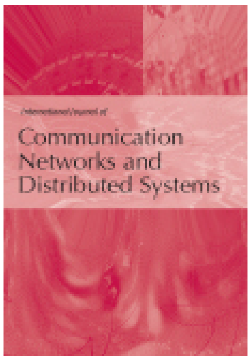


International Journal of Communication Networks and Distributed Systems

ISSN online: 1754-3924 - ISSN print: 1754-3916
https://www.inderscience.com/ijcnds

Cat swarm optimisation-based mobile sinks scheduling in large-scale wireless sensor networks

Srinivasulu Boyineni, K. Kavitha, Meruva Sreenivasulu

DOI: 10.1504/IJCNDS.2022.10044814

Article History:

Received:	24 November 2021
Accepted:	01 December 2021
Published online:	06 December 2022

Cat swarm optimisation-based mobile sinks scheduling in large-scale wireless sensor networks

Srinivasulu Boyineni* and K. Kavitha

Department of CSE,
Annamalai University,
Chidambaram, India
Email: srinivasulu.inf@gmail.com
Email: kavithacseau@gmail.com
*Corresponding author

Meruva Sreenivasulu

Department of CSE,
K.S.R.M. College of Engineering,
Kadapa, Andhra Pradesh, India
Email: mesrinu@rediffmail.com

Abstract: In wireless sensor networks (WSNs), the hotspot problem is one of the major challenging issues because it isolates some network parts and interrupts the data routing. The hotspot problem is mitigated through a mobile sink, where it visits a set of nodes in the network called rendezvous points, whereas the remaining nodes traverse their data to it. In large-scale WSNs, the travelling distance of MS is longer, and it increases the delay of reaching an RP. So, the data overflow may occur due to a limited buffer of sensor nodes. This problem is avoided by increasing the number of mobile sinks in the WSNs. In this context, a cat-swarm optimisation algorithm is used to decide the optimal set of mobile sinks and a simple geometric method to determine the optimal visiting order for each mobile sink. The proposed work is compared with start-of-art literature, and the proposed work outperforms them.

Keywords: wireless sensor networks; WSNs; data acquisition; multiple mobile sinks; cat swarm optimisation; ant colony optimisation.

Reference to this paper should be made as follows: Boyineni, S., Kavitha, K. and Sreenivasulu, M. (2023) 'Cat swarm optimisation-based mobile sinks scheduling in large-scale wireless sensor networks', *Int. J. Communication Networks and Distributed Systems*, Vol. 29, No. 1, pp.47–70.

Biographical notes: Srinivasulu Boyineni received his BTech in Computer Science and Engineering from the SK University, Anantapuram, in 2007, his MTech in Software Engineering from the JNTU Hyderabad, Hyderabad, in 2010, and currently pursuing his PhD in Computer Science and Engineering from the Annamalai University, Chidambaram, India. He is currently working as Assistant Professor of Computer Science and Engineering Department,

Gouthami Institute of Technology and Management for Women, Proddatur, India. His current research interests include wireless sensor networks and ad hoc network.

K. Kavitha received her BE in Computer Science and Engineering and her ME in Computer Science and Engineering in 1998 and 2007 respectively. She received her PhD in Computer Science from the Annamalai University in 2014. She is currently an Associate Professor in the Computer Science and Engineering Department, Annamalai University. Her research interest includes ad hoc network, vehicular networks, sensor networks and IoT.

Meruva Sreenivasulu received his BTech in Computer Science and Engineering from the K.L. College of Engineering (K.L. University), Vaddeswarm, AP, India, in 1990. He completed his ME in Computer Science and Engineering from the Jadavpur University, Calcutta, India, in 1999. He also received his PhD in Computer Science and Engineering from the JNTUA, Anaanthapuramu, in 2013. He is having 26 years of teaching experience. He is currently working as a Professor and the Head of the Computer Science and Engineering Department, K.S.R.M.College of Engineering (Autonomous), Kadapa. He is life member of ISTE and member of IE(I). He published 16 papers in international journals and presented 13 papers in national and international conferences. His research interests includes computer networks, cloud computing, data mining, machine learning, and natural language processing.

1 Introduction

Wireless sensor network (WSN) is one of the most promising and using many applications, including structural health monitoring (SHM), forest fair, healthcare, etc. A group of battery-equipped sensor nodes (SNs) is deployed randomly in an area to observe the activities of the field of interest and intimate to the sink for further analytic and decision (Praveen Kumar et al., 2019). Due to low battery, some of the SNs fall heavy data relays causes the drain of the battery soon. The early drain of the battery disconnects the network and interrupts the data routing from the disconnected network partition (Najjar-Ghabel et al., 2020; Sah et al., 2019; Banoth et al., 2021). Replacing or recharging the battery of such a node is always not possible in the harsh network. Instead, it is recommended to optimise the routing or energy consumption to prolong the network more longer with no interruptions for data gathering (Singh and Kumar, 2020; Liu et al., 2020a).

Data gathering using a mobile sink (MS) is an alternate solution to minimise the SN's energy usage. The MS acquires the packets from the SNs instead of routing through the relay nodes to the base station. But, visiting each SN leads a travel path to the MS, and limited buffered nodes can drop their packets if the MS reaches late. Also, So, the hop count between the SNs to be minimised to overcome the issue. An alternate solution is proposed for this challenge in Kumar et al. (2018) and Donta et al. (2020). Here, the WSN is partitioned into clusters, and a head node called rendezvous point is identified, where the MS visits only it during the packet collection. But identifying such an RPs is challenging because increasing RPs also increases the delay of reaching it to an RP due to a longer path. Fewer RPs increase the number of hops between the SNs and an RP. It is more complex in large-scale WSNs.

Maintenance cost of the WSN increases while increasing the MSs, and using fewer MSs faces similar problems like a single MS. So, this paper identifies the near-optimal solution, including optimal MSs determination and their scheduling strategies. Multiple MSs (MMS) can mitigate these issue. The MMS can be scheduled in different parts of the network to acquire the packets and hand over to the sink. In this, the MMS can receive either directly from the SNs or RPs depends on their usability (Hojjatinia et al., 2021; Anwit et al., 2020). But, identify the requirement of MSs to gathering the data is a challenging task. In case we increase with more MSs, the hardware cost also increases. It also increases the maintenance cost. In the case of fewer MS, it further leads to the same issues we faced through the single MS. So, deciding the optimal MMS to gather the data from the network is a challenging task (Srinivas et al., 2020), and efficient handling of all these devices is a further challenging issue.

In this article, we propose an efficient data gathering approach using cat swarm optimisation (CSO) through a near-optimal set of MS by keeping all the above challenges in mind. It is also necessary to control all the MS through optimal scheduling for data collection in a WSN. In this context, the proposed work adopts a computational geometric approach to efficiently schedule the MS in each network partition to collect the data. Further, an additional mobile device is used to acquire the data packets from scheduled MSs in each division of the WSN. The contributions of this paper are summarised as:

- A CSO technique is used to partition the WSN into clusters from the deployed SNs. This process also decides the optimal set of SNs to acquire data through MS.
- A MS is assigned to each partition of the WSN and a static path is constructed using the computational geometry algorithm with minimal computational complexity.
- A global MS is scheduled in the WSN to acquire the data packets from each local MS by traversing an efficient and dynamic travelling route using ACO. This process does not interrupt or hold the local MS operations.
- The performance efficiency of the CSOMMS is tested through simulations using various performance metrics and compared them with the state-of-the-art works.

The remaining paper is organised as follows. In Section 2, we present the state-of-art literature. In Section 3, we present the system model along with the problem formulation in detail. In Section 4, we present the proposed CSOMMS along with its complexity. In Section 5, we test the proposed CSOMMS algorithm using simulations and compare the results with the existing but related approaches. Finally, we conclude with future scope in Section 6.

2 Related work

In this section, we study the recent MSs scheduling algorithms along with their benefits and limitations. We summarise the discussion using Table 1.

Table 1 Summary on MSs-based data collection literature

Author	*Methods/algorithms*	*Advantages*	*Limitations*
Najjar-Ghabel et al. (2019)	Ant colony optimisation	Delay minimisation	Not tested for large scale WSNs
Krishnan et al. (2019)	Ant colony optimisation	Energy optimisation	Optimal MSs are not determined
Farzinvash et al. (2019)	A spanning tree-based approach	Delay minimisation	Computationally high
Deng et al. (2016)	Online training with minimal data	Minimum previous data used	High complexity
Yang et al. (2016)	Queuing analysis theory	Arbitrary number of MS	High complexity
Restuccia and Das (2016)	Swarm intelligence	Optimal RPs selection	High complexity
Aravind and Chakravarthi (2020)	Fractional rider optimisation	Optimal MS location	Delay increased
Anwit et al. (2020)	Shark smell optimisation	Optimal RPs identified	Longer path for MSs
Keskin and Yiğit (2020)	Mixed-integer linear programming	MS speed control	Longer path
Liu et al. (2020b)	Convex hull and genetic algorithm	Minimal travelling distance	High complexity
Liu et al. (2019)	Dual approximation of anchor points	Best set of RPs	Increased delay
Tao et al. (2020)	Multi-objective joint optimisation	Full-coverage subgraph	High maintenance cost
Koosheshi and Ebadi (2019)	Fuzzy logic-based algorithm	Energy optimised	Not results optimal MSs
Faheem et al. (2019)	A tree-based data forwarding	Energy optimised	Not results optimal MSs
Lakshminarayanan and Rajendran (2019)	Mixed linear integer programming	Energy balanced between nodes	Unnecessary data transmissions
Wen et al. (2018)	Heuristic approach	Improved network lifespan	Not results optimal MSs
Zhu et al. (2018)	MS fault handling approach	Optimal location prediction for MS	High energy drain of SNs
Faheem and Gungor (2018)	Heuristic approach	Determined optimal number of MSs	Longer MS path
Doostali and Babamir (2020)	Probability-based scheduling	Efficient path for MSs	Unbalanced energy among nodes
Srinivas et al. (2020)	Graph theory-based technique	Optimal set of MSs	Failed to estimate optimal location for RPs
Yim et al. (2020)	Virtual tube storage strategy	Optimal utilisation of buffers	Longer MS paths

Table 1 Summary on MSs-based data collection literature (continued)

Author	*Methods/algorithms*	*Advantages*	*Limitations*
Kumar and Kumar (2019)	Location-aware heuristic approach	Delay minimisation	Unbalanced energy among nodes
Peixoto and Costa (2017)	A relevance-based heuristic approach	Optimal RPs selection	Computationally high complexity
Gutam et al. (2021)	Spanning tree and geometry	Optimal RPs and path	Work for small scale WSNs
Al-Kaseem et al. (2021)	Stable election algorithm	Optimise the message exchange	High computational
Vimala and Manikandan (2021)	Dolphin swarm optimisation	Optimal MS path	Computationally high
Khalid et al. (2021)	Consensus-based routing protocol	Improve routing efficiency	Longer MS path
Kharati and Khalily-Dermany (2021)	Tabu search algorithm	Energy optimisation	High complexity
Jain et al. (2021)	Hierarchical clustering	Dynamic path adjustment	Computationally high
Gowda and Jayasree (2021)	Bald eagle search algorithm	Optimal RPs and path selection	Maintenance cost is high
Sapre and Mini (2021)	Moth flame optimisation	Optimal route for MS	High complexity
Thomson et al. (2021)	Mobility pattern prediction	Energy optimisation	High complexity

ACO-based MS scheduling mechanism is presented in Najjar-Ghabel et al. (2019) for data collection in WSNs. In which the Najjar-Ghabel et al. primarily concentrated on energy minimisation and delay avoidance to enhance the efficiency of data gathering process. This work prolong the lifespan of a WSN through minimised energy consumption. ACO-based clustering is presented in Krishnan et al. (2019) for scheduling the MMS in the WSN to gather the data packets from the SNs. This approach performs the clustering operation using ACO and does not focus on determining the MSs count for data gathering efficiently. Farzinvash et al. (2019) distinguish the data into delay-tolerance and delay-sensitive. In contrast, the delay-sensitive data has given high priority to acquire using a a minimum spanning tree towards the base station. The delay-tolerance data are gathered using the MMS. A MMS-based data acquisition approach is present in Deng et al. (2016) for large-scale WSNs using online training approaches using minimal previous information in this work. This work identifies the efficient number of MSs required to fulfill the data collection task and schedule it efficiently. Yang et al. (2016) have presented the MMS-based data collection approach. Here, Yang et al. used an arbitrary count of MSs to get the data packets from the SNs. A swarm-intelligence-based data collection algorithm is presented in Restuccia and Das (2016) for WSNs using MSs. Aravind and Chakravarthi (2020) identify the optimal location to place the MSs to gather the data packets from each SNs in WSNs.

The path construction for MMS has been determined using the meta-heuristic shark smell optimisation technique in Anwit et al. (2020) for WSNs. In this, Anwit et al. find the optimal number of RPs using the efficient computational strategies. This approach also addresses the optimal required MSs to acquire the data from the SNs efficiently.

The MS travelling speed controlled lifetime maximisation algorithm has been proposed in Keskin and Yiğit (2020) for WSNs. However, this approach unable to identify the optimal requirement of MSs and their scheduling strategies. Mobile elements-based data collection for disjoint WSNs using convex hull and genetic algorithms is discussed in Liu et al. (2020b). In this, Liu et al. focused on travelling distance minimisation for MS in the WSNs during the data collection. Still, the lifetime of the network is prolonged in this strategy. Similarly, in Liu et al. (2019) address the latency-aware MMS scheduling for data packet gathering in the disjoint network. Tao et al. (2020) present UAVs -based data gather the mechanism for WSNs. However, it is very costly to use UAVs and manage their energy during the data collection process. A fuzzy-logic-based uneven clustering mechanism is presented in Koosheshi and Ebadi (2019) with focus on energy balance among the nodes in WSN. This work does not considered the optimal number of clusters to minimise MSs for data acquisition in the large-scale networks. In Faheem et al. (2019), an efficient data routing protocols designed for smart grid application using WSNs. In this work, MMS used to gather the data packets from the network. An efficient path construction for MS-based data gathering for WSN using clustering in Lakshminarayanan and Rajendran (2019). In this, the SNs are transmitted their data to the nearest RP and the MSs used to collect data form RPs to reach the data to base station.

Wen et al. (2018) MMS-based efficient data collection mechanism has been proposed for WSNs. In this, Wen et al. present an efficient data gathering mechanism to balance the energy while prolong the network life. However, this approach does not focus to choose the optimal number of MSs. A location predictive and high available data collection algorithm for WSNs is presented in Zhu et al. (2018). The MS location is tracked by the SNs to forward their data packets to the MS. This approach is inefficient because the SNs consume more energy. Faheem et al. proposed a MMS-based data gathering strategy for industrial WSNs in Faheem and Gungor (2018). In this, various performance metrics are evaluated, but this work is not considered to minimises the number of MSs are required in the network with an efficient data acquisition mechanism. In Doostali and Babamir (2020), an efficient RPs selection method has been proposed based on the probability estimation to schedule the MSs during the data collection in WSNs. Srinivas et al. (2020) are determining the requirement of MSs to collect the data packets from the SNs by improving the QoS with cost-effective. Storage-based MS deployment is discussed for WSNs in Yim et al. (2020). Here, the deployment of the MS depends on the data generation at the source level. In Kumar and Kumar (2019), the MSs are controlled depends on their locations during the data collection. This approach minimised the delay of the MSs to reach the SNs while collecting the data. Similarly, in Peixoto and Costa (2017) also focus on location aware MSs scheduling for data acquisition.

Gutam et al. (2021) introduced a spanning tree-based clustering mechanism to identify the optimal visiting points for the MS and a geometric approach to determine the efficient path between them. However, this algorithm is suitable for small-scale WSNs. A stable election algorithm is used in Al-Kaseem et al. (2021) for identifying the required amount of RPs and a path between them. A Dolphin swarm optimisation strategy is used in Vimala and Manikandan (2021) for grid-based WSNs for data collection using MSs. This algorithm can identify the optimal path between the RPs in the network. A consensus-based routing protocol is implemented for data collection WSNs using MSs is presented in Khalid et al. (2021). A Tabu search strategy is used to mitigate the hotspot issue for the WSNs using MSs in Kharati and Khalily-Dermany (2021). This algorithm efficiently optimises the energy, but it is computationally high. Jain et al. (2021) uses hierarchical clustering for data gathering in WSNs with MSs. A bald eagle search algorithm is used to identify the required RPs set along with a path between them in Gowda and Jayasree (2021). A moth flame optimisation algorithm is used for MS-based data acquisition for WSNs (Sapre and Mini, 2021). Thomoson et al. invented a mobility-pattern-based MS location prediction for data gathering for WSNs using MS.

From the above discussion, we identify that majority of the existing works does not consider the optimal number of MSs identification, and their scheduling. In fact, the scheduling algorithms are high computational for the MSs. So, in the proposed work we determine the optimal requirement of MSs to gather the data packets from the SNs efficiently with minimal complexity scheduling strategy. In the proposed work, we separate the MSs into local and global MS. The local MSs communicate with the SNss for data acquisition and global MS for data deposit whereas the global MS can communicate only to the local MS and the base station.

3 System model and problem formulation

The n SNs $S = \{s_1, s_2, ..., s_n\}$ are deployed in an area of size A. The SNs are connected as a graph G if their distance is less than the communication range r_c. The transmission range is denoted using r_t and $r_t \leq r_c$. Every SNs S will forward their data to nearest local MS, i.e., $M = \{m_1, m_2, ..., m_k\}$ where total k local MS and one global MS is used. The i^{th} partition of the network is denoted using P_i, and $1 \leq i \leq k$. The global MS can communicate only to the local MS and the BS (indicated using S_0). The local MSs can communicate only to the SNs and the global MS. The properties of all the SNs are unique, and they are equipped statically. The distances between S and S_0 are maintained in D. Each SN is run with a battery of capacity E_0 initially. The buffer availability of a s_i is indicated as B_i, and initially, it is empty. The two SNs s_i and s_j distance is indicated using d_{ij}. Initially, the whole network G is in a single cluster. While MS collects the data from any SN, we assume that the other SNs also transmit their data if the local MS is in its range. We also assume that the SNs may be connected or isolated (few) nodes. The global MS is equipped with higher storage capacity and a limited chance to overflow. The value of k for the CSO is determined as $k = \log_2 n$. For the reader convenient, the frequently used notations are summarised using a Table 2.

The energy model of the proposed model is as follows. Here, we present the energy consumption (EC) to transmit to the local MS by the node s_i is shown in Eq. (1)

$$E_{tx}(i) = \begin{cases} \alpha_t \mathcal{B} & \text{MS stay at node } s_i \\ \alpha_t \mathcal{B} + \alpha_a \mathcal{B} d_{i0} & \text{Otherwise} \end{cases} \tag{1}$$

where energy for amplification is represented using α_a, energy for processing a bit of data is denoted using α_t and $\mathcal{B}$ bits are transmitted by node i to M_0. The distance from the node s_i to the local MS M_k is denoted using d_{i0}. The d_{i0} is zero when MS visit a particular node s_i. Here, there is a possibility to transmit the data of SNs (other than MS visit node) which are in the transmission range of the local MS.

Table 2 Frequently used notations

Notation	*Meaning*
n	Number of SNs
S	Set of SNs
G	WSN
S_0	Base station
M	Set of MSs
M_0	Global MS
k	Number of local MSs
E_0	Initial available energy
d_{ij}	Distance between SNs i and j
d_{i0}	Node i to S_0
D	Distance matrix of G
$\mathcal{B}$	Number of bits transferred
N	Network lifetime
Υ_i	Distance travelled by global MS
φ	Number of tours completed by global MS
Fit	Fitness function
Fit_t	Fitness for tracing
Fit_s	Fitness for seeking
H_i	Random cluster generated initially
$Prob_i$	Probability function
SMP	Seeking memory pool
SPC	Self-position consideration
SRD	Seeking a range of the selected dimension
λ_i	Velocity update of cat i during tracing
ξ	A constant
ζ	A random value between the [0, 1]
C_i	Cluster set
ρ	Pheromone evaporation
a	Ant
$\tau_{ij}(t)$	Pheromone update at time t
p_{ij}^{a}	Probability to choose next local MS by ant a
α	Relative importance of τ_{ij}
β	Relative importance of d_{ij}

The the network lifetime (N) determine how long the network can operate perfectly with no data losses. In this work, we consider the time until the global MS can travel in the WSN to gather the data before the local networks are isolated. The computation of the N is shown in equation (2)

$$N = \frac{1}{v} \times \sum_{i=0}^{\lfloor \varphi \rfloor} \Upsilon_i \tag{2}$$

where the number of tour completed by global MS is denoted as $\lfloor \varphi \rfloor$ and Υ_i is the global MS travelling distance with the velocity v in terms of m/s from the BS.

The primary goal of the CSOMMS is to prolong the N by deciding the best number of, i.e., $|M| > 1$.

$$\max \; N \tag{3}$$

subjected to $\min |M| > 1$.

4 Proposed work

This section presents the proposed CSOMMS, which is a combination of three parts. Initially, we identify the optimal set for visiting each MS to fulfill the data collection task. Further, we recognise the optimal low computational path for each MS to acquire the data. Later, we find the traversal route of global MS to gather the data packets from the local MSs.

4.1 Determine MS visiting nodes set

Determine the set of nodes that each SN can visit for data collection is an important task. In this context, this section uses the CSO (Santosa and Ningrum, 2009) to partition the network by grouping a set of SNs. Each partition, an MS is scheduled to acquire the data from it. The CSO is initially introduced by Chu et al. (2006). There are several machine learning-based clustering approaches (Praveen Kumar et al., 2019), and meta-heuristic-based clustering approaches such as ACO (Abubakar et al., 2022), PSO (Hasan et al., 2021), etc. Still, CSO is one of the best clustering mechanisms over than and provides the optimal results. In this approach, unlike the traditional CSO algorithm, we did not consider the mixed ratio and considered 100% for the count of dimension to change (CDC) (Santosa and Ningrum, 2009).

The CSO algorithm works mainly by depending on the behaviour of the cat, such as seeking and tracing. Initially, it performs the clustering mechanism, and then it makes the clusters optimal. The algorithm requires the number of clusters k as an input and makes the network into k partitions. The optimal k value in this algorithm is decided on the estimated cost for the project. Depends on the number of available MS, we can start the algorithm. If we have an unlimited number of MS, we can apply the optimal number of cluster selection using Donta et al. (2019). Once the number of clusters k is decided, we partition the network and identify the initial virtual cluster heads (CHs) using a distance function as shown equation (4)

$$d_{ij} = ||s_i - s_j|| = \sqrt{\sum_{i=1}^{n} (s_i - s_j)^2} \; \forall \; j = ((i+1)\%n) \tag{4}$$

During the seeking and tracing process, we need to update the fitness function to optimise the clusters using equation (5)

$$Fit = \sum_{i=1}^{k} \sum_{s \in C_i} (||s - H_i||)^2 \tag{5}$$

The clustering CSO algorithm initially assigned the fitness values to infinity, and during the running, it must be achieved the minimised clustering results. The CSO-based clustering is presented in Algorithm 1. Initially, identify the k CHs randomly from each partition named H_i. It is grouped using the distance function shown in equation (4). Once the random CHs H_i are identified, we perform the *Seeking()* and *Tracing()* operations by keep on updating the *fitness* function.

Algorithm 1 Clustering through CSO

```
1: $Fit_s = Fit_t = \infty$
2: $k$ random points are initialised in the network as a temporary cluster heads $H_i \forall 1 \leq i \leq k$
3: Group the nearest points from the CH to make a cluster using an Euclidean distance using
   equation (4)
4: Calculate the fitness value for the algorithm using equation (5)
5: while $i \leq I_{limit}$ do
6:    while $Fit_s \leq Fit$ do
7:       Operate Seeking(); (Call Algorithm 2)
8:    end while
9:    while $Fit_t \leq Fit$ do
10:      Operate Tracing(); (Call Algorithm 3)
11:   end while
12: end while
```

Algorithm 2 CSO_Seeking()

```
1: Define $SMP, SRD$, and $SPC$
2: for $i = 1$ to $k$ do
3:    $H_i \longrightarrow SMP$
4:    if $SPC == TRUE$ then
5:       $j = SMP - 1$
6:    end if
7:    $SV \longleftarrow SRD * H_i$ //SV indicates shifting value
8: end for
9: for $h = 1$ to $SMP$ do
10:   $SV \pm RandomInt()$
11: end for
12: Use equation (4) to group the data
13: Calculate $Fit_s$ using equation (5)
14: Identify the new cluster head $H_i$
```

4.1.1 Seeking()

During the *Seeking()* the cats are idle (rest position), but keep on observe the environment to search for goal or food. The traditional *seeking()* require four main functionalities, whereas, in this approach, we use only three: seeking memory pool (SMP), self-position consideration (SPC), and seeking a range of the selected dimension (SRD) – the CDC by default considered as 100%, so there is no much importance to this metric. The number of cluster centre copies is represented using SMP, SPC is a Boolean value $\{TRUE, FALSE\}$, and SRD is the ratio of the mutation between [0, 1]. The *seeking()*, initially starts at SMP from the current CH H_i and keep on update the j value depends on the SPC. Next, we compute the mutative ratio using $SRD \times H_i$. Calculate the fitness value of the each H_i using the equation (5).

Once the *fitness* is computed then start at each H_i to adjust or move the H_i to new value. Among the multiple H_i, we identify the one who had the highest probability using the equation (6) to avoid the composition between them.

$$Prob_i = \frac{|Fit_i - Fit_{\max}|}{Fit_{\max} - Fit_{\min}} \; \forall \; 0 < i < j \tag{6}$$

Now, keep on update the fitness value and move the current H_i to new H_i in each iteration and also update the cluster members according to the equation (4). We continue the *Seeking()* process until the Fit_s must be the minimum of previous Fit_s value.

4.1.2 Tracing()

Once the *seeking()* process holds, the *tracing()* will start. During the *tracing()*, the cat react for the targets. The primary goal of the tracking is to shift the CH from one location to other, which is optimal. The *Tracing* updates the velocity of the cat i (H_i) as shown in equation (7).

$$\lambda_i = \lambda_i + (H_{best} - H_i) \times \zeta \times \xi \tag{7}$$

where ξ is a constant and ζ is a random value between the [0, 1]. H_{best} is the best position achieved through minimum *fitness* value. Next, update the H_i position from current position to the optimal position using equation (8).

$$H_i = H_i + \lambda_i \tag{8}$$

We iterate the process for each H_i where $1 \leq i \leq k$ until the *fitness* value of Tracing results minimum.

We achieve the optimal partitions set while iterating the CSO algorithm. These partitions are the inputs to the local MS trajectory construction (MSTC) algorithm.

Algorithm 3 CSO_Tracing()

```
1: for $i = 1$ to $k$ do
2:     Update $\lambda_i$ using equation (7)
3:     Update pos(i); using equation (8)
4:     $H_i$ = new($H_i$)
5: end for
6: Use equation (4) to group the data
7: Calculate $Fit_t$ using equation (5)
```

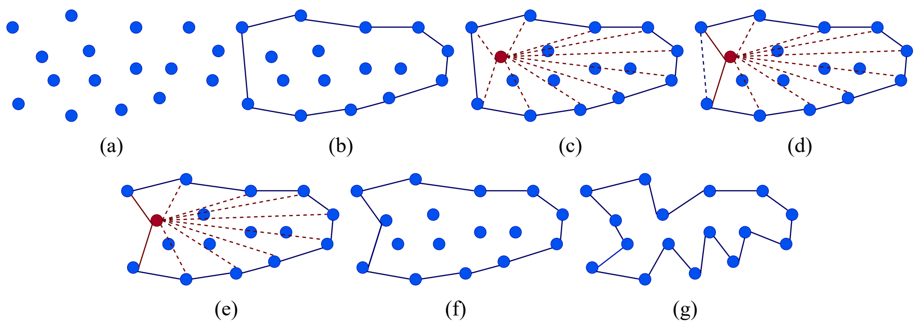
4.2 Local MSTC

From Subsection 4.1, we partition the network in to several parts. In this section, we assign a MS to each partition of the network and construct an optimal Hamiltonian cycle for MS traversal to acquire the data from the SNs. However, the swarm intelligence algorithms need more number of iterations but the local MS traversal path takes the less computations using a simple geometric algorithms. So, the path construction for the local MS is shown in Algorithm 4.

Algorithm 4 Local MSTC

```
INPUT: $C_i \ \forall \ 1 \leq i \leq k$
OUTPUT: Visiting order $X$
 1: Choose the rightmost least point $q_0$ from $C_i$
 2: Sort($C_i - q_0$) angularly about $q_0$ and label $q_1, q_2, ...., q_{n-1}$
 3: PUSH $q_0$ and $a_{n-1}$ to a STACK
 4: for $i = 1$ to $n - 1$ do
 5:     Construct a line from top two entries of the stack i.e., $q_0$ and $q_{n-1}$
 6:     if $q_i$ is on top of the line then
 7:         PUSH $q_i$ to STACK
 8:     else
 9:         POP STACK
10:     end if
11: end for
12: $X \leftarrow$ elements(STACK) //Same order
13: $Y \leftarrow C_i -$ elements(STACK)
14: $Z \leftarrow$ edgeSet($C_i$)
15: repeat
16:     Choose closest $v_i$ to $X$, from $Y$
17:     Form the edges incidence with $v_i$ and $X$ (let us assume $u_j$, $u_{j+1} \in A$)
18:     Find the longest angle at $v_i$ (assume that the longest angle created at $v_i u_j$ and $v_i u_{j+1}$),
        then consider $v_i u_j$ and $v_i u_{j+1}$ edge and remove all other edges including $u_j u_{j+1}$
19:     Move $v_i \longrightarrow X$
20:     $Y = Y - v_i$
21: until $Y = \phi$
```

Figure 1 Illustration through an example for local MSTC, (a) cluster C_i (b) convex hull of C_i (c) find the angles from node v_i to all other nodes of X (d) longest angle nodes (e) connect with u_j and u_{j+1} and also remove the edge $u_j u_{j+1}$ (f) remove all other edges (g) final solution (see online version for colours)



Here, we explain Algorithm 4 along with an illustration through an example for better understanding of the proposed local MSTC. Consider a partition (C_i) from the network as shown in Figure 1(a). Initially, we construct a convex hull for the given set of SNs in each partition C_i. From Algorithm 4, lines 1–11 represent the construction of the convex hull. It reflects in Figure 1(b). Once the convex hull (X) is created, we applied a computational geometry approach to decide the optimal visiting order, and it represents in lines 12–21 from Algorithm 4. Figure 1(c), we consider one of the non-convex hull points (from Y) and find the angles from each convex hull adjacent point. Identify the longest angle from it. From Figure 1(d), the brown lines represent the longest angle among the others. So, we remove the edge between u_j and u_{j+1} and make an edge between the $v_i u_j$ and $v_i u_{j+1}$ as shown in Figure 1(e). Remove all the remaining edges as shown in Figure 1(f) and v_i is included in the path. Now, we iterate the same process for remaining nodes and get the absolute path for a local MS as shown in Figure 1(g).

4.3 Global MS path planning

The global MS path is dynamic because it must construct the path according to the movement of the local MS. So, an efficient path-finding algorithm is necessary to fulfill the traversing task. In this context, we consider Ant system algorithm (Dorigo and Gambardella, 1997) whereas several advantages of using it for path construction in the dynamic environments. The proposed global MS path construction is summarised using Algorithm 5.

Algorithm 5 Global MS path construction

```
INPUT: Locations of local MS
OUTPUT: Optimal visiting order
 1: Initialise values to $\alpha$, and $\beta$
 2: Initialise value to $\rho$
 3: repeat
 4:    for $a = 1$ to $\mathcal{A}$ do
 5:       Compute $\eta$ using equation (10)
 6:       Find the $p_{ij}^a$ using equation. (9)
 7:       Select the node $i$ which is max($p_{ij}^a$) value
 8:       Update $\tau_{ij}(t)$ value using equation (11)
 9:    end for
10:    $I_{\max} = I_{\max} - 1$
11: until $I_{\max} > 0$
```

Algorithm 5 runs iteratively to produce the optimal solution. In this, we consider a set of virtual ants $\mathcal{A}$, where they start different and random places initially and construct a unique solution after a particular set of iterations. Each ant a uses a probabilistic function to choose the next visiting local MS (m_j) from m_i as defined in equation (9).

$$p_{ij}^a = \frac{\tau_{ij}^\alpha \times \eta_{ij}^\beta}{\sum\limits_{f \in N_i^a} \tau_{if}^\alpha \times \eta_{if}^\beta} \forall f \in N_i^a \tag{9}$$

$$\eta_{ij} = \frac{1}{d_{ij}} \tag{10}$$

where η_{ij} is the inverse distance of the local MS i to j represented as shown in equation (10). It is also treated as the heuristic information for the η. The α and β are the relative importance of the τ and η where they are ranging between the [0, 1). The N_i^a is the feasible un-visited neighbour MSs of the i^{th} MS. The τ_{ij} is treated as the pheromone updation and it is considered as shown in equation (11).

$$\tau_{ij}(t) = (1 - \rho)\tau_{ij}(t-1) + \sum_{a=1}^{w} \overline{\tau_{ij}^a}(t-1) \tag{11}$$

where evaporation of the pheromone is represented using $\rho - (0 < \rho < 1)$. The $\overline{\tau_{ij}^{a}}(t-1)$ is denoted as the pheromone updated by ant a during the previous visit, and it is computed as shown in equation (12).

$$\overline{\tau_{ij}^{a}}(t-1) = \begin{cases} \frac{1}{L^{a}(t-1)} & \text{if } a \text{ traversed from } i \text{ to } j \\ 0 & \text{Otherwise} \end{cases} \tag{12}$$

Here, $L^{a}(t-1)$ is the trajectory determined by an ant a during the previous visit, i.e., $(t-1)$. In case it is not visited between the i and j, then we can consider it as a *zero*. The BS iterated the above steps to identify the optimal visiting order through the shortest distance by covering all the local MS by the global MS. The global MS starts at BS to visit all local MS and get data to return the BS. The ACO returns the near-optimal results compared to the traditional TSP technique, and the experimental results justify it.

4.4 Computational complexity

The computational complexity of the proposed work involve in three parts. Initially, the network is disjointed into clusters using the CSO. The time require to split the network using CSO is $O(n^3)$, where n is the total deployed SNs in the WSN. Next, the computational time to traverse the local MS is $O(n^2)$. Finally, the time require to identify the global MS trajectory is $O(1/\rho(mnq \log n))$ (Kumar et al., 2018). So, the asymptotic complexity for the CSOMMS algorithm is $O(n^3) + O(n^2) + O(1/\rho(mnq \log n)) \approx O(n^3)$.

5 Experiment results and numeral analysis

The experiments are conducted through simulator using Python. We considered various scenarios and different metrics to justify the QoS of the proposed algorithm. In this context, we compared our proposed CSOMMS model using existing algorithms such as IARP (Moussa and El Alaoui, 2021), CORL2 (Wen et al., 2021) and ACOMMS (Krishnan et al., 2019). The network data generated using Sah et al. (2021). The size of the network is considered by varying the number of sensors between 200 to 500 in an area of size 500 m^2. We considered two scenarios including event driven and non-event driven. We assume that all the deployed SNs are static and they are not moving once they deployed. We also do not considered any obstacles in the network during the simulation study. The communication and transmission range of the nodes are considered as 35 m and 25 m, respectively. The data transmissions are follows the mesh topology and during evaluations it uses tree topology. The local MSs travel on a fixed path whereas no restrictions on the global MS. The sojourn time of the local and global MS are sufficient to acquire the data from the SNs or the MSs. The travelling speed of both the MS are fixed as 1 m/sec. The remaining metrics are summarised using Table 3.

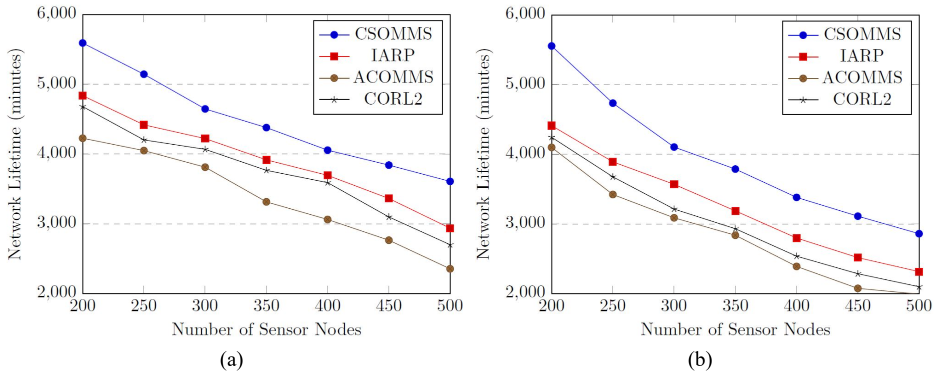
Table 3 Performance metrics used

Metric	*Value*
n	200–500
Area size	500 m^2
Packet size	30 bytes
α_t	42 mJ
α_r	29 mJ
Data transmission rates	80–250 kbps
E_0	100 J
MAC Protocol	TDMA
Velocity of MS	1 m/s
Communication range	35 m
Transmission range	25 m
Topology	Tree
Simulator used	Python

5.1 Network lifetime

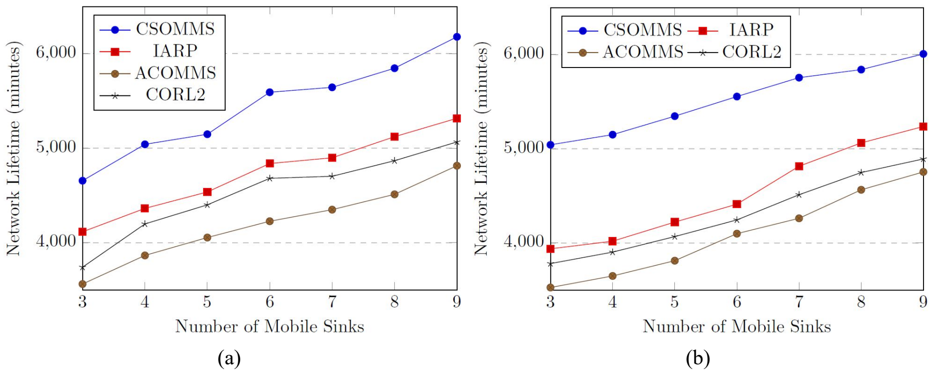
In our work, we consider the amount of time the global MS able to acquire the data from the local MS until the clusters are isolated as shown in equation (2).

Figure 2 Network lifetime vs. #SNs, (a) event-driven (b) non-event-driven (see online version for colours)



We evaluate the N of the CSOMMS and IARP, CORL2, and ACOMMS algorithms with different SNs set under teh event and non-event driven scenarios are as shown in Figures 2(a) and 2(b), respectively. In Figure 2(a), we observe the reduced NL while increasing the number of nodes. But, the proposed CSOMMS result in best over the existing IARP, CORL2, and ACOMMS algorithms approximately 19–22%, 21–27% and 25–32%, respectively. In the non-event-driven scenarios, we notice the lifetime which prolongs approximately 14–21% than IARP, 19–28% than CORL2, and 22–33% than ACOMMS algorithms. The even-driven results are more better over the non-event driven, since the proposed work performs better in non-event driven as well. The higher the QoS of the proposed work is due to identify the optimal requirement of MS and also their scheduling strategy.

Figure 3 Network lifetime vs. #MSs, (a) event-driven (b) non-event-driven (see online version for colours)



However, the proposed algorithm resulting in the best number of MSs to acquire the data packets from the nodes, so it is also necessary to evaluate the performance changes when increasing the MSs count. We also assess the N while increasing and decreasing the MSs. Figure 3 shows the comparisons of the CSOMMS and IARP, CORL2, and ACOMMS algorithms by varying the MSs count in a WSN of 500 SNs. Here, we also evaluate the performance in both continuous and burst conditions, and the results are plotted in Figures 3(a) and 3(b), respectively. From Figure 3(a), the N of the CSOMMS is always better, and increasing the number of MS also increasing the lifetime. Similarly, the N of the CSOMMS is even outperforming the existing approaches in the burst scenarios by increasing the number of MSs. Here, we observe that the lifetime of network increases when increasing the MS count. Simultaneously it also increases the maintenance cost. So, the proposed work not only focuses on the lifetime increase but also focus on cost minimisation.

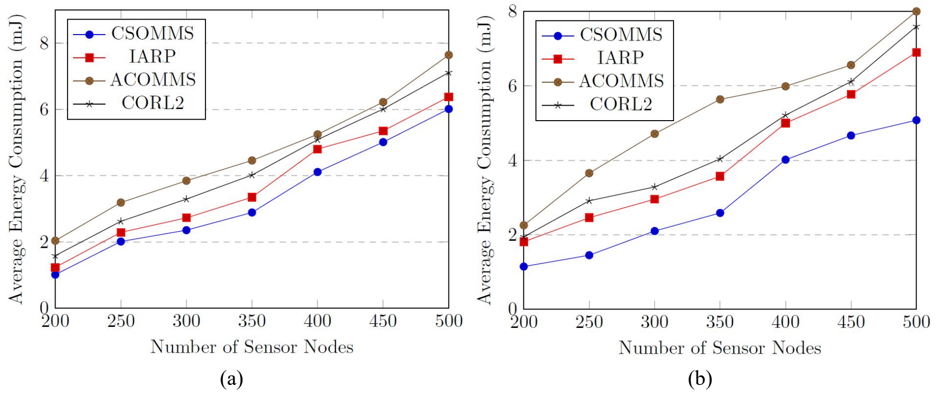
5.2 Average energy consumption

The AEC of the proposed and existing algorithms of this article are considered as the energy drained by each and every SNs during one complete trajectory of the global MS and it is computed using equation (13)

$$E_a = \frac{\sum_{i=1}^{n} E_i}{n} \tag{13}$$

The E_a of the CSOMMS is evaluated in event and non-event driven applications and compared in Figures 4(a) and 4(b), respectively. From Figure 4(a), we observe the improvement of the proposed CSOMMS approximately 19%, 25%, 29% compared to IARP, CORL2, and ACOMMS, respectively. In the non-event-driven scenario, the improvements are noticed approximately 17%, 21%, and 27%, respectively, in IARP, CORL2, and ACOMMS algorithms. The optimal results were achieved due to the best selection of the number of MS and also their scheduling.

Figure 4 Average energy consumption, (a) event-driven (b) non-event-driven (see online version for colours)

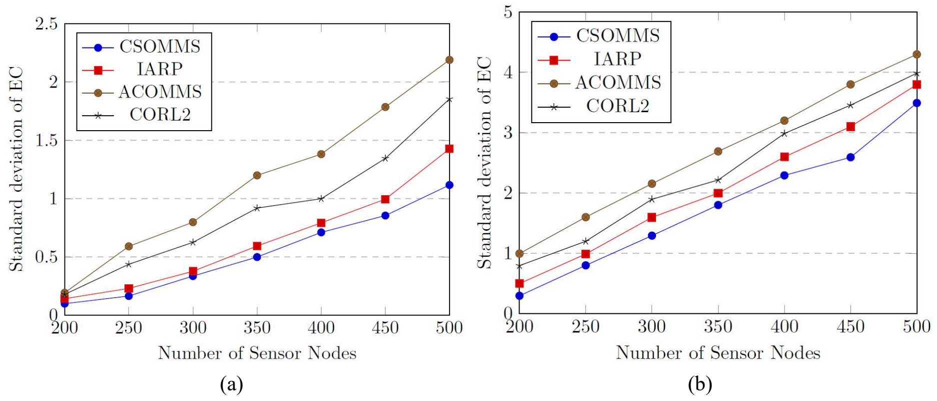


5.3 Standard deviation of energy consumption

The standard deviation of the proposed and existing works are compared as per the Kumar et al. (2018), and rendering again here for reader convenient.

$$\sigma_e = \sqrt{\frac{\sum_{i=1}^{n}(E_i - E_a)^2}{n}} \tag{14}$$

Figure 5 Standard deviation of EC, (a) event-driven (b) non-event-driven (see online version for colours)



The proposed CSOMMS and exiting algorithms SD_e are plotted in Figure 5. The σ_e in the event-driven and non-event-driven scenarios are evaluated in Figures 5(a) and 5(b), respectively. We notice the lower SD of the proposed work approximately $8\times$, $13\times$, and $19\times$ for the IARP, CORL2, and ACOMMS, respectively. Further, we evaluated the σ_e of the CSOMMS and IARP, CORL2, and ACOMMS algorithms in Figure 5(b). The improved performance of the proposed algorithms over the existing IARP, CORL2 and

ACOMMS algorithms are 9%, 17%, and 19%, respectively. These improvements are noticed because of the optimal scheduling of the local and global MSs. It is also because of the optimal number of the MS selections.

5.4 Latency

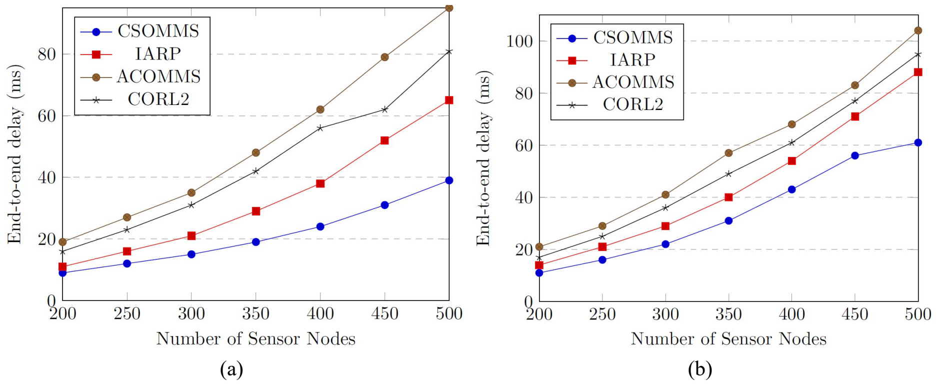
The time variation from a data packet generated at SN and it reach to the BS is considered as latency (Δ). It is the combination of the time the SNs hold until the local MS reach to the node, the time the local MS hold until the global MS reach it, and the travelling time of the MS from the local MS to the sink. The average latency of the proposed CSOMMS is determined using equation (15)

$$\Delta = \sum_{i=1}^{n} \left(\sum_{j=1}^{|s_i|} \left(\frac{\Delta_1(p_j) + \Delta_2(p_j) + \Delta_3(p_j)}{|s_i| \times n} \right) \right) \tag{15}$$

where

- Δ_1 – delay of a packet (p_j) at SN (s_i)
- Δ_2 – waiting time of a packet at the local MS
- Δ_3 – latency of a packet (p_j) spent at global MS before transmit it to the BS.

Figure 6 End-to-end delay vs. #SNs, (a) event-driven (b) non-event-driven (see online version for colours)



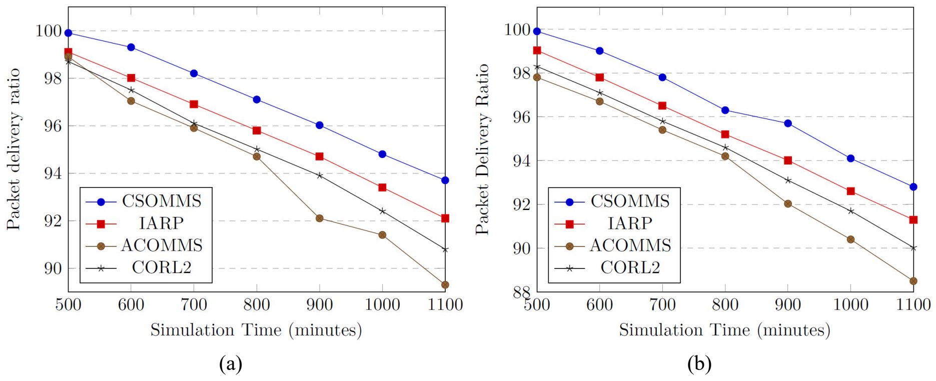
The latency of the CSOMMS is tested with the IARP, CORL2, and ACOMMS algorithms in both event-driven and non-event-driven approaches in Figure 6. The proposed CSOMMS algorithm results in 27× better performance than IARP, 37× better than CORL2, and 53× better than ACOMMS algorithms. Further, we evaluate in non-event-driven application in Figure 6(b), and we notice the improvements of IARP, CORL2, and ACOMMS algorithms 19×, 23× and 48×, respectively, better performance in the proposed CSOMMS algorithm. We notice that the latency of the CSOMMS is always lower compared to existing works in both scenarios. The better results are achieved due to efficient scheduling of the local and global MS with optimal requirement of MS for data acquisition.

5.5 Packet delivery ratio

The PDR of the CSOMMS is considered as the ratio of total generated packets by all the SNs $P(S_i)$ in the network and the received count of packets by the BS ($\mathcal{R}$) during the simulation time from 0 to $\mathcal{T}$. It is determined using equation (16).

$$PDR = \frac{\mathcal{R}}{\sum_{i=0}^{n} P(S_i)} \tag{16}$$

Figure 7 Packet delivery ratio vs. simulation time, (a) event-driven (b) non-event-driven (see online version for colours)



The comparison results of proposed and existing algorithms in terms of PDR are shown in Figure 7. Figure 7(a) plots the comparison of PDR in continuous scenario WSNs and Figure 7(b) shows the comparison of burst scenario. From Figure 7(a), we noticed that the PDR of the CSOMMS is alway better compared to existing IARP, ACOMMS, and CDCA approximately 9%, 11% and 16%, respectively in the continuous scenario. Similarly, in the burst case the performance improvements observed as 11% better than IARP, 15% better over ACOMSS, and 19% better over CDCA algorithms. The PDR improvement of the CSOMMS is noticed because of the efficient MMS scheduling in the WSNs environment.

5.6 Throughput

The throughput (τ) of the network is computed using equation (17).

$$\tau = \frac{\mathcal{R}}{\mathcal{T}} \tag{17}$$

The τ of the existing and proposed works are compared in event-driven and non-event-driven applications of WSNs in Figures 8(a) and 8(b), respectively. The proposed work improves the throughput over the existing works IARP, CORL2, and ACOMMS are approximately 9×, 12×, and 16×, respectively in the event-driven applications. Further, non-event-driven applications are evaluated in Figure 8(b). Here, we noticed the improved QoS of the CSOMMS over the existing IARP, CORL2, and ACOMMS approximately 10×, 14×, and 18×, respectively. So, from both the scenarios, we notice the best performance, approximately 91% in the event-driven applications and 92% in the non-event-driven applications. These improvements are noticed because of the best set of MSs and their optimal scheduling.

Figure 8 Throughput vs. simulation time, (a) event-driven (b) non-event-driven (see online version for colours)

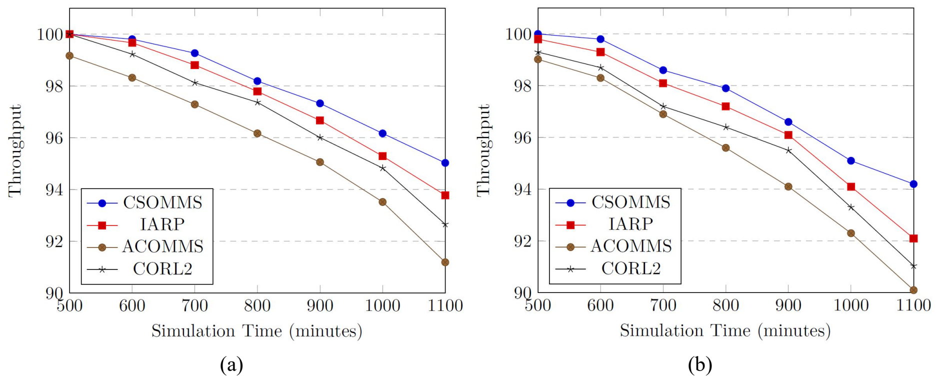
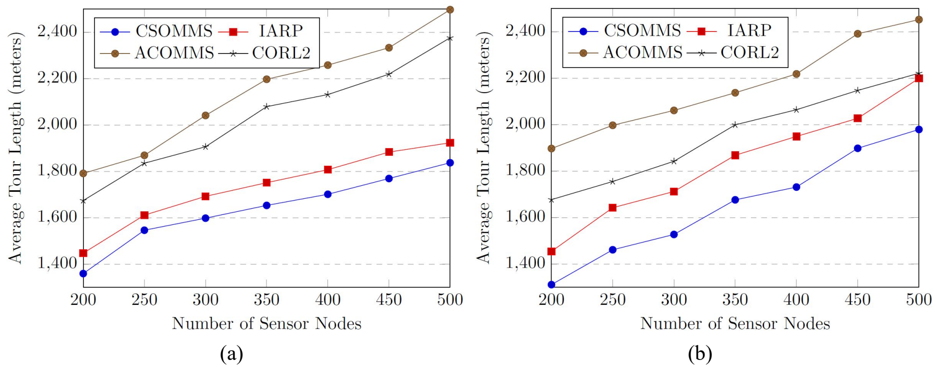


Figure 9 Average tour length vs. #SNs, (a) event-driven (b) non-event-driven (see online version for colours)



5.7 *Average tour length of global MS*

The trajectory length constructed by the global MS (Υ_i) and the number of tours completed ($\lfloor\delta\rfloor$) during the data collection is treated as the average tour length of the global MS. It is calculated as shown in equation (18).

$$ATL = \frac{\sum_{i=0}^{\lfloor\delta\rfloor} \Upsilon_i}{\lfloor\delta\rfloor} \tag{18}$$

The ATL of the CSOMMS and existing IARP, CORL2, and ACOMMS algorithms are compared under the event-driven and non-event-driven scenarios in Figure 9. From

Figure 9(a), we notice the QoS grow of the CSOMMS in the event-driven scenario, and the proposed CSOMMS improves 9× over the IARP, 13× than CORL2 and 18× than ACOMMS algorithms. Similarly, in non-event driven applications, the CSOMMS algorithms result in 11× better performance than IARP, 16× better performance over the CORL2, and 23× better performance over the ACOMMS algorithms. These improvements are noticed because of the optimal number of MS and their traversal routes.

6 Conclusions

This article proposes an MMS-based data collection in large-scale WSNs using a cat-swarm optimisation mechanism. Initially, this algorithm partitions the network using CSO and assigns an MS to each partition. Each assigned MS travelled in a fixed path constructed using a lightweight geometric approach for data gathering. Then, a global MS traverse in the network to acquire the local MS's data and hand it to the BS timely. The global MS path is dynamic and constructed using a popular ACO algorithm. This path is constructed according to the positions of the local MS without disturbing them. The performance of the CSOMMS algorithm is simulated and tested using various simulation runs and compared using the existing algorithms. It outperforms them in multiple metrics, including the lifetime, delay, throughput, PDR, and energy efficiency. The major limitation of the proposed approach is the find optimal k value. Hence, in the future, we focus on identifying the best value for k to collect data packets in WSNs.

References

Abubakar, H., Muhammad, A. and Bello, S. (2022) 'Ants colony optimization algorithm in the hopfield neural network for agricultural soil fertility reverse analysis', *Iraqi Journal For Computer Science and Mathematics*, Vol. 3, No. 1, pp.32–42.

Al-Kaseem, B.R., Taha, Z.K., Abdulmajeed, S.W. and Al-Raweshidy, H.S. (2021) 'Optimized energy efficient path planning strategy in WSN with multiple mobile sinks', *IEEE Access*, Vol. 9, pp.82833–82847.

Anwit, R., Tomar, A. and Jana, P.K. (2020) 'Tour planning for multiple mobile sinks in wireless sensor networks: a shark smell optimization approach', *Applied Soft Computing*, December, Vol. 97, p.106802.

Aravind, A.R. and Chakravarthi, R. (2020) 'Fractional rider optimization algorithm for the optimal placement of the mobile sinks in wireless sensor networks', *International Journal of Communication Systems*, Vol. 34, No. 4, p.e4692.

Banoth, S.P.R., Donta, P.K. and Amgoth, T. (2021) 'Dynamic mobile charger scheduling with partial charging strategy for wsns using deep-q-networks', *Neural Computing and Applications*, Vol. 33, pp.15267–15279.

Chu, S-C., Tsai, P-W. and Pan, J-S. (2006) 'Cat swarm optimization', in *Pacific Rim International Conference on Artificial Intelligence*, Springer, pp.854–858.

Deng, R., He, S. and Chen, J. (2016) 'An online algorithm for data collection by multiple sinks in wireless-sensor networks', *IEEE Transactions on Control of Network Systems*, Vol. 5, No. 1, pp.93–104.

Donta, P.K., Rao, B.S.P., Amgoth, T., Annavarapu, C.S.R. and Swain, S. (2019) 'Data collection and path determination strategies for mobile sink in 3D WSNs', *IEEE Sensors Journal*, Vol. 20, No. 4, pp.2224–2233.

Donta, P.K., Amgoth, T. and Annavarapu, C.S.R. (2020) 'An extended ACO-based mobile sink path determination in wireless sensor networks', *Journal of Ambient Intelligence and Humanized Computing*, Vol. 12, pp.8991–9006.

Doostali, S. and Babamir, S.M. (2020) 'An energy efficient cluster head selection approach for performance improvement in network-coding-based wireless sensor networks with multiple sinks', *Computer Communications*, December, Vol. 164, pp.188–200.

Dorigo, M. and Gambardella, L.M. (1997) 'Ant colony system: a cooperative learning approach to the traveling salesman problem', *IEEE Transactions on Evolutionary Computation*, Vol. 1, No. 1, pp.53–66.

Faheem, M. and Gungor, V.C. (2018) 'MQRP: Mobile sinks-based QoS-aware data gathering protocol for wireless sensor networks-based smart grid applications in the context of Industry 4.0-based on internet of things', *Future Generation Computer Systems*, May, Vol. 82, pp.358–374.

Faheem, M., Butt, R.A., Raza, B., Ashraf, M.W., Ngadi, M.A. and Gungor, V.C. (2019) 'Energy efficient and reliable data gathering using internet of software-defined mobile sinks for WSNs-based smart grid applications', *Computer Standards & Interfaces*, October, Vol. 66, p.103341.

Farzinvash, L., Najjar-Ghabel, S. and Javadzadeh, T. (2019) 'A distributed and energy-efficient approach for collecting emergency data in wireless sensor networks with mobile sinks', *AEU-International Journal of Electronics and Communications*, October, Vol. 108, pp.79–86.

Gowda, C.S. and Jayasree, P. (2021) 'Rendezvous points based energy-aware routing using hybrid neural network for mobile sink in wireless sensor networks', *Wireless Networks*, Vol. 27, No. 4, pp.2961–2976.

Gutam, B.G., Donta, P.K., Annavarapu, C.S.R. and Hu, Y-C. (2021) 'Optimal rendezvous points selection and mobile sink trajectory construction for data collection in WSNs', *Journal of Ambient Intelligence and Humanized Computing*, pp.1–12 [online] https://doi.org/10.1007/s12652-021-03566-2.

Hasan, R.A., Shahab, S.N. and Ahmed, M.A. (2021) 'Correlation with the fundamental PSO and PSO modifications to be hybrid swarm optimization', *Iraqi Journal For Computer Science and Mathematics*, Vol. 2, No. 2, pp.25–32.

Hojjatinia, H., Jahanshahi, M. and Shehnepoor, S. (2021) 'Improving lifetime of wireless sensor networks based on nodes' distribution using Gaussian mixture model in multi-mobile sink approach', *Telecommunication Systems*, Vol. 77, pp.255–268.

Jain, S.K., Venkatadari, M., Shrivastava, N., Jain, S. and Verma, R.K. (2021) 'NHCDRA: a non-uniform hierarchical clustering with dynamic route adjustment for mobile sink based heterogeneous wireless sensor networks', *Wireless Networks*, Vol. 27, No. 4, pp.2451–2467.

Keskin, M.E. and Yiğit, V. (2020) 'Maximizing the lifetime in wireless sensor networks with multiple mobile sinks having nonzero travel times', *Computers & Industrial Engineering*, October, Vol. 148, p.106719.

Khalid, N.A., Bai, Q. and Al-Anbuky, A. (2012) 'Distributed consensus-based routing protocol with multiple mobile sinks support for wireless sensor network', *IET Wireless Sensor Systems*, Vol. 11, No. 3, pp.131–145.

Kharati, E. and Khalily-Dermany, M. (2021) 'Determination of the multicast optimal route for mobile sinks in a specified deadline using network coding and Tabu search algorithm in wireless sensor networks', *Iranian Journal of Science and Technology, Transactions of Electrical Engineering*, Vol. 45, No. 2, pp.447–459.

Koosheshi, K. and Ebadi, S. (2019) 'Optimization energy consumption with multiple mobile sinks using fuzzy logic in wireless sensor networks', *Wireless Networks*, Vol. 25, No. 3, pp.1215–1234.

Krishnan, M., Yun, S. and Jung, Y.M. (2019) 'Enhanced clustering and ACO-based multiple mobile sinks for efficiency improvement of wireless sensor networks', *Computer Networks*, September, Vol. 160, pp.33–40.

Kumar, V. and Kumar, A. (2019) 'Improving reporting delay and lifetime of a WSN using controlled mobile sinks', *Journal of Ambient Intelligence and Humanized Computing*, Vol. 10, No. 4, pp.1433–1441.

Kumar, P., Amgoth, T. and Annavarapu, C.S.R. (2018) 'ACO-based mobile sink path determination for wireless sensor networks under non-uniform data constraints', *Applied Soft Computing*, August, Vol. 69, pp.528–540.

Lakshminarayanan, R. and Rajendran, P. (2019) 'Efficient data collection in wireless sensor networks with block-wise compressive path constrained sensing in mobile sinks', *Cluster Computing*, pp.1–12.

Liu, X., Qiu, T., Zhou, X., Wang, T., Yang, L. and Chang, V. (2019) 'Latency-aware path planning for disconnected sensor networks with mobile sinks', *IEEE Transactions on Industrial Informatics*, Vol. 16, No. 1, pp.350–361.

Liu, X., Obaidat, M.S., Lin, C., Wang, T. and Liu, A. (2020a) 'Movement-based solutions to energy limitation in wireless sensor networks: state of the art and future trends', *IEEE Network*, Vol. 35, No. 2, pp.188–193.

Liu, X., Lin, P., Liu, T., Wang, T., Liu, A. and Xu, W. (2020b) 'Objective-variable tour planning for mobile data collection in partitioned sensor networks', *IEEE Transactions on Mobile Computing*, Vol. 21, No. 1, pp.239–251.

Moussa, N. and El Alaoui, A.E.B. (2021) 'IARP: An intelligent aco-based routing protocol with multiple mobile sinks support for wireless sensor networks', *Wireless Personal Communications*, Vol. 120, pp.545–563.

Najjar-Ghabel, S., Farzinvash, L. and Razavi, S.N. (2019) 'HPDMS: high-performance data harvesting in wireless sensor networks with mobile sinks', *The Journal of Supercomputing*, Vol. 76, pp.2748–2776.

Najjar-Ghabel, S., Farzinvash, L. and Razavi, S.N. (2020) 'Mobile sink-based data gathering in wireless sensor networks with obstacles using artificial intelligence algorithms', *Ad Hoc Networks*, September, Vol. 106, p.102243.

Peixoto, J.P.J. and Costa, D.G. (2017) 'Wireless visual sensor networks for smart city applications: a relevance-based approach for multiple sinks mobility', *Future Generation Computer Systems*, November, Vol. 76, pp.51–62.

Praveen Kumar, D., Tarachand, A. and Rao, A.C.S. (2019) 'Machine learning algorithms for wireless sensor networks: a survey', *Information Fusion*, September, Vol. 49, pp.1–25.

Restuccia, F. and Das, S.K. (2016) 'Optimizing the lifetime of sensor networks with uncontrollable mobile sinks and QoS constraints', *ACM Transactions on Sensor Networks (TOSN)*, Vol. 12, No. 1, pp.1–31.

Sah, D.K., Kumar, D.P., Shivalingagowda, C. and Jayasree, P. (2019) '5G applications and architectures', in *5G Enabled Secure Wireless Networks, Chapter 2: 5G Applications and Architectures*, pp.45–68, Springer.

Sah, D.K., Donta, P.K. and Amgoth, T. (2021) 'EDGF: empirical dataset generation framework for wireless network networks', *Computer Communications*, December, Vol. 180, pp.48–56.

Santosa, B. and Ningrum, M.K. (2009) 'Cat swarm optimization for clustering', in *2009 International Conference of Soft Computing and Pattern Recognition*, IEEE, pp.54–59.

Sapre, S. and Mini, S. (2021) 'A differential moth flame optimization algorithm for mobile sink trajectory', *Peer-to-Peer Networking and Applications*, Vol. 14, No. 1, pp.44–57.

Singh, S.K. and Kumar, P. (2020) 'A comprehensive survey on trajectory schemes for data collection using mobile elements in WSNs', *Journal of Ambient Intelligence and Humanized Computing*, Vol. 11, No. 1, pp.291–312.

Srinivas, M., Donta, P.K. and Amgoth, T. (2020) 'Finding the minimum number of mobile sinks for data collection in wireless sensor networks', in *2020 IEEE International Conference on Communication, Networks and Satellite (Comnetsat)*, IEEE, pp.256–260.

Tao, M., Li, X., Yuan, H. and Wei, W. (2020) 'UAV–aided trustworthy data collection in federated-WSN-enabled IoT applications', *Information Sciences*, September, Vol. 532, pp.155–169.

Thomson, C., Wadhaj, I., Tan, Z. and Al-Dubai, A. (2021) 'Towards an energy balancing solution for wireless sensor network with mobile sink node', *Computer Communications*, March, Vol. 170, pp.50–64.

Vimala, D. and Manikandan, K. (2012) 'Grid based energy efficient routing using dual mobile sinks', *Arabian Journal for Science and Engineering*, pp.1–10 [online] https://doi.org/10.1007/s13369-021-06134-0.

Wen, W., Chang, C-Y., Zhao, S. and Shang, C. (2018) 'Cooperative data collection mechanism using multiple mobile sinks in wireless sensor networks', *Sensors*, Vol. 18, No. 8, p.2627.

Wen, W., Chang, C-Y., Wu, S-J. and Roy, D.S. (2021) 'CORL2: exploring cooperative opportunities, reducing latency and prolonging network lifetime for data collection using mobile sinks in wireless sensor networks', *IEEE Sensors Journal*, Vol. 21, No. 17, pp.19596–19610.

Yang, S., Adeel, U., Tahir, Y. and McCann, J.A. (2016) 'Practical opportunistic data collection in wireless sensor networks with mobile sinks', *IEEE Transactions on Mobile Computing*, Vol. 16, No. 5, pp.1420–1433.

Yim, Y., Mo, H-S., Kim, C., Kim, S-H., Leung, V.C. and Lee, E. (2020) 'Virtual tube storage scheme for supporting mobile sink groups in wireless sensor networks', *Computer Communications*, June, Vol. 159, pp.245–257.

Zhu, C., Quan, K., Han, G. and Rodrigues, J.J. (2018) 'A high-available and location predictive data gathering scheme with mobile sinks for wireless sensor networks', *Computer Networks*, November, Vol. 145, pp.156–164.