Motion-based Routing for Opportunistic Ad-hoc Networks

Weihan Wang
Department of Electrical and Computer Engineering
University of Toronto
weihan@eecg.toronto.edu

Cristiana Amza
Department of Electrical and Computer Engineering
University of Toronto
amza@eecg.toronto.edu

ABSTRACT

In this paper, we introduce novel motion-based routing protocols for opportunistic packet relay in ad-hoc opportunistic networks. Unlike existing ad-hoc routing protocols, which require global knowledge about the mobility patterns of other nodes, thus raising scalability and privacy concerns, our routing protocols store and use only local motion information, such as the current direction of the local node. Such information is easily obtainable by any modern GPS-equipped mobile device.

We design and evaluate a series of protocols using different motion information, ranging from simple, easy to capture metrics i.e., speed and direction, to more complex metrics, i.e., the past or expected trajectory of the local node. These metrics allow for increasing degrees of routing accuracy at a correspondingly higher cost in terms of memory and computation. However, all of our routing protocols have constant complexity in the size of the network, and none of them rely on nodes sending or storing information about other forwarding nodes. This makes our schemes ideal for large-scale networks and non-community networks, where membership is very dynamic.

Our simulation study using real-world traces shows that our motion-based schemes perform comparably to or considerably better than existing complex, global-knowledge-based protocols with significant less utilization on memory, CPU, and bandwidth.

Categories and Subject Descriptors

C.5.3 [**Computer System Implementation**]: Portable devices; C.2.2 [**Computer - Communication Networks**]: Network protocols—*routing protocols*

General Terms

Simulation, Modeling, Performance

MSWiM'11, October 31–November 4, 2011, Miami, Florida, USA.

Keywords

Mobile Computing, Opportunistic Ad-hoc Networks, Routing Protocol

1. INTRODUCTION

Opportunistic ad-hoc networks [7, 29], support wireless communication within some type of spontaneously forming, self-organizing networks of mobile users. In such opportunistic, localized networks, usually some type of specialized wireless communication is either desirable or absolutely necessary. For example, communication may be performed opportunistically on Bluetooth because of its low-power consumption and availability at short range. Alternatively, Bluetooth, or some other specialized wireless technologies may be used together with a certain type of vehicle, e.g., cars or buses, or in regional networks unreachable by standard telecommunication technologies.

In these opportunistic ad-hoc networks, the mobile users themselves are commonly contributors to the network infrastructure as packet forwarders on behalf of other nodes. Mobile devices relay packets to neighboring devices as they come in contact with them, by taking advantage of communication opportunities that arise in the course of user mobility, until the packet eventually reaches its destination. Since mobile devices become increasingly pervasive, almost anyone with a Bluetooth device in her or his pocket becomes a potential participant in the forwarding process. We call these type of routing protocols based on mobility opportunities delay-tolerant routing (DTR).

In order for DTR to be widely accepted and deployed, two pre-requisites must be met for mobile devices involved in packet transfers: i) users should be required to reveal little private information and ii) resource consumption, for message transfers, computation and memory, required by the routing protocol should be kept as low as possible.

Many existing DTR protocols rely on revealing and aggregating detailed private statistics to other nodes, including meeting records, connection histories, and even details of buffered packets [6, 19, 24, 2, 22, 4, 15], which might raise privacy concerns. Furthermore, in order to calculate routes, these protocols commonly require each node to store the learned topology of other nodes in the whole system, such as the link state and meeting history. Thus, the cost of such DTR algorithms is $O(\sqrt{n})$ at best, which makes these solutions unsuitable for memory constrained devices or large-scale networks. In addition, computation and dissemination of such global information may consume considerable power and bandwidth, often crucial resources in mobile environ-

ments. Finally, these protocols require continuous observation, and some even rely on a priori knowledge, of other nodes to learn system topologies. Hence, they work well only for *community networks*, where membership is static and nodes are coordinated. In contrast, in *non-community networks*, such as, vehicles on a highway, or people in a shopping center, where the ad-hoc network is organized on-the-fly, and membership is very dynamic, or cannot be accurately established, these existing protocols may not work efficiently, or at all.

In this paper, we observe that, in addition to Bluetooth, GPS-capabilities are becoming mainstream on mobile devices; various alternative techniques for outdoor and indoor positioning, including WiFi and RFID localization and GSM fingerprinting, are also maturing [12, 17, 32, 25]. Based on this observation, our key idea is to use information about node movement in order to compute the *utility*, which reflects the node's probability of delivering any given packet to its known destination. Intuitively, nodes moving towards the intended packet destination are likely to successfully deliver the particular packet. Moreover, frequent visitors to the destination's neighborhood have a high probability to deliver the respective packet as well. We therefore propose a series of motion-based routing protocols using different utility functions, corresponding to the above intuitions: i) velocity-based protocols and ii) trajectory-based protocols that infer utilities from either velocity or past/expected trajectories, respectively. Each protocol in the above categories exploits increasingly more metrics, allowing for varying levels of trade-off between routing accuracy and costs in terms of memory and computation.

All the proposed protocols compute utilities locally and require no motion data about other forwarding nodes. This provides three advantages over existing DTR protocols. First, the complexity of our algorithms is *constant* with respect to the size of the network, making our protocols scalable and viable for very large networks, such as an ad-hoc network spanning the mobile users in a whole city. Second, not requiring storing and using information about other nodes makes our protocols ideal for non-community networks. Third, privacy is better preserved than previous approaches as a side-effect of localized computation.

We run simulations using two real-world traces, one consisting of more than 30 buses and the other with 300 campus wireless users, and compare our protocols with two existing global-knowledge-based DTR protocols: *MV* and *RAPID*. They are chosen as they are supposed to outperform protocols that exploit less knowledge. *MV* [6] is a classical protocol using global meeting histories to compute probabilities of delivery. *RAPID* [2] exchanges meeting histories and also metadata about each packet to estimate per-packet delay. According to previous work [2], *RAPID* significantly outperforms three existing protocols including *MaxProp* [4], *Spray and Wait* [28], and *PRoPHET* [22].

We adopt a novel metric, called *Relative CDF*, which we believe is more accurate than the traditional combination of delivery ratio and delay, as well as the more traditional metrics, to compare the performance of our protocols with the above state-of-the-art protocols. We also compare resource utilization of all protocols.

Our study show that our algorithms are resilient to inaccurate location information. With orders of magnitude lower costs in various computing resources, our algorithms perform comparable to *RAPID* and significantly better than *MV*, even in community networks, which is the environment providing the most fair comparison since they have bounded network sizes and allow global statistics to be accumulated, which are necessary for global-knowledge-based protocols such as *RAPID* and *MV* to work efficiently.

The rest of this paper is organized as follows. § 2 describes the system model and protocol design. § 3 presents the utility functions we use, the key component of our protocols. § 4 discusses privacy and other issues. § 5 presents evaluation methodology and results. Related work is summarized in § 6. § 7 concludes the paper and presents directions for future work.

2. PROTOCOL OVERVIEW

2.1 System model and assumptions

We model an opportunistic ad-hoc network as a dynamic set of mobile nodes. Nodes may join and leave the network at any time. Communications are based on pair-wise contacts: two nodes in the network are called *neighbors* and can transfer data packets bidirectionally if within communication range. Each packet has a single destination node and can be delivered from the source to the destination directly or via intermediate nodes. Naming and addressing for mobile nodes is an orthogonal issue beyond the scope of this paper; therefore, as others before us e.g., Burns et al. [6], we assume that the address for the destination of a packet can be obtained by orthogonal means.

For example, destinations for packets could be stationary nodes, part of an infrastructure which can be assumed to be always on and whose addresses, and locations can be easily obtained. Specifically, in many applications, destinations are knwon data collectors in sensor networks [27], gateways to the Internet [11], or packet relays to out-of-band channels. Finally, much like geocasting [14], applications leveraging our motion-based routing may transmit packets towards a specific region and then broadcast them within that region.

We also assume that nodes always know their own locations through localization systems. The buffer size on each node dedicated to packet forwarding is limited and when the buffer is full, old packets have to be dropped in order to receive new packets. Network bandwidth is also limited; the amount of bytes that can be transmitted during a contact is finite and proportional to the duration of the contact. Therefore, protocols must wisely transmit important data first.

2.2 Protocol operations

Our protocols operate as follows. When a node A meets another node B, A first delivers the packets that are destined to B, and then exchanges metadata to compute the *utility function* for the packets remained in its buffer. The utility function $U(n, d)$ of node n w.r.t. node d is defined to reflect the protocol's belief on n's probability to visit d's neighborhood region in the near future. Node A sorts the packets in decreasing order of *marginal utilities* gained by forwarding packets to B, i.e. $U(B, d) - U(A, d)$, with d the destination of a packet in question. All packets with *positive* marginal utilities are transferred from A to B in that order and removed from A's buffer. Transmission ends when all these packets have been forwarded or the transfer opportunity ends.

When node B receives a forwarded packet destined to d, it delivers the packet if d is among its immediate neighbors. Otherwise, it calculates $U(n,d)$ for every neighbor n and forwards the packet to a neighbor with the highest utility and which is higher than $U(B,d)$. This neighbor in turn performs the same operation. Such iterative process is necessary to pass packets on to the best node in the same cloud as, but not directly neighboring, the initiating node. The process continues until the receiving node finds no better candidate than itself. In this case, the node stores the packet in its own buffer, and when the buffer is full, drops the oldest packet, i.e. the packet with the longest time since creation.

Only one copy for each packet is kept in the network. Alternatively, nodes can always retain the packet after forwarding it to other nodes, or retain it only when observing from various facts that the probability of delivery is too low. For example, the packet's maximum utility among neighbors is below a threshold, or the node has not met others for a while indicating a sparse neighborhood network. In § 6, we summarize existing replication as well as buffer management techniques directly applicable to our protocols.

3. THE UTILITY FUNCTION

Information about a node's movement, such as, its current direction or the locations it visited in the past, can be used to predict the regions the node is going to visit. Once it is in vicinity with another node, it is likely to deliver packets to the latter using direct contacts or through intermediate nodes between them. Hence, the utility function $U(n,d)$ uses n's motion information to computes a *utility* which reflects n's probability of visiting d's neighborhood region in the near future, and therefore the likelihood of n delivering packets to d. We propose several utility functions using different motion information, each realizing a distinct motion-based protocol. *Velocity-based functions* compute utilities from n's current velocity, and *trajectory-based functions* use either n's past or expected trajectory. These functions exploit increasingly more information about n's movement for better routing performance at higher costs in computation and storage. Note that the functions support both 2D coordinates and 3D coordinates as in underwater and interplanetary networks.

None of the utility functions rely on information about *any* other forwarding nodes. The only information they require besides n's own movement is the destination's *current* geographical location, if the destination is mobile. Ideally, a packet should move towards the location where it meets the destination at the time of delivery but not where the destination dwells at present. However, this requires estimating packet delay and predicting locations based on the delay which involves non-trivial computation and potential inaccuracy.

3.1 Velocity-based utility functions

A node's current velocity is a hint to its future movement: the higher speed and less yaw of a node approaching the other node, the more likely it is to reach the latter in the imminent future. We propose three velocity-based utility functions. By convention, we refer to *velocity* as a vector whose magnitude is *speed*. Illustrated in Figure 1, we denote node n's velocity as $\vec{v}_n$ or $\vec{v}$ when used unambiguously, the vector from n to node d's current location as $\vec{d}_{n,d}$ or $\vec{d}$,

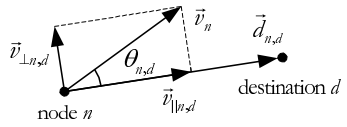

Figure 1: Notations. We omit subscripts when used unambiguously.

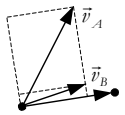

Figure 2: Node A and B have the same $\vec{v}_\parallel$ but B has a lower $\vec{v}_\perp$.

and the angle between $\vec{v}$ and $\vec{d}$ as $\theta_{n,d}$ or θ. In practice, $\vec{v}$ can be always estimated from most recent readings from n's localization system.

3.1.1 Velocity projection ($U_{\texttt{vp}}$)

It is defined as the projection of $\vec{v}$ on $\vec{d}$. Hence, a node moving faster along $\vec{d}$ gains a higher utility. Note that the utility becomes negative when moving against the destination.

$$U_{\texttt{vp}}(n,d) = |\vec{v}| \cdot \cos\theta$$

3.1.2 Velocity cotangent ($U_{\texttt{vcot}}$)

$U_{\texttt{vp}}$ ignores $\vec{v}_\perp$, $\vec{v}$'s perpendicular components to $\vec{d}$. However, as illustrated in Figure 2, when two nodes have identical $\vec{v}_\parallel$ but different $\vec{v}_\perp$, the one with a smaller $|\vec{v}_\perp|$ is more likely to reach d. A better utility function shall increase as $|\vec{v}_\perp|$ decreases given the same velocity projection. As one of many approaches, we divide $U_{\texttt{vp}}$ by $\sin\theta$, which is equivalent to multiplying $|\vec{v}|$ and $\cot\theta$. In the final equation, we use the absolute value of $\sin\theta$ to guarantee a positive utility when moving towards the destination and negative otherwise. We avoid an infinite utility by keeping the denominator above a small constant ε.

$$U_{\texttt{vcot}}(n,d) = |\vec{v}| \cdot \frac{\cos\theta}{\max(|\sin\theta|, \varepsilon)}$$

3.1.3 Estimated time to arrive ($U_{\texttt{eta}}$)

The distance to the destination also contribute to the probability as well as the delay of packet delivery. We compute $U_{\texttt{eta}}$ as the reciprocal of the estimated travel time to the destination's radio coverage based on n's speed along $\vec{d}$, and similar to $U_{\texttt{vcot}}$, divide the result by $\max(|\sin\theta|, \varepsilon)$. Hence, the final equation is equivalent to dividing $U_{\texttt{vcot}}$ by the distance. As the time to arrive would make no sense if the node is departing from the destination (i.e. $\cos\theta \leq 0$), we simply use $U_{\texttt{vcot}}$ in this case. Assuming the radius of radio range is R, the utility function is defined as follows:

$$U_{\texttt{eta}}(n,d) = \begin{cases} \dfrac{U_{\texttt{vcot}}(n,d)}{|\vec{d}| - R}, & \cos\theta > 0; \\ U_{\texttt{vcot}}(n,d), & \text{otherwise.} \end{cases} \quad (1)$$

3.2 Trajectory-based utility functions ($U_{\texttt{tp}}$ and $U_{\texttt{tf}}$)

Either past or future trajectories can be used to estimate probabilities of delivery. We denote the utility function based on history trajectories as $U_{\texttt{tp}}$. Leguay et al. point out that users' mobility pattern is not uniformly random but

can be often characterized by power-law distributions [19]. That is, a user visits few places frequently with long stay while visiting other places rarely with short stay. If a node's past trajectory is known, we can therefore expect that its future movement follows similar patterns, and use this data to predict the probability of reaching specified regions in the future. However, this approach would be disadvantageous if mobility patterns are uniform or changing quickly over time.

Future trajectories can be accurately obtained if node schedules are known in advance. This is the case for most public transit systems. Additional information like traveling history and road condition forecasts may help refining the information. We refer to the utility function using future trajectories as $U_{\texttt{tf}}$.

We assume that nodes store trajectories as a time series of sample points sampled at a fixed time interval t. Depending on the pace the system evolves, the interval can be as short as one second or as long as several days. We define the *individual utility* of each sample point to be negatively correlated to its distance to the destination. Sample points within the destination's radio range are more advantageous than other points because direct delivery is guaranteed when the node is at these points. We therefore define the individual utility of node n's ith sample point as

$$\hat{U}(n,d,i) = \begin{cases} \beta R^{-\lambda}, & |\vec{d}_{n_i,d}| < R; \\ |\vec{d}_{n_i,d}|^{-\lambda}, & \text{otherwise.} \end{cases} \quad (2)$$

where R is the radius of radio range, $|\vec{d}_{n_i,d}|$ is the sample point's distance to d, the exponent λ is a positive number controlling how fast the utility declines as distance increases, and the *boosting factor* β is a constant greater than one to encourage sample points within the radio range. The overall utility is then the weighted sum over all sample points:

$$U_{\texttt{tp},\texttt{tf}}(n,d) = \sum_{i=1}^{S} \omega(n,i)\hat{U}(n,d,i) \quad (3)$$

with S the total number of n's sample points and $w(n,i)$ the weight of the ith sample point reflecting the importance of the point. There are numerous methods to assign weights, which could be dependent on application semantics. For instance, heavier weights can be given to sample points lying in venues such as central stations and meeting rooms where users are likely to meet destination nodes or better candidates as packet carriers. In this paper, we employ *exponential moving average* to give higher importance to more recent sample points, as described below.

Supposing the sample point at index one is the earliest, we set $\omega(n,i) = \omega_0^{S-i}$ when using past trajectories. The constant ω_0 which is referred to as the *smoothing factor* satisfies $0 < \omega_0 < 1$, so that the smaller the value, the quicker old data discounts. The rationale behind this scheme is that more recent data predict future mobility patterns better than older data. When applied to future trajectories, the weight becomes $\omega(n,i) = \omega_0^i$ so that sample points in the immediate future are given more importance. It also makes sense in that timely delivery of packets counts more on node movement in the near future than in the far future. As a special case, setting $\omega_0 = 1$ will treat all sample points equally. Finally, The summation in Equation 3 ignores the sample points whose weight is below a threshold, in order to save computation as well as storage for insignificant sample points. We term this threshold as the *cutoff weight*.

In practice, trajectory-based functions incur higher costs than velocity-based ones in computation and storage, rendering them infeasible for extremely resource constrained devices. $U_{\texttt{tf}}$ is unavailable if no future schedule is known; $U_{\texttt{tp}}$ is also excluded if users are not willing to be tracked by their own devices for any reasons. We will study the performance and costs of these utility functions through experiments.

4. DISCUSSION

Because utilities are determined solely from local information (in addition to the destination's location), our protocols are highly scalable, with *constant* complexities in time, space, and communication with respect to the size of the network. In contrast, the complexities of existing DTR protocols grow at least $O(\sqrt{n})$ with the system size, making them infeasible for large-scale networks. The considerable amount of power, memory, and bandwidth consumed by these protocols also impose heavy burden to resource constrained mobile devices. However, DTR must keep resource utilization as low as possible in order to be widely accepted in today's mobile environments. Moreover, algorithms relying on global state require a priori knowledge about and continuous statistics over other nodes, which works well in community networks but not in non-community networks where the number of participants is unbounded and what a node meets most are strangers, making meeting histories and other global statistics unable to accumulate.

As a byproduct of localized computation, our protocols preserve privacy for forwarding nodes better than existing protocols whose computation is based on synthesis of private information about other nodes such as meeting histories and buffer state [6, 19, 24, 2, 22, 4, 15]. In our protocols, nodes compute their own utilities without information about other forwarding nodes, and the only peers that require a node A's utility values besides A itself are its immediate neighbors. When a neighbor, say B, requests A's utilities, rather than sending A's motion details to B, A only replies with a ready-made utility corresponding to a destination given by B. Nodes therefore exchange only high-level utility values rather than low-level private information. However, by repeatedly querying a node with well-crafted destination addresses, one can still infer private information through interpolation. This is an area for future exploration.

The above method of "delegating" utility computation also reduces bandwidth usage for trajectory-based protocols: only a single value rather than the whole set of sample points are transmitted. As another optimization, we can save repetitive computation and transmission by caching utilities for a short while on both sending and receiving nodes. If the underlying network supports broadcast, the third optimization is to broadcast when replying with utilities. Nodes receiving the broadcast may cache these values for later use.

5. EVALUATION

In this section, we first present a novel technique for network performance analysis (§ 5.1). After describing evaluation methodology (§ 5.2), we present evaluation results in the last five subsections, each answering one of the following questions:

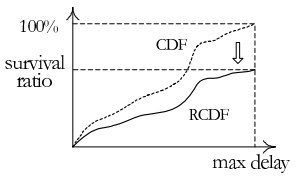

Figure 3: Converting CDF to RCDF

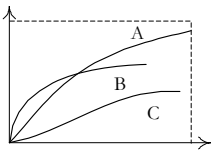

Figure 4: Comparing RCDF

- How to tune parameters for trajectory-based protocols?
- How do different protocols perform against each other?
- How sensitive are our protocols to localization errors?
- How do the protocols perform against existing ones?
- Do our protocols consume less resources than others?

5.1 Relative CDF (RCDF)

Two commonly used performance metrics for opportunistic ad-hoc networks are delivery ratio, i.e. the number of delivered packets versus the number of generated packets, and packet delay distribution, or one of its summaries like arithmetic mean or median. However, if two systems have different delivery ratios, their delay distributions are distorted and thus comparing delay distribution is biased. A system might attain worse delay distribution only because it delivers more packets. We also argue that delivery ratio is affected by delay and thus is not an independent metric. Given other factors being the same, a system with longer delay would suffer lower delivery ratio because of the time lag of the initial packets being delivered. In the following text, we describe a metric that we call Relative Cumulative Distribution Functions or RCDF to address these issues by defining *survival ratio* and combining it with delay distribution. RCDF is generally applicable to networks with packet loss.

Definition Let **A**, **D** and **I** respectively be the set of generated packets, delivered packets, and in-transit packets. Given the CDF $f(x)$ over packet delay x, the RCDF $f'(x) \equiv Sf(x)$, where the *survival ratio* S is a constant satisfying

$$\frac{|\mathbf{D}|}{|\mathbf{A}| - |\mathbf{I}|} \leq S \leq \frac{|\mathbf{D}| + |\mathbf{I}|}{|\mathbf{A}|}. \quad (4)$$

The idea of RCDF is to normalize delay distribution against all rather than received packets, and to use S to represent packets that are not lost. It is important to understand that in addition to delivered packets, *S also includes a portion of in-transit packets that will be successfully delivered in the future*. Figure 3 illustrates the transformation from CDF to RCDF.

In general cases, the exact value of S cannot be expressed in a closed form. It can be approximated as delivery ratio in some situations. Readers are referred to the appendix for the deduction of the bounds in Equation 4 and approximation. Figure 4 shows three hypothetic RCDF curves. The x-coordinate at the end of each curve corresponds to the system's maximum delay. System A and B is said to be absolutely advantageous over system C. Although the overall survival ratio of A is higher than B, B achieves more packets at the low-delay end. Whether A is better than B depends on end-user preferences. In this paper, we plot RCDF using mean of the two bounds as survival ratio. When comparing with previous work, we also include the traditional delivery ratio and delay metrics.

	diesel	*campus*
# mobile nodes	34	300
coverage	49×20 km	1.6×1.1 km
duration	80 days	45 days
radio range	10 m (Bluetooth Class 2)	
bandwidth	2.1 Mbps payload, duplex (Bluetooth 2.0)	
packet arrival	Exponential(1) min./packet/node	
packet size	1 KB	
buffer size	10 MB	
cutoff weight	1×10^{-6}	

Table 1: Default parameters for simulation

5.2 Methodology

We develop a custom discrete event simulator which, along with protocol implementations, is publicly available [1]. Since our interest is in comparing application-layer protocols, the simulator assumes ideal, lossless radio channels. Network parameters are set according to typical values in Bluetooth. Packets are continuously generated from each mobile node in a poisson process, with destinations being randomly selected fixed nodes. Table 1 summarizes the parameters used in our simulation.

We find two real-world, wireless mobility traces annotated with geographical information, namely the DieselNet trace and the Dartmouth trace. The DieselNet trace (*diesel*) [5] records bus movement around the UMass campus and the surrounding county. Each bus's log records the time and GPS coordinates when meeting other buses. We restore the bus's movement by linear interpolating these data points.

The Dartmouth campus trace (*campus*) [16] is an extensive dataset collected in the Wi-Fi network of Dartmouth College. It records over three years of mobility of around 13,000 wireless devices. Since running the whole trace is impossible on today's hardware, Leguay et al. [20] selected a subset of 45 days between January 26, 2004 to March 11, 2004 as users in this period "make an intensive and regular use of the network." The authors randomly chose 300 devices from this period for their study. We adopt the same subset in our paper. In *campus*, the log for each device records the names of the access points (APs) the device has been associated with, annotated with time. APs' geographic data is stored in a separate database. We restore the device's movement by assuming it is collocated with the AP at the beginning of each association, and linear interpolating these positions (APs with unknown locations are ignored). APs are dispersed among 185 buildings. Because our work focuses on outdoor communications where DTR is most useful, we avoid indoor networking which raises fundamentally different problems by moving all the APs in the same building to their geometric center.

Packets are continuously generated from each node in a poisson process. Packet destinations are randomly chosen points in the map which is passed through by at least one node during the simulation. Finally, we run each experi-

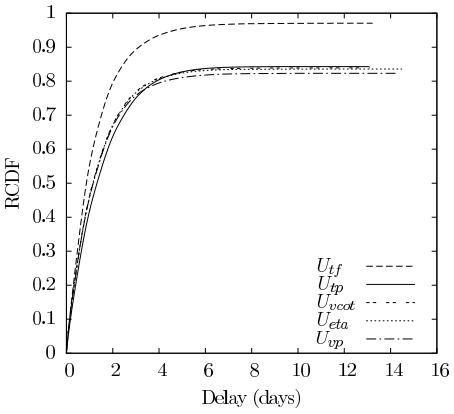

Figure 5: Comparing utility functions in *diesel*

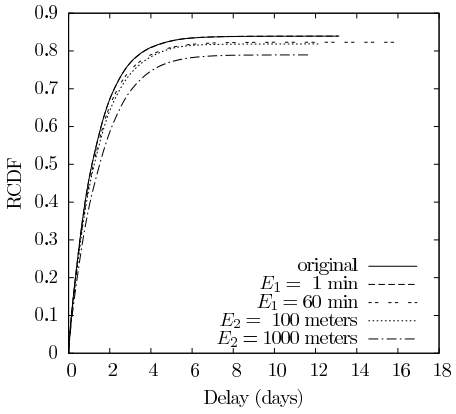

Figure 7: Sensitivity to localization errors in *diesel*

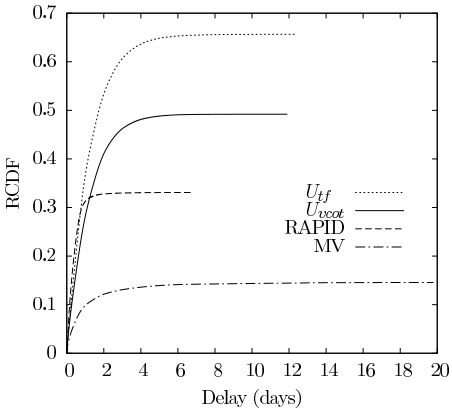

Figure 9: Performance against existing work in *diesel*

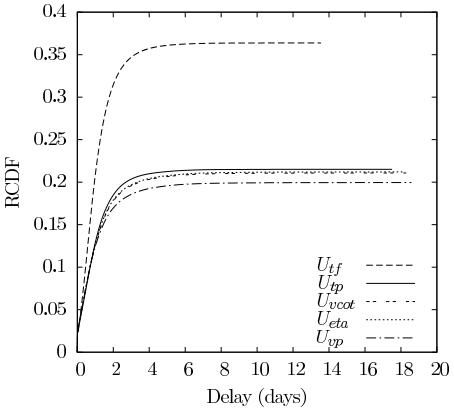

Figure 6: Comparing utility functions in *campus*

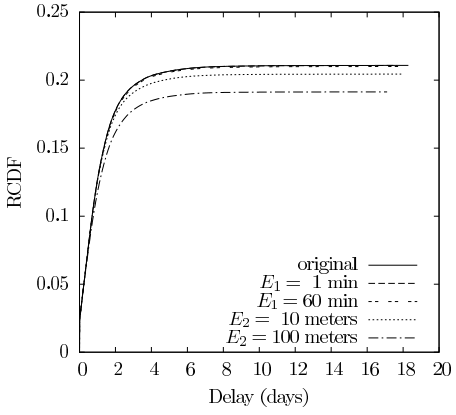

Figure 8: Sensitivity to localization errors in *campus*

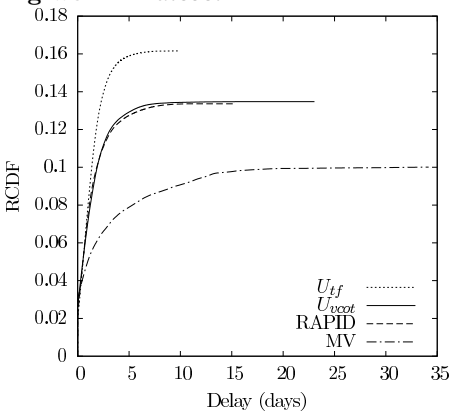

Figure 10: Performance against existing work in *campus* with 100 nodes

ment three times with different random seeds and report the average.

5.3 Parameter tuning for trajectory-based utility functions

There are four parameters for $U_{\texttt{tp}}$ and $U_{\texttt{tf}}$, the sampling interval t, the smoothing factor ω_0, the exponent λ, and the boosting factor β. We study the impact of each parameter in that order, by varying its value while setting others to a default.

The sampling interval controls the algorithms' granularity. Finer granularities result in better performance but higher demand on storage and computation. Experiments in our particular scenarios show that performance improves as t decreases, but the improvement becomes marginal when t is below several minutes.

A smaller ω_0 discounts faster the sampling points farther away the present. Proper assignment of ω_0 depends on the speed in which the system evolves. A small value shall be chosen for a fast-pacing system. For our particular scenarios, we find a value between 0.95 and 0.995 gains maximum performance. Treating all points equally, i.e. $\omega_0 = 1$, leads to low performance, which justifies the need of moving average.

According to Equation 2, λ controls how fast the utility recedes as the distance from sample points to the destination increases. A larger λ speeds up the recession and thus stresses more the preference of forwarding packets to visitors of the destination's close neighborhood. However, as desirable as it seems, if all packets are flooded to these visitors, their buffers might be overloaded and start dropping many packets. In this sense, a smaller λ spreads load more evenly leading to fewer lost packets. Our experiments show that performance peaks at $\lambda = 8$ or so.

The boosting factor β encourages nodes with direct connections to the destination. Same as λ, one may prefer a small β to distribute load evenly from direct contacts to indirect ones. The optimality of β also depends on λ. A large λ already emphasizes closeness and a small β may suffice. Our experiments suggest that when $\lambda = 1$, the best value of β is above 5,000; it reduces to 10 or so when $\lambda = 8$.

5.4 Comparing utility functions

We compare the performance of all the proposed utility functions. To calculate velocity for velocity-based functions, nodes read their location once per second and calculate velocity from two most recent readings. The parameters for $U_{\texttt{tp}}$ and $U_{\texttt{tf}}$ are set to $t = 10$ min, $\omega_0 = 0.95$, $\lambda = 8$, and $\beta = 10$. Results are shown in Figure 5 and 6. Readers may notice that the *campus* trace generally leads to less survival ratio than *diesel* (Figure 5 to 10). We believe this is caused by the less mobility of *campus* users whose move-

ments tend to be infrequent and localized if compared to those in the *diesel* trace. Meanwhile, we strive to preserve accuracy of the simulation instead of manipulating configurations for better but less realistic results.

Jain et al. [13] found that a DTR protocol using more knowledge is expected to outperform the one with less information. Our results confirm their findings: $U_{\texttt{tf}}$ significantly outperforms other protocols because it possesses the most complete information–nodes' future movement; others only conjecture such information using history or current information. The performance difference between $U_{\texttt{tf}}$ and $U_{\texttt{tp}}$ depends on how well past trajectories predict future trends. The better such *predictability*, the smaller the difference is expected to be. Similarly, the predictability using the current velocity determines the difference between $U_{\texttt{tf}}$ and velocity-based protocols. Although the two predictabilities are generally different, they happen to be comparable in our particular experiment.

Velocity-based protocols perform closely to each other. $U_{\texttt{vcot}}$ exploits more information and thus is slightly more effective than $U_{\texttt{vp}}$. $U_{\texttt{eta}}$ shows no more advantageous than $U_{\texttt{vcot}}$, because the radio range in our experiment is so small compared to the arenas' dimensions (Table 1), that the difference between two neighboring nodes in their distances to a destination is usually negligible. In this case, $U_{\texttt{eta}}$ degenerates to $U_{\texttt{vcot}}$. We expect a higher performance of $U_{\texttt{eta}}$ in long-range radio networks.

5.5 Sensitivity to localization errors

Motion-based protocols rely on localization systems to obtain the nodes' locations. In this subsection, we examine the protocols' resilience against localization errors. We model error distance from each node's estimated location to its actual position as a random variable following a Gaussian distribution with a mean of 0 and a standard deviation of E meters.

Figure 7 and 8 show the results of a representative utility function, $U_{\texttt{vcot}}$. Under highly pessimistic localization errors, the degradation on survival ratio is less than ten percent in each scenario (we use larger errors in *diesel* considering that vehicles move faster than people and the area it covers is much bigger than *campus*). Such robustness against location inaccuracy can be explained by the *distance effect* [3], which states that nodes farther away from the target are less sensitive to location errors than closer nodes. Meanwhile, a packet is very likely to be delivered when it has already been forwarded to a node close to the destination.

5.6 Comparing performance with existing protocols

We compare our work with *MV* and *RAPID*, two global-knowledge-based DTR protocols. We choose them for fair comparison because intuitively, they are expected to outperform other protocols using less knowledge. As a tradeoff however, the former's runtime cost is also expected to be higher than the latter, magnifying the advantage of our protocols when comparing resource utilization.

MV [6] is a classical protocol exploiting meeting histories for routing and buffer management. Because *MV* does not specify how global knowledge is disseminated, we assume *perfect* global knowledge on each node which gives *MV* ideal performance. Because *MV* does not support mobile destinations, we plant 30 fixed packet receivers at random locations in each trace and when generating packets, select destinations only among these receivers. *RAPID* [2] is an intentional routing algorithm aiming to optimize a specific metric. It "significantly outperforms existing protocols" [2] including *MaxProp* [4], *Spray and Wait* [28], and *PRoPHET* [22]. In *RAPID*, nodes exchange meeting histories as well as per-packet statistics and use them to optimize average delay. For fairness, we disable acknowledgements as other protocols do not have this feature. Adding it should improve all protocols equally. Two champions of motion-based protocols are $U_{\texttt{tf}}$, the best one given future schedules, and $U_{\texttt{vcot}}$, a representative one when no such information exists.

	Time	Space	Comm.
Motion-based	$O(b)$	$O(1)$	$O(b)$
MV	$O(bn)$	$O(n^2)$	$O(n^2)$
RAPID	$O(bn^3)$	$O(n^2+bn)$	$O(bn)$

Table 2: Comparing complexities

Table 2 summarizes the complexities of the protocols under comparison. It shows the time complexity of computation on each node, the space required on each node to store metadata, and the communication overhead during each contact. In the table, n denotes the number of nodes and b the number of packets buffered in a node. All the complexities of motion-based protocols are constant with the size of the network, whereas the complexities of *MV* and *RAPID* grow either exponentially or linearly. Empirical study on algorithm overhead is presented in the next subsection.

Both *diesel* and *campus* represent community networks, which are the environment providing the most fair comparison. In non-community networks, the advantages of our motion-based protocols would be greatly magnified, since global statistics are difficult to accumulate in such networks but such statistic is required by global-knowledge-based protocols to work efficiently. It is also because the number of nodes in a non-community network is unbounded, leading to a sharp increase in the complexities of these protocols. In fact, *RAPID* exhausts the memory of our 3.5 GB-memory computer system with the *campus* trace (the JVM on which our simulator runs needs all of the heap to reside in physical memory). We have to reduce the number of nodes 100 for the rest of the paper. Comparison with *MV* using the original *campus* trace is reported in a separate document [1].

Results are shown in Table 3 in terms of the traditional delivery ratio and average delay. With much lower complexities, motion-based protocols outperform *MV* in both scenarios and outperform *RAPID* in *campus*. In *diesel* however, our protocols achieve higher delivery ratio but lower delay, which generates a tie. Better presentation of the results using RCDF is shown in Figure 9 and 10. Analysis on RCDF shows that in terms of survival ratio, *RAPID* in *diesel* (Figure 9) gains 3.7% and 7.0% more low-delay packets than $U_{\texttt{tf}}$ and $U_{\texttt{vcot}}$, respectively. However, the gain is largely offset by the overall ratio: $U_{\texttt{tf}}$ and $U_{\texttt{vcot}}$ achieves 98.4% and 48.7% higher survival ratio than *RAPID*. In the 100-node *campus* trace (Figure 10), *RAPID* performs almost identically to $U_{\texttt{vcot}}$.

5.7 Comparing resource utilization with existing protocols

We measure and compare four types of resource utiliza-

	diesel		*campus (100 nodes)*	
	delivery	delay	delivery	delay
$U_{\texttt{tf}}$	64.4%	105.0	11.8%	73.5
$U_{\texttt{vcot}}$	47.1%	97.7	6.6%	77.6
RAPID	32.6%	34.3	7.4%	117.5
MV	13.9%	106.0	4.1%	256.6

Table 3: Comparing delivery radio and average delay (in 10^3 sec)

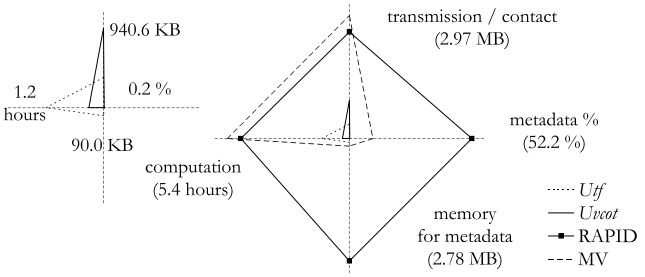

Figure 11: Resource utilization in *diesel* using the radar plot. Each axis depicts one type of resource utilization. The inset on the left is the magnified view of the plot's central area. In both views, numbers show the maximum value seen on each axis.

tion: i) the maximum amount of memory each node consumes to store metadata. We assume that basic data types including float points and packet identifiers are four-byte long. ii) communication overhead, i.e. the percentage of bandwidth used for metadata exchange. iii) the total amount of data (payload plus metadata) transmitted during each contact. iv) CPU time each node spends in computing utilities. Similar to our protocols, a key component of *MV* and *RAPID* is also the computation of a utility function. Peripheral components such as housekeeping and statistics are inexpensive and less meaningful to compare. To obtain the CPU time, we run the single-thread simulator on an Intel Xeon 2.8 GHz machine and record the time spent in executing utility functions. Each of the above metrics is reported as an average over nodes or contacts. For fairness, caching and broadcasting optimizations (§ 4) are disabled from our protocols. Finally, as mentioned earlier, readers shall notice that better performance of global-knowledge-based protocols at the cost of higher overhead, which (unavoidably) overestimates our protocols' advantage in resource utilization of many DTR protocols other than *MV* and *RAPID*.

Results are given in Figure 11 and 12 using *radar plots*, in which the total resource utilization of a scheme roughly corresponds to the area of that scheme. The results confirm the complexity analysis in Table 2. (Due to a larger constant factor in the complexity, *MV* spends more CPU time than *RAPID* despite of its lower complexity. The same situation is for *RAPID*'s communication overhead.) Our protocols consume far less resources than other protocols in all cases. If comparing $U_{\texttt{vcot}}$ with *RAPID*, the two protocols with the closest performance, $U_{\texttt{vcot}}$'s CPU time, memory footprint, and communication overhead are several, up to five, orders of magnitude less than *RAPID*. $U_{\texttt{vcot}}$'s total data transmission is also significantly lower than *RAPID*.

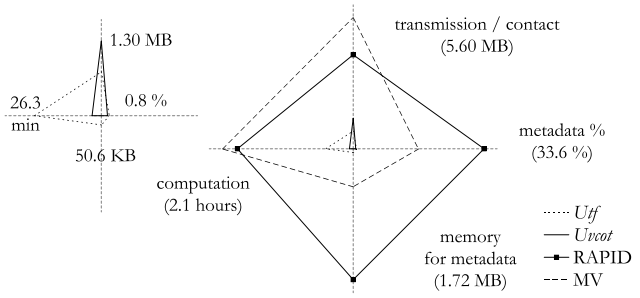

Figure 12: Resource utilization in *campus* with 100 nodes

6. RELATED WORK

6.1 Routing protocols

Numerous geographical routing protocols have been proposed in the context of mobile ad-hoc networks (MANETs) [23, 14]. These schemes forward packets to the next hop that is "closer" to the destination. Our work fundamentally differs from them for two reasons. First, MANET protocols assume end-to-end paths and may fail when no such path exists, whereas DTR protocols must cope with opportunistic connections. Second, because of this difference in requirement, geographical routing infers static network topology from node locations, and in turn find a path out of the topology, whereas our motion-based protocols infer mobility pattern from node movement, and exploit the mobility to realize packet forwarding.

DTR protocols either flood packets blindly or use certain knowledge about the network to limit flooding and to direct packet flows. Commonly used knowledge includes past or scheduled contacts, connectivities, and energy level. Such information is often disseminated across the network so that nodes can learn others' probabilities of delivery or calculate their own probabilities or both. Zhang et al. [31] provides a detailed survey on existing DTR protocols. The knowledge exploited by our protocols is the physical motion of the local node. However, the information is not propagated in order to preserve privacy and resources.

As a rudimentary work, LeBrun et al. [18] propose to use geographical information for DTR. Their protocol computes the *expected closest distance* to the destination based on the heading of a node, and decides forwarding using this distance. We go further by designing a complete protocol framework, proposing utility functions that exploit varying levels of motion information, and taking more factors into account including privacy, design tradeoffs, and mobile destinations. Furthermore, we provide empirical study and compare performance and overhead against state-of-the-art protocols with real-world traces.

Leontiadis and Mascolo [21] propose to take advantage of suggested routes from the vehicles' navigation system to select vehicles that are more likely to deliver packets to fixed destinations. Their protocol is similar to ours in the use of estimated time of arrival. However, their approach highly relies on integration with navigation systems as well as the availability of road maps. Moreover, intensive computation required by the approach also limits its applicability on power-constraint devices.

6.2 Replication and buffer management

Packets may be replicated in the network to increase their chances to reach destinations or to tolerate faults. Replication has been widely studied in the past. In particular, network coding in DTR [30] has become a promising approach to attain maximum network flow and fault tolerance using information theory. These efforts are orthogonal to and can be integrated with our work.

Our protocols drop the oldest packets when buffers full, while other dropping policies are also possible [8, 10]. More sophisticated buffer management schemes incorporate acknowledgements [4], application semantics [26], or client interests [9]. These schemes would improve certain performance metrics while calling for supports from the upper layers of the network stack.

7. CONCLUSIONS

We introduce routing protocols for opportunistic ad-hoc networks that estimate probabilities for packet delivery based on current velocity or past/expected trajectories. As an important property, these protocols employ only local information for routing decisions, which enables three advantages over existing DTR protocols: privacy protection, scalability with network sizes, and support for non-community networks. Our algorithms use increasingly more information to gain better routing accuracy at the cost of higher resource utilization in terms of memory and computation. Simulation with real-world traces show that our protocols achieve comparable or better performance while consuming far less resources than state-of-the-art protocols using global meeting histories and per-packet statistics. We also propose RCDF as a more accurate metric than the traditional combination of delivery ratio and delay for network performance analysis. It encodes both delay distribution and survival ratio in a single graphical representation.

We would like to investigate these issues in the future:

- Routing across building floors raises interesting problems. A node moving towards a destination on a different floor is unlikely to reach it if the elevator is behind the node. The protocol might also need to consider the effect of non-line-of-sight transmission through walls and floors.

- Coexisting with other protocols and switching to them on appropriate opportunities. For example, CAR [24] uses a traditional MANET protocol if an end-to-end link exists.

8. REFERENCES

[1] Additional resources. `http://www.eecg.toronto.edu/~weihan/motion`.

[2] A. Balasubramanian, B. Levine, and A. Venkataramani. DTN routing as a resource allocation problem, 2007.

[3] S. Basagni, I. Chlamtac, and V. Syrotiuk. Geographic messaging in wireless ad hoc networks, Jul 1999.

[4] J. Burgess, B. Gallagher, D. Jensen, and B. N. Levine. MaxProp: Routing for vehicle-based disruption-tolerant networks, April 2006.

[5] J. Burgess and B. N. Levine. CRAWDAD trace umass/diesel/transfer/spring2006 (v. 2007-12-02). Downloaded from `http://crawdad.cs.dartmouth.edu/umass/diesel/transfer/spring2006`.

[6] B. Burns, O. Brock, and B. N. Levine. MV routing and capacity building in disruption tolerant networks, 2005.

[7] A. Chaintreau, P. Hui, J. Crowcroft, C. Diot, R. Gass, and J. Scott. Pocket Switched Networks: Real-World mobility and its Consequences for Opportunistic Forwarding, 2005.

[8] J. Davis, A. Fagg, and B. Levine. Wearable computers as packet transport mechanisms in highly-partitioned ad-hoc networks, 2001.

[9] P. Eugster and R. Guerraoui. Probabilistic multicast, 2002.

[10] P. Eugster, R. Guerraoui, A.-M. Kermarrec, and L. Massoulie. Epidemic information dissemination in distributed systems, 2004.

[11] S. Guo, M. H. Falaki, E. A. Oliver, S. U. Rahman, A. Seth, M. A. Zaharia, and S. Keshav. Very low-cost internet access using KioskNet, 2007.

[12] J. Hightower and G. Borriello. Location systems for ubiquitous computing, 2001.

[13] S. Jain, K. Fall, and R. Patra. Routing in a delay tolerant network, 2004.

[14] X. Jiang and T. Camp. A review of geocasting protocols for a mobile ad hoc network, 2002.

[15] E. Jones, L. Li, J. Schmidtke, and P. Ward. Practical routing in delay-tolerant networks, Aug. 2007.

[16] D. Kotz, T. Henderson, and I. Abyzov. CRAWDAD trace dartmouth/campus/movement/01_04 (v. 2005-03-08). Downloaded from `http://crawdad.cs.dartmouth.edu/dartmouth/campus/movement/01_04`.

[17] A. LaMarca et al. Place Lab: Device positioning using radio beacons in the wild, 2005.

[18] J. LeBrun, C.-N. Chuah, D. Ghosal, and M. Zhang. Knowledge-based opportunistic forwarding in vehicular wireless ad hoc networks, 2005.

[19] J. Leguay, T. Friedman, and V. Conan. DTN routing in a mobility pattern space, 2005.

[20] J. Leguay, T. Friedman, and V. Conan. Evaluating mobility pattern space routing for DTNs, 2006.

[21] I. Leontiadis and C. Mascolo. GeOpps: Opportunistic Geographical Routing for Vehicular Networks. In *Proceedings of the IEEE Workshop on Autonomic and Opportunistic Communications. (Colocated with WOWMOM07)*, Helsinki, Finland, June 2007. IEEE Press.

[22] A. Lindgren, A. Doria, and O. Schelén. Probabilistic routing in intermittently connected networks, 2004.

[23] M. Mauve, A. Widmer, and H. Hartenstein. A survey on position-based routing in mobile ad hoc networks, 2001.

[24] M. Musolesi, S. Hailes, and C. Mascolo. Adaptive routing for intermittently connected mobile ad hoc networks, 2005.

[25] L. M. Ni, Y. Liu, Y. C. Lau, and A. P. Patil. LANDMARC: Indoor location sensing using active RFID, 2004.

[26] J. Pereira, L. Rodrigues, and R. Oliveira. Semantically reliable multicast: Definition, implementation, and performance evaluation, 2003.

[27] T. Small and Z. J. Haas. Resource and performance tradeoffs in delay-tolerant wireless networks, 2005.
[28] T. Spyropoulos, K. Psounis, and C. S. Raghavendra. Spray and wait: an efficient routing scheme for intermittently connected mobile networks, 2005.
[29] J. Su, A. Chin, A. Popivanova, A. Goel, and E. de Lara. User mobility for opportunistic ad-hoc networking, 2004.
[30] Y. Wang, S. Jain, M. Martonosi, and K. Fall. Erasure-coding based routing for opportunistic networks, 2005.
[31] Z. Zhang. Routing in intermittently connected mobile ad hoc networks and delay tolerant networks: overview and challenges, 2006.
[32] Z. Zhou, J.-H. Cui, and A. Bagtzoglou. Scalable localization with mobility prediction for underwater sensor networks, April 2008.

APPENDIX

In order to derive Formula 4, we divide $\mathbf{I}$ into two subsets, $\mathbf{I_s}$ for packets that will be successfully delivered in the future, and $\mathbf{I_l}$ for those will be lost. Survival ratio S is then

$$S \equiv \frac{|\mathbf{D}| + |\mathbf{I_s}|}{|\mathbf{A}|} \tag{5}$$

As the best case scenario, packets are dropped at the earliest stage of the system and packets already in the system will never be lost. Therefore,

$$|\mathbf{I_s}| \leq |\mathbf{I}|. \tag{6}$$

In the worst case, the system will lost most packets from $\mathbf{I}$. Statistically, the loss rate *within* $\mathbf{I}$ cannot be greater than the overall loss rate:

$$1 - \frac{|\mathbf{I_s}|}{|\mathbf{I}|} \leq 1 - S = 1 - \frac{|\mathbf{D}| + |\mathbf{I_s}|}{|\mathbf{A}|} \tag{7}$$

Combining Equation 5, 6 and 7 yields to Formula 4. Note that RCDF is the delay distribution over $\mathbf{D}$ and $\mathbf{I_s}$ combined. Although RCDF is actually derived from $\mathbf{D}$, because $|\mathbf{D}|$ is assumed to be sufficiently large, we can regard RCDF as an unbiased statistic over $\mathbf{I_s}$ as well.

If $|\mathbf{I}| \ll |\mathbf{A}|$ and $|\mathbf{D}|$, survival ratio can be approximated as

$$S \approx \frac{|\mathbf{D}|}{|\mathbf{A}|} = \textit{delivery ratio}.$$

It is useful when expensive or impossible to obtain $|\mathbf{I}|$. The last condition holds for most experiments that run sufficiently long. However, we observe that in opportunistic ad-hoc networks, the condition is not always true due to huge buffer sizes deployed in the network leading to large system capacities.