

On Controlled Node Mobility in Delay-Tolerant Networks of Unmanned Aerial Vehicles

Daniel Henkel, Timothy X Brown
University of Colorado at Boulder
530 UCB, ECOT 311, Boulder, CO, 80303
Tel.: 303-492-1630, Fax: 303-492-1112
{henk,timxb}@colorado.edu

Abstract

This paper explores the use of controlled node mobility in delay-tolerant networks. Mobile nodes are used to ferry data between otherwise unconnected network nodes. In particular, the problem of efficient route design for aerial data-ferrying nodes is studied. We introduce two new approaches to route design, namely the chain-relay model and the conveyor-belt model. Analytical evaluation of the proposed route designs shows their relative merits depending on node velocity, data rate, and buffer size. Further, we introduce a new problem framework, which departs from the traditional approach of seeing communication as limited to one data rate within a circular region around a node. Our model takes into account multiple data rates varying with the separation distance from a node. Preliminary evaluation suggests performance improvements over existing approaches when combining our route designs with this problem framework.

1 Introduction

Mobile ad-hoc networks (MANETS) are expected to play an increasing role in making ubiquitous computing and communications become a reality. Already mesh networks are increasingly being deployed to give people access to the Internet where no wired infrastructure is readily available. However, with the advent of ever cheaper computing power in ever smaller mobile devices the potential of MANETS goes beyond accessing the Internet. Example applications include search and rescue coordination without fixed infrastructure; scientific observation in high-risk areas; and inter-vehicle communication for collision avoidance.

The data that is used by applications in these networks can be classified as either *real-time* or *delay-tolerant*. Real-time traffic includes voice over IP, video conferencing, or monitoring of critical processes. Delay-tolerant traffic does not carry an urgency and only eventual, reliable reception is important. Examples are email, instant messages, or sensor data of a long-term experiment.

For a wireless ad-hoc network to be able to carry real-time data, there needs to be *contemporaneous, reliable path* from sender to receiver. Such a path is more likely the shorter the sender and receiver separation and the

more dense the radio nodes.

Real-time communication is challenging for wireless mobile ad-hoc networks because of inherent variability in wireless connections combined with changing node positions. In the extreme case of sparsely connected, moving nodes, some nodes might not be able to connect with any other node for a long time. In this scenario only delay-tolerant communication is feasible.

The Ad-hoc UAV-Ground Network project (AUGNet) at the University of Colorado lends itself to studying methods to support sparsely-connected, delay-tolerant networks [1, 2]. AUGNet features fast-moving planes which can span large distances in a relatively short time. It is exploring how moving nodes can be used to reduce delay and improve reliability of communication in sparsely connected networks. Controlled plane mobility can increase battery life of ground nodes, enable coordination of large-scale deployments of and among swarming planes, and simplify data collection and distribution among ground nodes.

This paper investigates the role of ferrying in two communication models: the first has a constant data rate up to a maximum distance and zero rate otherwise; the second has a continuously variable rate that decreases with distance and is always non-zero. Several models of ferry-

ing are developed for both a single transfer between two nodes and a transfer between several nodes and a hub. The results are three-fold. First, they reveal which fixed-rate ferrying model is the most suited for a given set of external operating conditions. Second, a delay-optimal scheduling algorithm is derived which assigns ferry time to nodes in a star network. Third, network performance is greatly improved by considering a variable data rate model where the data rate is a continuous function of node separation distance.

The paper proceeds as follows: Section 2 describes our model for delay-tolerant networking and route planning. Section 3 analyzes route design under full knowledge and introduces our new problem framework. Section 4 gives an overview of related work, and Section 5 concludes the paper with an outlook on future work.

2 Modeling Delay-tolerant Networks

2.1 System Components

Task Nodes

Task nodes form the basic sparsely connected ground network. The nodes operate over an area A , called the *survey area*. For our purposes these nodes are assumed to be stationary. Every node has a buffer of size b_{node} bytes and a wireless radio capable of transmitting or receiving at data rate R_{node} in a coverage area forming a concentric sphere around the node with radius r_{node} . For this scenario the data rate is considered constant within the coverage area.

It is assumed that the separation distance l between task nodes is greater than the communication range of the radios, i.e., $l > r_{node}$. Thus there is no connection between nodes and no possibility for routing traffic among them without the help of some other nodes.

A node can send data to a receiver that is at a distance less than r_{node} as long as the receiver is outside of the interference range of any other transmitting node. The interference range is assumed equal to the communication range.

We refer to this model as the fixed radius communication model.

Ferries

Mobile nodes called *transport ferries* are capable of moving around the survey area to any location requested. There are n ferries on the survey area at any given point in time, denoted F_1, F_2, \dots, F_n . They are areal vehicles and the motion can be controlled arbitrarily. The motion

is assumed linear with speed v , but ferries can also stop, e.g., for data exchange. Every ferry is equipped with a radio capable of transmitting or receiving at data rate R_{ferry} in a coverage area forming a concentric sphere around the ferry with radius r_{ferry} . The ferry can buffer b_{ferry} bytes of data. For simplicity we are assuming buffer size, communication radius, and data rate are the same for both, task nodes and ferries and so the subscripts will be dropped. Ferries can talk to nodes or other ferries, but not at the same time.

2.2 Performance Metrics

The analytical evaluation of the proposed ferry path models will compare the following performance measures. They are computed for the ideal case of perfect coordination between ferries.

2.2.1 Throughput

The throughput as experienced by the sender or receiver can be calculated as follows.

$$T = \frac{\text{buffers delivered per cycle}}{\text{cycle period}} \quad (1)$$

Herein the cycle period comprises the total time for a ferry to follow a given path and return to the original position and state. It includes ferry travel times plus times to fill and empty buffers. Such a cycle may involve one or more buffers of traffic delivered.

2.2.2 Average Packet Delay

The average delay experienced by one packet can be derived by looking at the best-case and worst-case delays incurred by the system with subsequent averaging¹. The best-case delay occurs when a packet stored in the source buffer gets loaded onto the ferry last, i.e., just before departure. The worst-case delay occurs when a packet arrives at the source buffer just as the ferry departs, i.e., it gets buffered at the source as the first packet until arrival of the next ferry. Figure 1 illustrates this method.

2.2.3 Buffers

Packet delay includes the ferry travel times, time that a packet spends waiting in ferry buffer while other packets load and unload, and time for buffering at the source. Propagation delay is assumed negligible. Packets are considered small relative to buffer sizes so that buffers

¹The validity of this approach can be shown for the chain relay and conveyor belt models. For other models it is an approximation.

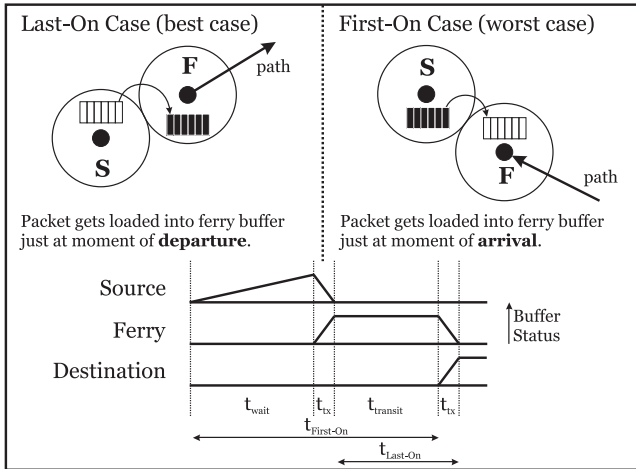


Figure 1: The two boundary cases of packet delay and corresponding timing diagram

smoothly fill and empty. Traffic flows are also smooth and continuous. Thus data is modeled as a fluid for simplicity so that we can focus on the role of mobility.

We assume a FIFO buffer policy. If R is the transmit rate then $\frac{b}{R}$ is the time required to completely fill or empty the buffer.

2.3 Traffic Flow and Ferry Path Models

When looking at traffic flow, there can be three different models: (a) Star model: all traffic flows between one special node (the hub) and the k task nodes, (b) Uni-directional model: traffic can flow between any nodes, but only in one direction, and (c) Bi-directional model: traffic can flow between all nodes in both directions.

We evaluated two different path models for ferry movement, namely the *chain-relay model* and *conveyor-belt model*. The following sections introduce these mobility models in the special case of $k = 1$ nodes communicating with the hub uni-directionally.

2.3.1 Chain-Relay Model

In this model it is assumed that all participating ferries are distributed along a connecting path from source to destination, forming a chain. The source passes available data on to the first ferry, which physically transports it some way along the path, hands it off to the next ferry, and returns to the source. This hand-off procedure is repeated until the last ferry hands the data to the destination. Figure 2 depicts this model.

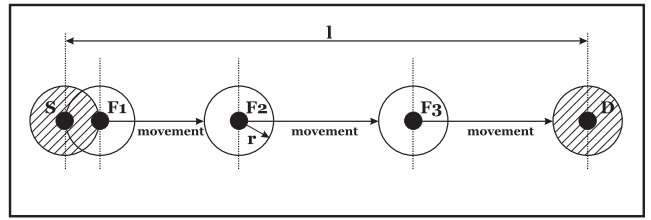


Figure 2: The Chain-Relay Model

Throughput and Ferry Travel Space

The cycle time is the time for F_1 to fill its buffer, travel to within the range of F_2 , offload the buffer, and travel back to the source node. The traveling distance is the total separation distance from sender to receiver, l , less $n + 1$ times the communication radius, divided by the number of ferries that operate on the link:

$$d_{neighbor}(n) = \frac{l - (n + 1)r}{n}.$$

Therefore

$$t_{cycle} = t_{travel} + t_{tx} = 2 \frac{l - (n + 1)r}{nv} + 2 \frac{b}{R}$$

and from (1) the throughput can be stated as

$$T = \frac{b}{2 \left(\frac{l - (n + 1)r}{nv} + \frac{b}{R} \right)}. \quad (2)$$

In this model even if $v \rightarrow \infty$ the maximum throughput is $\frac{R}{2}$. Substituting $T = \frac{R}{2}$ into (2) and solving for n yields $n = \frac{l}{R} - 1$. In this case, $d_{neighbor} = 0$, and we reach the “saturation” case of a connected chain of relays².

Delay Calculation

We compute the packet delay by averaging over worst possible delay and best possible delay. (a)The First-on

Case (worst-case delay)

The packets in the chain model get handed off between neighbor-ferries and one cycle happens between two neighbors. This cycle time needs to be accounted for in the wait time for the packets. The total delay seen by a single packet from source to destination consists of the following times.

²Note that when $d_{neighbor} = 0$ we rely on the interference distance to be equal to the communication distance. In fact, the nodes move a small distance away from the sending node and out of its interference range before communicating with the next node in the relay. A longer interference range would require a longer minimum separation between nodes.

i) packet wait delay until neighbor ferry shows up again and stops at distance r from center of source; this includes actual travel time of ferry as well as off-loading and loading at remote neighbor, plus off-loading time at the source. This is not the average wait time since the ferry had just departed and a full cycle needs to be traveled.

$$t_{wait} = 2 \frac{l - (n+1)r}{nv} + \frac{b}{R}$$

ii) ferry loading time. Even though the first packet is loaded immediately into the ferry buffer, the packet still has to wait until all packets are loaded and the ferry is ready to depart.

$$t_{loading} = \frac{b}{R}$$

iii) ferry travel delays to n remote neighbors, including the destination.

$$t_{transit} = n \frac{l - (n+1)r}{nv}$$

iv) buffer exchange time with $n - 1$ neighbors. We assume an empty neighbor buffer.

$$t_{exchange} = (n-1) \frac{b}{R}$$

(b) The Last-on Case (best-case delay) A packet experiences the following delays from source to destination.

i) n transit delays between neighbors.

$$t_{transit} = n \frac{l - (n+1)r}{nv}$$

ii) n buffer exchanges between neighbors, including the destination.

$$t_{exchange} = n \frac{b}{R}$$

The average delay of packets from source to destination is the average of best and worst case delays:

$$delay^{chain} = \left(1 + \frac{1}{n}\right) \frac{l - (n+1)r}{v} + \left(n + \frac{1}{2}\right) \frac{b}{R} \quad (3)$$

2.3.2 Conveyor-Belt Model

In this model it is assumed that every ferry travels the complete distance from source to destination, then returns to the source. Multiple ferries are sharing the same path, which increases throughput and robustness of the system. Figure 3 depicts the model.

Throughput and Buffer Status

One cycle in the conveyor belt model is given by a complete round-trip from source to destination including buffer transmission times. The corresponding cycle time is thus given by

$$t_{cycle} = t_{travel} + t_{tx} = 2 \frac{l - 2r}{v} + 2 \frac{b}{R}$$

Since there are n ferries hauling data between two nodes, the data delivered in one cycle time is $n \cdot b$.

According to (1), the total throughput is

$$T = \frac{n \cdot b}{2 \frac{l-2r}{v} + 2 \frac{b}{R}} = \frac{n}{\frac{2(l-2r)}{v \cdot b} + \frac{2}{R}}. \quad (4)$$

R is the maximum rate at which a node can transmit. $T = R$ implies that for the node to be transmitting at its maximum data, at least n_{opt} ferries are necessary:

$$n_{opt} = \left\lceil 2 \left(1 + \frac{l - 2r}{v} \frac{R}{b}\right) \right\rceil.$$

When $n > n_{opt}$, ferries arrive faster than buffers can fill. In this case, either (a) ferries get queued at the source or (b) ferries' buffers are filled partially to $\varepsilon \cdot b$, where $0 < \varepsilon \leq 1$. ε is the ratio of the time between ferry arrivals and the time to fill the buffer:

$$\varepsilon = \frac{\frac{2}{n} \left(\frac{l-2r}{v} + \frac{b}{R}\right)}{\frac{b}{R}} = \frac{2}{n} \left(1 + \frac{l - 2r}{v} \frac{R}{b}\right)$$

Packet Delay Calculation

In this section we calculate the average delay for one individual packet from source to destination analogous to the chain-relay model.

(a) The First-on Case (worst-case delay) The delay this packet experiences comprises these time components.

i) packet wait delay until the next ferry arrives and stops at a distance r from the center of the source; this includes the actual travel time of the ferry as well as off-loading and loading at the destination, plus off-loading at the source. In the single flow case there is no loading at the destination or subsequent off-loading at the source. For n ferries this delay is on average as the n^{th} fraction of the cycle time less one buffer transfer time. This is not the average wait time since the ferry had just departed and a full cycle needs to be traveled.

$$t_{wait} = \frac{2}{n} \left(\frac{l - 2r}{v} + \frac{b}{R}\right) - \frac{b}{R}$$

ii) ferry loading time. Even though the packet is loaded immediately into the ferry buffer, the packet still

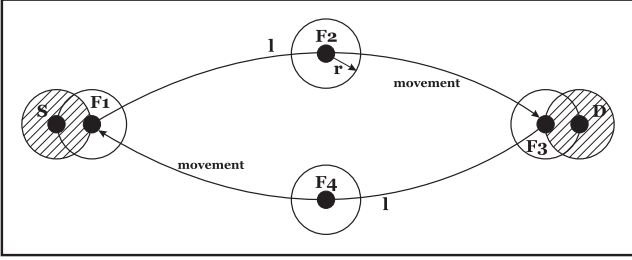


Figure 3: The Conveyor Belt Model

has to wait until all packets are loaded and the ferry is ready to depart.

$$t_{loading} = \frac{b}{R}$$

iii) *ferry transit delay* to destination. There will be no offloading delay since the packet is the first in the FIFO queue and can be handed to the application immediately.

$$t_{transit} = \frac{l - 2r}{v}$$

(b) The Last-on Case (best-case delay) A packet in the last-on case experiences the transit delay plus the offloading delay at the destination:

$$t_{best} = \frac{l - 2r}{v} + \frac{b}{R}$$

The average delay of packets from source to destination is calculated as the average of best case and worst case delays:

$$delay^{conveyor} = \left(1 + \frac{1}{n}\right) \frac{l - 2r}{v} + \left(\frac{1}{2} + \frac{1}{n}\right) \frac{b}{R} \quad (5)$$

2.4 Evaluation of proposed route designs

Each of our proposed route designs yields benefits over the other, depending on the environment it is deployed in. The following section compares both schemes in regards to throughput and packet delay. Environment variables include ferry speed v , ferry buffer size b , number of ferries n , data rate R , transmission radius r , and separation distance between sender and destination l .

Throughput

In the case of $n = 1$ ferries, both throughputs are equal, i.e.,

$$T = \frac{1}{\frac{2(l-2r)}{v} + \frac{2}{R}}, \quad (6)$$

which validates the model analysis.

For $v \rightarrow \infty$ or $b \rightarrow \infty$ the chain model throughput becomes $\frac{R}{2}$ and the conveyor belt model yields $\frac{n \cdot R}{2}$.³ Ferries with high buffer size can deliver more data per cycle in the conveyor belt model, since there are no intermediate buffer exchanges like in the chain-relay model. However, when using a fast radio, i.e., $R \rightarrow \infty$, the chain-relay model outperforms the conveyor-belt.

If communication range for each node is reduced, i.e., $r \rightarrow 0$, the conveyor belt model yields higher throughput. This suggests that when limited to short-range communication radios like Bluetooth, the chain-relay model should not be used.

The number of ferries such that $T = \frac{R}{2}$, which is the maximum data rate for the chain-relay model, is less for the conveyor belt model if $v > \frac{Rr}{b}$.

Delay

For $R \rightarrow \infty$ the chain model yields lower delay than the conveyor belt. When dealing with fast planes, i.e., $v \rightarrow \infty$, the conveyor belt model yields lower delay.

In summary, the conveyor belt model is superior with fast ferries, whereas the chain-relay model should be deployed with high data rate or long-range radios.

3 Algorithm Design

The previous section analyzed two models of possible ferry movement between one task node communicating to one data sink. Now we evaluate an algorithm which assigns ferry time to nodes in the case of multiple task nodes transmitting to one data sink.

3.1 Full Knowledge

The following scenario exemplifies the use of a single ferry to serve as communication enabler in a centralized, client-server star network. k stationary task nodes send traffic at their individual, but time-invariant, rates to a central server outside their coverage area. The ferry connects clients and server in a round-robin fashion, i.e., nodes are served one after the other. The ferry moves between client and server according to the conveyor-belt model. We assume full knowledge of the network parameters, including traffic flow rate f_i for each node i , distances d_i from node to server, maximum wireless data rate R at each node, buffer size b and constant speed v of the ferry. The ferry's radio can match any node's data rate, and the time for one buffer exchange is $\frac{b}{R}$. There is always a full buffer transmitted.

³ $n_{opt} = 2$ in this case.

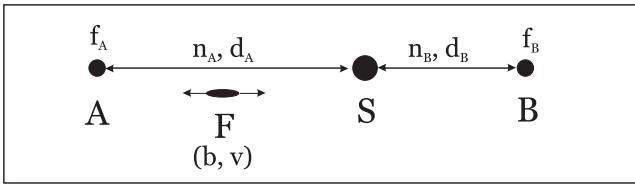


Figure 4: Problem notation: One ferry serving two nodes

The goal is to minimize the average packet delay in the network. The parameter to optimize for is the fraction of the total cycle time n_i that the ferry cycles between a given node i and the server before going on to serve the next node. A ferry can, for instance, spend more time serving nodes with higher flow rate requirement, than other nodes. Figure 4 shows the network setup with $k = 2$ nodes.

The objective function for packet delay in the network can be derived analogous to the conveyor-belt model by averaging over best case and worst case packet delay for each node. To consider all packets in the network, we further need to sum the delays over all nodes. Let each node's serving time, i.e., the total time it takes the ferry to cycle between node and server, be $n_i(2\frac{d_i}{v} + 2\frac{b}{R}) = n_i\tau_i$. The total cycle time is $T_{cycle} = \sum_i n_i\tau_i$.

A packet experiences the shortest delay when it gets loaded onto the ferry last, i.e., $\frac{d_j}{v} + \frac{b}{R_j}$. If a packet arrives just after a ferry has departed, it incurs the longest delay, i.e., the cycle time to serve all other nodes, plus its own serving time. This delay is given by $3\frac{d_j}{v} + 2\frac{b}{R_j} + \sum_{i, i \neq j} n_i\tau_i$.

Averaging over both these delays yields the objective function:

$$d = \sum_{j=1}^k \left(2\frac{d_j}{v} + \frac{3}{2}\frac{b}{R_j} + \frac{1}{2} \sum_{i=1; i \neq j}^k n_i\tau_i \right) \quad (7)$$

The first two terms of (7), which do not depend on n_i , are positive and constant by definition, thus they would only add to the total delay. For simplicity, these terms can be neglected without affecting the solution, and (7) simplifies to

$$d = \frac{k-1}{2} T_{cycle}.$$

Note that this model weighs the delays equally by nodes. An alternative would weigh each node's delay by its traffic volume (i.e., its flow rate f_i). It turns out that this does not affect the results and so will not be used for simplicity.

To meet the traffic demand for each node, the following flow constraints need to hold.

$$f_i \cdot T_{cycle} \leq n_i \cdot b \quad \forall i \in \{1, \dots, k\}, \quad (8)$$

$$n_i \geq 0. \quad (9)$$

Note that if the constraints are satisfied for any $\mathbf{n} = (n_1, n_2, \dots, n_k)$ then they can be satisfied for any scaled version of \mathbf{n} . Similarly the delay can be minimized by scaling \mathbf{n} so that $n_i \rightarrow 0$. Therefore for concreteness we let

$$\sum_{i=1}^k n_i\tau_i = T_{cycle} = 1. \quad (10)$$

Substituting (10) in (8) leads to

$$n_i \geq \frac{f_i}{b}. \quad (11)$$

Substituting (11) in (10) gives

$$\sum_{i=1}^k \left(\frac{f_i d_i}{v b} + \frac{f_i}{R_i} \right) \leq \frac{1}{2}. \quad (12)$$

Now even if $v \rightarrow \infty$, buffers need to be transmitted twice and (12) reduces to $\sum_i f_i \leq \frac{R}{2}$. If the data rate approaches ∞ the ferry still needs to fly back and forth twice to deliver data and (12) reduces to a constraint on flying time.

The fraction of time spent serving each node just depends on this node's flow requirement. The separation distance of node and server is not directly relevant. If only a few packets are transmitted from the node, the ferry visits this node less frequently, but instead delivers more packets on links with higher traffic flow.

Thus we have the following algorithm. First, the constraint in (12) is evaluated. If it is satisfied then the ferry can carry all flows successfully. Next the n_i are normalized to $n'_i = \frac{n_i}{\sum n_i}$ so that n'_i represents the fraction of time each node is visited. A simple stochastic algorithm would choose the next node to visit according to the probabilities $\{n'_i\}$. A deterministic algorithm would use a fractional visit counter c_i for each node to track visits.

1. Set $c_i = 0 \quad \forall i$.
2. While $\max\{c_i\} < 1$ set $c_i = c_i + n'_i \quad \forall i$.
3. Let $j = \arg \max_i \{c_i\}$.
4. Visit node j and set $c_j = c_j - 1$.

5. When the ferry visit is finished, go to 2.

This specifies a fair algorithm in the sense that if all n_i are equal then each node will be visited once per cycle. If traffic was in the opposite direction from the central hub to the task nodes, then the scheduling would be trivial. The ferry would choose the task node with the most data waiting. This strategy is equivalent to the above deterministic algorithm.

3.2 Variable Data Rate Communication

All aforementioned approaches to determining the best location or route for a ferry were based on a simple communication model. In this model two nodes are able to communicate if the separation distance between them does not exceed a certain threshold. Once this distance threshold is exceeded, direct communication between the nodes is assumed impossible.

The model in reality is richer than this. Many digital wireless interfaces such as IEEE 802.11 have multiple data rates. When communicators are close the communication rate is high (e.g. 54 Mbps in 802.11g). When communicators are far the communication rate is lower (802.11g has 14 different rates between 54 and 1 Mbps). The rate decision is based on signal power to noise power ratio (SNR) and packet success measurements made over the channel. This phenomenon is an example of a general principle. The well known Hartley-Shannon Law states that the channel capacity C , which is the theoretical maximum rate at which information passes error free over a channel, is equal to

$$C = W \log_2(1 + \text{SNR}), \quad (13)$$

where W denotes the channel bandwidth. Depending on channel coding, actual data rates can be pushed very close to this theoretical maximum. The assumption of a wireless propagation model which links distance and SNR in the fashion $\text{SNR} = \frac{k}{d^\epsilon}$ leads to a relationship between data rate and distance

$$C = W \log_2\left(1 + \frac{k}{d^\epsilon}\right), \quad (14)$$

where k is a constant, and ϵ the path-loss exponent, usually between 2 and 4. The above equation turns the discrete data rate assumption in earlier works into a continuous function. Hereby, there exists an obvious trade-off between data rate and distance, which can be leveraged for providing improved connectivity of multiple nodes through ferry placement. The idea is that the ferry can exploit communication while moving between multiple nodes in order to more quickly transfer data.

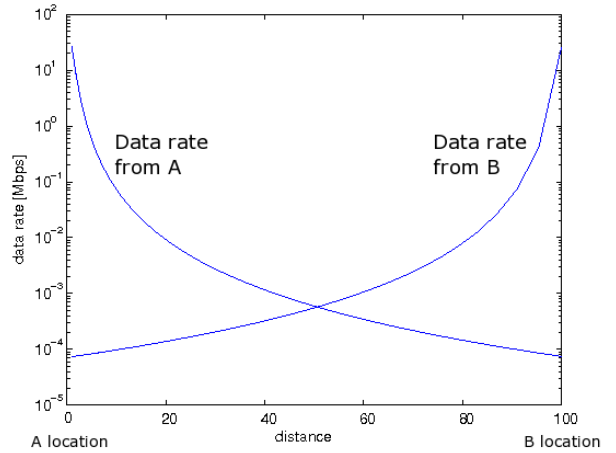


Figure 5: The possible data rates between ferry and nodes A and B as a function of ferry separation distance from A.

Figure 5 shows the achievable data rate between a ferry and node as a function of their separation distance for two nodes A and B.

In this paper we are concerned with the region when the ferries and nodes are relatively widely spaced. In this regime, $\text{SNR} \ll 1$ and we can approximate the Shannon capacity as:

$$R(d) = 1.44W \frac{k}{d^\epsilon} = R_0 \frac{d_0^\epsilon}{d^\epsilon} \quad (15)$$

where R_0 is the bandwidth at a reference location d_0 .

Scenario 1: One Single Ferry

Consider a node sending data to a sink. The nodes are separated by distance l . The optimal strategy for a single ferry is for the ferry to start at the midpoint, fly toward the node until its buffer is half full, then fly away from the node. Upon reaching the midpoint, its buffer will be full. Continuing to fly toward the sink, it will start to unload its buffer, until it has unloaded half of its buffer, then fly away from the sink. Upon reaching the midpoint, the rest of the buffer will have been unloaded and the transfer will be complete. In this way, the plane only moves as close to a node as is necessary to complete a transfer.

Consider two nodes separated by distance l . When a ferry is at distance y from a node and flying at velocity v , as it moves over a distance dy , it will relay $\frac{R(y)dy}{v}$ bits. So moving from the midpoint to within x of the node it will load:

$$\begin{aligned}
B(x) &= \int_x^{l/2} \frac{R_0 d_0^\epsilon}{v y^\epsilon} dy = \frac{R_0 d_0^\epsilon}{v(\epsilon-1)} \left(\frac{1}{x^{\epsilon-1}} - \frac{1}{(l/2)^{\epsilon-1}} \right) \\
&= \frac{R_0 d_0^\epsilon}{v(\epsilon-1)x^{\epsilon-1}} \left(1 - \left(\frac{2x}{l} \right)^{\epsilon-1} \right)
\end{aligned}$$

Setting $B(x) = b/2$, setting $d_0 = l/2$ and solving for x yields;

$$x = \frac{l}{2} \left(\frac{1}{1 + \frac{b(\epsilon-1)v}{R_0 l}} \right)^{\frac{1}{\epsilon-1}}$$

The cycle time is then

$$\begin{aligned}
T_{cycle} &= 4(l/2 - x)/v \\
&= \frac{2l}{v} \left(1 - \left(\frac{1}{1 + \frac{b(\epsilon-1)v}{R_0 l}} \right)^{\frac{1}{\epsilon-1}} \right)
\end{aligned}$$

The time $2l/v$ is the total time for the vehicle to fly from one node to the other and back. The quantity in the brackets is the fractional savings in time because of the transferred data enroute. To better see what is going on, we compute the throughput for the special case of free-space path loss, $\epsilon = 2$:

$$T = \frac{b}{T_{cycle}} = \frac{R_0}{2} + \frac{v \cdot b}{2l} \quad (16)$$

In this case we see that if v or b are large or R_0 or l are small then the throughput is one buffer per round-trip flight between nodes, $\frac{v \cdot b}{2l}$. If v or b are small or R_0 or l are large then the throughput is half the rate at the midpoint, $\frac{R_0}{2}$.

It would be instructive to compare this to the fixed radius communication model. In this model the data rate is R at radius r . Substituting this into the rate model yields $R_0 = Rr^\epsilon/(l/2)^\epsilon$ and:

$$T = \frac{2Rr^2}{l^2} + \frac{v \cdot b}{2l} \quad (17)$$

Let T_{vr} be the above throughput with variable rates, and T_{fr} be the fixed radius throughput in (6). We note that $T_{fr} \leq R/2$, while T_{vr} is unbounded as v or b increases. In the best case, when $l = 2r$, $T_{fr} = R/2$ which is less than $T_{vr} = R/2 + v \cdot b/(4r)$. While T_{vr} is generally superior, for large l they both have similar throughput of about $v \cdot b/(2l)$.

Scenario 2: Multiple Ferries

The superiority of the variable rate model is more pronounced in the case of multiple ferries. In the conveyor

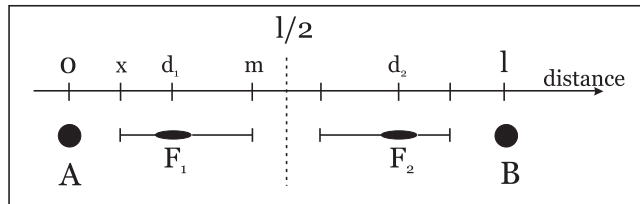


Figure 6: Multiple ferry scenario with $n = 2$ ferries.

belt, more ferries can always be added by passing closer to the source and destination nodes where the rates are higher. In the chain relay, the effective l is shortened with each additional ferry and the communication is more into the regime where the variable rate method dominates. The conveyor belt model is trivial, so we will look at multiple ferries in the chain relay model.

We assume n ferries which serve evenly spaced division points between communication points A and B . Figure 6 shows the two ferry scenario. The ferry operation dictates that the gap between two adjacent ferries is being closed at twice the speed as the gap between a fixed endpoint and a ferry. However, throughput goes down. Looking at $n = 2$ ferries, we see that $d_2 - d_1 = 2d_1 = 2(l - d_2)$. Substituting $2l$ for l , $2v$ for v , and $R_0 = \frac{Rr^\epsilon}{(l/2)^\epsilon}$ in (16) this fact becomes obvious. Now, with n ferries the approximate throughput between adjacent ferries is given by substituting $v = 2v$ and $l = \frac{2l}{n}$ in (17), resulting in

$$T_{vr} = n^2 \frac{Rr^2}{2l^2} + n \frac{v \cdot b}{2l}.$$

So we see that throughput scales quadratic with the number of ferries serving the endpoints⁴.

4 Related Work

Li and Rus [4] are one of the first to address deliberate trajectory changes to make message sending between two communicating nodes possible. Two algorithms are presented - one for full knowledge of node motions the other under partial knowledge. Each algorithm results in the local update of a node's trajectory. Errors in the estimation of neighbor node locations are studied, and the algorithms are simulated. Special attention is paid to the influence of the node's message relaying duty on the primary task of each node. They use an idealized propagation model, and have no consideration of multiple flows at a time.

⁴Recall this is the case where $\frac{l}{n}$ is large and (15) applies.

The first notion of message ferrying was developed by Zhao [7] who looks at routing of messages in partitioned wireless ad-hoc networks, where it is infeasible to route messages along pre-established routes. Instead, messages are *carried* by so-called “ferries” until they reach their respective destinations. Known traffic patterns are exploited to form efficient ferry routes that minimize delays and support bandwidth requirements. While at first looking at stationary ground nodes, in [8] the authors develop two algorithms for ferry route design in the mobile scenario. One ferry services all nodes. Nodes or the ferry can initiate the messaging scheme. Either the ferry location/trajectory is assumed to be known or the nodes are equipped with long range radios to notify the ferry of their locations. This way, the authors circumvent the problem of finding nodes and ferries. In [9] they extend the route design problem to include multiple ferries. Buffer constraints are not considered. Simulations are done in the network simulator ns-2. All path planning algorithms build upon the Traveling Salesman Approach.

Unlike our approach, the TSP solution will always find an optimal cycle path among all nodes. Thus every node is visited once per cycle. In the simple case of one outlying node with a low flowrate this will limit the network capacity and increase average packet delay in the network. The node will be visited in every cycle regardless of its location or flow requirement.

Shah et al. [6] deal with the problem of sensor data collection in sparse sensor networks. So-called MULEs pick up the data when in close range to the data-gathering nodes, buffer it for some time, and drop it off at wired access points. The main advantage is the power savings in the nodes, since only short-range transmission of data is necessary. An analytic model of the system is introduced, and scaling with regards to number of mules, sensors and access points is examined. Results are meant to provide guidelines for deployment of similar systems.

Merugu et al. [5] look at identifying recurring ‘paths over time’ in highly partitioned networks with explicit node movements. In these *store, carry, and forward* networks it sometimes is more efficient to carry data instead of forwarding it to the closest neighbor. Their work is based on an assumed periodicity of movement in most mobile nodes. The result is a space-time routing framework which relies on space-time routing tables where the next hop is selected from current as well as future neighbors. An algorithm for creating the routing tables as to minimize delay is introduced and empirically evaluated using simulations.

Chatzigiannakis et al. [3] introduce the notion of the

support of the network, which refers to a subset of mobile nodes dedicated to message relaying. The movement of the support is controlled by a subprotocol. Two possible motion models are proposed: (a) the *snake* protocol, where support nodes stay pairwise adjacent at all times, and (b) the *runners* protocol, where each support node performs an independent random walk on a motion graph. It is shown that basic message passing can be improved especially in highly dynamic environments. The drawback of the approach is a limitation of the support movement along an abstract graph, which does not relate to real-world coordinates. Further, nodes in the runners protocol perform random movements which can not guarantee expedited delivery of high-priority messages.

5 Conclusion and Future Work

This paper analyzes two novel ways of designing ferry routes in a delay-tolerant network, namely the chain-relay model and the conveyor belt model. It is shown that each of the models has its merits, depending on the environment it is to be deployed in. A simple client-server star network, where one ferry serves multiple nodes according to the conveyor belt model, is studied and a delay-optimal scheduling algorithm is derived. Further, a new problem framework for data ferrying in delay-tolerant networks is described. It is argued that viewing the communication data rate as a continuous function of node separation distance can greatly enhance network performance. The design of mobility algorithms as well as transmission protocols within this framework will be the focus of future work.

References

- [1] T. X. Brown, B. Argrow, C. Dixon, S. Doshi, R.-G. Thekkekkunnel, and D. Henkel. Ad hoc uav-ground network (augnet). In *AIAA 3rd Unmanned Unlimited Technical Conference*, Chicago, IL, September 2004.
- [2] T. X. Brown, S. Doshi, S. Jadhav, D. Henkel, and R.-G. Thekkekkunnel. A full scale wireless ad hoc network test bed. In *Proceedings of ISART’05*, NTIA Special Publications SP-05-418, pages 51–60, March 2005.
- [3] I. Chatzigiannakis, S. Nikolettseas, and P. Spirakis. Experimental evaluation of basic communication for ad-hoc mobile networks. In *Proc. of the 5th Annual Workshop on Algorithmic Engineering - WAE’01*, volume 2141 of *Springer Lecture Notes in Computer Science*, pages 159–171, 2001.

- [4] Q. Li and D. Rus. Sending messages to mobile users in disconnected ad-hoc wireless networks. In *Mobile Computing and Networking (Mobicom)*, pages 44–55, 2000.
- [5] S. Merugu, M. Ammar, and E. Zegura. Routing in space and time in networks with predictable mobility. Technical Report GIT-CC-04-07, Georgia Institute of Technology, March 2004.
- [6] R. C. Shah, S. Roy, S. Jain, and W. Brunette. Data mules: Modelling and analysis of a three-tier architecture for sparse sensor networks. *Elsevier Ad Hoc Networks Journal*, (1):215–233, Sept. 2003.
- [7] W. Zhao and M. Ammar. Message ferrying: Proactive routing in highly-partitioned wireless ad hoc networks. In *the 9th IEEE International Workshop on Future Trends of Distributed Computing Systems*, May 2003.
- [8] W. Zhao, M. Ammar, and E. Zegura. A message ferrying approach for data delivery in sparse mobile ad hoc networks. In *MobiHoc'04*, May 24-26 2004.
- [9] W. Zhao, M. Ammar, and E. Zegura. Controlling the mobility of multiple data transport ferries in a delay-tolerant network. In *Proc. of InfoCom 2005*, 2005.