Exact and Approximation Algorithms for Data Mule Scheduling in a Sensor Network*

Gui Citovsky, Jie Gao, Joseph S. B. Mitchell, and Jiemin Zeng

Stony Brook University, Stony Brook, NY, USA

Abstract. We consider the fundamental problem of scheduling data mules for managing a wireless sensor network. A data mule tours around a sensor network and can help with network maintenance such as data collection and battery recharging/replacement. We assume that each sensor has a fixed data generation rate and a capacity (upper bound on storage size). If the data mule arrives after the storage capacity is met, additional data generated is lost. In this paper we formulate several fundamental problems for the best schedule of single or multiple data mules and provide algorithms with provable performance. First, we consider using a single data mule to collect data from sensors, and we aim to maximize the data collection rate. We then generalize this model to consider k data mules. Additionally, we study the problem of minimizing the number of data mules such that it is possible for them to collect all data, without loss. For the above problems, when we assume that the capacities of all sensors are the same, we provide exact algorithms for special cases and constant-factor approximation algorithms for more general cases. We also show that several of these problems are NP-hard. When we allow sensor capacities to differ, we have a constant-factor approximation for each of the aforementioned problems when the ratio of the maximum capacity to the minimum capacity is constant.

1 Introduction

A number of sensor network designs integrate both static sensor nodes and more powerful mobile nodes, called *data mules*, that serve and help to manage the sensor nodes [25, 26, 33, 34]. The motivation for such designs are twofold. First, there are fundamental limitations with the flat topology of static sensors using short range wireless communication. It is known that such a topology does not scale – the network throughput will diminish if the number of sensors goes to infinity [23], while allowing node mobility will help [22]. Second, a number of fundamental network operations can benefit substantially from mobile nodes. We consider two example scenarios: sensor data collection and battery recharging.

* G. Citovsky and J. Mitchell are partially supported by a grant from the US-Israel Binational Science Foundation (project 2010074) and the National Science Foundation (CCF-1018388, CCF-1540890). J. Gao and J. Zeng are partially supported by grants from AFOSR (FA9550-14-1-0193) and NSF (DMS-1418255, DMS-1221339, CNS-1217823).

In both cases, data mules that tour around the sensors periodically can be used to maintain the normal functionality of the sensors. In addition, data collection by sensors using multi-hop routing to a fixed base station often suffers from the bottleneck issue near the base station, both in terms of communication and energy usage. Using short range wireless communication with a mobile base station can fundamentally remove such dependency and avoid the single point of failure [37].

Despite the potential benefits of introducing data mules with static sensors, a lot of new challenges emerge at the interface of coordinating the data mules with sensors. One of the most prominent challenges is the scheduling of data mule mobility to serve the sensors in a timely and energy efficient manner. This has been an active research topic for the past few years. However, as surveyed later, most prior work is evaluated by simulations or experiments [3]; algorithms with provable guarantees are scarce. In this paper we make contributions in this direction. We formulate data mule scheduling problems with natural objective functions and provide exact and approximation algorithms.

Our Problem. Suppose there are n sensors and a data mule traveling at a constant speed s to collect data from these sensors. A sensor i generates data at a fixed rate of r_i and has a storage ("bucket") capacity of c_i where $c_i \geq r_i$. When a data mule visits a sensor, all current data stored in the sensor is collected onto the mule. We assume that the mule has unbounded storage capacity. We also assume that data collection at each sensor happens instantaneously, i.e., we ignore the time of data transmission, which is typically much smaller than the time taken by the mule to move between the sensors. If the amount of data generated at a sensor goes beyond its capacity (i.e., its bucket is full), additional data generated is lost. Thus, a natural objective is to schedule data mules to efficiently collect the continuously generated data.

We assume that the data collection and the data mule movement continues indefinitely in time. Therefore, we are mainly concerned about the long-term data gathering efficiency by periodic schedules.

The same problem arises in the case of battery recharging and energy management. In that case, each sensor i uses its battery with capacity c_i at a rate of r_i. When the battery at a sensor is depleted the sensor becomes ineffective. Thus, one would like to minimize the total amount of time of ineffectiveness, over all sensors. We formulate the following three problems.

- **Single Mule Scheduling:** Find a route for a single data mule to collect data from the sensors that maximizes the data collection rate (the average amount of data collected per time unit).
- **k-Mule Scheduling:** Given a budget of k data mules, find routes for them to maximize the rate of data collected from the sensors.
- **No Data Loss Scheduling:** Find the minimum number of data mules, and their schedules, such that all data from all sensors is collected (there is no data loss).

Our Results. We report hardness results, exact algorithms for a few special cases, and approximation algorithms for all three problems. Our algorithmic

results are summarized in Table 1. When we assume that the capacities of all sensors are the same, we provide results for the different cases where the sensors lie in different metrics. For the case where the capacities of the sensors are different, we provide general results.

Without loss of generality, we assume that the minimum data rate is 1 and the mule velocity is 1. In fact, we can further assume that all sensors have a data rate of 1; if a sensor has data rate $r_i > 1$, we can replicate this sensor with r_i copies, each with unit data rate and capacity c_i/r_i. Thus, in the following discussion we focus on the case of all sensors having unit data rates and possibly different capacities. When we consider the case where all sensors have the same capacity, we simplify notation and let the capacity of all sensors be c.

We give the first algorithms for such data mule scheduling problems with provable guarantees. In addition, we provide upper and lower bounds on the optimal solution for both problems, and we evaluate the performance using simulations, for a variety of sensor distributions and densities.

With Sensors	Single mule	k-mule	No Data Loss
on a Line	exact	$\frac{1}{3}$	exact
on a Tree	exact pseudo-polynomial	$\frac{1}{3}(1-1/e^{\frac{1}{2+\varepsilon}})$	12
General Metric Space	$1/6-\varepsilon$		
Euclidean Space	$1/3-\varepsilon$	$\frac{1}{3}(1-1/e^{1-\varepsilon})$	
with Different Capacities	$O(\frac{1}{m})$		$O(m)$

Table 1. Our approximation algorithm results for different settings. Note that $m \leq \log(\frac{c_{max}}{c_{min}})$ where c_{max} is the largest capacity and c_{min} is the smallest capacity. For the results in the first four rows, we assume that the sensor capacities are all the same. ε is any positive constant.

2 Related Work

Vehicle Routing Problems. The problems we study belong to the general family of vehicle routing problems (VRPs) and traveling salesman problems (TSPs) with constraints [9, 29, 32, 39]. But our problem is the first one considering periodically regenerated rewards/prizes and thus is the first of this type.

Related TSP variations stem from the Prize-Collecting Traveling Salesman Problem (PCTSP) [10, 13] which was originally defined by Balas [11] as the problem, given a set of cities with associated prizes and a prize quota to reach, find a path/tour on a subset of the cities such that the quota is met, while minimizing the total distance plus penalties for the cities skipped. (Some recent formulations of this problem do not include penalties for skipped cities.) Archer et. al. [4] provided a $(2-\varepsilon)$-approximation algorithm for this formulation of PCTSP where $\varepsilon \approx 0.007024$.

The Orienteering Problem [21, 36] assigns a prize to each city and, given a constraint on the length of the path, aims to maximize the total prize collected. For the rooted version in a general metric space, Blum et. al. [14] had proven that the problem is APX-hard and provided a 4-approximation algorithm which

was improved to a 3-approximation by Bansal et. al. [12] and finally to a $(2+\varepsilon)$-approximation by Chekuri et. al. [15]. For the rooted version in $\Re^2$, Arkin et. al. [6]give a 2-approximation, which was improved to a $(1+\varepsilon)$-approximation (PTAS) [16] for fixed dimension Euclidean space. For additional information we refer to the review papers [20, 36].

Similar to our problems, the Profitable Tour Problem [8] balances the two competing objectives of maximizing total prize collected and minimizing tour length. In some problems, the profit collected is dependent on the latency [17].

Our problems are also very similar to many multi-vehicle routing problems [5, 18, 19, 28]. Arkin et. al. [7] give constant-factor approximation algorithms for some types of multiple vehicle routing problems including a 3-approximation for the problem of finding a minimum number of tours shorter than a given bound that cover a given graph. Nagarajan and Ravi [31] provide a 2-approximation for tree metrics and a bicriteria approximation algorithm for general metrics. Khani and Salavatipour [27] present a 2.5-approximation algorithm for the problem of finding, for a given graph and bound λ, the minimum number of trees, each of weight at most λ, to cover the graph (improving on a bound given in [7]).

Data Mule Scheduling. Increasingly, there has been interest in using mobile data mules to collect data in sensor networks. A common question that has arisen is how to schedule multiple mules effectively and efficiently. Many heuristics have been proposed to schedule multiple mules with various constraints and objective functions (e.g., evenly distributing loads [25], scheduling short path lengths [30, 38], and minimizing energy [2]). Somasundara et. al. [35] address a very similar problem to ours, but with different methods; we obtain provable polynomial-time algorithms, while they employ (worst case exponential-time) integer linear programming and explore heuristics.

3 Single Mule Scheduling

Given a single mobile data mule with unit velocity, n sensors with uniform capacity and unit data rate, the goal is to route the mule in effort to maximize its data gathering rate. We explore this problem with sensors on a line, on a tree, and in space.

3.1 Exact Algorithms on a Line or a Tree

Line Case We first look at the case when the sensors are on a line. We assume that the input data is integral; specifically, the sensors p_i are located at integer coordinates and the capacities c_i for all i are integers. With this assumption, the optimal schedule can be shown to be periodic.

Lemma 1. *The optimal schedule that minimizes data loss is periodic, assuming integral input data.*

Proof. If the sensors are located at integral positions, the distances between any two of them are integers as well. Thus, all states of the problem can be encoded by the position of the mule and the current amount of data at each sensor i. All of these values are integers. Thus, the total number of possible states is finite;

after a state reappears we realize that the robot must follow the same schedule, making the schedule periodic. □

Theorem 1. *Let there be n sensors, $p_1, p_2, \ldots, p_n$ on a line. Assume that the capacities and rates of all sensors are the same: $c_i = c$ and $r_i = 1$, for $1 \leq i \leq n$. Then there exists an optimal path that minimizes data loss with the following properties: (1) its leftmost and rightmost points are at sensors, (2) it is a path making U-turns only at the leftmost and rightmost sensors.*

Despite the simple and clean statement, the proof is in fact fairly technical. To provide intuition for Theorem 1, note that paths that have U-turns not at the outermost points are making a tradeoff of collecting more data from middle sensors at the cost of having more overflow at the outer sensors. If this tradeoff is worth it, then we can show it is also worth it to forgo collecting data from some of the outer sensors. The main technical challenge is to figure out and compare the data rate between the two choices. For the full proof, we refer to a more detailed version of this paper posted on arXiv.

The immediate consequence from Theorem 1 is that one can find the optimal schedule in $O(n^2)$ time, enumerating all possible pairs of extreme points.

It is important to note that it is sometimes necessary for the mule in the optimal solution to gather data more than once from a given sensor in a period. In Figure 1, sensors are split into six groups, where each group has either k or $2k$ sensors. Within each group, each sensor has the same x-coordinate. In the optimal solution, the data mule traverses the entire interval back and forth, picking up data whenever it reaches a sensor. This solution has data gathering rate $\frac{10.5k}{2} = 5.25k$. In comparison, the best solution that gathers data from a sensor at most once per period has rate $4k$.

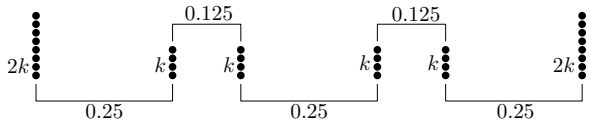

Fig. 1. The optimal solution repeats sensors

Tree Case We extend our results to a tree topology, with the sensors placed on a tree network embedded in the plane. Then, we show that the structure of an optimal schedule for the mule is to follow (repeatedly) a simple cycle (a doubling of a subtree). Again we assume that all sensors have the same capacity c and the same rate, 1, of data accumulation. We also assume that the input is integral, i.e., c is an integer and the distance between any two sensors on the tree network is an integer.

Theorem 2. *Let there be n sensors, $p_1, p_2, \ldots, p_n$ on a tree G. For all p_i, $1 \leq i \leq n$, let $c_i = c$ and $r_i = 1$, i.e. let the capacity and rates of all sensors be the same. There exists an optimal path that minimizes data loss with the following properties: (1) it only changes direction at sensors, (2) it is a cycle obtained by doubling a subtree.*

For the full proof, we refer to a more detailed version of this paper posted on arXiv. A consequence of Thereom 2 is that we can compute an optimal mule route (we can identify an optimal subtree of G) in time that is pseudo-polynomial, using a dynamic programming algorithm.

It is unlikely that there is a strongly polynomial time algorithm for an exact solution, since we show that the problem is weakly NP-hard.

3.2 Hardness

We show that single mule scheduling on a tree is weakly NP-hard. Further, we show that the data gathering problem for a single mule and sensors in Euclidean (or any metric) space is NP-hard.

Theorem 3. *The data gathering problem scheduling a single mule among uniform capacity sensors on a tree is (weakly) NP-hard.*

Proof. Our reduction is from PARTITION (or SUBSET-SUM): given a set $S = \{x_1, \ldots, x_n\}$ of n integers, does there exist a subset, $S' \subset S$, such that $\sum_{x_i \in S'} x_i = M/2$, where $M = \sum_i x_i$? Given an instance of PARTITION, we construct a tree as follows: There is a node v connected to a node u by an edge of length $M/2$. Incident on v are n additional edges, of lengths x_i; the edge of length x_i leads to a node where there are exactly x_i sensors placed. Also, at node u there are M^2 sensors placed. (If one disallows multiple ($x > 1$) sensors to be at a single node w of the tree, we can add x very short (length $\Theta(1/n)$) edges incident to w, each leading to a leaf with a single sensor.) Consider the problem of computing a maximum data-rate tour in this tree, assuming each sensor has capacity $2M$. Then, in order to decide if it is possible to achieve data collection rate of $M^2 + M/2$ we need to decide if it is possible to find a subtree that includes node u (where the large number, M^2, of sensors lie) and a subset of nodes having x_i sensors each, with the sum of these x_i's totalling exactly $M/2$. (If the sum is any less than $M/2$, we fail to collect enough data during the cycle of length $2M$ that is allowed before data overflow; if the sum is any more than $M/2$, we lose data to overflow at u, which cannot be compensated for by additional data collected at the x_i nodes, since M^2 is so large compared to x_i.) □

Theorem 4. *The data gathering problem scheduling a single mule among uniform capacity sensors in the Euclidean (or any metric) space is NP-hard.*

Proof. We reduce from the Hamiltonian cycle problem in a grid graph where n points are on an integer grid and an edge exists between two points if and only if they are unit distance apart. If we place a sensor at each point with capacity n, it follows that there exists a Hamiltonian cycle in this graph if and only if there exists a data gathering solution with no data loss. □

NP-hardness for this problem also holds for any general metric space.

3.3 Approximation Algorithm

Theorem 5. *For uniform capacity sensors in fixed dimension Euclidean space, there exists a* $(1/3-\varepsilon)$*-approximation for maximizing the data gathering rate of a single mule. For general metric spaces, a* $(1/6-\varepsilon)$*-approximation exists.*

Proof. In order to achieve this, we approximate the maximum number of distinct sensors a mule can cover in time $c/2$, the amount of time for sensors to fill from empty to half capacity (it can be shown that one half capacity is the optimal choice). The result will be a path, to which we assign one mule to traverse back and forth. The data gathering rate of this solution is equal to the number of distinct sensors covered as a mule on a schedule with period t will collect exactly t units of data from each sensor. We denote R to be the maximum number of distinct sensors that can be covered by a path of length $c/2$. Note that R can be approximated to within a factor of $1+\varepsilon$ in fixed dimension Euclidean space using the PTAS for orienteering [16]. In general metric spaces, R can be approximated to within a factor of $2+\varepsilon$ [15]. Let R^* be the data gathering rate of the optimal solution. We now show that $R^* \leq 3R$.

Consider the interval of time $c/2$ in the optimal solution that has the highest data gathering rate. This is an upper bound on R^*. In this time period, we know that the number of distinct sensors visited is at most R. We also know that during this time period at most $\frac{3}{2}c$ units of data can be downloaded from any visited sensor (at most c units of data immediately downloaded and at most $c/2$ units of data downloaded after $c/2$ units of time have passed). Therefore, the total amount of data collected in the optimal solution during this period of time is at most $\frac{3}{2}cR$. Averaging the data collected over the time interval $c/2$, the data gathering rate of the optimal solution is at most $3R$. □

4 *k*-Mule scheduling

Given a budget of k data mules, we now consider the problem of maximizing the total data gathering rate of these mules. We assume the n sensors have uniform capacity, unit data rate, and unit velocity. It is important to note that even with sensors on a line, the optimal solution may not assign mules to private tours; sensors may need to be visited by multiple mules. Consider an input with two mules and sensors uniformly spaced $c/4$ apart from one another. Any time a mule makes a U-turn, it will gather only $c/2$ data from the next sensor it visits. In order to maximize the frequency of full buckets collected, we want to minimize the frequency of U-turns made. This can be done by maintaining separation of length c between the mules and having the mules zig-zag across (nearly) the entire line (see Figure 2). Interestingly, this example also shows that mules can travel arbitrarily far distances.

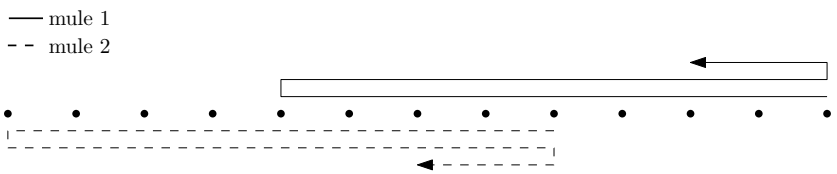

Fig. 2. With two data mules and sensors uniformly spaced $c/4$ apart, many sensors will be visited by both mules in the optimal solution.

4.1 Sensors on a line

Theorem 6. *Given a budget of k data mules, for uniform capacity sensors on a line, there exists a 1/3-approximation for maximizing the data gathering rate.*

Proof. Similar to the case when $k = 1$, we find the maximum amount of distinct sensors that k mules can cover in time $c/2$ (it can be shown that half capacity is the optimal choice). The result will correspond to a set of disjoint intervals; we assign one mule to each interval. The duration of a cycle for each mule is the length of time a sensor fills up to capacity so no sensor is allowed to overflow. Therefore, the data gathering rate of this solution, call it R, is equal to the number of sensors covered. Note that R, the maximum amount of distinct sensors that can be covered by k disjoint intervals of length at most $c/2$, can be computed exactly in polynomial time using dynamic programming. Let R^* be the data gathering rate of the optimal solution. It follows from the same argument given for the $k = 1$ case (Theorem 5) that $R^* \leq 3R$. □

4.2 Sensors in a general metric space

Theorem 7. *Given a budget of k data mules, for uniform capacity sensors in a general metric space, there exists a $\frac{1}{3}(1 - \frac{1}{e^\beta})$-approximation with $\beta = \frac{1}{2+\varepsilon}$ for maximizing the data gathering rate. In fixed dimension Euclidean space there exists a $\frac{1}{3}(1 - \frac{1}{e^\beta})$-approximation with $\beta = 1 - \varepsilon$.*

Proof. The proof is similar to the proof of Theorem 5. In order to approximate the maximum amount of distinct sensors that k mules can cover in $c/2$ time, we compute an orienteering path with a travel distance budget of $c/2$ on the uncovered sensors. We repeat this operation for a total of k times. In the Maximum Coverage problem, one is given a universe of elements, a collection of subsets, and an integer k. The objective is to maximize the number of elements covered using k subsets. It has been shown by Hochbaum et al. [24] that greedily choosing the set with the largest number of uncovered elements k times yields a $(1 - \frac{1}{e})$-approximation. Interestingly, Hochbaum et al. also show that using a β-approximation for covering the maximum amount of uncovered elements in each of the k rounds yields a $(1 - \frac{1}{e^\beta})$-approximation. Computing orienteering k times on only the remaining uncovered sensors, we achieve a $\frac{1}{(2+\varepsilon)}$-approximation each round and therefore a $(1 - \frac{1}{e^\beta})$-approximation for $\beta = \frac{1}{2+\varepsilon}$ for covering the maximum amount of sensors with k mules. Using similar arguments as the case where $k = 1$ (Theorem 5), it is now easy to see that having mules traverse the k orienteering paths back and forth yields a $\frac{1}{3}(1 - \frac{1}{e^\beta})$-approximation. □

5 No Data Loss Scheduling

In situations in which it is not possible for a fixed number of data mules to collect all data in the network, it is natural to increase the number of data mules and let them collectively finish the data collection task. In the no data loss scheduling problem, we seek to minimize the number of mules in order to avoid data loss. Throughout this section we assume that all sensors have unit data rate, unit velocity, and uniform capacity.

5.1 Exact Algorithm on a Line

When sensors all lie along a line, we show that the problem can be solved in polynomial time. As before, we can assume that the sensors lie at integer coordinates so that, by the same argument as in Lemma 1, the mules in an optimal solution follow periodic schedules.

Lemma 2. *For the minimum cardinality data mule problem with no data loss, if the sensors have uniform capacity and lie on a line, there is an optimal schedule in which all mules follow periodic cycles, zigzaging within disjoint intervals, each with length at most $c/2$.*

We refer to a more detailed version of this paper posted on arXiv for the full proof. By the above structural lemma, we can use a simple greedy algorithm to minimize the number of data mules necessary to collect data, without loss, for sensors on a line: Starting at the leftmost sensor, schedule a mule to zigzag within an interval of length $c/2$ whose left endpoint is the leftmost sensor, and then continue to the right, adding further intervals of length $c/2$ until all sensors are covered. This is an $O(n)$ algorithm for n (sorted) sensors.

5.2 Hardness

Even when the sensors lie on a tree, the problem of minimum cardinality data mule scheduling with no data loss is already NP-hard.

Theorem 8. *For uniform capacity sensors on a tree, the problem of minimum cardinality data mule scheduling with no data loss is (strongly) NP-hard.*

Proof. We reduce from 3-PARTITION. Given a multiset, $S = \{x_1, x_2, \cdots, x_{3n}\}$, of $3n$ integers with total sum M, 3-PARTITION asks whether there is a partition of S into n subsets, $S_1, \ldots, S_n$, such that each subset sums to exactly $B = M/n$. It is known that we can assume that the integers x_i satisfy $B/4 < x_i < B/2$, so that each subset S_i must consist of a triple of elements ($|S_i| = 3$). We create an instance of a star having a hub (center node) incident on $3n$ edges ("spokes") to $3n$ sensors, with edge lengths equal to x_i. Each sensor has capacity $2M/n$. Thus if there is a partitioning of S into triples of integers that each sum to M/n, then one (unit-speed) mule can traverse each corresponding 3-spoke subtree in time exactly $2M/n$, resulting in no data loss using n mules. On the other hand, any solution using exactly n data mules and having no data loss determines a valid partition of S into triples S_i. Thus, the 3-PARTITION instance has a solution if and only if n data mules suffice. □

For the general case, with sensors at points of a metric space or in a Euclidean space, the problem of determining the minimum number of data mules necessary to collect all data is also NP-hard. This can be seen by using a similar reduction from Hamiltonian cycle, as in Theorem 4.

5.3 Approximation Algorithm

In the following we describe an algorithm achieving a constant-factor approximation.

It is tempting to think that an optimal solution will allocate each mule to cover an exclusive set of sensors S', that are not covered by other mules. We denote such a set of tours as a *private tour set* on S'. However, the following example shows that this is no longer the case when sensors lie in the plane. Consider $n > 2$ sensors placed on a circle, uniformly spaced with adjacent sensors at (Euclidean) distance exactly $c - 1/n$. The convex hull of these sensors is a regular n-gon of perimeter $n(c-1/n) = nc-1$. The optimal solution would use $n-1$ mules, with each mule touring periodically at constant speed (1) along the boundary of this n-gon, with time/distance separation of exactly c between consecutive mules. This ensures that each sensor is visited exactly when its storage (bucket) becomes full. However, any solution using private tours will have to use n mules, since no mule can use a private tour to cover two or more sensors (since it would have length at least $2c - 2/n > c$).

While it may be that no private tour set is optimal, we now argue that the optimal schedule using only private tours is provably close to optimal (in terms of minimizing the number of data mules). Denote by k^* the minimum number of cycles, each of length at most c, to cover all nodes, which is denoted as a *light cycle cover*. And denote by m^* the minimum number of data mules required to collect all data.

Lemma 3. $m^* \leq k^* \leq 2m^*$.

Proof. First, note that using k^* mules, each traversing a (private) light cycle, results in all data being collected; thus, $m^* \leq k^*$.

Now consider an optimal schedule of m^* data mules. Mule i moves along a schedule C_i. Consider any particular time t. Each sensor j is visited by at least one mule. We assign it to the mule that visits it first, i.e., at the earliest time after t. We know this time is at most c, since no data is lost at sensor j. Thus, consider mule i, at current position p_i (at time t) and all the sensors along C_i that are assigned to it. They all lie on a path (along C_i) of length at most c. Let s_i be the sensor furthest away from p_i, measuring distance along C_i. Let γ_i be the corresponding path along C_i, from p_i to s_i. Let b_i be the midpoint of this path. Place a clone of mule i at point s_i, and create two private cycles for mule i and his clone: one cycle goes from p_i to b_i along γ_i, then returns to p_i directly (along a shortest path or a straight segment), the other goes from b_i to s_i along γ_i, then returns to b_i directly. Mule i traverses the first cycle; his clone traverses the second cycle. Do this for all mules.

We have doubled (via cloning) the number of mules, but now each mule/clone has a private cycle, of sensors assigned only to it, and these cycles are each of

total length at most c. Thus, this is a valid solution to the light cycle cover problem. Thus, the minimum number of light cycles, k^* is no greater than $2m^*$. □

By the above lemma, an α-approximation for the minimum light cycle cover gives a 2α-approximation for the minimum number of data mules. Arkin *et al.* [7] gave a 6-approximation algorithm for the minimum light cycle cover problem; thus, we have a 12-approximation for minimum data mule scheduling. This is summarized in the following theorem.

Theorem 9. *For uniform capacity sensors within a general metric space or in the Euclidean plane, computing the minimum number of data mules to collect all data is NP-hard. There is a polynomial-time* 12*-approximation algorithm for sensors in a general metric space.*

6 Different Capacities

We now consider both the k-mule scheduling problem and the no data loss scheduling problem on n sensors with potentially different sensor storage capacities. Each sensor has unit data rate. The result for the k-mule scheduling problem obviously holds for the single mule problem (i.e. when $k = 1$).

6.1 k-Mule Scheduling

Lemma 4. *With m groups of sensors, each group having the same storage capacity, optimally solving each group independently and taking the solution with the highest data gathering rate yields a $O(1/m)$-approximation to the k-mule scheduling problem.*

Proof. Let $r(\cdot)$ be the data gathering rate of a solution. Let OPT_i be the optimal solution to group i and let OPT be the schedule with highest data rate. $r(OPT) \leq \sum_{i=1}^{m} r(OPT_i) \leq m \cdot \max_i\{r(OPT_i)\}$. The first inequality is from the following observation. Consider the optimal schedule OPT and modify it such that we only visit the nodes in group i. This is obviously a solution for collecting data from group i and thus has data rate no greater than $r(OPT_i)$. □

Let c_{max} and c_{min} be the storage capacities of the largest and smallest sensors respectively. We round the storage capacity of each sensor down to its nearest power of two. Doing so, we create m groups of sensors where m is at most $\log(\frac{c_{max}}{c_{min}})$. Note that m may be significantly smaller than $\log(\frac{c_{max}}{c_{min}})$. In the rounding down process, the storage capacity of each sensor is at most halved, thus the optimal solution on the new sensors has data gathering rate of at least $1/2$ of the same solution before rounding. We approximate the optimal solution to each of the groups within a constant factor and choose the one with highest data gathering rate. By Lemma 4, we have the following.

Theorem 10. *By rounding down the sensor capacities into $m \leq \log(\frac{c_{max}}{c_{min}})$ groups, the group with highest data gathering rate has rate at least $O(1/m) \cdot r(OPT)$ where OPT is the optimal solution to the k-mule scheduling problem.*

6.2 No Data Loss Scheduling

Theorem 11. *By rounding down the sensor capacities into* $m \leq \log(\frac{c_{max}}{c_{min}})$ *groups and solving each group independently, at most* $O(m) \cdot |OPT|$ *mules are used in total, where* $|OPT|$ *is the minimum number of mules needed to avoid data loss.*

Proof. Using the same rounding technique as the previous section, we again obtain m groups of sensors with $m \leq \log(\frac{c_{max}}{c_{min}})$. In the rounding down process, the capacity of any sensor is at most halved. Thus, the optimal solution on the rounded down sensors requires at most two times the number of mules as the optimal solution to the original set of sensors. Let $|OPT_i|$ be the minimum number of mules needed for no data loss to occur in group i and let $|OPT|$ be the number of mules in the optimal solution. Since $|OPT| \geq |OPT_i|$ for $1 \leq i \leq m$, we have that $m \cdot |OPT| \geq \sum_{i=1}^{m} |OPT_i|$. Approximating $|OPT_i|$ within a constant factor for all i, we use $O(m) \cdot |OPT|$ mules. □

7 Conclusion

Our exact and approximation algorithms for single mule scheduling, k-mule scheduling, and no data loss scheduling represent the state of the art results on mule scheduling problems and greatly deepens our understanding of vehicular routing with constraints. For future work, we would like to see if the approximation ratios can be improved, especially with sensors in Euclidean space.

References

1. National traveling salesman problems. `http://www.math.uwaterloo.ca/tsp/world/countries.html`, accessed: 2014-12-15
2. Almi'ani, K., Viglas, A., Libman, L.: Tour and path planning methods for efficient data gathering using mobile elements. International Journal of Ad Hoc and Ubiquitous Computing, accepted for publication (2014)
3. Anastasi, G., Conti, M., Di Francesco, M.: Data collection in sensor networks with data mules: An integrated simulation analysis. In: Computers and Communications, 2008. ISCC 2008. IEEE Symposium on. pp. 1096–1102 (July 2008)
4. Archer, A., Bateni, M., Hajiaghayi, M., Karloff, H.: Improved approximation algorithms for prize-collecting steiner tree and tsp. SIAM J. Comput. (2), 309–332 (Mar.)
5. Archetti, C., Speranza, M., Vigo, D.: Vehicle routing problems with profits. Tech. rep., Tech. Report WPDEM2013/3, University of Brescia (2013)
6. Arkin, E., Mitchell, J.S.B., Narasimhan, G.: Resource-constrained geometric network optimization. In: In Symposium on Computational Geometry. pp. 307–316 (1998)
7. Arkin, E.M., Hassin, R., Levin, A.: Approximations for minimum and min-max vehicle routing problems. Journal of Algorithms (1), 1 – 18
8. Ausiello, G., Bonifaci, V., Laura, L.: The online prize-collecting traveling salesman problem. Inf. Process. Lett. 107(6), 199–204 (Aug 2008)
9. Ausiello, G., Leonardi, S., Marchetti-Spaccamela, A.: On salesmen, repairmen, spiders, and other traveling agents. In: Bongiovanni, G., Petreschi, R., Gambosi, G. (eds.) Algorithms and Complexity, pp. 1–16. Lecture Notes in Computer Science, Springer Berlin Heidelberg

10. Awerbuch, B., Azar, Y., Blum, A., Vempala, S.: Improved approximation guarantees for minimum-weight k-trees and prize-collecting salesmen. In: SIAM JOURNAL ON COMPUTING. pp. 277–283 (1995)
11. Balas, E.: The prize collecting traveling salesman problem. Networks 19(6), 621–636 (1989)
12. Bansal, N., Blum, A., Chawla, S., Meyerson, A.: Approximation algorithms for deadline-tsp and vehicle routing with time-windows. In: Proceedings of the Thirty-sixth Annual ACM Symposium on Theory of Computing. pp. 166–174. STOC '04, ACM, New York, NY, USA
13. Bienstock, D., Goemans, M.X., Simchi-Levi, D., Williamson, D.: A note on the prize collecting traveling salesman problem. Math. Program. (3), 413–420 (May)
14. Blum, A., Chawla, S., Karger, D., Lane, T., Meyerson, A., Minkoff, M.: Approximation algorithms for orienteering and discounted-reward tsp. In: Foundations of Computer Science, 2003. Proceedings. 44th Annual IEEE Symposium on. pp. 46–55 (Oct 2003)
15. Chekuri, C., Korula, N., Pál, M.: Improved algorithms for orienteering and related problems. ACM Trans. Algorithms (3), 23:1–23:27 (Jul.)
16. Chen, K., Har-Peled, S.: The euclidean orienteering problem revisited. SIAM Journal on Computing 38(1), 385–397 (2008)
17. Coene, S., Spieksma, F.C.R.: Profit-based latency problems on the line. Oper. Res. Lett. (3), 333–337 (May)
18. Eksioglu, B., Vural, A.V., Reisman, A.: Survey: The vehicle routing problem: A taxonomic review. Comput. Ind. Eng. (4), 1472–1483 (Nov.)
19. Even, G., Garg, N., Knemann, J., Ravi, R., Sinha, A.: Covering graphs using trees and stars. In: Approximation, Randomization, and Combinatorial Optimization.. Algorithms and Techniques, Lecture Notes in Computer Science, vol. 2764, pp. 24–35 (2003)
20. Feillet, D., Dejax, P., Gendreau, M.: Traveling salesman problems with profits. Transportation Science (2), 188–205 (May)
21. Golden, B.L., Levy, L., Vohra, R.: The orienteering problem. Naval Research Logistics 34, 307–318 (1987)
22. Grossglauser, M., Tse, D.: Mobility increases the capacity of ad hoc wireless networks. IEEE/ACM Transactions on Networking 10(4), 477–486 (August 2002)
23. Gupta, P., Kumar, P.R.: The capacity of wireless networks. IEEE Transactions on Information Theory 46(2), 388–404 (2000)
24. Hochbaum, D.S., Pathria, A.: Analysis of the greedy approach in problems of maximum k-coverage. Naval Research Logistics 45(6), 615–627 (1998)
25. Jea, D., Somasundara, A., Srivastava, M.: Multiple controlled mobile elements (data mules) for data collection in sensor networks. In: Proceedings of the First IEEE International Conference on Distributed Computing in Sensor Systems. pp. 244–257. DCOSS'05 (2005)
26. Kansal, A., Rahimi, M., Kaiser, W.J., Srivastava, M.B., Pottie, G.J., Estrin, D.: Controlled mobility for sustainable wireless networks. In: IEEE Sensor and Ad Hoc Communications and Networks (SECON'04) (2004)
27. Khani, M.R., Salavatipour, M.R.: Improved approximation algorithms for the min-max tree cover and bounded tree cover problems. Algorithmica 69(2), 443–460 (2014)
28. Kim, D., Uma, R., Abay, B., Wu, W., Wang, W., Tokuta, A.: Minimum latency multiple data mule trajectory planning in wireless sensor networks. Mobile Computing, IEEE Transactions on 13(4), 838–851 (April 2014)

29. Laporte, G.: The vehicle routing problem: An overview of exact and approximate algorithms. European Journal of Operational Research (3), 345 – 358
30. Ma, M., Yang, Y.: Data gathering in wireless sensor networks with mobile collectors. In: Parallel and Distributed Processing, 2008. IPDPS 2008. IEEE International Symposium on. pp. 1–9. IEEE (2008)
31. Nagarajan, V., Ravi, R.: Approximation algorithms for distance constrained vehicle routing problems. Networks 59(2), 209–214 (2012)
32. Pillac, V., Gendreau, M., Guéret, C., Medaglia, A.L.: A review of dynamic vehicle routing problems. European Journal of Operational Research (1), 1 – 11
33. Shah, R.C., Roy, S., Jain, S., Brunette, W.: Data mules: Modeling a three-tier architecture for sparse sensor networks. In: IEEE SNPA Workshop. pp. 30–41 (2003)
34. Somasundara, A., Kansal, A., Jea, D., Estrin, D., Srivastava, M.: Controllably mobile infrastructure for low energy embedded networks. Mobile Computing, IEEE Transactions on 5(8), 958–973 (Aug 2006)
35. Somasundara, A.A., Ramamoorthy, A., Srivastava, M.B.: Mobile element scheduling with dynamic deadlines. Mobile Computing, IEEE Transactions on 6(4), 395–410 (2007)
36. Vansteenwegen, P., Souffriau, W., Oudheusden, D.V.: The orienteering problem: A survey. European Journal of Operational Research (1), 1 – 10
37. Vincze, Z., Vida, R.: Multi-hop wireless sensor networks with mobile sink. In: CoNEXT'05: Proceedings of the 2005 ACM conference on Emerging network experiment and technology. pp. 302–303. ACM Press, New York, NY, USA (2005)
38. Wu, F.J., Tseng, Y.C.: Energy-conserving data gathering by mobile mules in a spatially separated wireless sensor network. Wireless Communications and Mobile Computing 13(15)
39. Yu, W., Liu, Z.: Vehicle routing problems with regular objective functions on a path. Naval Research Logistics (NRL) (1), 34–43