-time cross online matching in spatial crowdsourcing. In Proceedings of the 2020 IEEE 36th International Conference on Data Engineering (ICDE), Dallas, TX, USA, 20–24 April 2020; pp. 1–12.
18. Tao, Q.; Zeng, Y.; Zhou, Z.; Tong, Y.; Chen, L.; Xu, K. Multi-worker-aware task planning in real-time spatial crowdsourcing. In Proceedings of the International Conference on Database Systems for Advanced Applications, Gold Coast, QLD, Australia, 21–24 May 2018; pp. 301–317.
19. Wang, Z.; Zhao, Y.; Chen, X.; Zheng, K. Task assignment with worker churn prediction in spatial crowdsourcing. In Proceedings of the 30th ACM International Conference on Information & Knowledge Management, Gold Coast, QLD, Australia, 1–5 November 2021; pp. 2070–2079.
20. Zhao, Y.; Zheng, K.; Li, Y.; Su, H.; Liu, J.; Zhou, X. Destination-aware task assignment in spatial crowdsourcing: A worker decomposition approach. *IEEE Trans. Knowl. Data Eng.* **2019**, *32*, 2336–2350. [CrossRef]
21. Zhao, Y.; Zheng, K.; Guo, J.; Yang, B.; Pedersen, T.B.; Jensen, C.S. Fairness-aware task assignment in spatial crowdsourcing: Game-theoretic approaches. In Proceedings of the 2021 IEEE 37th International Conference on Data Engineering (ICDE), Chania, Greece, 19–22 April 2021; pp. 265–276.
22. Li, Y.; Chen, W.; Yan, J.; Li, H.; Gao, L.; Xu, M. Gradient-guided credit assignment and joint optimization for dependency-aware spatial crowdsourcing. In Proceedings of the AAAI Conference on Artificial Intelligence, Philadelphia, PA, USA, 25 February–4 March 2025; pp. 14301–14308.
23. Feng, Z.; Xiao, R.; Xiao, M. Spatial crowdsourcing task allocation for heterogeneous multi-task hybrid scenarios: A model-embedded role division approach. *Front. Inf. Technol. Electron. Eng.* **2025**, *26*, 1144–1163. [CrossRef]
24. Li, X.; Zhao, Y.; Zhou, X.; Zheng, K. Consensus-based group task assignment with social impact in spatial crowdsourcing. *Data Sci. Eng.* **2020**, *5*, 375–390. [CrossRef]
25. Yao, J.; Yang, L.; Liu, H.; Xiong, H. Joint Dependency and Conflicting Task Allocation in Collaboration-Aware Spatial Crowdsourcing. In Proceedings of the 2025 IEEE 41st International Conference on Data Engineering (ICDE), Rotterdam, The Netherlands, 12–16 May 2025; pp. 3562–3574.
26. Zhao, Y.; Guo, J.; Chen, X.; Hao, J.; Zhou, X.; Zheng, K. Coalition-based task assignment in spatial crowdsourcing. In Proceedings of the 2021 IEEE 37th International Conference on Data Engineering (ICDE), Chania, Greece, 19–22 April 2021; pp. 241–252.
27. Li, Y.; Zhao, Y.; Zheng, K. Preference-aware group task assignment in spatial crowdsourcing: A mutual information-based approach. In Proceedings of the 2021 IEEE International Conference on Data Mining (ICDM), Auckland, New Zealand, 7–10 December 2021; pp. 350–359.
28. Zhao, Y.; Zheng, K.; Yin, H.; Liu, G.; Fang, J.; Zhou, X. Preference-aware task assignment in spatial crowdsourcing: From individuals to groups. *IEEE Trans. Knowl. Data Eng.* **2020**, *34*, 3461–3477. [CrossRef]
29. Tong, Y.; She, J.; Ding, B.; Wang, L.; Chen, L. Online mobile micro-task allocation in spatial crowdsourcing. In Proceedings of the 2016 IEEE 32nd International Conference on Data Engineering (ICDE), Helsinki, Finland, 16–20 May 2016; pp. 49–60.
30. Zeng, Y.; Tong, Y.; Chen, L.; Zhou, Z. Latency-oriented task completion via spatial crowdsourcing. In Proceedings of the 2018 IEEE 34th International Conference on Data Engineering (ICDE), Paris, France, 16–19 April 2018; pp. 317–328. [CrossRef]
31. Wang, T.; Xie, X.; Cao, X.; Pedersen, T.B.; Wang, Y.; Xiao, M. On Efficient and Scalable Time-Continuous Spatial Crowdsourcing—Full Version. *arXiv* **2020**, arXiv:2010.15404. [CrossRef]
32. Liu, L.; Cao, N.; Fan, Y. RAP: A Road Network-Aware Multi-Location Task Allocation Framework with Personalized Privacy-Preserving in Spatial Crowdsourcing. *Future Gener. Comput. Syst.* **2025**, *176*, 108129. [CrossRef]
33. Jaya, N.C.A.; Sajeev, G. Proximity-aware worker preferred task allocation for spatial crowdsourcing. *Procedia Comput. Sci.* **2020**, *171*, 1174–1183. [CrossRef]

34. Wen, H.; Lin, Y.; Wu, F.; Wan, H.; Guo, S.; Wu, L.; Song, C.; Xu, Y. Package pick-up route prediction via modeling couriers' spatial-temporal behaviors. In Proceedings of the 2021 IEEE 37th International Conference on Data Engineering (ICDE), Chania, Greece, 19–22 April 2021; pp. 2141–2146.
35. Pournajaf, L.; Xiong, L.; Sunderam, V.; Goryczka, S. Spatial task assignment for crowd sensing with cloaked locations. In Proceedings of the 2014 IEEE 15th International Conference on Mobile Data Management, Brisbane, QLD, Australia, 14–17 July 2014; pp. 73–82.
36. Chen, X. Multi-Objective Task Assignment in Crowdsourcing. In Proceedings of the 2025 IEEE 11th International Conference on High Performance and Smart Computing (HPSC), Las Vegas, NV, USA, 21–23 May 2025; pp. 96–101.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.