A Sensor Network for Biological Data Acquisition*

Tara Small, Zygmunt J. Haas,
Department of Electrical and Computer Engineering
Cornell University, Ithaca, NY, USA 14853.
tsmall@cam.cornell.edu, *haas@ece.cornell.edu*
http://wnl.ece.cornell.edu

Alejandro Purgue, and **Kurt Fristrup**
Cornell Lab of Ornithology
Ithaca, NY, USA 14850.
{*app22,kmf6*}*@cornell.edu*

1 Abstract

Infostations **offer geographically intermittent coverage at high data rates for mobile wireless networks. The** *Infostation* **model trades off delay of data delivery for increased network capacity. Replication and storage of information in multiple nodes of a mobile network can also be traded off for reduction in delay. Thus, augmenting the** *Infostation* **model with information replication, a new concept which we refer to here as the *Shared Wireless Infostation Model (SWIM)*, results in overall improved capacity-delay tradeoff at the expense of modestly increased storage requirements.**

In this paper, we propose to apply SWIM to solve a practical problem - information acquisition from radio-tagged whales. In particular, we calculate the expected storage increase for the reduction in delay. Storage requirements can be further improved without affecting the delay by wisely erasing the replicated information from the network nodes. We study the performance of five storage/erasure techniques, which increase the computational complexity of the storage algorithm, in order to further mitigate the storage increase. The results of our study will allow a network designer to implement such a system with a sufficient buffer size, as to ensure with some level of confidence that the information will be successfully carried through the mobile network.

*This work is based on an earlier work: "The Shared Wireless Infostation Model: A New Ad Hoc Networking Paradigm (or Where There is a Whale, There is a Way)," in Proceedings of the fourth ACM International Symposium on Mobile Ad Hoc Networking & Computing, pp. 233-244, June 2003 *http://doi.acm.org/10.1145/778415.778443*

2 Tagging Whales

Large whales, and marine mammals in general, are keystone species both in public interest and in assessing the environmental impacts of human activities. Eight species of large whales are on the Endangered Species list: *blue whale*, *bowhead whale*, *finback whale*, *humpback whale*, *northern* and *southern right whales*, *sei whales*, and *sperm whales*. Upon hearing noise from underwater tests, *beluga whales* will often flee the location at full speed for 2-3 days and not return to the site for weeks. *Beaked whales* have been stranded in association with naval exercises on several occasions. All of these species are difficult to study because of their enormous home ranges, the expense of oceanographic cruises, and the paucity of locations for fixed monitoring stations. Wireless telemetry offers unequalled opportunities for monitoring the movemets and behaviors of whales and other marine mammals. Whales are favorable subjects for radio telemetry because of their large size and their regular visits to the surface to breathe. Radio tagged whales can provide a wealth of oceanographic information along with data regarding their movements, because collectively they exploit a variety of resources across a wide range of oceanic habitats.

Implanting animals with miniature electronic sensing and transmitting tags provides unique opportunities to observe physiology, movements, and social behavior in a free-ranging context [1]. The addition of environmental sensors to animal tags provides the capacity to monitor ecological and oceanographic processes, that is an efficient method to monitor regions of biological interest,

which may be difficult to reach otherwise. Although some scenarios permit recovery of the tags, a much broader domain of applications requires implementation of a telemetry system to obtain the data from the tags, usually using radio frequency signals [2]. Design of radio tags confront conflicting demands. Transmit power must be minimized to enable extended operations in a small form factor. On the other hand, the enormous home ranges of whales argue for substantial transmission power to maximize the distance over which the tag telemetry can be received.

The vast majority of today's radio tags are simple beacons that broadcast signals with frequency on the order of a second. To recover the data from the tags, animals are tracked with intensive operator effort, either by approaching and following the animal or by making coordinated measurements of bearings to the triangulate signals from two or more locations. The operator often measures the bearing by swinging a directional antenna through an arc and deciding on the direction that presents the strongest signal. This approach yields valuable data; however it also suffers from severe limits on the number of animals that can be tracked and the area that can be monitored. Alternatively, satellite radio tags have been also in use for some time, with the ARGOS system being the primary provider of such a service[1]. ARGOS satellites orbit the Earth with an approximate 5000-kilometer-diameter "footprint," and in most areas they provide each day only a limited number of opportunities for offloading (recovery) of data. Furthermore, each offloading is limited to a data packet of 256 bits of data, and the system limits each tag to one message transmission every 45-200 seconds.

The seemingly irresolvable conflict between minimizing transmit power consumption and maximizing the area monitored can be successfully addressed by bringing the infrastructure for receiving the tag telemetry close to the tags themselves, where "close" usually ranges from a few hundred meters to a few kilometers. Of course, bringing the infrastructure close to the free-roaming animals may not be a trivial matter, when taking into the account the size of the animals' habitats. Many fixed receiving stations may be required, especially if the animals' movement patterns are not well specified or if it is unlikely that tagged individuals will pass close to a single receiver before exhausting the data storage capability or battery lifetime of their tags. Another option is to use mobile receiving systems, which systematically survey the animals' habitat. Data would be offloaded from each animal's tag when the receiving system reaches the vicinity of the animal. However, for large areas of habitat that are difficult to access (open ocean, tropical rainforest), the safety, expense, and logistical difficulties of sustaining regular surveys may be insurmountable.

Here, we advocate a different approach. In our approach, the infrastructure is extended to the mobile nodes by the mobile nodes themselves; i.e., by creating a sensor network [3]. In other words, we allow the information created in the network to be replicated among the network nodes. More specifically, a piece of information is allowed to propagate among the mobile nodes in the network. When the two nodes come into communication contact due to their mobility, the nodes exchange their stored information, saving a single copy of each packet on each whale tag. Then, when one of the network nodes which carries the information reaches the vicinity of a collecting station, the information is offloaded to the collecting station. To increase the probability that the information is recovered from the network, a number of collecting stations can be distributed throughout the habitat. Distribution of the collecting stations should be done in a way that maximizes the chances of information offloading.[2] Thus, only one replica of the information piece needs to reach only one collecting station to be successfully offloaded. Of course, this system might require each node to store and forward a substantial amount of data that originated from many other network nodes.

The idea of intermittent connectivity through a multiplicity of stations is not new; the $Infostation$ model proposed by researchers at WINLAB[3] at Rutgers University offers a similar approach [4]. The novelty in our design is the replication, storage, and propagation (i.e., diffusion) of the information within the $Infostation$ environment. Our system is essentially a marriage of the $Infostation$ model with the *ad hoc networking* technology [5]. Thus, we refer to this augmented $Infostation$ approach as the *Shared Wireless Infostation Model (SWIM)* [6]. SWIM allows delay reduction of the $Infostation$ model, especially when the number of $Infostations$ (SWIM stations) is relatively low.

The SWIM tags and network communications protocols

[1] `www.argosinc.com`

[2] For example, the collecting stations should be placed near areas that are frequented often by the animals, such as water reservoirs.

[3] `winwww.rutgers.edu/pub/docs/research/Infostations.html`

combine the best features of two existing marine mammal technologies: the small size and light weight of line-of-sight implantable radio tags with the global coverage, similar to what the ARGOS satellite system can provide. The SWIM tags exceed the capabilities of existing systems by enabling higher telemetry rates than satellite tags, with much lower power consumption and package size. The smaller package enables attachment to a wider range of organisms, from greater distances. The tags are equipped with microprocessors and frequency-synthesized transmitters, so they can make measurements from a variety of sensors and implement sophisticated digital telemetry protocols. The tags are designed to collect sensor data continuously, to store summaries of these data in time-stamped packets, and to store these packets in memory for subsequent uploading to a receiving system. Examples of desirable data regarding the animal's status are electrophysiological signals (cardiograms, myograms), body temperature, feeding activity, orientation, depth/altitude, and local movements (acceleration). Examples of desirable environmental data are ambient acoustic spectra, ambient temperature (and salinity in the ocean, humidity in the atmosphere), and light level. The value of these data increases when they are delivered relatively promptly, because this enables adjustment of other observational schemes to take advantage of the unexpected opportunities or phenomena.

3 The Tag Sensors

The radio tag utilizes a Texas Instruments MSP430F149 microprocessor to enable field programmable operation and to schedule transmissions for power savings. The MSP430 processor provides 60 kbytes of flash memory, very low dormant power consumption (0.9 μA), an extremely small footprint, and a very low cost per unit. The MSP430 provides opportunities to monitor a variety of sensors. These include pressure sensors, light and temperature sensors, accelerometers, clinometers, microphones, and physiological electrodes. The sensor integration strategy must emphasize the following factors: minimal addition in size and weight, power shutoff capability, breadth of potential research applications, and ease of incorporating flexible logging and telemetry features in the tag software.

The radio tags can be programmed in the field, which enables researchers to adapt the transmit schedule and operating frequency to local conditions. This embedded

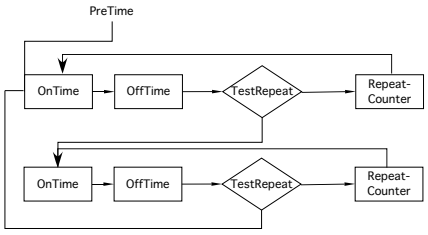

Figure 1: Timer algorithm flowchart

microprocessor is also the key to dramatic power savings. Scheduling transmissions to satisfy specific biological criteria can realize more efficient use of transmission power. Alternatively, tag sensor inputs (direct measures of activity) or an external signal (such as proximity to another whale or to a SWIM station) could be used to trigger transmissions. Scheduling is also relevant to the triggered systems, because neither the sensor nor receiver systems requires uninterrupted power. Accordingly, a flexible scheduling scheme is integral to the transmitter tags.

Scheduling requires accurate timekeeping, for this the tag uses a 32 KHz quartz crystal reference and achieves clock drift to less than one second per day. The researcher specifies the rate of regular timekeeping events. All systems are powered down between these events. The interval is defined by a 16-bit integer that determines the number of 32 kHz oscillator cycles per timekeeping event (or "chronos"), but specified by the user in seconds. Thus, chronos periods range from 30.5 μs to 2 s. Total elapsed time is stored using three sixteen-bit words, where the least significant bit corresponds to a single chronos. Forty-eight bits provide a maximum tag endurance of 272 years with a resolution of 30.5 μs. A thirty-two bit counter would impose restrictive limit on tag endurance of just over 1.5 days at the fastest chronos rate. The cost (memory, processing time) for using three words in not significant.

The scheduling algorithm is based on a repeating sequence of up to 256 tasks. The original 32 kHz clock signal is divided by a 16 bit integer. The resultant clock signal constitutes the chronos that drives the timer/counter. The task list begins with a series of tasks that are executed once, followed by a series of tasks that are repeated. Each task is stored as a pair of 16 bit counter values, representing the durations of a pair of ON and OFF actions. These counter values are followed by a field with binary flags, specifying branching conditions, and a sixteen bit

```
\event{\Ontime{03:30:00}\Offtime{4:45:10}
\flags{2}\repeatN{5}}
\event{\Ontime{00}\Offtime{10:5.301}
\flags{1}\repeatN{0}}
\event{\Ontime{0:0:0.010}\Offtime{0:0:59.990}
\flags{2}\repeatN{7}}
```

Figure 2: A short sample of the timer programing script language, depicting three events

integer specifying the number of times to repeat the task before moving to the next task. Thus, each individual sequence of ON and OFF actions can be repeated up to 2^{16} times. Note that the ON or OFF actions can have zero duration, to enable a sequence of tasks to behave as an uninterrupted period of dormancy (or, less likely, activity). A continuous period of almost 4.3 billion event cycles (2^{16} counter * 2^{16} repeat = 2^{32} event cycles) can be scheduled with a single task (36 MS with 30.5 μs event cycles). Tasks are processed in sequence, until a task with a "repeat indefinitely" flag is encountered, or until the end of the task list is reached. If the end is reached, the task sequence restarts with the first task following the initialization sequence.

Very complicated transmission schedules can be realized with this scheme. Very rapid "schedules" can be used to implement pulse code identifiers (including Morse code).

The time keeping system runs using interrupts, leaving the microprocessor in power-saving mode during the time between successive events. All microprocessor functions are implemented using interrupts, so the default state of the processor is dormant. This strategy results in approximately 64% lower power consumption than a constantly active microprocessor, with some variation in savings dependent on the precise mix of tasks.

The MSP430 software includes a simple monitor program, which manages communications with a notebook computer or PDA through a serial interface. The schedule is specified by a series of commands paired with the corresponding tasks. A host program running on the laptop (or PDA) enables the user to specify the timer tasks in an hs:mm:sec format using a simple script language (see Figure 2).

Once the schedule has been uploaded, the timer schedule is stored in flash eeprom and a flag is set internally to indicate that a valid schedule is in memory. If the tag remains powered up then the first scheduled task is executed immediately. When power is applied to the tag the processor checks for a valid schedule and proceeds to the first task if a schedule is present. Otherwise, the processor goes into a low-power state and waits for scheduling information. The monitor and schedule execution software are stored in flash memory and can be updated from a personal computer through a serial link.

The 150 MHz tag (see Figure 3) uses the Silicon Labs Si4112 phase-locked loop (PLL) RF synthesizer to generate the RF signal. The PLL is controlled by the MSP430 to produce the operating frequency specified by the user. The frequency reference is provided by the microprocessor's 4 MHz oscillator. A single external inductor determines the band of frequencies that can be programmed in the field. With appropriate inductors, this part can generate frequencies between 62 MHz and 1 GHz. With a given inductor, the operating frequency is controlled by writing to registers that specify the operating frequency as a multiple of the 4 MHz clock. At the center frequency of 150 MHz, the microcontroller can tune from 147 -153 MHz, with a resolution of 600 Hz. This allows the user to select the operating frequency at the time of deployment. The tag can also be programmed to produce ranges of frequencies in higher RF bands with this inductor, though operation in other bands would probably require changes to the matching network and antenna.

The microprocessor and PLL can generate CW or FM signals, and thus implement pulse interval coding or frequency shift keying (FSK) telemetry protocols with no additional parts. For FSK, the settling time of the Si4112 (typically 40μs) allows for modulation that supports data rates on the order of 25 kilobits per second. This is 250% of the typical voice telephony data rate, and about half the data rate of the fastest telephone modems. Output power is typically 0.4 mW into a 50 Ω load, for the low power configuration and 20 mW into a 11 Ω load for the high power version. Current consumption for the tag will be 0.9 μA dormant, 2.5 μA processing, 8 mA transmitting for the low power configuration and 35 mA for the high power configuration. Power to the Si4112 is cut when the tag is dormant.

The RF output of the PLL was boosted by a simple cascode RF amplifier to deliver a total of 20 mW of RF power. The prototype antenna for this system was a normal mode radial helix, to satisfy mechanical and hydro-

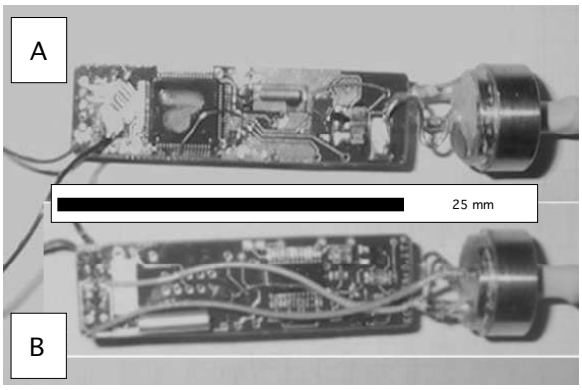

Figure 3: VHF-FM 148 MHz radio tag transmitter, showing serial port connectors and the base of the normal mode helical antenna mounted on a partially assembled Titanium housing. A: Microprocessor view; B: PLL side. Scale bar represents 25 mm

dynamic constraints while achieving desirable radiation efficiency.

The radio tag is controlled by a microprocessor, based on parameter settings specified by the researcher at the time of deployment. The researcher specifies these operating parameters using a host program running on a laptop or PDA. This program accepts a simple text script containing the researcher's specifications, and it performs consistency checks and display a visual summary of these settings to enable the researcher to scan for errors. This provides the capacity for last-minute changes in operating characteristics, including broadcast frequency.

Flexible support for complex sensor measurement and RF broadcast schedules is crucial to efficient use of battery power. Power will be supplied to all systems for very brief intervals as needed. In fact, the microprocessor itself will spend most of the time in a dormant state, with brief intervals of processing activity triggered by a combination of regular interrupts and sensor readings. The processor runs on a 4 MHz clock to enable rapid processing of interrupts. All the timekeeping functions are based on a more power efficient 32.768 kHz ("32 kHz") kHz clock (with the 4 MHz clock shut down between interrupts).

The electronics and battery are protected from the external environment by a custom made housing machined from 100% grade 2 Titanium with a pressure rating in excess of 1700 m. This choice of materials serve two purposes, to prevent corrossion and to reduce to a minimum the tissue reaction during implantation.

Here, we have presented the transmitting portions of the tags. To utilize our SWIM scheme, we require receiving functionality on the tags as well. In order to implement this full transceiving capability, we need to replace the RF chip with one that is a transceiver, like the RF Monolithics TR1000 or the ChipCon CC1000. All of these chips are programmed through a serial connection, so the changes to the electronic circuit layout would be minimal. The commands that control the transmitter would need to be changed, and we would need to implement physical and MAC layer protocols. One approach would be to work with TinyOS, the operating system developed at U. C. Berkeley to support wireless sensor networks.

4 The SWIM Networks

In the $Infostation$ model, users can connect to the network in the vicinity of ports (or $Infostations$), which are geographically distributed throughout the area of network coverage. The $Infostation$ architecture includes low-power base stations,[4] which collectively provide strong radio signal reception in small and disjoint geographical areas and, as a result, offer very high rates to users in those areas. However, due to the lack of continuous coverage, this high data rate comes at the expense of providing intermittent connectivity only. Consequently, the $Infostation$ network architecture should be used for applications that can tolerate significant delay, since a node that wishes to transmit data may be located outside the $Infostations$' coverage areas for an extended period of time. Thus, the $Infostation$ model trades delay for capacity by varying the degree of connectivity and by exploiting the mobility of the nodes.

Though significant delays can be tolerated in the whale tag application, if the delays are too long the data will likely be lost. The tags are foreign objects injected into the whales, and they are typically expelled from the host's body within 3 to 3 1/2 months. Therefore, data retrieval must occur through transmissions from the tag while it remains attached to the whale. In the original $Infostation$ model, a user must physically travel to the vicinity of an $Infostation$ to communicate, which could lead to a significant delay in our whale tag application. Thus, to address the requirements of our application, the *Shared Wireless Infostation Model* has been developed as a more timely method for data retrieval. We propose allowing informa-

[4] The information collecting stations

tion to travel through the network by sharing (replicating, storing, and diffusing) itself as well, using the mobile nodes as physical carriers.

Clearly, by allowing the packet to spread throughout the mobile nodes, the delay until one of the replicas reaches an $Infostation$ can be significantly reduced. However, this comes at a price; spreading of the packets to other nodes consumes network capacity and storage space. Thus, again, we are faced with the capacity-delay tradeoff. We have developed a new way to control this tradeoff by controlling the parameters of the spread; for example, by controlling the probability of packet transmission between two adjacent nodes, the transmission range of each node, or the number and distribution of the $Infostations$. In this paper, we examine the tradeoff between the amount of storage required and the delay experienced in the system.

First, we develop methods to calculate delays of packets in the system. Then, we examine the increase in the required storage of the SWIM system, as compared with the traditional $Infostation$ model, for a particular reduction in delay. Since the delay in both of these systems is a random variable and is unbounded, we define a probabilistic metric to describe the reduction in delay of the models. Let P_{thresh} be some threshold probability, which we choose, with which the packet will be offloaded (reach an $Infostation$) from the network. We compare the time necessary for the packet to be offloaded with probability P_{thresh} for the different network models. In general, we expect the storage capacity necessary for the SWIM model to increase, relative to the traditional $Infostation$ model, since in the SWIM model packets are copied on many nodes; however, the time necessary to store packets (before they are offloaded with probability P_{thresh}) is also smaller. Therefore, the overall and relative storage requirements of the two schemes are subject of our study here.

Note that the case in which the packets are shared between nodes with probability 1 each time two whales are "close" to each other represents the largest delay reduction and the highest increase in storage of the system. Sharing with probability 0 represents the pure $Infostation$ architecture. Thus, by sharing packets with probabilities between 0 and 1, SWIM can achieve many different instantiations of the tradeoff between network capacity and network delay.

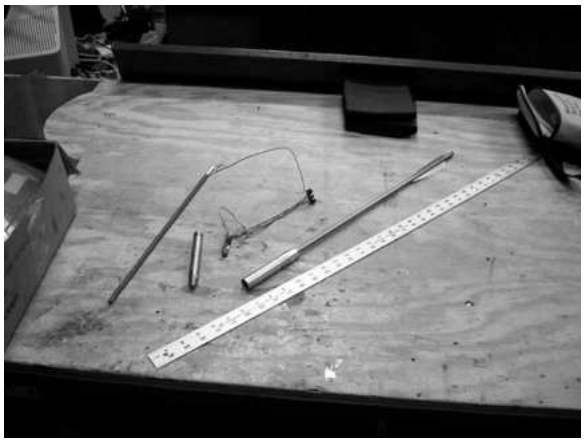

Figure 4: Whale tag prototype, to be delivered using a crossbow

5 The Information Propagation Model

Although our framework has a broad range of applications, our prime application that we address here is the support of biological data acquisition and animal tracking systems, such as the whale tags. Data that is collected on a whale tag, like the tag shown in Figure 4, is stored locally. As a whale comes in close proximity to another whale, the stored information is transmitted, with some "probability of packet transmission," p, and is stored in the recipient whale tag's memory as well. As the whales migrate throughout the system, a whale that surfaces and comes in close contact with one of the SWIM stations, offloads all the data in its memory (whether its own data or data from other whales) onto the SWIM station. Thus, as the whales feed and socialize near the surface of the water, the devices upload the packets of data at high data-rate to the appropriately placed SWIM stations.

Typically, the SWIM stations are placed on buoys, floating in the water. Since moving information from a whale tag to a SWIM station may be time-consuming, several SWIM stations are placed along the whales' paths. After receiving and storing the information from the whales, the SWIM stations transmit the information to shore, either by coordination with other SWIM stations, or directly to a satellite, whenever the next satellite passes overhead. SWIM stations could alternatively be placed on seabirds, high above the water. These stations would then be mobile, and the data would be collected at the known roosting grounds of these seabirds.

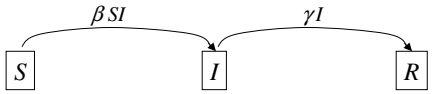

Figure 5: Markov chain model of an infectious disease with susceptible, infected, and recovered states

If a whale does not come in contact with any SWIM station for a long time, it may selectively discard the information in its memory when there is high probability that the information has already been offloaded to one of the SWIM stations by another whale. On the other hand, if a whale has been able to offload its stored information, the whale's memory could be cleared immediately. The whale tag might also retain the identifier of the packet that it offloaded, so that in the future it would not store (or even accept) information that it had stored previously. These different methods of erasure of the stored packets will be addressed in more detail later in this paper.

Delay experienced in the network (the time for a whale to reach a SWIM station) varies considerably depending on the mobility patterns of the whales, which are specific to the species of whales under consideration. One might expect daily surfacing near SWIM stations for humpback whales off the coast of the Hawaiian islands, leading to delays on the order of hours. In contrast, some migratory whales visit known feeding grounds once a year, so delays may be on the order of months.

In order to study the delay of packets, we model the propagation of each packet of data information generated by a whale as the spread of one infectious disease. We first consider the propagation of a unique packet. A whale is "infected" if it has the data packet stored in its memory. A whale is "susceptible" (to infection) if it does not yet have the packet stored in memory, but could potentially acquire the packet from another whale. A whale is "recovered" (healed from the disease) if it has offloaded the packet to a SWIM station. A packet is stored only once on each tag (one cannot be infected multiple times with the same disease); i.e., by storing the unique identifiers of the previously received packets, a whale may become "immune" to receiving the same packet again. By modeling the sharing of the packet in this way, we are able to use formulae from epidemiology to find the probability that a packet is offloaded (is "healed") as a function of the time it has spent in the system.

In Figure 5, the $S(t)$ represents the state of "susceptible" whales at time t, $I(t)$ represents the state of "infected" whales, and $R(t)$ represents the state of "recovered" whales. β is the average contact rate between two whales. Suppose that there are N whales in the system, then a whale contacts $\beta(N-1)$ other whales per unit time, of which $\frac{S}{(N-1)}$ do not yet have the disease. Therefore, the transition rate from state S to state I becomes

total infection rate
$= (\#\text{infected})(\text{contact rate})(\#\text{ susceptible whales})$
$= I[\beta(N-1)][\frac{S}{(N-1)}] = \beta SI.$

The recovery rate is labeled as γ – it is the rate of contact between a whale and a SWIM station.

total recovery rate
$=$ (whale-station contact rate)(# infected whales)
$= \gamma I.$

Recall that if there are multiple SWIM stations, then γ represents the contact rate per station; e.g., γ will double if the number of SWIM stations is doubled.

Let T be a random variable representing the amount of time a packet has spent in the system; that is, the time from packet creation until it is offloaded to a SWIM station. Once one packet reaches state R (meaning it has been offloaded), the rates will change, so we deem the model invalid. Since we consider only one packet in the model, at time $t = 0$ only one whale carries the packet, and since all the N whales are either in the state S or in the state I while the model is valid, this means

$$S(0) = N-1, \qquad I(0) = 1, \qquad S+I = N,$$
$$R(t) = 0 \text{ for } t < T \text{ and } R(T) = 1$$

By solving the differential equations defined by the rates of the Markov chain in Figure 5, it is possible to arrive at the cumulative distribution function $F(T)$, which represents the probability that the packet is offloaded after spending time T in the system. For example, if $F(300) = 0.5$, this means there is probability 0.5 that a packet is offloaded in 300 time-steps or less. By using the inverse of this function, we can choose a desired probability P_{thresh} and find the value T_p for which $T_p = F^{-1}(P_{\text{thresh}})$. This means that with probability P_{thresh}, by time T_p, the packet will be offloaded. The formula for this function $F(t)$ is given by

$$F(t) = 1 - K\left(\frac{N-1}{e^{\beta N t} + N - 1}\right)^{\frac{\gamma}{\beta}}. \tag{1}$$

6 Simulating the Delay

Many possible mobility patterns exist for the whales. Each of these mobility patterns is represented in Equation (1) through the values of the contact parameters β and γ. A simple mobility pattern is random linear mobility. This pattern will be used to examine some common $F(T)$ properties. In the simulation, the whales swim in straight lines for a fixed number of time-steps, s, with a randomly chosen velocity, and in a random direction. At the beginning of each group of s time-steps, a new velocity and a new direction are chosen for the whales to swim in a rectangular area. The area is a torus, with edges that wrap around, so a whale that swims off the right edge re-enters at the left edge; similar wrap exists for the top and bottom edges.

At the beginning of the simulation, one whale carries the only replica of the packet. At every iteration, if a whale carrying a packet is within the infection range of another whale, the packet is replicated at the other whale. If any whale carrying the packet is within infection range of a SWIM station, then the simulation is stopped, the time, T, is recorded from the creation of the packet until the termination of the simulation. The simulation was run multiple times, and the data was compiled, representing an empirical probability function, $F(T)$. As one would expect, the $F(T)$ curves are steeper (representing shorter delay) as the number of whales increases, and as the number of SWIM stations increases. Figure 6 shows the empirical $F(T)$ curves with different numbers of SWIM stations, $M = 1, 2, 3, 4$. In this example, swimming speeds of the whales were chosen from 0 to 6 units per timestep on a 300 by 300 toroidal area, and the reception radius of each station was 15 units. The curves are also steeper, due to increased sharing, as the number of whales increases. In order to validate the empirical $F(T)$, we found the corresponding theoretical $F(T)$ using the simulation to find β and γ. Through the use of the χ^2 goodness-of-fit test, we observed very good agreement between the theoretical and empirical solutions.

A more realistic mobility model captures the physical whale behavior by incorporating feeding grounds. In this enhanced model, three issues govern the direction of the whales' positions at any time: migration in a specified direction, grouping of whales, and direction of the nearest feeding ground. Females tend to group together with other females, while grown males tend to be more solitary

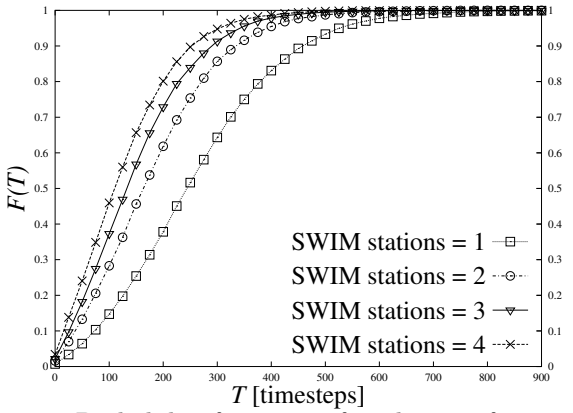

Figure 6: Probability functions of T, the time from packet creation until offloading, for different numbers of SWIM stations in the system

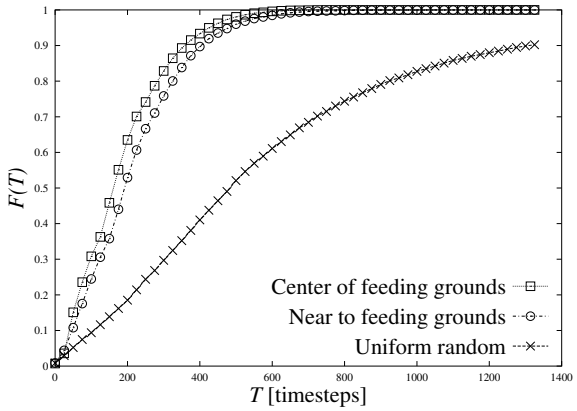

Figure 7: Effect of different SWIM station arrangements on the cumulative distribution $F(T)$

in their behavior and group with females, but not with other males. All whales are attracted to feeding grounds when they are hungry. Inside the feeding grounds, whales move slowly and sometimes stop. When a whale becomes less hungry it can leave the grounds for a significant time before returning. Direction for the whales' mobility is determined by a weighted vector sum of the directions of migration, of the direction to the nearest female, and of the direction the nearest feeding area.

Since the whales are attracted to the centers of the feeding grounds, they are likely to swim close enough to a SWIM station inside the feeding grounds to offload their packet. Thus, when SWIM stations are placed inside the feeding grounds, delays can be significantly reduced. This is shown in Figure 7 by the "Center of feeding grounds" and "Near to feeding grounds" curves. If the SWIM stations

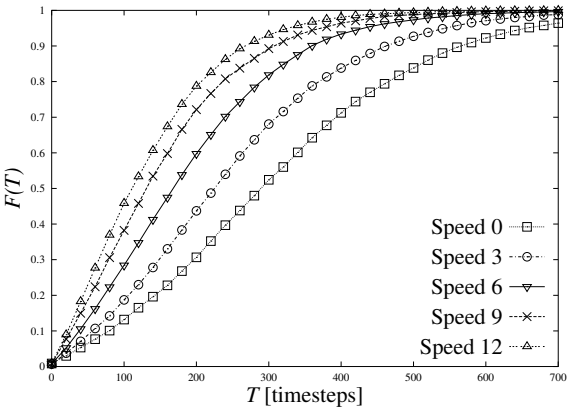

Figure 8: Cumulative distribution curves with varying speeds of the mobile SWIM stations

are sometimes placed outside the feeding grounds, delays increase, since the whales are attracted to regions far from the SWIM stations, shown by the "Poisson distributed" curve in Figure 7. Obviously, location of the SWIM stations is a very significant parameter. The grouping of whales can also significantly affect the delay, since more grouping promotes more sharing of packets.

Up to this point, we have assumed that the collection points (i.e., the SWIM stations) are fixed in their locations. Another possible model for the biological information acquisition system considers *mobile* collection points as well as mobile nodes – for example, SWIM stations mounted on seabirds that glide above the ocean along the turbulent air above the waves. Figure 8 shows that increasing the speed of the mobile SWIM stations has a positive effect on the packet offload time. Larger speeds of the SWIM stations allow them to pass through groups of whales more often, though staying near the groups for shorter periods each time. This larger frequency of visiting allows the packets to be offloaded more often and at more regular intervals.

7 Calculating the Storage Requirements

Equipped with these $F(T)$ curves, the information about the contact rate between the whales, and the contact rate between the whales and the SWIM stations, we are able to calculate the expected storage requirement for all the copies of one packet, given a desired confidence level of the packet delivery, P_{thresh}. As an example, suppose that the designer specified a confidence level of 0.9, then $T_{0.9} = F^{-1}(0.9)$ is the time necessary to wait to achieve the probability of 0.9 of packet offloading. This is the "expiration time" of the packet and its replicas. If any replica of the packet remains in the system for this maximum delay, it is erased, even if it has not yet been offloaded.

A quick, though naïve, approach of calculating the required storage for the system involves the average number of packets in the system at the time of offloading and applying Little's formula. Suppose that 10 adult whales are tagged and, at each timestep, placed randomly[5] in an area of 900 km^2 with 1 SWIM station. The transmitting range of the radio tags is 1.4 km and reception range of the stations is 3 km. This can be modeled as a system with $N = 10$ whales, $M = 1$ SWIM station, and the probability of transmission $p = 1$. From the corresponding $F(T)$ curve, we find that $F^{-1}(0.9) \approx 78$. The "expiration time" of the packets is therefore 78 time-steps.

Now suppose that each whale generates a packet every 30 time-steps. Using Little's formula with generation rate $\lambda = \frac{1}{30}$ time-steps per packet per whale, the expected number of all the packet replicas in the system is:

$$\begin{aligned} EP &= (\text{number of whales})\lambda T_{0.9} \\ &= 10\tfrac{1}{30}\ [\text{packets/time-step}](78\ [\text{time-steps}]) \\ &= 26\ [\text{packets}]. \end{aligned}$$

An estimate of the expected number of copies of each packet in the system, EI, is the average number of whales infected with the packet at the time of offloading. It can be shown from our simulation that $EI = 2.5523$ in this case. This number assists us in the calculation of a global storage requirement for the radio devices:

$$\begin{aligned} &\text{storage requirement} \\ &= (\text{duplicates})(\text{different packets})(\text{bytes/packet}) \\ &= EI * EP * (330\ \text{bytes/packet}) \\ &= (2.5523) * (26\ \text{packets}) * (330\ \text{bytes/packet}) \\ &= 21898.734\ \text{bytes} \\ &= 21.38548242\ \text{kB} = 2.138548242\ \text{kB/whale}. \end{aligned}$$

Recall that in this example, the probability of sharing packets between close-by whales is 1, so the results correspond to the largest delay decrease and the largest storage requirement of the SWIM model. Figure 9 shows that the increase in storage is very reasonable for the achieved large decrease in delay. The advantage of SWIM is even more pronounced as the number of whales increases.

[5] with a Poisson distribution

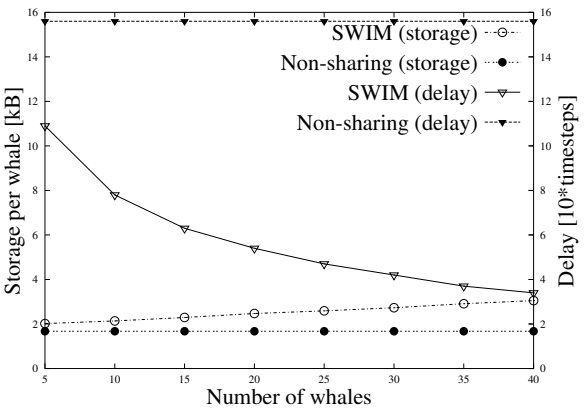

Figure 9: Necessary storage requirements and expected delays using SWIM vs. the non-sharing model

While in the non-sharing case, the per-whale storage requirement remains the same as the number of whales increases, the storage requirement in the SWIM case grows slightly due to the replication of packets. If the density of whales is larger, then the packet is shared with more whales and more storage is required. This increased storage requirement is mitigated by the fact that the delay is reduced; i.e., in SWIM, there are more packet copies in the network, but they remain for a shorter time.

The expected delay for the non-sharing system is constant over the different numbers of whales, since more whales in the system offer no advantage in this case. In other words, every whale must itself reach the SWIM station to offload its packets. On the other hand, SWIM replicates packets among the network nodes, so if there is a larger density of whales, there will be more copies of a packet present in the network. Thus, SWIM achieves smaller delays as the number of whales increases.

In practice, one would want to include an extra safety factor in the memory calculation to protect against statistical variability in the number of packets stored in a tag.[6] This safety factor is not included in our simple approach presented in this section. However, we will reexamine our storage evaluation in the more precise calculations of storage requirements in the following sections.

[6]Not including this factor would assume that the loss due to buffer overflow is negligible.

7.1 Single-Packet Storage Methods

There are numerous methods which could be used to model the packet generation, storage, and erasure. We will consider here five possible methods: JUST_TTL, FULL_ERASE, IMMUNE, IMMUNE_TX, and VACCINE. These methods progressively extend one another. In all of these methods, the original packet and **all** of its copies are erased by $T_p = F^{-1}(P_{\text{thresh}})$ time-steps after the original packet was created.

- **JUST_TTL** is the simplest method. All packets remain in the system until $T_p = F^{-1}(P_{\text{thresh}})$ time-steps have elapsed from the original packet creation.

- **FULL_ERASE** erases the copy of the packet completely from the offloading node just after it has been offloaded to a SWIM station. Once a copy of the packet has reached a SWIM station, there is no need for it to be stored on any of the whale tags. It is, however, possible that other whales still carry the packet once it has been erased from the offloading whale, so a whale might get infected with the same packet multiple times.

- **IMMUNE** erases the packet when it is offloaded like FULL_ERASE, but keeps an identifier of the offloaded packet, so the whale will not receive that packet again. We refer to this identifier as an "antipacket,[7]" since it prevents re-infection of packets.

- **IMMUNE_TX** erases the packet when it is offloaded, keeping the antipacket, like IMMUNE. It also shares this antipacket with other whales that carry copies of the offloaded packet. This means a whale may receive an identifier "antipacket" from a transmitting whale only if a copy of the offloaded packet is already stored. At that point, the copy would be erased and "antipacket" identifier kept.

- **VACCINE** erases the packet when it is offloaded, like previous methods. It also shares all packets and antipackets between whales. In this case, a whale may receive an antipacket from a transmitting whale even if the receiving whale does not have a copy of the packet stored.

[7]Similar to *antibody* of a biological agent

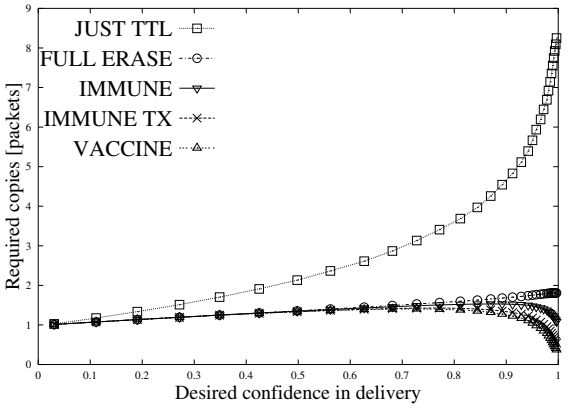

Figure 10: Expected storage required for 10 whales, assuming 4-byte identifier and packet contents of 326 bytes

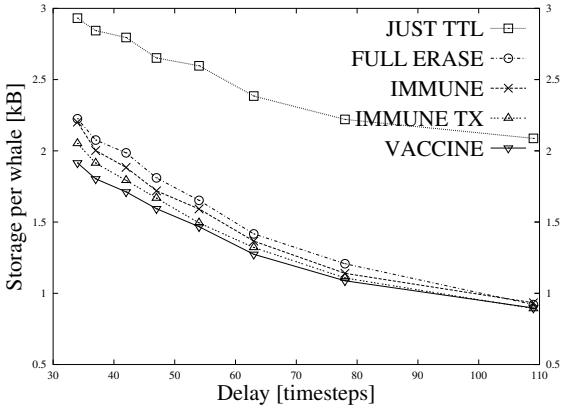

Figure 11: Storage-delay tradeoff of SWIM using the different methods of erasure

We are able to calculate the average number of copies of the packet in the system for a given time using each of these five methods. Call this average number $EI(T)$. Since the $F(T)$ curve expresses the desired confidence level as a function of time, we are able to plot parametrically the average storage requirement as a function of the desired confidence level, $(F(T), EI(T))$, as shown in Figure 10. Notice that for methods IMMUNE, IMMUNE_TX, and VACCINE the average storage begins to decrease at high confidence level. This is due to the dependence of the confidence level on T. As the confidence level approaches 1, the necessary time T for the packet to remain in the system becomes higher and higher, eventually $T \to \infty$ as $F(T) \to 1$. In the methods IMMUNE, IMMUNE_TX, and VACCINE the packet identifiers prohibit the whales from storing a copy of the packet again, so eventually as T gets large, nearly all the whales refuse storage of the packet, and the average required storage is thereby reduced.

We can also depict the storage-delay tradeoff using SWIM. We fix the desired $P_{\text{thresh}} = 0.9$, then to reduce the delay, we increase the sharing of the packets by increasing the density of whales in the system. Figure 11 exhibits the storage-delay tradeoff due to this increased sharing; clearly, to achieve shorter delay, one must invest more storage in the system.

7.2 Multiple-Packet Storage Methods

For each of these five methods, we can obtain a time-average of the number of replicas of a packet in the system, EI. With the help of the Little's formula, the mean storage requirement per whale is:

$$EI * \lambda * T_p * (330 \text{ bytes/packet}).$$

This does not, however, provide an indication of the variance of the number of packets stored on each whale tag. The variance is important to calculate the "safety factor" in evaluation of the necessary buffer size, as to ensure that due to the statistical behavior of the packet arrival process at the nodes, at most only a small fraction of the packets would be lost.

We would like to learn more about the probability distribution of the number of packets on each whale tag, Q_i. To assist us in the solution of Q_i, we model the system as an imaginary global queue that at each point in time contains all the packets present in the system. In particular, let $Q = \sum_{i=1}^{N} Q_i$ represent the number of all the copies of all the different packets in the system; i.e., the number of packets in the global queue. However, due to the complex nature of the global queue, we employ an approximation; we assume that the arrival of all the copies of a packet to the global queue occurs at the time of the original packet creation, rather than at time when the packet is replicated from one whale to another. This is a conservative approximation for the purpose of evaluation of the variance of Q_i, since in reality the arrival of the copies of a packet will be spread in time, reducing the variance due to the aggregation of such arrival processes of many other packets. We further assume that the number of replicas of the packet to arrive to the global queue is equal to the maximum number of the packet copies that will ever be present in the system. For the JUST_TTL case, pack-

ets are replicated when they are shared between whales, but are never removed from the system. Thus at time $T_p = F^{-1}(P_{\text{thresh}})$, the number of copies of a particular packet in the system will be a maximum. Using other methods, the maximum number of packets may occur at a value smaller than T_p.

We can simulate this global queue, Q. The simulation generates packets periodically for every whale in the system, given the set of periods and their offsets in time. Let $I(t_{max})$ be a random variable representing the distribution of the number of packets in the system when the expected number of packets in the system is a maximum. When a new packet is generated, the maximum number of its copies that will ever be present in the network is drawn from the distribution $I(t_{max})$. Those copies are then added to the global queue, Q, as soon as they are generated and removed after time T_p. After the simulation ends, the sample mean and sample variance of the number of packets in the global queue at steady-state are calculated.

The global queue can also be solved analytically. When the number of whales is moderately large and the arrival processes of new packets at different whales have slightly different periods, the arrivals of groups of packets act like a Poisson queue with batch arrivals. The system is said to have infinitely many servers, since all the packets are "served" at the same time. A Poisson queue with batch arrivals involves groups of customers which reach servers with *i.i.d.* exponentially distributed interarrival times. The numbers of customers in these groups is determined by the distribution function $I(t_{max})$. The service times in this case are deterministic, meaning that any customer leaves the system after a constant time T_p. Finally, since there are infinitely many servers available, customers never have to wait in the global queue; i.e., the only delay is due to the deterministic service time.

By using the global queue with deterministic service times described above and by assuming that all of the whale tags are *i.i.d* with respect to the number of packets they carry, we can simply divide the global queue by the number of whales to find the distribution of the number of packets on each individual tag. This provides not only the mean number of packets, which is already known from the single packet methods, but any quantile that the designer wishes to use in order to provide the "safety factor" in packet buffering.

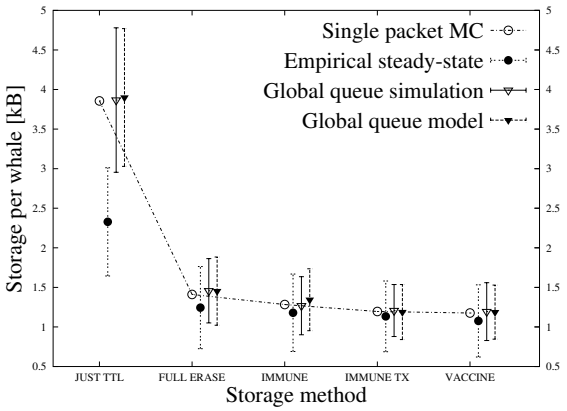

Figure 12: Mean number of packets with error bars indicating variance for a 10-whale system

Figure 12 compares the numbers of packets in the 10-whale system for four different metrics, exemplifying the methods described earlier, using a random mobility pattern for 10 whales. The first metric uses the single packet Markov chain to find $EI(t_{max})$ and uses the Little's formula $N * EI(t_{max}) * \lambda * T_p * (330 \text{ bytes/packet})$, described earlier, to give a conservative estimate of the mean number of packets. This method does not provide error bars since the variation in numbers of different packets on a tag cannot be measured using this method.

The second metric is the average number of packets in the system measured in the whale simulation at steady state. This is an empirical measurement of the actual number of packets in the system, rather than the conservative estimate used in the other methods. For this reason, the curve of the second metric is lower than the other curves. The third metric uses the simulation of the global queue with batch arrivals with distribution $I(t_{max})$, and the fourth metric calculates the probabilities analytically. These metrics all provide error bars for the variance, since they supply the entire distribution of the number of packets stored in the system.

As shown in Figure 12, the single packet Markov chain gives a reasonably conservative estimate of the packet in the system. By adding one standard deviation of the number of packets in the queue to the mean, we can ensure even less packet loss in the system. Using this estimate, the storage requirement per whale for JUST_TTL is 4.77 packets/whale, for FULL_ERASE is 1.89 kB/whale, for IMMUNE is 1.73 kB/whale, for IMMUNE_TX is 1.54 kB/whale and for VACCINE is 1.53 kB/whale. Compare

these value to the average requirement of 2.14 kB/whale calculated at the beginning of this section. We conclude that even accounting for variability in the tag queues, the storage requirements remain reasonable for practical implementation.

8 Conclusions

We have proposed SWIM, an augmented $Infostation$ model, and applied SWIM as an efficient method to solve the problem of data retrieval from animal tags. In this model, users disseminate information packets throughout the system, sharing them with other users, and since only one of the replicas needs to reach a collection point, the overall delay in offloading the data is reduced. Using a probabilistic metric for the delay, we show that the delay could be reduced by 320% for a 50% increase in storage compared to traditional $Infostation$ networks, comparing a system with 5 whales to one with 40 whales, each with packet generation every 30 timesteps.

We have shown a number of methods for storing and erasing packets, using the single and the multiple packet models. By using the single packet model to find the mean storage per whale and the multiple packet model to find the variance, one can efficiently design a system with reasonably-size storage requirements at each node and low packet loss.

Though this model is well-suited to design and evaluate moderately delay-tolerant applications, such as the above biological information acquisition system, the same methodology can be also used to model and evaluate other systems that use our augmented $Infostation$ model.

References

[1] H. Wang, J. Elson, L. Girod, D. Estrin, and K. Yao, "Target Classification and Localization in Habitat Monitoring", *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP 2003)*, Hong Kong, China, April 2003.

[2] I.G. Priede, "Wildlife telemetry: an introduction" in "Wildlife telemetry: remote monitoring and tracking of animals", *Ellis Horwood*, I.G. Priede and S.M. Swift, eds., Chichester, U. K., 1992, pp. 3-25.

[3] I.F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless sensor networks: a survey", *Computer Networks*, vol. 38, no. 4, pp. 393–442, March 2002.

[4] A. Iacono and C. Rose, "Infostations: New Perspectives on Wireless Data Networks", *WINLAB technical document*, Rutgers University, 2000.

[5] Z.J. Haas, J. Deng, P. Papadimitratos and S. Sajama, "Wireless Ad Hoc Networks", *Encyclopedia of Telecommunications*, John Proakis, editor, John Wiley, 2002.

[6] T. Small and Z.J. Haas, "The Shared Wireless Infostation Model – A New Ad Hoc Networking Paradigm (or Where there is a Whale, there is a Way)", ACM MOBIHOC'03, Annapolis, Maryland, June 2003.

[7] L. Kleinrock, "Queueing Systems Volume I: Theory", *John Wiley & Sons, Inc.*, 1975.