Collecting Data in Ad-Hoc Networks with Reduced Uncertainty

Liron Levin* Alon Efrat† Michael Segal, *Senior Member**

*Dept. of Communication Systems Engineering, Ben-Gurion University, Israel.
Emails: levinlir@gmail.com, segal@cse.bgu.ac.il

†Dept. of Computer Science, The University of Arizona, USA.
Email: alon@cs.arizona.edu

***Abstract*—We consider the data gathering problem in wireless ad-hoc networks where a data mule traverses a set of sensors, each with vital information on its surrounding, and collects their data. The mule goal is to collect as much data as possible, thereby reducing the information uncertainty, while minimizing its travel distance. We show that the problem is solvable by a generalized version of the Prize Collecting Steiner Tree Problem, and present a dual-primal 6-approximation algorithm for solving it. Simulation results show that the proposed schema converges to the optimal results for varying set of topologies, such as grids, stars, linear and random networks.**

I. Introduction

Ad-hoc networks have become a popular research area both theoretically [1] and practically [2]–[4]. One of the most interesting and practical class of applications for such networks is agriculture and environmental monitoring [5], [6], where wireless nodes are scattered over a geographical area and form a dynamic, infrastructure-less ad-hoc network. Typical monitoring scenarios consist of two stages. In the first step, the sensors periodically sense their surroundings, collect data and process it if needed. In the second step, the collected data is delivered to a base station using either *local* or *global* data collection process. In local data collection [7], the nodes use multi-hop communication over a static [8] or dynamic [9] topology to relay their message to the base station. This communication schema is most suitable when the distribution of nodes is dense, since the close proximity between nodes reduces the cost of transmitting and receiving messages [10].

In global data collection [7], [11]–[14], the nodes are visited by a traveling mule, which uses short-range wireless communication to collect data from nearby nodes. The mule aggregates the information received from the nodes and delivers it to the base station. The data mule approach is most suitable when the network topology is sparse, the distance to the nearest node or base station is too large, or when the communication infrastructure between nodes is unstable. In addition, this approach reduces the responsibility for message routing from the nodes thereby minimizing the processing power.

Most papers that use the data collector mule focus on energy efficiency [14], [15] or travel time optimization [16] without considering the quality of the collected data. Since data in each node is closely linked to the monitored area, deciding which data is significant and which is redundant can contribute greatly to the performance of the data collecting algorithm. For example, consider an environmental monitoring system with large distribution of sensors in a specific geo-location. The added knowledge from visiting one node in the area might contribute significantly to the overall understanding of the environment. However, the knowledge learned by visiting additional nodes in nearby locations may not be worth the time and transmission cost of the visit. Thus, the value from visiting a node is not constant and depends on the nodes that were visited before it. In this paper we study a dual-objective optimization problem, where the goal is to minimize the mule's traveling distance and the amount of information uncertainty, which depends on the total information collected so far.

Our contribution. We present a framework for solving the mule problem when data inside nodes is correlated. Our novel technique is used to find a tour or a tree under varying objective functions with 6-approximation ratio guarantee.

Outline of the paper. The rest of the paper is organized as follows: In Section II, we present the system model, provide motivation and present the Mule Tree Problem (MTP) and Mule Cycle Problem (MCP). We show a dual-primal formulation of the problem in Section III and discuss about background and previous work In Section IV. Our 6-approximation algorithm for MTP and MCP is developed in Sections V and VI, and in Section VII we show how to apply the algorithms for multiple optimization functions. We evaluate the performance of the algorithms against several common algorithms in Section VIII and conclude in Section IX.

II. Network Model

Let $G = (V, E, r)$ be a complete graph in the Euclidean plane, where V is the set of wireless nodes ($|V| = n$), E is the set of undirected edges between nodes ($|E| = n^2$), and r is the root. For each $v \in V$, let $i(v) \in \mathbb{R}_+$ be the amount of data sensed by v and for each $e \in E$, let c_e be the cost of traversing edge e. Consider a mule that visits a subset of nodes,$V_m \subseteq V$, to collect their information over some edges E_m starting from r. The mule can decide to skip over some nodes, $\overline{V_m}$ and to absorb a penalty $\varphi(\overline{V_m}) \in \mathbb{R}_+$. The value of the penalty reflects the data the mule didn't cover. A natural

question arises whether the mule should visit all nodes, or skip over some and absorb the penalty. We ask how to find a tour that is not too long, but contains enough monitored data. Formally, the problem is defined as follows:

The Mule Tree/Cycle Problem (MTP/MCP)
Input: Graph $G = (V, E)$, cost c_e per edge, information sensed by each node $i(v)$ and a root r
Output: A tree/tour $T_{mule} = (V_m, E_m)$
Objective: $\min(\varphi(\overline{V_m}) + \sum_{e \in E_m} c_e)$

In our problem φ represents the penalty function of not visiting $\overline{V_m}$. We present a framework for approximating the mule problem for a variety of penalty functions. The approximation holds as long as the following condition is fulfilled:

Condition 1: For any two disjoint node sets, S_i and S_j, $\varphi(S_i) \leq \varphi(S_i \cup S_j)$.

Specifically, we show that different penalty functions cover a variety of practical scenarios and present two concrete examples for the mule problem:

Temperature monitoring. Let $i(v)$ be the average temperature in the area covered by v and $T = (V, E^{'}, r)$ be a communication tree rooted at r, which is used to relay temperature samples between adjacent nodes. We assume channel capacity is limited so transmitting large messages in T is not possible. Let $d(v)$ be the set of the descendents of v in T, because of the capacity limitation, v can only send $i(v)$ and its descendant data $\sum_{u \in d(v)} i(u)$ to its parent in T. We define this process as data aggregation. After data aggregation is completed by all nodes, each node v maintains the aggregated information, $\mathcal{I}(v) = \sum_{u \in d(v)} i(u) + i(v)$, which represents what v knows about the temperature in its surrounding. When the mule visits a node v, it can learn the information stored in it $i(v)$ and the aggregated information $\mathcal{I}(v)$. From this information, the mule can infer on the data discrepancy between v's data to its descendants that was visited, $\mathcal{U}(v) = \mathcal{I}(v) - \sum_{u \in d(v) \cap V_m} \mathcal{I}(u)$. Since $\mathcal{U}(v)$ decreases as the amount of data we collect increases, we define this measure as information uncertainty. $\mathcal{U}(v)$ is illustrated in Figure 1. In Figure 1a, we have the input tree T, visited nodes $V_m = \{r, v_1, v_2, v_3\}$, and $d(r) = \{v_1, v_2, v_3\}$. The amount of information per node v, $\mathcal{I}(v)$, is depicted in Figure 1b. Intuitively, by visiting only 3 nodes, we learn the temperature of almost the entire area.

As the number of visited nodes becomes higher, the amount of uncertainty decreases. Specifically, the following penalty functions can be used to minimize the uncertainty:

$$\varphi_1(\overline{V_m}) = \frac{\sum_{v \in V_m} \mathcal{U}(v)^2}{|V|}$$

$$\varphi_2(\overline{V_m}) = \max_{v \in V_m} \mathcal{U}(v)$$

The first function guarantees that on average we learn sufficient information from each node (w.l.o.g the normalization factor is removed to simplify the calculation). The second function guarantees that sufficient information is known on every area in the graph. We provide an example for MCP with penalty function $\varphi_1(\overline{V_m})$. The input communication graph $G = (V, E)$ with 4 nodes, r, v_1, v_2 and v_3, is depicted in Figure 1c. the tree T, on which the messages get aggregated, is depicted by the non dotted directed edges in the figure. The dotted edges does not belong to T but can be used by the mule to traverse the graph. Each edge is marked with the Euclidean cost of traversing it. The Euclidean area each node covers $\mathcal{I}(v)$, is depicted by surronding circles in Figure 1d. The cost of selecting each subset of nodes S is depicted in Table I. The optimal mule tour is traversing nodes r and v_2, with total cost of 88.

Military surveillance system. We explore the following penalty function:

$$\varphi_3(\overline{V_m}) = \sum_{v \in \overline{V_m}} f(v)$$

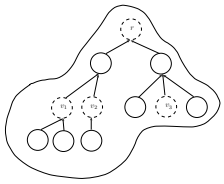
Where $f(v)$ is a monotonic increasing function. Given n surveillance sensors partitioned to k different areas. We would like to find a mule tour that visits as many areas as possible. We use φ_3 and set $f(v)$ to $\alpha \in \mathbb{R}_+$ if all nodes from specific area belong to $\overline{V_m}$or 0 otherwise. The penalty function ensures that sufficient area is under surveillance. Another option is to set $f(v) = |\overline{V_m}| i(v)$, and then we get, $\varphi_3(\overline{V_m}) = |\overline{V_m}| \sum_{v \in \overline{V_m}} i(v)$, which ensures that in addition to information loss per node we don't skip on too many nodes.

III. Problem Formulation

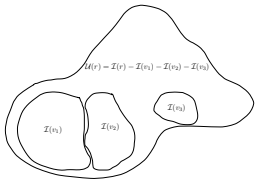
We formulate MTP as an ILP by using a variable x_e for each edge, and an boolean variable, γ_C, on any set $C \subset V$, which determines whether all nodes in C are excluded or included from the tree (i.e., $\gamma_C = 1$ iff $C = \overline{V_m}$).

$$\begin{aligned}
\min \quad & \sum_{e \in E} x_e c_e + \sum_{C \subset V; r \notin C} \gamma_C \varphi(C) & \\
\text{s.t.} \quad & \sum_{e \in \delta(S)} x_e + \sum_{S \subseteq C} \gamma_C \geq 1 & S \subset V; r \notin S \\
& \sum_{C \subset V; r \notin C} \gamma_C \leq 1 & \\
(\text{MTP} - \text{IP}) \quad & x_e \in \{0, 1\} & e \in E \\
& \gamma_C \in \{0, 1\} & C \subset V; r \notin C
\end{aligned}$$

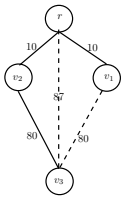
The formulation is standard for constraint tree problems. Let $\delta(S)$ be the number of outgoing edges from S. The linear constraint enforces that for each set of nodes S, either S has at least one outgoing edge or S is fully contained in the excluded set $\overline{V_m}$, i.e., it must be part of the tree or excluded. The target function is to minimize the sum of edges in the tree and the cost of the excluded set. To effectively solve the problem we create a linear programming relaxation MTP-LP by replacing the integer constraints with $x_e \geq 0$ and $\gamma_C \geq 0$.



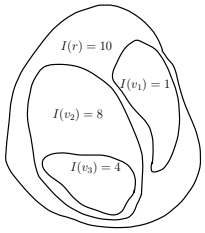
(a) The communication tree. Visited nodes are marked by dotted circles



(b) Area covered by the visited nodes



(c) Input graph with edge cost



(d) Area covered by nodes

Figure 1: Communication tree and covered areas.

V_m	$\sum_{v \in V_m} \mathcal{U}(v)^2$	$\sum_{e \in E_m} c_e$	Total
$\{r\}$	$10^2 = 100$	0	100
$\{r, v_2\}$	$(10-8)^2 + 8^2 = 68$	20	88
$\{r, v_1\}$	$(10-1)^2 + 1^2 = 82$	20	102
$\{r, v_1, v_2\}$	$(10-8-1)^2 + 8^2 + 1^2 = 66$	30	96
$\{r, v_3\}$	$(10-4)^2 + 4^2 = 52$	174	226
$\{r, v_3\}$	$(10-4)^2 + 4^2 = 52$	174	226
$\{r, v_2, v_3\}$	$(10-8)^2 + (8-4)^2 + 4^2 = 36$	177	213
$\{r, v_1, v_3\}$	$(10-4-1)^2 + 4^2 + 1^2 = 42$	177	219
$\{r, v_1, v_2, v_3\}$	$(10-8-1)^2 + (8-4)^2 + 4^2 + 1^2 = 34$	180	214

Table I: Mule tour cost when using penalty function φ_1.

The dual of the LP is as follows:

$$
\begin{aligned}
\max \quad & \sum_{S \subset V; r \notin S} y_s - \lambda & \\
\text{s.t.} \quad & \sum_{S | e \in \delta(S)} y_S \leq c_e & e \in E \\
(\text{MTP} - \text{DP}) \quad & \sum_{S \subseteq C} y_S - \lambda \leq \varphi(C) & C \subset V; r \notin C \\
& y_s \geq 0 & S \subset V; r \notin S \\
& \lambda \geq 0 &
\end{aligned}
$$

In the dual formulation, the variable y_S represents the cost of node set S. Let e be an edge that has one endpoint in S, the first constraint ensures that the total cost of this set is at most c_e. The second constraint ensures that the sum of dual variables for any set C is at most $\varphi(C)$.

IV. Previous Work.

Most of the work on data mules in ad-hoc networks is focused on finding efficient algorithms by means of energy or scheduling. For example, in [11], the authors purpose a schema for collecting data from sensors that transmit messages at a specific rate. They define the time a collector spends in collecting messages as the stability of the system, provide sufficient conditions for network stability, and derive upper bounds on the message delay of specific algorithms. Another interesting model was proposed in [17], where a collector with a bounded travel path must visit a set of sensors. A collector successfully receives a message only when it is within the transmission range of a sensor. The optimization criterion is to minimize the sum of transmission costs. Their solution involves reducing the problem to Traveling Salesman Problem with Neighborhoods [18]. Another approach was suggested in [12], where the authors considered a three-tier architecture that consists of sensors, transport agents (mules) and a top tier wireless access network (WAN). Assuming the mules perform a random walk, they analyze the time to collect data from all sensors and reach the top tier. In a subsequent paper [15], the authors extend their analysis with additional performance and model metrics, such as data transfer, latency, power and mobility pattern. Other work includes an experiment of the mule's three-tier architecture using real Mica2 sensors running the TinyOS operating system [19].

Some of our algorithms are based on a general framework for solving constraint degree forest problems. To solve the data gathering problem, we extended a well-known primal-dual framework for solving the Prize Collecting Traveling Salesman Problem (PCTSP) [20]. PCTSP extends the well-known Traveling Salesman Problem (TSP), where a salesman must travel to a known number of cities of the shortest tour, by enabling the salesman to skip over some cities and pay a constant penalty per city omitted. PCTSP was subject to several studies [21], [22] and has an 2-approximation algorithm. In PCTSP, the penalty of not visiting a subset of the cities is equal to the sum of penalties of the cities. There were several attempts to solve the problem for more general penalty functions. For example, the authors in [23] show a 3-approximation algorithm to the problem when the penalty function is sub-modular (for any two set S and T $\varphi(S) + \varphi(T) \geq \varphi(S \cup T) + \varphi(S \cap T)$).

V. Algorithm for Mule Tree Problem

In this section, we present a 6-approximation for MTP. Before presenting our algorithm, we introduce some basic definitions. We say that an edge $e \in \delta(S)$ is tight if the sum of dual cost of the sets that have e as an endpoint is equal to c_e, i.e., $\sum_{S|e\in\delta(S)} y_S = c_e$. In any step of the algorithm, any set of nodes can be in one of the following states: *not-active*, a set that contains the root, *dead*, a set that is not included in the tree, and *active*, a set that still "grows" and might be included in the final tree. The dead sets, $C_1, C_2, ..., C_k$ are ordered by the round they were declared dead. Algorithm 1 uses standard tree constraint dual-primal approximation technique [21] with modification to the set killing condition in Line 14. We show that even with this modification, the approximation ratio of the algorithm remains constant.

Algorithm 1: Mule Tree Problem Algorithm

Input: Graph $G = (V, E)$, root node r, and a penalty function φ.
Output: A tree $T = (V_m, E_m)$.

1 Start with a dual feasible solution by creating a corresponding set $S = \{v\}, \forall v \in V$, with $y_S = 0$, and set $\lambda = 0$.
2 Activate all sets except from the one that contains the root.
3 **while** *Some sets are active* **do**
4 Simultaneously increase the weight y_S by equal amount for all active sets until one of the following happens:
5 **case** *An edge e between an active set S_i to a set that is connected to r becomes tight*
6 S_i becomes deactivated.
7 S_i is merged with the set containing r.
8 $E_m = E_m \cup \{e\}$.
9 **end**
10 **case** *An edge between two sets S_i and S_j becomes tight*
11 The sets are merged to a new active set S_w, with $y_{S_w} = y_{S_j} + y_{S_i}$.
12 $E_m = E_m \cup \{e\}$.
13 **end**
14 **case** *The cost of an active set S_i, including the cost of all other dead sets $\sum_{i=1}^{k} y_{C_i}$ is equal to or larger than the penalty of the union of those sets, i.e., $y_{S_i} + \sum_{i=1}^{k} y_{C_i} \geq \varphi(S_i \cup_{i=1}^{k} C_i)$.*
15 S_i is declared dead.
16 **end**
17 Remove as many edges as possible from E_m while keeping the following conditions: a) All nodes that do not belong to dead components are connected to r, and b) If a node that belongs to a dead component C_i is mandatory to maintain condition a), then all other nodes in C_i must also be included in V_m. V_m is constructed from the endpoints of E_m.
18 **end**

Lemma 2: After k components are declared dead, the penalty constraint is violated by at most 2, i.e., $\sum_{S\subseteq\cup_{l=1}^{k} C_l} y_S - \lambda \leq 2\varphi(\cup_{l=1}^{k} C_l)$.

Proof: Let S_i and S_j be two components that were joined by a tight edge, and $C_1, C_2, ..., C_k$ be the dead components. By the construction of the algorithm, we get that $y_{S_i} + \sum_{l=1}^{k} y_{C_l} < \varphi(S_i \cup_{l=1}^{k} C_l)$ and $y_{S_j} + \sum_{l=1}^{k} y_{C_l} < \varphi(S_j \cup_{l=1}^{k} C_l)$. By definition of a dead set, $\varphi(\cup_{l=1}^{k} C_l) \leq \sum_{l=1}^{k} y_{C_l}$. When combining the equations and setting $\lambda = 0$ we get:

$$y_{S_i} + y_{S_j} + 2\varphi(\cup_{l=1}^{k} C_l) < \varphi(S_i \cup_{l=1}^{k} C_l) + \varphi(S_j \cup_{l=1}^{k} C_l)$$

Hence, $y_{S_i} + y_{S_j} + \varphi(\cup_{l=1}^{k} C_l) < \varphi(S_i \cup_{l=1}^{k} C_l) + \varphi(S_j \cup_{l=1}^{k} C_l) - \varphi(\cup_{l=1}^{k} C_l)$. According to the definition of the penalty function φ, for any disjoint sets S_i and S_j, $\varphi(S_i) \leq \varphi(S_i \cup S_j)$. Therefore:

$$y_{S_i} + y_{S_j} + \varphi(\cup_{l=1}^{k} C_l) < 2\varphi(S_i \cup S_j \cup_{l=1}^{k} C_l)$$

■

Let $T^*_{MTP-LP}, T^*_{MTP-DP}$ be the optimal solutions to the primal and dual problems, T^* be the optimal solution to MTP, and y_S be the value of the dual variable after Algorithm 1 completes. We claim the following:

Lemma 3: After Algorithm 1 completes, $\sum_S y_S \leq 2T^*$.

Proof: Let y^*_S be the dual variable value for a set S in the optimal solution to MTP-DP. From Lemma 2, we have:

$$\sum_S y_S \leq 2 \sum_S y^*_S \leq 2T^*$$

The last inequality follows from the duality principle. ■

Lemma 4: After Algorithm 1 completes, $\varphi(\overline{V_m}) \leq \sum_{S\subset V} y_S$.

Proof: Let C_k be the last set that was declared dead, by the construction of the algorithm we have:

$$\varphi(\overline{V_m}) = \varphi(\cup_{j=1}^{k} C_k) \leq \sum_{j=1}^{k} y_{C_j} \leq \sum_{S\subset V} y_S$$

■

We use the following theorem in our analysis:

Theorem 5: [21, Theorem 2.4] After Algorithm 1 completes, $\sum_{e\in E_m} c_e \leq 2 \sum_{S\subset V} y_S$.

The following theorem summarizes our results:

Theorem 6: Algorithm 1 is a 6-approximation to MTP problem.

Proof: Combining Theorem 5 and Lemma 4 we get:

$$\sum_{e\in E_m} c_e + \varphi(\overline{V_m}) \leq 2 \sum_{S\subset V} y_S + \sum_{S\subset V} y_S = 3 \sum_{S\subset V} y_S \leq 6T^*.$$

The last inequality follows from Lemma 3. ■

Following simillar argument as in [21], the running time of Algorithm 1 is $O(n^2 \log n)$.

VI. Algorithm for Mule Cycle Problem

In this Section we show how to extend the ideas learned in previous sections to create an efficient algorithm for finding a tour while keeping the 6-approximation ratio. We note that

any α-approximation algorithm for MTP can be converted to a 1.5α-approximation algorithm by using Christofides' algorithm [24] to transform the tree to a tour.

First, we formulate MCP as an ILP:

$$
\begin{array}{lll}
\min & \sum_{e\in E} x_e c_e + \sum_{C\subset V; r\notin C} \gamma_C \varphi(C) & \\
\text{s.t.} & \sum_{e\in\delta(S)} x_e + 2\sum_{S\subset C; r\notin S} \gamma_C \geq 2 & S\subset V; r\notin S \\
& \sum_{C\in V; r\notin C} \gamma_C \leq 1 & \\
(\text{MCP}-\text{IP}) & x_e \in \{0,1\} & e\in E \\
& \gamma_C \in \{0,1\} & C\subset V; r\notin C
\end{array}
$$

γ_C is equal to 1 for the set of nodes that are not selected and 0 otherwise. The second constraint enforces that each node included in the tour gets visited at least twice (i.e., the solution is a tour). The linear relaxation MCP-LP is formulated by relaxing the second and third constraints as in MTP-LP. The dual problem is as follows:

$$
\begin{array}{lll}
\max & 2\sum_{S; r\notin S} y_S - \lambda & \\
\text{s.t.} & \sum_{S|e\in\delta_S} y_S \leq c_e & e\in E \\
(\text{MCP}-\text{DP}) & 2\sum_{S\subseteq C} y_S - \lambda \leq \varphi(C) & C\subset V; r\notin C \\
& y_s \geq 0 & S\subset V; r\notin S \\
& \lambda \geq 0 &
\end{array}
$$

We use the following algorithm to solve MCP:

Algorithm 2: Mule Cycle Problem Algorithm

Input: graph $G=(V,E)$
Output: A tour $T=(V_m,E_m)$, and a set of nodes $\overline{V_m}$ not in the tour.

1 Produce tree T' by running algorithm 1 on G with penalty $\frac{\varphi(S)}{2}$.
2 Produce a tour T by running a depth first traversal on $T^{'}$.

We claim the following.

Lemma 7: Algorithm 2 is a 6-approximation for MCP

Proof: First note that the output of Algorithm 2 is a tour on the subset of V thus a valid solution to the primal problem. The dual solution also holds since we have $\sum_S y_S \leq \frac{\varphi}{2}$. Let y_S^* be the value of set S in the optimal solution, y_S be the value of set S after running Algorithm 2, and T^* be the optimal solution to MCP-IP. By similar arguments to Lemma 2 we have $\sum_S y_S \leq \sum_S 2y_S^* \leq 2T^*$. The cost of the depth-first traversal is $2\sum_{e\in E_m} c_e$. Thus, we get:

$$2\sum_{e\in E_m} c_e + \varphi(V_m) \leq 4\sum_S y_S + 2\sum_S y_S \leq 6(2y_S^*) \leq 6T^*.$$

■

VII. Penalty functions

We turn to prove that φ_1 and φ_2 are valid penalty functions and thus can be used in Algorithms 1 and 2. The proof for φ_3 follows directly from its definition.

Let C be the set of dead nodes in some iteration of the algorithm. We show that by increasing C by any number of nodes, the cost functions φ_1 and φ_2 are always not decreasing and therefore satisfy Condition 1. W.l.o.g we prove the claim on adding a single node v to C.

Lemma 8: Cost function φ_1 satisfies Condition 1

Proof: $\varphi_1(C)$ is equal to:

$$\underbrace{(\mathcal{I}(u) - \sum_{z\in d(u)\setminus\{v\}} \mathcal{I}(z) - \mathcal{I}(v))^2}_{\#1} + \underbrace{(\mathcal{I}(v) - \sum_{l\in d(v)} \mathcal{I}(l))^2}_{\#2} + \sum_{w\in V\setminus c\cup\{v,u\}} \mathcal{U}(w)^2.$$

Let u be the first ancestor of v that is does not belong to C. After removing v, the penalty $\varphi_1(C\cup\{v\})$ is:

$$(\mathcal{I}(u) - \sum_{z\in d(u)} \mathcal{I}(z))^2 + \sum_{w\in V\setminus C\cup\{v,u\}} \mathcal{U}(w)^2.$$

Clearly the increase in term $\#1$ is at least $\mathcal{I}(v)^2$, while the decrease in term $\#2$ is at most $\mathcal{I}(v)^2$. Therefore, the target function is not decreasing. ■

Lemma 9: Cost function φ_2 satisfies Condition 1

Proof: The penalty $\varphi_2(S_i)$ is equal to:

$$\max\{\underbrace{\mathcal{I}(u) - \sum_{z\in d(w)\setminus\{v\}} \mathcal{I}(z) - \mathcal{I}(v)}_{\#1}, \underbrace{\mathcal{I}(v) - \sum_{l\in d(v)} \mathcal{I}(l)}_{\#2}, \underbrace{\max_{w\in V\setminus C\cup\{v,u\}} \mathcal{U}(w)}_{\#3}\}.$$

After v is removed we get that the penalty $\varphi_2(C\cup\{v\})$ is:

$$\max\{\mathcal{I}(u) - \sum_{z\in d(w)} \mathcal{I}(z), \max_{w\in V\setminus C\cup\{v,u\}} \mathcal{U}(w)\}.$$

Since the increase in term $\#1$ $\mathcal{I}(v)$, is larger than term that was removed $\#2$, the target function increases. ■

VIII. Simulations

In this section, we present numerical results corresponding to the analysis from previous sections. We compare the performance of Algorithms 1 and 2 to competitive algorithms on

varying topologies, such as linear networks, stars, grids and random graphs. We compare our algorithms to several greedy algorithms that select k nodes (where k varies between 1 to n) and use the **optimal** tree or tour. To provide a wide range of possible costs and node selection, we set k to n, $\log n$, $\sqrt{n}$ and 1 (i.e., only the root is selected).

The first experiment studies the quality of our algorithms on linear ad hoc networks. Linear ad hoc networks can be used for multiple data collection activities such as railway or structural monitoring [25]. To construct the topology, we distributed 100 nodes, with covered area of 1, on a straight line with distances d between them. We calibrated d between 1 to 200, and compared between Algorithm 2 to the competitive algorithms described earlier. The results for penalty function φ_1 are shown in Figure 2a. When d is between 1 to 50, the optimal solution is to select all nodes with optimal path between them (since the cost of edges is relatively negligible to paying the penalty of a node). Note that for lower values of d the ratio between the cheapest solution to the most expensive one is above 100, and for those, Algorithm 2 cost is exactly the same as the optimal solution. When d increases, the cost of selecting edges becomes significantly expensive to pay the penalty of all nodes, and thus the optimal solution is to select only the root. In this interval, Algorithm 2 cost converges to the optimal value. The zig-zag in the mule cost is explained by the fact that the the difference between the no nodes cost and the all nodes cost is negligible, so the solution may change due to nodes distribution. When this difference becomes substantial the mule cost converges to the no nodes solution.

The results for penalty φ_2 are shown in Figure 2b. The results show similar trends as the first experiment, but since φ_2 yields a lower penalty than φ_1, the shift between optimal solutions happens for lower values of d. For example, the cost of empty solution for φ_1 is 10^4 while the cost for φ_2 is 10^2. As the value of d grows, the optimal solution changes between selecting all nodes to selecting only the root. For all values of d, the Algorithm 2 cost converges to the optimal value as well.

In our second experiment, we examined the quality of Algorithm 2 on star network. To construct the star, we placed the root r in the center on the graph, and equally distributed 10 nodes, each with a constant distance from r, on the circumference of the circle covering r. The covered area of each node was randomly and equally distributed between 1 to some upper bound $\max \mathcal{I}(v)$. We compared the performance of Algorithm 2 when using φ_1 and φ_2 to the four rival algorithms mentioned earlier.

The results of running the algorithms for varying upper bounds when using φ_1 and φ_2 are plotted in Figure 2c and 2d, respectively. In those experiments, the value of the star radius was set to 99, which is high enough to drop nodes if their penalty is too low. Clearly, the cost of not selecting a node increases proportionally to its covered area. For small values of $\max \mathcal{I}(v)$, the solution that contains only the root has the minimum cost, while as $\max \mathcal{I}(v)$ increases, the optimal solution is to include all the nodes. The cost of Algorithm 2 always converges to the optimal value and never exceeds the optimal by more than 2 (which fits the theoretical bound of 6).

In the next experiment, we randomly and uniformly distributed a varying number of nodes o'n a 10×10 Euclidean grid. To construct the data gathering tree T, we start with a tree that contains only r. During an iterative process, until all nodes belong to T, we select a leaf v, and a random integer k between 1 to 5. Then, we connect the closest k nodes that are not in T to v. The covered area per node is proportional to the density of nodes. The results for the cost functions φ_1 and φ_2 are plotted in Figure 2g and Figure 2h, respectively. For φ_1, Algorithm 1 converges to the optimal solution, achieving the best results in more than 60 percent of the samples, and for φ_2, Algorithm 1 yields optimal solution over all samples. The spikes in Figure 2g in the mule solution are due to the node distribution. When some nodes appear in the corners of the graph, the mule algorithm may decide to take them, which can increase the traversal cost. This phenomenon can be solved by scaling the penalty factor.

In our final experiment, we placed a varying number of nodes on the $\sqrt{n} \times \sqrt{n}$ grid. The grid topology can be used for reliable communication in ad hoc networks [26]. The covered area per node is proportional to the density of nodes in the graph (i.e., $\propto \frac{area}{n}$). The results when using φ_1 and φ_2 are plotted in Figures 2e and 2f, respectively. They show similar trends as in the previous experiments.

IX. Concluding remarks

In this paper, we developed a framework for finding near-optimal route for data gathering mule when the information gain of a node depends on the data collected so far. We identified a relationship between the mule problem to an extended variation of the Prize Collecting Traveling Salesman Problem, and presented a 6-approximation algorithm for it. This work is a first attempt towards a framework for solving data gathering problems that focus on the quality of data collected in addition to scheduling and power efficiency. A future extension of our work could investigate how to solve the mule problem when the traveling distance is bounded (similar to the constrained orienteering problem [27]) or when multiple mules work together to collect data.

References

[1] B. Wietrzyk and M. Radenkovic, "Energy efficiency in the mobile ad hoc networking approach to monitoring farm animals," in *Proceedings of the Sixth International Conference on Networking*. IEEE Computer Society, 2007.

[2] F. Ingelrest, G. Barrenetxea, G. Schaefer, M. Vetterli, O. Couach, and M. Parlange, "Sensorscope: Application-specific sensor network for environmental monitoring," *ACM Transactions on Sensor Networks*, vol. 6, no. 2, pp. 1–32, 2010.

[3] P. Sikka, P. Corke, P. Valencia, C. Crossman, D. Swain, and G. Bishop-Hurley, "Wireless adhoc sensor and actuator networks on the farm," in *Proceedings of the 5th international conference on Information processing in sensor networks*, 2006, pp. 492–499.

[4] S. R. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong, "Tinydb: an acquisitional query processing system for sensor networks," *ACM Trans. Database Syst.*, vol. 30, no. 1, pp. 122–173, 2005.

[5] Y. Ling and B. Xu, "Minimizing energy with a risk based temperature monitoring protocol for wireless digital sensor network," in *Proceedings of the 2006 IET International Conference on Wireless, Mobile and Multimedia Networks*, 2006, pp. 1–5.

[6] L. G. Chen and B. Xu, "Towards efficient temperature monitoring and controlling in large grain depot," in *3rd Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks*, 2006, pp. 881–885.

[7] L. Levin, M. Segal, and H. Shpungin, "Optimizing performance of ad-hoc networks under energy and scheduling constraints," in *WiOpt*, 2010, pp. 11–20.

[8] C. Buragohain, D. Agrawal, and S. Suri, "Power aware routing for sensor databases," in *24th Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings IEEE*, 2005, pp. 1747–1757.

[9] D. B. Johnson and D. A. Maltz, "Dynamic source routing in ad hoc wireless networks," *Mobile Computing*, vol. 353, pp. 153–181, 1996.

[10] B. Liu, D. Towsley, and A. Swami, "Data gathering capacity of large scale multihop wireless networks," in *5th IEEE International Conference on Mobile Ad Hoc and Sensor Systems*, 2008, pp. 124–132.

[11] G. D. Çelik and E. Modiano, "Dynamic vehicle routing for data gathering in wireless networks," in *CDC*, 2010, pp. 2372–2377.

[12] R. C. Shah, S. Roy, S. Jain, and W. Brunette, "Data mules: modeling and analysis of a three-tier architecture for sparse sensor networks," *Ad Hoc Networks*, vol. 1, no. 2-3, pp. 215–233, 2003.

[13] R. Shah, S. Roy, S. Jain, and W. Brunette, "Data mules: modeling a three-tier architecture for sparse sensor networks," in *Proceedings of the First IEEE International Workshop on Sensor Network Protocols and Applications*, 2003, pp. 30–41.

[14] G. Anastasi, M. Conti, and M. Di Francesco, "Data collection in sensor networks with data mules: An integrated simulation analysis," in *IEEE Symposium on Computers and Communications*, 2008, pp. 1096–1102.

[15] S. Jain, R. Shah, W. Brunette, G. Borriello, and S. Roy, "Exploiting mobility for energy efficient data collection in wireless sensor networks," *MONET*, vol. 11, pp. 327–339, 2006.

[16] R. Sugihara and R. K. Gupta, "Speed control and scheduling of data mules in sensor networks," *ACM Transactions on Sensor Networks*, vol. 7, pp. 1–29, 2010.

[17] D. Ciullo, G. D. Celik, and E. Modiano.

[18] E. M. Arkin and R. Hassin, "Approximation algorithms for the geometric covering salesman problem," *Discrete Applied Mathematics*, vol. 55, pp. 197–218, 1995.

[19] M. Conti, E. Gregori, and C. Spagoni, "Motes sensor networks in dynamic scenarios," *International Journal of Ubiquitous Computing and Intelligence*, vol. 1, 2006.

[20] E. Balas, "The prize collecting traveling salesman problem," *Networks*, vol. 19, pp. 621–636, 1989.

[21] M. X. Goemans and D. P. Williamson, "A general approximation technique for constrained forest problems," *SIAM Journal on Computing*, vol. 24, pp. 296–317, 1995.

[22] S. Gutner, "Elementary approximation algorithms for prize collecting steiner tree problems," in *Proceedings of the 2nd international conference on Combinatorial Optimization and Applications*, 2008, pp. 246–254.

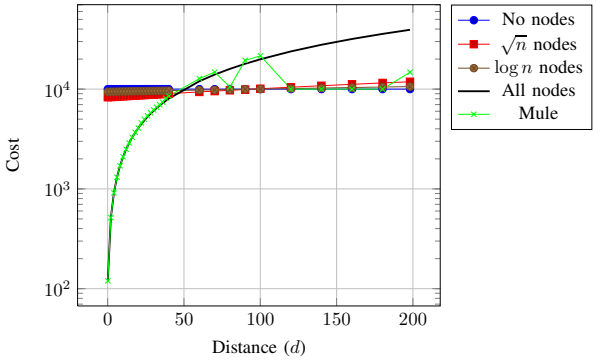
[23] Y. Sharma, C. Swamy, and D. P. Williamson, "Approximation algorithms for prize collecting forest problems with submodular penalty functions," in *SODA*, 2007, pp. 1275–1284.

[24] N. Christofides, "Worst-case analysis of a new heuristic for the travelling salesman problem," Graduate School of Industrial Administration, Carnegie Mellon University, Technical Report 388, 1976.

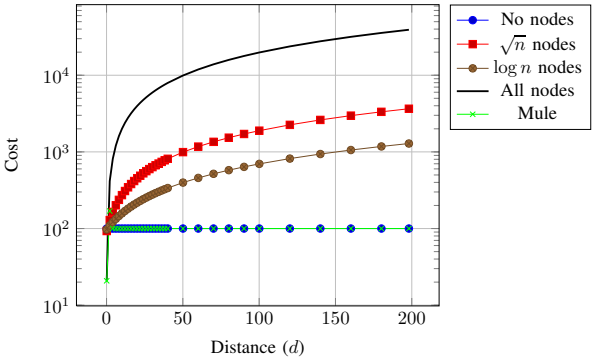
[25] S. Kim, S. Pakzad, D. Culler, J. Demmel, G. Fenves, S. Glaser, and M. Turon, "Health monitoring of civil infrastructures using wireless sensor networks," in *IPSN*. ACM Press, 2007, pp. 254–263.

[26] K.-H. Chen, C.-R. Dow, S.-C. Chen, Y.-S. Lee, and S.-F. Hwang, "Harpiagrid: A geography-aware grid-based routing protocol for vehicular ad hoc networks," *J. Inf. Sci. Eng.*, vol. 26, no. 3, pp. 817–832, 2010.

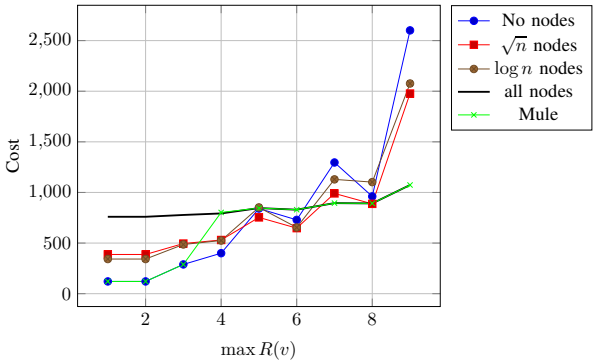
[27] B. L. Golden, L. Levy, and R. Vohra, "The orienteering problem," *Naval Research Logistics (NRL)*, vol. 34, no. 3, pp. 307–318, 1987.



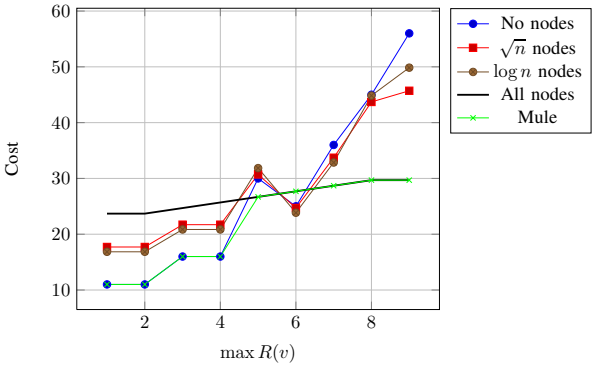
(a) Linear network - φ_1



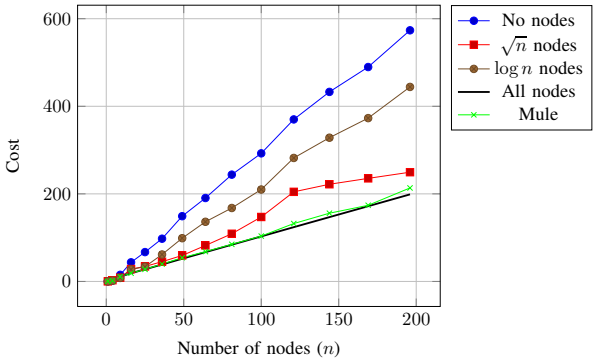
(b) Linear network - φ_2



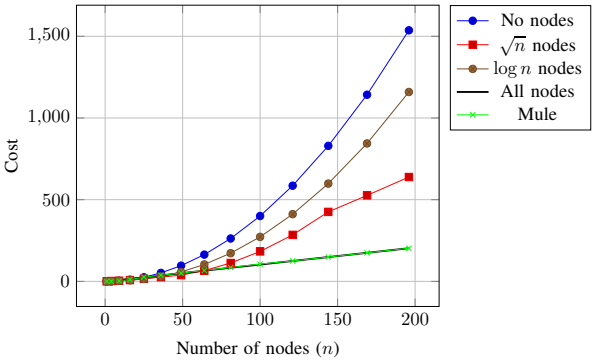
(c) Star network - φ_1



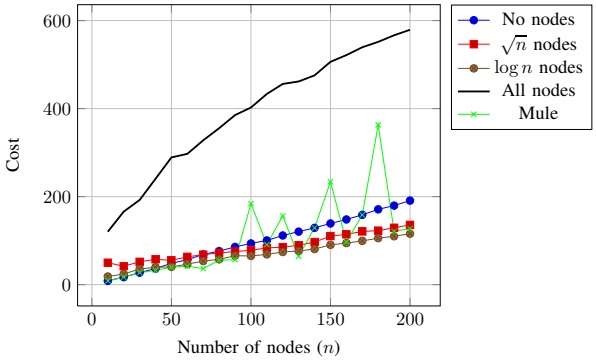
(d) Star network - φ_2



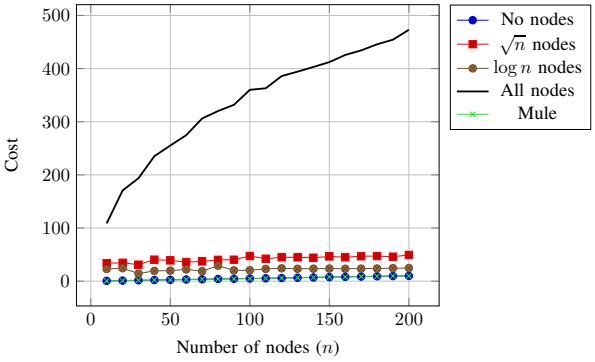
(e) Grid network - φ_1



(f) Grid network - φ_2



(g) Random network - φ_1



(h) Random network - φ_2

Figure 2: Simulation results for different topologies.