Chapter 4

Opportunism in Mobile Ad Hoc Networking

Marcello Caleffi
University of Naples Federico II Naples, Italy

Luigi Paura
University of Naples Federico II Naples, Italy

Contents

4.1 Introduction

In the last two decades, great attention has been devoted to the *ad hoc networking* paradigm and there are a large number of routing protocols designed for it. These protocols cover a wide range of design choices and approaches, from simple modifications of traditional solutions for wired networks to more innovative and complex schemes. Most of these protocols [9,12,16,27,28] are based on the *multihop* paradigm, which allows nodes to extend the limited coverage range of wireless communications by exploiting neighbors as cooperative relays.

In fact, direct forwarding allows nodes to communicate only if they are within the direct transmission range of each other. With reference to the simple topology shown in Figure 4.1, if nodes *s* and *d* have to communicate and the link quality is poor, they cannot communicate at all. In contrast, if the network layer adopts multihop forwarding, a pair of nodes can also communicate if they are not within the direct transmission range of each other or if their link quality is poor. With reference to the previous example (Figure 4.2), the neighbor *r* allows *s* to communicate with *d* acting as a relay, that is, by storing in its buffer the packets received from *s* and by sending them to *d*. Clearly, multihop communication can involve multiple relays, and in such a case the step above is repeated until the packet is received at the destination.

For more than a decade, multihop forwarding has been considered a suitable strategy for networking in ad hoc networks, since it well fits in scenarios characterized by dynamic topology with no available infrastructure or central management. However, the main issue with multihop routing is that it tries to *fortify* the scenario so that it behaves like a wired network, instead of exploiting the key features of wireless technology: the broadcasting and the unreliability.

In fact, multihop routing completely hides the broadcast nature of wireless communications in data forwarding by imposing a requirement at the data-link layer that nodes have to discard data packets not directly sent to them, although they have correctly received such packets. Moreover, it usually counteracts the

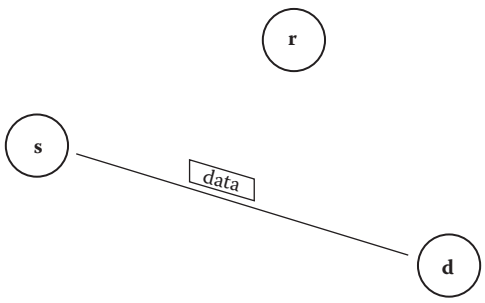

Figure 4.1 Direct forwarding.

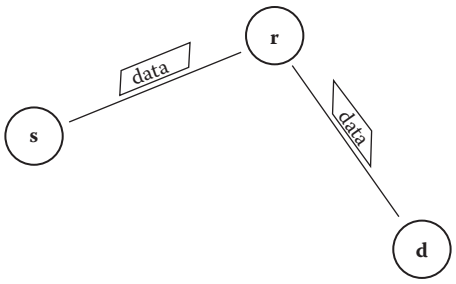

Figure 4.2 Multi-hop forwarding.

time-variant impairment of the wireless propagation by means of Automatic Repeat Request (ARQ) or Forward Error Control (FEC) data-link techniques.

As opposed to *fortifying* the environment, a concept recently proposed is to exploit the good nature of wireless communications, namely the broadcasting, to compensate for the unreliability. This design philosophy—opportunistic routing—aims at relaxing the assumption that the wireless propagation conditions are stationary enough so that they allow a persistent communication among neighboring nodes. In such a way, opportunistic routing is able to provide connectivity in scenarios where traditional ad hoc networking fails. This chapter describes the fundamental characteristics of opportunistic routing (Section 4.2), along with the main features of some representative routing protocols belonging to this class (Section 4.3). The routing protocols have been selected for one or more of the following reasons: (1) they are popular choices in the research community; (2) they may be interesting, illustrative examples of this class; (3) they may have unique features that make them interesting. In the following we do not make comparisons among the considered protocols, since there are many published performance comparisons [20,23,30,35–38]. Moreover, the citations for the considered protocols themselves often provide performance evaluations of the protocol.

4.2 Opportunistic Routing

As mentioned in the previous paragraph, opportunistic routing is a class of routing protocols that, rather than counteracting, tries to take advantage of the time-variant nature of the environment to provide end-to-end connectivity in scenarios where traditional networking fails. Let us provide an example of how opportunistic routing works by comparing it with traditional ad hoc routing.

If the network layer adopts the multihop routing paradigm, data communications are unicast. Therefore, the next hop for each packet has to be singled out before sending that packet on the link, i.e., the next hop selection happens at the sender side. Clearly, this strategy is not optimal since routing progress, i.e. the

progress of a data packet toward the destination, is achieved only if the packet is received by the selected next hop. In other words, traditional ad hoc routing is unable to exploit the opportunity offered by an unselected relay closer to the destination than the source.

With reference to the example reported in the previous paragraph (Figure 4.2), where the source *s* has to communicate with *d* and *r* is a neighbor of *s* that is closer* to *d* than *s* itself, the source can select *r* or *d* as the next hop. If we suppose that *s* selects *r*, i.e., it selects the most reliable link, it will be unable to take advantage of favorable wireless propagation conditions that allow *d* to receive the packet. If we suppose that *d* has been selected as the next hop and that the packet reaches *r* but not *d*, again no routing progress is made. In contrast, opportunistic routing requires that the source simply broadcast the data packets without worrying about next hop selection, which happens at the receiver side. In such a way, routing progress is achieved every time that a node closer to the destination than the source receives the packet, and that node becomes the next hop. With reference to the above example, if the packet is received by the destination, i.e., if favorable propagation conditions exist, significant routing progress is achieved. However, if only node *r* receives the packet, it becomes responsible for packet forwarding and routing progress is achieved anyway. Clearly, if neither *r* nor *d* receive the packet, *s* retains the responsibility for packet forwarding.

We note that ad hoc networks exhibit an inherent distributed spatial diversity, due to both node mobility and wireless propagation instability. In fact, with respect to node mobility, an opportunity happens when the selected next hop moves outside the sender's transmission range (and the packet is received by a less favorable neighbor), or when a packet is received by a more favorable neighbor that has moved into the sender's transmission range but has not yet been recognized as a neighbor by the neighbor discovery mechanism. Moreover, in wireless propagation, an opportunity happens when the propagation conditions of the selected link worsen during or just before the packet transmission (and the packet is received by a less favorable neighbor), or when a packet is received by a more favorable neighbor along a link previously classified as unreliable.

Opportunistic routing allows knows to exploit such opportunities to increase the probability that packets progress toward their destinations as covered in Section 4.2.1. However, these opportunities are not "free" since they require coordination among nodes, which means routing overhead as will be pointed out in Section 4.2.2.

* Here and in the following text, we use the term closer to indicate that a node is a preferable forwarder with respect to another node.

4.2.1 Benefits

Opportunistic routing differs from traditional ad hoc routing in that it exploits the broadcast nature of a wireless medium by deferring the route selection to the receiver side. Clearly, this feature copes well with unreliable and unpredictable wireless transmission, and in this section we describe the two main advantages of opportunistic routing in the presence of unreliable links—*multiuser diversity* and *route opportunism*.

In the following we assume the presence of perfect coordination among the nodes and we neglect the additional overhead introduced by opportunistic routing. Such issues will be discussed in the next section.

4.2.1.1 Multiuser Diversity

Opportunistic routing exploits multiuser diversity, that is, the availability of multiple neighbors whose links can be modeled as statistically independent channels [38] in order to manage the unreliability of wireless communications. As an example, consider the diamond topology depicted in Figure 4.4 where the source s can reach the destination d through five relays r_i, and $d_{i,j}$ is the average delivery ratio of the packets sent by i to j:

$$d_{s,r_i} = 0.2 \quad \forall i \tag{4.1}$$

$$d_{r_i,d} = 1.0 \quad \forall i. \tag{4.2}$$

Since traditional ad hoc routing selects the next hop at the sender side, it is unable to take advantage of a transmission that reaches a node other than the selected one. So, the average number of link transmissions $\overline{n}_{s,d}(s,r_i,d)$ to deliver a packet from s to d along the path (s, r_i, d) is:

$$\overline{n}_{s,d}(s,r_i,d) = \overline{n}_{s,r_i} + \overline{n}_{r_i,d} = \frac{1}{d_{s,r_i}} + \frac{1}{d_{r_i,d}} = \frac{1}{0.2} + \frac{1}{1} = 6 \quad \forall i. \tag{4.3}$$

On the other hand, opportunistic routing generalizes the multihop paradigm by means of *cooperative relaying* (Figure 4.3): the source treats all the available relays as a unique unit that cooperatively forwards the packet to the destination. In other words, the next hop selection is postponed at the receiver side, allowing it to take advantage of a transmission that reaches whichever neighbor. In such a way, with reference to the previous example and assuming that the link success

PE: Figures 4.4 and 4.3 are not referenced in order. OK?

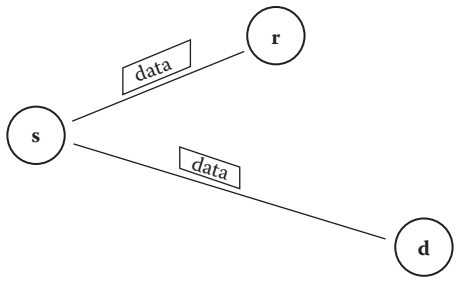

Figure 4.3 Opportunistic forwarding.

events are statistically independent, the combined link (s,r_o) has the following delivery ratio:

$$d_{s,r_o} = 1 - \prod_{i=1}^{5} (1 - d_{s,r_i}) = 1 - (1 - 0.2)^5 = 0.67 \tag{4.4}$$

and thus the average number of link transmissions $\overline{n}_{s,d}(s,r_o,d)$ to deliver a packet from s to d using opportunistic routing is:

$$\overline{n}_{s,d}(s,r_o,d) = \overline{n}_{s,r_o} + \overline{n}_{r_o,d} = \frac{1}{d_{s,r_o}} + \frac{1}{d_{r_o,d}} = \frac{1}{0.67} + \frac{1}{1} = 2.49, \tag{4.5}$$

which allows opportunistic routing to achieve 2.4 times the traditional routing throughput.

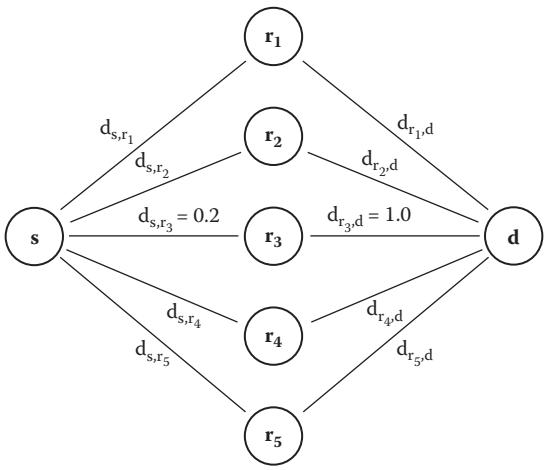

Figure 4.4 Diamond topology.

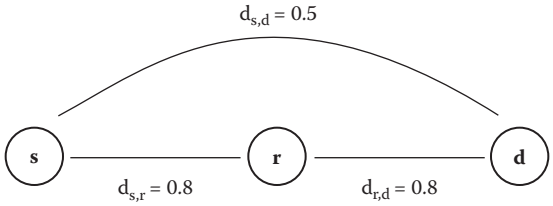

Figure 4.5 Linear topology.

4.2.1.1.1 Route Opportunism

Traditional ad hoc routing tries to fortify the wireless channel so that it behaves like a wired channel by selecting links with the highest delivery ratios [10]. This choice often involves a trade-off between link quality and routing progress.

Let us consider the linear topology shown in Figure 4.5 where node *s* sends a packet to *d* along one of the possible paths {(*s,d*); (*s,r,d*)}. Traditional routing singles out the next hop at the sender side. So, if *r* is chosen as the next hop, the link quality is good and no retransmission is required with probability $d_{s,r} = 0.8$, but the routing progress is small. Alternatively, if the final destination is chosen as the next hop, the highest routing progress is achieved if the packet reaches the destination; however, since the link quality is poor, the probability of single transmission is just $d_{s,d} = 0.5$. The average number of link transmissions $\overline{n}_{s,d}$ to deliver a packet from *s* to *d* depends, of course, on the routing strategy, that is, it depends on the selected route:

$$\overline{n}_{s,d}(s,r,d) = \overline{n}_{s,r} + \overline{n}_{r,d} = \frac{1}{d_{s,r}} + \frac{1}{d_{r,d}} = \frac{1}{0.8} + \frac{1}{0.8} = 2.50$$

$$\overline{n}_{s,d}(s,d) = \overline{n}_{s,d} = \frac{1}{d_{s,d}} = \frac{1}{0.5} = 2.0 \tag{4.6}$$

In contrast, in opportunistic routing the sender broadcasts the packet, allowing it to pick as a relay the node closest to the destination among the nodes that receive the packet. In this way, it is able to opportunistically leverage unexpected paths related to node mobility and/or changes in wireless propagation conditions; in other words, it exploits *route opportunism*.

With reference to the previous example, let us denote with $e_{i,j}$ the event "node *j* is the closest node to the destination among those that have received the packet sent by *i*" and denote with $e_{i,j}$ the event "no one node has received the packet sent by *i*." Clearly, the event $e_{i,j}$ represents the amount of progress toward the destination reached by the packet.

At the first transmission, we have three possible mutually exclusive events: $e_{s,d}$, $e_{s,r}$ and $e_{s,s}$, and the related probabilities are:

$$\begin{aligned} p_{s,d} &= P(e_{s,d}) = d_{s,d} = \frac{1}{2} \\ p_{s,r} &= P(e_{s,r}) = (1-d_{s,d})d_{s,r} = \frac{1}{2}\frac{4}{5} \\ p_{s,s} &= P(e_{s,s}) = (1-d_{s,d})(1-d_{s,r}) = \frac{1}{2}\frac{1}{5}. \end{aligned} \tag{4.7}$$

If $e_{s,d}$ happens, the packet has reached the destination and no additional transmissions are required. Otherwise, a second transmission is needed, and if $e_{s,r}$ occurs, the possible events are $e_{r,d}$ and $e_{r,r}$ with probabilities respectively:

$$\begin{aligned} p_{r,d} &= P(e_{r,d}) = d_{r,d} = \frac{4}{5} \\ p_{r,r} &= P(e_{r,r}) = 1-d_{r,d} = \frac{1}{5}. \end{aligned} \tag{4.8}$$

In contrast, if $e_{s,s}$ happens, the events and the probabilities are the same as the first transmission, Equation (4.7). Further transmissions follow the same event flow, as shown in Figure 4.6, where the number of links from the root to a leaf accounts for the number of transmissions.

By exploring all the branches of the event flow, after simple algebraic manipulations the average number of opportunistic link transmissions $\overline{n}_{s,d}(s,n_o,d)$ for the considered topology is:

$$\begin{aligned} \overline{n}_{s,d}(s,n_o,d) &= \sum_{i=1}^{\infty}(p_{s,s})^{i-1}(ip_{s,d} + p_{s,r}\sum_{j=1}^{\infty}(i+j)p_{r,d}(p_{r,r})^{j-1}) = \\ &= \frac{1}{1-p_{s,s}}\left[\frac{p_{s,d}}{1-p_{s,s}} + \frac{p_{s,r}p_{r,d}}{1-p_{r,r}}\left(\frac{1}{1-p_{s,s}} + \frac{1}{1-p_{r,r}}\right)\right] = \\ &= 1.8381. \end{aligned} \tag{4.9}$$

Therefore, opportunistic routing outperforms traditional routing and it can be shown that the throughput gain increases with the number of links exploited by the routing procedure [3].

We note that the selection of the next hop at the receiver side is the distinguishing feature that differentiates opportunistic routing from multipath routing

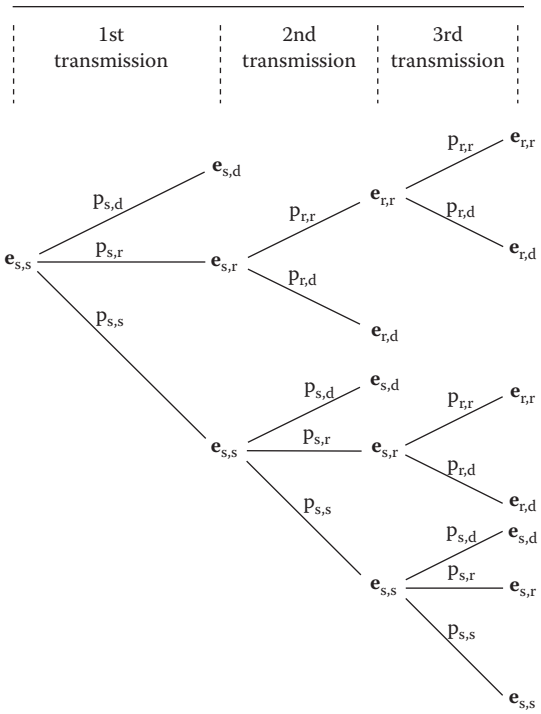

Figure 4.6 Opportunistic routing flow.

[6,21,24,31,32]. In fact, both exploit the spatial diversity (i.e., the availability of multiple routes to increase the throughput and to gain resilience against unreliable links), but since multipath routing singles out the next hop at the sender side, it exploits only a subset of the opportunities offered by the wireless propagation.

4.2.2 Challenges

The major challenge in opportunistic routing is to maximize the routing progress of each data transmission toward the destination without causing duplicate transmissions or incurring significant coordination overhead.

In order to achieve the potential benefits of opportunistic routing and avoid the above-mentioned problems, an effective protocol should implement the following tasks according to a distribute strategy:

1. candidate selection
2. forwarder election
3. forwarding responsibility transfer
4. duplicate transmission avoidance

Candidate selection guarantees that among all the neighbor nodes, only those closer to the destination than the actual forwarder can potentially become the next hop. In fact, it is completely useless to send a packet toward nodes farther from the destination than the actual forwarder, since in this case no routing process would be achieved at all. We note that the more accurate the forwarder election, the less coordination overhead is required for the responsibility transfer phase.

The *forwarder election* provides a mechanism to single out, among all the candidates that have successfully received the packet, the one that is closest to the destination. In other words, the forwarder election allows the selection of the next hop at the receiver side. Clearly, the more accurate the forwarder election is, the more the throughput increases.

The *forwarding responsibility transfer* allows the nodes involved in the forwarding process—the actual forwarder plus the candidates—to become aware of the winner of the election. The responsibility transfer is the distinguishing feature that differentiates opportunistic routing from flooding. In fact, in both opportunistic routing and flooding, multiple nodes receive the packet. However, unlike flooding, opportunistic routing allows only one node at a time to be in charge of packet forwarding. The more effective the responsibility transfer is, the less duplicate transmissions happen, and thus, less overhead is generated by the duplicate transmission avoidance mechanism.

Finally, *duplicate transmission avoidance* is required only in cases of imperfect responsibility transfer. If the forwarding responsibility is correctly transferred to the winning forwarder, there is only one node in charge of packet forwarding at any one time. In contrast, several packet transmissions occur but only one is innovative, i.e. the one made by the winning forwarder. In such a case, a mechanism is necessary to stop useless transmissions, and the more the mechanism is effective the less network throughput is wasted.

Figures 4.7 through 4.10 present an example of the different tasks in which a node *f* has to forward a packet toward *d* and it has 5 neighbors around it, namely

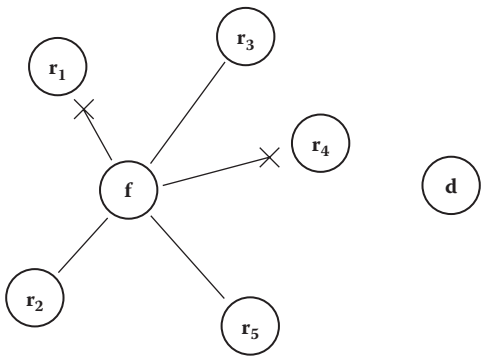

Figure 4.7 Broadcast packet forwarding.

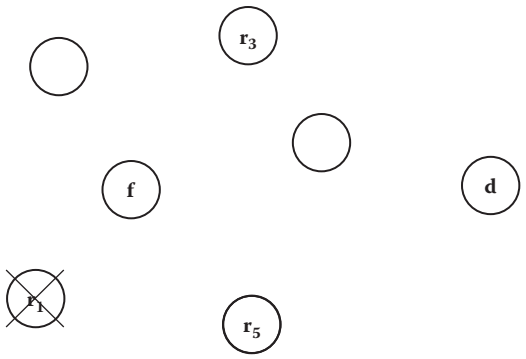

Figure 4.8 Candidate selection.

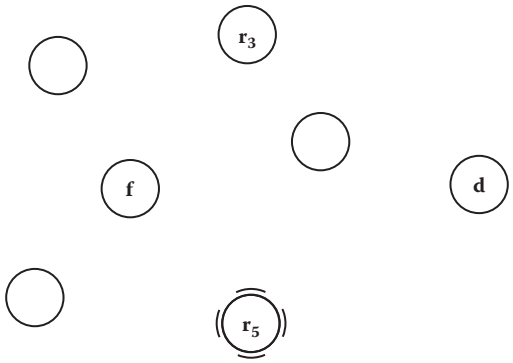

Figure 4.9 Forwarder election.

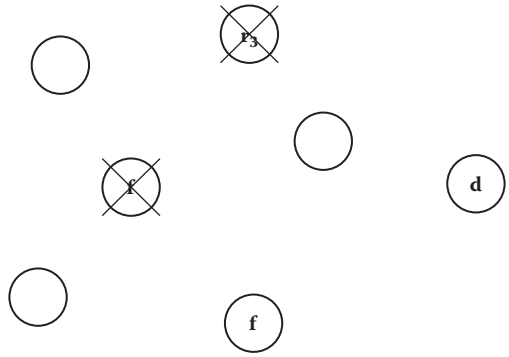

Figure 4.10 Responsibility transfer.

r_1–r_5. First, the forwarder broadcasts the packet, which is received only by nodes r_1, r_3, and r_5 (Figure 4.7). Then, the candidate selection task singles out as possible forwarders only nodes r_3 and r_5 (Figure 4.8), since r_1 is farther from the destination than *f*. Among the candidates, the closest node is r_5, which wins the forwarder election and becomes the next forwarder (Figure 4.9). Finally, the responsibility transfer informs the previous forwarder along with the losing candidate that r_5 won the election (Figure 4.10).

4.3 Opportunistic Protocols

In this section we describe the main features of some representative opportunistic routing protocols. The Extremely Opportunistic Routing (ExOR) (Section 4.3.1) is the most popular opportunistic routing protocol, while both MORE (Section 4.3.3) and MIXIT (Section 4.3.4) generalize the opportunistic routing paradigm by adopting, respectively, packet-level and symbol-level network coding. Simple Opportunistic Adaptive Routing (SOAR) (Section 4.3.2) proposes a simple packet-level responsibility transfer process based on time division multiple access (TDMA). The same mechanism is adopted by Opportunistic DHT-based Routing (ODR) (Section 4.3.6), but it is the first protocol to propose a scalable mechanism to distribute loss rate estimates across the network. Finally, the Multi-Channel Extremely Opportunistic Routing (MCExOR) protocol (Section 4.3.5) extends opportunistic routing to multichannel environments.

AU: Expansion correct?

In Section 4.3.7 the main characteristics of the considered protocols are summarized, and a qualitative comparison is offered covering the advantages and drawbacks of each.

4.3.1 Extremely Opportunistic Routing

Extremely Opportunistic Routing (ExOR) [1,2] is the most popular opportunistic routing protocol and one of the first protocols proposed to exploit the broadcast nature of wireless communications for increasing resilience and throughput.

ExOR assumes that the estimates of the path loss rates for each pair of nodes are available at each node. Such loss rates are evaluated by means of a metric similar to that of Expected Transmission Count (ETX) [10]. Although the authors suggest using a link-state flooding technique to distribute loss rate estimates across the networks, in the performance evaluation they do not account for it by resorting to a simple centralized mechanism for loss rate distribution.

To contain the overhead due to the forwarding responsibility transfer mechanism, ExOR operates on batches of packets, that is, the receiving nodes buffer the packets until the end of the batch. Clearly this increases the end-to-end delay and makes ExOR unsuitable for real-time applications. Moreover, the authors point out that the batches could badly interact with the TCP congestion avoidance mechanism, since in the presence of low loss rates, the window's size would limit the batch sizes.

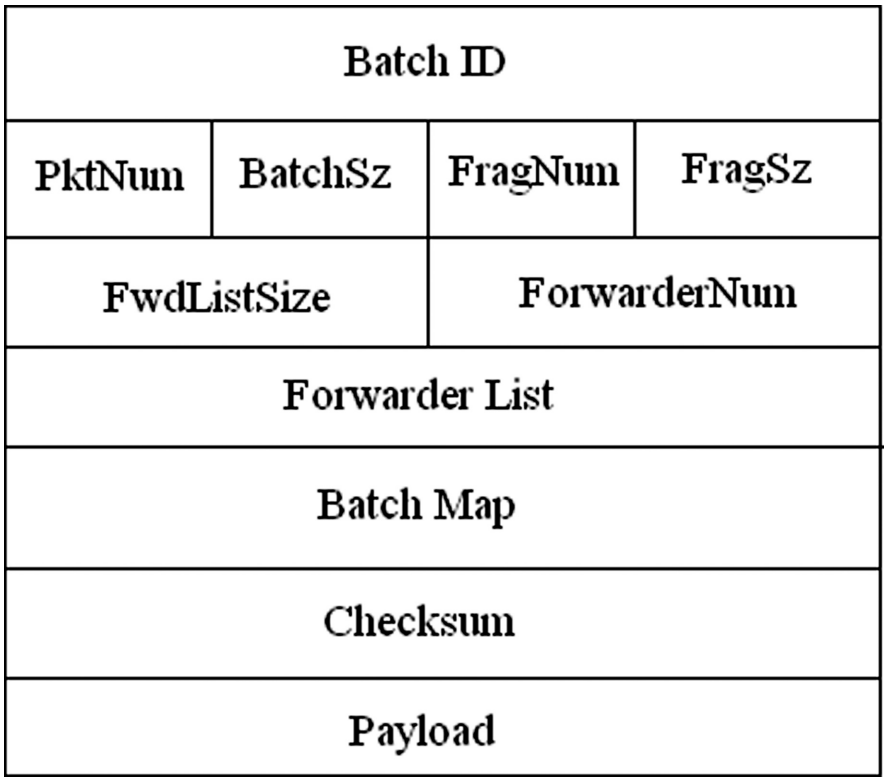

Figure 4.11 ExOR packet header.

AU: Please check Figure 4.11.

The loss rates are used for both candidate selection and the forwarder election. According to ExOR, the sender must include in the header of each packet the list of candidates (Figure 4.11), namely, the *forwarder list*, prioritized by closeness to the destination according to the ETX-like metric. For a given batch, the forwarder list never changes. Thus, both the candidate set and the forward election are predetermined at the sender side during the transmission of the first packet of the batch. Clearly, this could potentially reduce the opportunism of the protocol.

The forwarder responsibility transfer mechanism implements an implicit strategy based on the *batch map* field in the packet header (Figure 4.11). This field lists, for each packet in the batch, the sender's best guess of the highest priority node that has received such a packet. From an operational point of view, when a node receives a packet it first checks if it is included in the forwarder list. If so, it first buffers the packet and then updates its local batch map by replacing an entry if the packet's header indicates a higher-priority node. If the header does not include a higher-priority node, it simply discards the packet. The batch map acts like a gossip

mechanism, carrying reception information from higher-priority nodes to lower-priority nodes.

When the batch is complete, each candidate forwards the packets not yet acknowledged by the highest priority candidates. Clearly, each forwarded packet also acknowledges the packets already received by means of the batch map stored in its header.

The timing among candidates is implemented by means of local timers, whose expire times are estimated by nodes using the header fields. Each candidate first estimates the sender's transmission rate using an exponential weighted moving average (EWMA) filter, and then uses that rate to estimate the batch end time. The candidate with the highest priority will start forwarding packets at the batch end time. It will also delay its transmission according to its priority.

Such a TDMA strategy avoids collisions among candidates since ExOR exploits marginal links where carrier sense often does not operate satisfactory. However, it introduces considerable overhead and for this reason ExOR operates on batches of packets and the candidates do not forward any packet when the batch map indicates that over 90% of the batch has been received by higher-priority nodes. Moreover, a major drawback of ExOR forwarding responsibility transfer is that its overhead is proportional to the number of candidates. For this reason, ExOR limits the candidate set size by selecting the nodes whose ETX metric does not exceed a certain threshold. Another issue related to this TDMA strategy is that it prevents the candidates from exploiting spatial bandwidth reuse by allowing only one transmission at a time.

ExOR duplicate transmission avoidance is a passive distribute procedure based on the gossip mechanism implemented by the batch lists. Since there is no explicit cancellation of redundant transmissions, a candidate can need several responsibility transfer phases to become aware that its buffered packets are not innovative.

4.3.2 Simple Opportunistic Adaptive Routing

The Simple Opportunistic Adaptive Routing (SOAR) protocol [29] tries to solve one of the issues of ExOR, the lack of support for multiple simultaneous flows due to batch processing, by introducing an explicit forwarding responsibility transfer.

Like ExOR, SOAR implements a predetermined candidate selection process based on the estimates of the path loss rates for each pair of nodes according to the ETX metric. The candidate set, namely the *forwarder list*, is included in the packet header, and is prioritized by closeness to the destination.

When a candidate receives a packet, it stores the packet in a buffer and sets a timer based on its priority (i.e., its position in the forwarder list). The higher the candidate priority is, the earlier the timer will expire. Since when a timer expires the node broadcasts the packet, the other candidates can become aware that a node closest to the destination is in charge of packet forwarding, and will therefore discard the packet.

Thus, like ExOR, both the candidate selection and the forwarder election processes are predetermined at the sender side on the basis of the loss rates. However, unlike ExOR, the SOAR forwarder responsibility transfer process implements an explicit acknowledgment strategy based on the packet reception. Moreover, while ExOR implements a batch-level acknowledgment, SOAR adopts a packet-level acknowledgment, that is, each candidate becomes aware of the winner of the election of each packet.

Clearly, the priority-based timers require that all the candidates can hear each other. To ensure this condition, SOAR selects the allowed candidates at the sender side in order to avoid diverging routes. The candidate selection consists of two phases: (1) shortest-path candidate selection, that is, the selection of the nodes belonging to the shortest-path, and (2) near shortest-path candidate selection, that is, the selection of additional nodes that allow an increase in opportunities, but at the same time do not produce diverging routes.

Assume that node *i* belongs to the shortest route between the source and the destination (i.e., its distance to the destination is the shortest one). Then, *i* will select neighbor *j* as a candidate if all of the following conditions hold:

1. *j* is closer (according to the ETX metric) than *i* to the destination
2. the quality of the link between *i* and *j* is above a certain threshold
3. the qualities of the links between *j* and each other candidate are above the threshold

The first constraint ensures routing progress, while the second and the third conditions assure that the actual forwarder plus the candidate set are connected to avoid duplicate transmissions. However, these constraints do not avoid the presence of diverging routes if they are used by nodes that do not belong to the shortest path.

As an example, let us consider the topology depicted in Figure 4.12, where node *s* wants to send a packet to node *d*. According to the previous constraints, *s* selects as

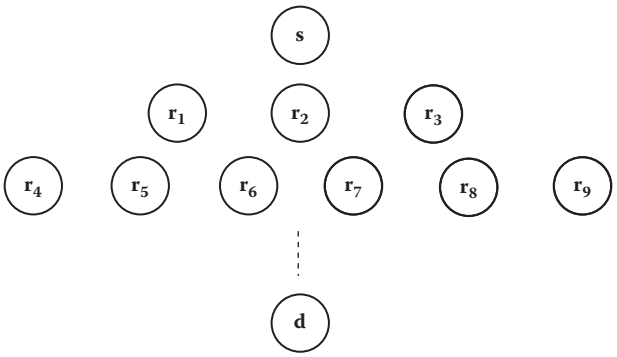

Figure 4.12 SOAR example.

candidates the nodes r_1, r_2, and r_3, r_1 selects the candidates r_4, r_5, and r_6, and r_3 selects the candidates r_7, r_8, and r_9. If it happens that r_1 and r_3 do not hear each other forwarding the packet due to some packet loss, they each forward a copy of the packet. Since r_4 is far away from r_9, these two nodes further perform duplicate forwarding and the paths will further diverge and yield many duplicate transmissions.

For these reasons, nodes that do not belong to the shortest path use the following additional constraints for candidate selection. Assuming that node i does not belong to the shortest path, i first determines the node belonging to the shortest path that is closest to the destination, say node j. Then, j becomes a candidate for i if it is closer than i is to the destination. Moreover, for each node k in j's forwarding list, i adds k in its candidate set if:

- k is closer than i to the destination
- the quality of the link between i and k is above a certain threshold

By applying the above constraints to the example reported in Figure 4.12, we observe that even if r_1 and r_3 perform duplicate forwarding, since their forwarding lists include only r_6 and r_7, the routes do not further diverge.

Besides the implicit duplicate transmission avoidance based on the diverging route prevention, SOAR also implements an explicit mechanism based on selective and piggybacked acknowledgments (ACKs). The ACKs are selective since the same ACK can acknowledge multiple data packets, and they are piggybacked because if there is a data packet in the queue, the acknowledgment is stored in the data packet header, limiting the throughput related to the duplicate transmission avoidance.

We note that SOAR shares some similarities with the MAC-layer anycast mechanism proposed in [14], where a sender sends an RTS packet, and multiple receivers respond to the RTS according to their closeness to the destination. However, in [14] the reception of the RTS does not guarantee the reception of the subsequent data packet.

AU: Please spell out – 1st mention.

4.3.3 MORE

The MORE protocol [8] has been proposed to overcome the issues related to ExOR forwarding responsibility transfer, mainly the lack of spatial reuse. The key feature of MORE is the adoption of network coding at the packet level (intra-flow), and in the following we provide two examples to show the synergy between opportunistic routing and network coding.

In the first example, we consider the linear topology shown in Figure 4.13, where node s has to send two packets to node d, namely p_1 and p_2. While traditional ad hoc routing sends the packets unicast along one of the paths $\{(s, d); (s, r, d)\}$ by selecting the next hop at the sender side, opportunistic routing simply broadcasts the packets and the next hop selection happens at the receiver side. However,

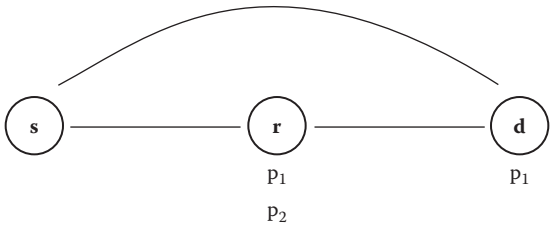

Figure 4.13 MORE unicast example.

opportunistic routing requires a certain amount of coordination, which introduces additional overhead. In fact, if we suppose that *r* receives both the packets but only one has been overheard by the destination, *r* has no way to guess which packet it has to forward.

MORE exploits the network coding to solve such an issue: *r* simply sends a random linear combination of the received packets p_1 and p_2—the sum $p_1 + p_2$—and the destination will retrieve the missing packet without any additional coordination. In other words, MORE adopts the network coding to accomplish the *forwarder responsibility transfer*.

The second example (Figure 4.13) illustrates a multicast transmission: the source *s* has to multicast four packets—p_1 through p_4, to three nodes, r_1 through r_3. We assume that each node receives the packets shown in the figure. Without network coding, the source has to retransmit all four packets. However, with network coding, it is sufficient to transmit two linear combinations of the four packets, which will be used by the destination to retrieve the original packets. For example, if the sender sends:

AU: Should this be 4.14? If not, please mention Figure 4.14 in text.

$$p_1' = p_1 - p_2 + p_3 - p_4$$

$$p_2' = p_1 + 2p_2 + p_3 + 3p_4$$

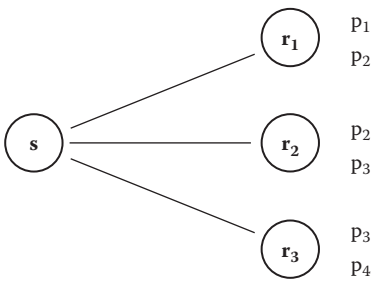

Figure 4.14 MORE multicast example.

the node r_1, which has received p_1, p_2, p_1' and p_2', retrieves all the original packets by inverting the matrix of coefficients and multiplying it with the received packets, as follows:

$$\begin{pmatrix} p_1 \\ p_2 \\ p_3 \\ p_4 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & -1 & 1 & -1 \\ 1 & 2 & 1 & 3 \end{pmatrix}^{-1} \begin{pmatrix} p_1 \\ p_2 \\ p_1' \\ p_2' \end{pmatrix} \tag{4.10}$$

which reduces the number of retransmissions from four packets to just two.

MORE shares several features with ExOR. Both protocols implement a predetermined candidate selection process based on the estimates of the path loss rates for each pair of nodes, and both adopt the ETX metric [10] to estimate such loss rates. Both include the *forwarder list* in the packet header, prioritized by closeness to the destination, and both operate on batches of packets. Finally, both limit the candidate set size to reduce the overhead.

However, unlike ExOR, the forwarder election process allows multiple nodes to forward the packets. In fact, when a node receives a packet, it first checks whether it is in the packet's forwarder list. If so, the node checks if the packet is an *innovative* one, that is, whether it is linearly independent of the packets of the same batch previously received. If both conditions are satisfied, the node stores the packet in the buffer and broadcasts a linear combination of the received packets.

Thus, MORE does not implement any forwarder election within the candidate set, and the forwarding responsibility transfer is implicit, i.e., it is based on the packet reception event. As a distinguishing feature of MORE, the classical CSMA/CA strategy offered by the 802.11 MAC layer is used to avoid collisions among forwarder nodes.

AU: Spell out – 1st mention.

Another difference between MORE and ExOR is that each packet sent by MORE is a coded packet, i.e., a linear combination of all the packets in the batch. Therefore, a duplicate transmission occurs every time a packet is linear dependent from the packets previously received. MORE does not use any explicit strategy to avoid duplicate transmissions, since there is no explicit cancellation of redundant transmissions. Instead, it resorts to the path loss rates to estimate the number of transmissions needed to forward a packet to a node closest to the destination, and such estimates implicitly limit duplicate transmission events. Each time that a packet is received from the farthest node, a *credit counter* is incremented by such an estimate, and each time that the node forwards a packet, its credit counter is decremented by one.

AU: Please clarify.

An explicit acknowledgment strategy is used to notify the source that a batch is correctly received by the destination, and the ACK is routed using traditional unicast routing. Clearly, batch size affects the MORE overhead because the smaller the

batch sizes, the more frequent the ACKs. Moreover, the batch size also affects the duplicate transmission occurrence because the smaller the batch, the more likely the duplicate transmission event.

4.3.4 MIXIT

The MIXIT [18,19] protocol extends the network coding proposed by MORE at the symbol level with three differences. First, MIXIT deals with packets with errors, while MORE does not. Second, MIXIT network coding is an end-to-end rateless error-correcting code while MORE network code cannot correct errors. Third, MIXIT designs a MAC that exploits looser constraints on packet delivery to significantly increase concurrent transmissions, while MORE carrier sense requires correct packet delivery, preventing it from achieving high concurrency.

The key insight MIXIT is that, by insisting on receiving fully correct packets, traditional protocols are missing the bulk of their opportunities. In fact, over long links it is hard to receive the whole packet correctly. On the other hand, it is likely that each symbol will be received correctly by some node thanks to spatial diversity [25]. MIXIT opportunistically exploits such partial packets received at the intermediate nodes to assemble them into a complete packet at the destination, thus increasing the network throughput.

AU: Please clarify.

The assumption behind MIXIT is the availability at the physical layer of a confidence measure for each decode symbol [15,33]. This allows the nodes to identify which symbols in a corrupt packet are likely correct and to forward them. More specifically, a symbol is error free if its confidence value is above a threshold, γ, and faulty otherwise, since as γ increases, the probability that a symbol is corrupted becomes vanishingly small [33].

Such an assumption is used to define a symbol-level network coding that also works as a rateless error-correcting code, addressing one of the main challenges in opportunistic routing: duplicate transmission avoidance. In fact, with network coding, nodes forward random linear combinations of their correctly received symbols, reducing the probability of duplicate transmission. Moreover, since MIXIT exploits symbol-level network coding by forwarding symbols belonging to corrupt packets, there is a chance that a forwarded symbol is incorrect. Therefore, duplicate transmissions provide an amount of redundancy to correct corrupted symbols; in other words, they behave as a rateless error-correcting code. To provide an example of how MIXIT works, let us consider the scenario in Figure 4.15, where a source s wants to deliver two packets, namely p_a and p_b, to the destination d, and where there are two possible relays, r_1 and r_2. We assume that when the source broadcasts p_a and p_b, the nodes in the network receive some corrupted symbols, and in particular the relays receive less corrupted symbols than the destination. Figure 4.15 illustrates such corrupted symbols using grey cells. Due to spatial diversity [25], however, the few corrupted symbols received by r_1 and r_2 are unlikely to be in the same locations.

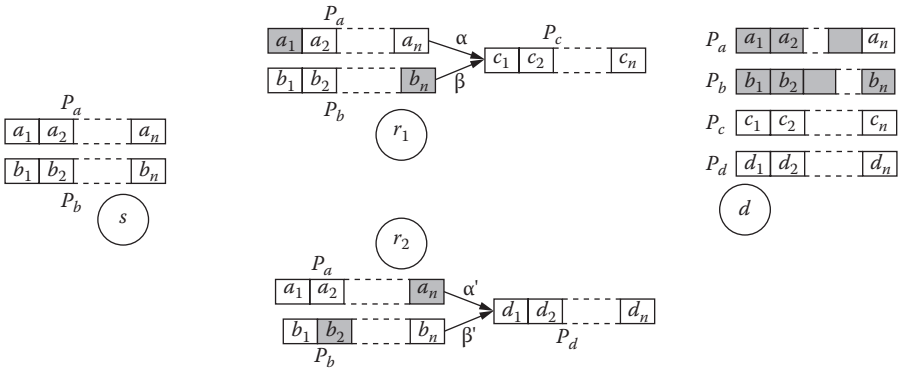

Figure 4.15 MIXIT example.

Since the faulty symbols can be recognized thanks to the confidence measure, according to MIXIT the nodes forward linear combinations of the received error-free symbols. In particular, if we assume that a_i and b_i are the i-th correct symbols in p_a and p_b, respectively, the node r_1 picks two random numbers α and β and creates a coded packet p_c where the i-th symbol, c_i, is computed as follows:

$$c_i = \begin{cases} \alpha a_i + \beta b_i & \text{if } a_i \text{ and } b_i \text{ are clean simbols} \\ \alpha a_i & \text{if } a_i \text{ is clean and } b_i \text{ is faulty} \\ \beta b_i & \text{if } a_i \text{ is faulty and } b_i \text{ is clean} \\ \text{none} & \text{if } a_i \text{ and } b_i \text{ are both faulty} \end{cases} \tag{4.11}$$

Similarly, r_2 generates a coded packet p_d by picking two random values α' and β' and applying the same logic in the above equation.

When r_1 and r_2 broadcast their respective packets, p_c and p_d, the destination receives corrupted versions where some symbols are incorrect as shown in Figure 4.15. Thus the destination has four partially corrupted receptions: p_a and p_b, directly overheard from the source and containing many erroneous symbols, and p_c and p_d, which contain a few erroneous symbols. For each symbol at the i-th position, the destination needs to decode two original symbols a_i and b_i. As long as the destination receives two uncorrupted independent symbols in location i, it will be able to perform the decoding [13]. For example, with respect to the second symbol, the destination has received:

$$\begin{aligned} c_2 &= \alpha a_2 + \beta b_2 \\ d_2 &= \alpha' a_2 \end{aligned} \tag{4.12}$$

Given that the header of a coded packet contains the multipliers (e.g., α and β), the destination has two linear equations with two unknowns, a_2 and b_2, which are

easily solvable. Once the destination has decoded all symbols correctly, it broadcasts an ACK, causing the nodes to stop forwarding packets.

Apart from the symbol-level network coding, MIXIT behaves identically to MORE. Therefore, both share the same advantages and the same issues. We note also that besides MORE and MIXIT, other recently proposed protocols deal with network coding in opportunistic routing [22,34].

4.3.5 Multi-Channel Extremely Opportunistic Routing

The Multi-Channel Extremely Opportunistic Routing (MCExOR) protocol [39] extends the ExOR protocol by adopting multi-channel forwarding requiring a single RF transceiver per device. The simultaneous use of multiple RF channels is in fact a promising approach to increase the capacity of multihop wireless networks, and MCExOR improves the network performance by choosing the RF channel with the most promising candidate set for every transmission. However, the multi-channel approach introduces new issues related to channel management and MCExOR does not deal with these issues because it assumes that the channel assignment is decoupled by the routing protocol.

While both candidate selection and the forward election processes are simple tasks in ExOR, MCExOR introduces the additional issue related to choosing the transmission channel. Such an issue involves the construction of a candidate set for each RF channel, and then the selection of the most promising candidate set.

The selection of the candidates for each channel is based on the ETX metric, which is very similar to the ExOR metric. In contrast, the selection of the most promising candidate set requires a moderate amount of sophistication.

Using $p_1(x, y)$ to define the success probability of the link between nodes x and y and with $g(x, y, z)$ the ETX metric for the path between source x and destination z using y as the next hop, the priority q of a candidate set $C_s = \{c_i\}_{i=1}^{n}$ for the forwarder w is heuristically defined as:

$$q(C_s) = \sum_{i=1}^{n} g(w, c_i, d) \frac{p_c(w, c_i)}{p_{nc}(w)} \tag{4.13}$$

where d is the destination, $p_c(w, c_i)$ is the probability that the i-th candidate is w's next forwarder, that is, the probability that the packet sent by w is not received by the $i - 1$ candidates with highest priority:

$$p_c(w, c_i) = p_l(w, c_i) \prod_{j=1}^{i-1} (1 - p_l(w, c_j)) \tag{4.14}$$

and $p_{nc}(w)$ is the probability that the packet is not received by any of the candidates:

$$p_{nc}(w, c_i) = \prod_{j=1}^{n} (1 - p_l(w, c_j)). \tag{4.15}$$

Like SOAR, the MCExOR forward responsibility transfer is based on a TDMA strategy. However, MCExOR uses the TDMA mechanism to regulate the node access for ACK packet transmission. From an operational point of view, when a candidate receives a packet, it stores the packet in a buffer and sets a timer based on its priority, i.e., its position in the forwarder list. The higher the candidate priority, the sooner the timer will expire. When a timer expires, the node sends an ACK packet to the source, and the other candidates send their ACKs according to their priorities. In such a way, all the candidates can become aware of which node is in charge of packet forwarding.

4.3.6 Opportunistic DHT-Based Routing

The Opportunistic DHT-based Routing (ODR) protocol [7] resorts to a location-aware addressing schema [4,5,11], which allows it to group nodes based on their addresses. This approach lets nodes estimate, by means of the ETX metric, their distances from sets of nodes sharing the same address prefix, instead of individually tracking each node. In this way, ODR provides a scalable strategy for loss rate distribution, which is a common issue of opportunistic routing protocols.

However, such a procedure requires the availability of a distribute procedure to allow nodes to retrieve the destination addresses before starting a communication. ODR accomplishes this task by resorting to a Distributed Hash Table (DHT) system, which exploits a globally known hash function $h(\cdot)$, defined on the IP address space and with values in the location-aware address space.

Every node is part of the DHT system, storing a subset of *pairs* <IP address, location-dependent address> in accordance with the hash function. Specifically, the pair $<ip_1, add_1>$ is stored by the node whose location-dependent address is equal to $h(ip_1)$, namely the *rendezvous node*. Thus, to find a location-dependent address, a node simply sends a pair request to the rendezvous node, as shown in Figure 4.16. After the reception of the pair reply, the node is able to establish the communication.

AU: Please mention Fig. 4.17 in text.

More in detail, Opportunistic DHT-based Routing (ODR) assigns location-dependent addresses, namely strings of l bits, to nodes by means of a distribute procedure that resorts to locally broadcasted hello packets. The address allocation procedure guarantees that nodes sharing a common address prefix are close in the physical topology, allowing the routing tables to easily group nodes.

ODR represents the address space as a *complete binary tree* of $l + 1$ levels, that is, as a binary tree in which every vertex has zero or two children and all leaves are at the same level (Figure 4.18a). In the tree structure, each leaf is associated with

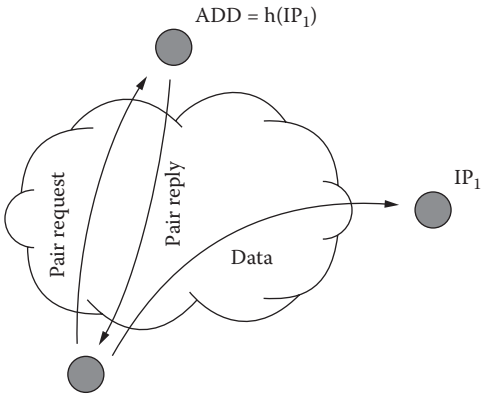

Figure 4.16 Location-dependent address discovery.

an address, and an inner vertex of level *k*, namely a *level-k subtree*, represents a set of leaves (that is, a set of peer identifiers) sharing a prefix of $l - k$ bits. For example, with reference to Figure 4.18a, the vertex with the label *01X* is a level-1 subtree and represents the leaves *010* and *011*.

Defining a *level-k sibling* of a leaf as the level-*k* subtree, which shares the same parent with the level-*k* subtree to which the leaf belongs, and referring to the previous example, the vertex with the label *1XX* is the level-2 sibling of the address *000*.

By means of the sibling concept, nodes can reduce the overhead due to maintaining a distance state by a logarithm factor. Each node stores a limited-size distance table composed of *l* entries, one for each sibling, and the *k*-th section contains the distance estimated according to the ETX metric with the *nearest* node whose location-dependent address belongs to the level -*k* sibling.

Clearly, this approach raises a new problem because the hierarchy related to the sibling concept gives rise to an estimate inaccuracy. In fact, the *k*-th section stores the estimated distance toward the nearest node whose address belongs to the level-*k* sibling; in other words, the section stores a lower bound on the distance.

The proposed solution to this issue requires that when a node forwards a packet, it stores its location-dependent address in the packet header along with its estimated

Destination	Path quality	Route log
011	1.60	001
00X	3.80	010
1XX	1.25	010

AU: Please provide citation i n text?

Figure 4.17 Node 010 routing table.

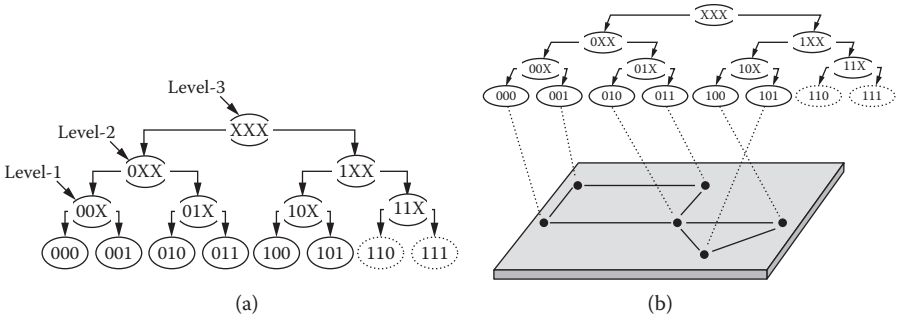

Figure 4.18 Relationship between the address space structure and the physical topology.

distance from the destination. A receiving node determines whether its overlay distance to the destination (i.e., the length of the address prefix shared by the node address and the destination address) is shorter than the forwarding overlay distance and then checks whether its path quality is better than the quality of the forwarder node. If both of these checks fail, the node does not belong to the candidate set and it stores the packet in its ACK queue. If both checks are successful, it stores the packet in its packet queue together with a delay time evaluated according to the following heuristic relation:

AU: Change correct?

$$delay = \tau * \frac{q_p(r,d)}{q_p(f,d)} * \left[\frac{1}{o_d(r,d) - o_d(f,d) + 1} \right] \tag{4.16}$$

where τ is the maximum delay time (2 seconds in our implementation), f is the forwarding node, r is the receiving node, d is the destination node, q_p is the estimated quality, and o_d is the overlay distance. By means of this heuristic approach for the delay estimation, the authors account for the estimated inaccuracy mentioned before, since the ratio between the estimated qualities ratio is weighted by a factor (i.e., the term in the square brackets in Equation [1.16] depending on the overlay distances) that measures the size of the cluster of nodes, namely the siblings, to which the qualities refer.

Thus, the delay times allow nodes to implement a distributed candidate election procedure, by exploiting a TDMA-based scheduling. Because the closest node stores its estimated distance from the destination in the packet header, and since it is the first node that forwards the packet, the other candidates can listen to the packet transmission and therefore give up responsibility for packet forwarding.

AU: Change correct?

Such a strategy does not require explicit acknowledgment for forwarding responsibility transfer, although it is not tolerant of the hidden terminal problem. In such a case, ODR resorts to explicit acknowledgment.

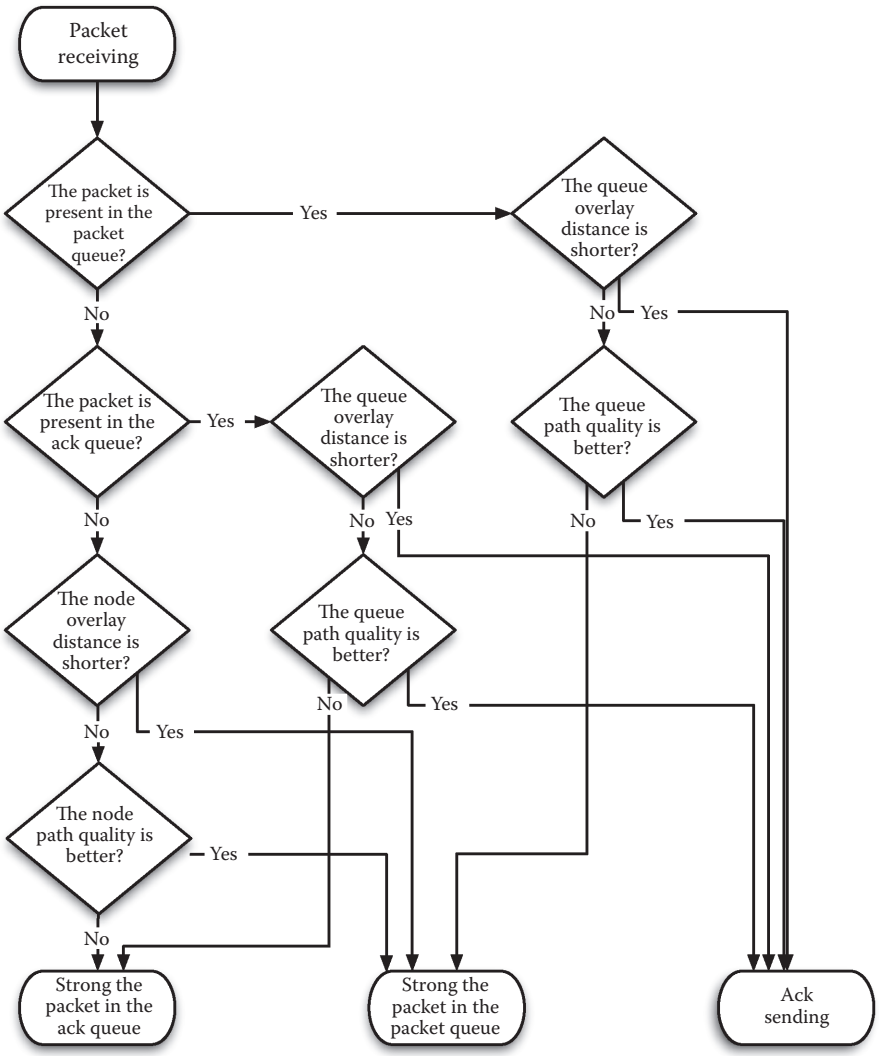

Figure 4.19 Packet forwarding process.

Figure 4.19 gives a detailed description of the whole forwarding process in a flow chart.

4.3.7 Comparison of the Considered Protocols

In this section we summarize the main characteristics of the considered protocols and offer a qualitative comparison in terms of advantages and drawbacks.

More specifically, Table 4.1 compares the strategies adopted by the different protocols for each task, which are *candidate selection*, *forwarder election*, *responsibility transfer*, and *duplicate transmission avoidance*. With respect to candidate election, all the considered protocols adopt an ETX-based strategy. However, only

Table 4.1 Basic Characteristics of the Considered Protocols

Protocol	*Candidate Selection*	*Forwarder Election*	*Responsibility Transfer*	*Duplicate Tx Avoidance*
ExOR	Fixed at the sender side according to the ETX metric	Established at the sender side according to the ETX metric	Implicit at the batch-level based on TDMA and according to the gossip mechanism implemented by the batch maps	implicit at the packet level based on the gossip mechanism, explicit at the batch level based on ack packets
SOAR	Fixed at the sender side according to the ETX metric	Established at the sender side according to the ETX metric	Explicit based on TDMA	Implicit based on the diverging route avoidance, explicit based on ACKs
MORE	Fixed at the sender side according to the ETX metric	None: multiple forwarder allowed	Implicit at the batch level, based on packet reception	Implicit based on the estimation of the number of transmissions
MIXIT	Fixed at the sender side according to the ETX metric	None: multiple forwarder allowed	Implicit at the packet level, based on packet reception	Implicit based on the estimation of the number of transmissions
MCExOR	Fixed at the sender side according to the ETX metric	Established at the sender side according to the ETX metric	Explicit based on TDMA	Explicit based on ACKs
ODR	Dynamic at the receiver side according to the ETX metric	Dynamic at the receiver side according to the ETX metric	Explicit based on TDMA	Implicit based on packet reception and explicit based on ACKs

Table 4.2 Comparison of the Considered Protocols

Protocol	*Advantage*	*Drawbacks*
ExOR	Low overhead due to explicit acknowledgment	Complex forward responsibility transfer likely to produce duplicate transmissions operates on batch of packets
SOAR	Simple forward responsibility transfer operates on single packets	Limited opportunities due to the route divergence avoidance
MORE	Multiple forwarders allowed	Complicated duplicate transmission avoidance operates on batch of packets
MIXIT	Multiple forwarders allowed	More coordination needed operates on faulty packets
MCExOR	Simple forward responsibility transfer increased throughput thanks to multi-channel	High overhead due to ACKs channel assignment
ODR	Simple forward responsibility transfer scalable mechanism for path losses distribution	Unsuitable in scenarios characterized by high mobility

ODR allows the candidate set to be dynamically selected at the receiver side, thus increasing the capability to explore the opportunities offered by wireless propagation. According to the forwarder election process, the considered protocols fall into two classes: those that allow a single forwarder and those that allow multiple forwarding by means of network coding. Clearly, the last strategy assures an increased throughput, although it operates only on batches of packets. Finally, concerning responsibility transfer and duplicate transmission avoidance, all the considered protocols adopt a TDMA-like strategy, which can or cannot take into account the candidate priority.

In Table 4.2 we synthesize the main advantages of each proposal, along with the main drawbacks.

4.4 Future Work Issues

As mentioned in Section 4.1, opportunistic routing protocols try to take advantage of the time-variant nature of the environment to provide end-to-end connectivity in scenarios where traditional networking fails.

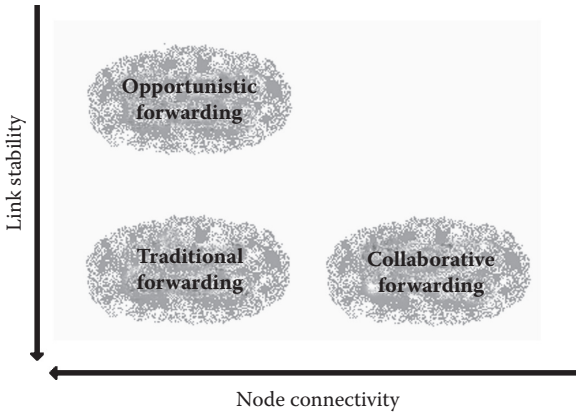

Figure 4.20 Routing protocols taxonomy.

Another class of routing protocols, the *collaborative routing* protocols, exploits a similar paradigm that assumes that the network topology is quite dense in order to assure that the presence of a persistent path between each pair of nodes is relaxed.

This class exploits the time-variant nature of the network topology to provide connectivity for sparse topologies, usually by resorting to the so-called *store-carry-forward* paradigm [17,26]. Delay-tolerant networks are the typical application domain for collaborative routing, since they aim to provide connectivity in rural and developing areas where the costs associated with traditional dense networks are not affordable.

In Figure 4.20, a taxonomy of the different classes of routing protocols is shown. The majority of routing protocols previously described relax the assumption that the wireless propagation conditions are stationary enough to allow persistent communication among neighbor nodes. The protocols belonging to the collaborative routing class relax the assumption of a dense network topology.

In the future, we expect that a new class of routing protocols will be developed that will be able to provide connectivity when both the assumptions are not verified.

AU: Please clarify.

4.5 Conclusions

As it has been shown in this chapter, there are a vast variety of routing protocols designed specifically for ad hoc mobile networks. These networks create a hostile routing environment due to the instability of wireless propagation conditions and the mobility of the nodes. However, with the introduction of the opportunistic paradigm, significant advances have been made toward the development of robust routing protocols that can assure end-to-end connectivity, even in hostile environments.

It is likely that there is not currently a single opportunistic routing protocol that can satisfy the needs of every conceivable network scenario. In fact, some protocols limit the set of candidates to bound the overhead for forward responsibility transfer, while also limiting the opportunities offered by the network. Other protocols resort to network coding, simplifying the responsibility transfer process but operating on batches of packets. Most of the proposed protocols need estimates of the path losses, but do not provide any scalable mechanism to distribute them.

AU: Please clarify

The understanding gained from these first proposals can be used, in the next few years, to improve future designs of wireless routing protocols. There still remains much to do in terms of understanding, developing, and deploying a network layer for ad hoc scenarios.

Acknowledgments

The authors would like to express their thanks to the editor, Prof. Mieso Denko, for his invitation to contribute to this book, and to Dr. Angela Sara Cacciapuoti for her insightful comments and contributions to this chapter.

References

[1] Sanjit Biswas and Robert Morris. Opportunistic routing in multi-hop wireless networks. *SIGCOMM Comput. Commun. Rev.,* 34(1):69–74, 2004.

[2] Sanjit Biswas and Robert Morris. ExOR: Opportunistic multi-hop routing for wireless networks. *SIGCOMM Comput. Commun. Rev.,* 35(4):133–144, 2005.

[3] Angela Sara Cacciapuoti, Marcello Caleffi, and Luigi Paura. Analytical evaluation of data-link transmissions in opportunistic routing. In *Submitted to ICUMT '09: the IEEE International Conference on Ultra Modern Telecommunications,* October 2009.

AU: Is this part of the title?

[4] Marcello Caleffi. Mobile ad hoc networks: The DHT paradigm. In *PerCom '09: The Seventh Annual IEEE International Conference on Pervasive Computing and Communications,* pages 1–2, March 2009.

[5] Marcello Caleffi, Giancarlo Ferraiuolo, and Luigi Paura. Augmented tree-based routing protocol for scalable ad hoc networks. In *MASS '07: the IEEE International Conference on Mobile Ad hoc and Sensor Systems,* pages 1–6, October 2007.

[6] Marcello Caleffi, Giancarlo Ferraiuolo, and Luigi Paura. A reliability-based framework for multi-path routing analysis in mobile ad-hoc networks. *International Journal of Communication Networks and Distributed Systems,* 1(4-5-6):507–523, 2008.

[7] Marcello Caleffi and Luigi Paura. Opportunistic routing for disruption tolerant networks. In *AINA '09: the IEEE 23rd International Conference on Advanced Information Networking and Applications,* May 2009.

[8] Szymon Chachulski, Michael Jennings, Sachin Katti, and Dina Katabi. Trading structure for randomness in wireless opportunistic routing. *SIGCOMM Comput. Commun. Rev.,* 37(4):169–180, 2007.

[9] T. Clausen, P. Jacquet, A. Laouiti, P. Muhlethaler, A. Qayyum, and L. Viennot. Optimized link state routing protocol. In *IEEE INMIC,* December 2001.

[10] Douglas S. J. De Couto, Daniel Aguayo, John Bicket, and Robert Morris. A high-throughput path metric for multi-hop wireless routing. *Wireless Networks,* 11(4):419–434, 2005.

[11] J. Eriksson, M. Faloutsos, and S. V. Krishnamurthy. Dart: Dynamic address routing for scalable ad hoc and mesh networks. *IEEE/ACM Transactions on Networking,* 15(1):119–132, 2007.

[12] Z. J. Haas. A new routing protocol for the reconfigurable wireless networks. *Universal Personal Communications Record, 1997. Conference Record, 1997 IEEE 6th International Conference on,* 2:562–566, October 1997.

[13] T. Ho, R. Koetter, M. Medard, D. R. Karger, and M. Effros. The benefits of coding over routing in a randomized setting. In *Information Theory, 2003. Proceedings. IEEE International Symposium on,* pages 442–, June 2003.

AU: Need ending page.

[14] Shweta Jain and Samir R. Das. Exploiting path diversity in the link layer in wireless ad hoc networks. *Ad Hoc Networks,* 6(5):805–825, 2008.

[15] Kyle Jamieson and Hari Balakrishnan. PPR: Partial packet recovery for wireless networks. *SIGCOMM Comput. Commun. Rev.,* 37(4):409–420, 2007.

[16] D. B. Johnson and D. A. Maltz. Dynamic source routing in ad hoc wireless networks. In *Mobile Computing,* Volume 353, pages 153–181. Kluwer Academic Publishers, 1996.

[17] Philo Juang, Hidekazu Oki, Yong Wang, Margaret Martonosi, Li S. Peh, and Daniel Rubenstein. Energy-efficient computing for wildlife tracking: design trade-offs and early experiences with ZebraNet. In *ASPLOS-X: Proceedings of the 10th International Conference on Architectural Support for Programming Languages and Operating Systems,* volume 37, pages 96–107, October 2002.

[18] Sachin Katti and Dina Katabi. Mixit: The network meets the wireless channel. In *HotNets-VI: Proceedings of the Sixth ACM Workshop on Hot Topics in Networks,* 2006.

[19] Sachin Katti, Dina Katabi, Hari Balakrishnan, and Muriel Medard. Symbol-level network coding for wireless mesh networks. In *ACM SIGCOMM,* August 2008.

AU: If these are proceedings, need full title.

[20] Jonghyun Kim and Stephan Bohacek. A comparison of opportunistic and deterministic forwarding in mobile multihop wireless networks. In *MobiOpp '07: Proceedings of the 1st International MobiSys Workshop on Mobile Opportunistic Networking,* pages 9–16, 2007.

[21] S. Lee and M. Gerla. Split multipath routing with maximally disjoint paths in ad hoc networks. In *ICC '01: Proceedings of the IEEE International Conference on Communications,* pages 3201–3205, 2001.

[22] Yunfeng Lin, Baochun Li, and Ben Liang. Codeor: Opportunistic routing in wireless mesh networks with segmented network coding. In *IEEE International Conference on Network Protocols, 2008. ICNP 2008,* pages 13–22, October 2008.

[23] Chun-Pong Luk, Wing-Cheong Lau, and On-Ching Yue. An analysis of opportunistic routing in wireless mesh network. In *IEEE International Conference on Communications, 2008. ICC '08,* pages 2877–2883, May 2008.

[24] Mahesh K. Marina and Samir R. Das. Ad hoc on-demand multipath distance vector routing. *SIGMOBILE Mob. Comput. Commun. Rev.,* 6(3):92–93, 2002.

[25] Allen Miu, Hari Balakrishnan, and Can Emre Koksal. Improving loss resilience with multi-radio diversity in wireless networks. In *MobiCom '05: Proceedings of the 11th Annual International Conference on Mobile Computing and Networking,* pages 16–30, 2005.

[26] Alex (Sandy) Pentland, Richard Fletcher, and Amir Hasson. Daknet: Rethinking connectivity in developing nations. *Computer,* 37(1):78–83, 2004.

[27] C. Perkins and E. Royer. Ad hoc on-demand distance vector routing. In *2nd IEEE Workshop on Mobile Computing Systems and Applications,* pages 90–100, 1999.

[28] Charles Perkins and Pravin Bhagwat. Highly dynamic destination-sequenced distance-vector routing (DSDV) for mobile computers. In *SIGCOMM '94: ACM Conference on Communications Architectures, Protocols and Applications,* pages 234–244, 1994.

[29] E. Rozner, J. Seshadri, Y. Mehta, and Lili Qiu. Simple opportunistic routing protocol for wireless mesh networks. In *2nd IEEE Workshop on Wireless Mesh Networks, 2006. WiMesh 2006,* pages 48–54, Sept. 2006.

[30] R.C. Shah, S. Wietholter, A. Wolisz, and J. M. Rabaey. When does opportunistic routing make sense? In *Third IEEE International Conference on Pervasive Computing and Communications Workshops, 2005. PerCom 2005 Workshops,* pages 350–356, March 2005.

[31] A. Tsirigos and Z. J. Haas. Analysis of multipath routing, part 2: Mitigation of the effects of frequently changing network topologies. *Wireless Communications, IEEE Transactions on,* 3(2):500–511, March 2004.

[32] A. Tsirigos and Z. J. Haas. Analysis of multipath routing, part 1: The effect on the packet delivery ratio. *Wireless Communications, IEEE Transactions on,* 3(1):138–146, January 2004.

[33] Grace R. Woo, Pouya Kheradpour, Dawei Shen, and Dina Katabi. Beyond the bits: Cooperative packet recovery using physical layer information. In *MobiCom '07: Proceedings of the 13th Annual ACM International Conference on Mobile Computing and Networking,* pages 147–158, 2007.

[34] Yan Yan, Baoxian Zhang, H. T. Mouftah, and Jian Ma. Practical coding-aware mechanism for opportunistic routing in wireless mesh networks. In *IEEE International Conference on Communications, 2008. ICC '08,* pages 2871–2876, May 2008.

[35] Kai Zeng, Wenjing Lou, and Hongqiang Zhai. Capacity of opportunistic routing in multi-rate and multi-hop wireless networks. *IEEE Transactions on Wireless Communications,* 7(12):5118–5128, December 2008.

[36] Jian Zhang, Y. P. Chen, and I. Marsic. Network coding via opportunistic forwarding in wireless mesh networks. In *IEEE Wireless Communications and Networking Conference, 2008. WCNC 2008,* pages 1775–1780, April 2008.

[37] Rong Zheng and Chengzhi Li. How good is opportunistic routing?: A reality check under Rayleigh fading channels. In *MSWiM '08: Proceedings of the 11th International Symposium on Modeling, Analysis and Simulation of Wireless and Mobile Systems,* pages 260–267, 2008.

[38] A. Zubow, M. Kurth, and J.-P. Redlich. Considerations on forwarder selection for opportunistic protocols in wireless networks. In *14th European Wireless Conference, 2008. EW 2008.* pages 1–7, June 2008.

[39] Anatolij Zubow, Mathias Kurth, and Jens-Peter Redlich. Multi-channel opportunistic routing. In *European Wireless,* 2007.

AU: Need full title.