Available at
www.ComputerScienceWeb.com
POWERED BY SCIENCE DIRECT®

ELSEVIER

Ad Hoc Networks 1 (2003) 215–233

Ad Hoc Networks

www.elsevier.com/locate/adhoc

Data MULEs: modeling and analysis of a three-tier architecture for sparse sensor networks

Rahul C. Shah [a,*], Sumit Roy [b,1], Sushant Jain [c,2], Waylon Brunette [c]

[a] *Department of EECS, UC Berkeley and Intel Research Seattle, Berkeley Wireless Research Center, 2108 Allston Way, Suite 200, Berkeley, CA 94704, USA*

[b] *Intel Labs, 1100 NE 45th St 6th Floor, Seattle, WA 98105, USA*

[c] *Computer Science and Engineering, University of Washington, Box 352350, Seattle, WA 98195, USA*

Abstract

This paper presents and analyzes a three-tier architecture for collecting sensor data in sparse sensor networks. Our approach exploits the presence of mobile entities (called MULEs) present in the environment. When in close range, MULEs pick up data from the sensors, buffer it, and deliver it to wired access points. This can lead to substantial power savings at the sensors as they only have to transmit over a short-range. This paper focuses on a simple analytical model for understanding performance as system parameters are scaled. Our model assumes a two-dimensional random walk for mobility and incorporates key system variables such as number of MULEs, sensors and access points. The performance metrics observed are the data success rate (the fraction of generated data that reaches the access points), latency and the required buffer capacities on the sensors and the MULEs. The modeling and simulation results can be used for further analysis and provide certain guidelines for deployment of such systems.

Keywords: Sensor network; Mobility; Energy efficient; Random walk

1. Introduction

Advances in device technology, radio transceiver designs and integrated circuits along with evolution of simplified, power efficient network stacks have enabled the production of small and inexpensive wireless sensor devices [1–4]. These small devices can be networked together to enable a variety of new applications that include environmental monitoring, seismic structural analysis, data collection in warehouses, traffic monitoring etc. Such networks should collect data (typically infrequently) from the sensors for long periods of time without requiring human intervention. The sensors must be low in cost and work within a limited energy budget. Therefore, in order to achieve network longevity, a primary concern in such networks is power management.

Depending upon the application, sensors may need to be spread over a large geographical area resulting in a *sparse* network. The sensor distribution can be homogeneous (uniform spread of sensors) or

* Corresponding author. Tel.: +1-510-666-3176; fax: +1-510-883-0270.

E-mail addresses: rcshah@eecs.berkeley.edu (R.C. Shah), sumit.roy@intel.com (S. Roy), sushjain@cs.washington.edu (S. Jain), wrb@cs.washington.edu (W. Brunette).

[1] Tel.: +1-206-221-5261; fax: +1-206-633-6504.

[2] Tel.: +1-206-543-1695; fax: +1-206-543-2969.

1570-8705/$ - see front matter
doi:10.1016/S1570-8705(03)00003-9

heterogeneous (islands of sensors separated by large distances). Sensors at each city intersection are an example of a homogeneous distribution while sensors for habitat monitoring [5] are distributed heterogeneously. Possible approaches to ensure connectivity in such sparse networks include:

- Install multiple base stations to relay the data from sensor nodes in their coverage area.
- Deploy enough sensors to effectively form a *dense* connected network [6].

The base station approach trades off high communication power needed by the sensors with the cost of installing additional stations. On the other hand, deploying cheap nodes to form a dense, fully-connected ad hoc network may not be cost-effective either. The proposed architecture in this paper seeks to retain the advantages of both approaches—i.e. achieve cost-effective connectivity in sparse sensor networks while reducing the power requirements at sensors.

The key to making this feasible is the ubiquitous existence of mobile agents [7] in many of our target scenarios that we term MULEs (Mobile Ubiquitous LAN Extensions) [8]. In the case of traffic monitoring application, this role is served by vehicles (cars, buses) outfitted with transceivers; in a habitat monitoring scenario, animals can perform this role. MULEs are assumed to be capable of *short-range* wireless communication and can exchange data from a nearby sensor or access point they encounter as a result of their motion. Thus MULEs can pick up data from sensors when in close range, buffer it, and drop off the data to wired access points when in proximity.

The primary advantage of our approach is the potential of large power savings that can occur at the sensors because communication now takes place over a short-range. Promising new radio technologies like Ultra-Wideband (UWB) [9] which operate at extremely low-power with large burst data capacity are potentially suited for sensor to MULE communication. The primary disadvantage of this approach, however, is increased latency because sensors have to wait for a MULE to approach before the transfer can occur. Nevertheless for many data collection applications (that require data for analysis purposes only on the order of hours or even a day) such increased latency is acceptable. The proposed three-tier MULE architecture is thus suitable for such delay-tolerant scenarios where power budgets at the sensor are the over-riding constraint. Note that the above argument does not address the issue of energy consumed during radio listening. This can be potentially high because a sensor has to continuously listen to identify when a MULE passes by. The same issue occurs in ad hoc networks also, where a node has to continuously listen because it may have to forward some other node's data. Many researchers are working on addressing this issue for ad hoc networks [2,10]. We believe that the ideas can be extended to our architecture also and hope to address this more fully in future.

The relative strengths and weaknesses of various approaches for data collection in sparse sensor networks are qualitatively summarized in Table 1. In the base station approach there are a few base stations (same as access points) that cover the entire geographical area and each sensor communicates directly with the nearest base station. In the ad hoc network approach, enough sensor nodes are present to form a network. The sensors then send their data to the wired access points by multi-hop routing over this ad hoc network. Note that while the MULE approach suffers from higher latency, it has both low sensor power consumption

Table 1
Performance of different approaches for data collection in sparse wireless sensor networks

Approaches	Performance metrics			
	Latency	Sensor power	Data success rate	Infrastructure cost
Base stations	Low	High	High	High
Ad hoc network	Medium	Medium–low	Medium	Medium–high
MULE	High	Low	Medium	Low

and low infrastructure cost; characteristics that may be important for many applications.

The use of mobility to improve performance in ad hoc networks has been considered previously in different contexts [7,11–14]. The primary objective has been to provide intermittent connectivity in a disconnected ad hoc network. Theoretical capacity of such networks was considered in [14] and it was shown that mobility can provide scalable throughput at the cost of latency. Controlling mobile nodes to achieve connectivity and efficiency has been discussed in [11,13]. However, the application of mobility to the domain of sensor networks is relatively new and has not been addressed in detail; the ZebraNet project [5] and the Manatee project [15,16] are also exploring the idea of using mobility in sensor networks. The primary difference is that we derive analytical results to understand the scalability of performance metrics. Another difference is in the data delivery model. These projects focus on ensuring the data reaches all access points, whereas the MULE architecture tries to deliver data to only one access point.

The next section gives an overview of the MULE architecture. The rest of the paper focuses on modeling the system to obtain initial insights into the performance of such an architecture. The goal of modeling was to understand the scaling of the system characteristics as the parameters (number of sensors, MULEs etc.) change. The model chosen was very simple, which enabled us to obtain closed form analytical results for many quantities of interest, including data success rate (the fraction of generated data that reaches access points), latency and buffer occupancies at MULEs and sensors. In addition to detailing the analysis, system simulation results are also presented. These verify the analysis while providing some more insight into system performance. The paper finally concludes with the insights gained from the modeling analysis and simulation results and outlines future research directions based on this initial work.

2. The MULE three-tier architecture

The MULE architecture provides wide-area connectivity for a sparse sensor network by exploiting mobile agents such as people, animals, or vehicles moving in the environment. These mobile nodes act as an intermediate layer to form a packet transport system to connect the network. This creates a three-tier layered abstraction (Fig. 1) that can be adjusted to different types of situations and distribution needs:

- A top tier of WAN connected devices,
- a middle tier of mobile transport agents and
- a bottom tier made of fixed wireless sensor nodes.

The top tier is composed of access points that upload the data cached on a MULE to the WAN. These devices can be set up at convenient locations where network connectivity and power are available. Access points can communicate with a central data warehouse that enables them to synchronize the data that they collect, detect duplicates, and return acknowledgments to the MULEs (acks may be necessary to ensure reliability of data for certain applications).

The middle tier is made up of mobile transport agents that provide the network with connectivity as well as give the system scalability and flexibility for a relatively low cost. MULEs are assumed to be serendipitous agents whose movements cannot be predicted in advance. When the MULEs' motion bring them into the same vicinity as the sensors, they upload the data and carry it until it can

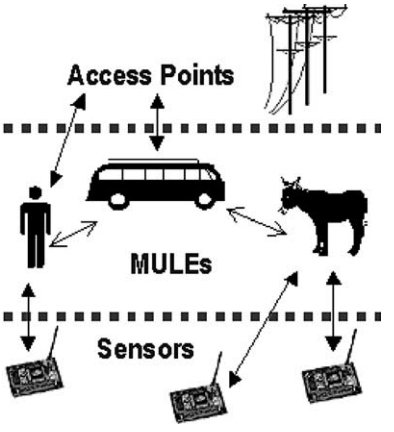


Fig. 1. The MULEs three-tier architecture.

be delivered to an access point. MULEs have large storage capacities (relative to sensors), renewable power, and the ability to communicate with both sensors and access points. To improve system performance MULEs can exchange data with each other, allowing a multi-hop MULE network to be formed, thereby reducing the latency times on the MULEs and increasing overall reliability from replication of data.

The bottom tier of the network consists of randomly distributed wireless sensors. Sensors communicate using a short-range radio and are limited in both power and memory. Work performed by these sensor nodes should be minimized as they have the most constrained resources of any of the tiers.

Depending on the application requirements and environment, a number of tiers in our three-tier abstraction can be collapsed onto one device. This expands the situations to which our architecture view can be applied. For example, to reduce latencies in the traffic monitoring application, the MULEs can be equipped with an always-on connection (such as a cellular or satellite phone) which would allow them to act as the top and the middle tier.

Other key advantages of the MULE architecture are its robustness and scalability as compared to centralized solutions. The three-tier abstraction allows increased reliability as the redundant access points and multiple MULEs create a fault-tolerant design wherein failures merely reduce throughput and increase latency. No sensor depends on any individual MULE, hence failure of any particular MULE does not disconnect the sensor from the sparse network. The MULE architecture is also easily scalable as deployment of new sensors or MULEs requires no network configuration and (most importantly) obviates the need for algorithmic scalability for key functions such as routing of packets. Additional MULEs and access points can easily be deployed to increase throughput and decrease latency.

Acknowledgments can be used to improve reliability. One can choose to use an end-to-end or tier-to-tier acknowledgment system. In a "tier-to-tier" acknowledgment system, the MULEs ack the sensor and the access points ack the MULEs in turn. It has the limitation that the MULEs may fail at any time without delivering the data to the access points. Another option is end-to-end acknowledgments; however, the high variability in end-to-end latency causes a challenge in determining when to retransmit data.

In summary, the benefits of our system include:

- Far less infrastructure than a fixed base station approach. For applications with few sensors spread over a large area, the cost savings could be orders of magnitude.
- Sensors do not have any overhead associated with routing packets from other sensors. For large ad hoc networks, the routing overhead can lead to a substantial increase in energy consumption at a node.
- Given a sufficient density of MULEs, the system is more robust than a traditional fixed network. Since sensors only rely on MULEs and MULEs are interchangeable, the failure of any number of MULEs does not mean connectivity failure. The failure merely increases the latency and decreases the data success rate of the network.
- System flexibility allows the same transport medium to be used simultaneously by different applications. The MULE system can be viewed as a mobile transport mechanism for connecting heterogeneous nodes.
- Data transfers only occur at short distances leading to power savings.

The drawbacks of our system are:

- Latency for this type of network is high and limits the types of situations this solution would be applicable for. Deterministic delay bound guarantees seem feasible only if MULEs traverse fixed routes.
- The system presupposes a sufficient amount of physical movement in the environment, which is a property in many sensor network environments.
- While no network is guaranteed to successfully deliver data all the time, our serendipitous network can encounter unexpected failures such as loss of a MULE or inability to reach sensors because of change in terrain causing limitations in mobility.

3. System modeling

We now focus on a simple and fully discrete (in time and space) model of the network that nevertheless allows us to investigate system performance as the parameters are scaled. Fig. 2 shows a pictorial representation of different system components. We make the following assumptions in our modeling:

- The underlying topology on which sensors, MULEs and access points are placed is assumed to be a discrete and finite two-dimensional grid. Further, for analytical simplicity the planar topology is assumed to be the surface of a torus (i.e. the grid is wrapped in both the north–south and the east–west direction).
- Only a fraction of the grid points are occupied by sensors and access points. The access points are modeled to be uniformly spaced on the grid while the sensors are randomly distributed.
- The network evolves synchronously with a global clock. At every clock tick the following events take place:
  - Sensors generate one unit of data.
  - Every MULE moves on the grid.
- The MULE motion is modeled as a simple symmetric random walk on the grid. At every clock tick, a MULE moves with equal probability to any of the four neighbors of its current grid position.
- The MULEs communicate with the sensors or access points only when they are co-located at the grid points.

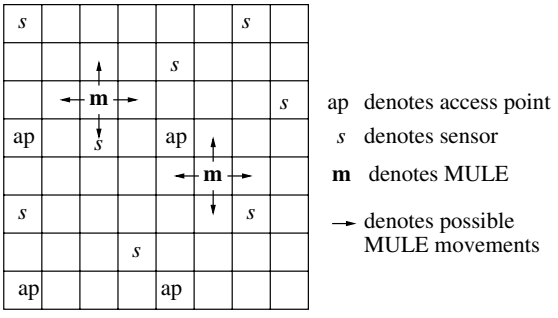


Fig. 2. A two-dimensional grid with the different system components.

- We assume sufficiently large bandwidth between the MULE and the sensor so as to transfer all the data residing at the sensor in one contact. Although over-simplistic we believe that for certain environments with less data to transfer this is a practical assumption.
- We ignore reliability issues and assume that the communication is error-free.
- MULEs move independent of each other and do not exchange any data among themselves when they intersect (occupy the same grid point).
- Both sensors and MULEs have buffers to store data. The sensors store generated data in its buffer if it has space otherwise the new data is dropped. Similarly any data transferred from sensors to MULEs is placed in the MULE buffer only if space is available, else it is dropped. Initially all buffers are empty.

Based on this model, we analyze the performance of the system as the grid size, number of access points and the number of MULEs are changed. The performance measures that we focus on are:

- *Data Success Rate:* It measures the fraction of generated data at the sensors that the system is able to transfer to the access points. In an ideal system all the data generated by the sensors would be transferred to the access points. This would yield a data success rate of one.
- *Latency:* Latency has two components—the latency on a sensor before a data packet is picked up by a MULE, and the latency on a MULE before the data is dropped off at the access point. Ideally, we would like both components to be as small as possible.
- *Buffer Sizing:* As mentioned earlier both sensors and MULEs have buffers. While small buffers could lead to high packet drop rates, reducing the data success rate, large buffers have an associated penalty in terms of energy consumption, physical size and manufacturing costs. Thus we would like to determine minimum buffer sizes that would ensure high data success rate while being cost-effective.

The model presented above is very simple and excludes many real-world aspects such as radio

propagation, link failure and bandwidth constraints. Another major concern is the choice of mobility model for analysis. We realize that a discrete random walk is not an accurate representation of the motion of vehicles, people etc. However, the simplicity of this model enables us to obtain closed-form results for the quantities of interest, giving us insight into system scalability. Also as mentioned in a recent survey [17], random walk is a widely used mobility model which is useful in modeling the unpredictable motion of entities. We hope to develop a more sophisticated stochastic model which can incorporate more generalized mobility models such as Smooth Random Mobility Model [18] or Brownian motion with drift [19,20]. However, note that with the increasing complexity of mobility models the hope of closed form analysis diminishes and one has to rely primarily on simulation. Thus we believe that a first order analysis with our simple model provides us with a useful base.

4. Glossary of notation and symbols

This section lists all the commonly used symbols and notation in this paper:

$(X_n)_{n \geqslant 0}$	a discrete-time Markov chain
$\mathscr{S}$	state space of the Markov chain
p_{ij}	the transition probability $P\{X_{n+1} = j \mid X_n = i\}\ \forall i, j \in \mathscr{S}$
$\pi = (\pi_i : i \in \mathscr{S})$	stationary distribution for the Markov chain
$\lvert A \rvert$	the cardinality of a set A
N	the number of points on the grid, i.e. the grid is $\sqrt{N}$ on a side
N_{mules}	the number of MULEs in the system
N_{AP}	the number of access points (AP) in the system
N_{sensors}	the number of sensors in the system
ρ_{mules}	the ratio of the number of MULEs to the grid size (N_{mules}/N); $(0 \leqslant \rho_{\text{mules}} \leqslant 1)$
ρ_{AP}	the ratio of the number of access points to the grid size (N_{AP}/N); $(0 \leqslant \rho_{\text{AP}} \leqslant 1)$
ρ_{sensors}	the ratio of the number of sensors to the grid size (N_{sensors}/N); $(0 \leqslant \rho_{\text{sensors}} \leqslant 1)$
MB	the total buffer capacity on a MULE (in number of packets)
SB	the total buffer capacity on a sensor (in number of packets)
AP	access point
H_i	the hitting time to a sensor i in the grid, i.e. the time taken by a MULE starting from the stationary distribution to first hit i when there is only one MULE in the system
R_i	the inter-arrival time to a sensor i in the grid, i.e. the time between consecutive MULE arrivals to i when there is only one MULE in the system
$H_i^{N_{\text{mules}}}$	the hitting time to a sensor i in the grid by any mule when there are N_{mules} in the system
$R_i^{N_{\text{mules}}}$	the inter-arrival time at a sensor i in the grid by any mule when there are N_{mules} in the system
R_{AP}	the time taken by a particular MULE to start from the set of access points and return back to it
Z_i	the buffer occupancy for a sensor i with $\text{SB} = \infty$ when a MULE visits it
$M^{(k)}$	the buffer occupancy for MULE k on one excursion from the set of access points back to the set. If there is only one MULE, then we will drop the superscript for convenience
S	the data success rate of the system, which is the fraction of generated data that reaches the access points
D_{s}	the latency experienced by the data packet at the sensor
D_{m}	the latency experienced by the data packet on a MULE

5. Basic results

The simplest scenario consists of one access point ($N_{\text{AP}} = 1$) and one MULE ($N_{\text{mules}} = 1$) in the system. We assume that the MULE and the sensors have infinite buffer capacity. The AP is at some position (the exact position is not critical) in the grid of size $\sqrt{N}$ on a side. The MULE is assumed to perform a simple symmetric random walk on the grid. The state space $\mathscr{S}$ consists of the points on the grid scanned in any order to form

a vector of length N (i.e., $|\mathscr{S}| = N$). This simple model allows us to apply the large body of relevant results from discrete-time, finite state Markov chains. We rely on the stationary distribution $\pi = (\pi_i : i \in \mathscr{S})$ to estimate average values of the quantities of interest.

The transition probabilities for the Markov chain with state space $\mathscr{S}$ are:

$$p_{ij} = \begin{cases} \frac{1}{4} & \text{if } (i,j) \text{ has an edge,} \\ 0 & \text{otherwise.} \end{cases} \quad (1)$$

Since $\sum_{i\in\mathscr{S}} \pi_i = 1$ and all states are equiprobable (i.e. $\pi_i = \pi_j \ \forall i,j \in \mathscr{S}$), we get,

$$\pi_i = \frac{1}{N}. \quad (2)$$

We next compute the following:

- average inter-arrival time at a sensor node i, $E[R_i]$,
- average length that the MULE traverses before it returns to the AP, $E[R_{\mathrm{AP}}]$,
- average number of data samples the MULE picks up during one traversal, $E[M]$.

The average time it takes for the MULE to return to the same sensor node i is the inverse of the stationary probability by Markov chain theory. Therefore,

$$E[R_i] = \frac{1}{\pi_i} = N. \quad (3)$$

Since a unit data is generated every clock tick, this is also the average value of the buffer occupancy at the sensor $E[Z_i]$ *when* the MULE visits it (because $\mathrm{SB} = \infty$, so the buffer occupancy is the same as the amount data generated). Note that this is the average value of the sensor buffer occupancy observed only when the MULE visits the sensor, not over all instants of time (the second quantity is not of much use in analyzing the system and is also harder to characterize).

Similarly, the average number of steps the MULE takes before returning to the access point is

$$E[R_{\mathrm{AP}}] = \frac{1}{\pi_{\mathrm{AP}}} = N. \quad (4)$$

The number of data samples the MULE picks up during one traversal depends on three things—the length of the traversal R_{AP}, number of sensors encountered which depends on ρ_{sensors} and the buffer occupancy at the sensors Z_i. Since the three quantities are independent, the average is simply given by (since $\mathrm{MB} = \infty$),

$$\begin{aligned} E[M] &= E[R_{\mathrm{AP}}] \cdot \rho_{\mathrm{sensors}} \cdot E[Z_i] \\ &= E[R_{\mathrm{AP}}] \cdot \rho_{\mathrm{sensors}} \cdot E[R_i] = \rho_{\mathrm{sensors}} N^2. \end{aligned} \quad (5)$$

The above results provide useful preliminary insights into the performance of the system as the grid is scaled. Clearly, the time between MULE visits to a sensor grows linearly with the grid size as shown in (3). This has two implications. Firstly, the required buffer at the sensor needs to scale with the grid size to prevent loss of data. [3] Secondly, the latency for data samples also increases with the grid size. Both these problems can be mitigated by having multiple MULEs in the system, a case considered in Section 7.

The second insight is that with only one access point in the system, the length of MULE excursions that begin and end at the AP grows linearly as shown in (4). Similar to the case above, there are two implications. The first is that the required MULE buffer needs to be large to prevent loss of data. In fact, the required buffer size grows as the square of the grid size as shown by (5) above (again we use $E[M]$ to get an idea of the buffer sizes needed to avoid packet drops). The second implication is that the latency for the data when traveling from the sensor to the access points grows linearly. This means that the number of access points in the system needs to scale with the grid size, a case considered in Section 6.

[3] Notice that while we assume $\mathrm{SB} = \infty$, in reality the buffer capacity has to be finite but sufficiently large to avoid packet drops. Thus we use $E[R_i]$ to provide an indication of *sufficiently large*.

6. Scaling with number of access points

In this section, we analyze the effect of multiple access points in the system. We assume that the access points are spaced at a distance of $\sqrt{K}$ points on the grid in both the x and y directions. Therefore, $K = N/N_{\mathrm{AP}} = 1/\rho_{\mathrm{AP}}$. We still assume that only one MULE is present in the system.

Result 1. *If the access points are regularly spaced at a distance of $\sqrt{K}$ points on the grid in both the x and the y directions, then the expected length of excursion for the MULE starting from the set of access points till it reaches the set again* (*could be the same* AP *or another one*),

$$E[R_{\mathrm{AP}}] = K = \frac{1}{\rho_{\mathrm{AP}}}. \tag{6}$$

Proof. Looking at the symmetry of the grid in Fig. 3, we can reduce the state space to a smaller grid of size $\sqrt{K} \times \sqrt{K}$ as shown in Fig. 4. This can be seen to be the result of folding the entire grid onto the smaller box containing only one access point A (which represents all the access points). This is possible because from the perspective of a MULE, all access points are equivalent. The resultant grid also remains a torus (wraps around in the north–south and east–west directions).

As in Section 5 the stationary distribution for a node i in this reduced grid (size $\sqrt{K} \times \sqrt{K}$) can be shown to be

$$\pi_i = \frac{1}{K}. \tag{7}$$

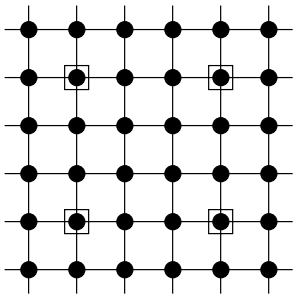


Fig. 3. A two-dimensional grid with the squares representing the positions of the access points.

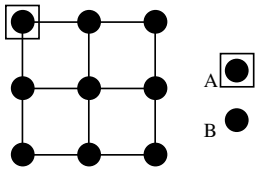


Fig. 4. Folded version of the two-dimensional grid to form a smaller grid (the types of nodes and their transition probabilities are also shown).

Using this stationary distribution, the return time to the point "A" can be calculated. This is also the required excursion time of the MULE from the AP set to the AP set since the point "A" represents all the access points of the original grid.

$$E[R_{\mathrm{AP}}] = \frac{1}{\pi_{\mathrm{A}}} = K = \frac{1}{\rho_{\mathrm{AP}}}. \qquad \square$$

Thus we see that the MULE excursion length between the access point set is independent of the grid size as long as the number of access points scale as a fraction of the grid size.

7. Scaling with number of MULEs

In this section, we analyze the case when there are multiple MULEs in the system. The fraction of MULEs in the system is kept constant as the size of the grid is increased, i.e., $N_{\mathrm{mules}}/N = \rho_{\mathrm{mules}}$. We first calculate the average number of visits observed at a sensor per unit time. We then calculate the expected inter-arrival times for MULEs to a sensor. That will extend the result (3) obtained in Section 5. As mentioned before, we assume that all the MULEs are performing independent random walks, with no communication among each other. Also, note that every MULE starts in the stationary distribution, and subsequently performs a random walk, thus remaining in the stationary distribution.

Now consider a sensor and a particular MULE M_0. Then the probability that M_0 intersects the sensor is given by

$$P\{M_0 \text{ intersects sensor}\} = \frac{1}{N}. \tag{8}$$

Define:

$$Y_k = \begin{cases} 1 & \text{if one or more MULEs intersects the sensor at time } k, \\ 0 & \text{if no MULE intersects the sensor at time } k. \end{cases} \tag{9}$$

Hence the probability that no MULE intersects with the sensor is given by

$$P\{Y_k = 0\} = \left(1 - \frac{1}{N}\right)^{N_{\text{mules}}}$$

$$\Rightarrow P\{Y_k = 1\} = 1 - \left(1 - \frac{1}{N}\right)^{N_{\text{mules}}}. \tag{10}$$

Therefore the expected number of MULE visits to a sensor per unit time is, [4]

$$\lim_{n\to\infty} \frac{1}{n} E\left[\sum_{k=0}^{n-1} 1_{\{Y_k=1\}}\right] = \lim_{n\to\infty} \frac{1}{n} \sum_{k=0}^{n-1} P\{Y_k = 1\}$$

$$\approx 1 - \left(1 - \frac{1}{N}\right)^{N_{\text{mules}}}$$

$$\approx 1 - e^{-\rho_{\text{mules}}} \quad (\text{large } N) \tag{11}$$

$$\approx \rho_{\text{mules}} \quad (\text{small } \rho_{\text{mules}}). \tag{12}$$

Result 2. *The average inter-arrival time between MULE visits to a sensor i when there are N_{mules} in the system is given by*

$$E[R_i^{N_{\text{mules}}}] = \frac{1}{1 - \left(1 - \frac{1}{N}\right)^{N_{\text{mules}}}} \tag{13}$$

$$\approx \frac{1}{1 - e^{-\rho_{\text{mules}}}} \quad (\textit{large } N) \tag{14}$$

$$\approx \frac{1}{\rho_{\text{mules}}} \quad (\textit{small } \rho_{\text{mules}}). \tag{15}$$

Proof. To find the average inter-arrival time at a sensor i, we consider the Markov chain composed of the product of the Markov chains of each of the MULEs. Thus the new state space is given by

$$S' = \underbrace{S \times S \times \cdots \times S}_{N_{\text{mules}} \text{times}}.$$

In the modified state space S', we are interested in the set of states A which represent one or more MULEs intersecting i. Since all the states are equally likely, the stationary distribution for the set A can be calculated as

$$\pi(A) = \frac{|A|}{|S'|} = \frac{|S'| - |S' - A|}{|S'|}$$

$$= \frac{N^{N_{\text{mules}}} - (N-1)^{N_{\text{mules}}}}{N^{N_{\text{mules}}}}$$

$$= 1 - \left(1 - \frac{1}{N}\right)^{N_{\text{mules}}}. \tag{16}$$

Thus, using Kac's formula [21], the average inter-arrival time between MULE visits to a sensor i is

$$E[R_i^{N_{\text{mules}}}] = \frac{1}{\pi(A)} = \frac{1}{1 - (1 - \frac{1}{N})^{N_{\text{mules}}}}. \qquad \square$$

Corollary 3. *Average buffer occupancy on a sensor (with sufficiently large buffer capacity) can now be calculated as*:

$$E[\textit{Sensor Buffer}] = E[R_i^{N_{\text{mules}}}] \approx \frac{1}{\rho_{\text{mules}}}. \tag{17}$$

Here we have used the observation that the sensor buffer occupancy at the times of MULE visits is exactly the same as the inter-arrival times between MULEs. Hence the average values are also the same. Also note that this is just the average buffer occupancy seen at the times of MULE arrivals at the sensor; not at all times.

Corollary 4. *Average buffer occupancy on a MULE (with sufficiently large buffer capacity) can also be calculated as*:

$$E[\textit{Mule Buffer}] = \rho_{\text{sensors}} E[R_{\text{AP}}] E[R_i^{N_{\text{mules}}}]$$

$$\approx \frac{\rho_{\text{sensors}}}{\rho_{\text{AP}} \rho_{\text{mules}}}. \tag{18}$$

Similar to the previous corollary, we use the expected value of the inter-arrival times at a sensor as the expected value of the sensor buffer occupancy when a MULE visits it. Again similar to the sensor buffer occupancy, this is the average buffer occupancy

[4] Multiple MULEs intersecting the sensor at the same time is considered to be just one intersection.

on the MULE as seen at the times of MULE intersections with an AP; *not at all times. Thus this is the average amount of data that is picked up by the MULE during one excursion between the* AP *set.*

It is interesting to note that the problem of increasing buffer requirements at the sensor as the grid increases which we encountered in Section 5 is eliminated. As long as ρ_{mules} remains constant, the buffer requirements remain the same. So far we have just found the average value of the inter-arrival times for MULEs to a sensor. We next need to obtain the probability distribution. However, we first find the probability distribution for the hitting time at a sensor as that is needed for the result on the inter-arrival times.

7.1. Hitting time distribution at a sensor

For our purposes, the hitting time for a sensor i is defined as the first time a MULE hits i when all the MULEs start from the stationary distribution. We first find the probability distribution of the hitting time for a system with a single MULE before evaluating the general case of multiple MULEs. [21] shows that the mean of the hitting time for a single MULE is $\Theta(N \log N)$ for a simple symmetric random walk on the surface of a torus. Furthermore, the distribution of hitting times for an ergodic Markov chain can be approximated by an exponential distribution of the same mean [21]. Therefore,

$$P\{H_i > t\} \approx \exp\left(\frac{-t}{cN \log N}\right), \tag{19}$$

where the constant $c \approx 0.34$ as $N \to \infty$ (valid for $N \geqslant 25$) [22]. Note that this result uses the continuous time version of the discrete-time Markov chain, but the result is still correct for the discrete-time case [21]. However, writing in continuous time simplifies the analysis considerably, thus all the hitting and return time probability distribution results will be for the continuous time chain. Using this we can now extend the result for the case when there are $N_{\text{mules}} (> 1)$ in the system.

Result 5. *The hitting time for a sensor i when there are* N_{mules} *in the system, all of which start in the stationary distribution is given by*

$$P\{H_i^{N_{\text{mules}}} > t\} \approx \exp\left(\frac{-t}{0.34 \frac{N}{N_{\text{mules}}} \log(N)}\right). \tag{20}$$

Proof. Let $H_i^{(k)}$ denote the hitting time to sensor i for a single MULE k. Then

$$H_i^{N_{\text{mules}}} = \min_{k \in \text{MULEs}} H_i^{(k)}. \tag{21}$$

Thus, we obtain

$$\begin{aligned} P\{H_i^{N_{\text{mules}}} > t\} &= [P\{H_i > t\}]^{N_{\text{mules}}} \\ &\approx \left[\exp\left(\frac{-t}{0.34N \log(N)}\right)\right]^{N_{\text{mules}}} \\ &= \exp\left(\frac{-t}{0.34 \frac{N}{N_{\text{mules}}} \log(N)}\right). \end{aligned}$$

□

7.2. Inter-arrival time distribution at a sensor

To find the inter-arrival time distribution at a sensor i, we first consider the case when there is only one MULE in the system. In that case, the inter-arrival time at i is the same as the return time R_i for the MULE. Unfortunately, there is no closed form result for the distribution, but can only be approximated as $\pi/\log t$ for $t \to \infty$ for an infinite grid [23]. For smaller times and for finite grid sizes, this only provides a very loose upper bound on the tail probability.

To obtain a better characterization we derive a recursive equation to compute $P\{R_i = t\}$ (inter-arrival time distribution for a single MULE). Let the initial position of the MULE be at the grid position 0. Define $L_{i,j}(t)$ to be the number of paths starting from i and ending at j of length t, avoiding the point 0 at all the intermediate steps. Also, let the neighbors of a node k in the torus be denoted by the set $\mathcal{N}(k)$. Then, without loss of generality, for any sensor node i,

$$P\{R_i = t\} = L_{0,0}(t)/4^t. \tag{22}$$

In the above equation, $L_{0,0}(t)$ denotes the total number of valid paths that return to 0 in t steps and 4^t denotes the total number of possible paths of t steps. The following recursive equation can now be used to compute $L_{0,0}(t)$:

$$L_{i,j}(t) = \sum_{k \in \mathcal{N}(i) \wedge k \neq 0} L_{k,j}(t-1), \quad t > 1,$$

$$L_{i,j}(1) = \begin{cases} 1 & \text{if } j \in \mathcal{N}(i), \\ 0 & \text{otherwise.} \end{cases}$$

Result 6. *If the number of MULEs in a system is N_{mules}, the inter-arrival time at a sensor i can be written as*

$$P\{R_i^{N_{\text{mules}}} > t\} \approx P\{H_i^{N_{\text{mules}}-1} > t\} \cdot P\{R_i > t\}. \quad (23)$$

Proof. To find the inter-arrival time distribution at a sensor i, we consider only the moments at which one MULE intersects the sensor. We ignore multiple MULEs at the sensor which is a very unlikely event for low mule densities. At this instant in time, the rest of the MULEs are in the stationary distribution. Thus,

$$R_i^{N_{\text{mules}}} = \min(R_i, H_i^{N_{\text{mules}}-1}),$$

since the MULE at the sensor has to return to the sensor, but for the $(N_{\text{mules}} - 1)$ remaining MULEs, it is identical to hitting the sensors starting from stationarity. The result follows from this observation. □

7.3. Return time distribution to the access points set

We now compute the distribution of the excursion times of a MULE between the access point set. As in Section 6 we consider the folded torus (Fig. 4) in which all the access points are represented as a single grid point. Since this point represents the set of all access points, we need to compute the return time distribution to this single grid point. For this we can apply (22) to the folded torus to obtain the required return time distribution. Thus,

$$P\{R_{\text{AP}} = t\} = L_{0,0}(t)/4^t \quad (24)$$

with $L_{0,0}(n)$ defined on the surface of the folded grid of Fig. 4.

8. Data success rate

We now have the pieces in place to calculate the data success rate. We define the data success rate as the ratio of the average amount of data delivered to the access points by time t to the total data generated by time t as $t \to \infty$.

Result 7. *The data success rate of the system is given by*

$$S = \sum_{k \in \text{MULEs}} \frac{E\left[\min(\rho_{\text{sensors}} \sum_{i=1}^{R_{\text{AP}}} \min(R_i^{N_{\text{mules}}}, \text{SB}), \text{MB})\right]}{E[R_{\text{AP}}] N_{\text{sensors}}}. \quad (25)$$

Proof. We use renewal reward theory [24] to derive data success rate. One excursion of the MULE from the access point set back to the set is considered a cycle. Therefore R_{AP} is the length of a cycle. Recall that the sensors generate data at the constant rate of one packet per unit time therefore the average data generated in the system per unit time is N_{sensors}. We now get the data success rate S as

$$S = \frac{E\left[\sum_{k \in \text{MULEs}} M^{(k)}\right]}{E[R_{\text{AP}}] N_{\text{sensors}}}.$$

Here,

$$M^{(k)} = \text{Data picked up by the MULE } k \text{in time } R_{\text{AP}} = \min\left(\rho_{\text{sensors}} \sum_{i=1}^{R_{\text{AP}}} Y_i^{(k)}, \text{MB}\right).$$

The min-function is included because the buffer capacity of the MULE bounds the total amount of data a MULE can carry. Now, $Y_i^{(k)}$ is the amount of data at a sensor visited by MULE k at time i. This is given by

$$Y_i^{(k)} = \min(Z_i, \text{SB}).$$

Similar to the previous step, the sensor buffer capacity bounds the amount of data that can be present at a sensor, hence the min-function. Also, since Z_i is the amount of data generated and not yet picked up at the sensor, it has the same distribution as the inter-arrival time at a sensor.

Hence, putting this all together,

$$S = \sum_{k \in \mathrm{MULEs}} \frac{E\left[\min(\rho_{\mathrm{sensors}} \sum_{i=1}^{R_{\mathrm{AP}}} \min(R_i^{N_{\mathrm{mules}}}, \mathrm{SB}), \mathrm{MB})\right]}{E[R_{\mathrm{AP}}] N_{\mathrm{sensors}}}.$$

□

9. Latency

Latency is an important performance metric of the MULE architecture. It has two components—latency at the sensor before data is picked up by a MULE and the latency on the MULE before it is delivered to an access point. We now consider each of these components in more detail.

9.1. Latency at sensor (D_s)

The latency at the sensor is the queueing delay experienced by a data packet once it is generated till the time a MULE picks up the packet. To calculate the latency, we just consider the packets that are picked up by the MULE; i.e., we do not consider the packets that are dropped due to the sensor buffer getting full before a MULE arrives. Depending on the queueing discipline followed at the sensor buffer, the latency would be different

9.1.1. Droptail queuing discipline

In this protocol, the sensor node stops generating any more data once the sensor buffer gets filled up. Thus this is similar to the droptail queueing discipline where new data is dropped at the end of the queue if it is full; hence the name.

Result 8. *For the droptail protocol, the distribution of the latency at a sensor is given by*

$$P\{D_s = t\} = P\{H_i^{N_{\mathrm{mules}}} = t\}. \tag{26}$$

Proof. When a data packet is generated, the MULEs are in stationary distribution around the grid. Also, once data is added to the queue, it is never dropped, hence the time the packet is in the sensor queue is equal to the time taken by the MULEs to hit the sensor. □

Corollary 9. *The average latency at the sensor for the droptail protocol is given by*

$$E[D_s] = E[H_i^{N_{\mathrm{mules}}}] = \frac{0.34N \log N}{N_{\mathrm{mules}}}. \tag{27}$$

This result follows directly from the mean of the exponential distribution.

9.1.2. Drophead queueing discipline

Unlike the previous queueing discipline, in this protocol, new data pushes out old data if the sensor buffer gets full. Thus the oldest data is the first to get dropped when new data is generated. This behavior may be suitable in cases where older data has less relevance than newer data. In this protocol, it is obvious that the maximum latency experienced by a data packet is equal to the sensor buffer size (after which it gets dropped).

Result 10. *For the drophead protocol, the distribution of the latency at a sensor is given by*

$$P\{D_s = t\} = \begin{cases} \frac{P\{H_i^{N_{\mathrm{mules}}} = t\}}{P\{H_i^{N_{\mathrm{mules}}} \leqslant \mathrm{SB}\}}, & t \leqslant \mathrm{SB}, \\ 0, & t > \mathrm{SB}. \end{cases} \tag{28}$$

Proof. We first note that when a data packet gets generated, the MULEs are in stationary distribution around the grid. However, if a MULE does not pick up the data within SB ticks, the data packet gets pushed out of the buffer. Hence we get

$$\begin{aligned} P\{D_s = t\} &= P\{H_i^{N_{\mathrm{mules}}} = t | H_i^{N_{\mathrm{mules}}} \leqslant \mathrm{SB}\} \\ &= \frac{P\{\{H_i^{N_{\mathrm{mules}}} = t\} \wedge \{H_i^{N_{\mathrm{mules}}} \leqslant \mathrm{SB}\}\}}{P\{H_i^{N_{\mathrm{mules}}} \leqslant \mathrm{SB}\}} \end{aligned}$$

which gives us the required result. □

Corollary 11. *The average latency at the sensor for the drophead protocol is given by*

$$E[D_s] = \sum_{t=1}^{\mathrm{SB}} \frac{t \cdot P\{H_i^{N_{\mathrm{mules}}} = t\}}{P\{H_i^{N_{\mathrm{mules}}} \leqslant \mathrm{SB}\}} \tag{29}$$

$$= \frac{1}{1 - \mathrm{e}^{-1/\overline{H}}} - \frac{\mathrm{SB} \cdot \mathrm{e}^{-\mathrm{SB}/\overline{H}}}{1 - \mathrm{e}^{-\mathrm{SB}/\overline{H}}}, \tag{30}$$

where

$$\overline{H} = E[H_i^{N_{\text{mules}}}] = \frac{0.34N\log(N)}{N_{\text{mules}}}.$$

9.2. Latency on MULE (D_{m})

Once a data packet is picked up by a MULE from a sensor, it experiences a delay before the MULE encounters an access point and drops off the data. This delay depends on the motion of the MULE as well as the proximity of the sensor to an access point. However, the next result gives the average distribution of latency on a MULE for data packets from all sensors.

Result 12. *The distribution of latency on a MULE for data picked up at a random sensor is given by*

$$P\{D_{\text{m}} > t\} \approx \exp\left(\frac{-t}{0.34\frac{N}{N_{\text{AP}}}\log(N)}\right). \tag{31}$$

Proof. As done earlier, we consider the folded torus (Fig. 4) in which all the access points are represented as a single grid point. Since data is picked up from all sensors, the latency is identical to starting in the equilibrium distribution and finding the time taken to hit the access point. Hence the result. □

Corollary 13. *The average latency on a MULE is given by*

$$E[D_{\text{m}}] = \frac{0.34N\log N}{N_{\text{AP}}}. \tag{32}$$

This result follows directly from the mean of the exponential distribution.

10. Simulation setup

A custom event driven simulator was written to verify the preceding analysis and also explore the conditions under which it holds. In this section we present a brief description of the simulator.

The simulator is a discrete event driven simulator where time is measured in abstract units of clock-ticks. The underlying grid structure is the surface of a torus with the size N specified during initialization. Depending on the values of ρ_{sensors} and ρ_{mules}, appropriate number of sensors and MULEs are placed randomly on the grid in the beginning. Buffer sizes on both the sensors and the MULEs can also be specified and are completely empty when the simulation is started. Finally, the APs can be either randomly placed on the grid or regularly spaced, [5] with the number of APs depending on the value of ρ_{AP}. All the input parameters to the simulator are shown in Table 2. A summary of the various events handled by the simulator is given in Table 3.

The simulator also assumes a perfect radio channel, i.e., there is no loss of packets during transmission. The only way packets can be lost is if

Table 2
Input parameters to the simulator

Parameter	Description
Grid size	Number of points on the grid N
# of sensors	$= N\rho_{\text{sensors}}$
# of MULEs	$= N\rho_{\text{MULEs}}$
# of access points	$= N\rho_{\text{AP}}$
Sensor buffer size	Number of data samples each sensor can hold
MULE buffer size	Number of data samples each MULE can hold

Table 3
Events defined by the simulator

Event	Action
MULE motion	Changes grid position of the MULE
Data generation	Generates new data at the sensor and stores it in the buffer. If the sensor buffer is full data is dropped
MULE–sensor interaction	Transfers all data from the sensor to the MULE. If the MULE buffer is full, all the extra data is dropped
MULE–AP interaction	Transfers all data from the MULE to the AP

[5] Interestingly, simulations showed similar results for both uniform and random placement of access points.

the sensor or MULE buffers overflow. However, the sensors do not maintain any state (such as acks etc.) to implement reliability. Also there is no MULE to MULE interaction, even though they may occupy the same grid point.

11. Simulation results

In this section, simulation results are presented which verify all the major results of the analysis and also provide certain insights.

To verify scaling with access points, $E[R_{\mathrm{AP}}]$ was measured for a variety of grid sizes from 25×25 to 200×200. As expected, $E[R_{\mathrm{AP}}]$ remained constant across all grid sizes (Fig. 5) when ρ_{AP} was kept constant, verifying (6).

Fig. 6 shows the effect of scaling the number of MULEs on the average inter-arrival time to a sensor. As expected $E[R_i]$ remained constant for different grid sizes as long as the value of ρ_{mules} did not change, in accordance with (15).

Fig. 7 plots the cumulative distribution function of the hitting time $H_i^{N_{\mathrm{mules}}}$ for $\rho_{\mathrm{mules}} = 1\%$, 10% and 20% on a 20×20 grid. The figure verifies that using the hitting time result for the continuized chain is valid for the discrete-time case also. Similarly, Fig. 8 plots the cdf of $R_i^{N_{\mathrm{mules}}}$ for a 20×20 grid with the same values of ρ_{mules}. Finally, Fig. 9 plots the cdf of R_{AP} for a mule on a 20×20 grid where $\rho_{\mathrm{AP}} = 0.25\%$, 1% and 4%.

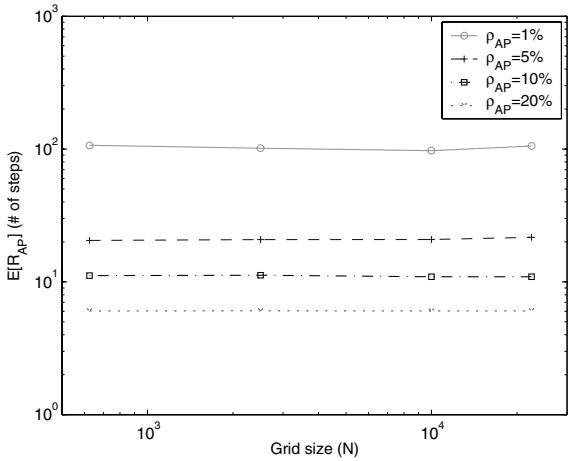


Fig. 5. $E[R_{\mathrm{AP}}]$ while scaling the grid size with $\rho_{\mathrm{AP}} = 1\%$, 5%, 10% and 20%.

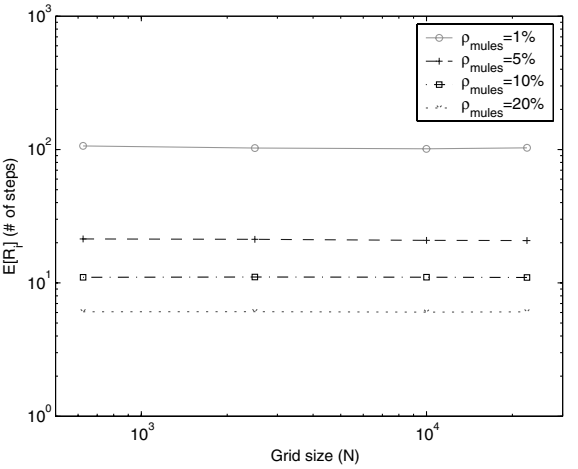


Fig. 6. $E[R_i]$ while scaling the grid size with $\rho_{\mathrm{mules}} = 1\%$, 5%, 10% and 20%.

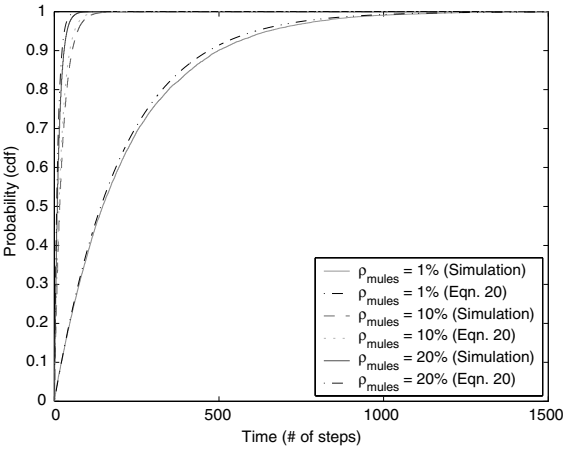


Fig. 7. Cdf of the hitting times ($H_i^{N_{\mathrm{mules}}}$) at a sensor (20×20 grid).

Figs. 10 and 11 plot the data success against the normalized MULE and sensor buffers respectively.

$$\text{Normalized MULE Buffer} = \frac{\text{Actual value of the MULE Buffer}}{E[\text{MULE Buffer}]}, \tag{33}$$

$$\text{Normalized Sensor Buffer} = \frac{\text{Actual value of the Sensor Buffer}}{E[\text{Sensor Buffer}]}. \tag{34}$$

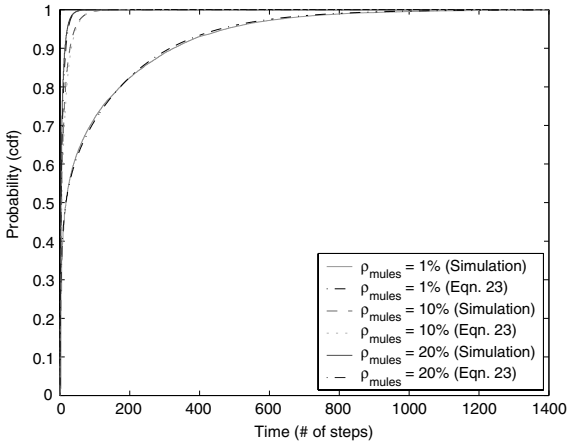


Fig. 8. Cdf of the inter-arrival times ($R_i^{N_{\text{mules}}}$) at a sensor (20×20 grid).

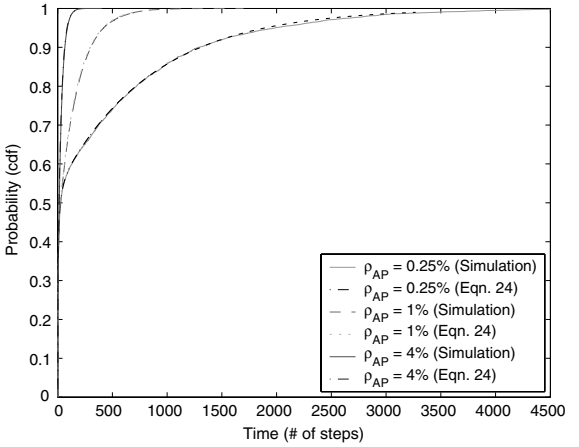


Fig. 9. Cdf of the return times (R_{AP}) for the access point set (20×20 grid).

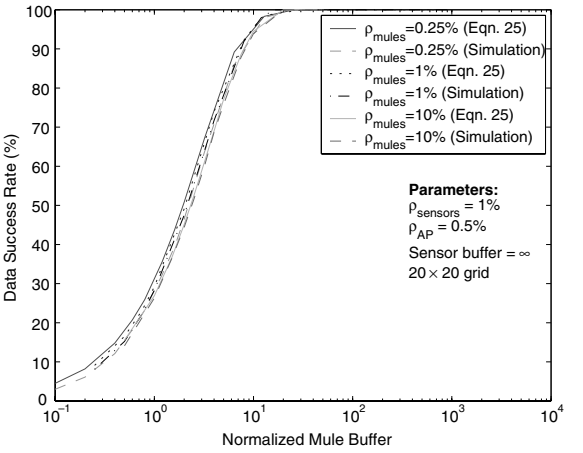


Fig. 10. Data success rate vs. normalized MULE buffer size for $\rho_{\text{mules}} = 0.25\%$, 1% and 10% ($20 \times 20$ grid).

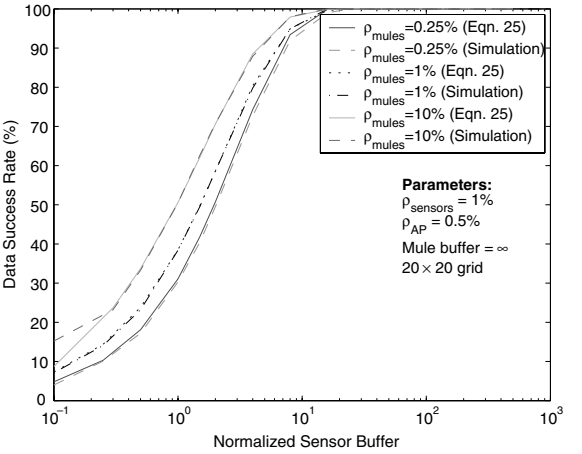


Fig. 11. Data success rate vs. normalized sensor buffer size for $\rho_{\text{mules}} = 0.25\%$, 1% and 10% ($20 \times 20$ grid).

For Fig. 10, the sensor buffer size was infinitely large. Note the steep drop-off of the data success rate with the MULE buffer size. Also, more than 95% data success rate is achieved when each MULE buffer is greater than $10E[M]$. Interestingly, the plot also shows that one can trade off the number of MULEs in the system with the amount of buffer capacity on each MULE. This is evident from the fact that the data success rate curves are roughly the same for different MULE densities, but reducing the number of MULEs by a factor k increases the expected MULE buffer size by k (and vice versa). This will obviously impact latency, as the sensors will have to wait longer (or shorter as the case may be) before a MULE comes by to pick up the data.

Similarly, for Fig. 11 the MULE buffer size was infinitely large. Again, a steep curve was obtained for the data success rate. Also, the data success rate saturates for each MULE density when the sensor buffer capacity reaches roughly $10E[R_i^{N_{\text{mules}}}]$. However, the figure shows that we cannot trade off a decrease in MULE density by increasing the

Table 4
Sample values of MULE and sensor buffer sizes for 50% and 90% data success rates

	ρ_{mules}	MULE buffer	
		50% data success rate	90% data success rate
Sensor buffer $= \infty$	0.25%	43,800	163,000
	1%	4280	14,800
	10%	480	1660
		Sensor buffer	
MULE buffer $= \infty$	0.25%	2030	7410
	1%	152	630
	10%	10	47

buffers at each sensor. Higher MULE densities lead to higher data success rates, in general, until the sensor buffers are sufficiently large.

The reason for the difference lies in the Law of Large Numbers. When $MB = \infty$, the drop in the data success rate is due to packets getting dropped at the sensors. Now as N_{mules} reduces, the inter-arrival time at a sensor grows larger, and consequently, there are larger amounts of data that are dropped due to sensor buffer overflow. On the other hand, when $SB = \infty$, data overflow occurs at the MULEs when the amount of data it picks up from all the sensors exceeds the MULE buffer capacity. However, due to the Law of Large Numbers, the probability of the total amount of data on the MULE exceeding the buffer threshold is smaller than in the finite sensor buffer case.

Table 4 shows the actual values of the buffer sizes needed to achieve data success rates of 50% and 90%. These are shown for both the cases of infinite MULE and infinite sensor buffers.

Figs. 12 and 13 show the cdf of the latency at a sensor for the droptail and drophead protocols respectively. The expected value of sensor buffer occupancy was 100 by (17). The simulations verify the results obtained by (26) and (28). The effect of the number of MULEs on the latency can also be seen in Fig. 14 where the average latency for the drophead protocol is shown against the number of MULEs. For large sensor buffer sizes, the reduction in latency is very steep initially as the inter-arrival time of MULEs reduces exponentially. However, for small sensor buffer sizes, the buffer tends to be nearly full for a small number of

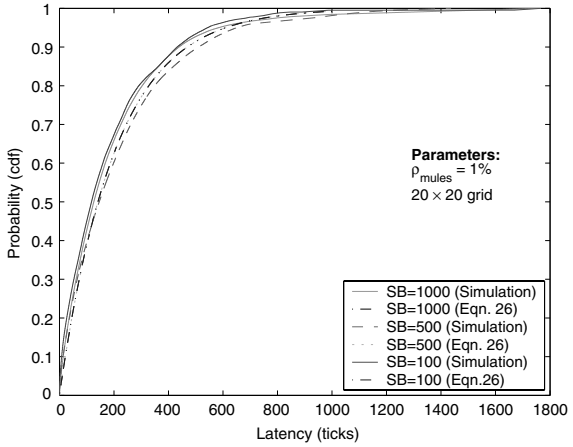


Fig. 12. Cdf of the latency at a sensor with droptail protocol for SB = 1000, 500 and 100 (20×20 grid).

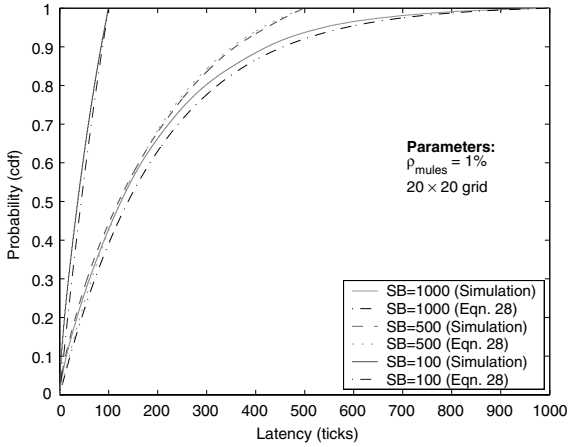


Fig. 13. Cdf of the latency at a sensor with drophead protocol for SB = 1000, 500 and 100 (20×20 grid).

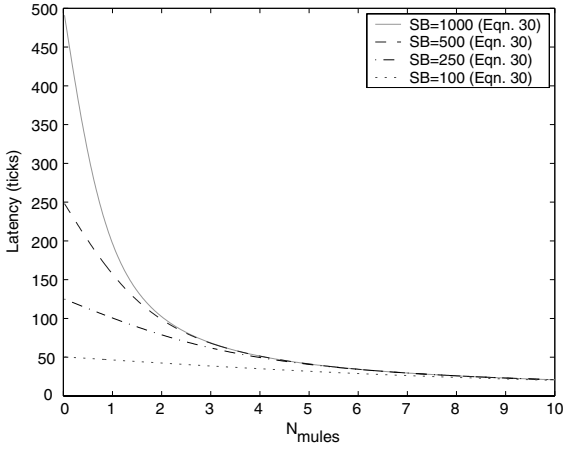


Fig. 14. Average latency on a sensor with drophead protocol for SB = 1000, 500, 250 and 100 (20×20 grid).

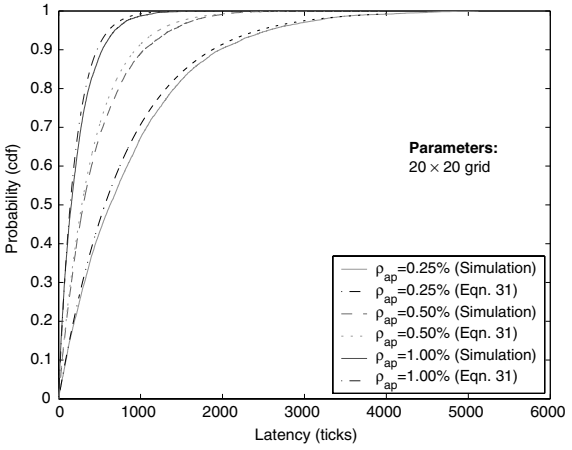


Fig. 15. Cdf of the latency on a MULE with $\rho_{AP} = 0.25\%$, 0.50% and 1% (20×20 grid).

MULEs, hence the average latency does not decrease dramatically. Fig. 15 plots the cdf of the latency on a MULE as the number of access points are varied. As expected, the simulations results follow (31).

12. Conclusion and future work

In this paper we have presented an architecture to connect sparse sensor networks at the cost of higher latencies. The main idea is to utilize the motion of the entities that are already present in an environment to provide a low-power transport medium for sensor data. After introducing the architecture, the focus of the paper was on presenting a simple analytical model based upon two-dimensional random walks to provide insight into various performance metrics (data success rate, latency and buffer sizes). Our key observations are:

- The sensor buffer requirement is inversely proportional to ρ_{mules}.
- The MULE buffer requirement is inversely proportional to both ρ_{mules} and ρ_{AP}.
- When the sensor buffer is large, the buffer capacity on each MULE can be traded off with the number of MULEs to maintain the same data success rate.
- The change in the buffer capacity on each sensor needs to be greater than the change in the number of MULEs to keep the same data success rate.
- The average latency at a sensor for the droptail protocol is inversely proportional to the number of MULEs.
- For a large sensor buffer, the average latency at the sensor (drophead protocol) drops dramatically with increase in the number of MULEs; for small sensor buffers, the decrease is linear.
- The average latency on a MULE is inversely proportional to the number of access points in the system.

We plan to expand our work in a couple of directions. One is to develop a more complete stochastic model to address some of the current simplifications such as infinite bandwidth, random-walk mobility model and error-free communication. Here we plan to use ideas from queuing theory and renewal processes.

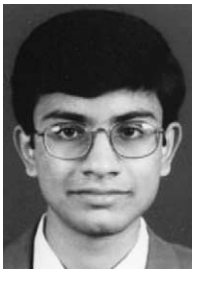
From the protocol point of view, MULE-to-MULE communication and reliability using acknowledgments are issues of interest. Another limitation of the current work is the assumption that the sensors have to continuously listen in order to identify a MULE's presence. Approaches to increase the sleep time for sensors, such as reduced duty cycle, need to be explored along with their effect on system performance.

Acknowledgements

We would like to thank Jason Jenks (University of Washington) and Anthony LaMarca (Intel Research, Seattle) for their active participation in defining the system architecture. We are thankful to Prof. Krzysztof Burdzy (Department of Mathematics, University of Washington, Seattle) and Prof. David Aldous (Department of Statistics, University of California, Berkeley) for providing us with information and references on non-interacting particle theory of Markov chains. Finally we are grateful to the reviewers for their comments. The work was partially supported by DARPA grant N66001-99-2-8924.

References

[1] D. Estrin et al., Embedded, everywhere: A research agenda for networked systems of embedded computers, Computer Science and Telecommunications Board (CSTB) Report, 2001.
[2] J. Rabaey et al., Picoradio supports ad hoc ultra-low power wireless networking, IEEE Computer 33 (7) (2000) 42–48.
[3] J. Kahn, R. Katz, K. Pister, Next century challenges: Mobile networking for smart dust, in: ACM/IEEE MobiCom, 1999.
[4] UC Berkeley TinyOS. Available from <http://webs.cs.berkeley.edu/tos/>.
[5] P. Juang et al., Energy-efficient computing for wildlife tracking: Design tradeoffs and early experiences with zebranet, in: International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS), 2002.
[6] O. Dousse, P. Thiran, M. Hasler, Connectivity in ad hoc and hybrid networks, in: IEEE Infocom, 2002.
[7] A. Vahdat, D. Becker, Epidemic routing for partially-connected ad hoc networks, Technical report, Duke University, Durham, NC, 2000.
[8] W. Brunette, D. Hoke, J. Jenks, Mule. Available from <http://www.cs.washington.edu/education/courses/cse476/02wi/projectwebs/476mule/>.
[9] D.G. Leeper, A long term view of short range wireless, IEEE Computer 34 (6) 39–44.
[10] W. Ye, J. Heidemann, D. Estrin, An energy-efficient MAC protocol for wireless sensor networks, in: IEEE INFOCOM 2002, New York, June 23–27, 2002.
[11] I. Chatzigiannakis, S. Nikoletseas, P. Spirakis, An efficient communication strategy for ad hoc mobile networks, in: Proceedings of the Twentieth Annual ACM Symposium on Principles of Distributed Computing, ACM Press, New York, 2001, pp. 320–322.
[12] Z.D. Chen, H.T. Kung, D. Vah, Ad hoc relay wireless networks over moving vehicles on highways, in: MobiHoC, 2001.
[13] Q. Li, D. Rus, Sending messages to mobile users in disconnected ad hoc wireless networks, in: Proceedings of the Sixth Annual International Conference on Mobile Computing and Networking, ACM Press, New York, 2000, pp. 44–55.
[14] M. Grossglauser, D. Tse, Mobility increases the capacity of ad hoc wireless networks, IEEE/ACM Trans. Network. 10 (4) (2002) 477–486.
[15] Manatee web page. Available from <http://distlab.dk/manatee/>.
[16] A. Beafour, M. Leopold, P. Bonnet, Smart tag based data dissemination, in: ACM Workshop on Wireless Sensor Networks and Applications, 2002.
[17] T. Camp, J. Boleng, V. Davies, A survey of mobility models for ad hoc network research, Wireless Communications & Mobile Computing 2 (5) (2002) 483–502, Special Issue on Mobile Ad Hoc Networking: Research, Trends and Applications.
[18] C. Bettstetter, Smooth is better than sharp: a random mobility model for simulation of wireless networks, in: Proceedings of the 4th ACM International Workshop on Modeling, Analysis and Simulation of Wireless and Mobile Systems, 2001, pp. 19–27.
[19] Z. Lei, C. Rose, Probability criterion based location tracking approach for mobility management of personal communications systems, in: Proceedings of the IEEE GLOBECOM, 1997, pp. 977–981.
[20] Z. Lei, C. Rose, Wireless subscriber mobility management using adaptive individual location areas for pcs systems, in: Proceedings of the IEEE International Conference on Communications (ICC'98), 1998, pp. 1390–1394.
[21] D.J. Aldous, J.A. Fill, Reversible markov chains and random walks on graphs, manuscript under preparation. Available from <http://stat-www.berkeley.edu/users/aldous/book.html>.
[22] R. Ellis, Torus hitting times from green's functions. Available from <http://www.math.ucsd. edu/rellis/comb/torus/torus.html>.
[23] F. Spitzer, Principles of Random Walk, Springer, Berlin, 2001.
[24] S.M. Ross, Introduction to Probability Models, Academic Press, New York, 2000.



Rahul C. Shah completed the B.Tech. (Hons) degree from the Indian Institute of Technology, Kharagpur in 1999 majoring in Electronics and Electrical Communication Engineering. He is currently pursuing his Ph.D. in Electrical Engineering at the University of California, Berkeley. His research interests are in energy-efficient protocol design for wireless sensor/ad hoc networks, design methodology for protocols and next generation cellular networks.

Sumit Roy received the B.Tech. degree from the Indian Institute of Technology (Kanpur) in 1983, and the M.S. and Ph.D. degrees from the University of California (Santa Barbara), all in Electrical Engineering in 1985 and 1988 respectively, as well as an M.A. in Statistics and Applied Probability in 1988. His previous academic appointments were at the Moore School of Electrical Engineering, University of Pennsylvania, and at the University of Texas, San Antonio. He is presently Professor of Electrical Engineering, University of Washington where his research interests center around analysis/design of communication systems/networks, with a topical emphasis on next generation mobile/wireless networks. He is currently on academic leave at Intel Wireless Technology Lab working on high speed UWB radios and next generation Wireless LANs. His activities for the IEEE Communications Society includes membership of several technical committees and TPC for conferences, and he serves as an Editor for the IEEE Transactions on Wireless Communications.

Sushant Jain is a Ph.D. candidate in the Department of Computer Science and Engineering at the University of Washington. His research interests are in design and analysis of networking systems. He is currently investigating energy efficiency issues in sensor networks. In the past he has worked in overlay networks, ad hoc networks and multicast. He received a M.S. in Computer Science from the University of Washington in 2001 and a B.Tech. degree in Computer Science from IIT Delhi in 1999.

Waylon Brunette is a Research Engineer in the Department of Computer Science and Engineering at the University of Washington. His research interests include mobile and ubiquitous computing, wireless sensor networks, and personal area networks. Currently, he is engaged in collaborative work with Intel Research Seattle to develop new uses for embedded devices and RFID technologies in ubiquitous computing. He received a BS in Computer Engineering from the University of Washington in 2002.