Delay Tolerant Routing Protocols for Energy-Neutral Animal Tracking

Mahzad Kaviani*†, Branislav Kusy†, Raja Jurdak†, Neil Bergmann†, Kun Zhao†, Vicky Liu*
*Queensland University of Technology, †CSIRO, ‡The University of Queensland, Australia
Mahzad.kaviani@hdr.qut.edu.au, Brano.Kusy@csiro.au, Raja.Jurdak@csiro.au, bergmann@itee.uq.edu.au, kun.zhao@csiro.au, v.liu@qut.edu.au

Abstract
This paper investigates communication protocols for relaying sensor data from animal tracking applications back to base stations. While Delay Tolerant Networks (DTNs) are well suited to such challenging environments, most existing protocols do not consider the available energy that is particularly important when tracking devices can harvest energy. This limits both the network lifetime and delivery probability in energy-constrained applications to the point when routing performance becomes worse than using no routing at all. Our work shows that substantial improvement in data yields can be achieved through simple yet efficient energy-aware strategies. Conceptually, there is need for balancing the energy spent on sensing, data mulling, and delivery of direct packets to destination. We use empirical traces collected in a flying fox (fruit bat) tracking project and show that simple threshold-based energy-aware strategies yield up to 20% higher delivery rates. Furthermore, these results generalize well for a wide range of operating conditions.

Categories and Subject Descriptors
C.2 [Computer-Communication Networks];
C.2.1 [Network Architecture and Design]: Distributed networks, Wireless communication, Network communication;
C.2.2 [Network protocols]: Routing protocols

Keywords
Delay Tolerant Network (DTN); Energy-aware routing protocol; Forwarding algorithm

1. Introduction
Wireless Sensor Networks (WSNs) are able to sense, store, and communicate rich information about the environment. When WSN nodes are attached to animals, they can track the location and activity of those animals to assist with understanding of their behavior. However, for small animals, these sensors must be lightweight and necessarily have limited energy available. Therefore, new approaches are needed to design WSN system architectures and protocols to maximize the retrieved information for such systems.

Tracking position and activity of wildlife species, farm animals, and people is an important application domain of WSNs. The literature reports tracking of numerous animal species and mobile assets, including zebras [1], badgers [2], whooping cranes [3], flying foxes [4], cattle [5], or rental bicycle networks [6]. Such systems collect information about location and activity of the tracked objects on a periodic basis. The underlying protocols deliver the collected data to a base station (sink), which has Internet connectivity, typically through Wi-Fi or 3G [7]. Because low energy animal tracking nodes typically have only local area communications, data cannot be continuously streamed to a base-station. Instead, data is stored locally until a base-station is encountered.

To further improve the likelihood of data delivery, nodes may pass data to other nodes on other animals to increase the chance of data delivery. Such a store-and-forward architecture is known as Delay Tolerant Network (DTN). DTNs have been studied in the wider field of wireless mobile ad hoc networks [8] and were applied in large-scale WSNs [4], satellite [9] and underwater acoustic networks [10]. Previous work has focused on improving the probability of delivering packets to the destination and considered challenges, such as dynamically changing network routes, existence of transient low-quality network links, and large non-deterministic delays between data sources and sinks.

However, previous work has assumed unconstrained energy resources and the mainstream DTN protocols do not adjust their behavior based on the available energy. This paper shows that energy considerations are critically important in energy-constrained deployments and can substantially improve data delivery rates. We assume that sensor nodes periodically visit data sinks between energy replenishment events (such as solar harvesting) and propose two simple yet efficient energy-aware strategies that extend existing DTN protocols. The main idea is to balance the energy spent for sensing and network communication to ensure energy-neutral operation of the nodes between harvesting events. We define an energy conservation state in which the nodes stop sensing and participating in routing to conserve energy for transmitting existing packets directly to the sink. We propose two simple strategies that build on this conservation state: (1) the Threshold strategy enters the conservation state if the total energy falls below a fixed threshold provided by the user; and (2) the Remaining Required Energy (RRE) strategy determines the optimal threshold based on an estimate of the energy that the node needs to transmit existing data to a sink. The efficiency of these simple strategies clearly motivates more research into sophisticated energy-aware strategies for DTN protocols.

We evaluate both Threshold and RRE strategies through empirically motivated simulations using the ONE simulator [11]. We use animal motion traces from a flying foxes tracking application [4] and extend the ONE simulator to simulate animals in their typical daily patterns, such as food foraging within a feeding area or transitioning between different feeding areas. Our results show that the energy-aware versions of Spray&Wait [12], Epidemic [13] and Prophet [14] routing protocols can outperform their original counterparts by 10% to 20% in packet delivery rate. Furthermore, the RRE strategy matches performance of the Threshold strategy for the optimal threshold across all simulation rounds. We ensure repeatability of our results by averaging results over several motion traces and show that the results generalize well to different daily energy budgets.
2. Background and Related Work

2.1. DTN Classification of Routing Protocols

DTN routing protocols can be classified in relation to the type of information collected by nodes to make the routing decisions. Shen et al. [8] characterize two different approaches to replication and knowledge building for finding the destination: flooding and forwarding routing protocols.

In the flooding strategy, the node spreads a number of copies of each message to other nodes that it encounters without requiring any information about the structure of the network. The probability of message delivery is increased by using message replication. The flooding routing protocols include Direct contact, Epidemic Routing, Two-Hop relying, Tree-Based Flooding and Prioritized Epidemic Routing [8].

The forwarding routing style is a knowledge-based approach without replication. The nodes collect information about other nodes in the network to select the best path to forward the message to the destination. Location-Based Routing, Source Routing, Per-Hop Routing and Per-Contact Routing [8] belong in this category. The following paragraphs briefly describe the protocols that will be investigated as part of this research [15].

- **Direct Delivery** routing [16] is the most simple flooding routing protocol in which the node directly sends the message to the destination without any further message transmission.  

- **Epidemic** routing [13] was historically the first flooding-based protocol, where the messages are replicated to all encountered nodes. Given sufficient storage capacity and radio bandwidth to base stations, epidemic routing provides the highest delivery probability, but also uses the most energy.

- **PROPHET** [14] attempts to improve delivery probability for a limited number of copied messages by sending copies to the “best” nodes it meets. It uses delivery predictability to estimate which node is more likely to deliver the message to the final destination.

- **Spray and Wait (SAW)** routing [12] combines the distribution speed of Epidemic and the simplicity of Direct Delivery protocols. In this protocol the source node sprays L message copies to the “best” nodes it meets. It uses delivery predictability to estimate which node is more likely to deliver the message to the final destination.

- **Binary Spray and Wait** is a version of SAW where the node transfers half of the L message copies between the source node and the neighbor and keeps the remaining copies at the source. When only a single message copy remains, the node delivers the message only to the destination [15].

### 2.2. The ONE simulator

The Opportunistic Networking Environment (ONE) simulator can be used for complex DTN simulations and combines movement modelling, routing simulation, visualization and reporting in one framework [17]. ONE allows for easy extensibility using Java and has been widely used for DTN and mobility research [18]. The modelling of node movement, inter-node contacts, routing and message handling are the main functions of the ONE simulator. Result collection and analysis are done through visualization, reports and post-processing tools. ONE is capable of visualizing results of simulation in two ways: via an interactive Graphical User Interface (GUI) and by generating images from the information gathered during the simulation.

This work extends the open-source ONE simulator by allowing protocols to make forwarding decisions based on the available energy resources. We also extend energy modeling to include energy harvesting, which allows energy resources to be periodically replenished.

2.3. Energy-Efficient DTN Routing Protocols

Previous work on energy-efficient DTN protocols has mostly analysed the energy performance of protocols. However, there has been limited research on protocols which adapt their behaviour based on the available energy.

Sociievole et al. [18] evaluated the performance of the five routing protocols including Epidemic, Spray and Wait, PROPHET, MaxProp and Bubble Rap with the Opportunistic Network Environment (ONE) simulator. Their analysis revealed the most effective routing protocol with metrics such as delivery ratio, delivery latency with energy consumption. They also indicated the causes of delivery latency and energy consumption. While this work only measures energy consumption and ignores the impact of the energy constraints, the investigation is relevant to this study in terms of evaluating the routing performance related to energy consumption.

Li et al. [19] evaluated the performance of routing protocols under resource constraints and proposed Markovian and Ordinary Differential Equations (ODEs) to address the problem. This work evaluated the performance of a number of routing protocols, including two-hop relaying, epidemic routing and k-hop forwarding with an energy constraint based on a continuous time Markov chain. They derived expressions for the performance of message delivery delay and delivery cost analytically rather than using simulation, which limits insights into the stochastic interactions that can emerge from introducing the energy constraints.

Another body of literature focuses on maximizing network lifetime. [20] proposed EXLIOSE (EXTending network Lifetime in Opportunistic SEnsor Networks), a routing protocol for mobile WSN based on a new metric called Energy Shortage Cost (ESC). The aim of this approach is maximizing network lifetime by choosing the next hop based on ESC which considering the energy consumption and residual energy of each sensor node. Another approach is Energy-Aware Epidemic Routing (EAER) [21] which is an extension of the n-Epidemic routing which aimed to achieve the best performance in terms of packet delivery ratio and energy consumption. This approach focused on optimizing the possibility of sending messages from a node to its neighbors.

Previous researchers have investigated energy consumption of protocols and have not modified the behavior of algorithms in terms of available energy. There has been little research done in energy-aware DTN routing protocols for mobile nodes [21, 22]. However, while these proposals increase the message delivery rate, their energy consumption and node residual energies only consider transmission and listening energy costs for the radio, but not the sensing cost or the ability to gain energy through harvesting. This paper provides a more comprehensive modeling of energy costs (radio and sensing) and gain (energy harvesting) in DTN. Based on this energy model, we propose a novel approach that enhances existing routing protocols for higher packet delivery through energy-aware forwarding decisions.

3. Flying Fox Monitoring Application

While the results of this work may be more generally applicable to mobile DTN applications, the algorithms and approaches are designed for a specific animal monitoring application – the tracking of fruit bats across large foraging areas in Australia [4, 22]. A special WSN node for flying fox behaviour monitoring (called Camazotz) has been designed and deployed, which includes a GPS receiver for position determination, accelerometers for activity analysis, a
microprocessor, data store and radio for communicating data, a small battery and a solar cell for energy harvesting. The node is restricted to a weight of 30g. Figure 1 shows a Camazotz node on a flying fox.

Flying foxes periodically visit foraging areas at night where they feed. The foraging areas are seasonal, depending on which fruits or nectars are currently in season. During the day, they sleep in roosting camps. These camps are semi-permanent, and may contain thousands of flying foxes. Flying foxes roost in trees, so they can harvest energy during the day from their solar cells (as opposed to microbats which roost in caves). WSN base stations can be placed at the roosting sites to download data collected during the night.

Figure 1. The Camazotz node on an animal

An individual does not always return to the same roosting site, and not every roosting site can be equipped with a base station. DTN protocols can improve delivery rates by relaying data from flying foxes, which do not roost near base stations. Despite frequent changes in the motion patterns, feeding areas, and resting areas, flying foxes follow regular daily patterns that can be exploited by routing protocols to improve delivery probability of packets to a base station.

4. Energy-Aware Routing Protocols

This paper investigates two energy-aware strategies for DTN protocols, namely the Threshold and RRE strategies. These enhancements can be applied to any of the DTN existing protocols.

4.1. Threshold Algorithm

The threshold algorithm has a user-defined energy threshold below which the node will conserve its energy, and will only transmit packets directly to a base station.

The energy threshold $E_{th}$ is a fixed percentage of the harvested energy in the system. A threshold of 0% corresponds to no energy awareness and a threshold of 100% is equivalent to the Direct Delivery algorithm.

Once the stored energy falls below the threshold, the protocol will only transmit to a base station and it will not forward packets to other nodes or receive packets for forwarding (effectively the node changes to the Direct Delivery protocol). The nodes will also stop collecting data from sensors (such as GPS) if the energy falls below a fixed threshold, as the energy requirements for sensing can also be significant.

Implementation of the Threshold strategy in the ONE simulator is straightforward. Most implementations of DTN routing protocols rely on underlying low-level functionality that is shared across all higher-level protocols. For example, the “ActiveRouter” java class implements the functionality associated with dropping packets due to the expired time-to-live

4.2. Remaining Required Energy Algorithm (RRE)

In the Threshold algorithm, users control the performance of the routing protocol through the threshold parameter. However, it can be a tedious task to find the optimal value of the threshold as it depends on the operating conditions, such as the change in the energy harvested per day, or the change in the motion patterns of the tracked objects. The RRE algorithm uses a heuristic to derive the threshold automatically.

The underlying principle is to conserve sufficient energy to allow nodes to transmit existing packets in their buffer to a base station at the time of their next encounter. Effectively, the strategy is to prioritize transmission of direct packets to the base station over multi-hop data-muling packets and over sensing additional data. Energy that a node should reserve depends on two factors: 1) the energy required to transmit all packets collected since the last encounter with a data sink ($E_{sink}$), which is dominated by scanning the radio channel for the presence of a base station. The energy threshold for entering the energy conservation state is therefore:

$$E_{RRE} = E_{tx\_packet\_Today} + E_{scan}$$

where $E_{tx\_packet\_Today}$ estimates the energy needed to deliver packets to a base station based on the node’s current packet buffer size and $E_{scan}$ can be estimated given the remaining time until the next energy harvesting event.

The following equations define the energy that is used for forwarding current packets and scanning:

$$E_{tx\_packet\_Today} = N_{packet\_Today} x E_{per\_packet} x C_{tx\_loss}$$

$$N_{packet\_Today} = N_{current\_packets} - N_{previous\_day\_packets}$$

$$E_{scan} = (t_{remaining\_time} + t_{scan\_interval}) x E_{rx\_scan}$$

where $N_{packet\_Today}$ is the number of packets that the node received and/or generated today, $E_{per\_packet}$ is the amount of energy that is needed to transmit one packet. As some packets may be lost due to collisions, we use a multiplication factor $C_{tx\_loss}$ to ensure enough energy is reserved for packet retransmissions. $t_{remaining\_time}$ is the time until the node starts harvesting energy next time, $t_{scan\_interval}$ is the time interval that node scans its radio for presence of another node, and $E_{rx\_scan}$ is the amount of energy that node uses to scan the radio channel.

Similar to the threshold algorithm, RRE algorithm can be implemented in the lower layer of the network stack and thus is reusable across many mainstream DTN routing protocols. The RRE algorithm requires the nodes to be able to accurately estimate the time when they next recharge their battery resources and the time of the next contact with a base station. As flying foxes sleep in the roosting camps and harvest energy during the day both assumptions are true in our motivating application, or indeed other solar energy based deployments of sensor networks that collect data at regular periods.

5. Performance Evaluation

Comprehensive evaluations of both the Threshold and RRE algorithms with mainstream DTN protocols are undertaken using
5.1. Simulator Extensions and Scenarios

Configuring a simulation scenario in the ONE simulator typically involves defining motion traces, packet generation events, and hardware-related simulation parameters. The ONE simulator supports external motion traces for the simulated nodes and accepts time-series of node locations (t, id, x, y) as an input.

As the flying foxes dataset is relatively sparse in recorded contacts between animals, we model behavior and motion patterns of the animals based on empirical traces. Specifically, we classify the animal behavior into one of three states: resting, foraging, and travelling between foraging sites. The animals are stationary in the resting state, follow random movement within the foraging area in the foraging state and follow the shortest path with random fluctuation to travel between foraging areas. Parameters of the random motion were obtained empirically.

More specifically, we define five roosting areas and five foraging areas. Two roosting areas are equipped with base stations. Twenty animals are equipped with Camarozzi sensors. Flying fox travels to a random foraging area and then returns to a randomly chosen roosting area. Ten flying foxes return to a roosting area with a base station every night, the other animals never roost at a base-station camp.

The scenario runs for one week (7 days and 7 nights). The flying and foraging time is 3 hours, and during this time GPS position readings are made every 4 - 6 minutes. Each reading generates a 100 byte packet, and each node generates approximately 3.5KB per day. If a node runs out of energy, it cannot take any more GPS readings or participate in communication. The energy consumption model in the ONE simulator [23] is insufficient for our purpose, as it does not support energy harvesting. We extend the simulator to allow for periodic harvesting of energy at predefined time intervals (such as daylight periods for solar harvesting) and introduced a parameter that limits capacity of the energy storage.

The implemented parameters are motivated by the existing system deployment. Our nodes include batteries with 300mAh capacity corresponding to 4000J of stored energy. Each GPS reading consumes energy of 0.75 J. We use a low-power listening scheme set with 1% radio duty cycle [24] that uses 0.0523 to transmit a packet and 0.0023 to receive a packet or scan the radio channel for presence of another node. Radio range is 500m, transmission speed is 250 kps, and each node has 5MB buffer space. The packet buffers are large enough to store all messages generated during the simulation.

Each day, sensor nodes harvest energy up to the maximum battery capacity. Three different harvesting scenarios are investigated to evaluate how the energy-aware algorithms work under different conditions:

- **Low energy**: Daily energy harvesting is 20J, and is slightly higher than the minimum energy required for direct transmissions of generated packets to a base station and continuous scanning of the radio channel.
- **Medium energy**: The energy harvesting is 30J, which is sufficient to support some store-carry-forward of packets to the base station. Flooding protocols (such as Epidemic) still run out of energy.
- **High energy**: The energy harvesting is 40J, which is double the low energy setting, which should be sufficient to transmit all data to the base station.

5.2. Performance Metrics

The main performance metric is Packets Delivered (per day). The number of packets delivered to the base station is used rather than the more standard delivery rate metric (i.e., percentage of packets that are delivered to the destination). The reason is that the number of generated packets can vary for different nodes and routing protocols, as nodes do not generate packets when they run out of energy.

We normalize the metric per simulation day, so it is independent of the duration of the simulation. The number of potential packets generated per day (if no nodes run out of energy) is approximately 20 nodes x 12 packets per hour x 3 hours, The exact number for these simulations is 727. Dividing the packets delivered by 727 gives the potential packet delivery percentage, e.g. since only half the flying foxes return to base station equipped roosting areas, Direct Delivery delivers around 360 packets per day, or 50% delivery rate. We use two other metrics in this paper. Overhead Ratio is defined as the ratio of total number of packets transmitted in the network and the number of uniquely delivered packets. Average Latency is calculated as the average time it takes a packet to get delivered to a base station.

5.3. Simulation Results

We evaluate performance of both Threshold and RRE algorithms with 4 popular DTN routing protocols: Epidemic, Prophet, SprayAndWait, and Direct Delivery. We calculate statistics based on 3 randomized trials and show average results.

Figure 2 shows the number of packets that are delivered to the base station per day using the Threshold algorithms for the medium energy scenario. Note that the threshold of zero in the Threshold algorithm corresponds to an unmodified operation of the routing protocol that ignores the available energy. When the threshold is one, on the other hand, the protocols only deliver direct packets.

![Figure 2. Delivered packets per day for medium energy budget (30J) in Threshold algorithm for different thresholds.](image)

In general, while all three protocols improve Direct Delivery protocol when ignoring the available energy, their delivery rate can be increased by up to 19% for the optimal threshold. Epidemic protocol performs worse due to the high forwarding costs. SprayAndWait and Prophet perform similarly as both constrain the network traffic to prevent network-wide flooding. However, Prophet has more space for improvement once we enable energy-awareness as it is able to use constrained energy resources more efficiently. In summary, an improvement of 10-19% can be achieved by optimizing the threshold. Prophet achieves the best performance, as it is able to utilize periodic motion patterns of the animals to maximize the probability of data delivery at the base station. Epidemic protocol performs the
worst as it consumes energy on duplicate packets that potentially get delivered to the base station multiple times.

Note that the threshold at which individual routing protocols perform the best is different: Epidemic and SprayAndWait perform best at 0.3, while Prophet performs best at 0.5. This is because Epidemic and SprayAndWait algorithms forward packets without any information about the network and thus benefit more when they stop forwarding packets early. The optimal threshold improves the original SprayAndWait, Epidemic, Prophet routing protocols by 9%, 17%, 13% respectively. The optimal threshold can change under different operating conditions, such as different energy harvesting rates, packet generation rates, or changed motion patterns. This illustrates the need for a protocol that can achieve optimal routing performance when operating conditions change.

Figures 3 and 4 shows the Average Latency and Overhead Ratio for medium energy budget scenario.

![Figure 3](image3.png)

**Figure 3. Average Latency for medium energy budget (30J) in Threshold algorithm for different thresholds**

![Figure 4](image4.png)

**Figure 4. Average Latency for medium energy budget (30J) in Threshold algorithm for different thresholds**

We can see that as the threshold increases, or as the nodes increasingly limit their participation in data muling, the number of packet retransmissions (the packet overhead ratio) decreases (Figure 3). Larger threshold allows nodes to conserve their energy for direct transmissions to the base stations. Therefore, the nodes will be able to transmit packets collected in the foraging area to the base station. As many of these packets were generated before nodes entered the foraging area, the average latency will increase (Figure 4). However, if we increase the threshold past the optimum, the nodes will spend all their allocated energy before they enter the foraging area, which leads to a sharp fall in the average latency.

Finally, Table 1 compares the performance of the routing protocols for all three energy scenarios, in each case showing results for the energy-unaware version (effectively a threshold of 0%), the best threshold algorithm, and the RRE algorithm. Relative improvements of the best threshold and RRE to the base algorithm are also shown. The results show Direct Delivery is unaffected by the new protocols and is consistently the worst performer. The Threshold strategy with an optimal threshold improves performance of energy unaware algorithms by 9-17%. RRE strategy performs similarly to the best threshold algorithm and slightly outperforms the best threshold algorithm in scenarios with medium and low energy budget. However, for the high energy harvesting scenario the best threshold strategy sometimes outperformed the RRE strategy. This is because RRE reserves energy based on the packets stored in its buffer. If many copies of packets are generated (e.g. by Epidemic) and stored in different nodes, then unnecessary energy is reserved for delivery of these multiple packets, and less can be used for collecting data and gathering other packets. The overall best performer in lower and medium energy is the Prophet algorithm with RRE enhancements.

### 6. Conclusion and Future Work

This paper investigated DTN routing algorithms for a specific application scenario of wildlife tracking. Observing that substantial improvement in data yields is possible across all protocols, two energy-aware enhancements for DTNs are presented that consider the available energy in making their routing decisions. These techniques are evaluated in terms of appropriate utilization of residual energy that translates to higher packet delivery rates. Overall, the Threshold algorithm with optimum threshold and the RRE algorithm can improve data yields by 10-19% compared to energy-unaware routing protocols. RRE can achieve similar or better performance compared to optimum threshold in low energy and medium energy scenarios.

There is clearly a need for further research in more advanced energy-aware routing protocols. Future work will investigate the application of these energy-aware algorithms in different application scenarios, for example in scenarios where data collection and energy harvesting are both being undertaken during the day. Currently, the best threshold is chosen manually. It is planned to investigate how this can be determined automatically. Our current project investigates a fixed energy input during each day’s energy harvesting. It is also planned to investigate how variable energy harvesting affects the algorithms’ performance.

<table>
<thead>
<tr>
<th>Harvesting</th>
<th>Low Energy</th>
<th>Medium Energy</th>
<th>High Energy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Algorithm</td>
<td>RRE</td>
<td>RRE</td>
<td>RRE</td>
</tr>
<tr>
<td></td>
<td>Energy</td>
<td>Best</td>
<td>Energy</td>
</tr>
<tr>
<td></td>
<td>Unaware</td>
<td>Threshold</td>
<td>Unaware</td>
</tr>
<tr>
<td>Epidemic</td>
<td>280</td>
<td>316 (13%)</td>
<td>324 (16%)</td>
</tr>
<tr>
<td>Prophet</td>
<td>315</td>
<td>354 (12%)</td>
<td>355 (13%)</td>
</tr>
<tr>
<td>Spray&amp;Wait</td>
<td>312</td>
<td>341 (9%)</td>
<td>343 (10%)</td>
</tr>
<tr>
<td>DirectDelivery</td>
<td>234</td>
<td>234</td>
<td>234</td>
</tr>
</tbody>
</table>
7. References


