EDAL: an **E**nergy-efficient, **D**elay-**a**ware, and **L**ifetime-balancing Data Collection Protocol for Wireless Sensor Networks

Yanjun Yao[1] and Qing Cao[1]
[1]Electrical Engineering and Computer Science
University of Tennessee, Knoxville, TN, US
Email: {yyao9, cao}@utk.edu

Athanasios V. Vasilakos[2]
[2]Electrical and Computer Engineering
National Technical University of Athens, Greece
Email: vasilako@ath.forthnet.gr

***Abstract*—In many wireless sensor network (WSN) applications, a subset of nodes (source nodes) are selected to sense the environment, generate data, and transmit them back to the sink over multiple hops. Many previous research efforts have tried to achieve trade-offs in terms of delay, energy cost, and load balancing for such data collection tasks. Our work in this paper stems from the insight that, recent research efforts on open vehicle routing (OVR) problems, an active area in operations research, are based on similar assumptions and constraints compared to sensor networks. This insight motivates us to adapt these techniques so that we can solve or prove certain challenging problems in WSN applications. To demonstrate that this approach is feasible, we develop one data collection protocol called EDAL, which stands for Energy-efficient Delay-aware Lifetime-balancing data collection. The algorithm design of EDAL borrows one research result from OVR to prove that its problem formulation is inherently NP-hard. We then proposed both a centralized heuristic to reduce its computational overhead, and a distributed heuristic to make the algorithm scalable for large scale network operations. We also develop EDAL to be closely integrated with compressive sensing, an emerging technique that promises considerable reduction in total traffic cost for collecting sensor readings under loose delay bounds. Finally, we systematically evaluate EDAL to demonstrate its performance superiority compared to related protocols.**

I. Introduction

In recent years, wireless sensor networks (WSNs) have emerged as a new category of networking systems with limited computing, communication, and storage resources. A WSN consists of nodes deployed to sense physical or environmental conditions for a wide range of applications, such as environment monitoring [1], scientific observation [2], emergency detection [3], field surveillance [4], and structure monitoring [5]. In these applications, prolonging the lifetime of WSN and guaranteeing packet delivery delays are critical for achieving acceptable quality of service.

Many sensing applications share in common that their source nodes deliver packets to sink nodes via multiple hops, leading to the problem on how to find routes that enable all packets to be delivered in required time frames, while simultaneously taking into account factors such as energy efficiency and load balancing. Many previous research efforts have tried to achieve trade-offs in terms of delay, energy cost, and load balancing for such data collection tasks [6], [7]. Instead of proposing yet another protocol, our motivation for this work stems from the insight that, recent research efforts on open vehicle routing (OVR) problems are usually based on similar assumptions and constraints compared to sensor networks [8], [9], [10]. Specifically, in OVR research on goods transportation, the objective is to spread the goods to customers in finite time with the minimum amount of transportation cost. One may wonder, naturally, if we treat packet delays as delivery time of goods, and energy cost as the delivery cost of goods, it may be possible to exploit research results in one domain to stimulate the other.

Motivated by this observation, our work in this paper develops EDAL, an **E**nergy-efficient **D**elay-**A**ware **L**ifetime-balancing data collection protocol. Specifically, EDAL is formulated by treating energy cost in transmitting packets in WSNs in a similar way as delivery cost of goods in OVR, and by treating packet latencies similar to delivery deadlines. We then prove that the problem addressed by EDAL as NP-hard. To reduce its computational overhead, we introduce both a centralized meta-heuristic based on tabu search [11], and a distributed heuristic based on ant-colony gossiping, to obtain approximate solutions. Our algorithm designs also take into account load balancing of individual nodes to maximize the system lifetime. Finally, we integrate our algorithm with compressive sensing, which helps reduce the amount of traffic generated in the network. We evaluate both approaches using large-scale simulations with NS-3 [12], and present the evaluation results. More specifically, our major contributions are as follows:

- We propose a data collection protocol called EDAL, which employs those techniques developed for OVR in operations research to find the minimum cost routes to deliver packets within their deadlines. The problem formulation is proven to be NP-hard.
- To reduce the computation complexity, we introduce a centralized meta-heuristic, which employs tabu search [11] to find approximation solutions.
- We also proposed a distributed heuristic for large-scale WSN, where each source node independently forms the most energy-efficient route to forward packets.

The remaining of the paper is organized as follows. Section II includes the background introduction about open ve-

hicle routing problem, compressive sensing, and some related works. Section III describes the details about the centralized heuristic and the distributed heuristic. The simulation results on NS-3 [12] as well as comparisons with baseline protocols are presented in section IV. Finally, Section V concludes the paper.

II. Background

A. Vehicle Routing Problems

The vehicle routing problem (VRP) [8] is a well-known NP-hard problem in operational research. VRP finds routes between a depot and customers with given demands so that the transportation cost is minimized with the involvement of the minimum number of vehicles, while satisfying capacity constraints. With additional constraints, VRP can be further extended to solve different problems, where one of the most important is the vehicle routing problem with time windows (VRPTW) [9]. This problem happens frequently in the distribution of goods and services, where an unlimited number of identical vehicles with predefined capacity serve a set of customers with demands of different time intervals (time windows). VRPTW tries to minimize the total transportation cost through the minimum number of vehicles, without violating any timing constraints in delivering goods. If vehicles are not required to return back to the depot, and if the time windows are replaced by deadlines, VRPTW can be further extended to the open vehicle routing problem with time deadlines (OVRP-TD) [10].

As an NP-hard problem, OVRP-TD has inspired many heuristics. Ozyurt et al. [10] proposed the nearest insertion method, where the farthest node is chosen first to be connected with a route. Then, repeatedly, each selected node chooses the nearest neighbor that has not been assigned a route so far, and connects itself to this neighbor. This procedure repeats until all customers are connected by routes. Solomon [13] developed the push forward insertion heuristic (PFIH), which repeatedly selects the customer with the lowest additional insertion cost as the next node, until all customers are routed. Once initial routes have been found, various algorithms [14], [15], [16], [10] are developed to generate near optimal solutions based on simulated annealing [17], tabu search [11], or genetic programming [18].

B. Compressive Sensing

Once routes have been found using EDAL, we further refine the data collection efficiency through an emerging technique called *compressive sensing (CS)*. CS is a technique through which data are compressed during their transmission to a given destination, by exploiting the fact that most sensors may not always have valid data to report when they sample the environment [19], [20], [21], [22]. It works as follows: suppose that there are N nodes generating N segments of data, where the data are K-sparse, meaning only K of them are non-zero. We can compress these N pieces of data into M pieces through a linear transformation, such as Equation 1, to reduce the number of packets, where $K < M \ll N$. Formally we have:

$$y = \Phi x \tag{1}$$

where y is a $M \times 1$ column vector, Φ is a $M \times N$ matrix, and x is a $N \times 1$ column vector. As $M \ll N$, recovering x from y is an ill-posed problem. However, as long as $M \geq K \log N$, x can be accurately reconstructed with very high probability through $l_1 - norm$ minimization [23].

Because CS promises improved energy efficiency and lifetime balancing properties [21], data gathering protocols have been proposed to exploit CS for better performance. Xiang et al. [24] proposes a new data aggregation technique derived from CS to minimize the total energy consumption through joint routing and compressed aggregation. Mehrjoo et al. [25] employs compressive sensing and particle swarm optimization algorithms to build up data aggregation trees and decrease communication rate. These two methods are different from EDAL in that they involve all nodes to contribute sensing data during the data collection phase. On the other hand, Wang et al. [26] proposes random routing methods based on different network topologies to collect data from a subset of nodes, which fits to the similar application scenario as EDAL. However, EDAL achieves better energy efficiency because it optimizes on the number of constructed routes such that the total number of packets is decreased. We further compare the performance of EDAL with that reported in [26] in the evaluation section to show that a better gain in energy efficiency is achieved because it exploits the topological requirements of compressive sensing.

III. EDAL Algorithm Design

In this section, we describe the EDAL algorithm. First, we describe the problem model, and prove that the problem formulation is NP-hard. Next, we develop both centralized and distributed heuristics for obtaining approximate solutions.

A. Problem Model

Our formulation of the problem follows a similar approach with those in the literature. We assume that there are N sensor nodes deployed, which are modeled by a connectivity graph of $G = (V, E)$, where E represents wireless links between nodes. Each link is assumed to be directional, and associated with a metric q representing its link quality. To perform sensing tasks, there are M nodes selected as sources. All packets must be sent to the sink within a deadline of d. The objective function of the delivery tasks is that all packets need to be delivered with the minimum total cost. The lifetime of a node defined as the time for it to deplete its energy. A list of these definitions is shown in the Table I.

Based on these notations, for each link $l_{ij} \in E$ and each route k, we define x_{ijk} as

$$x_{ijk} = \begin{cases} 1, & \text{if route } k \text{ contains link } l_{ij} \\ 0, & \text{otherwise} \end{cases} \tag{2}$$

N	Total number of nodes
M	Total number of source nodes
E	Total number of links
K	Total number of routes
L	Maximum level of node energy
E_{max}	Total energy on each node
p_{ki}	The ith node on path k
l_{ij}	The link connecting node i and j
q_{ij}	The link quality of the link l_{ij}
c_{ij}	The weight of the link l_{ij}
t_{ij}	The time for transmitting a packet over l_{ij}
e_i	The current remaining energy of node i
l_i	The current energy level of node i
d	The delay requirement of all packets
t_i	The processing time on node i

TABLE I
NOTATIONS OF EDAL

Next we initialize c_{ij} for links with appropriate values. If the link quality is poor, then the link cost should be proportionally higher. On the other hand, to meet our goal of lifetime balancing, it is appropriate to assign a higher weight to those links connecting nodes with less remaining energy, so that they will be less frequently selected by the algorithm during execution. Based on this logic, we develop the following formula to assign c_{ij} with values:

$$c_{ij} = \frac{L - \min l_i, l_j}{q_{ij} \times q_{ji}} \tag{3}$$

where:

$$l_i = \left\lceil L \times \frac{e_i}{E_{max}} \right\rceil \tag{4}$$

where Equation 4 defines a step equation for computing the remaining energy level of node i. The ceiling value is computed to differentiate fully energy depletion and almost energy depletion. Together, Equation 3 and 4 ensure that those nodes with less remaining energy or poor communication links will have a lower chance of being chosen as forwarders.

We now formulate our optimization objective, i.e., delivering all packets to the destination under the constraint that no packet deadline is violated, as follows:

$$\min \sum_{k \in K} \sum_{i \in N} \sum_{j \in N} c_{ij} x_{ijk} \quad \text{s.t.,} \tag{5}$$

$$\sum_{j \in N} x_{j0k} = 1, \forall k \in K, \tag{6}$$

$$\sum_{i \in N} x_{ihk} - \sum_{j \in N} x_{hjk} = 0, \forall h \in (N - \{p_{k1}\}), \forall k \in K, \tag{7}$$

$$\sum_{i \in p_k, j \in p_k} t_i + \frac{t_{ij}}{q_{ij}} < d, \forall k \in K. \tag{8}$$

where the objective function 5 minimizes the total communication cost (and if two approaches lead to the same cost, the one with lower number of participation nodes should be chosen), and the constraints 6, 7, and 8 ensure that 1) all routes must end at the sink; 2) the number of routes incoming a node should be the same as the number of routes outgoing it, if the node is not located at the beginning of a route; and 3) the time for the packets being transmitted on the routes should not violate packet delay requirements.

B. Complexity Analysis

In this section, we prove that the aforementioned formulation is NP-hard.

Theorem 3.1: The problem of finding the minimum cost routes to deliver packets within their deadlines, as defined in the previous section, is NP-hard.

Proof: To prove this fact, we need to select a known NP-hard problem, and show that in a polynomial number of steps, it can be reduced to our problem. The particular NP-hard problem we select is the open vehicle routing problem with time deadlines (OVRP-TD) [10], which is a variant of vehicle routing problem with time windows (VRPTW) [9]. This problem tries to find the least cost routes from one point to a set of scattered points, and has been proved as NP-hard. Formally, this problem is defined as following: given a graph $G = (V, E)$ with $n + 1$ vertices V and a set of edges E. Let V contain 1 depot node and n customer nodes that need to be served within specified time windows. Each edge in E has a nonnegative weight, d_{ij}, and a travel time, tr_{ij}. Specifically, tr_{ij} includes the service time on node i, which we denote as ts_i, and the transportation time from node i to node j, which we denote as tl_{ij}. The objective is to minimize the total travel cost with the involvement of a small number of routes.

We now show that OVRP-TD can be reduced to our problem within polynomial steps. The graph G in OVRP-TD can be easily transformed to a corresponding sensor network topology by representing vertices with sensor nodes. The depot corresponds to the sink node, and the customers correspond to the source nodes. The cost of the edges, d_{ij}, is a little tricky to handle. Specifically, we need to solve equation 3 by adjusting the values of l_i, l_j, or the link quality q. On the other hand, however, the link quality q is related to the transmission time from i to j. That is, given tl_{ij} as a known parameter in the OVRP-TD formulation, we can obtain the appropriate value of q by enforcing that t_{ij}/q_{ij} (in WSN formulation) $= tl_{ij}$ (in OVRP-TD formulation). Recall that t_{ij} is the minimum transmission time of a packet over link l_{ij}. When links are unreliable, multiple transmissions are needed to ensure reliable delivery. Because each transmission is independent, the expected number of transmission rounds is $1/q_{ij}$. Therefore, the total transmission time is t_{ij}/q_{ij}. Since t_{ij} is a fixed parameter depending on the radio hardware and bandwidth, we can decide appropriate q_{ij} for each link based on tl_{ij}. After that, we are able to obtain the appropriate $l_{i(j)}$ values according to Equation 3.

The remaining formulation works as follows. The nonnegative transportation cost of each edge in E represents the cost of

the path connecting two source nodes with an edge. The path cost is computed based on the minimum remaining energy of the adjacent nodes. The time window for each customer is (t_s, t_d), where t_s is the start time of the window, and t_d is the end of the time window. If we set $t_s = p_i$, and $t_d = p_i + d$, where p_i is the start time of ith period, and d is the packet delay constraint, then the time window in OVRP-TD is transformed to the delay bounds in the WSN domain. In this way, we have transformed OVRP-TD to a specific case of EDAL problem formulation in polynomial time. Given that OVRP-TD is NP-hard, the problem defined by EDAL must also be NP-hard.

■

Algorithm 1 Heuristic Algorithm based on Revised Push Forward Insertion

Input: Topology graph G, the source node set S, the deadline D, and the sink node t

Output: A set of routes R with the minimum cost

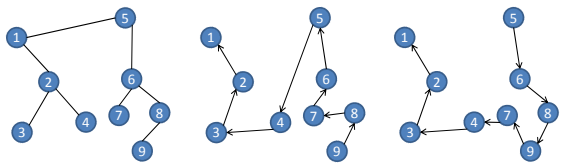
1: Set candidate list $L = \emptyset$ and $R = \emptyset$
2: Calculate the minimum path cost of all source nodes $s_i \in S$ to the sink t using the Dijkstra's algorithm
3: Put all nodes in the source set S into the candidate list L
4: Find the node s_{new} that has the maximum path cost to the sink from L, and assign the global variable $s_m = s_{new}$
5: **while** $L \neq \emptyset$ **do**
6: Remove the node s_{new} from L
7: **for all** node $s_i \in L$ **do**
8: Compute the partial delay d_p from s_m to s_{new} through the route constructed so far
9: Compute the total delay $d_{total} = d_p + \text{DELAY}(s_{new}, s_i) + \text{DELAY}(s_i, t)$
10: Compute the insertion cost as $\text{PATHCOST}(s_{new}, s_i) + \text{PATHCOST}(s_i, t)$
11: If the insertion cost is the lowest, and the delay $d_{total} \leq D$, pick s_i as s_{new}
12: **end for**
13: **if** No candidate s_{new} is found **then**
14: Put the currently found route into R
15: Start a new route construction procedure
16: **end if**
17: **end while**
18: Return R as the output

C. Centralized Heuristics

Given that we have proven the problem of data collection with deadline constraints as NP-hard, we now present heuristic solutions to reduce its computational overhead. In this section, we propose a centralized meta-heuristic that employs tabu search [11] to find approximation solutions. We assume that M nodes have been selected as sources at the beginning of each data collection period. The heuristic algorithm consists of two phases: route construction, which finds an initial feasible route solution, and route optimization, which improves the initial results using the tabu-search optimization technique.

In the route construction phase of this algorithm, we present a heuristic algorithm based on the revised push forward insertion (RPFIH) method, as shown in Algorithm 1. The original push forward insertion algorithm was proposed by [13], and we modify it to fit the needs of wireless sensor network. At the beginning of RPFIH, for each node, the minimum cost path to the sink is found and saved. RPFIH then finds the node that has the largest path cost to the sink, and incrementally selects candidate nodes with the lowest additional insertion cost. For each candidate node, RPFIH also checks its feasibility by making sure that the overall delay requirement is met. If no candidate node can guarantee the delay, RPFIH initializes a new route with the node that has the largest path cost to the sink in the remaining sources, and repeats this process until all sources are connected with the sink. Finally, RPFIH generates a set of routes as the final output.

We now analyze the time complexity of RPFIH. As the Dijkstra's algorithm is used, it takes $O(E \log N)$ time to find a minimum weight path between two nodes. In RPFIH, a maximum number of $\frac{(M-1)\times M}{2}$ paths between source nodes need to be computed. Therefore, the overall time complexity is $O(M^2 E \log N)$.



(a) MST (b) R (c) R^*

Fig. 1. The worst case and optimal solution of RPFIH.

Next, we demonstrate the following result concerning the approximation ratio of RPFIH.

Theorem 3.2: RPFIH is a polynominal-time 2-approximation algorithm for the VRPTW.

Proof: We have already shown that RPFIH is a polynominal-time algorithm with time complexity as $O(M^2 E \log N)$. Let R^* denote the optimal routes for the given set of source nodes, and R denote the routes generated by RPFIH. When the delay bound is very tight, each source node must follow the minimum cost shortest path toward the destination. In that case, we can expect that approximately R^* = R, and $C(R^*)$ = $C(R)$, where C represents the total cost of the routes.

On the other hand, when the delay bound is very loose, VRPTW is equivalent to VRP. When the vehicle capacity is not restricted, the lower bound on the cost of an optimal route is the weight of the minimum spanning tree T [27] of source nodes, where $C(T) \leq C(R^*)$. On the other hand, in the worst case, we can observe that R becomes a pre-order tree walking of T, while the insertion cost of nodes are ordered in the pre-order tree walking sequence, as shown in Figure 1. Since a full walk W will travel through every edge of T exactly twice, we know that $C(W) = 2C(T) \leq 2C(R^*)$. As R is a route equals to W deleting the last link, we have $C(R) \leq C(W) \leq$

$2C(R^*)$. Hence, RPIFH is a 2-approximation algorithm. ■

Algorithm 2 The Centralized Heuristic Algorithm in EDAL

Input: The list of routes R from RPFIH
Output: A list of optimized routes OR

```
Initialize Tabu move list ML = ∅ and candidate list CL = {R}
while Total number of steps is less than M: do
    Perform λ-interchange LSD based intensification on each route in CL
    if A better route is found: then
        Record the partial solution into R
    else
        Perform λ-interchange LSD based diversification on each route in CL
    end if
end while
Output the best solution found so far in R
```

While RPFIH generates a list of routes, they are by no means optimal in the sense of the overall cost and delay. We further optimize the initial solution using tabu search. Tabu search is a popular memory-based search strategy for guiding local search beyond local optimum points.

In our tabu search implementation, we adopt the λ-interchange local search descent (LSD) method, which uses a systematic insertion and swapping of nodes between routes to produce mutations of the current solution. Up to λ nodes can be exchanged. For example, if $\lambda = 2$, a total of eight interchange operations are possible, including $(0,1)$, $(1,0)$, $(1,1)$, $(0,2)$, $(2,0)$, $(1,2)$, $(2,1)$ and $(2,2)$, where (i,j) means to choose i nodes in route r_1 and swap it with j nodes in route r_2, while r_1 and r_2 may not necessarily be different. The tabu search exploits LSD in two steps: intensification and diversification. In intensification, the algorithm implements the 2-interchange LSD procedure on each route individually to find the best potential order of nodes. The diversification step enables the algorithm to search out of the local optimum by making random 2-interchange operations between routes so that better routes that are combinations of the original ones can be found. The detailed steps of this algorithm are shown in Algorithm 2.

D. Distributed Heuristics

One problem with the centralized heuristic algorithm we have developed in EDAL is that it requires information to be collected from each node to a centralized one. In distributed sensor networks, this step will typically incur additional overhead. Therefore, it is usually desirable to distribute the algorithm computation into individual nodes. In this section, we develop a distributed heuristics algorithm for EDAL, where at the beginning of each period, each source node independently chooses the most energy-efficient route to forward packets.

Our developed algorithm is based on the ant colony optimization [28] and geographic forwarding. It consists of two phases: status gossiping, and route construction. In the status gossiping phase, each source node sends forward ants spreading its current status, including its remaining energy level, toward its neighbor source nodes within H hops. Meanwhile, the status data of nearby nodes are collected by each source node with the received backward ants. During the gossip phase, the ants are forwarded with a modified geographic forwarding routing protocol, which chooses the node with the maximum remaining energy while making geographical progress towards the destination as the next hop. Once a node collects status information of all its nearby sources, it enters the route construction phase, and runs RPFIH distributedly based on collected nearby neighbor status, and the estimation of node status outside the immediate neighborhood.

More specifically, the algorithm works as follows. Each node knows the random seeds of all the nearby nodes. At the beginning of each period, each source node predicts which nearby nodes to be the source nodes, based on the given random seeds. Then the node generates forward ants (an ant is implemented as one or more packets) targeting at each of its nearby source nodes. The role of the forward ant is that it explores the path and collects information along the travel, and the role of the backward ant is that it travels back to the source node and informs their pass-by nodes to update their knowledge with the collected information.

When a relay node gets a forward ant, it selects the neighbor node with the most remaining energy to make progress to the destination as the next hop, and sends the ant out. The forward ant collects the information of the status and remaining energy level of each encountered node along the path. The backward ant is released when one of three cases happens: first, if the forward ant meets another ant sent from other source nodes, where they exchange information with each other immediately; second, if the initial target of the ant has been reached, and it is found to be a source node; third, if the initial target is reached, but it is not a source node. Instead, a newly picked one along the path is. In any of these cases, the backward ant will be sent along the traveled path of the forward ant, and each node along the path will be updated by the collected information carried by the backward ant. With ant colony gossip, one advantage is that we can now shrink the information collected by nodes by making the collected status more relevant. The computation complexity of ant-colony based gossip is at most $4H^2$ in the worst case, where H is the maximum number of hops in the gossip range.

At the end of gossip phase, each source node s collects a list of source nodes, S, and the cost of e edges W_s. Each node in S can be inserted into the route in the route construction phase. For W_s, it contains the costs of e edges traveled by all ants sent or received by node s.

The route construction phase is based on the RPFIH introduced in the previous section. For each source node, it triggers the RPFIH if no other nearby source node with a longer distance to the sink is detected from the ant-colony gossip phase. As all nodes start with the same amount of energy, the source node can accurately estimate the status of nearby nodes. In that case, the minimal weight path from a source node to

Algorithm 3 The Distributed Heuristic for EDAL

Input: Topology graph G, the source node s, the neighbor source node set S, the deadline D, and the sink node t

Output: Constructed routes with $s_i \in S$ with the minimum insertion cost such that D is not violated

1: Run the ant-colony based gossip to collect neighborhood status
2: Estimate the minimum path cost of s and all $s_i \in S$ to the sink t using the Dijkstra's algorithm
3: Put all nodes in the source set S into the candidate list L
4: **if** $\forall$DISTANCE$(s,t) >$ DISTANCE(s_i,t) **then**
5: goto 14
6: **end if**
7: **if** Route construction packet rc is received **then**
8: Extract partially constructed route pr from rc, and partial delay d_p of pr
9: **if** s is a part of the other route **then**
10: Send a packet to inform the previous source node, and terminate
11: **end if**
12: Remove $n \in pr$ from S, and goto 14
13: **end if**
14: **for all** Node $s_i \in L$ **do**
15: Compute the total delay $d_{total} = d_p +$ DELAY$(s,s_i) +$ DELAY(s_i,t)
16: Compute the insertion cost as PATHCOST$(s,s_i) +$ PATHCOST(s_i,t)
17: If the insertion cost is the lowest, and the delay $d_{total} \leq D$, append s_i to pr
18: Send a construction packet to s_i with payload pr and $d_p = d_p +$ DELAY(s,s_i)
19: **end for**
20: **if** No candidate s_i is found **then**
21: Choose t as the next node, and send construction packet to t
22: Send construction packets with empty route to each $s_i \in S$
23: **end if**

a nearby source node can be calculated with the currently held information. The tricky part is how to find the minimal weight path to the sink, so that the source can examine if the newly formed route violates the delay constraint. We solve this problem by letting each source node first compute the minimal weight path to each of the nodes on the border of its gossip range, and estimate the weight of the path from that node to the sink, so that it can choose the one with minimal total path weight. Assume that there is a path p, which takes n hops from node s to the sink, from the gossip phase, node s knows the edge weight of the first k hops as c, then the cost of the whole path is computed as:

$$C_p = \sum_{1<w<k} c_w + (n-k) * \frac{\sum_{w \in W_s} W_{sw}}{e} \qquad (9)$$

That is, by using the number of hops and the average cost per link, the source can estimate the whole path cost from itself to the sink. The overall distributed algorithm is shown in Algorithm 3. Note that here, a source node i will be triggered to select the next target node by either receiving a route construction packet or being selected as the farthest node to sink. The algorithm terminates after all source nodes are included into their own route. In the route construction phase, only the neighborhood statuses are taken into consideration for finding the minimum cost path between nearby source nodes. Therefore, the computation complexity for Dijkstra's algorithm is reduced to $O(H^2 \log H^2)$. Overall, the total complexity of the distributed heuristic is $O(H^2) + O(H^2 \log H^2)$, which is $O(H^2 \log H^2)$, where H is the size of gossip range in the number of hops.

IV. System Performance Evaluation

To evaluate EDAL, we implement both the centralized heuristic (C-EDAL) and the distributed heuristic (D-EDAL) described in Section III, and compare their performance in terms of network lifetime, selected nodes, and packet delay, with and without the integration of compressive sensing, to two selected baselines. The network lifetime is defined as the time for critical nodes deplete their energy in the network. The details are shown in the following parts of this section.

A. Experimental Settings

In order to understand the network performance of EDAL under different delay requirements, we simulate our design in NS-3 [12]. In the simulation, a uniform network topology with 256 IRIS nodes with CPU speeds as 8 MHz. Each sensor node is equipped with two AAA batteries. The node density is 16 nodes in 40 meter by 40 meter area. On average, each node has at least four adjacent neighbors to communicate with. To accurately reflect true radio properties, we adopt unreliable links in our simulations. The link quality of all links is set to be 0.9 for our comparison purposes. While we acknowledge that more complicated radio communication patterns can be used, even adopting this relatively simple radio model is already sufficient to demonstrate the performance differences between our approach and alternative baselines. The sink node is placed on the top of the grid. We assume that the data collection task has been deployed, and it executes periodically with a period length of 2 minutes. At the beginning of each period, a random collection of nodes are selected as sources. Once they have sensing data to report, either C-EDAL or D-EDAL is triggered to generate new routes based on current selection of source nodes. In D-EDAL, we set the gossip range to be 3 hops.

We compare the performance of C-EDAL and D-EDAL with the following two routing baselines:

- Minimum spanning tree (MST) routing: this is a widely used, conventional routing algorithm of WSNs, where a minimum spanning tree is constructed for collecting data to the sink.

- Location-aware random routing (LRR) [26]: this algorithm works in a similar way to EDAL in the sense that it also focuses on collecting data from a subset of sources. It also integrates a level of randomness in its design, which makes the comparison particularly interesting since EDAL exploits AI based search to introduce similar randomness.

As the goal of EDAL is to connect all source nodes with minimum total cost under the constraint that it should guarantee packet delay requirements and lifetime balancing, we compare EDAL with baselines on a set of metrics as network lifetime, the number of nodes selected to form routes, and packet delays. We also integrate compressive sensing with EDAL to achieve a better gain in energy efficiency, by exploiting the topological requirements of compressive sensing.

B. Algorithm Overhead

Before we look into the energy efficiency and lifetime balancing performance of EDAL on the experimental network, in this section, we evaluate the average time consumed to finish one round of algorithm computation, to show the scalability and practicability of our algorithm. The time overhead is computed as the needed CPU cycles for running the algorithm divide the desktop or sensor node CPU speed.

We first discuss the scalability of the centralized heuristic, based on time overhead vs. network size, as shown in Figure 2. This analysis provides a sense of the feasibility for implementing the centralized heuristic in a real sensor network. The centralized heuristic provides better performance; however, it is infeasible to run the centralized heuristic for large topologies with more than 400 nodes, as the computation takes longer time than the communication period.

We also measured the average time used by each source node for building the routes distributedly, based on time overhead vs. gossip range, as shown in Figure 3. The computation overhead of the distributed heuristic on each node is tightly correlated with the gossip range, while the algorithm completion time is tightly correlated with network size. In that case, we collected the time for finishing algorithm computation on each node with different gossip range on a uniform network with 256 nodes. It is easy to understand that the larger the gossip range, the more network status can be collected. However, this also leads to longer time to finish computation. In general, there is too much overhead for a sensor node to have a gossip range larger than 5. In this section, we choose the gossip range as 3.

C. Experiment Results

1) EDAL without CS: To evaluate the energy efficiency and lifetime balancing effects of EDAL, we first run a set of experiments with EDAL running alone under different delay requirements. The delay bounds of packets are set to be 45 ms, 67.5 ms, 90 ms, 112.5 ms, 135 ms, 157.5 ms, 180 ms, and 202.5 ms separately for each run of experiments. As the chosen baselines do not take delay requirements into consideration, their results do not have big variations under different delay bounds.

As one goal of EDAL is to connect source nodes with minimal number of relay nodes, we collect the number of nodes used to form routes under different delay bounds, as shown in Figure 4. Observe that the average node number used by MST and LRR are almost constant. For MST, this is because the routing tree is fixed, and the source nodes are randomly selected with a constant probability. For LRR, this is because the routes are randomly constructed based on predefined rules; therefore, the ending result does not have considerable fluctuations, either. For C-EDAL and D-EDAL algorithms, on the other hand, the number of participating nodes is decreasing with the increasing of delay constraints. This is because as the delay requirement is relaxed, more source nodes can be added into the same route, a feature that is made possible by the tabu search and ant-colony algorithms adopted by EDAL. As a result of that, fewer and fewer nodes are selected due to triangle inequality, where relay nodes can serve more source nodes simultaneously.

To explicitly show the energy efficiency performance of our design, we measure the average energy consumption of the whole network in each period, as shown in Figure 5. We can observe that C-EDAL and D-EDAL consume less energy on average compared to the two baselines. This is because on average they are using fewer nodes to transmit packets in each period.

As the average number of nodes used by MST stays constant under different delay bounds, the network lifetime of MST remains the same as well. As a result, we take the network lifetime of MST as the standard unit, and for each routing algorithm, we compute the ratio of its network lifetime to the standard MST network lifetime. The larger this ratio, the longer the lifetime is. As shown in Figure 6, the network lifetimes under C-EDAL and D-EDAL are increasing considerably with the delay bounds, while the network lifetime of LRR remains almost constant. This is expected, because as the delay requirements are relaxed, fewer number of nodes are selected for routing as shown in Figure 4. In that case, the total energy consumption also decreases accordingly. In general, compared to MST, C-EDAL increases the overall system lifetime by up to 59.4%, and D-EDAL increases the overall lifetime by up to 54.8%. On the other hand, compared to LRR, C-EDAL increases the lifetime by up to 15.4%, and D-EDAL increases the lifetime by up to 12.1%. The lifetime gains of EDAL over LRR are not very large because in LRR, preliminary optimization has already been applied to filter out those inefficient routes. On the other hand, EDAL takes the delay bound into consideration, such that it is likely to choose the shortest paths especially when delay bound is tight. That is why the network lifetime is comparably shorter with tight delay bounds.

To show that EDAL also meets delay constraints, we set the delay bound of data collection tasks to be 90 ms, and measure the overall packet delay for C-EDAL, D-EDAL and the two baselines. As shown in Figure 7, the packet delay of MST is

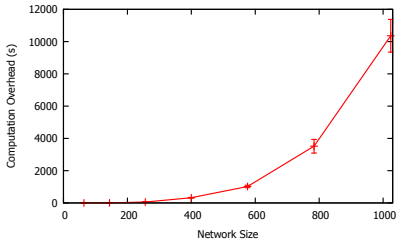


Fig. 2. Time overhead of the centralized heuristic under different network sizes

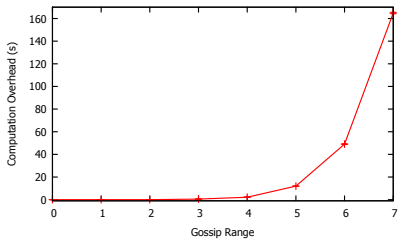


Fig. 3. Time overhead of the distributed heuristic with different gossip ranges

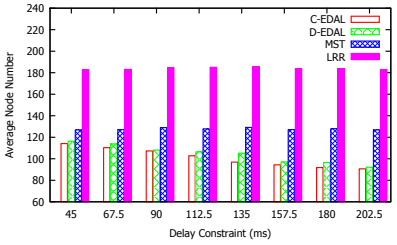


Fig. 4. The average node number used in each period by different algorithms

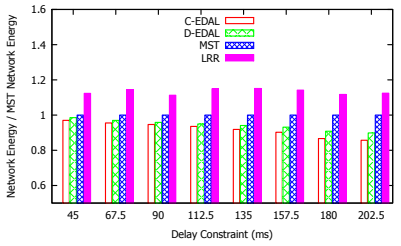


Fig. 5. The average energy consumption of the network

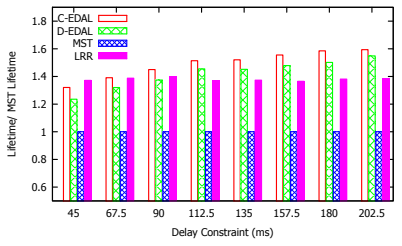


Fig. 6. The network lifetime while running different routing algorithms

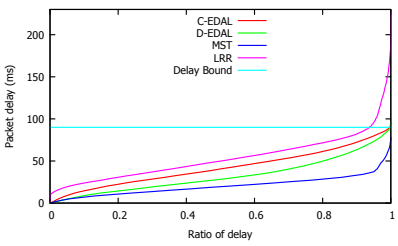


Fig. 7. The CDF of delay of packets generated in the whole network duration

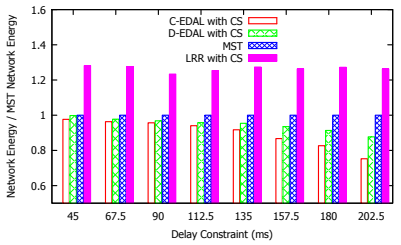


Fig. 8. The average energy consumption of the network with CS

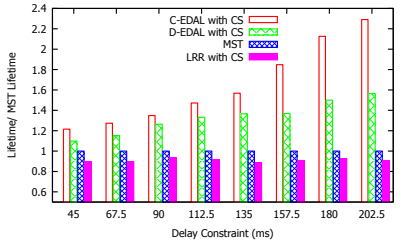


Fig. 9. The network lifetime with CS implemented on different routing algorithms

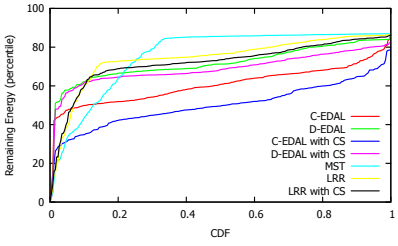


Fig. 10. The CDF of remaining energy of each node while running different routing algorithms

the shortest, as the packets take the fewest number of hops to the sink along the routing tree. However, as LRR does not take delay as a constraint when generating random routes, 6.4% of packets violates their delay requirements, and sometimes, the actual delay could be very large. On the other hand, only 0.3% of packets violates delay constraint in C-EDAL because of the unreliable links. D-EDAL works a little worse than C-EDAL to have 0.5% of packets violating the delay constraints, because of the limited gossiping ranges.

2) EDAL with CS: Our implementation of EDAL also integrates compressive sensing since this will provide us with better energy efficiency and lifetime balance. In compressive sensing, sparse data are compressed into a small number of packets.

Assume that the sparsity of data is known ahead, we evaluate the network lifetime of EDAL and two baselines while changing the packet delay constraints. To measure energy efficiency performance of EDAL with CS, we also measure the average energy consumption of the whole network in each period. Compare Figure 8 and Figure 5, we can observe that EDAL with CS consumes less energy on each period than pure EDAL with loose delay bounds. This is because CS enables the network to transmit fewer packets. Figure 9 shows the network lifetime of C-EDAL with CS, D-EDAL with CS, MST, and LRR with CS. The lifetime of MST is still used as the standard unit. In general, the network lifetime increases with more relaxed packet delay constraints. In fact, compared to MST, the network lifetime is increased by up to 129.1% for C-EDAL with CS, and up to 56.5% for D-EDAL with CS. This is because EDAL considers generating the minimal number of routes as one of its primary optimization goals. On the other hand, when the network delay increases, more source nodes can be added into the same path, which results in a further decrease of the number of routes. As a result, the overall relay nodes involved decrease in number, so does the number of packets transmitted by the shared nodes. In contrast, the lifetime for LRR with CS is decreased by 47.1% on average compared to MST. This is because each source node tries to independently generate a random path towards the sink in LRR, causing the nodes near the sink to be shared by many routes. This phenomenon causes such nodes to transmit more packets in total than LRR without CS, which also explains why the lifetime of EDAL with CS is shorter in tight delay bounds compared to EDAL without CS implemented.

To measure the load balancing feature of EDAL, we fix the delay bound to be 180 ms, and collect the CDF of remaining energy on each node at the end of the simulation. As shown in Figure 10, as enabled by the load balancing performance of CS, we can observe that the curves of EDAL are more gently than the curves of the two baselines. This shows that EDAL performs better on load balancing than baselines. Finally, EDAL has less remaining energy because it runs for more periods.

V. Conclusion

In this paper, we propose EDAL, an Energy-efficient Delay-Aware Lifetime-balancing protocol for data collection in wireless sensor networks, which is inspired by recent techniques developed for open vehicle routing problems with time deadlines (OVRP-TD) in operational research. The goal of EDAL is to generate routes that connect all source nodes with minimal total path cost, under the constraints of packet delay requirements and load balancing needs. The lifetime of the deployed sensor network is also balanced by assigning weights to links based on the remaining power level of individual nodes. We prove that the problem formulated by EDAL is NP-hard, therefore, we develop a centralized heuristic to reduce its computational complexity. Furthermore, a distributed heuristic is also developed to further decrease computation overhead for large scale network operations. Based on the simulation results, we observe that compared to baseline protocols, EDAL achieves a significant increase on network lifetime without violating the packet delay constraints. Finally, we demonstrate that by integrating compressive sensing with EDAL, additional lifetime gains can be achieved.

Acknowledgment

The work reported in this paper was supported in part by the National Science Foundation grant CNS-0953238, CNS-1017156, CNS-1117384, and CNS-1239478.

References

[1] G. Tolle, J. Polastre, R. Szewczyk, D. Culler, N. Turner, K. Tu, S. Burgess, T. Dawson, P. Buonadonna, D. Gay, and W. Hong, "A macroscope in the redwoods," in *Proceedings of the 3rd international conference on Embedded networked sensor systems*, ser. SenSys '05. New York, NY, USA: ACM, 2005, pp. 51–63.

[2] G. Werner-Allen, K. Lorincz, J. Johnson, J. Lees, and M. Welsh, "Fidelity and yield in a volcano monitoring sensor network," in *Proceedings of the 7th symposium on Operating systems design and implementation*, ser. OSDI '06. Berkeley, CA, USA: USENIX Association, 2006, pp. 381–396.

[3] M. Li and Y. Liu, "Underground coal mine monitoring with wireless sensor networks," *ACM Trans. Sen. Netw.*, vol. 5, pp. 10:1–10:29, April 2009.

[4] P. Vicaire, T. He, Q. Cao, T. Yan, G. Zhou, L. Gu, L. Luo, R. Stoleru, J. A. Stankovic, and T. F. Abdelzaher, "Achieving long-term surveillance in vigilnet," *ACM Trans. Sen. Netw.*, vol. 5, pp. 9:1–9:39, February 2009.

[5] N. Xu, S. Rangwala, K. K. Chintalapudi, D. Ganesan, A. Broad, R. Govindan, and D. Estrin, "A wireless sensor network for structural monitoring," in *Proceedings of the 2nd international conference on Embedded networked sensor systems*, ser. SenSys '04. New York, NY, USA: ACM, 2004, pp. 13–24.

[6] L. Liu, X. Zhang, and H. Ma, "Optimal node selection for target localization in wireless camera sensor networks," *Vehicular Technology, IEEE Transactions on*, vol. 59, no. 7, pp. 3562 –3576, sept. 2010.

[7] "Sensor selection for parameterized random field estimation in wireless sensor networks," *Journal of Control Theory and Applications*, vol. 9, pp. 44–50, 2011.

[8] B. Eksioglu, A. V. Vural, and A. Reisman, "The vehicle routing problem: A taxonomic review," *Computers and Industrial Engineering*, vol. 57, no. 4, pp. 1472 – 1483, 2009.

[9] O. Bräysy and M. Gendreau, "Vehicle routing problem with time windows, part i: Route construction and local search algorithms," *Transportation Science*, vol. 39, no. 1, pp. 104–118, Feb. 2005.

[10] Z. Ozyurt, D. Aksen, and N. Aras, "Open vehicle routing problem with time deadlines: Solution methods and an application," in *Operations Research Proceedings 2005*, ser. Operations Research Proceedings, H.-D. Haasis, H. Kopfer, and J. Schnberger, Eds. Springer Berlin Heidelberg, 2006, vol. 2005, pp. 73–78.

[11] K. Tan, L. Lee, Q. Zhu, and K. Ou, "Heuristic methods for vehicle routing problem with time windows," *Artificial Intelligence in Engineering*, vol. 15, no. 3, pp. 281 – 295, 2001.

[12] "Ns-3," Tech. Rep. [Online]. Available: http://www.nsnam.org/

[13] M. M. Solomon, "Algorithms for the vehicle routing and scheduling problems with time window constraints," *Oper. Res.*, vol. 35, no. 2, pp. 254–265, 1987.

[14] L. Du and R. He, "Combining nearest neighbor search with tabu search for large-scale vehicle routing problem," *Physics Procedia*, vol. 25, no. 0, pp. 1536 – 1546, 2012.

[15] C.-B. Cheng and K.-P. Wang, "Solving a vehicle routing problem with time windows by a decomposition technique and a genetic algorithm," *Expert Systems with Applications*, vol. 36, no. 4, pp. 7758 – 7763, 2009.

[16] W.-C. Chiang and R. Russell, "Simulated annealing metaheuristics for the vehicle routing problem with time windows," *Annals of Operations Research*, vol. 63, pp. 3–27, 1996.

[17] C. C. Skiścim and B. L. Golden, "Optimization by simulated annealing: A preliminary computational study for the tsp," in *Proceedings of the 15th conference on Winter Simulation - Volume 2*, ser. WSC '83. Piscataway, NJ, USA: IEEE Press, 1983, pp. 523–535.

[18] J. H. Holland, *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control and Artificial Intelligence*. Cambridge, MA, USA: MIT Press, 1992.

[19] C. Caione, D. Brunelli, and L. Benini, "Distributed compressive sampling for lifetime optimization in dense wireless sensor networks," *Industrial Informatics, IEEE Transactions on*, vol. 8, no. 1, pp. 30 – 40, feb. 2012.

[20] J. Wu, "Ultra-lowpower compressive wireless sensing for distributed wireless networks," in *Military Communications Conference, 2009. MILCOM 2009. IEEE*, oct. 2009, pp. 1 –7.

[21] G. Cao, F. Yu, and B. Zhang, "Improving network lifetime for wireless sensor network using compressive sensing," in *High Performance Computing and Communications (HPCC), 2011 IEEE 13th International Conference on*, sept. 2011, pp. 448 –454.

[22] C. Luo, J. Sun, F. Wu, and C. W. Chen, "Compressive data gathering for large-scale wireless sensor networks," in *in Proc. ACM Mobicom09*, 2009, pp. 145–156.

[23] D. Donoho and Y. Tsaig, *Fast solution of l1-norm minimization problems when the solution may be sparse*. Department of Statistics, Stanford University, 2006.

[24] L. Xiang, J. Luo, and A. Vasilakos, "Compressed data aggregation for energy efficient wireless sensor networks," in *Sensor, Mesh and Ad Hoc Communications and Networks (SECON), 2011 8th Annual IEEE Communications Society Conference on*, june 2011, pp. 46 –54.

[25] S. Mehrjoo, J. Shanbehzadeh, and M. Pedram, "A novel intelligent energy-efficient delay-aware routing in wsn, based on compressive sensing," in *Telecommunications (IST), 2010 5th International Symposium on*, dec. 2010, pp. 415 –420.

[26] X. Wang, Z. Zhao, Y. Xia, and H. Zhang, "Compressed sensing for efficient random routing in multi-hop wireless sensor networks," in *GLOBECOM Workshops (GC Wkshps), 2010 IEEE*, dec. 2010, pp. 266 –271.

[27] T. H. Cormen, C. Stein, R. L. Rivest, and C. E. Leiserson, *Introduction to Algorithms*, 2nd ed. McGraw-Hill Higher Education, 2001.

[28] G. Chen, T.-D. Guo, W.-G. Yang, and T. Zhao, "An improved ant-based routing protocol in wireless sensor networks," in *Collaborative Computing: Networking, Applications and Worksharing, 2006. CollaborateCom 2006. International Conference on*, nov. 2006, pp. 1 –7.