Journal of Electronic Research and Application, 2024, Volume 8, Issue 3
http://ojs.bbwpublisher.com/index.php/JERA
ISSN Online: 2208-3510
ISSN Print: 2208-3502

# Optimizing Spatial Crowdsourcing: A Quality-Aware Task Assignment Approach for Mobile Communication

**Jiali Weng, Xike Xie***

University of Science and Technology of China, Hefei 230026, Anhui Province, China

******Corresponding author:*** Xike Xie, xkxie@ustc.edu.cn

**Abstract:** The widespread use of advanced electronic devices has led to the emergence of spatial crowdsourcing, a method that taps into collective efforts to perform real-world tasks like environmental monitoring and traffic surveillance. Our research focuses on a specific type of spatial crowdsourcing that involves ongoing, collaborative efforts for continuous spatial data acquisition. However, due to limited budgets and workforce availability, the collected data often lacks completeness, posing a data deficiency problem. To address this, we propose a reciprocal framework to optimize task assignments by leveraging the mutual benefits of spatiotemporal subtask execution. We introduce an entropy-based quality metric to capture the combined effects of incomplete data acquisition and interpolation imprecision. Building on this, we explore a quality-aware task assignment method, corresponding to spatiotemporal assignment strategies. Since the assignment problem is NP-hard, we develop a polynomial-time algorithm with the guaranteed approximation ratio. Novel indexing and pruning techniques are proposed to further enhance performance. Extensive experiments conducted on datasets validate the effectiveness of our methods.

**Keywords:** Spatiotemporal crowdsourcing; Mobile communication; Task quality

**Online publication:** June 14, 2024

## 1. Introduction

Spatial crowdsourcing, also known as crowdsensing, involves harnessing human knowledge or smartphone sensors for tasks related to physical locations [1]. Current research mainly focuses on short-term assignments, where tasks are matched and assigned to workers until completion [2]. However, if we consider a crowdsourced task set consisting of multiple spatial tasks, each of which takes a long time to finish, time-sharing collaboration from multiple workers is needed. This phenomenon is referred to as spatiotemporal crowdsourcing (STCS) [3]. STCS involves collaboration among multiple workers over time and finds applications in citizen science projects such as environmental monitoring and traffic surveillance. For task assignment in crowdsourcing, existing solutions cannot be directly applied in STCS, as none of them investigate the continuous nature and corresponding spatiotemporal reciprocality between tasks. The novelty of utilizing spatiotemporal reciprocity lies in its ability to leverage the inherent relationships between space and time, allowing for more efficient task assignment and completion. By incorporating spatiotemporal reciprocity, we can better optimize resource allocation, enhance task completion rates, and improve overall data quality in STCS applications. As such, novel approaches are needed to tackle these challenges in STCS effectively.

Achieving a continuous crowdsourced task for all time slots is impractical due to limited budgets [4] and worker availability, resulting in inherently incomplete probed data. To address this incompleteness, interpolation (or extrapolation) techniques estimate unprobed values based on the probed ones. However, the accuracy of these techniques is crucial for maintaining data quality and avoiding the data deficiency problem. Therefore, balancing budget constraints and data quality is paramount, necessitating a valid metric to summarize incompleteness and imprecision in crowdsourced results. Moreover, STCS task assignment involves high computation overhead and the problem is NP-hard. Thus, efficient solutions for task assignment scenarios.

To address the problems we mentioned above, we propose a general entropy-based metric that enables quality-aware continuous crowdsourcing with spatiotemporal reciprocality, ensuring a balance between planned expenditure and observable outcomes. We propose and formalize the STCS problem. Then, we develop an approximation algorithm with a quality guarantee. We also introduce novel indexing and pruning techniques for efficiency enhancement. Experiments on synthetic and real data are conducted to evaluate efficiency and scalability.

# 2. Preliminaries

## 2.1. Basic definitions

### 2.1.1. Task set, tasks, and subtasks

A crowdsourcer sends a task set to the server, denoted as $T = \{\tau_1, \tau_2, ..., \tau_i, ..., \tau_x\}$, where $x$ is the size of the task set, and $\tau_i$ represents a task with location $\tau_i.loc$ and time duration $\tau_i.dur$. Without loss of generality, each task's duration consists of at most $m$ equal-sized time slots. Thus, a spatiotemporal crowdsourcing task $\tau_i$ can be decomposed into its subtasks, $\tau_i = \left\{\tau_i^{(j)}\right\}_{j=1}^{m}$, where $\tau_i^{(j)}.loc = \tau_i.loc$ and $\tau_i^{(j)}.dur = \tau_i.dur/m$.

### 2.1.2. Worker

$W = \{w_1, w_2, ..., w_n\}$ represents a set of $n$ workers. $w_i^{(j)} = 1$ indicates that workers ($w_i$) available at time slot $j$.

### 2.1.3. Cost

$c(\tau_i^{(j)})$ denotes the cost of subtask $\tau_i^{(j)}$. $c(\tau_i^{(j)})$ is the Euclidean distance between $\tau_i^{(j)}$'s location and the assigned $\sum_{j=1}^{m} c\left(\tau_i^{(j)}\right)$ worker $w$'s location. Assuming uniform costs for all workers, the total cost of task $\tau_i$ is $c(\tau_i^{(j)}) =$ .

### 2.1.4. STCS task assignment

Following the task-worker matching scenario [5], task assignment is the mapping of workers to subtasks, and it generates the assigned pairs whose form is <*subtask*, *worker*> at certain time slots. In **Figure 1**, the task set $T$ consists of three tasks, $\tau_1$, $\tau_2$, and $\tau_3$. Each task contains four subtasks and selected workers are assigned to corresponding subtasks under the limit budget.

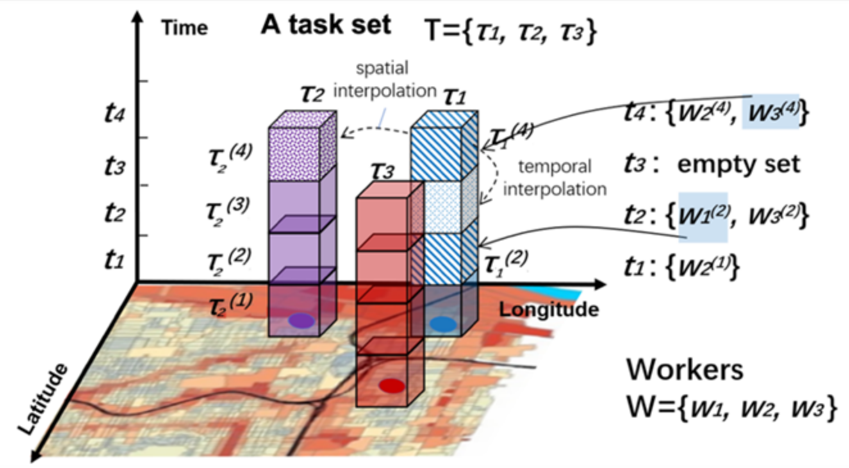


**Figure 1.** An Example of STCS Task Assignment.

## 2.2. Quality metric

### 2.2.1. Task quality

Given a task set $T = \{\tau_1, \tau_2, ...\}$, the task set quality of $T$ as $\boldsymbol{Q(T)} = \sum_{i=1}^{|T|} \boldsymbol{q(\tau_i)}$. Assume a STCS task consists of $m$ subtasks in total, denoted as $\boldsymbol{\tau_i} = \left\{\boldsymbol{\tau_i^{(1)}}, \boldsymbol{\tau_i^{(2)}}, ..., \boldsymbol{\tau_i^{(m)}}\right\}$. Therefore, the quality of task $\boldsymbol{\tau_i}$ is as follows:

$$q(\tau_i) = \sum_{j=1}^{m} q(\tau_i^{(j)}) = -\sum_{j=1}^{m} p(\tau_i^{(j)}) log_2(p(\tau_i^{(j)})) \quad (1)$$

Given the inherent challenge of explicitly defining task quality, we turn to assessing the degree of task execution through task uncertainty. Entropy, a cornerstone of information theory used to quantify uncertainty in events and random variables, serves as our chosen metric for task quality. This decision forms the basis for evaluating both task quality and the information gain resulting from executing a subtask. In Equation (1), a lower entropy value signifies that more subtasks within a task are completed, indicating greater task certainty and thus higher quality. To determine a task's quality, we must retrieve the finishing probabilities $p(\tau_i^{(j)})$ for each of its subtasks.

### 2.2.2. Subtask finishing probability

The finishing probability of a subtask represents its completion status. In an ideal scenario where all $m$ subtasks of an STCS task are executed, the total task finishing probability is 1. However, in cases where m = 1, indicating a single subtask, its finishing probability ranges between [0,1]. Generally, considering practical losses, the finishing probability of a subtask $\tau_i$ is at most $\frac{1}{m}$. We introduce the spatiotemporal interpolation error ratio $\rho_{err}$ to quantify the information loss due to interpolation errors.

$$p(\tau_i^{(j)}) = \frac{1}{m}(1 - \rho_{err}(\tau_i^{(j)})) \quad (2)$$

### 2.2.3. Spatiotemporal interpolation error ratio

The spatiotemporal interpolation error ratio uses spatiotemporal distance for calculation. The following shows the definition:

$$\rho_{err}(\tau_i^{(j)}) = \frac{\sum_{e \epsilon S_{KNN}(\tau_i^{(j)})} \left(|\tau_i^{(j)}, e|\right)}{k \cdot (w_s \cdot |D| + w_t \cdot m)} \quad (3)$$

For spatiotemporal interpolation, we take the spatial weight ($w_s$) and temporal weight $w_t$ to integrate the two types of distances of heterogenous domains, where $w_s + w_t = 1$ and $w_s$, $w_t \geq 0$. This way, the spatiotemporal distance between two subtasks $\tau_i^{(i)}$ and $\tau^{(j)}$ can be represented by $|\tau^{(i)},\tau^{(j)}|=w_s \cdot S_{Dis}|\tau^{(i)},\tau^{(j)}|+w_t \cdot T_{Dis}|\tau^{(i)},\tau^{(j)}|$. So, for spatiotemporal interpolation, the weighted distance is used for retrieving the k nearest neighbors. $S_{Dis}|\tau^{(i)},\tau^{(j)}|$ measure the spatial proximity between subtasks $\tau_i^{(i)}$ and $\tau^{(j)}$. For temporal interpolation, $T_{Dis}|\tau^{(i)},\tau^{(j)}|$ is used to measure the temporal closeness between subtasks $\tau_i^{(i)}$ and $\tau^{(j)}$, referring to the absolute difference of $\tau^{(i)}$ and $\tau^{(j)}$'s timestamps [6].

# 3. Methodology

## 3.1. Problem definition

The objective of task assignment optimization is to maximize the overall quality of all tasks within the task set $T$, subject to a fixed budget constraint. This is formalized as follows:

Problem 1: Task quality maximization (TQM) with fixed budgets — Given a set of tasks $T = \{\tau_1, \tau_2, ...\}$, the TQM problem aims to find an assignment for tasks in $T$, such that the total task set quality $Q(T) = \sum_{i=1}^{|T|} q(\tau_i)$ is maximized, while ensuring that the total cost $\sum c(\tau_i)$ does not exceed the budget ($b$).

Maximize Q($T$)

subject to $\sum_{i=1}^{|T|} c(\tau_i) \leq b$

The NP-hardness of the TQM problem can be proved by related work [7].

## 3.2. Approximation algorithm

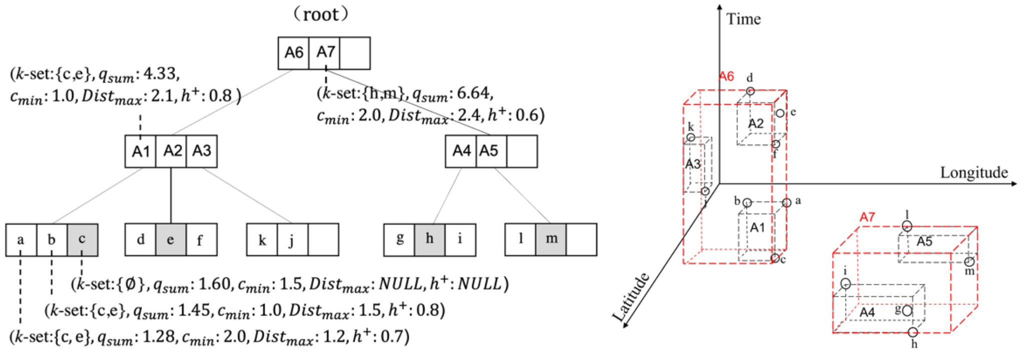
Leveraging the properties of submodularity and non-decreasingness [7] of Q(·), we can derive a suboptimal solution with guaranteed approximation ratios in Algorithm 1. By iteratively selecting the subtask that maximizes the heuristic value $\tau^{(*)}$, the algorithm achieves a (1-1/e) approximation to the optimal solution, as demonstrated in the method [8]. The total time complexity of the algorithm is $O(|T|^2 m^3 \log(m|T|))$.

Algorithm 1: Task assignment algorithm

```
Data: Given budget $b > 0$, a set of workers $W$, a task set $T$
Output: Assigned executed set $T_{cur}$
Initialize the states of subtasks $\{ \tau_i^{(j)} \in \tau_i\}$ in the $T$ as NULL and initialize $T'_{cur}$ and $T_{cur}$ as two empty sets.
For each subtask get $\tau_i^{(j)}$ the corresponding cost c $(\tau_i^{(j)})$;
Execute the subtask $\tau^{(h)}$ yielding the highest quality but not exceeding the budget, $T'_{cur} \leftarrow \{\tau^{(*)}\}$;
while $\sum_{i=1}^{|T|} \boldsymbol{c(\tau_i) \leq b}$ do
    for $\tau_i$ in $T$ do
        for $\tau_i^{(j)} \in \tau - T_{cur}$ dofi
            Compute $\frac{\boldsymbol{Q(T_{cur} \cup \tau_i^{(j)}) - Q(T_{cur})}}{\boldsymbol{c(\tau_i^{(j)})}}$;
    $\tau^{(*)} \leftarrow$ argmax $\left\{\frac{\boldsymbol{\Delta Q(T)}}{\boldsymbol{c(\tau^{(*)})}} : \boldsymbol{\tau^{(*)} \in \tau_i}\right\}$;
    Update selected subtask 's state to Executed;
    $T_{cur} \leftarrow T_{cur} \cup \tau^{(*)}$;
returns $T'_{cur}$ or $T_{cur}$ with the highest quality as the final result;
```

### 3.3. Optimization techniques

However, despite the use of an approximation solution, the algorithm still faces quadratic overhead growth with $|T|$, which optimization techniques are needed. We note that computing the task set quality Q(·) relies on calculating $\rho_{err}(\cdot)$, which in turn depends on retrieving subtasks' *k*-NN. Therefore, we investigate the locality of *k*-NN searching. Our approach involves: (1) saving computation if a subtask's *k*-NN remains unchanged during an iteration, and (2) reducing computation overhead by sharing *k*-NN results among closely distributed subtasks.


(root)
A6 A7
($k$-set:{c,e}, $q_{sum}$: 4.33, $c_{min}$: 1.0, $Dist_{max}$: 2.1, $h^+$: 0.8 )
($k$-set:{h,m}, $q_{sum}$: 6.64, $c_{min}$: 2.0, $Dist_{max}$: 2.4, $h^+$: 0.6)
A1 A2 A3
A4 A5
a b c
d e f
k j
g h i
l m
($k$-set:{∅}, $q_{sum}$: 1.60, $c_{min}$: 1.5, $Dist_{max}$: NULL, $h^+$: NULL)
($k$-set:{c,e}, $q_{sum}$: 1.45, $c_{min}$: 1.0, $Dist_{max}$: 1.5, $h^+$: 0.8)
($k$-set:{c, e}, $q_{sum}$: 1.28, $c_{min}$: 2.0, $Dist_{max}$: 1.2, $h^+$: 0.7)
Time
Longitude
Latitude
A6
A7
A1
A2
A3
A4
A5
a b c d e f g h i k l m

**(a) Subta**sks and MBRs. (b) Aggregated R-tree based on (a).

| Action | Heap | | | | | Result |
|---|---|---|---|---|---|---|
| Visit Root | $h^+(A_6)$: 0.8 | $h^+(A_7)$: 0.4 | | | | {*Empty*} |
| Follow $A_6$ | $h^+(A_1)$: 0.8 | $h^+(A_2)$: 0.7 | $h^+(A_3)$: 0.6 | $h^+(A_7)$: 0.4 | | {*Empty*} |
| Follow $A_1$ | $h(b)$: 0.68 | $h(a)$: 0.65 | $h^+(A_2)$: 0.7 | $h^+(A_3)$: 0.6 | $h^+(A_7)$: 0.4 | {*Empty*} |
| Follow $b$ | $h(a)$: 0.65 | $h^+(A_2)$: 0.7 | $h^+(A_3)$: 0.6 | $h^+(A_7)$: 0.4 | | $\{\tau_s = b\ h(\tau_s) = 0.68\}$ |
| Follow $a$ | $h^+(A_2)$: 0.7 | $h^+(A_3)$: 0.6 | $h^+(A_7)$: 0.4 | | | $\{\tau_s = b\ h(\tau_s) = 0.68\}$ |
| Follow $A_2$ | $h(d)$: 0.69 | $h(f)$: 0.60 | $h^+(A_3)$: 0.6 | $h^+(A_7)$: 0.4 | | $\{\tau_s = b\ h(\tau_s) = 0.68\}$ |
| Follow $d$ | $h(f)$: 0.60 | $h^+(A_3)$: 0.6 | $h^+(A_7)$: 0.4 | | | $\{\tau_s = d\ h(\tau_s) = 0.69\}$ |
| Follow $f$ | $h^+(A_3)$: 0.6 | $h^+(A_7)$: 0.4 | | | | $\{\tau_s = d\ h(\tau_s) = 0.69\}$ |
| Follow $A_3$ | $h^+(A_7)$: 0.4 | | | | | $\{\tau_s = d\ h(\tau_s) = 0.69\}$ |
| Follow $A_7$ | ∅ | | | | | $\{\tau_s = d\ h(\tau_s) = 0.69\}$ |

**(c) Sear**ching process of the subtask that has the maximum heuristic value.

**Figure 2.** Index structure for three-dimensional aggregated R-tree.

#### 3.3.1. Index structure

We introduce a three-dimensional aggregated R-tree index to handle spatial and temporal dimensions [9]. Each node represents a spatiotemporal cuboid covering its child nodes' MBRs. In **Figure 2 (a)**, subtasks and their MBRs are shown, with 13 subtasks illustrated as circles. **Figure 2(b)** displays the aggregated R-tree with some nodes featuring aggregation information. Each index node includes aggregation information like *k*-set (k nearest neighbors), $q_{sum}$ (sum of subtask qualities), $c_{min}$ (minimum subtask cost), $Dist_{max}$ (influence region), and $h^+$ (upper bound of heuristic value change). These support subtask selection by bounding heuristic value calculation and pruning irrelevant branches. Initially, the index is constructed in bulk-loading for all unexecuted subtasks. During each iteration, updates propagate bottom-up if an index node changes. We first explain $Dist_{max}$, the influence region, followed by how aggregation information aids subtask selection, enhancing efficiency.

#### 3.3.2. Influence region of an index node

The so-called influence region of the index node ($x$) is a region, such that if another object is beyond the region, it cannot be  nearest neighbors for any object in 's subtree. It is hard to derive the closed-form equation for the influence region of an index node. Alternatively, it is easy to use $Dist_{max}$ to arbitrate if a subtask touches its influence region. We start by considering the influence region of an object ($o$), whose $k$-NNs are $k\_set(o) = \{o_1, ..., o_k\}$. Let $Dist^{o}_{max}$ be the maximum distance between object $o$ and $\{o_i\}_{i \leq k}$. Intuitively, another object $o'$ cannot be $k$NN of $o$, if the weighted distance $|o,o'|$ is higher than $Dist^{o}_{max}$. Thus, the influence region of object $o$ can be obtained by expanding $o$'s spatiotemporal region with  (called the Minkowski Sum).

### 3.3.3. Index-based subtask selection

To select the subtask with the highest heuristic value, we traverse the index using a best-first approach aided by a priority heap. This heap prioritizes index nodes based on their upper-bound heuristic values, facilitating efficient selection. The upper bound, denoted as $h^{+}(x) = \frac{x.q_{sum}^{new} - x.q_{sum}^{old}}{x.c_{min}}$, quantifies the potential increase in quality achieved by executing subtasks within the node $x$ relative to their cost. In practice, we employ an example to illustrate this selection process using **Figures 2(b)** and **(c)**.

### 3.3.4. Index-based subtask selection

During each round of subtask tentative execution, an identification bit is used to determine whether the node is affected or not. If not, the node quality remains unchanged, otherwise, we update the quality by incorporating the corresponding subtask quality change.

### 3.3.5. Index-based subtask selection

The initial complexity is $O(|T|^2m^3\log(m|T|))$. Considering the subtask solution space is $m|T|$, and with the optimizations, the subtask selection process now takes $O(\log(m|T|))$. The primary computational load occurs during index updates, where each round involves $m|T|)$ subtasks, each requiring $\log(m|T|)$ for updating. Consequently, the index structure update cost is $m|T|\log(m|T|)$ and the overall computational cost for making an execution decision is now reduced to $m|T|\log(m|T|)$. This represents a significant reduction from the original complexity of $O(|T|^2m^3\log(m|T|))$ to $O(|T|m^2\log^2(m|T|))$ Additionally, further improvements in the complexity of finding knearest neighbors could lead to a total complexity of $O(|T|m^2\log(m|T|))$.

## 4. Results

## 4.1. Experiment settings

(1) Datasets

We use a real dataset of 10,357 trajectories for workers' movements. Trajectories are segmented randomly into 1-5 time slots to simulate active periods. A Beijing POI dataset represents task locations. STCS task locations are generated using Uniform, Gaussian, and Zipfian distributions. Costs are based on worker travel distance. We test scalability with 100, 300, and 500 tasks, each with lengths of 10, 20, and 30 subtasks. Budgets range from \$3, \$5, and \$7 for each task assignment, with \$1 representing a unit distance cost. By default, $k$ is set to 3 for the -NN interpolation and the corresponding weight for spatial and temporal dimension  and  are 0.4 and 0.6.

(2) Implementation

Algorithms are implemented in Java and run on a PC with an Intel(R) Xeon(R) CPU E5-2698v4 @ 2.20GHz and 256GB main memory. Experiments focus on task assignment, with reported values averaged across 10 runs.

## 4.2. Quality results

We compare our quality-aware task assignment method, Approx, with a random assignment approach Rand

and the optimal result Opt, in **Figure 3**. Approx consistently achieves higher quality results across different data distributions and budgets compared to Rand and is close to Opt. Overall, Approx provides a superior and deterministic task assignment solution.

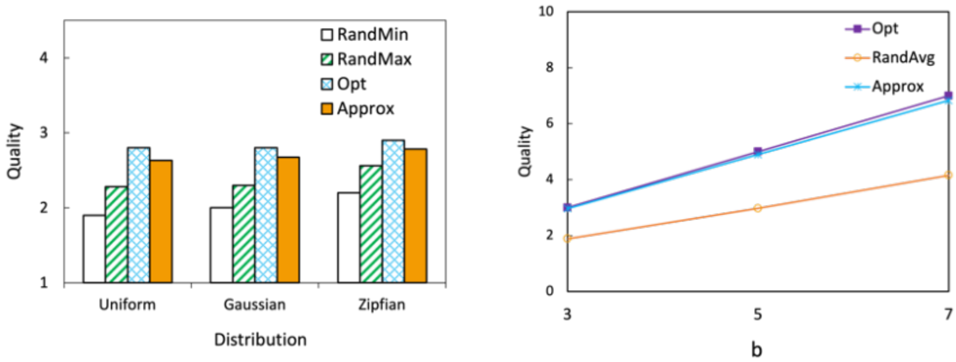
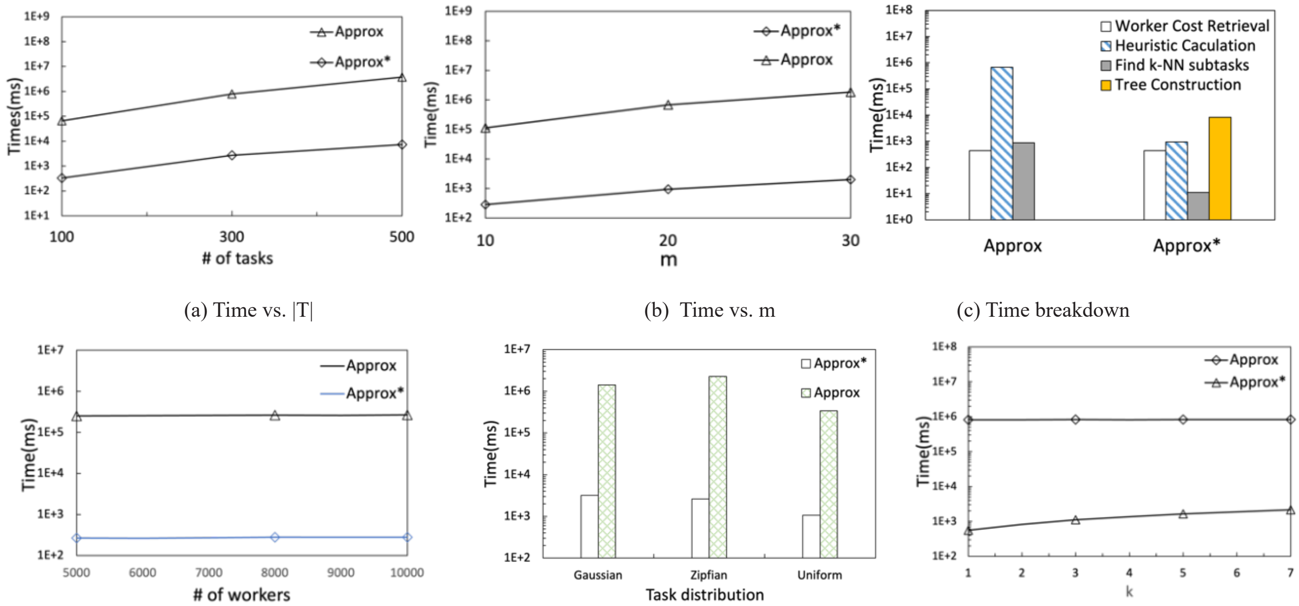


**Figure 3.** Quality of single STCS task assignment.

## 4.3. Efficiency results

We compare the efficiency and scalability of two methods, Approx and Approx*. Approx* improves upon Approx by utilizing a three-dimensional index and employing best-first searching with upper-bound pruning. In our experiments, shown in **Figures 4(a)** and **4(b)**, Approx* consistently outperforms Approx by over two orders of magnitude, demonstrating superior scalability. The breakdown in **Figure 4(c)** reveals that Approx* achieves this through efficient -NN computation and index-based pruning, reducing heuristic value calculation costs significantly. Further tests in **Figure 4(d)** confirm Approx*'s stability and efficiency even with varying worker numbers. Across different task distribution scenarios in **Figure 4(e)**, Approx* consistently outperforms Approx by more than two orders of magnitude, showcasing its dominance in various settings. We test the effect of parameter *k* on the time cost of data interpolation in **Figure 4(f)**.



(a) Time vs. |T|

(b) Time vs. m

(c) Time breakdown

(d) Time of varying |W|

(e) Time of varying distributions

(f) Effect of k

**Figure 4.** Efficiency results of STCS task assignment.

## 5. Conclusion

In this paper, we address the STCS problem, facilitating collaboration among workers for long-term spatiotemporal crowdsourcing. To overcome practical constraints like limited budgets and worker availability, we propose a reciprocal framework for optimizing task assignments, aiming to maximize task finishing quality. We introduce an entropy-based quality metric to measure the incompleteness of crowdsourced results and develop quality-aware task assignment algorithms with budget constraints. We give a unified approximation framework and devise an index structure to enhance processing efficiency. Extensive experiments on datasets demonstrate the effectiveness of the approach.

## Disclosure statement

The authors declare no conflict of interest.

## Author contributions

*Conceptualization:* Jiali Weng
*Writing:* Jiali Weng and Xike Xie

## References

[1] Ye G, Zhao Y, Chen X, et al., 2021, Proceedings of the 30th ACM International Conference on Information & Knowledge Management, November 1–5, 2021: Task Allocation with Geographic Partition in Spatial Crowdsourcing, Association for Computing Machinery, New York, 2404–2413.

[2] Kazemi L, Shahabi C, 2018, Proceedings of the 20th International Conference on Advances in Geographic Information Systems, November 6–9: GeoCrowd: Enabling Query Answering with Spatial Crowdsourcing, Association for Computing Machinery, New York, 189–198.

[3] Xia J, Zhao Y, Liu G, et al., 2019, Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, August 10–16, 2019: Profit-Driven Task Assignment in Spatial Crowdsourcing. Macao, 1914–1920

[4] Chen X, Zhao Y, Zheng K, et al., 2022, 2022 IEEE 38th International Conference on Data Engineering (ICDE), May 9–12, 2022: Influence-Aware Task Assignment in Spatial Crowdsourcing. Kuala Lumpur, 2141–2153.

[5] Karam R, Melchiori M, 2013, Proceedings of the Joint EDBT/ICDT 2013 Workshops, March 18–22, 2013: Improving Geo-Spatial Linked Data with the Wisdom of the Crowds Association for Computing Machinery. New York, 68–74.

[6] Wang T, Xie X, Cao X, et al., 2021, 2021 IEEE 37th International Conference on Data Engineering (ICDE), April 19–22, 2021: On Efficient and Scalable Time-Continuous Spatial Crowdsourcing. Chania, 1212–1223.

[7] Krause A, Guestrin C, 2005, A Note on the Budgeted Maximization of Submodular Functions, Carnegie Mellon University.

[8] Guttman A, 1984, R Trees: A Dynamic Index Structure for Spatial Searching. ACM SIGMOD Record, 14(2): 47–57. https://doi.org/10.1145/971697.602266