On Time-Constrained Data Harvesting in Wireless Sensor Networks: Approximation Algorithm Design

Lin Chen, *Member, IEEE*, Wei Wang, *Senior Member, IEEE*, Hua Huang, and Shan Lin

***Abstract*—In wireless sensor networks, data harvesting using mobile data ferries has recently emerged as a promising alternative to the traditional multi-hop communication paradigm. The use of data ferries can significantly reduce energy consumption at sensor nodes and increase network lifetime. However, it usually incurs long data delivery latency as the data ferry needs to travel through the network to collect data, during which some delay-sensitive data may become obsolete. Therefore, it is important to optimize the trajectory of the data ferry with data delivery latency bound for this approach to be effective in practice. To address this problem, we formally define the *time-constrained data harvesting problem*, which seeks an optimal data harvesting path in a network to collect as much data as possible within a time duration. We then investigate the formulated data harvesting problem in the generic m-dimensional context, of which the cases of $m = 1$, 2, 3 are particularly pertinent. We first characterize the performance bound given by the optimal data harvesting algorithm and show that the optimal algorithm significantly outperforms the random algorithm, especially when network scales. However, we mathematically prove that finding the optimal data harvesting path is NP-hard. We therefore devise an approximation algorithm and mathematically prove the output being a constant-factor approximation of the optimal solution. Our experimental results also demonstrate that our approximation algorithm significantly outperforms the random algorithm in a wide range of network settings.**

***Index Terms*—Approximation algorithm design, data harvesting, trajectory planning, wireless sensor network.**

I. Introduction

WIRELESS sensor networks (WSNs) play an important role in many applications ranging from environment monitoring and event detection, to target counting and tracking. In typical applications of WSNs, a major task of spatially distributed sensors is to report their sensing data to a designated data collector called the sink. In most WSNs deployed today, sensors usually have limited energy supply as they are battery-powered. A critical concern is thus how the sensing data from individual sensors can be collected to the sink with minimum energy consumption.

The task of data collection is traditionally accomplished by multi-hop forwarding, which is known to suffer from high energy consumption of forwarding nodes, especially those near the sink. Recently, as an efficient alternative, data harvesting using mobile devices, also termed as data mules [2] or data ferries [3], has been proposed and implemented in several applications such as underwater environmental monitoring [4]. The core idea can be summarized as follows: a data ferry (e.g., robot, vehicle) travels across the sensor field and harvests data from sensor nodes while they are within each other's communication range, and later transfers the harvested data to the sink. The use of data ferries in data harvesting can significantly reduce energy consumption at sensor nodes and thus increase network lifetime. However, as the data ferry can harvest data only when it travels close to the target node, it usually incurs longer data delivery latency, during which some delay-sensitive data may become obsolete. Therefore, optimizing the trajectory of the data ferry to limit or minimize data delivery latency is a primary concern for this approach to be effective in practice.

In this paper, we investigate the trajectory optimization problem in data collection applications for wireless sensor networks. This problem seeks an optimal data harvesting path to collect as much data as possible within a time duration. We call the problem *time-constrained data harvesting problem*. Specifically, our problem formulates the situation where delay-sensitive data needs to be reported to the sink within certain amount of time before they become obsolete. To make the analysis theoretically complete and generic, we investigate the generic m-dimensional context, of which the cases of $m = 1, 2, 3$ are particularly pertinent.

The contributions presented in this paper are naturally articulated as follows.

Theoretical Performance Bound. We formulate the time-constrained data harvesting problem. We analytically characterize the performance bound of the optimal data harvesting algorithm. Our analysis demonstrates that in a network where nodes are randomly deployed with fixed density and the data ferry moves at constant speed, the quantity of harvested data does not scale with the number of nodes in the network under the random data harvesting algorithm, while this quantity scales logarithmically for the optimal algorithm design, indicating a

Manuscript received January 30, 2015; revised September 21, 2015 and November 04, 2015; accepted November 24, 2015; approved by IEEE/ACM TRANSACTIONS ON NETWORKING Editor Y. Yi. The work of L. Chen was supported by the ANR under Grant Green-Dyspan (ANR-12-IS03). The work of W. Wang was supported by the NSFC under Grants 61571396 and 61261130585. The work of H. Huang and S. Lin was supported by the NSF under Grants CNS 1536086, CNS 1463722, and IIS 1460370. A preliminary version of this paper has been presented at IEEE INFOCOM 2015.

L. Chen is with the Laboratoire de Recherche en Informatique (LRI-CNRS UMR 8623), Université Paris-Sud, 91405 Orsay, France (e-mail: chen@lri.fr).

W. Wang is with the Department of Information Science and Electronic Engineering, Zhejiang University, Hangzhou 321000, China (e-mail: wangw@zju.edu.cn).

H. Huang and S. Lin are with the Department of Electrical and Computer Engineering, Stony Brook University, Stony Brook, NY 11794 USA (e-mail: huanghua.yh@gmail.com; shan.x.lin@stonybrook.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TNET.2015.2504603

significant performance gain when the network scales. Even though the trend is logarithmic, the gap can still be significant in large networks. In other words, a data harvesting algorithm not carefully chosen, such as randomly choosing a data harvesting path, can be very inefficient.

Approximation Algorithm Design. We give a formal proof on the NP-hardness of the time-constrained data harvesting problem. Motivated by the NP-hardness of the problem, we focus on the design of constant-factor approximation algorithms. Given the complexity of the problem, we first study a specific scenario with non-overlapping neighborhoods, i.e., the network is sufficiently sparse such that the data ferry cannot harvest data from multiple sensors without changing its location. We then extend the analysis to the generic case with overlapping neighborhoods, i.e., the network is sufficiently dense such that the data ferry can harvesting data from multiple sensors without changing location. As the key technicality in our design, we develop a methodology that relates the performance of topological paths to geometrical paths, based on which we mathematically prove the output of our algorithm is a constant-factor approximation of the optimal solution.

In our analysis, despite our focus on the data harvesting problem, the generic problem formulation of our work makes the analysis methodology and the obtained results broadly applicable to several engineering domains ranging from mobile charger scheduling, target monitoring to security patrolling, with a common generic objective of designing an optimal path such that a time-constraint utility function depending on the number of encountered targets is maximized.

The rest of the paper is organized as follows. We formulate the time-constraint data harvesting problem in Section III. In Section IV, we derive performance of the optimal data harvesting algorithm and the random algorithm, laying the theoretical foundation of the problem. In Section V, we first establish the NP-hardness of the time-constraint data harvesting problem, and then design a constant-factor approximation algorithm. Section VI analyzes the one-dimension case where exact solution can be found. Section VII presents simulation analysis of the proposed algorithm. Section VIII concludes the paper.

II. Related Work

The problem we address and the methodology we employ are related to the following research fields.

A. Data Ferry Assisted Data Harvesting

There is a large body of existing work on data ferry assisted data harvesting [5]–[9] (cf. [10] for a comprehensive survey). The problem we address is the optimization of data harvesting trajectory of the data ferry, which is a hard problem in general, since we are constrained in both space (communication range between the data ferry and sensors) and time domain (limiting data harvesting latency). Existing solutions contour this difficulty by either using simple mobility and communication models [5]–[9] or assuming that the trajectory is already given [5].

The authors in [3], [11] address a similar problem of designing data harvesting path for data ferries to minimize the data harvesting latency under the constraint that all sensors are visited. The algorithms they propose are based on the well-known travel salesman problem (TSP) [12] and its variant TSP with neighbors (TSPN) [13]. However, our problem is different because TSP requires the path to pass all sensors while we seek the most profitable path to harvest maximum data given the time constraint. Our problem formulation complements the TSP formulation and is particularly pertinent when the network is large and it is impossible for the data ferry to traverse every node. Technically, as detailed in the main part of the paper, our problem requires an original study that cannot draw from existing results.

B. Mobile Charger Scheduling

Another similar problem is the mobile agent scheduling problem where a mobile charger needs to travel within the charging range of each sensor node to recharge them under the constraint of the battery life of sensor nodes, which is similar to the time constraint in our data harvesting problem (cf. [14]–[16] and references therein). However, they rely on additional assumptions or simplifications to make the problem tractable. For example, the authors of [15] find out a near-optimum traveling path to recharge all sensor nodes using linear programming, assuming the traveling speed being infinite, and then remove this assumption and derive a bound of performance degradation. However, their algorithm implicitly assumes the travelling is fast enough. In our work, we remove these assumptions and analytically establish the performance properties of the proposed data harvesting algorithm.

C. Related Theoretical Problems

From a theoretical point of view, the problem we address is related to several fundamental problems in theoretical computer science, particularly the orienteering problem [17] and the weight-constrained minimum spanning tree problem [18]. In the orienteering problem, each node of a given graph has certain quantity of reward. The problem is to find a path that maximizes the reward collected, subject to a constraint on the path length. In the weight-constrained minimum spanning tree problem, each edge has a cost and weight. The problem is to find a spanning tree with minimum total cost subject to a upper-bound on the total weight. Both problems are NP-hard and have constant-factor approximation algorithms. However, these approximation algorithms cannot be directly applied in our problem as they are focused on topological paths.

III. Time-Constrained Data Harvesting Problem

A. Network Model

We consider a sensor network composed of n nodes, denoted by the set $\mathcal{V} = \{v_1, v_2, \cdots, v_n\}$, deployed in an m-dimensional Euclidean cube $[0, D]^m$. We are interested in the asymptotic scenario where both n and D are large with the node density $\lambda = n/D^m$ being a constant. We note that our motivation of considering a generic m-dimensional problem is to make our analysis mathematically complete and generically applicable. For the particular problem of time-constrained data harvesting in

TABLE I
NOTATIONS

$\mathcal{V}$	Set of nodes in the network
v_i	Node i
s	The mobile sink (data ferry)
n	Number of nodes in the network
D	Side length of the network area
m	Network dimension
λ	Node density
r	Communication range of nodes
D_i	Neighborhood ball of node v_i
$\mathcal{P}$	Path set
$\Lambda(P)$	Number of nodes covered by path P
P^*	The optimal path maximising harvested data
$\mathcal{P}_t$	Topological path set
$\mathcal{P}_g$	Geometrical path set
$\mathcal{P}_b$	Backbone path set
$d(P)$	Euclidean length of path P
T	Time duration within which data need to be deposed, T is also the maximum path length s can traverse before deposing data moving at unit speed

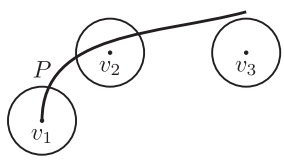


Fig. 1. An example illustrating notations and terminologies.

WSNs, the cases with $m = 1, 2$, and 3 are relevant. Table I lists the major notations used in the paper.

In the considered sensor network, each node v_i has unit data message[1] to be harvested by a data ferry, denoted by s, moving at a constant speed. To harvest data generated by v_i, s needs to move into the communication range of v_i, which is modeled as an m-dimensional ball D_i centered at v_i with radius r. We call D_i the neighborhood of node v_i. By slightly abusing notations, we also use D_i to denote the border of the ball which is an m-dimensional sphere. We use $\mathcal{P}$ to denote the possible path set for s. For a path $P \in \mathcal{P}$, we denote $d(P)$ the Euclidean length of P. We say that a path P covers a point M if there exists a point on P within distance r to M. In other words, if s moves along P, it can harvest the data generated by all the nodes that it covers. Denote $\Lambda(P)$ the number of nodes P covers. The following example further illustrates the notations and terminologies in our study.

Example 1: Consider the two-dimensional network illustrated in Fig. 1 composed of three sensor nodes v_1, v_2, v_3 with the circles around them indicating their neighborhoods D_1, D_2, D_3. The path P covers both v_1 and v_2, but not v_3. When moving along P, s can harvest data generated at both v_1 and v_2. We thus have $\Lambda(P) = 2$. $d(P)$ is the Euclidean length of P.

B. Problem Formulation

We consider the data harvesting problem faced by s in which it seeks an optimal data harvesting path to harvest as much data as possible within a time duration T. The problem we address models the situation where delay-sensitive data should be reported to the sink within certain time in order to be further analyzed. To make the notation concise, we let s move at unit speed and thus T is the maximum path length s can traverse before deposing the harvested data. The results obtained can be easily scaled to arbitrary speed by scaling the time duration T. Throughout our analysis, we are interested in the non-trivial case where $r \ll T$ and $Tr^{m-1} \ll D^m$, i.e., the maximum path length is much larger than the communication range, while the space covered by a path of length T is much smaller than the network space. The time-constrained data harvesting problem is formalized as follows.

Problem 1 (Time-Constrained Data Harvesting Problem): The time-constrained data harvesting problem is as follows:

$$\begin{aligned} &\text{maximize} \quad \Lambda(P), \\ &\text{subject to} \quad d(P) \le T. \end{aligned}$$

That is, s seeks the optimal path $P^* \in \mathcal{P}$ of Euclidean length $d(P^*) \le T$, along which it can harvest the maximum quantity of data. When there are more than one maximum, the optimal path P^* is the one with minimum Euclidean length.

We conclude this section by stating the following properties of P^* that will be useful in subsequent proofs and analysis.

Lemma 1(Properties of P):* Let $P^*(\tau)$ denote the optimal solution of Problem 1 with parameter $T = \tau$, the following properties hold:

- Monotonicity: $\Lambda(P^*(\tau_1)) \le \Lambda(P^*(\tau_2))$, $\forall \tau_1 \le \tau_2$;
- Scalability: $\Lambda(P^*(\kappa\tau)) \ge (\kappa\Lambda(P^*(\tau)))/(1+\kappa), \forall \tau$, $\forall \kappa \in (0,1)$.

Proof: The monotonicity follows straightforwardly from the definition of P^*. We now prove the scalability of P^*. Let l denote the integer such that $l < 1/\kappa \le l+1$. Divide $P^*(\tau)$ into $l+1$ non-overlapping parts of length $\tau/(l+1)$ each, it follows from the pigeonhole principle that there exists at least one part, denoted as p, which covers at least $(\Lambda(P^*(\tau)))/(l+1)$ nodes. It follows from $l < 1/k \le l+1$ and the monotonicity property that

$$\begin{aligned} \Lambda\left(P^*(\kappa\tau)\right) &\ge \Lambda\left(P^*\left(\frac{\tau}{l+1}\right)\right) \ge \Lambda(p) \\ &\ge \frac{\Lambda\left(P^*(\tau)\right)}{l+1} \ge \frac{\kappa\Lambda\left(P^*(\tau)\right)}{1+\kappa}, \end{aligned}$$

which completes the proof. □

It is worth noting that the time-constrained data harvesting problem has a number of important variants. In some applications, we require that the data harvesting path to be a cycle or have predefined starting and end points; it is sometimes required to differentiate sensor nodes by giving weights to them (e.g., giving higher weights to sensors at key positions) and seek the path maximizing the weighted sum of harvested data; furthermore, we may dispose multiple data ferries to for data harvesting. Many of these variants can be addressed using the framework established in this paper to design and optimize data harvesting path.

[1]The case where nodes generate multiple data messages can be tackled by devising the node generating k unit data messages to k virtual nodes at the same position, each generating unit data message.

IV. Theoretical Performance Bound: Optimal and Random Data Harvesting Algorithms

In this section, we establish the theoretical foundation of the time-constrained data harvesting problem by quantifying the performance of the optimal data harvesting algorithm and a natural algorithm where the data harvesting path is randomly chosen. In our analysis, we assume that nodes are randomly deployed in $[0, D]^m$.

A. Random Data Harvesting Algorithm

A simple data harvesting algorithm is to randomly choose a data harvesting path of length T. We call this algorithm random data harvesting algorithm, termed concisely as random algorithm. Our motivation of starting with the random algorithm is two-fold:

- It is a natural strategy and easy to implement;
- It provides a reference for performance comparison with more sophisticated algorithms as well as the optimal one.

The following theorem states the main result.

Theorem 1 (Performance of Random Data Harvesting Algorithm): Consider the random data harvesting algorithm where s randomly chooses a path P of length T, it holds that

- $\mathbb{E}[\Lambda(P)] = O(\lambda r^{m-1}T)$;
- $\Pr\{\Lambda(P) \geq n^{\epsilon}\mathbb{E}[\Lambda(P)]\} \to 0$, when $n \to \infty$, $\forall \epsilon > 0$, that is, $\Pr\{\Lambda(P) = \Theta(n^{\epsilon})\} \to 0$.

Proof: A point M is covered by a path P if the minimum distance between any point on P and M is at most r. Recall that $r \ll T$, the volume covered by a path P can be bounded in order-of-magnitude by $V = O(V_{m-1}T)$, where V_{m-1} denotes the volume of a ball of radius r in the $(m-1)$-dimensional space. In particular, we have $V_1 = 2r, V_2 = \pi r^2, V_3 = (4/3)r^3$. Generically, $V_m = \Theta(r^m)$ for given m. We can then bound the number of nodes covered by a random path P as

$$\mathbb{E}\left[\Lambda(P)\right] \leq \lambda V = O(\lambda r^{m-1}T).$$

To prove the second part of the theorem, by regarding $\Lambda(P)$ as a random variable and letting $a = n^{\epsilon}\mathbb{E}[\Lambda(P)]$ and by applying Markov's inequality $\Pr\{X \geq a\} \leq \mathbb{E}[X]/a$, we have

$$\Pr\left\{\Lambda(P) \geq n^{\epsilon}\mathbb{E}\left[\Lambda(P)\right]\right\} \leq \frac{1}{n^{\epsilon}} \to 0, \text{ when } n \to \infty,$$

which quantifies the sharpness of $\Lambda(P)$. □

With Theorem 1, we have the following result:

- In average, the expected harvested data for the random algorithm does not scale with respect to either the population size n of the network or its geometrical size D;
- With high probability, we cannot expect better outcome than $\Theta(\lambda r^{m-1}T)$.

B. Optimal Data Harvesting Algorithm

Having derived the performance of the random algorithm, we proceed to investigate the performance of the optimal data harvesting algorithm, as stated in Theorem 2.

Theorem 2 (Performance of Optimum Algorithm): Let P^* denote the path of the optimal data harvesting algorithm, it holds that

- $\mathbb{E}[\Lambda(P^*)] = \Theta(\log n/\log\log n)$;
- $\Pr\{\Lambda(P^*) = \Theta(\log n/\log\log n)\} \to 1$, when $n \to \infty$.

Proof: Please refer to the Appendix A. □

The intuition behind Theorem 2 and the proof is that by the bins and balls problem, in average we can find a region in the network such that it contains at least $\Theta(\log n/\log\log n)$ nodes and all the nodes in the region can be covered by a data harvesting path of length T. We have also shown that this bound is tight.

C. Discussion

Comparing the performance of optimal and random data harvesting algorithms, we can observe that when the network scales, especially when $n \to \infty$, the optimal algorithm significantly outperforms the random one. Even though the trend is logarithmic not polynomial or exponential, the gap can still be significant in large networks. In other words, a data harvesting algorithm not carefully chosen, such as randomly choosing a harvesting path, can be very inefficient. The motivates our second part of work on the following fundamental question:

How to design efficient data harvesting algorithms that approaches the solution of Problem 1?

Remark: Theorem 2 establishes the performance of the optimal algorithm. However, it does not specify how the optimal path can be constructed given a network instance. Choosing the path as indicated in the first step in the proof of Theorem 2 only performs well in the average sense when a large number of instances are executed, but it cannot give the optimal path for a given network instance. In fact, as we will show in the next section by Theorem 3, the problem of constructing the optimal path as formulated in Problem 1 is NP-hard.

V. Approximation Algorithm Design

In this section, we first show that Problem 1 is NP-hard. We then design constant-factor approximation data harvesting algorithms with polynomial-time complexity.

Theorem 3 (NP-Hardness of Time-Constrained Data Harvesting Problem): Problem 1 is NP-hard.

Proof: Please refer to the Appendix C. □

A. Non-Overlapping Neighborhood Case

Given the complexity of the time-constrained data harvesting problem, we first investigate a specific scenario where the neighborhoods of any two nodes are non-overlapped (i.e., $D_i \bigcap D_j = \emptyset$, $\forall v_i, v_j \in \mathcal{V}$) and develop an approximation algorithm for Problem 1. We start by the following definition of topological path.

Definition 1 (Topological Path): A path P_t is called a topological path in a graph if P_t is composed of uniquely the edges in the graph.

Generically, we call a path *geometrical path*, denoted as P_g for presentation clarity, to emphasize that P_g is not necessarily a topological path as P_g may contain curves and may start and end at any point. Of course, a topological path is also a geometrical one, i.e., let $\mathcal{P}_g$ and $\mathcal{P}_t$ denote the sets of geometrical and topological paths, it holds that $\mathcal{P}_c \subset \mathcal{P}_g$. Fig. 2 illustrates the notions of topological and geometrical paths.

The key element towards designing approximation algorithms for Problem 1 is to establish the relationship between geometrical and topological paths in terms of path length and

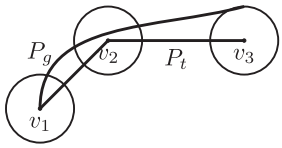


Fig. 2. A topological path P_t and a geometrical path P_g, both covering 3 nodes.

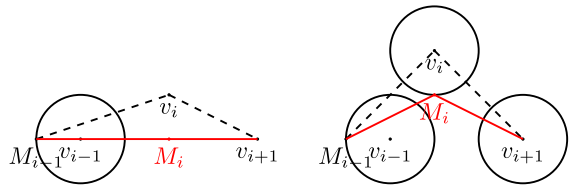


Fig. 3. Illustration of the core part of Algo. 1.

number of covered nodes, the two metrics on which we are focused. This relationship is established in two steps:

- *Step 1*: We show that any geometrical path P_g can be approximated by a topological path P_t such that

$$d(P_t) = O\left(d(P_g)\right), \text{ and } \Lambda(P_t) = \Lambda(P_g).$$

- *Step 2*: We show that any topological path P_t can be approximated by a geometrical path P_g via a *geometrization* procedure that we develop such that

$$d(P_g) = O\left(d(P_t)\right), \text{ and } \Lambda(P_g) \geq \Lambda(P_t).$$

We start by the first step to approximate a geometrical path P_g by a topological path P_t. By slightly abusing the notation, for a given path P, we reuse $\Lambda(P)$ to denote an ordered set of nodes covered by P. Using this notation, a topological path P_t can be uniquely noted by $\Lambda(P_t)$. Taking the topological path P_t in Fig. 2 as an example, P_t can be denoted by $\{v_1, v_2, v_3\}$. Given an ordered set of nodes $\mathcal{V}_g = \{v_1, v_2, \cdots, v_{|\mathcal{V}_g|}\}$, for any geometrical path P_g with $\Lambda(P_g) = \mathcal{V}_g$, we construct a topological path $P_t = \mathcal{V}_g$. It holds that $d(P_t) = O(d(P_g))$ and P_t covers all nodes in $\mathcal{V}_g$. Let the geometrical path P_g^* denote the geometrical path of minimum length among those covering $\mathcal{V}_g$, it holds that $d(P_t) = \Theta(d(P_g^*))$. This approximation result is mathematically formalized in Lemma 2.

Lemma 2: Given an ordered set of nodes $\mathcal{V}_g$, $\forall P_g$, $\Lambda(P_g) = \mathcal{V}_g$, let $P_t = \mathcal{V}_g$, it holds that $d(P_t) = O(d(P_g))$. Particularly, let $P_g^* = \arg\min_{\Lambda(P_g)=\mathcal{V}_g} d(P_g)$, it holds that $d(P_t) = \Theta(d(P_g^*))$.

Proof: To streamline our presentation, the proof, which is quite involved, is detailed in the Appendix D. □

We then proceed to the second step to approximate a topological path P_t by a geometrical path P_g by introducing geometrization, formally defined in the following.

Definition 2 (Geometrization): Given a topological path P_t, the geometrization procedure finds a geometrical path P_g that approximates P_t. By approximation we require that

$$d(P_t) = \Theta\left(d(P_g)\right), \quad \text{and} \quad \Lambda(P_t) \leq \Lambda(P_g).$$

Algo. 1 details the proposed geometrization procedure, whose core part is further illustrated in Fig. 3. It is straightforward to see that $d(P_g) < d(P_t)$. One technical point worth commenting is how to find M_i on D_i such that $|\overline{M_{i-1}M_i}| + |\overline{M_i v_{i+1}}|$ is minimized (line 6). M_i can be efficiently found by using the following technique: consider the outside border of D_i as a mirror; let a light beam be emitted from M_{i-1} and then be reflected by D_i to reach v_{i+1}; it follows from the theory of optics that light always travels using the shortest path; hence M_i corresponds to the reflection point of the light beam on D_i and can be found geometrically by equalizing the angle of incident and the angle of reflection.

Algorithm 1 Geometrization

Input: Topological path P_t passing nodes in $\mathcal{V}_t$
Output: Geometrized path P_g
1: Denote the intersection point of v_1v_2 and D_1 by M_1;
2: **for** $i = 2$ to $|\mathcal{V}_t| - 1$ **do**
3: **if** $\overline{M_{i-1}v_{i+1}}$ covers D_i **then**
4: Denote the first intersection point between $\overline{M_{i-1}v_{i+1}}$ and D_i by M_i; // See Fig. 3 (left);
5: **else**
6: Find a point M_i on D_i such that $|\overline{M_{i-1}M_i}|+|\overline{M_iv_{i+1}}|$ is minimized; // See Fig. 3 (right);
7: **end if**
8: **end for**
9: Denote the intersection point of $\overline{M_{|\mathcal{V}_t|-1}v_{|\mathcal{V}_t|}}$ and $D_{|\mathcal{V}_t|}$ by $M_{|\mathcal{V}_t|}$;
10: Return $P_g = \{\overline{M_1M_2}, \cdots, \overline{M_{|\mathcal{V}_t|-1}M_{|\mathcal{V}_t|}}\}$;

It is worth mentioning that the for loop in Algo. 1 can be repeated so as to further improve geometrization effectiveness (i.e., decrease $d(P_g)$). To make this clearer, let $P_g^{j-1} = \{\overline{M_1^{j-1}M_2^{j-1}}, \cdots, \overline{M_{|\mathcal{V}_t|-1}^{j-1}M_{|\mathcal{V}_t|}^{j-1}}\}$ denote the output of Algo. 1 at iteration $j-1$, for iteration j, it suffices to set $P_t = P_g^{j-1}$ by letting $v_k = M_k^{j-1}$ $(2 \leq k \leq |\mathcal{V}_t| - 1)$ in the algorithm. We observe via simulation that that the improvement is not significant or even negligible when Algo. 1 is executed more than a handful of times.

After establishing the relationship between geometrical and topological paths, we are now ready to present the global algorithm for Problem 1, as detailed in Algo. 2.

Algorithm 2 Approximation algorithm solving Problem 1: non-overlapping neighborhood case

Input: Coordinates of nodes in $\mathcal{V}$
Output: Π^*: a constant factor approximation of P^*
1: Construct a complete graph G with node set $\mathcal{V}$; set the length of the edge between v_i and v_j to be $\overline{v_iv_j}$;
2: For each node pair (v_i, v_j), find the topological path $\Pi_t(i,j)$ passing the maximum number of nodes in $\mathcal{V}$ whose geometrized path $\Pi_g(i,j)$ satisfies $d(\Pi_g(i,j)) \leq T$ using Algo. 1 and the algorithm of max-prize path in [17] by setting the prize for each node to be 1;
3: Return $\Pi^* = \arg\max_{\Pi_g(i,j),\ \forall v_i,v_j \in \mathcal{V}} \Lambda(\Pi_g(i,j))$;

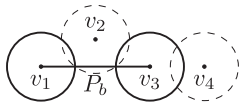


Fig. 4. An example of MIS composed of nodes v_1 and v_3.

The core idea of Algo. 2 is as follows: for each node pair, we find the topological path $\Pi_t(i,j)$ passing the maximum number of nodes in $\mathcal{V}$ whose geometrized path $\Pi_g(i,j)$ satisfies $d(\Pi_g(i,j)) \leq T$; we then return $\Pi^* = \arg\max_{\Pi_g(i,j),\ \forall v_i,v_j\in\mathcal{V}} \Lambda(\Pi_g(i,j))$. The two building blocks in Algo. 2 is the geometrization algorithm (Algo. 1) and the algorithm of max-prize path in [17]. Given a graph in which each node has a certain amount of prize, the max-prize algorithm finds in polynomial time a path collecting the maximum quantity of prize whose length is bounded by a constant, given as an input parameter. The following theorem formally establishes the performance of Algo. 2.

Theorem 4 (Performance of Algo. 2): Algo. 2 returns Π^* within polynomial time. It holds that $\Lambda(\Pi^*) = \Theta(\Lambda(P^*))$, where P^* denotes the optimal data harvesting path under time constraint T.

Proof: The polynomial-time complexity of Algo. 2 follows from the polynomial-time complexity of Algo. 1 and the algorithm of max-prize path.

The second part of the theorem $\Lambda(\Pi^*) = \Theta(\Lambda(P^*))$ can be proved using Lemma 2 and Lemma 1. Specifically, it follows from Lemma 2 that for any $\tau \leq T$, there exists a topological path P_t such that

$$\Lambda(P_t) = \Lambda\left(P^*(\tau)\right),\ d(P_t) \leq c\tau,$$

where $c \geq 1$ is a constant factor. Now let $\tau = T/c$, apply Lemma 1 by setting $\kappa = 1/c$, we have $d(P_t) \leq T$ and

$$\Lambda(P_t) = \Lambda\left(P^*(\tau)\right) \geq \frac{\Lambda\left(P^*(T)\right)}{c+1}.$$

On the other hand, it follows from the geometrization procedure and Algo. 2 that

$$\Lambda(\Pi^*) = \max_{\Pi_g(i,j),\ \forall v_i,v_j\in\mathcal{V}} \Lambda\left(\Pi_g(i,j)\right) \geq \Lambda(P_t) \geq \frac{\Lambda\left(P^*(T)\right)}{c+1}.$$

The theorem is thus proved. □

B. Overlapping Neighborhood Case

In this subsection, we extend our efforts to study the generic case with overlapping neighborhoods.

We first construct a graph G' whose node set is $\mathcal{V}$ and there is an edge between v_i and v_j if $\overline{v_iv_j} \leq 2r$.

We then construct a maximal independent set (MIS)[2] of G' using a coloring algorithm similar as presented in [11], [20], detailed in Algo. 3 for completeness. Fig. 4 illustrates an example of MIS composed of nodes v_1 and v_3.

[2]An independent set (IS) of an undirected graph is a subset $\mathcal{U}$ of nodes such that no two nodes in $\mathcal{U}$ are neighbors. An IS is maximal if no node can be added to $\mathcal{U}$ without violating IS. A maximal IS, or MIS, can be found in polynomial-time. Note that a related concept, a maximum IS (called MaxIS), is one IS of maximum cardinality. Finding MaxIS, however, is NP-hard.

Algorithm 3 MIS Construction of G'

Input: Graph G'
Output: MIS $\mathcal{U}$
1: **Initialization**: Set $\mathcal{U} = \emptyset$; Color all D_i $(v_i \in \mathcal{V})$ white;
2: **repeat**
3: Color a white ball D_i black and add v_i into $\mathcal{U}$;
4: Color every white ball D_j gray if v_j is v_i's neighbor;
5: **until** there is no white ball
6: Return $\mathcal{U}$;

We then define backbone topological paths, which can be regarded as topological paths using nodes in the MIS $\mathcal{U}$.

Definition 3 (Backbone Topological Path): A path P_b is called a backbone topological path, or backbone path for short, in a graph if P_b is composed of uniquely the edges whose endpoints are in the MIS of the graph except the source and the destination nodes.

As in the case of non-overlapping neighborhood, we call a path *geometrical path*, denoted as P_g, to emphasize that P_g is not necessarily a backbone path. Of course, a backbone path is also a topological path, and a geometrical one: i.e., let $\mathcal{P}_g$, $\mathcal{P}_t$ and $\mathcal{P}_b$ denote the sets of geometrical, topological and backbone paths, it holds that $\mathcal{P}_b \subset \mathcal{P}_t \subset \mathcal{P}_g$. As an example, the path P_b is Fig. 4 is a backbone path.

We apply the same analysis and design methodology in the non-overlapping neighborhood case and adapt it in the overlapping neighborhood case. A point M is said to be touched by path P if the minimum distance between any point of P and M is larger than r but smaller or equal to $2r$. The key element of designing approximation algorithm for Problem 1 with overlapping neighborhoods is to establish the relationship among geometrical, backbone, and geometrized backbone paths in terms of path length and number of touched and covered nodes. Specifically, we establish the relationship two steps:

- *Step 1*: We show that any geometrical path P_g can be approximated by a backbone path P_b such that $d(P_b) = O(d(P_g))$ and $\forall v_i$ covered by P_g, v_i is either covered or touched by P_b;
- *Step 2*: We show that any geometrical path P_g can be approximated by another geometrical path P'_g geometrized from a backbone path P_b via a *backbone geometrization* procedure such that

$$d\left(P'_g\right) = O\left(d(P_g)\right),\ \text{and}\ \Lambda\left(P'_g\right) \geq \Lambda(P_g).$$

We start with the first step by showing the following lemma. The proof uses similar reasoning technique as the proof of Lemma 2.

Lemma 3: Given any geometrical path P_g, there exists a backbone path P_b such that $d(P_b) = O(d(P_g))$ and $\forall v_i$ covered by P_g, v_i is either covered or touched by P_b. Particularly, let $P^*_g = \arg\min_{\Lambda(P_g)=\mathcal{V}_g} d(P_g)$, it holds that $d(P_b) = \Theta(d(P^*_g))$.

Proof: The lemma can be proved using similar reasoning technique in the proof of Lemma 2. We thus give the sketch of the proof. It follows from the definition of MIS that there exists

a subset of $\mathcal{U}$, denoted as $\mathcal{U}_s$, such that $\forall v_i$ covered by P_g, v_i is either covered or covered by at least one nodes in $\mathcal{U}_s$. To prove the lemma and bound the ratio $d(P_b)/d(P_g)$, it suffices to bound the length of the minimum-length backbone path passing by all nodes in $\mathcal{U}_s$. Using the same reasoning technique in the proof of Lemma 2, we can show that the ratio $d(P_b)/d(P_g)$ is maximized when P_g is packed with nearly overlapping nodes in $\mathcal{U}$. □

We then proceed to approximate a backbone path P_b by a geometrical path P_g by introducing *backbone geometrization*, formally defined in the following.

Definition 4 (Backbone Geometrization): Given a backbone path P_b, the backbone geometrization procedure finds a geometrical path P_g that approximates P_b. By approximation we require that $d(P_b) = \Theta(d(P_g))$, and $\Lambda(P_b) \leq \Lambda(P_g)$.

In [11], the authors develop a polynomial-time backbone geometrization algorithm, which will be used in our design.

The following lemma approximates a geometrical path by another geometrical path geometrized from a backbone path.

Lemma 4: Given any geometrical path P_g, there exists a path P_g' geometrized from a backbone path P_b such that

$$d\left(P_g'\right) = O\left(d(P_g)\right), \text{ and } \Lambda\left(P_g'\right) \geq \Lambda(P_g).$$

Proof: The lemma follows straightforwardly from Lemma 3 and the backbone geometrization algorithm. □

After establishing the relationship among geometrical, backbone and geometrized backbone paths, we now present the design of the global approximation algorithm for Problem 1 for the overlapping neighborhood case, as detailed in Algo. 4.

Algorithm 4 Approximation algorithm solving Problem 1: overlapping neighborhood case

Input: Coordinates of nodes in $\mathcal{V}$
Output: Π^*: a constant factor approximation of P^*

1: Construct a graph G' whose node set is $\mathcal{V}$ and there is an edge between v_i and v_j if $\overline{v_iv_j} \leq 2r$;
2: Run Algo. 3 on G' to construct an MIS $\mathcal{U}$;
3: Construct a complete graph G with node set $\mathcal{V}$; set the length of the edge between v_i and v_j to be $\overline{v_iv_j}$;
4: For each node pair (v_i, v_j), with the MIS $\mathcal{U}$ constructed in 2, find the backbone path $\Pi_b(i,j)$ passing the maximum number of nodes in $\mathcal{V}$ whose geometrized path $\Pi_g(i,j)$ satisfies $d(\Pi_g(i,j)) \leq T$ using the algorithm in [11] and the algorithm of max-prize path in [17] by setting the prize for each node i to be the number of nodes covered or touched by D_i;
5: Return $\Pi^* = \arg\max_{\Pi_g(i,j),\ \forall v_i,v_j \in \mathcal{V}} \Lambda(\Pi_g(i,j))$;

The core idea of Algo. 4 is as follows: for each node pair (v_i, v_j), we find the bcckbone path $\Pi_b(i,j)$ passing the maximum number of nodes in $\mathcal{V}$ whose geometrized path $\Pi_g(i,j)$ satisfies $d(\Pi_g(i,j)) \leq T$; we then return $\Pi^* = \arg\max_{\Pi_g(i,j),\ \forall v_i,v_j \in \mathcal{V}} \Lambda(\Pi_g(i,j))$. The two building blocks in Algo. 2 is the backbone geometrization algorithm [11] and the algorithm of max-prize path [17]. When running the algorithm of max-prize path, we set the prize of each node v_i to be the number of nodes covered or touched by D_i, which allows us to achieve constant-factor approximation (as detailed in the proof of Theorem 5). The following theorem establishes the performance of Algo. 4.

Theorem 5 (Performance of Algo. 4): Algo. 4 returns Π^* within polynomial time. It holds that $\Lambda(\Pi^*) = \Theta(\Lambda(P^*))$, where P^* denotes the optimal data harvesting path.

Proof: The polynomial-time complexity of Algo. 2 following from the polynomial-time complexity of the geometrization procedure [11] and the algorithm of max-prize path. We now prove second part of the theorem $\Lambda(\Pi^*) = \Theta(\Lambda(P^*))$.

Given a path P_g geometrized from a backbone path P_b, we denote the total collected prize along P_b by $Q(P_b)$ and set $Q(P_g) = Q(P_b)$. It can be noted that $\Lambda(\Pi^*) \leq Q(\Pi^*)$. It then follows from Algo. 4 and Lemma 4 that $Q(\Pi^*) = \Omega(\Lambda(P^*))$.

In the calculation of the max-prize in Algo. 4 (Step 4), a node may be counted multiple times in the final prize of the path. This is because a node can be covered by at most one node from the MIS $\mathcal{U}$ but can be touched by multiple nodes from $\mathcal{U}$. We next upper-bound the number of times a node is counted. To that end, we note that a node is counted more then once in the prize of a path if and only if it is not covered by any node in $\mathcal{U}$ and it is touched by multiple nodes in $\mathcal{U}$. Specifically, consider a node v_i, let $\mathcal{Y}$ denote the set of nodes such that v_i is not covered by any node in $\mathcal{Y}$ and is touched by all nodes in $\mathcal{Y}$. Let $Y = |\mathcal{Y}|$. To derive an upper-bound of Y, we note the following properties:

- Any two nodes y_i, y_j in $\mathcal{Y}$ do not have overlapping neighborhoods, i.e., $|\overline{y_iy_j}| > 2r$;
- For any node $y_i \in \mathcal{Y}$, it holds that $|\overline{v_iy_i}| \leq 2r$.

Recall the notation that D_i denote the neighborhood of v_i, i.e., the ball of radius r centered at v_i. It follows from the above properties that all nodes in D_i ($y_i \in \mathcal{Y}$) are within $3r$ to v_i. Hence the total volume of the Y balls of radius r must be strictly smaller than the volume of the ball of radius $3r$ centered at v_i. Mathematically, recall that the volume of an m-dimensional ball of radius r is $r^m\pi^m/\Gamma((m/2)+1)$ with $\Gamma(\cdot)$, we have the following inequality:

$$Y \cdot \frac{r^m\pi^m}{\Gamma\left(\frac{m}{2}+1\right)} < \frac{(3r)^m\pi^m}{\Gamma\left(\frac{m}{2}+1\right)} \quad \Longrightarrow \quad Y < 3^m.$$

We thus have $\Lambda(\Pi^*) = \Omega(Q(\Pi^*))$.

We have already proved that $Q(\Pi^*) = \Omega(\Lambda(P^*))$. It then holds that $\Lambda(\Pi^*) = \Omega(\Lambda(P^*))$. On the other hand, by the definition of P^*, we have $\Lambda(\Pi^*) \leq \Lambda(P^*)$. It then holds that $\Lambda(\Pi^*) = \Theta(\Lambda(P^*))$, which completes the proof. □

The time complexity of Algo. 2 and Algo. 4 is $O(n^5)$ following that the complexity of the max-prize path is $O(n^3)$. The approximation ratio can be derived from the approximation ratio of the prize-collecting problem (approximately 2) and the geometrization in the non-overlapping case (2) and the backbone geometrization in the overlapping case $(1 + (20/\pi))$ for the algorithm in [11]). The overall approximation ratio is thus 4 and $2(1 + (20/\pi))$ in the non-overlapping and overlapping cases.

VI. The One-Dimensional Case

For the one-dimensional (1-D) case, where the nodes are located within the interval $[0, D]$, we can explicitly derive the optimal data harvesting path P^*. For any realization of the locations of the nodes in the interval $[0, D]$, we label the nodes

from left to right to be $v_1, v_2, \cdots, v_n$. We denote the coordinate of node v_i $(1 \leq i \leq n)$ as x_i. Clearly it holds that $0 \leq x_1 \leq x_2 \leq \cdots \leq x_n \leq D$. For the 1-D case, without loss of generality, we assume that a path follows the direction from left to right. We can then uniquely denote a path P of length T with starting point x by $P(x)$. The problem of finding the path covering the maximum number of nodes can be formulated as:

$$\max_{0 \leq x \leq D-T} \sum_{1 \leq i \leq n} I(x, v_i),$$

where $I(x, v_i)$ is the indicator function defined as follows:

$$I(x, v_i) = \begin{cases} 1 & x - r \leq x_i \leq x + T + r, \\ 0 & \text{otherwise.} \end{cases}$$

Despite the NP-hardness of the data harvesting problem in its generic form, it turns out that it can be solved exactly in polynomial-time for the 1-D case by the simple searching algorithm illustrated in Algo. 5. The core idea is to calculate the number of covered nodes for paths starting from $x_i + r$ and then choose the one that covers the maximum number of nodes. Theorem 6 formally establishes the optimality of P^* computed by Algo. 5.

Theorem 6: Algo. 5 returns the optimal data harvesting path. Mathematically, it holds that

$$\Lambda(P^*) \geq \Lambda\left(P(x)\right), \quad \forall x \in [0, D].$$

Proof: We prove the theorem by distinguishing the following two cases:

- *Case 1*: $x \leq x_1 + r$. In this case, we have two subcases:
  - *Subcase 1.1* $x < x_1 - T - r$. In this subcase, $P(x)$ cannot cover any node, i.e., $\Lambda(P(x)) = 0$.
  - *Subcase 1.2*: $x_1 - T - r \leq x \leq x_1 + r$. In this subcase, noticing that there is no node in $[0, x_1)$, we have

$$\begin{aligned} \Lambda\left(P(x)\right) &= \sum_{1 \leq i \leq n} I(x, v_i) \leq \sum_{1 \leq i \leq n} I(x_1 + r, v_i) \\ &= \Lambda_1 \leq \Lambda(P^*). \end{aligned}$$

- *Case 1*: $x > x_1 + r$. In this case, we also have two subcases:
  - *Subcase 2.1*: $x > x_n + r$. In this subcase, $P(x)$ cannot cover any node, i.e., $\Lambda(P(x)) = 0$.
  - *Subcase 2.2*: $x_1 + r < x \leq x_n + r$. In this subcase, we can find $i_0 \in [1, n-1]$ such that $x_{i_0} + r < x \leq x_{i_0+1} + r$. Noticing that there is no node in (x_{i_0}, x_{x_0+1}), we have

$$\begin{aligned} \Lambda\left(P(x)\right) &= \sum_{1 \leq i \leq n} I(x, v_i) \leq \sum_{1 \leq i \leq n} I\left(x_{i_0+1} + r, v_i\right) \\ &= \Lambda_{i_0+1} \leq \Lambda(P^*). \end{aligned}$$

In both cases, we have $\Lambda(P(x)) \leq \Lambda(P^*)$, which completes the proof on the optimality of Algo. 5. □

Algorithm 5 Solving Problem 1 for 1-D case

Input: Coordinates of nodes x_i $(1 \leq i \leq n)$
Output: Optimal data harvesting path P^*

1: **for** $i = 1$ to n **do**
2: $\Lambda_i = \sum_{1 \leq j \leq n} I(x_i + r, v_j)$;
3: **end for**
4: $i^* = \arg\max_{1 \leq i \leq n} \Lambda_i$
5: $x^* = r + x_{i^*}$;
6: Return $P^* = P(x^*)$;

VII. Numerical Analysis

In this section, we conduct numerical analysis to evaluate the performance of the our constant-factor approximation algorithm of the time-constrained data harvesting problem. We evaluate the performance of our algorithm with respect to the following two algorithms:

- The random algorithm where the data harvesting path is randomly chosen;
- The greedy algorithm where s starts at some random location and then repeatedly goes to the nearest disk until the total length reach T.

Specifically, we set up a simulation area of an Euclidean cube $[0, 1000]^m$ for $m = 2$ (2-D case) and 3 (3-D case) and randomly deploy a number of n nodes in the cube, where n varies from 200 to 1000. The time constraint T is set to 100. We vary the communication range r to study various representative scenarios. By varying n and r, we can simulate both a sparsely deployed network where the neighborhoods of nodes are largely non-overlapping (small n and r) and a densely deployed network where the neighborhoods of nodes are largely overlapping (large n and r). For each set of chosen parameters, we run a number of independent simulations where the nodes' positions are randomly chosen and the required number of simulation runs is calculated using "independent replications" [21].

We trace the following metric to evaluate the performance of our algorithm compared to the random algorithm:

$$\Upsilon = \frac{\text{Quantity of data harvested by our algorithm}}{\text{Quantity of data harvested by random algorithm}}$$

The value of Υ characterizes the performance gain of our algorithm over the random one. We are particularly interested in tracing the following cases:

- *Worst-case performance gain*: Under given parameters n, r, we study the worst-case performance gain among the simulation runs, i.e., the minimum value of Υ, denoted as Υ_{min}. This result gives the lower-bound of the performance gain our algorithm can achieve over the random one;
- *Best-case performance gain*: Under given parameters n, r, we study the best-case performance gain among the simulation runs, i.e., the maximum value of Υ, denoted as Υ_{max}. This result gives the upper-bound of the performance gain our algorithm can achieve;
- *Average performance gain*: Under given parameters n, r, we study the average performance gain among the simulation runs, i.e., the average value of Υ, denoted as Υ_{avg}. This result gives the average of the performance gain of our algorithm.

Similarly, we define the same metric Υ' over the greedy algorithm to evaluate the performance of our algorithm compared to the random algorithm.

The simulation results are illustrated in Fig. 5 and Fig. 6. In Fig. 5, we fix $n = 200$ and trace Υ and Υ' as a function of r by varying r from 2 to 10. In Fig. 5, we fix $r = 6$ and trace Υ and Υ' as a function of n by varying n from 200 to 1000. Based on the simulation results, we make the following observations.

Compared to the random and the greedy algorithms, our algorithm achieves significant performance gain. Particularly, in

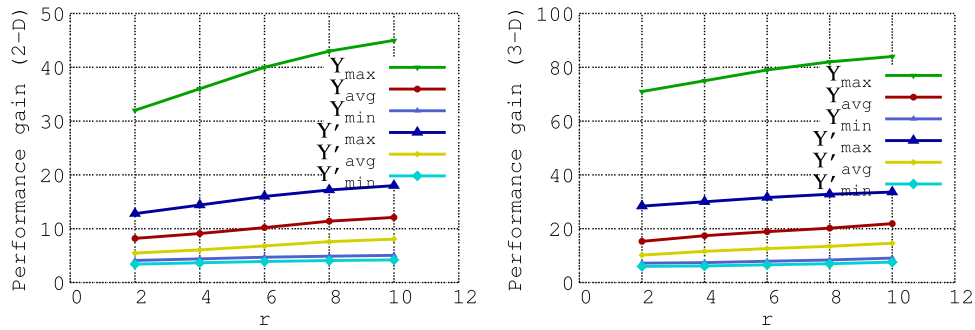


Fig. 5. Maximum, average, and minimum performance gain of our algorithm over the random algorithm as functions of r ($n = 200$).

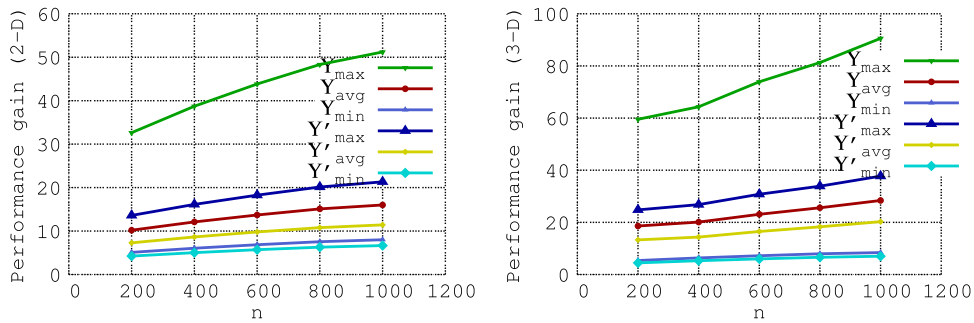


Fig. 6. Maximum, average, and minimum performance gain of our algorithm over the random algorithm as functions of n ($r = 6$).

the 2-D case, our algorithm can secure a performance gain of nearly 5 times better than the random algorithm and 3 times better than the greedy one in the simulated scenarios. In the extreme case, it performances 50 times better than the random algorithm and nearly 20 times better then the greedy one. The performance of our algorithm is even better in the 3-D case. The results also demonstrate our theoretical finding in Section IV that a data harvesting algorithm not carefully chosen, such as randomly choosing a data harvesting path, may lead to significant performance loss.

When the system scales, the performance gap between our algorithm and the random one increases, which again is in accordance of our theoretical analysis in Section IV. When the communication range r increases, the the performance gap also increases. This can be explained by the fact that our algorithm carefully chooses the data harvesting path so as to cover as many nodes as possible given the time constraint. In contrast, the random algorithm cannot fully take the advantage brought by larger r with a randomly chosen path.

VIII. Conclusion and Perspectives

In this paper, we have studied the problem of time-constrained data harvesting problem in which a data ferry seeks an optimal data collection path to collect as much data as possible within a time duration. This problem models the situation where time-sensitive data should be reported to the sink within certain time before they become obsolete. We have first characterized the performance bound given by the optimal data harvesting algorithm and shown that the optimal algorithm significantly outperforms the random algorithm, especially when the network scales. Motivated by the theoretical analysis and proving the NP-hardness of the time-constrained data harvesting problem, we have then devised an approximation algorithm of the problem and mathematically proved its output being a constant-factor approximation of the optimal solution.

As a small step towards characterizing efficient data harvesting algorithms, we expect that our work will stimulate further investigations in this field. The first interesting research direction is to use the methodology in the paper to study more sophisticated variants of the data harvesting problem, e.g., the case of multiple data ferries with heterogeneous moving speed. The second consists of investigating the data harvesting problem where the data ferry does not have full knowledge of the network topology and should make its decision based on only local information and interactions.

Appendix A
Proof of Theorem 2

We prove the theorem in two steps.

Step 1: Lower-Bound of $\Lambda(P^*)$: we show that $\Pr\{\Lambda(P^*) = \Omega(\log n/\log\log n)\} \geq 1 - (1/n) \to 1$, when $n \to \infty$. Our proof uses the well-known results in the *bins and balls* problem stated in the following lemma for completeness.

Lemma 5 (Maximum Load in Bins and Balls Problem [19]): When throwing k balls into $\Theta(k)$ bins, the max-loaded bin has $\Theta(\log k/\log\log k)$ balls with prob. at least $1 - (1/k)$ when $k \to \infty$.

We first construct "bins" in the following lemma.

Lemma 6: A path of length T can cover all nodes in an m-dimensional cube having sides of length $T^{1/m}(2r)^{1-(1/m)}/m^{(1/2)(1-(1/m))}$, asymptotically $\Theta(\sqrt[m]{Tr^{m-1}})$.

Armed with Lemma 6, we divide the network space $[0, D]^m$ into D^m/B^m non-overlapping cubes, each of sides of length $B = T^{1/m}(2r)^{1-(1/m)}/m^{(1/2)(1-(1/m))} = \Theta(\sqrt[m]{Tr^{m-1}})$.

Apply Lemma 5 by regarding nodes as balls, the path covering all nodes in the max-loaded bin covers at least $\Theta(\log n/\log\log n)$ nodes with probability $1-(1/n)$. Hence for the optimal path P^*, it holds that

$$\Pr\left\{\Lambda(P^*)=\Omega\left(\frac{\log n}{\log\log n}\right)\right\}\geq 1-\frac{1}{n}\to 1,\ \text{when } n\to\infty,$$

which completes Step 1 of the proof.

Step 2: Upper-Bound of $\Lambda(P^*)$: we show that $\Pr\{\Lambda(P^*)=O(\log n/\log\log n)\}\to 1$, when $n\to\infty$.

For a path of length T in the m-dimensional space, the maximum space it can cover has the volume $\Theta(Tr^{m-1})$. By dividing the network space $[0,D]^m$ into $\Theta(D^m/Tr^{m-1})$ non-overlapping "bins", each of volume $\Theta(Tr^{m-1})$, we can apply Lemma 5 to show that the max-loaded bin has $\Theta(\log n/\log\log n)$ nodes with probability at least $1-(1/n)$, i.e., the maximum number of nodes covered by a path of length T is upper-bounded by $\Theta(\log n/\log\log n)$ with probability at least $1-(1/n)$, mathematically:

$$\Pr\left\{\Lambda(P^*)=O\left(\frac{\log n}{\log\log n}\right)\right\}\geq 1-\frac{1}{n}\to 1,\ \text{when } n\to\infty.$$

This completes the second step of the proof.

Combining the two steps proves the sharpness result:

$$\Pr\left\{\Lambda(P^*)=\Theta\left(\frac{\log n}{\log\log n}\right)\right\}\to 1,\ \text{when } n\to\infty.$$

To prove $\mathbb{E}[\Lambda(P^*)]=\Theta(\log n/\log\log n)$, we proceed as below:

- Apply the result of Step 1 and notice the fact that $\Lambda(P^*)\geq 0$, we have
$$\mathbb{E}\left[\Lambda(P^*)\right]=\Omega\left(\frac{\log n}{\log\log n}\right);$$
- Apply the result of Step 2 and notice the fact that $\Lambda(P^*)\leq n$, we have
$$\begin{aligned}\mathbb{E}\left[\Lambda(P^*)\right]&\leq\Theta\left(\frac{\log n}{\log\log n}\right)\left(1-\frac{1}{n}\right)+n\cdot\frac{1}{n}\\&=\Theta\left(\frac{\log n}{\log\log n}\right),\quad n\to\infty.\end{aligned}$$

Combining above analysis yields $\mathbb{E}[\Lambda(P^*)]=\Theta(\log n/\log\log n)$.

Appendix B
Proof of Lemma 6

Consider the following m-dimensional lattice $\mathcal{K}$ with coordinates $\mathbf{K}b$ where $b=2r/\sqrt{m}$ and $\mathbf{K}=(k_1,k_2,\cdots,k_m)$ with $k_i=0,1,\cdots,K$. We first prove that any point in the m-dimensional cube $[0,Kb]^m$ is within distance r from at least one point from the lattice. To this end, for any point $\mathbf{X}=b(x_1,x_2,\cdots,x_m)$ in $[0,Kb]^m$ $(0\leq x_i\leq K, 1\leq i\leq m)$, let y_i denote the closest non-negative integer to x_i, it can be noticed that

- $\mathbf{Y}=b(y_1,y_2,\cdots,y_m)\in\mathcal{K}$,
- $|x_i-y_i|\leq 1/2, 1\leq i\leq m$.

The distance between $\mathbf{X}$ and $\mathbf{Y}$ can be bounded as:

$$|\overline{XY}|=\sqrt{\sum_{i=1}^{m}(x_i-y_i)^2}\leq\frac{b\sqrt{m}}{2}=r.$$

Hence, a path passing all nodes of $\mathcal{K}$ can cover all points in $[0,Kb]^m$.

We next construct the following sub-path P_0 that passes all nodes $\mathbf{X}\in\mathcal{K}$ where x_3 to x_m are fixed.

- Start from $(0,0,x_3\cdots,x_m)$ and move straightly towards $(K,0,x_3\cdots,x_m)b$;
- Move straightly from $(K,0,x_3\cdots,x_m)b$ to $(K,1,x_3\cdots,x_m)b$;
- Move straightly from $(K,1,x_3\cdots,x_m)b$ to $(0,1,x_3\cdots,x_m)b$;
- Repeat the above process until reaching $(0,K,x_3\cdots,x_m)b$ or $(K,K,x_3\cdots,x_m)b$.

By concatenating the sub-path P_0 from $(x_3,\cdots,x_m)=(0,\cdots,0)b$ to $(K,\cdots,K)b$ with a sub-path linking $(x_3,\cdots,x_i,\cdots,x_m)b$ and $(x_3,\cdots,x_i+1,\cdots,x_m)b$ for each $(x_3,\cdots,x_m)$, we can construct a path P that covers all nodes in $[0,Kb]^m$. The total length of the zigzag path P can be computed after some elementary geometrical operations as

$$d(P)=K^{m-2}(K^2+K+1)b=\Theta(K^m b).$$

When $T\gg r$, it can be calculated that with $d(P)=T$, P can cover all nodes in an m-dimensional cube with sides of length $T^{1/m}(2r)^{1-(1/m)}/m^{(1/2)(1-(1/m))}$, asymptotically $\Theta(\sqrt[m]{Tr^{m-1}})$.

Appendix C
Proof of Theorem 3

Consider the following problem which has been proved to be NP-hard in [22].

Problem 2: Find a tour in the travelling salesman problem (TSP) with an approximation factor better than 220/219.

To prove the NP-hardness of Problem 1, we prove that Problem 2 can be reduced to Problem 1 in polynomial time, i.e., Problem 2 $\leq_P$ Problem 1.

To that end, given any graph $G_t\triangleq(\mathcal{V}_t,\mathcal{E}_t)$ on which we need to solve the TSP, we instantiate Problem 1 by choosing r such that $r<\min_{e\in\mathcal{E}_t}d(e)$, for example, $r\to 0$. We consider the non-trivial case where $|\mathcal{V}_t|\geq 2$. Before showing how to reduce Problem 2 to Problem 1 in polynomial time, we prove the following property of Problem 1.

Lemma 7: Denote $P^*(\tau)$ the solution of Problem 1 on G_t with parameter $T=\tau$ and $r<\min_{e\in\mathcal{E}_t}d(e)$; let $t_{min}\triangleq\min_{e\in\mathcal{E}_t}d(e)$ and $t_{max}\triangleq\sum_{e\in\mathcal{E}_t}d(e)$, it holds that

$$\Lambda\left(P^*(t_{min})\right)=2\quad\text{and}\quad\Lambda\left(P^*(1.5t_{max})\right)=|\mathcal{V}_t|.$$

Proof: For the first part, it is easy to see that $\Lambda(P^*(t_{min}))=2$ when $r<\min_{e\in\mathcal{E}_t}d(e)$. To show $\Lambda(P^*(1.5t_{max}))\geq|\mathcal{V}_t|$, it suffices to notice that a spanning tree of G_t can be converted into a path using the famous 1.5-approximation algorithm for the TSP in [23]. Since the length of any spanning tree of G_t is upper bounded

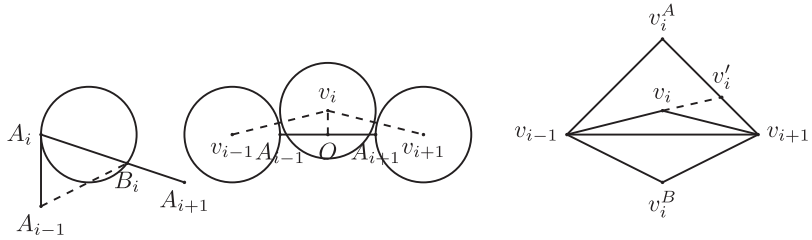


Fig. 7. Illustration of notations in Step 1 of the proof of Lemma 9: left: case 1; middle and right: case 2.

by t_{max}, we are sure to be able to find a path passing all nodes with the maximum length $1.5t_{max}$. In other words, $\Lambda(P^*(1.5t_{max})) = |V_t|$. □

Now we construct the following algorithm. The algorithm iterates on a variable t from $t = t_{min}+\epsilon$ to $1.5t_{max}$ by increasing t by ϵ from one iteration to the next, where ϵ is a small constant chosen such that $\epsilon \leq t_{min}/220$. In each iteration i, it solves Problem 1 with the constraint $d(P) \leq t_i \triangleq t_{min} + i \cdot \epsilon$, whose solution is denoted by P_i. The algorithm halts at the first iteration i^* where $\Lambda(P_{i^*}) = |\mathcal{V}_t|$ while $\Lambda(P_{i^*-1}) = |\mathcal{V}_t - 1|$ and outputs P_{i^*}. It follows from Lemma 7 and Lemma 1 (Monotonicity) that t_{i^*} exists.

It then follows that P_{i^*} is a TSP tour on G_t with an approximation factor $t_i/(t_i - \epsilon)$ which is upper bounded by 220/219 as $t_i \geq t_{min} + \epsilon$ and $\epsilon \leq t_{min}/220$. Hence P_2 is a solution of Problem 2.

The above analysis shows that Problem 2 can be reduced to Problem 1 in polynomial time. It then follows from the NP-hardness of Problem 2 that Problem 1 is NP-hard.

Appendix D
Proof of Lemma 2

We first investigate the structure of the minimum length path P_g^*.

Lemma 8: The minimum length path P_g^* is composed of an ordered set of line segments $\{\overline{A_1A_2}, \overline{A_2A_3}, \cdots, \overline{A_{|\mathcal{V}_g|-1}A_{|\mathcal{V}_g|}}\}$ where A_i is located on the border of D_i.

Proof: It is straightforward to notice that P_g^* must contain at least one point on the border of D_i ($1 \leq i \leq |\mathcal{V}_g|$). Let A_i and A_{i+1} denote the points on P_g^* located on D_i and D_{i+1} where $1 \leq i \leq |\mathcal{V}_g| - 1$. To prove the lemma, it suffices to notice that the distance between A_i and A_{i+1} is minimized when A_iA_{i+1} is a line and hence P_g^* must follow the segment $\overline{A_iA_{i+1}}$. □

Lemma 9: The ratio $d(P_t)/d(P_g^*)$ achieves its maximum when $|\overline{v_iv_{i+1}}| = 2r$ for $1 \leq i \leq |\mathcal{V}_g| - 1$.

Proof: We organize the proof in two steps.

Step 1: we show that when the ratio $d(P_t)/d(P_g^*)$ achieves its maximum, P_g^* intersects with D_i ($1 \leq i \leq |\mathcal{V}_g|$) at only one point. We can check that this holds for $i = 1$ and $|\mathcal{V}_g|$. Recall Lemma 8 that P_g^* is composed of a set of line segments $\{\overline{A_iA_{i+1}}\}$, we prove the case where $1 < i < |\mathcal{V}_g|$ by distinguishing two cases:

- *Case 1*: $\angle A_{i-1}A_iA_{i+1} \neq \pi$, i.e., A_i is not on the line segment $\overline{A_{i-1}A_{i+1}}$. Assume, by contradiction, that P_g^* intersects with D_i at more than one point. Recall Lemma 8, it holds that P_g^* consists of the line segment $\overline{A_iA_{i+1}}$, which intersects D_i at two distinct points, one being A_i, the other denoted as B_i (cf. Fig. 7). Geometrically, we have $|\overline{A_{i-1}A_i}| + |\overline{A_iA_{i+1}}| > |\overline{A_{i-1}B_i}| + |\overline{B_iA_{i+1}}|$. Hence, by replacing $\overline{A_{i-1}A_i}$ and $\overline{A_iA_{i+1}}$ in P_g^* by $\overline{A_{i-1}B_i}$ and $\overline{B_iA_{i+1}}$, respectively, we can decrease the length of P_g^*, which contradicts the fact that P_g^* has minimum length.
- *Case 2*: $\angle A_{i-1}A_iA_{i+1} = \pi$, i.e., $\overline{A_{i-1}A_{i+1}}$ intersects D_i. Let O denote the point such that $\overline{v_iO} \perp \overline{A_{i-1}A_{i+1}}$, as illustrated in Fig. 7. We now prove that under the constraint that $\overline{A_{i-1}A_{i+1}}$ intersects D_i, $|\overline{v_{i-1}v_i}| + |\overline{v_iv_{i+1}}|$ is maximized when $|\overline{v_iO}| = r$, i.e., P_g^* intersects D_i at only one point. Let v_i^A and v_i^B denote the points such that $|\overline{v_i^jO}| = r$ and $\overline{v_i^jO} \perp \overline{A_{i-1}A_{i+1}}$ ($j \in \{A, B\}$) (cf. Fig. 7 right). It suffices to show that for any point v_i ($v_i \neq v_i^A, v_i^B$) on the line segment $\overline{v_i^Av_i^B}$, at least one of the following inequalities hold

$$\begin{cases} |\overline{v_{i-1}v_i}| + |\overline{v_iv_{i+1}}| < \left|\overline{v_{i-1}v_i^A}\right| + \left|\overline{v_i^Av_{i+1}}\right|, \\ |\overline{v_{i-1}v_i}| + |\overline{v_iv_{i+1}}| < \left|\overline{v_{i-1}v_i^B}\right| + \left|\overline{v_i^Bv_{i+1}}\right|. \end{cases}$$

To prove this, assume, without loss of generality, that v_i is located in $\triangle v_{i-1}v_i^Av_{i+1}$, it holds that

$$\begin{aligned} \left|\overline{v_{i-1}v_i^A}\right| + \left|\overline{v_i^Av_{i+1}}\right| &= \left|\overline{v_{i-1}v_i^A}\right| + \left|\overline{v_i^Av_i'}\right| + \left|\overline{v_i'v_{i+1}}\right| \\ &> \left|\overline{v_{i-1}v_i'}\right| + \left|\overline{v_i'v_{i+1}}\right| \\ &= |\overline{v_{i-1}v_i}| + \left|\overline{v_iv_i'}\right| + \left|\overline{v_i'v_{i+1}}\right| \\ &> |\overline{v_{i-1}v_i}| + |\overline{v_iv_{i+1}}|. \end{aligned}$$

Hence, with the same P_g^* intersecting D_i, we can move v_i to v_i^A to increase the length of P_t, and thus the ratio $d(P_t)/d(P_g^*)$. This completes our proof of Case 2 and also Step 1.

Step 2: we show that $d(P_t)/d(P_g^*)$ achieves its maximum when $|\overline{v_iv_{i+1}}| = 2r$ for $1 \leq i \leq |\mathcal{V}_g| - 1$. We have just proved that P_g^* intersects D_i ($1 \leq i \leq |\mathcal{V}_g|$) at only one point A_i. Recall Lemma 8, we decompose P_g^* as $P_g^* = \sum_{i=1}^{|\mathcal{V}_g|-1} |\overline{A_iA_{i+1}}|$. We now consider the ratio $|\overline{v_iv_{i+1}}|/|\overline{A_iA_{i+1}}|$ for any given i. Assume that $|\overline{v_iv_{i+1}}| = x + 2r$, where $x \geq 0$ denotes the the shortest distance between any two points on D_i and D_{i+1}. Referring to the notations in Fig. 8, we have $|\overline{v_iv_{i+1}}|/|\overline{A_iA_{i+1}}| = (x + 2r)/|\overline{A_iA_{i+1}}|$.

Geometrically, we can show that when $\overline{A_iA_{i+1}}$ intersects $\overline{v_iv_{i+1}}$, as shown in Fig. 8 by $\overline{A_iA_{i+1}'}$, it holds that $|\overline{A_iA_{i+1}'}| > |\overline{A_iA_{i+1}}|$. Hence $|\overline{v_iv_{i+1}}|/|\overline{A_iA_{i+1}}|$ achieves its maximum when $\overline{A_iA_{i+1}}$ does not intersect $\overline{v_iv_{i+1}}$, which is

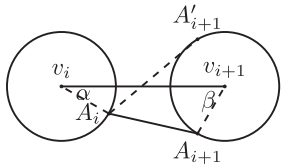


Fig. 8. Illustration of notations in Step 2 of the proof of Lemma 9.

our focus. Without loss of generality, assume that both A_i and A_{i+1} are below $\overline{v_iv_{i+1}}$, i.e., $\alpha, \beta \in [0, \pi/2]$. Trigonometrically, we can compute $|\overline{A_iA_{i+1}}|$ as

$$\begin{aligned}&\overline{A_iA_{i+1}}\\ &\quad= \sqrt{(x+2r-r\cos\alpha-r\cos\beta)^2+(r\sin\alpha-r\sin\beta)^2}.\end{aligned}$$

We now prove that $(x+2r)/|\overline{A_iA_{i+1}}|$ is maximized at $x=0$. In that regard, let $y \triangleq (x+2r)/r$, we need to show the following trigonometrical inequality:

$$\begin{aligned}&\frac{y^2}{(y-\cos\alpha-\cos\beta)^2+(\sin\alpha-\sin\beta)^2}\\ &\qquad\le \frac{4}{(2-\cos\alpha-\cos\beta)^2+(\sin\alpha-\sin\beta)^2}. \quad (1)\end{aligned}$$

By performing some algebraic operations, (1) can be simplified to the following inequality:

$$\begin{aligned}&\frac{1}{2}\left(1-\frac{4}{y^2}\right)(1+\cos\alpha\cos\beta-\sin\alpha+\sin\beta)\\ &\qquad\le \left(1-\frac{2}{y}\right)(\cos\alpha+\cos\beta). \quad (2)\end{aligned}$$

To prove (2), noticing that $y \ge 2$ in the non-overlapping neighborhood case, it suffices to prove the following inequality

$$1+\cos\alpha\cos\beta-\sin\alpha\sin\beta \le \cos\alpha+\cos\beta,$$

which can be proved as follows noticing that $\alpha, \beta \in [0, \pi/2]$:

$$\begin{aligned}&1+\cos\alpha\cos\beta-\sin\alpha\sin\beta-(\cos\alpha+\cos\beta)\\ &\quad= 1+\cos(\alpha+\beta)-(\cos\alpha+\cos\beta)\\ &\quad= 2\cos^2\left(\frac{\alpha+\beta}{2}\right)-2\cos\left(\frac{\alpha+\beta}{2}\right)\cos\left(\frac{\alpha-\beta}{2}\right)\\ &\quad= 2\cos\left(\frac{\alpha+\beta}{2}\right)\left[\cos\left(\frac{\alpha+\beta}{2}\right)-\cos\left(\frac{\alpha-\beta}{2}\right)\right]\\ &\quad= -4\cos\left(\frac{\alpha+\beta}{2}\right)\sin\frac{\beta}{2}\sin\frac{\alpha}{2} \le 0.\end{aligned}$$

It then holds that $d(P_t)/d(P_g^*)$ achieves its maximum when $|\overline{v_iv_{i+1}}| = 2r$ for $1 \le i \le |\mathcal{V}_g|-1$, which completes the proof of Step 2 and the lemma. □

Armed with the above lemmas, especially Lemma 9, we next show that $d(P_t) = O(d(P_g^*))$. Noticing Lemma 9, we only need to consider the case where $|\overline{v_iv_{i+1}}| = 2r$ for $1 \le i \le |\mathcal{V}_g|-1$. To compute $d(P_t)$, we have

$$d(P_t) = \sum_{i=1}^{|\mathcal{V}_g|-1} |\overline{v_iv_{i+1}}| = \Theta\left(r|\mathcal{V}_g|\right).$$

We next compute a lower-bound of $d(P_g^*)$. We first prove the following lemma.

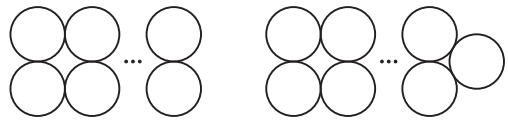


Fig. 9. $d(P_t)/d(P_g^*)$ is maximized when the neighborhoods of nodes D_i are "packed" one to the other: left (right): the number of nodes are even (odd).

Lemma 10: For any node v_i $(1 \le i \le |\mathcal{V}_g|-1)$ on P_t. Denote $c = 2^m$, The following inequality holds:

$$\exists j, i \le j \le i+c-1, w_j \in D_j, \text{ such that } |\overline{v_iw_j}| \ge 2r.$$

Lemma 10 states that for any node v_i on P_t, the maximum distance between v_i and any other node within the neighborhood of v_j where v_j is within c hops to v_i is lower-bounded by $2r$.

Proof: To prove the statement, assume, by contradiction, that

$$|\overline{v_iw_j}| < 2r \quad \forall w_j \in D_j, i \le j \le i+c-1. \quad (3)$$

Consider the c balls $D_i, D_{i+1}, \cdots, D_{i+c-1}$, it follows from (3) that any point of D_j $(i \le j \le i+c-1)$ is within $2r$ to v_i. Hence the total volume of the c balls of radius r must be strictly smaller than the volume of the ball of radius $2r$ centered at v_i. Mathematically, recall that the volume of an m-dimensional ball of radius r is $r^m\pi^m/(\Gamma((m/2)+1))$ with $\Gamma(\cdot)$ being the Leonhard Euler's gamma function [24], we have the following inequality:

$$c \cdot \frac{r^m\pi^m}{\Gamma\left(\frac{m}{2}+1\right)} < \frac{(2r)^m\pi^m}{\Gamma\left(\frac{m}{2}+1\right)} \quad \Longrightarrow \quad c < 2^m,$$

which contradicts to the fact that $c = 2^m$. □

Apply Lemma 10 to v_1, let j_0 $(1 < j_0 \le c+1)$ denote the index such that

$$\exists w_{j_0} \in D_{j_0}, \ |\overline{v_1w_{j_0}}| \ge 2r.$$

It follows immediately that the minimum distance between any point in D_1 and D_{j_0} is at least $r/2$. Consequently, for any geometrical path P_g^1 covering both v_1 and v_{j_0}, it holds that $d(P_g^1) = \Omega(r)$. Repeatedly apply this reasoning process to $v_{c+1}, v_{2c+1}, \cdots$, we have that for any geometrical path P_g covering all nodes in v_i $(i \le i \le |\mathcal{V}_g|)$, it holds that

$$d(P_g) = \frac{|\mathcal{V}_g|}{c}\Omega(r) = \Omega\left(r|\mathcal{V}_g|\right) = \Omega\left(d(P_t)\right).$$

Noticing that the path P_t itself can be regarded as a geometrical path, we then have $d(P_g^*) = \Theta(d(P_t))$, which complete the whole proof of Lemma 2.

Fig. 9 illustrates the 2-dimensional case where $d(P_t)/d(P_g^*)$ is maximized when D_i are "packed" one to the other. It can be checked that $d(P_t)/d(P_g^*) \to 2$ with large number of neighborhoods covered (i.e., large T).

REFERENCES

[1] L. Chen, W. Wang, H. Huang, and S. Lin, "Time-constrained data harvesting in WSNs: Theoretical foundation and algorithm design," in *Proc. IEEE INFOCOM*, 2015, pp. 999–1007.

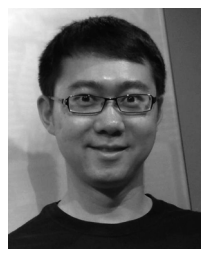
[2] R. Shah, S. J. S. Roy, and W. Brunette, "Data mules: Modeling a three-tier architecture for sparse sensor networks," in *Proc. IEEE SNPA Workshop*, May 2003, pp. 30–41.

[3] L. Xue *et al.*, "Multiple heterogeneous data ferry trajectory planning in wireless sensor networks," in *Proc. IEEE INFOCOM*, 2014, pp. 2274–2282.
[4] I. Vasilescu, K. Kotay, D. Rus, M. Dunbabin, and P. Corke, "Data collection, storage, retrieval with an underwater sensor network," in *Proc. ACM SenSys*, 2005, pp. 154–165.
[5] R. Sugihara and R. K. Gupta, "Speed control and scheduling of data mules in sensor networks," *ACM Trans. Sensor Netw.*, vol. 7, no. 1, Aug. 2010, Art. no. 4.
[6] M. Ma and Y. Yang, "SenCar: An energy-efficient data gathering mechanism for large-scale multihop sensor networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 18, no. 10, pp. 1476–1488, Oct. 2007.
[7] B. Yuan, M. Orlowska, and S. Sadiq, "On the optimal robot routing problem in wireless sensor networks," *IEEE Trans. Knowl. Data Eng.*, vol. 19, no. 9, pp. 1252–1261, Sep. 2007.
[8] D. Ciullo, G. Celik, and E. Modiano, "Minimizing transmission energy in sensor networks via trajectory control," in *Proc. WiOpt*, 2010, pp. 132–141.
[9] G. Xing, T. Wang, W. Jia, and M. Li, "Rendezvous design algorithms for wireless sensor networks with a mobile base station," in *Proc. MobiHoc*, 2008, pp. 231–240.
[10] E. Ekici, Y. Gu, and D. Bozdag, "Mobility-based communication in wireless sensor networks," *IEEE Commun. Mag.*, vol. 44, no. 7, pp. 56–62, Jul. 2006.
[11] D. Kim *et al.*, "Minimizing data collection latency in wireless sensor network with multiple mobile elements," in *Proc. IEEE INFOCOM*, 2012, pp. 504–512.
[12] D. L. Applegate, R. E. Bixby, V. Chvatal, and W. J. Cook, *The Traveling Salesman Problem: A Computational Study*. Cambridge, MA, USA: Harvard Univ. Press, 2006.
[13] A. Dumitrescu and J. Mitchell, "Approximation algorithms for TSP with neighborhoods in the plane," in *Proc. SODA*, 2001, pp. 38–46.
[14] L. Xie, Y. Shi, Y. T. Hou, and H. D. Sherali, "Making sensor networks immortal: An energy-renewal approach with wireless power transfer," *IEEE/ACM Trans. Netw.*, vol. 20, no. 6, pp. 1748–1761, Dec. 2012.
[15] L. Xie, Y. Shi, Y. T. Hou, W. Lou, and H. D. Sherali, "On traveling path and related problems for a mobile station in a rechargeable sensor network," in *Proc. MobiHoc*, 2013, pp. 109–118.
[16] S. Zhang, J. Wu, and S. Lu, "Collaborative mobile charging," *IEEE Trans. Comput.*, vol. 64, no. 3, pp. 654–667, Mar. 2015.
[17] A. Blum *et al.*, "Approximation algorithms for orienteering and discounted-reward tsp," *SIAM J. Comput.*, vol. 37, no. 2, pp. 653–670, 2007.
[18] V. Aggarwal, Y. Aneja, and K. Nair, "Minimal spanning tree subject to a side constraint," *Comput. Oper. Res.*, vol. 9, no. 4, pp. 287–296, 1982.
[19] M. Mitzenmacher, "The power of two choices in randomized load balancing," Ph.D. dissertation, University of California, Berkeley, CA, USA, 1996.
[20] W. Whitt, "The efficiency of one long run versus independent replications in steady-state simulation," *Manage. Sci.*, vol. 37, no. 6, pp. 645–666, 1991.
[21] P.-J. Wan, K. Alzoubi, and O. Frieder, "Distributed construction of connected dominating set in wireless ad hoc networks," in *Proc. IEEE INFOCOM*, 2002, pp. 1597–1604.
[22] C. H. Papadimitriou and S. Vempala, "On the approximability of the traveling salesman problem," *Combinatorica*, vol. 1, no. 26, pp. 101–120, 2006.
[23] N. Christofides, "Worst-case analysis of a new heuristic for the travelling salesman problem," Graduate School of Industrial Administration, CMU, tech. rep., 1976.
[24] A. F. Beardon, *The Geometry of Discrete Groups*. New York, NY, USA: Springer-Verlag, 1983.

Lin Chen (S'07–M'10) received his B.E. degree in radio engineering from Southeast University, China, in 2002 and the Engineer Diploma from Telecom ParisTech, Paris, France, in 2005. He also holds an M.S. degree of networking from the University of Paris 6. He currently works as an Associate Professor in the Department Of Computer Science of the University of Paris-Sud. He serves as Chair of the IEEE Special Interest Group on Green and Sustainable Networking and Computing with Cognition and Cooperation, IEEE Technical Committee on Green Communications and Computing. His main research interests include modeling and control for wireless networks, distributed algorithm design, and game theory.

Wei Wang (S'08–M'10–SM'15) received the B.S. degree and the Ph.D. degree from Beijing University of Posts and Telecommunications, China, in 2004 and 2009, respectively. Now, he is an Associate Professor with the College of Information Science and Electronic Engineering, Zhejiang University, China. From September 2007 to September 2008, he was a visiting student with the University of Michigan, Ann Arbor, MI, USA. From February 2013 to February 2015, he was a Hong Kong Scholar with Hong Kong University of Science and Technology. His research interests mainly focus on cognitive radio networks, green communications, and radio resource allocation for wireless networks. He is the editor of the book *Cognitive Radio Systems* (Intech, 2009) and serves as an Editor for IEEE ACCESS and the *Transactions on Emerging Telecommunications Technologies (ETT)*.

Hua Huang received the B.E. degree from Huazhong University of Science and Technology in 2012, and the M.S. degree from Temple University in 2014. He is currently working towards the Ph.D. degree in the Department of Electrical and Computer Engineering in Stony Brook University. His research interests include activity recognition in wearable devices and smart building, device-free indoor localization, deployment, and scheduling in wireless sensor networks.



Shan Lin is an Assistant Professor of the Electrical and Computer Engineering Department at Stony Brook University. He received his Ph.D. degree in computer science working under the direction of Prof. John A. Stankovic at the University of Virginia in 2010. His Ph.D. dissertation is on taming networking challenges with feedback control. His research is in the area of networked systems, with an emphasis on feedback control-based design for cyber physical systems and sensor systems. He works on wireless network protocols, interoperable medical devices, smart transportation systems, and intelligent sensing systems.