Mobile Networks and Applications 11, 327–339, 2006

DOI: 10.1007/s11036-006-5186-9

Exploiting Mobility for Energy Efficient Data Collection in Wireless Sensor Networks

SUSHANT JAIN*
Computer Science and Engineering, University of Washington, Box 352350, Seattle, WA 98195

RAHUL C. SHAH
University of California, 2108 Allston Way, Suite 200 Berkeley, CA 94704

WAYLON BRUNETTE and GAETANO BORRIELLO
Computer Science and Engineering, University of Washington, Box 352350, Seattle, WA 98195

SUMIT ROY
Department of Electrical Engineering, University of Washington, Box 352500, Seattle, WA 98195

Published online: 28 April 2006

Abstract. We analyze an architecture based on mobility to address the problem of energy efficient data collection in a sensor network. Our approach exploits mobile nodes present in the sensor field as forwarding agents. As a mobile node moves in close proximity to sensors, data is transferred to the mobile node for later depositing at the destination. We present an analytical model to understand the key performance metrics such as data transfer, latency to the destination, and power. Parameters for our model include: sensor buffer size, data generation rate, radio characteristics, and mobility patterns of mobile nodes. Through simulation we verify our model and show that our approach can provide substantial savings in energy as compared to the traditional ad-hoc network approach.

Keywords: simulations, stochastic processes, system design

1. Introduction

The problem of data collection in sparse sensor networks is encountered in many scenarios such as monitoring physical environments such as tracking animal migrations in remote-areas [1], weather conditions in national parks [2], habitat monitoring on remote islands [3], city traffic monitoring etc. The objective is to collect data from sensors and deliver it to an access point in the infrastructure. These systems are expected to run unattended for long periods of time (order of months). The principal constraint is the energy budget of the sensors which is limited due to their size and cost.

Current approaches involve forming an ad-hoc network among the sensor nodes to send data. However, this faces the following energy related issues. Firstly, in a sparse network, the energy required for transmitting data over one hop is quite large. This is because sensors may be far from each other and the transmission power required increases as the fourth power of distance. Secondly, in an ad-hoc network sensors have to not only send their data, but also forward data for other sensors. Thirdly, the network has routing hotspots near the access points. Sensors that are near the access points have to forward many more packets and drain their battery much more quickly. In this paper, we argue and analyze an alternative model for energy efficient data collection in sparse wireless sensor networks.

The key idea in our model is to exploit mobile entities present in an application scenario. We call these entities MULEs (Mobile Ubiquitous LAN Extensions) because they "carry" data from sensor to access point. For example, in a city traffic monitoring application vehicles can act as MULEs; in a habitat monitoring scenario, the role can be served by animals; in a national park monitoring scenario, people can be MULEs. MULEs are assumed to be capable of short-range wireless communication and can exchange data as they pass by sensors and access points as a result of their motion. Thus MULEs pick up data from sensors, buffer it and later on drop off the data at an access-point.

In the MULE architecture sensors transmit data only over a short range that requires less transmission power. However, latency is increased because a sensor has to wait for a MULE before its data can be delivered. Nevertheless, for many sensor network applications in which data is collected for future scientific analysis such high latency is acceptable.

*Corresponding author.

The idea of MULE architecture was introduced in our previous workshop paper along with a very simple analytical model [4]. However, many important aspects related to modeling and comparison with ad-hoc networks were left open and are the focus of this paper. The main contributions of this paper are:

1. An analytical model based upon queueing theory is presented to understand the relationship between performance metrics and system parameters. Performance is characterized along three dimensions: data transfer rate, latency, and energy requirements at the sensors. Our model incorporates system parameters such as sensor data generation rate, buffer size, sensor duty cycle, radio characteristics such as range and capacity, MULE velocity, MULE mobility model, etc. Our previous paper do not address sensor duty cycle, radio range, capacity and is limited to only one mobility model (random-walk). In particular, modeling radio characteristics increases the complexity of the problem significantly. However, by making suitable assumptions we are able to model radio range and capacity and further obtain closed form results.
2. Detailed simulations are performed to validate above analytical model and to gain finer understanding. For example, our results indicate that initially increasing radio bandwidth affects performance dramatically till a certain point beyond which increasing it further does not have much effect. Another interesting finding is that the random-waypoint mobility model can be modeled quite accurately using a Poisson arrival model of MULEs. Further, somewhat surprisingly, the performance of these two was similar to performance of MULEs with a fixed (or deterministic) mobility pattern. This indicates that controlling MULE motion precisely may not be beneficial.
3. We argue the benefits of the MULE architecture over ad-hoc networks both qualitatively and quantitatively using simulation. Our results indicate at least an order of magnitude energy savings. The improvement in the operational lifetime of the network was even more dramatic.
4. Issue of efficient discover of sensors is addressed by using a low duty cycle at the sensor and this is incorporated in the analysis. A novel discovery mechanism is discussed that permits significantly lower duty cycles while at the same time has very little impact on performance.

The paper is structured as follows. We next describe related work. Section 3 presents MULE architecture and its advantages over other approaches. Section 4 describes the analytical model and derives various results. Simulation results are presented in Section 6. We discuss some enhancements in Section 7 and conclude in Section 8.

2. Related work

Exploiting mobility for communication in ad-hoc networks has received much attention recently [5–9]. The work focuses on scenarios in which there is no immediate end-to-end path between two nodes that wish to communicate, usually because of limited radio range. If the nodes are mobile, end-to-end connectivity may be achieved by buffering data at the nodes and waiting to transfer until they are in range of access-points. The key difference is that our application context is focused on sensor networks unlike previous work where the focus was towards mobile ad-hoc networks. The severe resource constrained nature of sensors networks places different requirements on the optimization objectives. For example, our work tries to maximize sensor network lifetime by reducing the communication energy required at the sensors. In context of sensor networks, the ZebraNet [1] project collects data from sensors on zebras by exploiting the natural motion of the animals. Our architecture introduces MULE explicitly and encompasses Zebranet like scenarios. Also we focus on analytical modeling, energy efficient discovery and comparison with ad-hoc networks which were previously unaddressed.

3. MULE architecture

3.1. Overview

The MULE architecture provides connectivity by adding an intermediate layer of mobile nodes to the existing relationship between sensors and access-points used in typical sensor network designs [3] as shown in figure 1.

- *Lower tier—sensors:* Sensors provide data, communicate via a short-range radio, and have limited power and memory. The amount of work performed by sensors should be minimized because they have the most constrained resources among the three tiers.

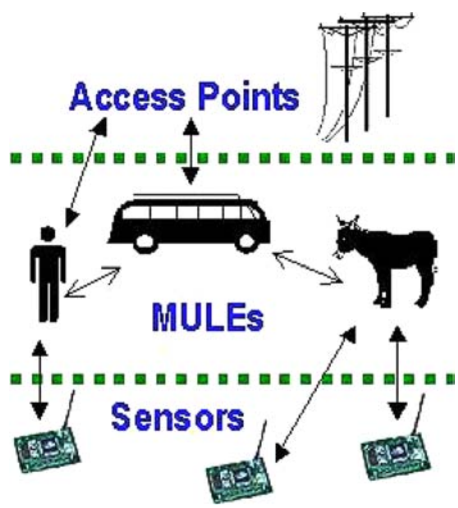


Figure 1. The three tiers of the MULE architecture.

– *Middle tier—MULEs:* MULEs are mobile entities with large storage capacities (relative to sensors), renewable power, and have the ability to communicate with sensors and access-points. A MULE has the responsibility to discover sensors and access-points and transferring data between them. In our basic model MULE(s) do not communicate with each other. In Section 7 we discuss the effect of MULE-to-MULE communication as an enhancement to our basic architecture.
– *Upper tier—access-points:* These are servers with Internet connectivity and enhanced power, storage and processing capabilities. For our purposes, these are the eventual destination of sensor data. They are used to offload the data collected by and stored in the MULEs.

Depending on the scenario, tiers in our architecture can be collapsed onto one device, increasing the applicability of our architecture. For example, sensors can be mobile as in the ZebraNet project [1] where sensors are attached to zebras, causing the sensor and the MULE tier to be mapped to the same device. Similarly, if MULE(s) have Internet connectivity they can act as an access-point, combining the MULE and access-point tiers.

3.2. MULE discovery

A sensor needs to discover a nearby MULE to be able to offload its data. In our architecture the prime responsibility of discovery is placed on the MULE, as our objective is to minimize the load on sensors. A MULE continuously sends out a discovery message to detect a nearby sensor. This requires a sensor to listen for discovery messages. Since listening consumes as much power as receiving [10], we need to reduce the duty cycle at the sensors. That leads to a tradeoff between minimizing the listen energy and maximizing the probability of rendezvous with a passing MULE. This affect of duty cycle on performance is analyzed in detail in Section 5.5. We also discuss some interesting techniques for reducing listening time in the enhancements Section 7.

3.3. Trade-offs

We now highlight the relative advantages and disadvantages of the MULE architecture.

3.3.1. Benefits

– *Energy Efficient:* Substantial energy is saved because sensors communicate over a short range. Moreover, there are no hotspots in the network as sensors do not forward data for other sensors.
– *Spatial Reuse:* The MULE architecture exploits spatial reuse of bandwidth by using short-range communication without losing long term connectivity and avoids radio communication complexities such as collisions.
– *No routing overhead:* In contrast to ad-hoc networks, the MULE architecture does not have any routing protocol overhead for sensors.
– *Robustness:* Performance degrades gracefully as MULEs fail. Any single MULE failure does not lead to a disconnected network. The primary effect of a MULE failure on the overall system is a slight increase in latency as there are now fewer MULEs to pick up data. In contrast, in an ad-hoc network failure of few critical nodes might lead to a disconnected network.
– *Scalable:* The MULE architecture is easily scalable as deployment of new sensors or MULEs requires no network reconfiguration.
– *Simplicity:* The data routing aspect of the MULE architecture is very simple and extremely lightweight for the sensors. This is important because sensors are the bottleneck of the system. The MULE architecture does not require any synchronization or location information; an assumption made by many approaches [10].

3.3.2. Limitations

– *Latency:* The MULE architecture has high latency and this limits its applicability to realtime applications (although this can be mitigated by collapsing the MULE and access-point tiers).
– *Best-effort delivery:* Data delivery in the basic architecture is best-effort; delivery is not guaranteed. The system requires sufficient mobility. For example, MULEs may not arrive at a sensor or after picking the data may not reach near an access-point to deliver it. Also, data may be lost because of radio-communication errors or MULEs crashing. To improve data delivery, higher-level protocols need to be incorporated in the MULE architecture. This is discussed further in the enhancements Section 7.

4. Analytical model

We begin with a discussion of the performance metrics and parameters involved in the MULE architecture followed by an analytical model based upon queuing theory.

4.1. Performance metrics

– *Data success ratio (DSR):* This measures the effectiveness of data delivery. It is defined as the ratio of the total amount of data transferred to the access-points to the total amount of data generated. This metric has been also been used in [1, 11]. Ideally, DSR will be one. Data may be lost because of errors in radio communication, failure of MULEs or buffer overflows.
– *Latency:* This is the average time taken by data to reach access-points from the time of its generation. Although, the MULE architecture is targetted for latency insensitive

applications some notion of latency can be important to meet application requirements.

– *Communication energy:* We consider both the average energy consumed per sensor as well as the worst case consumption which dictates the network lifetime. These are discussed in detail in evaluation Section 6.3.

4.2. Parameter space

The parameter space can be divided into the four following categories.

– *Sensor related:* The data generation rate (λ) defines the average amount of data that a sensor is generating. This directly affects the buffer requirements at the sensor. The sensor buffer size (*SB*) determines the maximum amount of data that can be stored on the sensor and can affect loss of data from buffer overflows. Another parameter is the duty cycle of sensor.

– *MULEs related:* The primary aspect is to determine when MULEs come into the communication range of a sensor. The MULE arrival within a sensor's range is modeled as a discrete event. The key parameter is the distribution of time between two MULE arrivals at a sensor. Determining this parameter is a complex problem that depends on factors such as MULE velocity, number of MULEs, sensor's radio range and a MULE's mobility pattern. For example, doubling number of MULEs or doubling velocity doubles the average MULE arrival rate. Our model abstracts out these complexities by assuming the knowledge of inter-arrival distribution. MULEs buffer size is another parameter, but for the purposes of this paper we assume that MULEs have sufficiently large buffers.

– *Access point related:* The important aspect here is the distribution and the number of access-points. This affects how frequently a MULE visits an access-point to deliver data. This is modeled by a parameter characterizing the distribution of the time interval between visits to access-point by a MULE.

– *Radio related:* The radio parameters affect the amount of data that can be transferred as a MULE passes by a sensor. We use a radial model for the radio, i.e. sensors and MULEs can communicate if they are within a distance r. The rate of data transfer is a fixed quantity B. Although simplistic, this provides a good approximation, particularly because the sensor to MULE communication will be over a short-range.

The discussion of the categories above highlights the fact that there are many knobs in the MULE architecture. Our approach is to identify a few basic parameters that are sufficient to characterize the performance metrics. These basic parameters are: (1) sensor data generation, (2) sensor buffer size (*SB*), (3) amount of data transferred between a MULE and a sensor, denoted by K (4) MULEs arrival at a sensor and (5) a MULE's visit to access-points.

The affect of other parameters can be understood by first studying how they change one or more of the basic parameters and subsequently studying how the performance is affected by the change in basic parameters. For example, the impact of increasing MULE velocity on performance can be examined in two steps. First, by examining the impact of increasing MULE velocity on the basic parameters. In this case, it increases the MULE arrival rate at the sensors/access-points and decreases K (see Section 5.4). Second, the analytical model is used to analyze the affect on performance due to the changes in these basic parameters. The effects of sensor duty cycle are modeled in a similar manner (see Section 5.5).

4.3. Model

The primary component of our model is a queue of generated data (but not delivered) at each sensor. In queuing theory terminology, generation of new data at a sensor corresponds to an arrival at the sensor's queue. The buffer size of the sensor defines the capacity of the queue. If the buffer is full then any new data is dropped. The queue is served whenever a MULE is in a sensor's range. For modeling purpose the arrival of a MULE in a sensor's range is considered as a discrete event. This event causes transfer of data from the sensor's queue to the MULE. The sensor then waits for the next MULE arrival event to transfer the data. Thus, the time between two MULE arrivals dictates when is the queue served.

The amount of data that can be transferred on a MULE arrival event is a random variable and depends on factors such as, the time the MULE is in the communication range of sensor. However, for analytical tractability, this is taken as a fixed quantity, denoted by K and is derived in Section 5.4.

The interaction between the MULEs and the access-points can be modeled on exactly the same principles. Because of space limitation we only focus on interaction between sensors and MULE, which is the primary bottleneck of the system. Interaction between the MULEs and access-points have been discussed in the technical report [12].

The above queueing model resembles the bulk service model in the queuing literature. The model is typically denoted as $G/G^K/\mathrm{I}/SB$ [13]. The two G's stands for the general input (data generation) and service (MULE arrival) distributions respectively. K is the service size, and SB is the maximum queue capacity. If less than K units of data are available at the sensor then that data is transferred and the MULE leaves without waiting for additional data.

The following list provides a summary of assumptions and key notational symbols.

– The MULEs arrival process at a sensor is a renewal process $\{S(t),\ t \geq 0\}$, where $S(t)$ is the total number of MULEs that have visited the sensor up untill time t. The renewal assumption means that the inter-arrival times (time between arrival of two MULEs) are independent and identically distributed (denoted by random variable X^s). Average MULE arrival rate is denoted by μ, and the variance of X^s is σ_{ms}.

- At a given time only one MULE interacts with a given sensor and vice-versa. Also, we assume that when a MULE visits a sensor no other sensor is near-by (and contending for service). This is reasonable because our networks are sparse. This assumption is verified using detailed simulations.
- Sensors are identical. Although not essential, we will assume that sensors are not mobile for ease of exposition.
- The data generation process at a sensor is a renewal process $\{U(t), t \geq 0\}$, where $U(t)$ is the total amount of data generated till time t. Average data generation rate is denoted by λ.
- The queueing discipline is FCFS. The data that is generated first is picked up first.
- MULEs have sufficiently large buffers.
- Without loss of generality, $SB \geq K$. If $SB < K$ then the maximum amount of data that is available at sensor buffer to transfer to MULE is SB. Therefore, $K = SB$ for such cases.
- Data transmission does not incur any loss. The only loss is due to sensor buffer overflow.
- The queueing system is stable and only the stationary (time independent) probabilities are considered. These are the probabilities as $t \to \infty$.

5. Results

5.1. Stability condition

Result 1. The system is stable (the queue reaches a unique stationary regime) iff

$$\frac{\lambda}{K\mu} \leq 1 \tag{1}$$

Proof. This follows directly from Theorem 3.1 in [14]. Intuitively, the equation says that the system is stable if the net service rate (product of K and the MULE arrival rate) is more than the data generation rate, else the sensor queues can grow arbitrarily large.

Our analysis assumes that $SB \geq K$ (see Assumptions 4.3). Incorporating this we get, $\frac{\lambda}{\min(SB,K)\mu} \leq 1$. The above equation can be used to derive the minimum value of K or SB (for a given λ, μ) required to reach a stable system.

5.2. Results for performance metrics

We now present results for different performance metrics. The rest of this section assumes the knowledge of the distribution of the queue length at the instance a MULE arrives at a sensor (denoted by the random variable Q). More specifically, P_j will denote the probability that the queue length Q is j (note that $P_j = 0$ for $j > SB$). Distribution of Q for specific scenarios is derived in next section.

The average of Q ($E[Q]$) is used as a measure of the average buffer occupancy of a sensor. By definition, $E[Q] = \sum_{j=0}^{SB} jP_j$

Result 2. Data Success Ratio (DSR) is given by:

$$DSR = \frac{\mu E[\min(K, Q)]}{\lambda} \tag{2}$$

$$= \frac{\mu\left(\sum_{j=0}^{K} jP_j + \sum_{j=K+1}^{SB} KP_j\right)}{\lambda} \tag{3}$$

Proof. Proof is given in Appendix A.

Later, we will see that $P'_j s$ depend only on the ratio of λ and μ,. From the above equation this will also be true for DSR. This tells us that the system performance (DSR and buffer occupancy) will not be affected if both parameters are scaled proportionately.

Result 3. Average queuing delay (W^q) is given by:

$$W^q = \frac{\mu^2\sigma_{ms} + 1}{2\mu} + \frac{E[B^{no}]}{\mu} \tag{4}$$

Proof. In general, a single MULE may not be able to transfer all the data in the sensor buffer. In such a case multiple MULEs may have to arrive before a data sample is served. Let, $E[B^{no}]$ denotes the average number of MULEs that arrive at the sensor while a data unit is in the queue excluding the MULE which serves the data unit itself. The expression for $E[B^{no}]$ is derived in Appendix B.

Recall that, μ is the average renewal rate and σ_{ms} is the variance of the MULE inter-arrival time distribution.

Consider a time t at which some data (call it d) is generated and accepted into the queue. The time spent by d in the queue is the time till the next MULE arrives after t, plus, the time till next $E[B^{no}]$ MULEs arrive. To compute the average time till the next MULE arrives, we will use the concept of Residual Life for renewal processes. Since the MULE arrival process is a renewal process, the average time till the next MULE arrival is by definition the average residual life of the MULE arrival process ($\{S(t)\}$). Therefore, by residual life theorem [15], the average residual life for $\{S(t)\}$ is $\frac{\mu^2\sigma_{ms}+1}{2\mu}$.

Since the average time between two arrivals of MULE is $\frac{1}{\mu}$, the average time for $E[B^{no}]$ MULEs to arrive is $\frac{E[B^{no}]}{\mu}$. Finally, W^q is the sum of the above two components.

If K is sufficiently large, a MULE can pick up all the data in the sensor queue. In this case $E[B^{no}]$ would be zero. Therefore, the average queuing delay is just the residual life of the MULE arrival process. The average queuing delay increases with σ_{ms}. Therefore, MULE arrival processes with lower variance will have lower queuing delay. Also, note that total latency from sensors to access-point also includes the time spent by data at the MULE before it is delivered to an access-point. As mentioned earlier, this can be dealt in the same manner as latency inside sensor and is discussed in technical report [12].

5.3. Specific scenarios

The previous section assumes that the distribution of Q is known. In general, this depends on the arrival pattern of the MULEs and the other system patterns. For the general case a closed form for the distribution of Q may be hard to obtain. In this section, we derive Q by making suitable assumptions. These distributions are later used in the evaluation section to compare results from the analytic model to the simulation results.

5.3.1. The MULE arrival distribution and the data generation process is Poisson

The Poisson assumption allows us to obtain closed form results and is reasonable under certain environments. For example, it is known by the Palm-Khintchine theorem (p. 156 [16]) that under mild conditions on the individual arriving entities (MULEs in our case), the aggregate arrival process (also called the superposition process) often looks approximately Poisson as $n \to \infty$. We directly apply the results from Section 4.5 of [13].

$$
\begin{aligned}
P_j &= (F_{SB-j} - F_{SB-j-1})/[F_{SB}], \quad j = 0, \ldots SB-1 \\
P_{SB} &= 1/[F_{SB}], \qquad \text{where} \\
F_i &= \sum_{s=0}^{[i/(K+1)]} (-1)^s \binom{i - sK}{s} (1-p)^s p^{sK-i} \quad i \geq 1
\end{aligned}
$$

Observe that P_j's depend only on the ratio of μ and λ. This indicates that the absolute value of μ and λ is not important. This would be useful in evaluating the effect of scaling parameters on performance (see Section 6) as one of the parameters can be fixed.

5.3.2. K is large ($K \geq SB$)

When $K \geq SB$, all the data is transferred when a MULE visits a sensor. Therefore, the amount of data in the sensor buffer (Q) is the minimum of: (1) the amount of data generated during the time between arrival of two MULEs, (2) the sensor buffer size. In most cases, by stationarity assumption, the amount of data generated in an interval depends only on the length of the interval. For example, for poisson or deterministic data generation process. Therefore, $Q = \min(U(X^s), SB)$. If SB is large, the equation can be further simplified to: $Q = U(X^s)$ The assumption SB large is valid when the load on the system is low which is particularly true with low data rates. In this case, the expected queue length is:

$$E[Q] = E[U[E[X^s]] = \frac{\lambda}{\mu} \tag{5}$$

5.4. Determining K

K is the average amount of data that can be transferred between a MULE and a sensor, as the MULE passes by a sensor. We assume that the sensor is stationary.

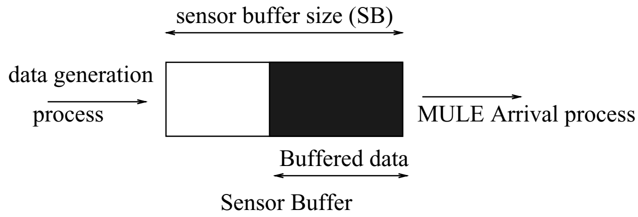


Figure 2. The queue model for MULE architecture.

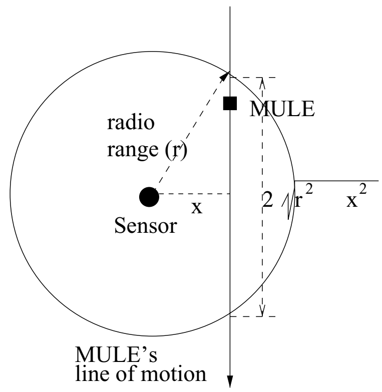


Figure 3. Amount of time a sensor is in contact with a MULE.

In our radio model, sensors and MULEs can communicate only if they are within a distance r. Therefore, the amount of data transferred is the radio data transfer rate (B) times the amount of time the MULE is in the radio range of sensor (called CT)[1] Thus, $K = CT \times B$.

The average contact time can be computed as follows. Let x be the perpendicular distance between the sensor and the MULE's line of motion as shown in figure 3.[2] Assume that x is uniformly distributed between 0 and r. The average distance that the MULE remains in contact with the sensor can now be computed as: $2\int_{x=0}^{r} \frac{\sqrt{r^2-x^2}}{r}dx$, which equals $\frac{\pi}{2}r$. If the MULE has a velocity v, we get $CT = \frac{\pi}{2}\frac{r}{v}$ Hence,

$$K = \left(\frac{\pi}{2}\frac{r}{v}\right) B$$

For example, consider a sensor-MULE interaction using a Berkeley mote. The mote has a radio range of 25 meter and data transfer rate of 40 Kb per second. If the MULE has a velocity of 10 m/s (10 m/s is approx 20 miles per hour), using above equation, we get $K = 150$ Kb.

[1] We are assuming that the discovery time is very small.

[2] In general an application may have additional constraints on x, such as for traffic monitoring application x is at-least few meters because of spatial constraints.

5.5. Impact of sensor duty cycle

We will assume that the sensor periodically listens for *DT* seconds every *BT* seconds. *DT* is the time for discovery and *BT* is the beacon interval. Duty-cycle (γ) by definition is the ratio, $\frac{DT}{BT}$. Also, we Compared to the 100% duty cycle case, performance will be affected because of two reasons:

1. A MULE may not be discovered at all because the sensor was asleep during the time the MULE was in communication range of sensor (figure 5(a). We model this by finding the probability of discovering a nearby MULE and use it to get the effective MULE arrival rate (called μ^*) . For example, if the probability of discovering a MULE is 0.25, then the effective arrival rate is one-quarter of the original rate.[3]

   The probability that a MULE is missed is the same as the probability that the MULE contact time interval (*CT*) does not oevrlap with the sensor's discovery interval when it was listening. Assuming that the MULE contact time can begin uniformly at any time with respect to a sensor's, the probability of discovering a MULE is $(CT - DT)/BT$. Therefore, μ^* is $\mu(CT - DT)/BT$. Note that, if $CT - DT \geq BT$ $\mu^* = \mu$.
2. The amount of data that can be transferred (*K*) in one contact may decrease if the MULE is discovered in the middle of the duration it is in the communication range of the sensor. We model this by finding an effective *K* (called K^*), the average amount of data transferred between the MULE and the sensor due to this late discovery.

   K^* depends on the time that is lost because of late discovery. If $CT - DT \geq BT$, the discovery starts in the first $BT/2$ seconds on average. Therefore, K^* is $K(1 - BT/2CT)$ in this case. On the other hand, when $CT - DT < BT$ the discovery starts on average in the first $(CT - DT)/2$ seconds. Therefore, K^* is $\frac{K}{2}(1 + DT/CT)$.

5.5.1. Example

As an example, suppose sensors have a duty cycle of 1/100. Consider a sensor-MULE interaction scenario, where the radio range is 25 m and the MULE velocity is 10 m/s. The contact time (*CT*) for these parameters is approximately 4 seconds. Discovery time is typically 10's of milliseconds, say 40 ms. For these parameters, $\mu*$ is the same as μ. The only affect is on *K*, which is halved. This shows that the sensors can operate at low duty cycles without substantially affecting performance.

[3] Here we are assuming that the MULE arrival process is Poisson and the results hold because random sampling of Poisson processes results in another Poisson process [15]. For general distributions, this provides a convenient approximation.

6. Evaluation

This section serves three purposes. One, to understand the affect of scafing system parameters on performance metrics. Second, to verify the analytical model presented earlier using detailed simulations. Third, to compare energy consumption between the MULE architecture and ad-hoc networks.

6.1. Simulation setup

A custom simulator was written to model the MULE architecture. The underlying topology was a two-dimensional grid. Sensors and access-points were fixed and randomly placed on the topology. MULEs were described by an initial position and a mobility model which guided there movement through the topology. Considered mobility models were: Random waypoint, random walk, deterministic arrivals (fixed route and velocity), and poisson arrivals. Data generation at sensor was defined by a generic distribution. We considered both poisson distribution (as assumed by the analytical model) as well as constant rate generation. In the constant data generation rate interarrival time between two events is fixed. Both sensors and MULEs had fixed buffer size (unlike analytical model where MULEs had infinite buffer). Details of sensor duty cycle and discovery were also modeled. Sensor's listen only a small fraction of time and randomization was used in sensor's sleep schedule.

A disc model was used for radio propagation, i.e whenever sensors and MULE were in each other's range *r* they can communicate at data rate *B*. Since our target networks are sparse and radio range small (25 m default) this is a reasonable model. Further, a link error rate of one percent was introduced to verify the robustness of our results to lossy links. The radio propagation model along with mobility model for MULEs leads to a variable *K*, unlike our assumption in analysis. Communication between MULEs and access-points were modeled in the same manner.

6.1.1. Parameter settings

As discussed earlier, the key parameters affecting performance are: λ (data generation rate), μ (MULE arrival rate), *SB* (sensor buffer) and *K* (data transferred in one interaction). Only, the above parameters are varied. *K* is varied by varying radio bandwidth *B* (default value 25 KB) (see Section 5.4) and μ (MULE arrival rate) is varied by increasing number of MULEs.

The topology used for sensor placement and MULE motion was a 2 km * 2 km grid. 100 sensors were placed randomly on it. There was one access-point and was placed at a corner of the topology.[4] MULE buffer was fixed at 10 MB. Default mobility model used for MULEs was random-waypoint. The MULE velocity was set to be at 10 m/s. The default radio-range was 25 m. Time to discover a sensor by a

[4] We tried few variations in the topology dimensions and qualitatively similar results were obtained.

MULE was taken to be 40 ms. Sensor duty cycle was 1/200. λ is fixed at 90 KB/Hour and constant rate data generation is assumed for simulations.[5] The results presented were averaged over 100 random simulations.

6.2. *Performance metrics*

We first study the effect of increasing μ and *SB*, assuming sufficiently large *K* ($K \geq SB$). Subsequently, the effect of *K* is considered. Both simulation and analytical results (using poisson arrivals) are presented. Finally, effect of different mobility models is considered.

6.2.1. *Scaling μ, and SB*

Figure 4 shows the effect of increasing μ, on the performance metrics. The three different lines on the plots corresponds to three different sensor buffer sizes 1 MB, 100 KB and 50 KB.

The plots verify that inspite of many simplistic assumptions our analysis matches closely to detailed simulations. Results produced by our analytical model were with-in 5 percentile of simulation results.

Figure 4(a) shows the affect of increasing μ, on *average sensor buffer occupancy.*

As expected, with increasing μ the average buffer occupancy decreases. This is because when MULEs come more frequently there is less amount of data generated between two arrivals. Further, interestingly, *SB* does not have much effect on buffer occupancy except when the MULE arrival rate is small which causes excessive load on the system.

Figure 4(b) shows the effect of increasing μ, on the *data success ratio* (*DSR*).

With increasing μ, the DSR increases sharply eventually reaching one. This is because when μ is large, the buffer occupancy decreases and therefore less data is dropped. The arrow shows the minimum value of μ, required for stability of the queuing system found using (equation 1 and taking into account the effect of duty cycle. DSR is very low (around 0.6) at that point. Therefore, μ, should be much larger (5 times for our experiments) than the minimum μ required for stability. The DSR is also higher, when *SB* is larger. This is expected because when *SB* is large, less data is dropped. In general, one can increase DSR by either increasing μ or *SB*.

Figure 4(a) shows the effect of increasing μ, on *latency*.

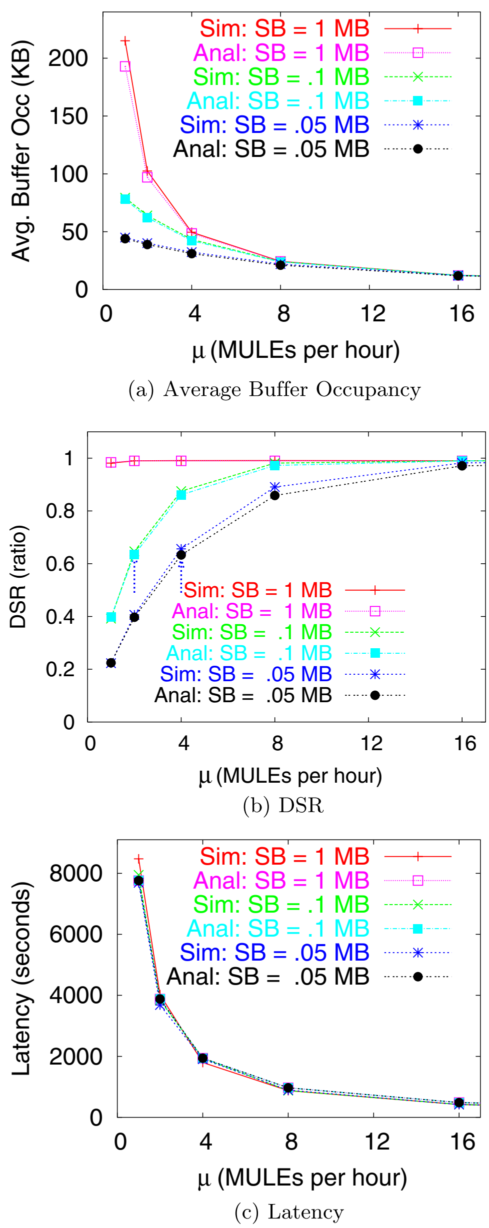
Since *K* is large, the queuing delay is simply the residual life of the MULE arrival process, which decreases as μ, is increased. Additionally, *SB* has no impact on latency.

[5] As mentioned during analysis only the ratio $\frac{\lambda}{\mu}$ is important.

6.2.2. *Effect of K*

Figure 5(a) and (b) shows the effect of increasing *K* on the average buffer occupancy and the fatency respectivefy (note that the y-axis is fogscafe). We chose μ as 2 per hour and relatively large *SB* of 1 MB. Since *SB* is large, the DSR is always close to one and is not shown.

When *K* is small, both buffer occupancy and latency is large. This is because a sensor cannot transfer all the data in the queue to a MULE during a single contact. This increases the average buffer occupancy. Latency is also increased because a data unit has to wait for multiple MULEs to arrive before it can be served. As *K* is increased, there is a sharp decrease in both the buffer occupancy and the

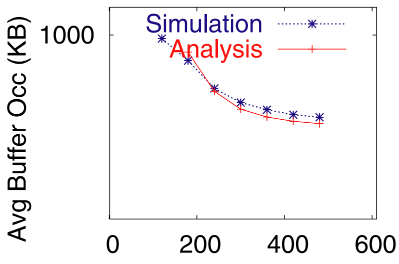


(a) Average Buffer Occupancy

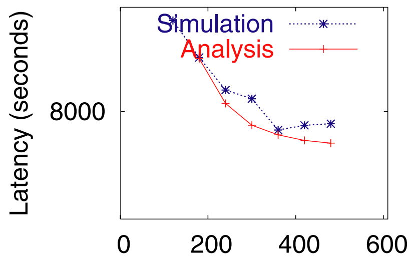
(b) DSR

(c) Latency

Figure 4. Effect of scaling μ, on performance metrics, (a) shows average buffer occupancy, (b) shows DSR and (c) shows latency.



(a) Buffer Occupancy



(b) Latency

Figure 5. Effect of scaling K. (a) buffer occupancy (b) latency.

latency initially. However, increasing K beyond a certain limit does not effect performance. This follows by observing the flat region of the plots. Intuitively, this is because K only needs to be large enough so as to absorb the occasional burst in the sensor buffer. For our experiments, we found that $K^* = 3 \times \frac{\lambda}{\mu^*}$ was sufficient to be in the flat region.[6]

6.2.3. Effect of mobility model

Four mobility models are considered: (1) Random waypoint (2) Random Walk (new direction is chosen on reaching a street intersection) (3) Deterministic (MULEs arrive at fixed interval) (4) Poisson arrival. The data generation rate was 90 KB/S. K and SB were fixed at 100 KB and only μ, was varied.

Figure 6 shows the DSR and the latency for different mobility models as μ scales. In all cases as μ increases, the DSR increases and the latency decreases. The performance is best when the MULE arrival is deterministic and worst under the manhattan model. The performance of random-waypoint model closely matches that of poisson model.

To understand this behavior, we considered the coefficient of variation (CVR) for different mobility models as shown in Table 1. CVR gives an idea of the burstiness of MULEs arrival. Large CVR means that the MULE arrival pattern is more bursty and vice-versa. Now, the performance would be better when the MULEs arrive at regular interval than in bursts (assuming same μ). This is because when the MULE arrival pattern is bursty, relatively longer periods exist when no MULE arrives. This can cause the sensor buffer overflow and reduce the DSR. This also affects latency because latency increases with the variance as discussed in the latency analysis (Results 3).

6.2.4. Summary

Table 2 summarizes the relationship between the different parameters and the metrics. We also find that:

– The performance results determined using analysis were close to (with in 5%) results of detailed simulation.
– DSR is less than 60% if the parameters are chosen such that the stability condition is just met. DSR can be made close to one by increasing μ, or SB. When K is large, choosing SB and μ, such that $\mu * B > 5\lambda$ resulted in a DSR greater than 95%.
– When K is small, the sensor buffer occupancy and latency is quite large. However, the performance improves sharply by increasing K initially and eventually saturates when $K^* > 3 \times \frac{\lambda}{\mu^*}$.
– Mobility models which have high variance perform worse than more deterministic models. The performance of poisson arrivals and random waypoint were almost same and similar to deterministic arrivals. This indicates that MULEs with fixed mobility pattern are not much beneficial than random-waypoint kind of motion.

Table 1
Coefficient of variation (CVR) for MULE inter-arrival distribution for different mobility models.

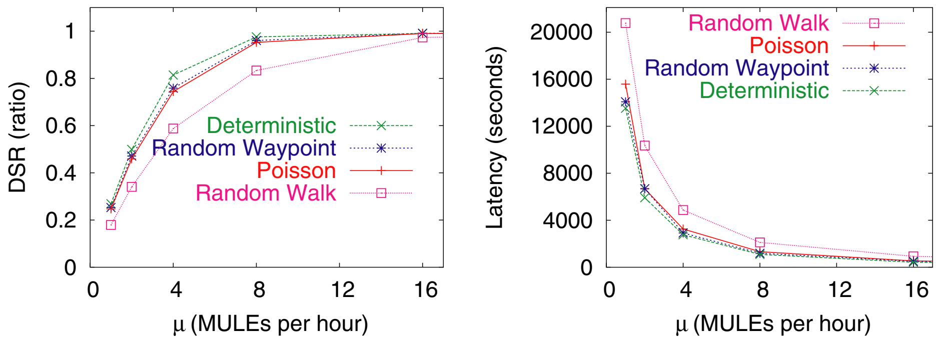
Mobility model	CVR
Poisson	1.0
Deterministic	0.0
Waypoint	0.75
Manhattan	2.1

Table 2
Effect of parameters on performance.

	Performance metrics		
Parameters	Buffer Occ	DSR	Latency
$\mu\uparrow$	$\downarrow$	$\uparrow$	$\downarrow$
$SB\uparrow$	–	$\uparrow$	–
$K\uparrow$	$\downarrow$	$\uparrow$	$\downarrow$
$\lambda\uparrow$	$\uparrow$	$\downarrow$	$\uparrow$

Note. ↑ indicates an increase in the quantity. ↓ indicates a decrease and — indicates no effect.

[6] K^* and μ^* are the effective quantitites after taking into account sensor's duty cycle (Section 5.5).



(a) DSR (b) Latency

Figure 6. Effect of different mobility models.

6.3. MULE vs Ad-hoc network

This section compares the energy consumption for sending data in the MULE architecture to an ad-hoc network. The MULE model and the adhoc network model presents two different paradigms of collecting data from a sensor network. The MULE model uses explicit mobile entities which are not used in the adhoc network model. A head-to-head comparison should therefore, be carefully interpretted. The main goal here is to understand the potential savings in energy in the MULE model. We believe that this would allow us to better understand the overall tradeoffs between the two paradigms.

The following metrics are used:

Average energy ratio: This is the ratio of the average energy consumed at a sensor in the ad-hoc network to the energy consumed in the MULE architecture.

Hotspot ratio: This is the ratio of *hotspot usage* in the ad-hoc network to the *hotspot usage* in the MULE architecture. *Hotspot usage* is the maximum energy consumed by any sensor. This gauges the network lifetime.

The model used for communication energy [17] is: $p_t = (\alpha_{11} + \alpha_2(d)^l)$ and $p_r = (\alpha_{12})$. p_t is the energy dissipated to transmit 1 bit of data to a node at a distance d. p_r is the energy dissipated to receive one bit of data. l is the path loss index and α's are positive constants. Here, $\alpha_{11} = 45$ nJ, $\alpha_{12} = 135$ nJ, $\alpha_2 = 10$ pJ/m^2 $(1 = 2)$ or .0001 pJ/m^4$(l = 4)$, if $d < 87$ m, $l = 2$, else $l = 4$.

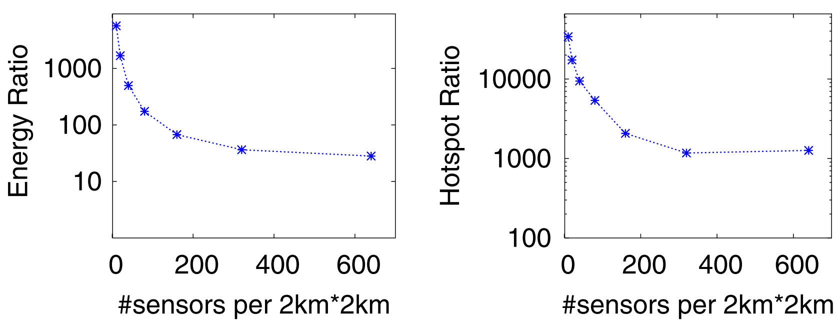
Energy Requirements in the MULE architecture: In the MULE architecture, a sensor communicates data only to a MULE within range r. Therefore transmit energy per bit (per sensor) is simply $\alpha_{11} + \alpha_2(r)^l$. ($r = 25$ m).

Energy Requirements in an ad-hoc network: This depends on the sensor network topology and the routing protocol. A sensor communicates data to a nearby sensor towards an access-point and the forwarding continues until the data reaches the access-point. We route the data through the minimum energy path [18]. Energy requirements for route maintenance are ignored, therefore, the energy computed here is only a lower bound on the overall energy requirements. Since energy requirement here depends on network density, number of sensors were varied.

6.3.1. Results

We chose μ as 16 per hour, *SB* and *K* as 100 KB. Other parameters were same as discussed in the simulation setup. Figure 7(a) shows the *Average Energy Ratio* as a function of the sensor density.

When the sensor density is low, the MULE architecture has over a factor of 100 less average energy consumption. This is because with few sensors the average distance between two sensors is large and the communication energy increases as the fourth power of distance. The benefits decrease as the



(a) Avg. Energy Ratio (b) Hotspot Ratio

Figure 7. Energy comparison of MULE vs Ad-hoc network approach as a function of sensor density.

sensor density is increased and eventually saturate with the average energy ratio around ten. This highlights that even for high sensor density the MULE architecture is more efficient. This is because in the ad-hoc network the data traverses multiple hops.

Figure 7(b) shows the *Hotspot Ratio* as a function of the sensor density. Same trend as in the Average Energy Ratio is observed indicating that the life-time in the MULE architecture will be much longer than the ad-hoc network. Additionally, the Hotspot Ratio is over an order of magnitude higher than the Average Energy Ratio. This is because the sensors near the access-points have to forward much more data than others whereas in the MULE architecture all sensors have the same energy consumption.

These results are not surprising and are somewhat biased because in the MULE architecture there is an additional energy consumption at the MULEs. However, MULEs are assumed to be entities with renewable energy whereas sensors are energy constrained and the primarily bottleneck of the system. Also, latency in MULE network was much much more (few minutes or more) than latency in ad-hoc network (few seconds at worst). Another metric of comparison is throughput between the two approaches. For our settings (90 Kb/Hour) both approaches were able to deliver the data. However, we suspect that because the capacity of an ad-hoc network decreases with increasing number of sensors [19] the MULE architecture can potentially provide more throughput (assuming sufficient number of MULEs) because of its better spatial reuse. This would confirm the results presented in [5] and we are currently exploring this further.

7. Enhancements

7.1. Reducing sensor duty cycle

Reducing sensor duty cycle saves energy but also affects the system performance as the sensors may not discover a nearby MULE. However, the probability of discovering a MULE can be improved by increasing the contact time (Section 5.5), thereby allowing a reduction in the duty cycle without affecting performance.

The basic idea involves MULEs using longer range radios to transmit discovery messages. Sensors then have the opportunity to hear the message for a longer period of time, thereby increasing the effective contact time (Section 5.4). Once the sensor hears the discovery message, it can keep the radio-on and wait for the MULE to come within the communication range of the sensor radio.

Application specific knowledge can also be used to reduce duty cycle. For example, if a sensor is aware of a MULE's arrival schedule then it can simply start listening at an appropriate time.

7.2. End to end reliability

A simple method to achieve reliability is to incorporate acknowledgements (acks). The main challenge is to determine when the sensors should retransmit their data. There is a trade-off, as retransmitting data too early may cause unnecessary transmissions that increase energy consumption; whereas, delaying retransmission may lead to buffer overflow and increased latency. The problem is particularly acute because of large and highly variable latencies.

7.3. Improving data reachability

The basic architecture assumes that a MULE eventually reaches an access-point and at-least one MULE reaches a sensor. This limitations can be overcome by using a more general framework in which MULEs and sensors can communicate among themselves. For example, MULE to MULE communication can be used to address scenarios in which a MULE may not reach an access-point. Similarly, if no MULE reaches a sensor, the sensor can send its data to other sensors (using an ad-hoc network) which might be able to forward its data eventually.

8. Conclusion

In this paper, we argued for exploiting mobility for energy efficient non real time data collection in sparse sensor networks as an alternative to forming an ad-hoc network. To this end, we presented and analyzed the MULE architecture, a three-tiered design. The key idea is to exploit the presence of mobile nodes in the environment by using them as forwarding agents. This approach extends the lifetime of the network by minimizing the communication responsibility of the resource-constrained sensors.

An analytical model based on queuing theory was presented. Our model incorporates many detailed aspects such as different MULE mobility models, radio characteristics etc. By making appropriate assumptions, analytically closed form results were also dervied which were validated through detailed simulations. Our results provide a usefull base to understand performance cost trade-offs such as buffer requirements at sensors, radio bandwidth requirements, sensitivity to mobility model etc. We also compared the MULE network with ad-hoc network. We found that energy savings of up to two-orders of magnitude (and even larger increases in network lifetime) can be achieved with MULEs as compared to the traditional ad-hoc network approach. However, the MULE architecture is limited to non real time applications which have mobility. Thus, the MULE architecture is not always the method of choice, but for certain applications it may be the most effective option.

This work is only a first step in understanding the feasibility of using mobility in sensor networks. It is clear that much more work remains to be done to fully understand the cost-effectiveness of this approach. We plan to investigate some of the enhancements discussed earlier, such as reliability and using MULE-to-MULE communication. Issues surrounding naming, network layer, and end-to-end connectivity semantics also needs to be addressed. Here we hope to leverage work from a recently proposed network architecture called the Delay Tolerant Network [20].

Appendix A: Proof of result for DSR result 2

Result 2: Data Success Ratio (DSR) is given by:

$$DSR = \frac{\mu E[\min(K, Q)]}{\lambda}$$

$$= \frac{\mu\left(\sum_{j=0}^{K} jP_j + \sum_{j=K+1}^{SB} KP_j\right)}{\lambda}$$

Proof. DSR is the ratio of data delivered to the access-points to the amount of data generated in time t as $t \to \infty$. By our assumptions once a MULE picks up the data it is delivered to the access-point. Therefore, DSR is the ratio of the data picked up by the MULEs in time t to the total data generated in time t.

$$DSR = \lim_{t\to\infty} \frac{P(t)}{U(t)} = \lim_{t\to\infty} \frac{P(t)}{t}\left(\lim_{t\to\infty}\frac{U(t)}{t}\right)^{-1}$$

Here $U(t)$ is the total amount of data generated at the sensor and $P(t)$ is the total amount of data picked up by the MULEs. Also recall, $S(t)$ is the number of arrivals of MULEs in time t. Now,

$$\lim_{t\to\infty}\frac{P(t)}{t} = \lim_{t\to\infty}\frac{P(t)}{S(t)}\lim_{t\to\infty}\frac{S(t)}{t}$$

By definition, $\lim_{t\to\infty}\frac{S(t)}{t} = \mu$, and$\lim_{t\to\infty}\frac{U(t)}{t} = \lambda$. The term $\frac{P(t)}{S(t)}$ represents the average amount of data transferred when a MULE visits the sensor. Let L be the amount of data picked up by a MULE at the sensor. Then,$\lim_{t\to\infty}\frac{P(t)}{S(t)} = E[L]$ Since only a maximum of K data units can be transferred, $L = \min(K, Q)$. Now using the fact that P_j is the probability Q equals j,

$$E[L] = \sum_{j=0}^{K} jP_j + \sum_{j=K+1}^{SB} KP_j$$

Putting everything together, we get the result.

Appendix B: Expression for $E[B^{no}]$

$$E[B^{no}] = \sum_{i=0}^{\lceil\frac{SB}{K}\rceil-1} i \sum_{j=(iK)}^{iK+K-1} P_j^e$$

Proof. $E[B^{no}]$ is the average number of MULEs that arrive at a sensor while a data unit is in the queue. This depends on the distribution of queue length at the instant a new data is accepted in the queue. To compute this we define P_j^e which is the probability that the queue length is j at the instant a new packet is accepted in the queue (excluding the new data unit itself).P_j^e can be related to P_j by (Theorem 4.1 of [14])

$$P_j^e = \sum_{i=j+1}^{\min(j+K,SB)} \frac{P_i}{E(L)} \quad 0 \le j < SB$$
$$P_j^e = 0 \quad j \ge SB$$

The B^{no} of a new data unit is i iff the queue length (exluding the packet itself) is between iK and $iK + K - 1$. This is because a single MULE arrival removes K data units from the queue.This gives,

$$E[B^{no}] = \sum_{i=0}^{\lceil\frac{SB}{K}\rceil-1} i \sum_{j=(iK)}^{iK+K-1} P_j^e$$

References

[1] P. Juang, H. Oki, Y. Wang, M. Martonosi, D. Rubenstein and L. Peh, Energy-efficient computing for wildlife tracking: Design tradeoffs and early experiences with zebranet, in *ASPLOS-X*, (October 2002).

[2] A. Beaufour, M. Leopold and P. Bonnet, Smart-tag based data dissemination, in *WSNA* (Atlanta, GA, Sept. 2002).

[3] A. Mainwaring, J. Polastre, R. Szewczyk, D. Culler and J. Anderson, Wireless sensor networks for habitat monitoring, in *WSNA* (Atlanta, GA, Sept. 2002).

[4] R.C. Shah, S. Roy, S. Jain and W. Brunette, Data mules: Modeling a three-tier architecture for sparse sensor networks, in *SNPA*, (Anchorage, Alaska, May 2003).

[5] M. Grossglauser and D. Tse, Mobility increases the capacity of ad-hoc wireless networks, in *Trans, on networking*, vol. 10(4) (Aug. 2002).

[6] I. Chatzigiannakis, S. Nikoletseas and P. Spirakis, An efficient communication strategy for ad-hoc mobile networks, in *Principles of Distributed Computing* (ACM Press, 2001) pp. 320–322.

[7] A. Vahdat and D. Becker, Epidemic routing for partially-connected ad hoc networks, in *TR* (Duke university, 2000).

[8] Q. Li and D. Rus, Sending messages to mobile users in disconnected ad-hoc wireless networks, in *Mobicom* (ACM Press, 2000) pp. 44–55.

[9] R.H. Frenkiel, B.R. Badrinath, J. Borras and R. Yates, Infestations challenge: Balancing cost and ubiquity in delivering wireless data, in IEEE Personal Communications (April 2000).

[10] B. Chen, K. Jamieson, H. Balakrishnan and R. Morris, Span: An energy-efficient coordination algorithm for topology maintenance in ad hoc wireless networks, in *Mobicom* (Rome, Italy, July 2001) pp. 85–96.

[11] F. Ye, H. Luo, J. Cheng, S. Lu and L. Zhang, A two-tier data dissemination model for large-scale wireless sensor networks, in *Mobicom* (ACM Press, 2002) pp. 148–159.

[12] S. Jain, R. Shah, W. Brunnette, G. Borriello and S. Roy, Exploiting mobility for energy efficient data collection in sensor networks, Tech. Rep. UW-CSE-03-06-04, UW-CSE(2003).

[13] M.L. Chaudhry and J.G.C. Templeton, *A First Course in Bulk Queues* (John Wiley and Sons, 1983).

[14] G. Hebuterne and C. Rosenberg, Arrival and departures in the general bulk-service system, in *Naval Research Logistics* ed-Vol45 (John Wiley, 1998).

[15] S.M. Ross, *Introduction to Probability Models* (Academic Press, 2001).

[16] D.P. Heyman and M.J. Sobel, *Stochastic Models in Operations Research*, vol. 1 (McGraw Hill, NY, 1982).

[17] W. Heinzelman, Application-specific protocol architectures for wireless networks, Phd thesis, mit (2000).
[18] S. Singh, M. Woo and C.S. Raghavendra, Power-aware routing in mobile ad hoc networks, in *Mobicom* (ACM Press, 1998) pp. 181–190.
[19] P. Gupta and P.R. Kumar, The capacity of wireless networks, in IEEE Transactions on Information Theory, vol. 46(2) (March 2000).
[20] K. Fall, A Delay Tolerant Network Architecture for Challenged Internet, in *Sigcomm* (Kalrushe, Germany, July 2003).

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 routing algorithms for networking systems. He received a MS in Computer Science from the University of Washington in 2001 and a B.Tech degree in Computer Science from IIT Delhi in 1999.
E-mail: sushjain@cs.washington.edu

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.
E-mail: rcshah@eecs.berkeley.edu

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.
E-mail: wrb@cs.washington.edu

Gaetano Borriello is a Professor in the Department of Computer Science and Engineering at the University of Washington. His research interests are in embedded and ubiquitous computing, principally new hardware devices that integrate seamlessly into the user's environment with particular focus on location and identification systems. His principal projects are in creating manageable RFID systems that are sensitive to user privacy concerns and in context-awareness through sensors distributed in the environment as well as carried by users.
E-mail: gaetano@cs.washington.edu

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 Prof, of Electrical Engineering, Univ. 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.
E-mail: roy@ee.washington.edu