Chapter 2
Delay Tolerant Routing and Applications

The fundamental goal of any communication network is to have the data/packets/messages delivered to their corresponding destinations. With nontrivial network topologies having multiple paths from a given source node to a destination node, the data/packets/messages are required to be routed along an optimum (for example, least cost) path. In general, one may group the routing protocols into two broad categories—distance vector (DV) and link state (LS). DV-based protocols consider a *distance* metric to determine the best path between a pair of nodes. The *vector* in DV indicates the interface to which traffic should be forwarded so as to optimally reach the concerned destination. A typical example is the routing information protocol (RIP) used in the Internet, where the hop counts between routers act as the distance. The LS-based protocol, on the other hand, maintains a graph of the network and typically employs a shortest-path algorithm to determine the best route. Open shortest-path first (OSPF) is such an example used in the Internet.

Although similar routing algorithms have also been used in MANETs, such approaches become difficult in DTNs. This can be accounted to the different characteristics of DTNs noted in Chap. 1. The network topology of DTNs/OMNs is not only highly dynamic, but also exhibits high degree of network partitioning. In particular, the concept of end-to-end communication paths practically ceases to exist in such networks.

To overcome such issues and constraints, several schemes have been proposed in the literature to achieve efficient routing in DTNs/OMNs. In this chapter, we take a look at a broad category of such protocols. A common characteristic of all these routing protocols is that they are replication-based, i.e., they create multiple copies of a message in the OMN concerned. As we shall shortly see, the protocols attempt to restrict the extent of replication in different ways to achieve efficiency. We, then, present a set of commonly used metrics for measuring the performance of OMNs. Subsequently, we discuss about real-life traces that are increasingly being used with network simulations nowadays. Finally, we conclude this chapter by retrospecting some of the applications of DTNs.

S. Misra et al., *Opportunistic Mobile Networks*,
Computer Communications and Networks, DOI 10.1007/978-3-319-29031-7_2

2.1 Routing Protocols

Routing is a key aspect in any kind of communication network including DTNs/OMNs. We, however, skip a detailed taxonomy of the routing protocols, and look at a single aspect—the number of copies of a message that is forwarded as shown in Fig. 2.1. Interested readers may look at [52–54] for a detailed treatment.

Single-copy routing is a rarity in OMNs, since intermittent connectivity results in low delivery of the messages as well as high delivery latencies. A trivial example is the direct delivery routing, where only a source node itself delivers its messages to the corresponding destination node(s). Another example is first-contact routing, wherein a node forwards a message to the first node that it comes in contact with, and the process continues until the message reaches its corresponding destination. In reality, these schemes are hardly used. For the sake of completeness, we note that a few single-copy-based routing algorithms have been proposed for DTNs/OMNs such as minimum estimated expected delay [55] and context aware routing [56]. However, in this book, we focus only on the multi-copy schemes.

The multi-copy routing schemes can be broadly classified into two types—limited and unlimited. In limited multi-copy routing, a message is replicated for a fixed number of times and forwarded to the different nodes in the network. A typical example is the Spray and Wait [42] routing protocol.

In unlimited multi-copy routing protocols, for example, PRoPHET [19], there is no fixed limit on the number of replications of a given message that occur in the network. In particular, if N be the number of nodes in a DTN and K_i be the number of replicas of the ith message, then $0 \leq K_i \leq N$. The extreme case is flooding, where a given message is replicated and forwarded to all the nodes in the network. Often some form of heuristic is used to decide upon the upper limit of K_i, i.e., how many times a given message should be replicated. It has been shown that determining a routing schedule even with complete information is an NP-hard problem [57], which justifies the use of heuristics.

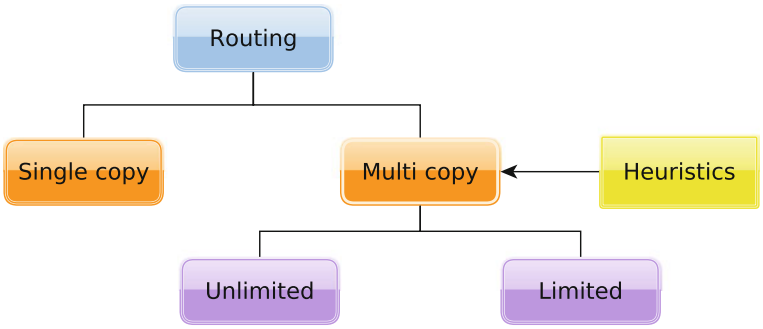

Fig. 2.1 Routing categories in DTNs. In practice, multiple replications of a message are created to ensure better delivery chances

In the remainder of this section, we briefly look at some of the well-known routing protocols used in DTNs. The review provides insights about their basic characteristics operations. Furthermore, we also look at the different extensions suggested and changes made to those routing protocols in the literature.

2.1.1 *Epidemic*

Vahdat and Becker [18] proposed the epidemic routing protocol for ad hoc networks with high network partitions. The epidemic routing protocol (and similar protocols in distributed systems) is modeled after the spreading of epidemic diseases in real life. With this analogy, a node consisting of a message, m, may be considered as already *infected*. Now, when such a node comes in contact with another node not having that message, the former node transmits m to the latter and thereby, infects it. Eventually, after infecting multiple nodes in the network, the message m reaches to its destination. The goal of the epidemic routing protocol has been to maximize the chances of message delivery ratio and minimize latency while minimizing the aggregate resource consumption (for example, bandwidth and energy) in the network [18].

As illustrated in Fig. 2.2, the interaction between a pair of nodes—when within the transmission range of one another—using the Epidemic routing protocol involves two steps:

1. **Exchange of summary vectors**: Each node maintains an index of the messages (for example, unique message IDs) that it is carrying in its buffer. Let, V_X and V_Y, respectively, be the summary vectors of the two nodes X and Y. As shown in Fig. 2.2, the nodes exchange V_X and V_Y between themselves.
2. **Exchange of messages**: The node X computes $V_X^{'} = V_Y \setminus V_X$, the set of messages that are carried by the node Y, but not present in the buffer of node X. Similarly, node Y computes $V_Y^{'} = V_X \setminus V_Y$. If such a set computed by a node is not empty, it requests its peer to transmit the messages with the corresponding message IDs.

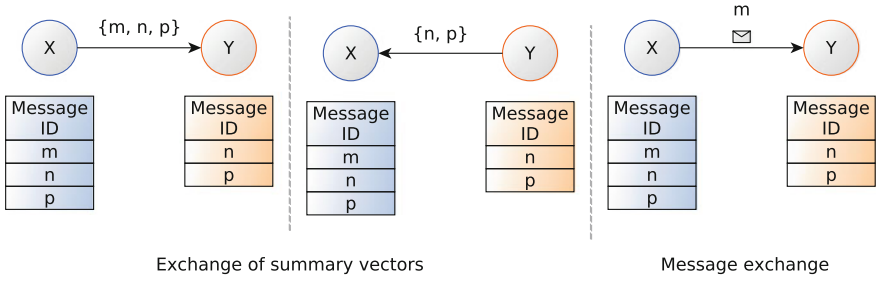

Fig. 2.2 Message exchange between two nodes using the epidemic routing protocol

Example 2.1 With reference to Fig. 2.2, we have $V_X = \{m, n, p\}$ and $V_Y = \{n, p\}$. Therefore, $V_X^{'} = \varnothing$ and $V_Y^{'} = \{m\}$. So, node Y requests node X to send the message m. However, node X makes no such request to node Y.

In a similar way, the nodes X and Y subsequently replicate m to the other nodes in the network with which they come in contact. When one such node meets with the destination node, it delivers m to the latter. □

The authors associated the following three properties [18] to each message generated in the network.

- **Message identifier**: This serves as the ID to uniquely identify a message in the network. The authors suggested that such an identifier can be obtained by concatenating the address of a node together with a local message ID value.
- **Hop count**: This field indicates the maximum number of time a given message can be exchanged using the aforementioned protocol. Messages with zero hop count can be removed from the buffer in the face of storage crisis depending upon the local buffer policies.
- **Acknowledgment request**: A node can optionally request for an acknowledgment message be sent by the destination node once a given message is delivered.

However, these properties are not just specifically used with Epidemic. In fact, these and others, such as source node identifier, destination identifier, TTL, message length, and so on, are some common metadata maintained with the messages by almost any routing protocol. Often other additional information are also maintained as demanded by specific protocols. Unlike traditional networks, acknowledgments are typically not used in DTNs/OMNs, since the round trip time of having a message delivered and subsequent receipt of its acknowledgment by the source node is often very large. However, if appropriate, acknowledgments can be used to "clean up" the network. Let us look at how such clean up can be useful in real life.

Since in epidemic routing a message is replicated to all possible nodes in a network, the overall storage requirements become high in such networks. Theoretically, if n messages are created in an OMN, the complexity of storage, and transmission, overhead becomes O(n)—linearly increasing with the number of messages. To overcome such effects, different recovery schemes, such as *immunity* and *vaccine*, and their effects have been considered by researchers [58–62]. The idea of these schemes arise from biology, wherein a person either has immunity to a given epidemic disease and does not get affected by the it, or is given a vaccine when affected by the same. The objective of these proposed schemes in communication networks is similar—to help propagate delivery information of messages to other nodes so that they (the other nodes) can stop replicating and storing those particular messages.

In the immunity scheme [58, 59], when a node delivers a message to its corresponding destination, the former node removes the message from its buffer. Moreover, it stores the ID of the delivered message with itself so that it does not accept the same message again from any other node in the future. In other words, the node is immune and could no more be *infected* by the *disease*. The stored ID (of the

delivered message) is often referred to as *anti-packet*, similar to the concept of antibody in biological systems.

A proposed variation of the immunity scheme is to propagate the anti-packet information to other nodes who are currently infected, i.e., contain a copy of the message that has been already delivered. In the vaccine scheme, on the other hand, such anti-packet information is propagated to all possible nodes with the intention of preventing any node receiving an already delivered message.

Zhang et al. [59] presented a performance modeling of the epidemic routing protocol together with different recovery schemes discussed above. The authors used a Markov model for the purpose and derived ordinary differential equations (ODEs) as the limiting process of such model considering natural scaling. On the other hand, Li et al. [60] considered a two-dimensional continuous-time Markov chain to represent the message delivery in the presence of social selfishness and studied the resulting effects on the routing performance.

Mundur et al. [61] considered epidemic routing in DTNs together with the immunity scheme. Similar to the summary vector list, the authors considered an "immunity" list maintained by each node, which stores the IDs of the messages already delivered. During an encounter, a pair of nodes exchange both the summary vector and the immunity list. This prevents a node from receiving a copy of a message again that it has delivered to the corresponding destination in the past.

Matsuda and Takine [62] considered the (p, q)-Epidemic routing protocol, wherein a message is probabilistically replicated by a node. In particular, a relay (non-source) node replicates a message to another relay node with probability p, whereas a source node replicates a message to a relay node with probability q. As such, the (p, q)-Epidemic protocol encompasses a large class of routing protocols:

- When $p = q = 0$, it degenerates to direct delivery routing protocol, where a source node itself delivers a message to the corresponding destination node (i.e., zero replication in the network).
- When $p = q = 1$, we get the original Epidemic routing protocol.
- When $p = 0$ and $q = 1$, (p, q)-Epidemic transforms into two-hop routing protocol.

Subsequently, the authors considered the dynamics of the network and presented a corresponding model for the performance of the (p, q)-Epidemic routing protocol together with the vaccine scheme.

2.1.2 Spray and Wait

Spyropoulos et al. [42] proposed the Spray and Wait (SnW) routing protocol, which imposes a maximum limit on the number of possible replications of a message. The SnW protocol consists of two phases:

1. **Spray phase**: In the spray phase, a particular message is spread to at most L different relay nodes. Such spraying begins with the source node, i.e., the node that created the message. Any node receiving such a message can, in turn, engage in spraying. The spraying process could be of different types, as we would soon discuss below.
2. **Wait phase**: If the destination node was not encountered with during the spray phase, the intermediate nodes carrying a copy of the message perform a direct delivery when they come in contact with the corresponding destination of the message.

It may be noted that multiple nodes carrying a copy of a message say, m, may come in contact with the destination node and forward the same. The destination node, however, is assumed to receive only a single copy of m—from the node that it first comes in contact with. When the other nodes try to forward the same message, the destination can "refuse" to receive them or just simply discard them.

There are different variations of the spraying technique of the messages. In the "source" spraying technique [42], the source node itself distributes L copies of a given message up to first L nodes that it comes in contact with which does not already have a copy of the message. Of course, if the destination node is encountered before that, the message is directly forwarded to it, and the routing of that particular message successfully terminates. Otherwise, any of those L nodes can directly deliver the message to the destination node if and when encountered.

The binary spraying technique [42] is slightly different. In this version, the source node initially has L copies of a given message. Now consider any node—whether the source or a relay—that has n copies of the message, $L \geq n > 1$. If one such node encounters another node having no copy of the message, the latter node is given $\lfloor n/2 \rfloor$ copies of the message, while the former itself retains $\lceil n/2 \rceil$ copies. It has been shown that the binary spraying technique is optimum and results in minimum expected delay among all Spray and Wait routing mechanisms.

Spyropoulos et al. [42] observed that SnW brings in the together the speed of Epidemic routing and the simplicity of direct delivery routing. As such, this can be viewed as a trade-off between single- and multi-copy routing protocols. The authors proved that the minimum value of L to achieve an expected delay does not depend on the size of the network or the transmission ranges of the nodes, but only on the number of nodes in the network. Subsequently, the authors also provided a method to estimate L when the network parameters are not known.

The authors further observed that in SnW, the message copies are sprayed as fast as possible to the nodes' neighborhood [63, 64]. Thus, if the mobility of such nodes is confined to a small locality (for example, movement of the students inside a university campus [65]), the chance of encountering the destination, and subsequent delivery of the message, is greatly reduced. So, although SnW is fast and resource friendly, constrained mobility imposes limitations to the protocol. Consequently, the authors [63, 64, 66] proposed the timer transitivity-based Spray and Focus (SnF) algorithm to overcome such adversity.

SnF [63] has a spray phase similar to SnW. However, instead of the wait phase, SnF introduced the "focus" phase, where a forwarding decision is based on a utility. The authors defined single-copy utility-based routing as follows. Let us consider that each node i in the network maintains a utility value, $U_i(j)$, for every other node j in the DTN, $\forall i, j$. Further, let us assume that a node X, which has a message destined for the node Z, comes in contact with another node Y, $Y \neq Z$. Then, X forwards the message to Y, if and only if their utility functions satisfy the following relationship:

$$U_Y(Z) > U_X(Z) + \theta, \tag{2.1}$$

where θ is some threshold.

The authors considered timer transitivity to define the utility functions. In particular, let $\tau_i(j)$ be the duration since the last encounter between the nodes i and j as maintained by the node i. Initially, $\tau_i(i) = 0$, $\tau_i(j) = \tau_j(i) = \infty$. Moreover, at every encounter between i and j, these counters are reset by the respective nodes, i.e., $\tau_i(j) = \tau_j(i) = 0$.

Let the distance between the nodes X and Y be d_{XY} at the time when they encountered. Further let, M be the mobility model followed by the nodes so that $t_M(d_{XY})$ be the time required, on an average, to move the distance d_{XY} as per the mobility model. Then, for any node $Z \neq Y$, if $\tau_X(Z) > \tau_Y(Z) + t_M(d_{XY})$, update the counter as $\tau_X(Z) = \tau_Y(Z) + t_M(d_{XY})$. The rationale here is that, due to the transitivity property, the node X had a more "recent" encounter with Z.

Based on this, the working of the SnF algorithm can be described as follows:

- Each node maintains a summary vector of the messages as described earlier.
- Each node maintains the last encounter timers as discussed above.
- Any node—either the source or an intermediate relay—with $1 < n \leq L$ copies of a message, sprays the message copies according to the binary spraying algorithm. However, if $n = 1$, the node forwards the message copy according to (2.1). As mentioned earlier, the last encounter timers are used to determine the values of these utility functions.

Example 2.2 Let us consider that the nodes A and B, moving with uniform velocity, came in contact with another node j at time instants $t = 1$ and $t = 3$, respectively. Further, let $t_M(d_{AB})$ be 1. Therefore, at an instant of time $t = 6$, we obtain $\tau_A(j) = 6 - 1 = 5$, $\tau_B(j) = 6 - 3 = 3$. So, $\tau_B(j) + t_M(d_{AB}) = 3 + 1 = 4 < \tau_A(j)$. Therefore, $\tau_A(j) = \tau_B(j) + t_M(d_{AB}) = 5$. □

The above example is meant for the purpose of simple illustration. A few aspects, however, should be kept in consideration in this context as discussed in [64]. Spyropoulos et al. suggested corrections in the (larger) timer value when two individual values differ by a large extent. However, in our example, we illustrated the procedure with small values. Moreover, the term $t_M(d_{XY})$ mentioned above is a property of the underlying mobility model M. So, unless the nodes are moving with constant velocity, certain related computations might be required. Interested readers should also look at [64] for a variation of the Spray and Focus algorithm.

Spray and Focus algorithm is not a single-copy routing protocol. Similar to SnW, messages are sprayed here, too. However, unlike the SnW algorithm, where a node with a single copy of a message directly delivers it to the corresponding destination, SnF uses forwarding (not replicating) of such messages. Whether or not such a message with a single copy should be forwarded to the other node is decided based upon their utility functions.

2.1.3 PRoPHET

Lindgren et al. [19] proposed the probabilistic routing protocol using history of encounters and transitivity (PRoPHET). PRoPHET is a greedy algorithm, where a node forwards a replica of a message to another node only if the latter node has a greater chance of encountering the destination of the message than the former itself. PRoPHET uses the concept of *delivery predictabilities* (DP), the likelihood of one node meeting with another node in the OMN. Every node running PRoPHET in the network maintains a table of such probabilities of meeting with every other known node. Moreover, the DPs are dynamic—if two nodes do not meet for a certain time period, their DP value for the other node decays appropriately. Therefore, a pair of frequently meeting nodes would have high values of DP for one another.

Equations (2.2) through (2.4) [19] related to updating delivery predictabilities on encounter, aging of predictabilities, and capturing transitive predictabilities, govern the functionality of this routing protocol:

$$P_{(a,b)} = P_{(a,b)\text{old}} + (1 - P_{(a,b)\text{old}}) \times P_{\text{init}} \tag{2.2}$$

$$P_{(a,b)} = P_{(a,b)\text{old}} \times \gamma^k \tag{2.3}$$

$$P_{(a,c)} = P_{(a,c)\text{old}} + (1 - P_{(a,c)\text{old}}) \times P_{(a,b)} \times P_{(b,c)} \times \beta \tag{2.4}$$

Here, $P(A, B)$ and $P(A, B)_{\text{old}}$, respectively, indicate the current and previous delivery predictabilities of node A for another node B, $P(A, B) \in [0, 1]$. The parameter $P_{\text{init}} \in [0, 1]$ is an initialization constant. The delivery predictabilities are aged with time when a pair of nodes does not encounter for long. The parameter $\gamma \in [0, 1)$ is the aging constant, and k denotes the number of time units expired since the last update of this predictability. Such aging helps in eliminating stale information maintained by the nodes. The delivery predictability $P(A, B)$ acts as heuristic here.

The scaling parameter $\beta \in [0, 1]$ controls the extent to which transitivity should affect the delivery predictability. The transitivity phenomenon is explained as follows. Suppose that node X encounters node Y frequently and node Y, in turn, encounters node Z frequently. Therefore, any other node i, which has a message to send to node Z, may as well forward the message to node X. Although node X may not

have frequent encounters with node Z, the transitivity property allows the messages to be transferred from X to Z via node Y. The proposed values of P_{init}, β, and γ, respectively, are 0.75, 0.25, and 0.998 [19].

Figure 2.3 illustrates the sequence of actions taken when two nodes A and B using PRoPHET come in contact with one another. After the exchange of summary vector of the messages and the delivery predictabilities, each node updates their own DP values. Subsequently, candidate messages, if any, are exchanged between them. In other words, node A (B) replicates messages to node B (A) if the latter has a higher likelihood to encounter the destination of the message—directly or via others—than A (B) itself.

Example 2.3 Let $P(A, B) = 0.5$, and the pair of nodes does not have any subsequent encounter. If a single time unit comprises 60 s, then after two time units, the updated delivery predictability would be $P(A, B) = P(A, B)_{\text{old}} \times \gamma^2 = 0.5 \times 0.998^2 = 0.498$. □

PRoPHET has inspired many subsequent protocols, and several improvements to PRoPHET itself have been proposed. The utility evaluation of the Spray and Focus protocol [63, 64] discussed earlier was inspired by similar transitivity. Huang et al. [67] proposed PRoPHET+, where, apart from delivery predictability of the nodes, other performance indicators, such as buffer capacity, power level, and popularity, are also considered. Such scalar values are combined together with appropriate weights to determine the *deliverability* of a node based on which the messages are replicated. On the other hand, Li and Das [68] proposed a trust-based mechanism for data forwarding, which was evaluated by integrating with PRoPHET. The proposed scheme was shown to be efficient against attacks.

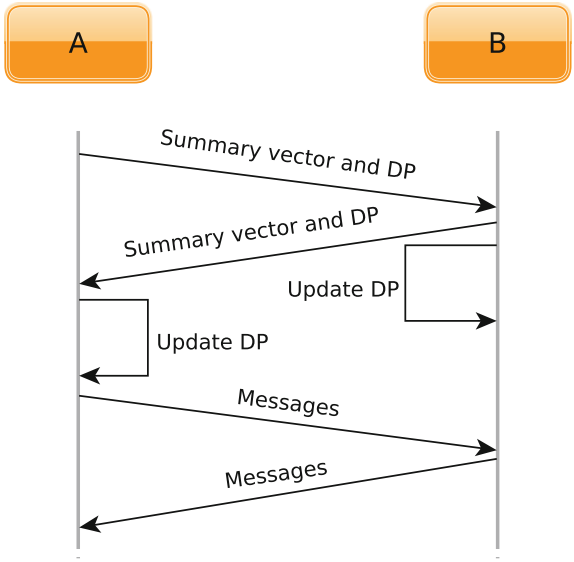

Fig. 2.3 Sequence of interaction between two nodes using PRoPHET. Here, DP indicates the delivery predictabilities maintained by each node

Grasic et al. [20], however, pointed out that the delivery predictabilities used in PRoPHET may not always be able to reflect the real-world dynamics. The authors discussed several scenarios where high values of delivery predictabilities were computed, but were undesirable. For example, in the parking lot problem [20], devices were apparently found to have frequent encounters in short time periods. In reality, the devices had multiple reconnection among themselves due to fluctuating Wi-Fi signals, which were treated as "new" encounters. Such observations do not correspond to typical human movement patterns scaling over hours or days. The authors observed that when $\beta > 0$, the DP evaluated by any node using (2.4) would always keep increasing.

To overcome such problems of PRoPHET, Grasic et al. [20] proposed an improved version of the protocol—PRoPHETv2. In this version, P_{init} is computed as

$$P_{\text{init}} = \begin{cases} P_{\max} \times \Delta_j / \Delta, & 0 \leq \Delta_j \leq \Delta \\ P_{\max}, & \text{otherwise,} \end{cases} \tag{2.5}$$

where Δ_j is the time since the last encounter with node j and Δ indicates the typical time interval, on an average, between two successive connections for a given network scenario. Moreover, the transitive predictability evaluation is modified as

$$P(a,c) = \max\{P(a,c)_{\text{old}},\ P(a,b) \times P(b,c)\ \times \beta\} \tag{2.6}$$

Such an operation stabilizes unexpected amplification of the delivery predictabilities.

2.1.4 RAPID

Balasubramanian et al. [57] treated routing as a resource allocation problem in DTN and proposed the resource allocation problem for intentional DTN (RAPID) protocol. By *intentional*—in contrast to *incidental*—it is meant that the proposed protocol can explicitly optimize a routing metric of choice. To achieve this, RAPID used an in-band control channel to exchange various metadata including expected contact time with other nodes, list of messages delivered, and average size of past transfer events.

RAPID essentially defines a per-message[1] utility function. A message i is replicated in such a way that it locally optimizes the marginal utility $\delta U_i / s_i$, where U_i is the contribution of message to the concerned routing metric, and s_i is its size. RAPID is composed of three functional components:

[1] In [57], the authors considered routing of packets. For the sake of consistency, we use the term message here. The general problem of routing, however, remains the same irrespective of whether the data unit considered is a packet or a message.

1. **Selection algorithm**: When two nodes come in contact, at first they exchange their metadata. Subsequently, they directly deliver the messages, if any, destined for the other node in decreasing order of their respective utilities. Finally, the remaining messages—that are not already carried by the other node—are replicated in decreasing order of their marginal utility $\delta U_i/s_i$.
2. **Inference algorithm**: It is used to compute the utility of a message for a specific routing metric. For example, in order to minimize the average delivery latency of a message, the concerned utility can be defined as $U_i = T_i + R_i$, where T_i and R_i, respectively, denote the time duration since creation and expected remaining time for delivery of the message i. Similarly, utilities for other desired metrics can be defined as well.
3. **Control channel**: The control channel is used to exchange the previously described metadata. It was found that the overhead due to the control channel was quite less. Moreover, although the control channel cannot always provide most recent and accurate information, even such inaccuracies were significantly helpful.

Balasubramanian et al. discussed examples of three particular metrics that can be used with RAPID to evaluate the utility function U_i. The first metric aims to minimize delivery delay, $D(i)$, of a message i, on an average, in the network so that $U_i = -D(i)$. The second metric aims to maximize the chances of delivering a message i to its destination before its TTL, $L(i)$, expires. In other words, the probability that $L(i) - T(i)$ is greater than the expected delivery time is considered. The final metric targets to minimize the maximum delay experienced by the set of messages in a node's buffer. Once the parameters are plugged into U_i, the marginal utility can be evaluated, and the replication decision subsequently taken. The authors evaluated RAPID using simulations and test beds. The performance in both cases was found to be very close and better than other contemporary routing protocols.

It may be noted here that although both PRoPHET and RAPID use heuristics for efficient routing of the messages, their approaches differ. In case of PRoPHET, the heuristic is defined as per destination node, whereas in RAPID, per-message utility is considered.

Due to intermittent connectivity, contemporary global information about the OMNs is often not available to the nodes. Therefore, while routing, a node typically attempts to make an optimal message forwarding decision based on locally available information.

2.1.5 Bubble Rap

In real life, interactions among people can largely be represented in terms of interacting communities. With such knowledge, it is possible to determine, or predict, how a message from a person passes on to a different person. Hui et al. [69] exploited such social relationships among the people—and therefore, their devices—in OMNs, and thereupon developed a message forwarding algorithm termed as Bubble. Two social metrics—*centrality* and *community*—are combined together for taking message forwarding decisions.

The Bubble algorithm assumes that every people in the OMN belong to at least one community. Moreover, each node has two rankings associated with themselves. The first ranking is a local ranking, where a node is assigned a relative rank in its own community. The other one is the global ranking, which is valid across the entire OMN. Message forwarding using Bubble essentially involves climbing—or *bubbling*—up the hierarchical ranking tree. Initially, a message is replicated by the nodes using their relative global ranking until the message reaches to a node that belongs to the same community as the destination of the node. Once a member of the desired community receives the message, it uses local ranking of the other nodes in the community to further disseminate it until the message reaches to its corresponding destination node. Figure 2.4 illustrates this process. The authors used the term "bubble" to indicate the communities in an OMN.

Bubble uses the inherent mobility patterns of human carriers of the devices as a metric for forwarding data in OMNs. The forwarding paths are selected based on the correlated interaction between humans in a community. More popular people have more number of connections. In other words, they have high *centrality*, and are identified as hubs. A group of nodes in an OMN having common interest and social links form a community. Based on the community, each node has two different types of centrality as follows:

- **Local centrality**: A node sends the message according to local ranking within the local community.
- **Global centrality**: A node sends the message according to global ranking across the whole network (global community).

In graph theory and social network analysis, centrality is a measure of the *importance* of a given node in a graph. For example, in a star topology, the central node has the highest centrality, since it makes communication with all the other nodes possible. This particular measure is called the degree centrality, where the "degree" of a node is the number of links that a node has. Several other measures of centrality have been proposed in the literature, for example, those based on closeness and betweenness. Interested readers may take a look at [70–72] for further details.

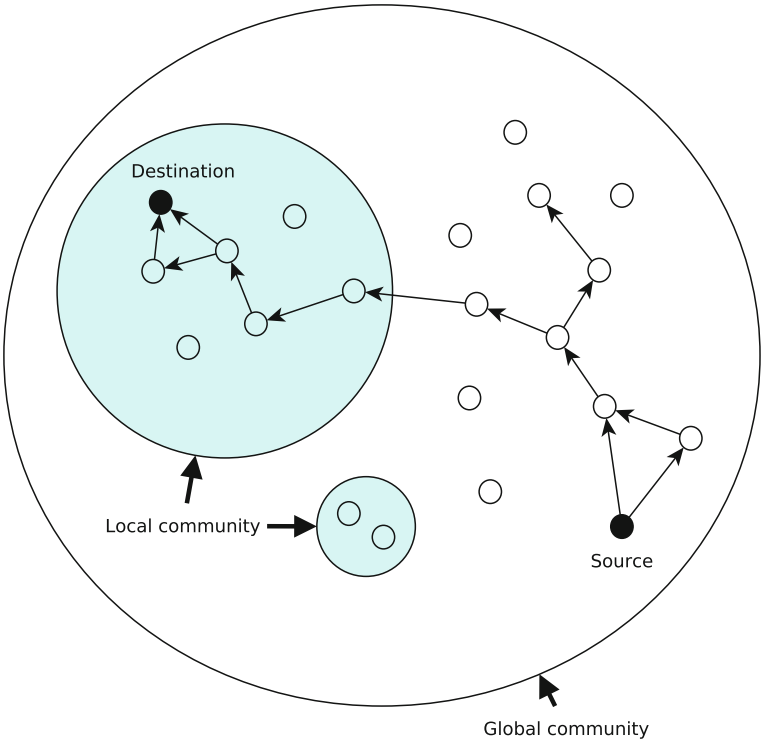

Fig. 2.4 An instance of message forwarding using Bubble

Bubble works well when the destination node's communities member belongs to high rank. It fails to work, when the destination node belongs to communities whose members have low global centrality values. In such situation, the nodes of global communities may not be able to reach the appropriate local communities of the destination node.

2.2 Routing Based on Encounter Statistics

In this section, we look at three routing protocols that take into consideration the number of encounters (contacts) among the nodes in an OMN. Their message replication decisions are optimized based on such encounter statistics. The simplicity of these routing protocols make them interesting candidates for study.

2.2.1 Encounter-Based Routing

Nelson et al. [73] proposed the encounter-based routing (EBR) protocol for DTNs. The authors noted that certain nodes (for example, ambulances in a post-disaster scenario) in a DTN might have more contact (or encounter) opportunities than other nodes. EBR essentially exploits this property to aim for increased delivery of the messages while maintaining a low routing overhead.

EBR introduces the concept of *quota-based* replications, where the maximum number of replications of any message is independent of the number of nodes in the network. The objective of quota-based replications is to reduce the number of replicas of messages in the network, on an average, and thus, to reduce the routing overhead.

In EBR, each node in the DTN maintains their average encounter rate with the other nodes in the network using exponentially weighted moving average method. In particular, each node maintains a variable, EV, to store the node's past rate of encounters. Additionally, the variable CWC stores the number of encounters with the node that happened in the current time window. Mathematically,

$$\mathrm{EV} = \alpha \times \mathrm{CWC} + (1 - \alpha) \times \mathrm{EV}. \tag{2.7}$$

The novelty of EBR lies in the fact that EBR uses the value of the variable EV to determine how many replica(s) of a given message should be sent to the other node. Specifically, if nodes A and B come in contact, and node A has a message m with itself, then the number of replicas of m, $r(m)$, sent to B is given by

$$r(m) = m_A \times \frac{\mathrm{EV}_B}{\mathrm{EV}_A + \mathrm{EV}_B}, \tag{2.8}$$

where m_A indicates the number of replicas of m carried by A. Subsequently, node A updates the value of m_A for the message m as

$$m_A = m_A - r(m). \tag{2.9}$$

The performance of the EBR protocol was evaluated under diverse scenarios. It was found that EBR resulted in improved message delivery ratio (up to about 40 %) as compared to the contemporary routing protocols, while concurrently yielding a better goodput. In other words, EBR was found to be resource friendly.

2.2.2 Contact-Based Routing in DTNs

Contact-based routing (CBR) [74] is a flooding-based protocol that reduces the average message storage time at a node, while maintaining similar or higher message

delivery ratio as compared to other routing protocols. In CBR, on every connection establishment between a pair of nodes, each node exchanges information such as frequency of contacts with the other nodes. Based on these frequency of contacts, each node identifies the relatively better candidate nodes, and forwards their messages to the such nodes.

Every node running CBR maintains the frequency of contact with other nodes in a table F. For example, a tuple $\langle a, k\rangle \in F$ indicates that there has been k contacts with a node identified by the address a, $a \in N$, where N is the set of nodes in the OMN. Whenever a new node comes in contact for the first time, a corresponding record is inserted into F with $k = 0$. For subsequent contacts, the value of n is incremented by 1 each time.

The CBR protocol has two main components—contact history maintenance and candidate node selection. During its lifetime, when a node comes in contact with another node, it increments the contact count for the latter and updates F accordingly, as discussed above. Subsequently, for every message held in the buffer, a node determines the most appropriate node currently in contact, and forwards the corresponding message to the selected node. In case of CBR, the frequency of encounters of a node determines its underlying utility. Therefore, when a node is in contact with multiple nodes, it selects the one that has maximum encounter frequency, and replicates a message to that node. To represent mathematically, let C_i be the set of nodes that node i is in contact with at a given time instant; $|C_i| \geq 1$. Moreover, let us consider that node i has a message m, which is destined for j. Further, let $I(m, k)$ be an indicator function, which is 1 if node k contains the message m, and 0 otherwise. Let

$$c = \underset{x, I(m,x)=0}{\arg\max} \quad F_x[j] \tag{2.10}$$

where F_x is the frequency table of node x. Then, node i replicates the message m to the candidate node c. In case multiple nodes have the same highest frequency of encounters, one such node is randomly chosen among them.

2.2.3 Delegation Forwarding

Erramilli et al. [75] proposed Delegation Forwarding (DF), a novel routing protocol for OMNs. The objective of DF is to reduce the overhead of message delivery in OMNs, which is achieved using a astonishingly simple algorithm.

In DF, every node is assigned a *quality* metric, which captures the likelihood of a node to deliver a given message to its corresponding destination. Thus, quality, $\mathbf{q}$, is a vector of real numbers for each node in the OMN. Additionally, every message created in the network is assigned a property called *threshold*. Initially, when a message m is generated by a source node say, s, its threshold, $\theta(m)$, is set to $\mathbf{q}[i]$, the

current measure of quality of s for the node i, which is the destination of the message. Subsequently, when the source node comes in contact with another node say, j, it (node s) replicates the message m to j if $\mathbf{q}[j] > \theta(m)$. The threshold value of the message is then updated to $\mathbf{q}[j]$ by both s and j. Similarly, when node j comes in contact with k, the replication decision is based on the result of comparison between $\mathbf{q}[k]$ and $\theta(m)$. This process is repeated for each message held by the nodes in the OMN.

Essentially, each node running DF always tries to find the *best* forwarding candidate with respect to time. The threshold property assigned to each message helps in storing the past history. Therefore, threshold of such a message is a strictly monotonically increasing function of time. The authors suggested various metrics that can be used to measure the quality of a node such as, frequency of encounters, last time of contact with any node, destination-specific frequency of encounters, and destination-specific last contact time. The first two metrics are independent of destinations of the messages. Therefore, in such cases, quality is a scalar quantity rather than a vector.

Experimental results indicate that the message delivery overhead obtained using DF is remarkably less when compared to other routing protocols. However, such optimization comes with a tradeoff. Waiting for a better candidate node to replicate a message increases the message delivery latency, on an average. Nevertheless, when overhead is a critical issue—and higher delays can be tolerated—DF stands out as an excellent choice for routing in OMNs.

Table 2.1 presents a comprehensive summary of the different routing protocols for OMNs studied so far. Table 2.1 shows that most of the protocols except for Epidemic and SnW use some form of intelligence to make routing decisions. This is true because both Epidemic and SnW blindly replicate the messages in the OMN. However, SnW, as well as SnF and EBR, replicates only up to a fixed upper limit. As such, they have relatively lower delivery cost in terms of the number of message replicas. Among these routing protocols, Bubble uses social metrics for message forwarding.

Overhead of the routing protocols in terms of storage and transmission cost are also noted in the Table. In particular, if the OMN has n nodes, the storage overhead of PRoPHET becomes $O(n)$. This is due to the reason that a node running PRoPHET has to store the predictabilites of the other nodes in the OMN, which asymptotically reaches to $O(n)$. Similarly, the transmission overhead of PRoPHET is also $O(n)$, since the delivery predictabilities vector need to be exchanged between two nodes. SnW and Epidemic, however, have zero overhead in both the cases assuming that they do not exchange summary vectors of the messages. If such an exchange is involved, then the transmission cost runs into $O(m)$, where m is the number of messages created in the OMN. Alternatively, use of Bloom filters can reduce that to constant cost. Finally, the overhead of DF is either $O(n)$ or constant depending upon whether or not destination-specific utility values are used by the protocol.

Table 2.1 Comparative summary of different routing protocols

Protocol	Routing intelligence	Maximum replications	Social metrics	Routing storage overhead	Routing transmission overhead
Epidemic [18]	–	Unlimited	–	0[a]	0
PRoPHET [19]	Delivery predictabilities	Unlimited	–	$O(n)$	$O(n)$
PRoPHET+ [67]	Delivery predictability, buffer capacity, popularity	Unlimited	–	$O(n)$	$O(n)$
Spray and Wait [42]	–	Limited (L)	–	0	0
Spray and Focus [63]	Timer transitivity	Limited (L)	–	O(1)	O(1)
RAPID [57]	Time duration since message creation and expected remaining time for delivery	Unlimited	–	$O(\max\{m, n\})$[b]	$O(\max\{m, n\})$
BUBBLE [69]	Mobility pattern of the nodes	Unlimited	Local and global centrality	O(1)	O(1)
Encounter-based Routing [73]	Most encounters	Limited	–	O(1)	O(1)
Contact-based Routing [74]	Neighbor with most encounters	Unlimited	–	$O(n)$	$O(n)$
Delegation Forwarding [75]	Quality and threshold	Unlimited[c]	–	O(1) to $O(n)$[d]	O(1) to $O(n)$

[a]Ignoring storage and transmission of summary vectors
[b]Metadata exchanged includes expected meeting times with every node and message (packet) replication information
[c]Practically, however, it is very low
[d]$O(n)$ when destination-specific metrics are used. Here, n is the number of nodes in the OMN

2.3 Performance Indicators and Key Insights

In this section, we present some metrics typically used for the performance evaluation of an OMN. However, by no means this is an exhaustive list. The reader should also consider metrics suitable for his/her specific scenario, if relevant. Subsequently, we also compare the performances of some of the previously discussed routing protocols under a few scenarios.

2.3.1 Performance Evaluation Metrics

Let, M and M_d, respectively, be the set of messages generated and delivered in the network, $M_d \subseteq M$. Further, let t_i and $t_i^{'}$, respectively, be the time instants when a message $m_i \in M_d$ was created and delivered to its destination, $t_i^{'} > t_i$. Note that due to the decentralized nature of OMNs, a given message can be "delivered" multiple times to its corresponding destination by different nodes in the OMN. However, by "delivery," we refer to the first such instance when a node receives a message destined for itself.

Moreover, let us consider that each message $m_i \in M$ has r_i replica in the OMN, $0 \leq r_i \leq N$, where N indicates the total number of nodes in the network. Based on these, we define the following metrics.[2]

- **Delivery ratio of the messages**: This metric indicates, on an average, the fraction of the messages created in the network that were successfully delivered to their corresponding destinations. The average delivery ratio of the messages is evaluated as $|M_d|/|M|$. This is one of the primary metrics used to evaluate the performance under a given scenario or using a particular protocol.
- **Mean message delivery latency (delay)**: This metric gives an indication of, on an average, the time required to deliver a message from its source to its corresponding destination. This is evaluated as $\sum_{i=1}^{|M_d|}(t_i^{'}-t_i)/|M_d|$. Typically, the average delivery latency of a message in OMNs is high and could be even up to few thousand seconds. Therefore, one of the objectives usually considered is minimizing this latency.
- **Median message delivery latency (delay)**: Let the set of delivery latencies of the messages be $D = \{t_i^{'} - t_i\}$, where $|D| = |M_d|$. Sort the elements of D in ascending order. The $(|D| + 1)/2$ th element of the sorted list gives the median message delivery latency.
- **Overhead ratio of message delivery**: Routing in OMNs is usually replication based. As such, multiple copies of a message exist in the network. This has repercussions on the storage capacities and energy levels of the nodes. The overhead ratio is computed as $(\sum_{i=1}^{|M|} r_i - |M_d|)/|M_d|$.

[2] An initial version of some of these definitions appeared in one of the authors blog at http://delay-tolerant-networks.blogspot.com/2014/03/commonly-used-metrics.html.

- **Average hop count**: Let h_i be the hop count for the delivered message $m_i \in M_d$. In other words, the message m_i has passed through h_i nodes before reaching to its destination. Then, the average hop count is determined as $\sum_{i=1}^{|M_d|} h_i / |M_d|$.

It may be noted that the above defined metrics are generic in nature and can be used in any scenario. However, when planning for performance evaluation under specialized metrics, the reader may consider using some additional metrics that are relevant and suitable for such scenarios. For example, if one is working toward detection of misbehaving nodes in an OMN, the degree and accuracy of detection would be metrics of interest in such a scenario.

Delivery ratio of the messages, delivery latency, and overhead ratio are three tightly intertwined metrics. In practice, whenever one of them improves, another one deteriorates. For example, having large number of replicas of a message is helpful to improve its delivery chances. But, at the same time, it also incurs high overhead. The goal of an optimal routing protocol is to design such a scheme that maximizes the delivery ratio, and at the same time, minimizes the average delivery latency and overhead ratio as well. Such an objective is attempted to achieve by utilizing various heuristics, some of which were discussed in this chapter.

2.3.2 General Insights into Routing

While a comprehensive performance evaluation of different routing protocols under various scenarios is out of scope of this text, we look at the performance of a few such protocols under the influence of different buffer capacities.[3] All simulations in this book were performed using the Opportunistic Network Environment (ONE) simulator [51].

Figure 2.5 shows the delivery ratio of messages obtained when SnW was used with different number of copies of the messages (L) and the nodes had different storage capacities. It can be observed that buffer size strongly influenced the delivery ratio when the simulation durations were for 12 h. This is due to the reason that in the absence of any TTL expiry, increased buffer size enabled the nodes to exchange greater number of messages during the available communication opportunities. When the simulation duration was increased, the nodes in the network eventually met with one another, and delivered the messages.

[3]This section has been reproduced by permission of the Institution of Engineering &Technology.
B.K. Saha, S. Misra, "Effects of heterogeneity on the performance of pocket switched networks," *IET Networks*, 2013, vol. 3, no. 2, pp. 110–118.

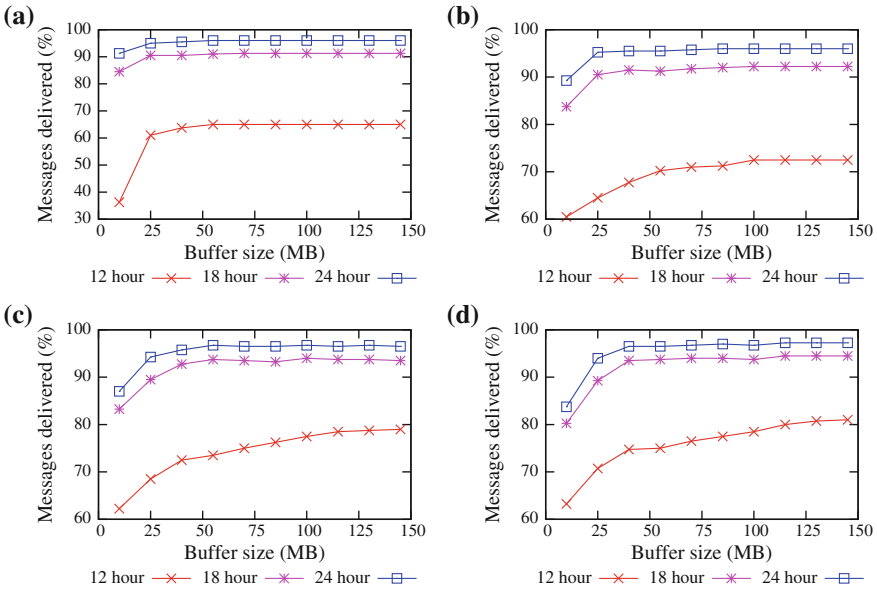

Fig. 2.5 Delivery percentage of the messages for different buffer sizes under SnW with **a** 8-, **b** 16-, **c** 24- and **d** 32-copies

The effect of the number of copies used by SnW on the delivery ratio of the messages is also evident. It could be seen that, as the number of copies increased, for a given buffer size, the delivery ratio also increased. The graphs indicate that unlimited buffer sizes may not help in achieving higher delivery of the messages.

Figure 2.5 also reveals that the rate of increase in the delivery ratio of the messages is faster for buffer sizes up to 40 MB. Although this threshold is specific to the parameters that were considered (message size and rate), this indicates that if more messages are to be delivered within a short time, the nodes should have a certain minimum buffer size.

Figure 2.6 shows the message delivery ratio with different buffer sizes when First contact [51, 76], PRoPHET [19], and Epidemic [18] routing protocols were used. In First-contact routing, a node forwarded (not replicated) any message it had to the first node that it came in contact with. Figure 2.6b, c indicate that with no fixed limit on message replication, larger buffer sizes enhanced the delivery ratio. This is due to the reason that during each communication opportunity, more number of nodes got a copy of a message, and, thereby, increased their respective chances of delivery.

Figures 2.5 and 2.6 provide some insights on how buffer size and message forwarding schemes are related:

- With a fixed limit on message replication, consideration of excessive buffer sizes is not useful. This is reflected by the performances of the SnW and the first-contact routing protocols.

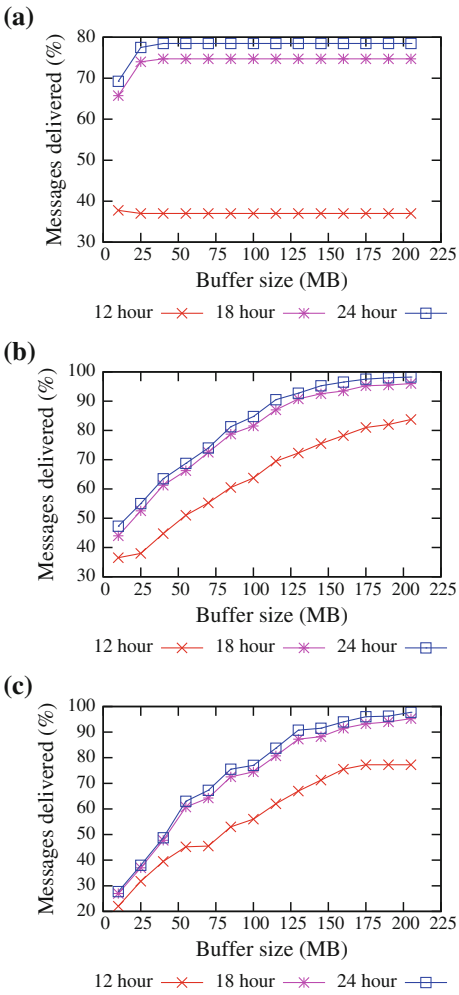

Fig. 2.6 Effects of buffer sizes on the message delivery ratio using **a** First contact, **b** PRoPHET and **c** Epidemic routing

- With no fixed limit on the number of message replicas, larger buffers enhance the delivery ratio. This behavior, verified from the evaluation of the PRoPHET and Epidemic protocols, is due to the reason that during each communication opportunity, more number of nodes get a copy of a message.

Thus, it might be helpful to keep the above observations in mind while developing a routing scheme for use in real life.

2.4 Real-Life Traces

Performance evaluation and model validation through simulations have been an integral part of research in the domain of computer networks. To facilitate the simulation of scenarios where the nodes are mobile, several synthetic mobility models (for example, Random Waypoint and Random Walk) have been proposed. When such models are integrated with the simulation tools, the nodes move according to the stochastic mobility model in the terrain concerned. Therefore, it is possible to add dynamism into the network scenario for example, a MANET with dynamically changing routes between a source and destination pair of nodes.

Although such synthetic models dominated the research domain for long, it was realized that these models, while simple, are idealistic and often do not reflect reality. For example, if one keeps track of the movement pattern of a human being, it would be far from random. Rather, a lot of similarity would possibly be observed in a person's daily movement patterns for example, traveling to the office, visiting a mall, and returning back to home. It may also be noted that in a network with mobile nodes—unlike networks with stationary nodes—the connectivity among the nodes is also governed by their movement. This is particularly true in OMNs, where the nodes typically have short communication ranges, for example, up to 10 m radius of communication for the Bluetooth interfaces. In fact, as noted in an earlier chapter, nodes' mobility is the driving force that enables communication in the otherwise disconnected network.

Kim et al. [77] extracted mobility statistics of the users from a large set of logs captured by several wireless access points over a time period of 13 months. The authors observed that the speed and pause time distribution of the users were log-normal. Moreover, contrary to the assumption of many mobility models, users are unlikely to move along all directions with equal probability. Rather, such mobility distribution is strongly related to the direction of the roads. Rhee et al. [22], on the other hand, analyzed human movement by recording the GPS-based locations of the users over a long period of time. Subsequent analysis revealed that human movement bears similarity with the truncated Levy Walk model, which is also observed in other animals.

> The CRAWDAD (http://crawdad.cs.dartmouth.edu/) repository has a huge collection of real-life traces belonging to different categories including, but not limited to, mobility logs, measurements related to Wi-Fi and WiMax, Bluetooth connectivity, GPS locations, and mobile phone activity records. As of October 2015, the CRAWDAD repository had a collection of 106 data sets, and 7717 registered users from across 111 countries. (Source: CRAWDAD news email list)

Table 2.2 summarizes a few characteristics of some of the data sets available in CRAWDAD. The "dataset" column indicates a contextual name for the concerned

Table 2.2 Summary of selected data sets available at CRAWDAD

Year	Dataset	Trace name	Trace type	Device	Network type	Transmission range (m)	No. of devices	Duration (days)
2005	MIT	Reality Mining [79]	Connection	Phone	Bluetooth, Cellular	10	94	246
2005	Cambridge/Haggle	Infocom'05 [80]	Connection	iMote	Bluetooth	30	41	4
2006	Cambridge/Haggle	Infocom'06 [80]	Connection	iMote	Bluetooth	30	78	4
2006	UPMC	Content [81]	Connection	iMote	Bluetooth	10, 20	54	∼2 months
2007	Nottingham	Cattle [82]	Mobility	Phone	Bluetooth, GPS		7	2
2008	Unimi (University of Milano)	PMTR [83]	Connection	PMTR (Pocket Mobile Trace Recorder)	RTX-RTLP	10	44	19
2008	UMass	Diesel [78]	Connection, Mobility	HaCom Open Brick	IEEE 802.11b		30–40	
2009	UPMC	RollerNet [84]	Connection	iMote	Bluetooth	10	62	0.12
2009	NCSU	KAIST [85]	Mobility	T-mote	GPS receiver		4	
2011	St_Andrews	Sassy [86]	Connection	T-mote	Bluetooth	10	27	79
2014	Roma	Taxi [87]	Mobility	Tablet	GPS		320	∼2 months

set of data, for example, name of the organization conducting the experiment, the place, and so on. Each such dataset consists of one or more traces collected from the corresponding experiment. The "Cambridge/Haggle" dataset has connection traces connected from different experiments, for example, during the INFOCOM conferences in 2005 and 2006. It can be noted from the Table that various kinds of devices have been used for the experiments based on the requirements. For example, in the UMass data set [78], computers fitted to buses were used. However, in other experiments, smaller devices were used since they were meant for carrying by human beings.

Availability of such a huge collection of data sets available at our disposal has altered the simulation dynamics in the related research domain. The use of real-life traces has now become the *de facto* approach toward simulation of OMNs and subsequent performance evaluation. As mentioned earlier, these traces capture a diverse category of information and can be used for appropriate purposes. A commonly observed pattern is the use of device connectivity information. Multiple data sets (for example, Infocom'06 [80] and Sassy [86]) capture information on the Bluetooth device sightings. For example, when a Bluetooth-enabled device B comes in the proximity of another similar device A, the two devices are said to have a "contact" (or encounter). Each device records the earliest time when one detected the presence of the other. The devices often record the contact duration as well—either the actual duration or the time when the other device was last seen. Therefore, from a simulation perspective, such information helps in determining when a link in the network should be up and for how long that should remain up.

As an example, let us consider the following records from the Infocom'06 [80] trace.

```
22  21  9259    9734    1   0
1   4   12641   12768   9   372
1   5   13750   14349   4   266
16  50  20534   20778   2   8958
```

Here, the first column indicates the address of the node which was recording. The second column indicates the address of the node with which the first came in contact. The third column indicates the time when the encounter began, and the fourth column indicates the time when the encounter was over. Therefore, the contact duration (time) for the pair of nodes in the first row was $9734 - 9259 = 475$ s.

In a similar way, one may use the GPS-based location traces (for example, [85]) of the nodes. In such a case, the nodes move as per the pre-specified coordinates in a given simulation terrain. The connection pattern among the nodes is derived from their underlying movement in the terrain. The following describes two popular ways—among others—the real-life traces can typically be used along with a network simulator:

- Make the nodes in the simulator move as per the mobility of the nodes captured via GPS logs. The connection among the nodes occur as dictated by such mobility.

- Ignore mobility of the nodes, but make them connect as per the connection traces available. As noted previously, such traces usually provide the precise time when a pair of nodes come in contact and how long they do stay in contact.

To close this section, one should, however, keep in mind that it may not be possible to use real-life traces in all kinds of scenarios. For example, when one is planning to simulate a post-disaster scenario, it would be difficult to take into consideration how would people communicate or move in a similar scenario in real-life. Therefore, the user should make judicious use of other simulation models in such cases. In Chap. 3, we shall look at how real-life traces can be imported into the ONE simulator and used in network simulations.

2.5 Applications

As noted earlier, DTNs are the suitable mode of communication in different challenged environments such as, deep-space networks and underwater sensor networks. Several implementations of DTN-related protocols are publicly available.[4] Some of the known implementations are the DTN2[5] project, Interplanetary Overlay Network (ION),[6] Postellation,[7] and IBR-DTN.[8] These implementations often come with sample applications for sending and receiving files, news delivery, and so on.

In this section, we discuss about some scenarios for DTN and DTN-like applications [88]. Some of these applications have been implemented and tested in real life. The following list is, however, by no means not exhaustive. A notable omission in this section is the Haggle project [89], which we will cover in details in Chap. 8.

2.5.1 DakNet

The DakNet project [90] has been developed at MIT Media Lab to provide low-cost and energy-efficient communication facilities. DakNet—more of an ad hoc network rather than a DTN—combines asynchronous services together with wireless mode of communication to connect remote villages with towns and cities. DakNet consists of kiosks, mobile access points (MAPs) equipped with portable storage devices and Internet access points or hubs. The MAPs are mounted on—and powered by—vehicles such as bus, motorcycle, bicycle and even ox cart. Whenever such a MAP comes within the proximity of a kiosk, data are transferred between them. Subse-

[4] A detailed list can be found at https://sites.google.com/site/dtnresgroup/home/code.

[5] https://sites.google.com/site/dtnresgroup/home/code/dtn2documentation.

[6] https://ion.ocp.ohiou.edu/.

[7] http://reeves.viagenie.ca/.

[8] https://trac.ibr.cs.tu-bs.de/project-cm-2012-ibrdtn.

quently, when a MAP comes near a hub (possibly in the town or city), data from the kiosks are synchronized with the Internet. The MAPs, therefore, essentially act as data mules. DakNet has been deployed in the villages of India and Cambodia.

2.5.2 Bytewalla

Ntareme et al. [91] developed Bytewalla—an Android application to enable delay-tolerant networking. Bytewalla implements the DTN v2 specification by the DTN research group, and can send/receive data bundles. A user can specify where to store the bundles as well as how much memory to be used. It has provision for sending emails via DTN, which is useful in rural areas with no Internet connectivity. A DTN server receiving such an email converts it into a bundle, and are forwarded to other smartphones using the application. On the other hand, a server extracts the email from a bundle before delivering it to a client.

2.5.3 DTWiki

Du and Brewer [92] implemented DTWiki—a Wiki system operating in DTNs with intermittent connectivity. The DTWiki application runs on the top of TierStore [93]—a distributed file system used to manage file replication, synchronization, and consistency. The backend of DTWiki is not a simple relational database, but consists of several components to ensure data consistency under intermittent connectivity. The data back end stores pages and their revisions with related metadata, user account information, attachments (for example, images and videos) contained in the pages, discussion pages for coordination among the users, facility for searching and indexing content, and mechanism for sharing. Such features help in providing robustness and scalability in the face of frequent network partitions and intermittent connectivity.

2.5.4 DT-Talkie

Islam et al. [94] developed DT-Talkie, a voice-based communication application much like the well-known walkie-talkies. At a high level, DT-Talkie consists of five modules. The first module, audio capture and play, is responsible for capturing the input voice message at the sender's side and storing the encoded voice message in the local file system. At the receiver's end, this module plays a received voice message. The MIME (Multipurpose Internet Mail Extensions) create and parse module creates multipart MIME message at the sender's side containing the encoded voice message and sender's information. At the receiver's end, this module is responsible for parsing the appropriate data from the received bundle. There is also a bundle

send and receive module for creating, sending, and receiving a bundle; a GUI module to let the user interact with the application. Finally, the DTN daemon runs in the background providing necessary service for operations with bundles. Apart from one-to-one "conversations," DT-Talkie also allows group-based communication.

2.5.5 ZebraNet

The ZebraNet [95–97] project was aimed at monitoring the movement and activity of zebras at the Mpala Research Center, Kenya. Such wildlife movement spanned over a large terrain size and a long temporal scale. Although ZebraNet can be considered as a typical project on ad hoc WSNs, it exhibited several challenging requirements. For example, the size and weight of the collars could not be large so that they do not inhibit the movement of the zebras. On the other hand, with collars attached to the necks of zebras, a sparse network topology is created with lower chances of communication. Moreover, since the tracking devices, equipped with GPS sensors, were required to operate over long periods of time without human intervention, relying only on battery sources for power was not feasible. These factors led toward the design of custom-made hardware for the collars. However, from the perspective of network communications, a more stringent requirement was that the base station, which itself was mobile, should receive almost all the tracking data gathered—as close to 100 % as possible. The delay in obtaining all such data was not critical. In other words, the ZebraNet project exhibited delay-tolerant characteristics, where eventual delivery of messages was acceptable. The collars attached to the zebras would normally transfer data in peer-to-peer fashion. Subsequently, the base station received data from the zebras when they came within proximity.

Other related application include Web searching from buses [98], Twimight—a Twitter client for disaster mode [99], and so on. More similar applications are expected to come up and used in the future. In this context, it may be noted that Lindgren et al. [100, 101] have presented valuable insights gathered while deploying and testing DTN systems in real-life. Such anecdote of experiences would be of much value and use to the future developers.

2.6 Summary

Routing is one of the fundamental requirement in any kind of communication network including OMNs. In this chapter, we reviewed some of the popular routing protocols proposed for OMNs along with their variations and improvements. Unlike traditional networks, routing in OMNs involve repeated replications of a message in order to improve its chances of delivery. Some schemes impose an upper limit on replications, whereas others do not. We observed that most of the routing protocols in existence today use some form of intelligence for decision making. They range

from as simple as total count of encounters to complex metrics, such as dynamic delivery predictability. Moreover, many routing protocols exploit the fact that social interactions among people exhibit some long-term patterns.

Subsequently, in this chapter, we formally defined a set of metrics commonly used for performance evaluation in OMNs. These metrics are generic in nature and can be used in any scenario. Some of the context-specific evaluation metrics would be discussed later in this book. Next, we discussed about the real-life traces and their characteristics. As we shall see in the following chapter, these traces can be used in network simulations. Finally, we closed this chapter by reviewing a few applications based on DTNs.

In the remainder of this book, we shall look at different aspects related to the OMNs, where different schemes are proposed and their effect on the network performance evaluated. This chapter, in fact, prepares the ground for introducing the reader to the latter chapters focused on specific research topics.

2.7 Review Terms

- Multi-copy routing
- First-contact routing
- Summary vectors
- Vaccine
- Binary spray
- Transitivity
- Community
- Message overhead ratio
- DakNet
- Direct delivery
- Flooding
- Immunity
- Anti-packet
- Delivery predictability
- Centrality
- Message delivery ratio
- Real-life traces
- Bytewalla

2.8 Exercises

2.1 Write down some metrics relevant to the evaluation of routing protocols. Should these metrics be maximized or minimized to obtain a better performance? Why routing overhead is typically considered in DTNs, but not in the traditional networks, for example, Internet and mobile ad hoc networks?

2.2 Consider the following contact pattern among a set of five nodes:
$\mathscr{C} = \{(A, B, 10), (A, C, 14), (C, D, 16), (B, D, 19), (D, E, 22)\}$,
where $(X, Y, t) \in \mathscr{C}$ indicates that nodes X and Y encountered one another at the time instant t. Further, let the set of messages created be $M = \{(m_1, A, 12), (m_2, B, 15), (m_3, C, 16)\}$, where $(m, X, t) \in M$ indicates that a message with ID m was created by the node X at the time instant t.

Which messages would be carried by the node E after time instant $t = 23$ when Epidemic and Spray and Wait routing protocols are used? Assume that each contact between a pair of nodes lasted for one time unit. Also, assume that SnW is used in binary mode with initial number of copies of a message as 4.

2.3 What are the advantages of SnF over SnW? What similarity does the Spray and Focus and PRoPHET protocols have?

2.4 Support or reject the following claim with justifications.

When the initial number of copies of a message is infinitely large, SnW degenerates into the Epidemic routing protocol. This is true irrespective of whether or not binary mode of spraying is used.

2.5 Assume that the nodes in a network move with a constant speed of 5 m/s. Suppose that the nodes A and j had their last encounter at $t = 1$ s, and the nodes B and j recently encountered at $t = 2$ s. Consider that at $t = 6$ s, nodes A and B come in contact, and the distance between them is 10 m. According to the Spray and Focus algorithm, what would be the value of $\tau_A(j)$ after this contact?

2.6 Assume that the delivery predictabilities between two pairs of nodes (using PRoPHET) just before the time instant t be $P(A, B) = 0.6$ and $P(B, C) = 0.7$, respectively. Further assume that $P(A, C)_{\text{old}}$ is 0.5. At time t the two nodes A and B come in contact. What would be the value of $P(A, C)$ after 120 s from the time instant t if the nodes A and B did not had any subsequent encounter? Assume that each time unit comprises of 60 s.

2.7 Consider some well-known algorithms for finding shortest path in a graph (for example, those proposed by Dijkstra and Floyd–Warshall). What is the challenge in their direct adaptation to OMNs?

2.8 What is a Bloom filter? How Bloom filters can be useful in the context of the vaccine scheme used with Epidemic (or other) routing protocol?

2.9 Suppose that the nodes in an OMN use SnW routing protocol. Let L be a constant that indicates the initial number of copies of any message created in the network. Prove that the overhead ratio in the OMN cannot be greater than L.

2.9 Programming Exercises

2.10 All routing protocols implemented in the ONE simulator are subclasses of the `ActiveRouter` class. However, the simulator also provides another class, `PassiveRouter`, which actually does no routing. Can you think of any scenario where such a nonfunctional router class would be useful?

2.11 The ONE simulator provides an implementation of the Epidemic routing protocol in the `EpidemicRouter` class. Complement the `EpidemicRouter` class with the vaccine scheme as discussed in this chapter. In particular, use a `HashSet` as an instance variable to store the ID of the messages delivered for the concerned node. During any contact, make the two nodes in contact exchange their sets and update accordingly.

2.12 The SnW routing protocol is interesting in the sense that it imposes a upper limit on replication of any message. On the other hand, the efficiency of PRoPHET is in the fact that it leverages the delivery predictability metric to identify better candidates for message delivery.

Develop a new, hybrid routing algorithm, which combines the essence of SnW and PRoPHET together. In other words, each message should have a maximum upper limit, L, on its replication. However, unlike SnW, a message should not be blindly replicated to another node. Rather, a node using the new routing algorithm should replicate to a given node only if the latter has higher delivery predictability for the destination of the concerned message.

http://www.springer.com/978-3-319-29029-4

Opportunistic Mobile Networks
Advances and Applications
Misra, S.; Saha, B.K.; Pal, S.
2016, XXXII, 303 p. 66 illus., 3 illus. in color., Hardcover
ISBN: 978-3-319-29029-4