Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 198 (2022) 67–76

Procedia Computer Science

www.elsevier.com/locate/procedia

The 12th International Conference on Emerging Ubiquitous Systems and Pervasive Networks (EUSPN 2021)
November 1-4, 2021, Leuven, Belgium

Multi-task versus consecutive task allocation with tasks clustering for Mobile Crowd Sensing Systems

Rebeca Estrada*, Irving Valeriano, Danny Torres

Escuela Superior Politécnica del Litoral, ESPOL, Information Technology Center - TRI Research Program, Guayaquil, Ecuador

Abstract

In this paper, we address the problem of multi-task allocation for dense mobile crowd sensing systems (MCS). The multi-task allocation problem is well-known to be time-consuming as the number of tasks or workers increases. However, the problem can be decomposed into small subproblems based on the location of both tasks and workers. For this reason, we propose to investigate two ways of reducing the complexity of the multi-task allocation: *i*) grouping tasks based on their proximity and their expected QoI such that the quality of information is maximized and *ii*) a meta-heuristic algorithm based on Particle Swarm Optimization (PSO) to solve the selection of workers for several tasks or sequentially solve the workers selection for single task which gives a consecutive allocation of the tasks for each worker. Simulation results shows that the consecutive allocation provides better quality of information for the allocated tasks in comparison with the multi-tasks allocation for a given cluster configuration while guaranteeing the delivery of the requested sensed data on time.

Keywords: Mobile Crowd Sensing; Task Allocation; Particle Swarm Optimization; clustering; K-means

1. Introduction

Currently, smart devices are essential to everybody's life. Thanks to the wide deployment of smart devices, large-scale mobile crowd sensing applications are considered as a promising technique to solve a variety of applications by means of gathering information from several individuals. Mobile crowd-sensing is a new paradigm that recruits ordinary people to contribute sensed data from the embedded sensors of their mobile devices, aggregates and fuses the data in the cloud for crowd intelligence extraction and human-centric service delivery [1]. Fig. 1 shows a typical architecture for mobile crowdsensing. It can be observed that several sensing tasks can be requested from different

* Corresponding author. Tel.: +593-42269-820.
E-mail address: restrada@espol.edu.ec

1877-0509
10.1016/j.procs.2021.12.212

task publishers and the requested information at specific time and locations. On the other hand, mobile users, who might be present at the specific location, accept and perform these sensing tasks. The MCS platform (task manager) is responsible of enabling budget-aware large-scale sensing applications and connecting a set of sensing tasks to the most appropriate set of mobile users. The task manager can allocate the sensing tasks to appropriate workers with different goals such as: maximizing the sensing revenue, providing fairness among the workers, or minimizing the budget. Task allocation approaches determine the set of mobile users to carry out one or several tasks, while taking

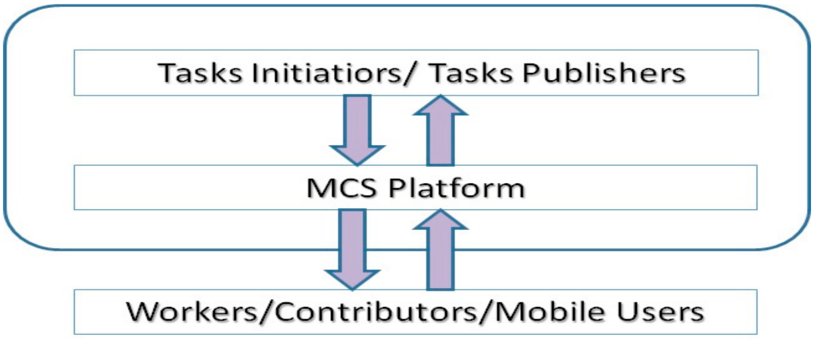


Fig. 1: Mobile Crowd Sensing Components

into account their locations, movement costs, speed of movement, worker payment and reputation levels. The majority of the research approaches are centralized aiming at different objectives such as: improving the aggregated quality of the sensed data [2], maximizing the social surplus [3], opportunistic energy-efficient collaborative sensing [4] or fair and energy-efficient task allocation in MCS [5].

In general, MCS systems rely on crowd-sourced information. In particular, a task may be answered by one or multiple workers depending on the application. Several MCS platforms require a single user to perform a task (i.e. uber) while in others, such as Google Maps, it is required that many users answer a task to guarantee the reliability of the sensed information. Currently, several well-known brands are the customers of Gigwalk[1]. Gigwalk is a MCS platform for checking the on-shelf availability of a product in a store. In this case, the mobile users collect the data at the location and a precise time[6]. Thus, any company can reduce the cost of taking inventories, while keeping the appropriate stock levels at different stores. This suggests that the collection of location-based and time-sensitive data using MCS is a practice of growing importance.

The multi-task allocation and worker selection face several challenges such as location dependency, diversity of quality of the sensing data, and budget and time constraints. The quality of information (QoI) of the sensed data from the mobile user is defined as a value that characterizes how well the sensed data from the worker satisfies the requirement of the task. In order to achieve an adequate level of quality, selecting workers with sufficient QoI is usually a requirement for MCS systems. Handling the tradeoff between the travelled distances of the workers and the QoI of the sensed data in order to maximize the sensing revenue based on limited budget has been investigated by several approaches [7, 8, 9]. Nevertheless, the limitations of the previous approaches can be summarized as follows:

- Centralized approaches tend to be time-consuming and very complex as the number of tasks and workers increases.
- Equal payment to perform a task can be discouraging for some workers that need to travel long distances to perform the same sensing task.
- Clustering techniques have not been investigated to reduce the complexity of the location-based multi-tasks allocation problem.

[1] http://www.gigwalk.com/

In this paper, we propose to analyze the performance of two multi-task allocation frameworks based on the principle of grouping the location-based tasks. Both multi-task allocation frameworks use a distance-based clustering aiming at the maximization of the average required quality of information of the selected clusters. The required quality of a cluster of tasks is determined as the average QoI of the tasks belonging to the cluster. Then, Particle Swarm Optimization approach is used to select the appropriate set of workers to perform the set of tasks using 1) a multi-task allocation approach or 2) a consecutive allocation approach.

The remaining of the paper is organized as follows: Section 2 presents an overview of the relevant related work. Section 3 formulates the multi-t ask allocation problem for a established set of clusters. Section 4 presents the cluster based task management framework using both approaches, multi-task allocation and consecutive allocation. Section 5 describes the most commonly used performance metrics for MCS systems. Section 6 presents the simulation scenarios and the numerical results obtained for the proposed model contrasted with the benchmark models. Finally, Section 7 concludes the paper.

2. Related Work

Crowd-sensing systems are classified by the phenomenon measured and the user involvement in the sensing process [10]. Considering the first criterion, MCS systems can be environmental such as NoiseTube [11], infrastructural (e.g. VTrack [12]), and social such as Foursquare. Regarding the second criterion, MCS can be either participatory or opportunistic. The participatory sensing requires the user involvement to send the sensed data to a server while for opportunistic sensing, the sensed information is sent automatically with minimal user involvement. In both cases, proper incentives should be offered to the users to encourage their participation in the system. Currently, existing crowd-sensing platforms offer three classes of incentives [13]: entertainment [14, 15], service [16, 17, 18], and money rewards [19].

Some researchers introduce redundancy [2] to ensure reliability in MCS systems. This means that several workers are asked to perform the same sensing task. These approaches usually employ averaging techniques, such as majority voting [20], to determine the answer for the requester. Doing this, these approaches reduce the impact of wrong answers on the final result [21], however, they require higher budget to perform a task. Therefore, a trade-off between maximizing the aggregated quality of information per task while minimizing the budget per task should be investigated.

Prior research works aim at optimizing the process of data sensing by efficient assessment of the available resources (i.e. workers with smart devices) to meet the task requirements. Since several factors can be taken into consideration when assigning tasks, this optimization is hard to achieve. Some approaches aim at optimizing only one of these factors such as sensing costs [22], coverage of targets of interest [23], quality or credibility of sensed data [7], and the revenue (i.e. difference between utility and cost) [24].

[25] proposes a sequential single-task allocation approach using Particle Swarm Optimization. The main idea of this approach is to sort the tasks in descending order according to the QoI and then a PSO based worker selection and payment distribution algorithm is carried out. In this model, PSO is used to determine if the worker will perform a specific task and the payment the worker will received per traveled kilometer.

3. System Model

We consider a mobile crowd-sensing system consisting of a set of mobile users (workers) $W = \{w_1, w_2, ..., w_M\}$, and a set of sensing tasks $V = \{v_1, v_2, ..., v_N\}$, where M and N are the number of mobile users and sensing tasks or task publishers, respectively. Each mobile user has a portable device, which is equipped with a set of sensors. Each sensing task v_j is associated with one task publisher β_j, with a given budget B_j representing the monetary incentive to encourage the participation of mobile users.

Our solution consists of two stages: *i*) the selection of the clusters of tasks that maximizes the average QoI among the possible partitions, *ii*) the worker selection based on Particle Swarm Optimization for each reduced set of tasks in each group which can be performed at once or in a sequential manner. The basic idea of the first stage is that clusters are generated by means of k-means algorithm for different cluster size, then, the QoI of each cluster is estimated as the average of the QoI of the tasks belonging to the cluster. Finally, the partition that provides the highest average QoI is selected for then carry out the second phase (i.e. the worker selection).

3.1. Model Parameters

The multi-task allocation problem is a very challenging task, especially when the tasks can be performed by multiple users. Our solution determines clusters of tasks based on their proximity and then aims at maximizing the average quality of information per cluster. To do so, the proposed approach determines the clustered tasks using a distance based clustering scheme and then, solve the maximization of the sum of the quality of information for each cluster, as follows:

$$\max_{\mathbf{X}} \sum_{c \in C} \sum_{j \in N} \sum_{i \in M} \beta_j^c X_i^j QoI_i^j \tag{1}$$

where β_j^c is a given parameter that indicates the task j is a member of cluster c. X_i^j is the binary variable that represents the assignment of the task i to the worker i. QoI_i^j is the quality of information that a worker i can provide for task i and is given as

$$QoI_i^j = R_i \times \phi_i(k) \tag{2}$$

where R_i and phi_i are the reputation and confidence of the worker to perform a task. The reputation is a parameter that is given by MCS system based on the historical worker performance and the confidence is a input parameter given by the worker device, which reflects the worker's confidence to perform any task at a given instant k.

The objective function (1) is subject to the following constraints:

$$\sum_{j \in N} X_i^j \leq N_i^{max} - N_i^A, \qquad ; i \in M \tag{3}$$

$$\sum_{i \in M} X_i^j P_i^j d_i^j \leq B^j, \qquad j \in N \tag{4}$$

$$\sum_{i \in M} X_i^j QoI_i^j \geq QoI_{min}^j \qquad j \in N \tag{5}$$

$$\max_{i \in M} X_i^j t_i^j \leq t_{max}^j, \qquad j \in N \tag{6}$$

Constraint (3) defines the number of tasks that can be allocated to a worker. Since the model runs sequentially for each cluster, some tasks might be already assigned to be performed by one worker, therefore, N_i^A represents the number of tasks previously allocated to worker i. Constraint (4) represents the upper bound for total payment given to the the selected workers while (5) ensures that the set of selected workers satisfy the quality of information required by task. Finally, constraint (6) ensures that the sensed data is delivered on time.

4. Task Allocation Models

4.1. Geo-location based Tasks' Clustering

To reduce the complexity of the multi-task allocation problem, tasks are clustered based on their geographic location. Since the clusters of tasks should be independent, a distance based partitioning clustering algorithm (k-means [26]) is used to group the tasks. We evaluate the average required QoI for all partitions (i.e. Π_k). Then, the partition that achieves the highest average QoI of the clusters is selected. There are no constraints on the number of tasks in every cluster. The requirements of the tasks in each cluster are averaged such that the QoI of the cluster is given by:

$$QoI_{req}^C = \frac{\sum_{j \in C} QoI^j}{N^C} \tag{7}$$

where N^C is the number of tasks in the cluster. The QoI of the partitioning is estimated as the average of the QoI of the clusters in the partitioning

$$QoI^{\Pi_k} = \frac{\sum_{C \in \Pi_k} QoI^C_{req}}{|\Pi_k|}. \tag{8}$$

Algorithm 1 presents how the model evaluates all partition (i.e. possible configurations of clusters) generated by the function k-means.

Algorithm 1: Geo-location based Clustering algorithm

Data: Tasks coordinates l^i,
Tasks (q^i, B^i_{max}, t^i)
Result: Partition Π_k and its set of Cluster,
Tasks belonging to each cluster γ^c_i,
Workers Selection for each cluster β^i_j
begin
  Let T be the number of Tasks;
  eval ← Silhoutte Evaluation;
  for $k \leftarrow 1$ *to* T **do**
    $[idx_c, C]$ =kmeans(l^i, k);
    for $i \leftarrow 1$ *to* k **do**
      Evaluate QoI^C_{req} using Eq. (7);
    end
    for *each cluster* $c \in C$ **do**
      Run Algorithm 2 for all tasks or sequentially;
    end
    Evaluate QoI^{Π_k} using Equation (8);
  end
  Select the partition k with higher QoI^{Π_k};
end

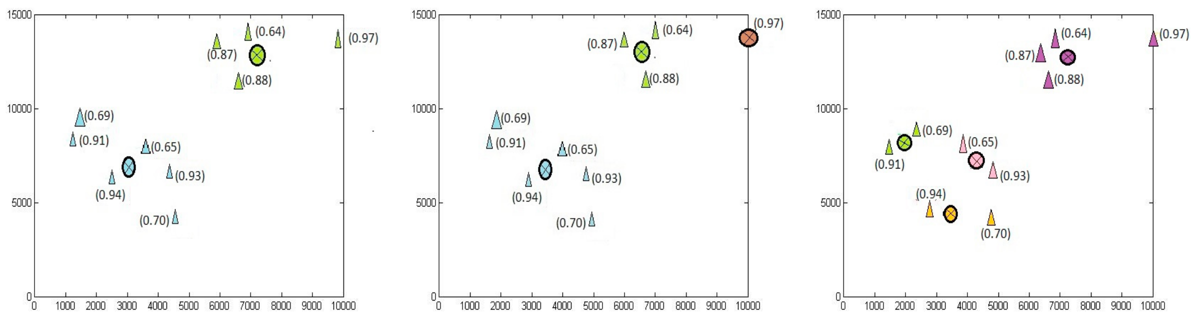
Algorithm 2: PSO multi-task allocation algorithm

Data: MS Worker Locations l_i,
Tasks' Location l^j,
Maximum Budget per Tasks B^j,
Required Time per Tasks t^j_{max}
Result: Set of worker allocated to the task and the price to be paid per worker (X^j_i).
begin
  Generate initial swarm with the particle positions and velocities as follows;
  $\mathbf{X_j} \leftarrow \mathbf{randi(0, 1)}$;
  $\mathbf{v_X} \leftarrow \mathbf{randi(0, 1)}$;
  Evaluate Fitness Function (1);
  Determine first global best of the swarm;
  while $k \leq MaxIteration$ **do**
    Update Position;
    Evaluate Fitness Function;
    Determine best local for each particle;
    Determine best global in the swarm and update the best global;
    Update velocity;
  end
end

Then, the selected partition is the one that maximizes the average QoI among all the partitions, which is expressed as follows:

$$\Pi^{(*)} = \max_k \frac{\sum_{C \in \Pi_k} QoI^C_{req}}{|\Pi_k|} \tag{9}$$

Fig. 2 shows several partitions obtained using the algorithm k-means for a set of 10 tasks for k values between 2 to 4. The tasks are represented by a triangle and were randomly located in an area of 10 km x 15 km. This figure presents the centroid (represented by a circle) and tasks that are clustered with the same color. In addition, the QoI of each task is shown next to its location. We use 8 to determine the average QoI of the partition k in Table 1. It can be observed



(a) Partition with 2 cluster of tasks (b) Partition with 3 cluster of tasks (c) Partition with 4 cluster of tasks

Fig. 2: Example of k-means partition with k = 2, 3 and 4

that third row gives the highest average QoI, therefore, this partition is used for the second stage, which determines the set of worker for each set of clustered tasks in partition Π_3 using Particle Swarm Optimization.

Table 1: QoI of Partitions using k-mean

Number of Clusters	C1	C2	C3	C4	Avg QoI of Partitioning
2	0.81	0.84			0.825
3	0.81	0.79	0.97		0.86
4	0.8	0.82	0.79	0.84	0.8125

4.2. PSO based Multi-task allocation algorithm (PSO-multi)

The second stage of the proposed model is based on Particle Swarm Optimization technique to solve the reduced multi-task allocation problem owing to the fact that PSO has been proven to obtain a satisfying near-optimal solution while speeding up the optimization process in comparison to other evolutionary-based optimization algorithms [27]. PSO is a population-based search approach that requires information sharing among the population members to enhance the search process using a combination of deterministic and probabilistic rules. In the proposed PSO-based task allocation algorithm, one vector X is used to represent the location of each particle n in the search space and one for the speed of particle movement, v_x.

In our model, the vector (X) represents the allocation of the task j to worker i. We use a parameter-less scheme [28], where penalties are based on the average of the objective function and the level of violation of each constraint during each iteration. Thus, the penalty coefficients p_l are determined by

$$p_l = |\overline{f}(x)| \frac{\overline{g_l}(x)}{\sum_{j=1}^{P} [\overline{g}(x)]^2}, \tag{10}$$

where $\overline{f}(x)$ is the average objective function, $\overline{g}(x)$ is the average level of l_{th} constraint violation over the current population and C is the number of constraints. The constraints (3 - 6) are included in $\sum_{l=1}^{C} k_l \widehat{g}(\mathbf{X})$ to penalize unfeasible solutions.

The average of the fitness function for any population is approximately equal to $\overline{f}(x) + |\overline{f}(x)|$. Algorithm 2 presents the PSO-based multi-task allocation algorithm.

4.3. Consecutive Task allocation based on sequential single-task allocation (PSO-CONSECUTIVE)

In this section, we propose to solve the multi-task allocation as the work in [25] to compare the results of using consecutive allocation versus multi-task allocation for the given cluster selection. For this approach, the clustered tasks are sort in descending order according to their quality of information and then a PSO based single-task allocation is perform for each task. This approach is similar to the problem in (1) with the main difference that there is not need of the cap j since the worker selection is performed only for one task each time. One can expect that the running time of this approach will be lower than the purely multi-task allocation approach. The algorithm for this approach is similar to the one shown in Algorithm 2 but with the variation of the variable X_i instead of X_i^j.

5. Performance Metrics

5.1. MCS Platform metrics

Regarding the performance of the MCS platform, two metrics are evaluated, which are the task allocation rate and the effective crowd size given by equations (11) and (12). The first metric represents the percentage of tasks being effectively allocated to workers and the second one measures the number of participating workers in the MCS system.

$$\overline{\phi_T} = \frac{T_{assigned}}{N} \tag{11}$$

$$\overline{SIZE} = \sum_{j \in N} \sum_{i \in M} X_i^j \tag{12}$$

where $T_{assigned}$ is the number of tasks that are effectively allocated and performed within their respective response time.

5.2. Task Performance metrics

Several metrics are considered to evaluate the performance of the model from the point of view of the task publishers, such as the average response time per task, average budget per task, average QoI satisfaction per task and aggregated QoI per Task, which are given in Eqs. (13-16).

$$\overline{t_T} = \frac{\sum_{j \in N} \max_{i \in M}(t_i^j)}{T_{assigned}} \tag{13}$$

$$\overline{S_{QoI}} = \frac{\sum_{i \in N} max\left(1, \sum_{i \in M} QoI_j^i X_i^j - QoI^j\right)}{T_{assigned}} \tag{14}$$

$$\overline{Agg_{QoI}} = \frac{\sum_{j \in N} \sum_{i \in M} \left(X_i^j \times QoI_i^j - QoI^j\right)}{T_{assigned}} \tag{15}$$

$$\overline{B_T} = \frac{\sum_{j \in N} \sum_{i \in M} X_i^j P_i d_i^j}{T_{assigned}} \tag{16}$$

5.3. Workers Performance metrics

As workers performance metrics we consider the average distance traveled and the payment per worker given by Eqs. (17) and (18) respectively. The average distance traveled estimates the average distance that a workers travel to perform the set of allocated sensing task and payment per worker indicates the average payment received by the worker per traveled kilometer.

$$\overline{D_T} = \frac{\sum_{j \in N} \sum_{i \in M} X_i^j d_i^j}{\sum_{j \in N} \sum_{i \in M} X_i^j} \tag{17}$$

$$\overline{P_W} = \frac{\sum_{j \in N} \sum_{i \in M} X_i^j d_i^j P_i}{\sum_{j \in N} \sum_{i \in M} X_i^j d_i^j} \tag{18}$$

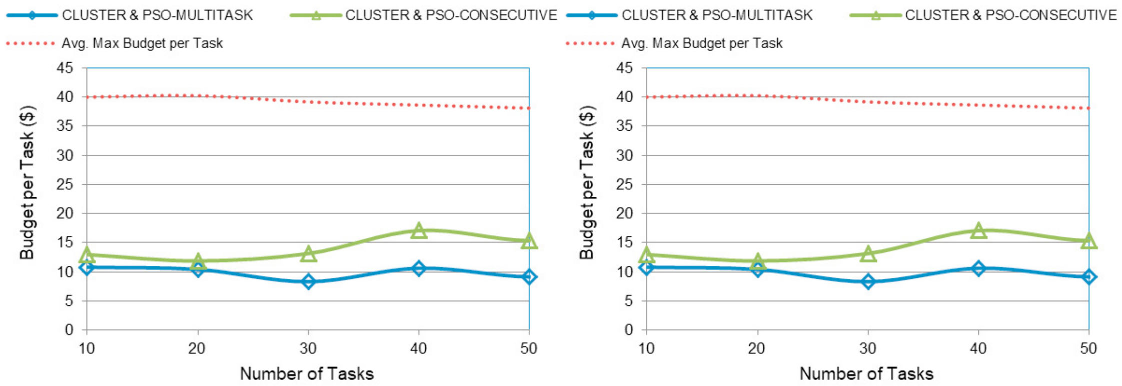
6. Simulation Results

In this section, the two proposed models are evaluated, and their performance metrics are compared. Fig. 3 shows the task allocation rate and number of selected workers to perform the tasks respectively. It can be noticed that the PSO-multi-task model enhances the task allocation rate in comparison with the PSO-consecutive model, however, both models select almost the same number of workers to carry out the tasks.

Fig. 4 presents the average budget per task and response time. The dashed line in Fig. 4a represents the average maximum budget per task. It can be observed that both models used less budget that the maximum allowed per task, however, the PSO-multi model presents the lowest budget. In Fig. 4b, we present the response time plus the running

(a) Task Allocation Rate (b) Crowd Size

Fig. 3: MCS Platform Performance Metrics

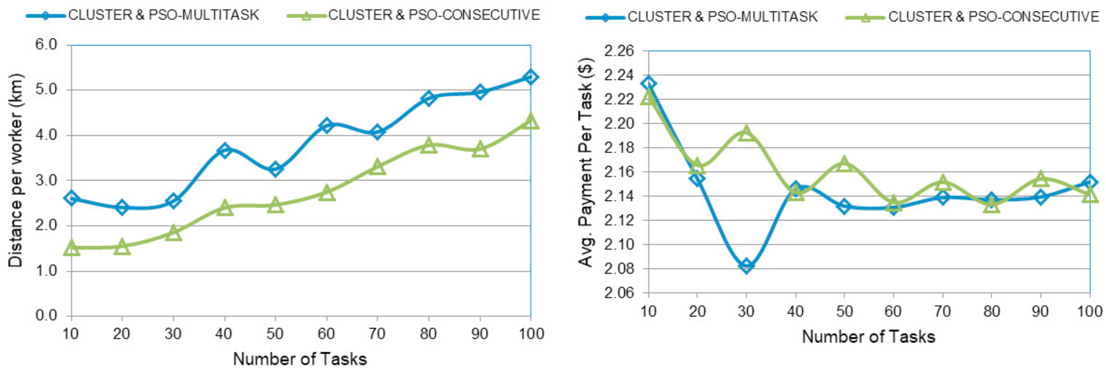


(a) Budget per Task (b) Response Time plus Running Time

(c) Quality of Information per Task (d) Aggregated Quality of Information

Fig. 4: Task Performance Metrics

time of the model, which is the total time for the platform to perform the algorithm and assign the tasks to the selected workers. In the case of more than 40 tasks, the PSO-multi-task model fails to deliver the sensed data on the required response time while PSO-consecutive model can effectively deliver the sensed data on the required time.



(a) Traveled Distance per Worker

(b) Payment per Worker (per Km)

Fig. 5: Worker's Performance Metrics

Regarding the quality of information per task, Figures 4c and 4d present the satisfaction of the delivered QoI and the average aggregated QoI per task for both models respectively. It can be observed that both model presents good QoI satisfaction. However, PSO-consecutive model improves QoI satisfaction in comparison with the PSO-multitask model. PSO-multi model improves the aggregated QoI of information for the cases with less than 30 tasks in comparison with PSO-consecutive model. For more than 50 tasks, the PSO-seq model presents a gain up to 17%.

Worker's performance metrics are shown in Fig. 5. We can observe that the worker's payment per traveled km varies from 2 to 2.24 for both models. From 5a, it is observed that the traveled distance per worker is higher for the PSO-multi model, which gives better total payment for each worker. However, it should be noticed that if the MCS platform would not pay the workers when they deliver the data out the time span, then, the total payment for the workers will be reduced owing to the fact that the PSO-multi algorithm takes longer to allocate the tasks to be performed by each worker (see Fig. 5b).

In summary, the sequential task allocation model enhances the QoI of the allocated sensing tasks while guaranteeing the delivery of the sensed data on time. However, it requires higher budget than the multi-task allocation model and reduces the task allocation rate.

7. Conclusion

This paper focuses on demonstrating the trade-off between using the consecutive single-task allocation versus the multi-tasks allocation for task management in MCS systems. We investigated the performance metrics both type of allocation under the assumption of having established cluster of tasks, which allows to reduce the complexity of the multi-task allocation problem for dense number of tasks and workers. Both proposed solutions are based on PSO, which is an evolutionary-based optimization that has been proven to obtain a satisfying optimal solution while speeding up the optimization process. Simulation results show that the consecutive single-task allocation can effectively enhance the quality of information of the sensing tasks while guaranteeing the delivery of the sensed data on time owing to the fact that the multi-tasks allocation requires longer time to determine the set of workers that will perform the set of clustered tasks. On the other hand, the numeric results show that the multi-task allocation model reduces the budget and increases the task allocation rate in about 10%. As future work, other evolutionary-based optimization mechanism can be investigated to reduce the running time of the PSO-based allocation model or other clustering techniques that can reduce the clustering time.

References

[1] B. Guo, Z. Wang, Z. Yu, Y. Wang, N. Y. Yen, R. Huang, and X. Zhou, "Mobile crowd sensing and computing: The review of an emerging human-powered sensing paradigm," *ACM Comput. Surv.*, vol. 48, no. 1, pp. 7:1–7:31, 2015.

[2] L. Mo, R. Cheng, B. Kao, X. S. Yang, C. Ren, S. Lei, D. W. Cheung, and E. Lo, "Optimizing plurality for human intelligence tasks," in *CIKM*, 2013, pp. 1929–1938.

[3] S. He, D. H. Shin, J. Zhang, and J. Chen, "Toward optimal allocation of location dependent tasks in crowdsensing," in *IEEE INFOCOM 2014 - IEEE Conference on Computer Communications*, April 2014, pp. 745–753.

[4] X. Sheng, J. Tang, and W. Zhang, "Energy-efficient collaborative sensing with mobile phones," in *2012 Proceedings IEEE INFOCOM*, March 2012, pp. 1916–1924.

[5] W. Wu, J. Wang, M. Li, K. Liu, F. Shan, and J. Luo, "Energy-efficient transmission with data sharing in participatory sensing systems," *IEEE Journal on Selected Areas in Communications*, vol. 34, no. 12, pp. 4048–4062, Dec 2016.

[6] M. H. Cheung, R. Southwell, F. Hou, and J. Huang, "Distributed time-sensitive task selection in mobile crowdsensing," in *MobiHoc*. ACM, 2015, pp. 157–166.

[7] C. Zhou, C.-K. Tham, and M. Motani, "QOATA: Qoi-aware task allocation scheme for mobile crowdsensing under limited budget," in *ISSNIP*, April 2015, pp. 1–6.

[8] H. Yu, C. Miao, Z. Shen, and C. Leung, "Quality and budget aware task allocation for spatial crowdsourcing," in *AAMAS*, 2015, pp. 1689–1690.

[9] R. Estrada, R. Mizouni, H. Otrok, and A. Mourad, "Task coalition formation for mobile crowdsensing based on workers' routes preferences," *Vehicular Communications*, vol. 31, 2021.

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

[11] N. Maisonneuve, M. Stevens, M. Niessen, and L. Steels, "Noisetube: Measuring and mapping noise pollution with mobile phones," in *ITEE*. Springer, 2009, pp. 215–228.

[12] A. Thiagarajan, L. Ravindranath, K. LaCurts, S. Madden, H. Balakrishnan, S. Toledo, and J. Eriksson, "Vtrack: Accurate, energy-aware road traffic delay estimation using mobile phones," in *SenSys*, 2009, pp. 85–98.

[13] X. Zhang, Z. Yang, W. Sun, Y. Liu, S. Tang, K. Xing, and X. Mao, "Incentives for mobile crowd sensing: A survey," *Communications Surveys Tutorials, IEEE*, vol. PP, no. 99, pp. 1–1, 2015.

[14] M. Bell, S. Reeves, B. Brown, S. Sherwood, D. MacMillan, J. Ferguson, and M. Chalmers, "Eyespy: Supporting navigation through play," in *CHI*. ACM, 2009, pp. 123–132.

[15] L. Barkhuus, M. Chalmers, P. Tennent, M. Hall, M. Bell, S. Sherwood, and B. Brown, "Picking pockets on the lawn: The development of tactics and strategies in a mobile game," in *UbiComp*. Springer-Verlag, 2005, pp. 358–374.

[16] K. Lan, C. Chou, and H. Wang, "An incentive-based framework for vehicle-based mobile sensing," in *3rd International Conference on ANT, 9th International Conference MobiWIS*, 2012, pp. 1152–1157.

[17] C.-M. Chou, K. chan Lan, and C.-F. Yang, "Using virtual credits to provide incentives for vehicle communication," in *ITST)*, Nov 2012, pp. 579–583.

[18] A. Gupta, W. Thies, E. Cutrell, and R. Balakrishnan, "mclerk: Enabling mobile crowdsourcing in developing regions," in *CHI*, 2012, pp. 1843–1852.

[19] W. Mason and D. J. Watts, "Financial incentives and the performance of crowds," *SIGKDD Explor. Newsl.*, vol. 11, no. 2, pp. 100–108, 2010.

[20] L. Varshney, J. Rhim, K. Varshney, and V. Goyal, "Categorical decision making by people, committees, and crowds," in *ITA Workshop*, Feb 2011, pp. 1–10.

[21] J. Surowiecki, *The Wisdom of Crowds*. Anchor, 2005.

[22] I. Koutsopoulos, "Optimal incentive-driven design of participatory sensing systems," in *INFOCOM*, April 2013, pp. 1402–1410.

[23] L. Jaimes, I. Vergara-Laurens, and M. Labrador, "A location-based incentive mechanism for participatory sensing systems with budget constraints," in *PerCom*, March 2012, pp. 103–108.

[24] C. Do, Ngoc .and Hsu and N. Venkatasubramanian, "Crowdmac: A crowdsourcing system for mobile access," in *Middleware*. Springer-Verlag New York, Inc., 2012, pp. 1–20.

[25] R. Estrada, R. Mizouni, H. Otrok, A. Ouali, and J. Bentahar, "A crowd-sensing framework for allocation of time-constrained and location-based tasks," *IEEE Transactions on Services Computing*, vol. 13, no. 5, pp. 769–785, 2020.

[26] T. Kanungo, D. M. Mount, N. S. Netanyahu, C. D. Piatko, R. Silverman, and A. Y. Wu, "An efficient k-means clustering algorithm: analysis and implementation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 7, pp. 881–892, Jul 2002.

[27] E. Elbeltagi, T. Hegazy, and D. Grierson, "Comparison among five evolutionary-based optimization algorithms," *Adv. Eng. Inform.*, vol. 19, no. 1, pp. 43–53, 2005.

[28] Perez.R. and K. Behdinan, "Particle swarm optimization in structural design," in *Swarm Intelligence: Focus on Ant and Particle Swarm Optimization*, 2007, pp. 532–555.