Lebanese American University Repository (LAUR)

Conference

Publication metadata

Title: Data Collection in Wireless Sensor Networks Using UAV and Compressive Data Gathering

Author(s): Dariush Ebrahimi, Sanaa Sharafeddine, Pin-Han Ho, Chadi Assi

Conference title: 2018 IEEE Global Communications Conference (GLOBECOM)

DOI: https://doi.org/10.1109/GLOCOM.2018.8647924

Handle: http://hdl.handle.net/10725/10575

How to cite this post-print from LAUR:

Ebrahimi, D., Sharafeddine, S., Ho, P. H., & Assi, C. (2018, December). Data Collection in Wireless Sensor Networks Using UAV and Compressive Data Gathering. Paper presented at the 2018 IEEE Global Communications Conference (GLOBECOM), DOI: 10.1109/GLOCOM.2018.8647924, http://hdl.handle.net/10725/10575

© Year 2018

This Open Access post-print is licensed under a Creative Commons Attribution-Non Commercial-No Derivatives (CC-BY-NC-ND 4.0)

This paper is posted at LAU Repository

For more information, please contact: archives@lau.edu.lb

Data Collection in Wireless Sensor Networks using UAV and Compressive Data Gathering

Dariush Ebrahimi, Sanaa Sharafeddine, Pin-Han Ho, Chadi Assi

Abstract—**Fifth generation wireless networks are expected to provide advanced capabilities and create new markets spanning a wide range of use cases. Among these, massive IoT is standing out with the proliferation of sensors and wearable devices that continuously monitor and transmit data for further processing. This paper proposes a novel data collection technique using Unmanned Aerial Vehicles (UAVs) in dense wireless sensor networks (WSNs) using projection-based Compressive Data Gathering (CDG) as a solution methodology. CDG is utilized to aggregate data en route from sets of sensor nodes to a set of projection nodes (heads) in order to notably reduce the number of transmissions leading to energy savings and extended WSN lifetime. The UAVs forward the gathered data from heads to a remote sink to enhance efficiency by avoiding long range transmissions from heads to the sink or multi-hop communications among sensors to the sink. We formulate a joint optimization problem that captures clustering, heads selection, routing trees construction, and UAV trajectory planning. In order to overcome the complexity of the joint optimization problem, we decompose the problem into separate parts and propose a heuristic to solve each subproblem for large-scale network scenarios.**

I. Introduction

5G systems are designed to accommodate tremendous improvements in operational aspects of multitude vertical services such as smart cities, autonomous cars, and e-health. The 5G technology promises ubiquitous support of smart services enabled by massive amounts of sensors that function within an interworking framework [1]. Reducing energy consumption of the sensors is very critical in extending the lifetime of the whole network. Oil and gas pipeline monitoring, natural disaster predictions, fire detection, agricultural and environmental monitoring all involve deployment of sensing devices in hard-to-reach areas. Unmanned aerial vehicles (UAVs) have recently gained popularity in the telecom industry. UAVs serve as an effective way to collect data from wireless sensor nodes dispersed in rural and hard-to-reach areas. Furthermore, UAVs' ability to establish line of sight connection with sensors improve the wireless channel quality between the UAV and the sensor thus boosting data rate and reducing energy consumption. In many of their applications, sensor nodes measure and send collected data to a central unit (eg. sink) for processing either directly or through multiple hops. In the case of massive sensor deployment, there is an inevitable need for innovative data collection solutions that can further save energy for the survival and longevity of WSNs. Hence, in this paper, we address the problem of gathering data from large scale WSN with randomly deployed sensors in the most energy-efficient manner. To this end, a UAV is deployed to fly over the area of interest, collect, and deliver measured data to the sink; thus, saving sensor energy needed for multiple data relaying to the sink. UAVs are, an effective and energy-efficient solution for sensors placed in hard-to-reach areas where no direct connection exists from any sensor node to the sink.

Previous research on WSN explored using UAVs for energy-efficient data collection [2,3] by dividing the region into clusters, each with a designated cluster head (CH). Nodes within a cluster transmit their data to the CH when UAVs eventually deliver the collected data to the sink. In this work, we focus on achieving major energy-savings to suit large-scale networks through incorporating the benefits of UAV-assisted data collection together with those of Compressive Data Gathering (CDG). With CDG, rather than receiving readings from all sensors, the sink receives few encoded sums of all the readings, from which the sink will be able to recover (decode) the original data, as long as the readings can be transformed or compressed in some sparse orthonormal transform domain [4,5]. In our work, we suppose the original data is compressible in some transform domain, and it is recovered at the sink by receiving sparse projections [6]. Each projection is gathered by establishing forwarding tree from nodes in one cluster to the CH, where subsequently all the gathered data at the CHs is sent through UAV to the sink. Upon collecting all projections, the sink then attempts to recover the original data [7]. Therefore, instead of transmitting sensor's reading from each node to the CH in a separate packet through multi-hop, the data is aggregated en-route to the CH. Hence, we may construct an efficient aggregation tree to gather data for each cluster such that it minimizes the total number of transmissions and thus minimizing the energy consumption of sensor nodes. This requires the construction of an aggregation tree per cluster to minimize the total number of transmissions and reduce the sensors total energy consumption. Our problem becomes of jointly clustering the network, choosing the right CHs, finding the UAV trajectory over the chosen CHs, and constructing aggregation trees based on CDG technique, such that the total energy consumption of the sensor nodes and the UAV trajectory are minimized. We mathematically formulate this problem as a mixed integer linear program (MILP), and owing to its complexity, we decompose the problem into four complementary disjoint subproblems. After underlining the complexity of each part, we present low-complexity algorithms to solve the problem efficiently for large scale networks. Finally, we demonstrate the performance and efficiency of our methods through simulation. The rest of the paper is organized as follows. Section II presents the system model, a brief description of the projection-based CDG framework, and the problem description. The mathematical problem formulation is given in Section III. The proposed low complexity algorithms are then presented in Section IV, followed by results and analysis in Section V, and finally conclusion in Section VI.

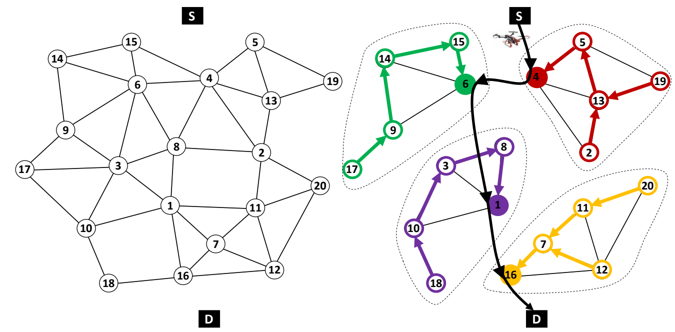
II. System Model

We model a wireless sensor network as a connected graph $G = (V, E)$, where V is a set of N sensor nodes deployed randomly in a given region, and E is the set of links or edges between any two sensor nodes with enough transmit power to be within radio range with respect to each other. We assume the density of nodes and the transmit power capability per node are high enough to have a connected graph. We assume each sensor node at each round has a data reading x_i which it intends to send to a sink node, e.g., cellular base station or data collection central unit. Consequently, at each round, the sink node needs to gather, in total, a data vector of size N (i.e., $X = [x_1, x_2, ..., x_N]^T$) from all the sensor nodes in the network. In order to avoid long range transmissions or relaying data over multiple hops from each sensor node to the sink, we utilize a UAV to navigate over the region in order to collect data from the sensors and deliver to the sink. This can directly enhance the energy efficiency in the network and the connectivity to any remote sensor node due to the high flexibility in controlling the UAV's trajectory.

A sample graph is shown in Fig. 1(a), where sensor nodes are represented by circles; the UAV's source location (S) and destination location (D) are represented by dark squares. Note that the destination of the UAV represents the desired sink node where the sensors' readings are intended to be delivered and processed. The source and the destination can be co-located or reside in different locations depending on the use case and the structure of the network.

However, flying the UAV over all the nodes in the network to collect data is not efficient as it would lead to long travel trajectory, high data collection delay and the need to frequently recharge the UAV's battery. To achieve a balanced trade-off between shortening the UAV trajectory and reducing the energy consumption of the sensor nodes, we divide the sensor network into mutliple clusters where in each cluster one of the sensor nodes is selected as a CH. The CH is responsible for gathering data from all other nodes in the same cluster, and consequently sending the aggregated data to the UAV to be delivered to the sink node. The UAV has then to traverse all the selected CHs where it should visit each CH once and arrive at the destination with minimized total trajectory distance, similar to the traveling salesman problem.

Without loss of generality, we assume the UAV flies at an appropriate low altitude over each CH to collect data with fixed transmit power and high bit rate over a line of sight channel. Moreover, the UAV flies with a suitable velocity to have enough contact time with each CH to collect its aggregated data. Therefore, our problem aims at clustering the nodes and choosing the most suitable CHs such that the total transmitted power for data collection in the network and the trajectory of the UAV over the CHs are jointly minimized. In order to further optimize performance, we integrate a projection-based compressive data gathering approach as explained in the subsections below. Fig. 1(b) shows an example scenario with four clusters and an optimized UAV trajectory passing over the four selected CHs on the trajectory from the UAV source location to the destination sink node (represented by a black



(a) Sample input. (b) Sample output solution.

Fig. 1. Example network scenario with UAV-aided projection-based CDG.

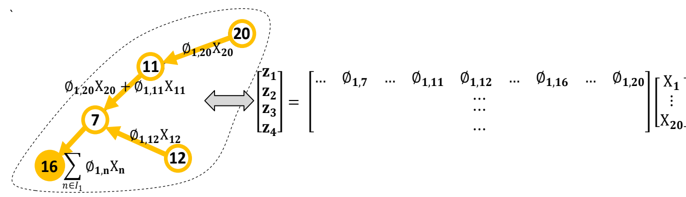


Fig. 2. Example forwarding tree construction using projection-based CDG.

thick line traversing the CHs). In addition, the figure shows sample forwarding tree construction per cluster (represented by a set of arrows).

A. Compressive Data Gathering (CDG)

CDG promises to efficiently recover N sensors' readings at the sink with far fewer sample measurements, as long as the original readings could be transformed or compressed in some sparse orthonormal transform domain. Suppose the original data $X = [x_1, x_2, ..., x_N]^T$ has a k-sparse representation under a proper matrix Ψ, where Ψ is a Fourier transform matrix of size $N \times N$. That is $X = \Psi\hat{S}$, where $\hat{S}$ is a k-sparse column vector representation of X and only k coefficients of $\hat{S}$ are non-zero and $k \ll N$. According to the Restricted Isometry Property (RIP) of the Compressed Sensing (CS) theory [5], the sink may receive $M = O(k \log N)$ measurements instead of N readings, where $M \ll N$; that is $Z = \Phi\Psi\hat{S} = \Phi X$, where Z is a column vector of sample measurements of size $M \times 1$ and Φ is a sample measurement matrix of size $M \times N$. In other words, the sink can perfectly recover the original data X by receiving $Z = [z_1, z_2, ..., z_M]^T$, where $z_m = \sum_{n=1}^{N} \phi_{m,n} x_n$, $m = 1, 2, ..., M$ and $\phi_{m,n}$ is a coefficient in matrix Φ at row m and column n. Each z_m represents a weighted sum of measurements from nodes in the network with non-zero coefficients in a row of the matrix Φ. We refer to these nodes as interest nodes and the data aggregated from them as one projection. The matrix Φ has M rows, one row for each codded sum (projection), and N columns, one column for each sensor node. Now, from M measurements (Z), using the random sample matrix Φ and the Fourier transform matrix Ψ, the sink recovers the sparse representation of the data $\tilde{S}$ (not the original data) by solving the convex optimization problem:

$$\min_{\tilde{S}} \|\tilde{S}\|_1 \qquad \text{subject to} \qquad Z = \Phi\Psi\tilde{S} = A\tilde{S} \tag{1}$$

After recovering the sparse vector $\tilde{S}$, the original data (X) is obtained by letting $X = \Psi\tilde{S}$. In this paper, similar to [8]–[10], the number of non-zero coefficients in each row of the matrix Φ is chosen as $\lceil \frac{N}{M} \rceil$ (almost equal size) such that none of the columns in Φ has all-zero entries. For more details on CS, the reader is referred to [5,7].

B. Problem Description

Given a connected graph G, sample matrix Φ, the problem of UAV-aided projection-based CDG consists of finding M forwarding trees, where each tree should collect the coded sum from a set of nodes belonging to one cluster (i.e., nodes with non-zero coefficient in a corresponding row of matrix Φ) en-route to the selected CH with optimized energy efficiency. Each tree m ($1 \leq m \leq M$) corresponds to one cluster (or projection in CDG) which gathers one coded sum z_m from a set of nodes in a cluster (or interest nodes) at a CH; the set of M clusters is denoted as W. Our objective is to construct the forwarding trees such that the total power transmitted in the network is minimized. The gathering on each forwarding tree is performed as follows: each interest node (n, $n \in I_m$, where I_m is the set of interest nodes of tree m or set of nodes in cluster m) upon collecting its measurement, multiplies its reading x_n with its random coefficient $\phi_{m,n}$ and combines the data $\phi_{m,n}x_n$ with those received from its descendants (if any) and sends the obtained coded sum in one packet to the parent node. Finally, the CH which is the root of the tree receives one coded sum $z_m = \sum_{n \in I_m} \phi_{m,n}x_n$. Fig. 2 illustrates an example of the projection-based CDG process for one forwarding tree (e.g., z_1) in a given cluster.

The interest nodes set I_m can be represented as a matrix with $I_{m,n} \in \{0,1\}$, i.e., $I_{m,n} = 1$ when node n is in cluster m. It should be noted that the number of clusters (M) is determined in advance and depends on the number of nodes in the network and the sparsity representation of the sensor data, i.e., $M = O(k \log N)$, where k is the sparsity representation of the data in the Fourier transform domain. Consequently, the size of each cluster is almost equal to $\lceil \frac{N}{M} \rceil$. It should be also noted that without compressive data gathering, nodes closer to the CH will perform more forwarding than nodes which are farther away. Thus, with CDG, the transmission load is distributed across multiple sensor nodes in the network which leads to more balanced energy consumption yielding an extended network lifetime.

III. Problem Formulation

In this section, we formulate the UAV-aided CDG problem as a mixed linear integer program (MILP), with an objective to divide the sensor nodes into almost equal-sized clusters with one selected CH per cluster. The aim is to generate forwarding trees for data gathering at CHs with minimum total transmission power, while at the same time minimizing the UAV trajectory to traverse all CHs from source location to destination sink node. Let $P_{ij}^{m,h}$ be the transmission power required to transmit data from nodes i to j (or over link (i,j)) in cluster m if h is the CH. And let $U_{ij} \in \{0,1\}$ indicate whether the UAV will traverse from node i to j, where $i\&j \in \{V \bigcup S \bigcup D\}$, V is the set of sensor nodes, S is the source node and D is the destination node.

Objective: The objective is to jointly minimize the total transmission power required for data gathering in the network and the total length of the UAV trajectory.

$$\text{Minimize} \quad \omega \sum_{\substack{m \in W \\ h \in V \\ (i,j) \in E}} P_{ij}^{m,h} + (1-\omega) \sum_{i\&j \in \{V \bigcup S \bigcup D\}} d_{ij}U_{ij} \tag{2}$$

Subject to: constraints (3) - (18), which are described below.

The first term in the objective function corresponds to the total power transmission of active links in the constructed forwarding trees, and the second term depicts the total trajectory distance of the UAV. The parameter ω ($0 \leq \omega \leq 1$) indicates the weight of each term in the objective function. Depending on the use case, the weight can be adapted to balance the level of importance between the sensor nodes' energy efficiency and the UAV's trajectory.

Clustering constraints: For the projection-based CDG, the only constraints required for the clustering are to enforce the size of the clusters to be equal to $\lceil \frac{N}{M} \rceil$ such that no node belongs to more than one cluster. This is normally feasible since we assume a dense sensor network scenario with enough transmit power capability per sensor node to have a connected network graph. Let $I_{m,n} \in \{0,1\}$ indicate whether node n belongs to cluster m or not. Hence, constraint (3) enforces the size of the clusters and constraint (4) asserts that each node should belong to only one cluster.

$$\sum_{n \in V} I_{m,n} \leq \lceil \frac{N}{M} \rceil, \qquad \forall m \in W. \tag{3}$$

$$\sum_{m \in W} I_{m,n} = 1, \qquad \forall n \in V. \tag{4}$$

Flow conservation constraints: These constraints are required in order to construct a routing-path from the transmitter to the receiver of each data packet. In graph theory [11], for a network routing-path, the flow conservation asserts that the amount of incoming flow to a node equals to the amount of outgoing flow, except for a transmitter (which has only outgoing flow) and a receiver (which has only incoming flow). In our problem, a flow is a data packet, which needs to be directed from a node in each cluster to its corresponding CH. Let $F_{hj}^{h,m} \geq 0$ be the data packet (flow) imposed by certain routing on edge (i,j) (i.e., between nodes i and j) to aggregate data in cluster m if h is a CH. Constraints (5) and (6) assert that in each cluster, if h is a CH, there is no flow going out of h and there has to be $C_m - 1$ number of flows coming into h, respectively. C_m is the size of cluster m, i.e., the number of nodes in the cluster. On the other hand, constraint (7) ensures that the total number of incoming minus outgoing flows for all other nodes in the cluster has to be utmost one to cause one flow being forwarded from each node in the cluster to the CH. In addition, the constraints in (8) make sure that there is no flow on any edge (i,j) which is not in cluster m.

$$\sum_{j:(h,j) \in E} F_{hj}^{h,m} = 0, \qquad \forall m \in W, h \in V. \tag{5}$$

$$\sum_{i:(i,h)\in E} F_{ih}^{h,m} = I_{m,h}(C_m - 1), \qquad \forall m \in W, h \in V. \quad (6)$$

$$\sum_{j:(n,j)\in E} F_{nj}^{h,m} - \sum_{i:(i,n)\in E} F_{in}^{h,m} \le I_{m,n}, \qquad \forall m \in W, h \in V, n \in V/h. \quad (7)$$

$$\begin{cases} F_{ij}^{h,m} \le I_{m,i}B, \\ F_{ij}^{h,m} \le I_{m,j}B, \end{cases} \qquad \forall m \in W, h \in V, (i,j) \in E. \quad (8)$$

Forwarding tree link creation constraints: These constraints create forwarding links for a tree. Let $\chi_{ij}^{m,h} \in \{0,1\}$ indicate whether there is a forwarding link between nodes i and j in cluster m if h is a CH. $\chi_{ij}^{m,h} = 1$, if there is a positive traffic flow from i to j, and zero otherwise. This implies that $F_{ij}^{m,h} = 0 \Rightarrow \chi_{ij}^{m,h} = 0$ and $F_{ij}^{m,h} > 0 \Rightarrow \chi_{ij}^{m,h} = 1$, which is achieved by the following inequalities (B is a large constant greater than any possible value for $F_{ij}^{m,h}$):

$$\begin{cases} \chi_{ij}^{m,h} \le F_{ij}^{m,h}, \\ \chi_{ij}^{m,h} \ge \frac{F_{ij}^{m,h}}{B}, \end{cases} \qquad \forall m \in W, h \in V, (i,j) \in E. \quad (9)$$

Transmit power constraints: The following constraints make sure that, for any forwarding link in a tree, enough transmission power is consumed for successful data transmission modeled via a signal-to-noise-ratio (SNR) greater than a threshold β; otherwise, the power level is set to zero. In other words, when $\chi_{ij}^{m,h} = 1 \Rightarrow \frac{P_{ij}^{m,h} d_{ij}^{-\alpha}}{\eta} \ge \beta$ and when $\chi_{ij}^{m,h} = 0 \Rightarrow P_{ij}^{m,h} = 0$, where d_{ij} is the distance between i and j, α is the path loss exponent, and η is the background thermal noise. This can be represented as follows:

$$\begin{cases} P_{ij}^{m,h} \le \chi_{ij}^{m,h} B, \\ P_{ij}^{m,h} \ge \chi_{ij}^{m,h} \frac{\eta\beta}{d_{ij}^{-\alpha}}, \end{cases} \qquad \forall m \in W, h \in V, (i,j) \in E. \quad (10)$$

Cluster head selection constraints: These constraints are needed to choose the most suitable CHs for minimizing the UAV trajectory. Let H_h^m indicate whether node h is chosen as a CH for cluster m and let T_h indicate the UAV's trajectory. Constraint (11) asserts that a node cannot be chosen as a CH if it is not in the cluster in the first place, i.e., $H_h^m = 0$ when $I_{m,h} = 0$. Constraint (12) makes sure that there is only one CH for each cluster, and constraint (13) ensures that M CHs should be chosen for the UAV's trajectory.

$$H_h^m \le I_{m,h}, \qquad \forall m \in W, h \in V. \quad (11)$$

$$\sum_{h\in V} H_h^m = 1, \qquad \forall m \in W. \quad (12)$$

$$T_h = \sum_{m\in W} H_h^m, \qquad \forall h \in V. \quad (13)$$

UAV trajectory constraints: These constraints make sure that a UAV will start from the source, traverse all the CHs in the network for data collection, and eventually return or arrive at the destination. Let U_{ij} indicate whether the UAV will traverse from nodes i to j. Constraints (14), (15) and (16), similar to the flow conservation constraint, force the UAV to traverse from the source over the network to the destination. More specifically, constraints (14) and (15), respectively, enforce the UAV to start from the source S and arrive at the destination D. Constraint (16) ensures that the UAV will traverse through some nodes in the network. In other words, if node n is on the trajectory of the UAV, in total one UAV will arrive at node n and the same one will leave, which results in the difference of incoming and outgoing on node n to be zero. Constraint (17) makes sure that the UAV will traverse only the CHs, and finally the sub-tour elimination constraint (18) ensures that the UAV will not trap into cycles. Note that R_i is a dummy variable required for the sub-tour elimination.

$$\sum_{j\in V} U_{Sj} = 1; \quad (14)$$

$$\sum_{i\in V} U_{iD} = 1; \quad (15)$$

$$\sum_{\substack{i\in\{V\bigcup S\bigcup D\}\\ i!=n}} U_{in} - \sum_{\substack{j\in\{V\bigcup S\bigcup D\}\\ j!=n}} U_{nj} = 0, \quad \forall n \in V. \quad (16)$$

$$\begin{cases} T_i = \sum_{j\in\{V\bigcup S\bigcup D\}} U_{ij}, & \forall i \in V \\ T_j = \sum_{i\in\{V\bigcup S\bigcup D\}} U_{ij}, & \forall j \in V \end{cases} \quad (17)$$

$$R_i - R_j + U_{ij}M \le M - 1, \ \forall i \in V, j \in V : i \ne j. \quad (18)$$

IV. Proposed Low Complexity Algorithms

The problem of UAV-aided projection based CDG to jointly cluster the network, choose the appropriate CHs, find the UAV trajectory over the chosen CHs, and construct a forwarding tree to gather data in each cluster in the most energy efficient manner is a complex problem of combinatorial nature. The execution time of solving the joint optimization problem grows exponentially as the size of the network grows, which limits its scalability. For example, it takes 18 minutes to solve for a 20-node network size and more than 39 hours for 30-node network. Hence, in this section we decompose the joint problem into four complementary subproblems and after analyzing the complexity of each subproblem, we present a heuristic low-complexity algorithm to efficiently solve each subproblem for large scale networks.

Clustering subproblem: The clustering subproblem can be represented as an updated version of K-means clustering [12]. In K-means clustering, the aim is to partition N nodes into K clusters where nodes belonging to one cluster have the nearest mean using Euclidean distance; this results in partitioning the network into Voronoi cells. On the other hand, in the formulated clustering subproblem, a key difference is that the sizes of the clusters should be uniform even if the solution distorts the shapes of the Voronoi cells. In the proposed algorithm, after applying K-means clustering method, we update the clusters to end up with almost equal size cells. Note that K-means clustering is proven to be NP-hard in [13]. Hence, this implies that our clustering subproblem is also NP-hard; the proof is omitted due to the similarity to K-means clustering.

Algorithm 1 initiates by the K-means clustering algorithm (Phase I). The algorithm starts by placing random $K = M$ points (as center points for clusters) inside the region, and assigns each node in the network to the nearest random points. Iteratively, the K-means algorithm finds the center point for

Algorithm 1: Clustering subproblem

Data: Graph $G(V, E)$, Number of clusters M
Result: Clusters I_m, $m = 1, 2, ..., M$.

Phase 1 (K-means):
Randomly initialize $K = M$ cluster center points $\mu_1, \mu_2, ..., \mu_m$,
$I_m \leftarrow \emptyset$.
repeat
  $I_m \leftarrow I'_M$.
  for *each node* n **do**
    for *each cluster* m **do**
      $I_m \leftarrow min\|v_n - \mu_m\|^2$.
  for *each cluster* m **do**
    μ_m = mean of I_m.
until $I_m = I'_m$;
Phase II (updates to equalize cluster sizes):
repeat
  for *each cluster* m **do**
    while $size(m) < \frac{N}{M}$ **do**
      for *each node* i *that shares an edge with cluster* m **do**
        $Q_{sorted} \leftarrow Add(i)$.
      if $Found_{Node}$ = *node in* Q_{sorted} *that its cluster has size* $> \frac{N}{M}$ **then**
        $I_m \leftarrow Add(Found_{Node})$.
        $Restricted_{Set} \leftarrow Add(m)$.
      else if $Found_{Node}$ = *node in* Q_{sorted} *that its cluster* $\notin Restricted_{Set}$ **then**
        $I_m \leftarrow Add(Found_{Node})$.
        $Restricted_{Set} \leftarrow Add(m)$.
      else
        $Found_{Node}$ = node in Q_{sorted} that its cluster has lower restriction level.
        Increase restriction level for the borrowed cluster.
        $I_m \leftarrow Add(Found_{Node})$.
        $Restricted_{Set} \leftarrow Add(m)$.
until *all clusters have* $\frac{N}{M}$ *or* $\frac{N}{M} + 1$ *nodes*;

nodes in each cluster by taking the mean of the vertical and horizontal Euclidean axes, and reallocate nodes in the network to the nearest center points until convergence; this step is repeated until there are no differences in clustering. Next, the algorithm in Phase II nearly equalizes the size of the clusters. For each cluster m, which has less that $\frac{N}{M}$ nodes, it first tries to borrow a node (nearest one) from a neighbor cluster that has a size greater than $\frac{N}{M}$, if any, and an edge connecting the node to cluster m. Otherwise, a node can be borrowed from a neighbor cluster which has no restriction on borrowing, if any. Else, cluster m will borrow a nearest node that shares an edge from a cluster with lower restriction level. By borrowing from a restricted cluster, its restriction level is increased by one. Consequently, when a cluster enlarges its size, it restricts itself more from borrowing. The running time of Phase I (K-Means algorithm) is $O(NMT)$, where T is the number of iterations needed until convergence. It should be noted that the number of iterations is often small, and therefore K-Means algorithm is often considered to be of *linear* complexity in practice, although it is in the worst case *superpolynomial* when performed until convergence [14]. Phase II takes $O(MYJ \log(J))$, where Y is the maximum number of nodes that a cluster might have shortage to $\frac{N}{M}$ nodes, after clustering with K-means algorithm, and J is the maximum number of nodes in Q_{sorted} (queue which holds neighbor nodes that share edges with cluster m). Note that the complexity of sorting Q_{sorted} in place is $O(\log(J))$. Hence, in total, the time complexity of the clustering subproblem is $O(NMT + MYJ \log(J))$, where the algorithm takes more time to run Phase I (K-means) than Phase II (updates). Therefore, the running time of the algorithm is $O(NMT)$.

Cluster heads selection subproblem: To elect M CHs in a network of size N nodes, there are $\binom{N}{M}$ distinctive combinations. In a large scale network, obtaining the optimal CHs through is complex due to the huge search space, and is known to be an NP-hard problem [15]. The proposed algorithm for choosing CHs is summarized in Algorithm 2. The algorithm uses two methods based on the position of the source S and destination D. In the first method, when S and D are positioned at two sides of the network which requires the UAV to travel from one side to the other, a straight line is drawn from S to D. Next, the Euclidean distance from each node to the drawn line is computed. Finally, for each cluster, the node with the smallest computed distance to the line is chosen as the CH of that cluster. In the second method, when both S and D are positioned on one side (or $S = D$), the node in each cluster that has the shortest Euclidean distance to S or D is chosen as the CH. In the first method, the algorithm to compute the distances for all nodes in the network to the drawn line and also to find the closest node in each cluster to the line has complexity $O(N)$. As for the second method, the running time of the algorithm is also $O(N)$. Hence, the time complexity of Algorithm 2 is $O(N)$.

UAV trajectory subproblem: The UAV trajectory subproblem can be solved similar to the Traveling Salesman Problem (TSP), and it is proven to be NP-hard in [16]. Therefore, we use a heuristic algorithm implementation for TSP (nearest-neighbor algorithm) to solve our UAV trajectory subproblem. Algorithm 3 summarizes the steps of finding the UAV trajectory (U_{ij}). The algorithm starts by finding the nearest CH to the source (S), and assigns it to its first traversal node. Next, the algorithm keeps finding and assigning the nearest unvisited CH to the UAV's trajectory, until there is no CH left to be added. Finally, the destination (D) is added as the last node. The running time of the algorithm, similar to the nearest-neighbor algorithm of the TSP, is $O(M^2)$.

Forwarding tree construction subproblem: We may use one of the Minimum Spanning Tree (MST) methods (e.g., Kruskal algorithm [11]) to efficiently solve the forwarding tree construction subproblem for each cluster. The complexity of the algorithm is $O(\lambda_m \log(C_m))$, where λ_m and C_m are, respectively, the total number of edges and nodes in cluster m. This algorithm can solve for all M clusters in parallel.

Algorithm 2: Choosing cluster heads subproblem

Data: Graph $G(V,E)$, clusters I_m, distances d_{ij}.
Result: Cluster heads H_m, $m = 1, 2, ..., M$.
if *S and D are resided on two ends* **then**
  $Line_{(S,D)} \leftarrow$ Draw a straight line from S to D.
  for *each node n* **do**
    Find $d_{V_n Line_{(S,D)}}$.
  for *each cluster m* **do**
    for *each node $i \in I_m$* **do**
      $H_m =$ smallest($d_{V_i Line_{(S,D)}}$).
else
  for *each cluster m* **do**
    for *each node $i \in I_m$* **do**
      $H_m =$ smallest($d_{V_i S}$, $d_{V_i D}$).

Algorithm 3: UAV trajectory subproblem

Data: Cluster head set H.
Result: UAV trajectory U_{ij}, $(i, j) \in L$.
$C \leftarrow$ Find nearest cluster head to S.
$U_{SC} \leftarrow$ link (S, C).
Remove C from set H.
while *H is not empty* **do**
  $C' \leftarrow$ Find nearest cluster head to C.
  $U_{CC'} \leftarrow$ link (C, C').
  $C = C'$.
  Remove C from set H.
$U_{CD} \leftarrow$ link (C, D).

V. Performance Evaluation

In this section, we first study the performance of the proposed algorithmic method versus the joint problem for relatively small network scenarios by varying the number of sensor nodes and clusters. We also study the performance by positioning the UAV source and destination once by residing both at one side of the network (i.e., $S = D$), and then by positioning them on opposite sides (i.e., $S \neq D$). We use the total power transmissions and UAV trajectory distance, as well as execution time as the key metrics for the comparisons. We then study the performance of the proposed algorithms (denoted as UAV-CDG) on a relatively large network scenario with comparison to benchmark methods that either gather data without using compressive sensing (denoted as Non-CDG), or without UAV assistance (denoted as Non-UAV). For the numerical results, we generate arbitrary networks with N nodes where the nodes are randomly distributed over a region of $900 \times 700 m^2$, such that the resulting graph is connected. The density (average nodal degree) of the network is tuned by increasing or decreasing the communication range of each node. Source (S) and destination (D) are placed 50 meters away from the network. We use network sizes of 20, 25 and 30 nodes with communication ranges of 200, 190, and 180 meters, respectively, for the small network scenarios. For large networks we use the following combinations of number of nodes and radio ranges: 100 nodes (with communication range of 120m), 250 nodes (100m), 500 nodes (80m), 750 nodes (60m) and 1000 nodes (40m). We assume the SNR threshold for successful transmission is set to $\beta = 2$, corresponding to a given target bit rate level. We further assume $\omega = 0.5$, giving equal weights to both terms in the objective function. We use CPLEX to solve the formulated optimization problem, and JAVA based simulator to simulate the operation of the low complexity algorithms. We generate results on CPU with Intel(R) Core(TM) i7-4790 CPU @ 3.6 GHz speed, 16 GB memory ram and 64-bit windows operating system.

We start by examining the results obtained by solving the joint optimization model and compare it with the results obtained from the heuristic algorithms using small networks. Fig. 3 and Fig. 4, respectively, show the total transmission power and UAV trajectory distance required to gather data in the network for both $S! = D$ and $S = D$. Note that the total transmission power for the both positioning is equal. For the compressive data gathering, we use the number of projections (clusters) to be $M = 0.2N$. As depicted in the figures, the proposed algorithms achieve notably good performance in terms of the total needed transmission power and the UAV trajectory distance when $S! = D$ (gap less than 4% and 3% respectively), yet with worse performance (gap less than 24%) when $S = D$; this is due to the efficiency of the implemented TSP heuristic algorithm, where UAV, in case of $S = D$, traverses CHs from the closer to the farthest one and then returns to the sink node, which results in longer distance. However, in terms of computational complexity, the algorithmic method returns solutions in less than a second, whereas, the joint optimization method takes 18 minutes to generate solution for a 20-node network, two hours and 56 minutes for a 25-node network, and 38 hours plus 39 minutes for a 30-node network. These results confirm that the joint optimization problem is indeed very complex and that the low complexity algorithms are much simpler with close to optimal performance. One more observation that we can notice from the figures is that as the number of nodes in the network increases, the total energy consumption as well as the UAV trajectory distance increase.

In Fig. 5, we study the performance by varying the number of clusters ($M = 5$ to $M = 25$) on a network of size 20-node. As plotted in the figure, the algorithmic method in terms of the UAV trajectory distance when $S! = D$ performs very close to the joint optimization method with a maximum gap of 2% only. Furthermore, its performance when $S = D$ is not worst than 24%, which is still acceptable for such an algorithm with a very low time complexity. To be noted that the total power transmissions for both positioning of S and D is equal and similar to the joint method. From the figure, we may also observe that the UAV trajectory distance grow by the number of clusters. The reason goes for the increasing number of cluster heads where a UAV is required to traverse them. In addition, the figure shows that the UAV has to traverse longer distance in case when $S = D$ for all different number of clusters except for networks with very few number of clusters (eg., $M = 2$ shown in the figure). This is because CHs can be chosen very close to the source (which is also the destination), and avoid traversing the entire network's region to reach the

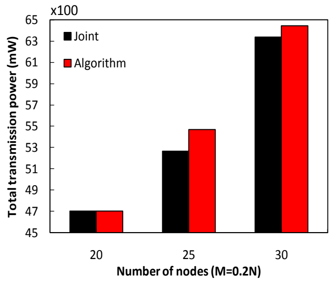


Fig. 3. Transmission Power Vs. number of nodes ($M = 0.2N$).

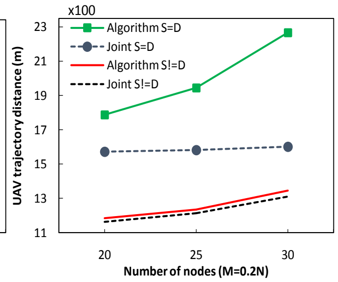


Fig. 4. UAV trajectory distance Vs. number of nodes ($M = 0.2N$).

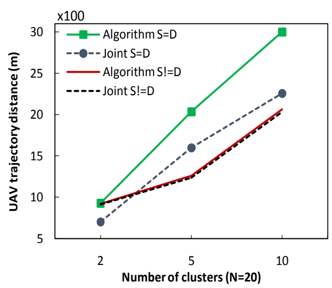


Fig. 5. UAV trajectory distance Vs. number of clusters ($N = 20$).

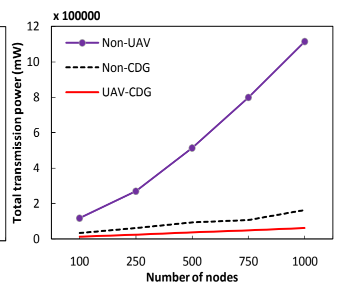


Fig. 6. Transmission power Vs. number of nodes for different algorithms.

destination on the other end.

Finally, we consider a large network scenario and we compare the performance of our proposed algorithm (UAV-CDG) in term of total transmission power with methods that either do not use compressive sensing for data gathering (Non-CDG) or do not utilize UAV assistance for data collection (Non-UAV). The results for $M = 0.1N$ with average over ten runs are shown in Fig. 6. The plots in the figure show that gathering data from a network in a basic way without relying on any techniques of compressive sensing and UAV assistance (Non-UAV) require significantly more energy consumption in order to transmit and relay data through multiple hops to the destination sink node. Consequently, it is obvious that the performance of the Non-UAV method is worse than the other methods. With UAV assistance and gathering data at cluster heads instead of the sink which decreases the number of relaying transmissions, the needed energy consumption is reduced notably. As it can be seen from the figure, the Non-CDG method saves 71% to 85% in terms of energy compared to the Non-UAV method. Furthermore, our UAV-CDG method, owing to compressive data gathering, performs 55% to 65% better than Non-CDG method and respectively 89% to 94% better than the Non-UAV method. This shows that our proposed method substantially outperforms other methods in terms of energy consumption, which leads to enhanced wireless sensor network lifetime.

VI. Conclusion

In this paper, we proposed a novel data collection technique in WSNs using projection-based Compressive Data Gathering (CDG) and UAVs. CDG is utilized to reduce the number of transmissions and corresponding energy consumption through aggregating data en-route from sets of sensor nodes to a set of projection heads. The UAV is used to further enhance the energy efficiency of the sensors by avoiding long range and multiple hop transmissions to reach the destination sink node. As part of the solution approach, we divided the sensor nodes into clusters and constructed a forwarding tree for each cluster based on the CDG technique to minimize an objective that is a function of both the total transmission power in the network and the total UAV flight distance. We mathematically formulated the joint optimization problem including clustering, cluster head selection, forwarding trees construction, and UAV trajectory planning. Owing to the complexity of the joint problem, we decomposed the problem into multiple disjoint subproblems and proposed a heuristic algorithm for each separately with notably less complexity in order to scale up the solution to dense sensor network scenarios. Finally, we presented extensive performance results to demonstrate the efficiency and superiority of the proposed algorithmic method.

References

[1] 3GPP, "Feasibility study on new services and markets technology enablers for massive internet of things," Tech. Rep., 2016.

[2] J. R. Martinez-de Dios, K. Lferd, A. de San Bernabé, G. Nunez, A. Torres-González, and A. Ollero, "Cooperation between uas and wireless sensor networks for efficient data collection in large environments," *Journal of Intelligent & Robotic Systems*, vol. 70, no. 1-4, pp. 491–508, 2013.

[3] P. Sujit, D. E. Lucani, and J. Sousa, "Joint route planning for uav and sensor network for data retrieval," in *Systems Conference (SysCon), 2013 IEEE International*. IEEE, 2013, pp. 688–692.

[4] C. Luo, F. Wu, J. Sun, and C. W. Chen, "Compressive data gathering for large-scale wireless sensor networks," in *Proceedings of the 15th annual international conference on Mobile computing and networking*. ACM, 2009, pp. 145–156.

[5] D. L. Donoho, "Compressed sensing," *Information Theory, IEEE Transactions on*, vol. 52, no. 4, pp. 1289–1306, 2006.

[6] W. Wang, M. Garofalakis, and K. Ramchandran, "Distributed sparse random projections for refinable approximation," in *Proceedings of the 6th international conference on Information processing in sensor networks*. ACM, 2007, pp. 331–339.

[7] E. Candès and M. Wakin, "An introduction to compressive sampling," *Signal Processing Magazine, IEEE*, vol. 25, no. 2, pp. 21–30, 2008.

[8] D. Ebrahimi and C. Assi, "On the interaction between scheduling and compressive data gathering in wireless sensor networks," *IEEE Transactions on Wireless Communications*, vol. 15, no. 4, pp. 2845–2858, 2016.

[9] ——, "Network coding-aware compressive data gathering for energy-efficient wireless sensor networks," *ACM Transactions on Sensor Networks (TOSN)*, vol. 11, no. 4, p. 61, 2015.

[10] ——, "Compressive data gathering using random projection for energy efficient wireless sensor networks," *Ad Hoc Networks*, vol. 16, pp. 105–119, 2014.

[11] T. H. Cormen, *Introduction to algorithms*. MIT press, 2009.

[12] 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, 2002.

[13] M. Mahajan, P. Nimbhorkar, and K. Varadarajan, "The planar k-means problem is np-hard," in *International Workshop on Algorithms and Computation*. Springer, 2009, pp. 274–285.

[14] D. Arthur and S. Vassilvitskii, "How slow is the k-means method?" in *Proceedings of the twenty-second annual symposium on Computational geometry*. ACM, 2006, pp. 144–153.

[15] Y. Zhou, N. Wang, and W. Xiang, "Clustering hierarchy protocol in wireless sensor networks using an improved pso algorithm," *IEEE Access*, vol. 5, pp. 2241–2253, 2017.

[16] M. R. Garey and D. S. Johnson, *Computers and intractability*. wh freeman New York, 2002, vol. 29.