Towards Reliable Forwarding for Ad Hoc

Marco Conti, Enrico Gregori, and Gaia Maselli

IIT Institute - CNR,
Via G. Moruzzi, 1 - 56124 Pisa, Italy
{conti.marco,gregori.enrico,maselli.gaia}@iit.cnr.it

Abstract. Ad hoc networking is a new paradigm of wireless communications for mobile nodes. Mobile ad hoc networks work properly only if the partecipating nodes cooperate to network protocols. Cooperative algorithms make the system vulnerable to user misbehavior as well as to malicious and selfish misbehavior. Nodes act selfishly to save battery power, by not cooperating to routing-forwarding functions. Lack of cooperation may severely degrade the performance of the ad hoc system. This paper presents a new approach to cope with cooperation misbehavior, focusing on the forwarding function. We present a general framework, based on reliability indices taking into account not only selfish/malicious misbehavior, but also situations of congestion and jammed links. We aim at avoiding unreliable routes and enforcing cooperation, thus increasing network "performability" (performance and reliability).

1 Introduction

A mobile ad hoc network is composed by a group of wireless nodes that cooperatively form the network without the support of any fixed infrastructure. All nodes are capable of movement and can be connected dynamically in an arbitrary manner. Nodes of these networks function as routers which discover and maintain routes to other nodes in the network. Essentially, the mobile ad hoc networking technology enables an autonomous system of mobile nodes, and introduces the notion of a *spontaneous network* [1], created when a group of people come together for some collaborative activity. Such networks have been proposed for several goals: data collection in sensor arrays, providing a communication mean in hostile environment (battlefield) or rescue operations, providing connectivity to people attending conferences, meetings or lectures with their laptops. Another emerging application of ad hoc networks concerns vehicular networks, where intervehicle communications and vehicle to road communications are considered to have extensive potential for the development of efficient safety systems installed in vehicles [2].

⋆ This work was partially funded by the Information Society Technologies programme of the European Commission, Future and Emerging Technologies under the IST-2001-38113 MOBILEMAN project

The lack of centralized points leads to the necessity of distributing basic functions like packet routing and forwarding to all available nodes in the network that must cooperate, and provide services to each other. The lack of a fixed infrastructure, and consequently of a centralized authority, leads adversaries to exploit this vulnerability for new types of attacks designed to break the cooperative paradigm. For example, routing mechanism is vulnerable in ad hoc networks because each device acts as a router. Forwarding mechanism is cooperative as well: communications between nodes more than 1-hop away are performed by intermediate nodes that act as relays.

A node that does not cooperate is called a misbehaving node, and routing-forwarding misbehavior can be caused by nodes that are broken, overloaded, malicious or selfish [3]. A broken node is not able to cooperate because of a software/hardware fault. An oveloaded node does not cooperate because it lacks CPU cycles, buffer space, or network bandwidth. A malicious node does not cooperate because it wants to intentionally damage network functioning by dropping packets. A selfish node is unwilling to spend battery life, CPU cycles, or available network bandwidth to forward packets not of direct interest to it, even though it expects others to forward packets on its behalf. It uses the network but does not cooperate, saving battery life for its own communications: it does not intend to directly damage other nodes. While the first two cases (broken and overloaded) define misbehavior due to uncontrollable events, in case of a malicious or selfish node, cooperation misbehavior is motivated by an intentional action of the node, even if with different aim: malicious or not. In any case, a cooperation misbehavior can have severe effects on the network functioning. Cooperation misbehavior due to an intentional action is a new problem that arises in the context of ad hoc networks, so new mechanisms are needed to face the problem of service availability.

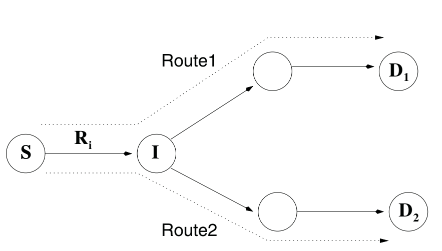
Cooperation among nodes has been previously addressed in [4], [5], [6]. The starting step is given in [4]. They present a solution aimed at detecting and avoiding misbehaving nodes through a mechanism based on a *watchdog* and a *reputation system*. The watchdog identifies misbehaving nodes by performing a neighborhood monitoring: it observes the behavior of neighbors by promiscuously listening to communications of nodes in the same transmission range. According to collected information, the reputation system maintains a value for each observed node that represents a reputation of its behavior. The reputation mechanism allows to avoid sending packets through misbehaving nodes. In this way, malicious nodes are rewarded and strengthened, while cooperation enforcing is totally absent. The following works, CONFIDANT [5] and CORE [6], extend such a scheme with a punishment mechanism that isolates misbehaving nodes by not serving their requests. When a neighbor's reputation falls down a predefined threshold, service provision to the misbehaving node is interrupted. In such a way, there is no advantage for a node to misbehave because any resource utilization will be forbidden. These solutions present some limitations. First, the watchdog's weaknesses are not negligible: in presence of collisions, dishomogeneous transmission ranges, or directional antennas, the watchdog is not able to properly monitoring the neighborhood, and misbehaving nodes detection can fail. Another important aspect that must be considered is the employing of cooperation in security mechanisms. This approach may have severe drawbacks in term of traffic overhead and wrong accusation spreading. The CONFIDANT protocol generates some additional traffic for reputation propagation. The produced overhead may result heavy, and malicious nodes may perform a new attack by sending false alarms about other nodes. Furthermore, it is worth noticing that both CONFIDANT and CORE do not take into account the network utilization: by totally avoiding all routes containing misbehaving nodes, they risk deviating all the traffic on well behaving nodes, with the result of overloading them and links between them. Finally, both mechanisms work as extensions to the Dynamic Source Routing (DSR) [7] protocol. This can be a big constraint as there is not yet a standard routing protocol for ad hoc networks [8].

2 Estimating Routes Reliability

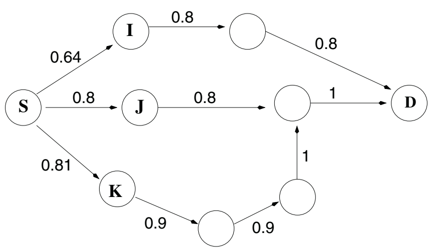
We address forwarding misbehavior due to intentional actions, malicious or selfish, as well as uncontrollable events, such as congestion or jammed links. The basic idea is to control nodes' in/out traffic to optimize network utilization, and enforce cooperation. The system is based on the principle that we can trust only ourselves and we cannot solve the cooperation misbehavior problem by using cooperation. So, the proposed mechanism is distributed, but not cooperative, and based on nodes internal knowledge. Every node acts independently, without sharing any information with other nodes, and trusts only information coming from the other communication peer (communication between peers can be encrypted to avoid forged acknowledgements by intermediate malicious nodes). According to our approach a node is responsible not only for forwarding a packet, but it shall forward it on the route that maximizes its success probability.

The framework we define is based on reliability indices. Every node has a dynamically updated reliability table containing a value for every outgoing link to a neighbor. Such a value represents a reliability index for paths rooted at that neighbor. Every time the node sends a packet on a path, it updates the reliability value associated to the neighbor through which the packet has passed: the updating is positive whenever source node receives an acknowledgement from destination, negative otherwise. The reliability value is unique for all paths rooted on that neighbor (see Fig. 1(a)). If source node observes that the reliability index of that subtree decreases, then it should immediately reduce the traffic sent through that neighbor, by preferring routes passing through a neighbor with a higher reliability index. Figure 1(b) shows an example: source node S has three possible routes to send a packet to destination node D. Each route passes through a different neighbor (I, J, K), and each link to a neighbor has a reliability index[1]. By comparing such values, source S finds out that path through K is the better

[1] Node S does not have the knowledge of the reliability of all the links, but it has a reliability index for each subtree rooted on its neighbors. This index summarizes the reliability of all the links crossed by the S-D path.



(a) Every time node S sends a packet on Route 1 or Route 2, it updates index R_i, that is associated to neighbor I. Thus R_i indicates the reliability level of the network subtree rooted at I.

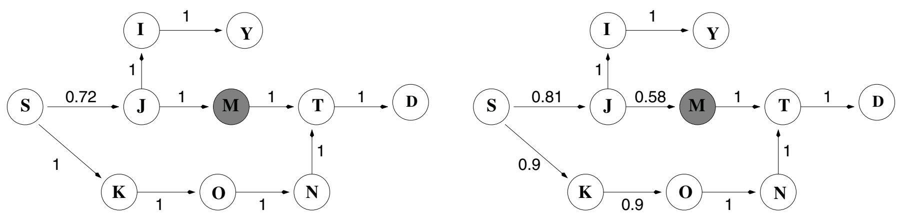


(b) The graph shows for each node, the reliability indices associated to its neighbors. Node S has three possible routes to reach destination D and can choose one of them according to the reliability index associated to its neighbors (route through K seems the most reliable)

Fig. 1. Reliability index

one, even if the longer in terms of hops number. Source S may decide to take that path to maxime the success probability of the packet forwarding. This procedure of choosing the best next hop can be executed by each intermediate node between source and destination, if the routing protocol adopted does not provide that the packet shall follow a predefined route (i.e., DSR protocol specifies the whole route a packet has to follow to reach destination, so intermediate nodes cannot choose the next hop to which forward packets). The only difference between source node and intermediate nodes is the possibility to update reliability values. While source node expects an acknowledgement from destination, intermediate nodes do not have any way to understand if the packet has reached destination. Thus, intermediate nodes can use reliability values (calculated when sending their own traffic) to take decisions about available routes, but they cannot modify them, because in such a case they do not get any acknowledgement back.

The reliability index associated to each neighbor reflects the behavior of all paths passing through it. In this way, we do not point to single misbehaving nodes, but to zones (potentially) containing one or more of them. In reality, when communication between nodes increases (most of nodes communicate with each other), there is a strong relation between a reliability value and the node pointed out by it: closer is the misbehaving node, lower is the reliability value in its direction. Figure 2 shows an example. M is a selfish node dropping packets with probability p. At first all reliability values are initialized to 1. Node S starts sending packets to Node D through K. On such a route there is a selfish node (M) that drops some packets. Thus, every time Node M drops a packet, Node S updates J's reliability that decreases, for example, to value 0.72 (Fig. 2(a)). Than Node S starts sending all packets on the other available route (through neighbor K) that keeps a good reliability value. When communication between nodes increases (e.g., Node S starts communicating with Y, J with T, and S with



(a) M is a selfish node that drops packets with a probability p. When S starts sending packets to D through J, reliability associated to J decreases because M's behavior affects it. Thus S starts deviating its traffic through J.

(b) At full capacity, J's reliability increases because of communication from S to Y, while reliability on link (J,M) continues decreasing every time M drops a packet sent by J to T or D.

Fig. 2. Relation between node's behavior and reliability calculated by its neighbors.

D), J's reliability calculated by S increases, thanks to successful communication between S and Y, while M's reliability calculated by J decreases deeply, because of the selfish node M (Fig. 2(b)). This example highlights how reliabilities of all nodes sending packets on a path with a selfish node are affected by that misbehavior. Closer is the misbehaving node, lower is the reliability calculated in that direction.

To enforce cooperation we propose an approach analogous to the outgoing traffic control. The basic idea is to accept an incoming packet according to the reliability associated with the incoming link: if the value is low, then the receiving node may neglet the packet. This reciprocal approach to in/out traffic allows to use links in a way that agrees with their functioning level. A link that has a low reliability value is used to send messages with a frequency proportional to its reliability value, as well as messages received from that link are accepted according to that value. The correspondence between the reliability index and the accepting level of incoming traffic must still be defined.

The formula used to choose a route, in case of multiple available choices, takes into account the network utilization problem: besides looking at the probability value, the formula considers also traffic balance.

Every node builds its own reliability table by only referring to its internal actions, without any cooperation with other nodes. In this manner, there is no way for a malicious node to deceive other nodes because there is no information exchange. This absence of cooperation in the security mechanism has two important advantages. First, the system is resistant to attacks performed using the security mechanism itself: it is impossible for a node to maliciously decrease another node's probability. Second, there is an advantage in terms of traffic overhead, because no additional traffic is generated.

Another advantage of the proposed scheme is the overcoming of watchdog's weaknesses: probabilities are calculated by means of local traffic observations instead of neighborhood monitoring. Furthermore, the mechanism is flexible because it is independent on the routing protocol adopted.

3 The Proposed Scheme

3.1 Model and Assumptions

We model a network as a graph G = (V, L), where V is a set of mobile nodes and L is a set of direct links. Each node i ∈ V has a unique node identifier (ID). A link (i, j) ∈ L represents a connection between the two nodes i and j, meaning that j is in the transmitting range of i, and viceversa. In that case, nodes i, j are said adjacent (or neighbors), and we call N(i), neighbor set of i, the set of nodes adjacent to a given node i:

$$N(i) = \{j | j \in V \wedge (i, j) \in L\}$$

In this model we assume the following:

1. links allow two-way communication (bidirectional links), so that connected nodes can communicate with each other in either direction;
2. an end-to-end acknoledgement notifies packets delivery between peer nodes[2];
3. we know multiple routes to a destination, and for each route the source node knows the next hop to reach destination.

Regarding to hypothesis 3, it can be easily satisfied in a sensor network where each node has view of the network topology that is static. In a mobile ad hoc network, a slight modification of one of the existing routing protocols may allow the executing node to store the different routes it identified.

Given any node i ∈ V, for each j ∈ N(i) we have a probability value R_j that represents the reliability level of link (i, j). Probability R_j is dinamically updated every time node i sends a packet on link (i, j) and represents a reliability measure for paths rooted at neighbor j: it increases if the sent packet reaches the destination, it decreases otherwise. We suppose to have an end-to-end notification acknowledgement on packet delivery: if node s is the source node, node d the destination, with an arbitrary number n of hops between s and d ($n > 0$), when destination node d receives the packet, it sends back to s an ack message. If s does not receive any acknowledgement before a specified timeout, then we assume the packet did not reach destination, and some node on the path did not forward it. For each packet sent, we denote with M the result of the packet delivery and we estimate a smoothed reliability value R_j using a low-pass filter, with the same approach used in the TCP protocol for Round-Trip Time measurement [9]:

$$R_j \leftarrow \alpha R_j + (1 - \alpha) M \tag{1}$$

[2] Communication between source and destination nodes can be encrypted to avoid forged acknowledgements

where α, $0 \leq \alpha \leq 1$, is a smoothing factor and represents the percentage of the previous estimate considered on each new estimate. If $\alpha = 0.9$, then ninety percent of the new estimate is from the previous estimate, and 10% is from the new measurement.

M is the result of a packet delivery process from s to d, and it can assume the following values:

$$M = \begin{cases} 0 \text{ if } s \text{ does not receive ack from } d \\ 1 \text{ if } s \text{ receives ack from } d \end{cases}$$

If packet does not reach destination, then the reliabilty on outgoing link of source node decreases by a α factor. If packet reaches destination then nodes are cooperating and reliability on outgoing link of source node is smoothed by a α factor and increased by $(1 - \alpha)$.

In the following, we show how reliability indices are used to control outgoing and incoming traffic, achieving reliable forwarding and enforcing cooperation.

3.2 Outgoing Traffic Control

In case of multiple routes available for packet forwarding to a destination node, source node can choose one of them according to a certain principle. Routing protocols for ad hoc networks usually choose the shorter one, in terms of hops number, or the fresher one in terms of discovering time. Such criteria do not take into account links reliability. Hereafter, we propose two route selection policies dealing with reliability values associated to outgoing links, and we investigate their effectiveness.

Policy-1. Source node takes always the most reliable route. In such a case, source node compares reliability values for available routes and forwards packets on the link with the greatest value. This policy assures source node of taking always the most reliable route. The main drawback of such a choice is the deviation of all traffic on most reliable links which, in case of high traffic load, can quickly get congested.

Policy-2. This policy relates reliability values of available routes to build a probabilistic scheme. Let us suppose we have several possible routes to a destination through different source's neighbor nodes, i_1, i_2,..., i_n. Each neighbor has its respective reliability value R_{i_1}, R_{i_2},...,R_{i_n}. We associate a probabilistic value to each of such neighbors, p_{i_j}, $1 \leq j \leq n$, defined in the following way:

$$p_{i_j} = \frac{R_{i_j}}{\sum_{k=1}^{n} R_{i_k}} \qquad (2)$$

where $\sum_{j=1}^{n} p_{i_j} = 1$. Equation (2) relates reliability values so that the resulting probabilistic value reflects the link reliability level. Routes are chosen according to the probabilistic value associated to the first node on the path: the greater the probability, the higher the route selection frequency. This probabilistic policy allows nodes to take even less reliable routes: traffic forwarding function is better distributed on all available routes and links congestion becomes rarer.

3.3 Incoming Traffic Control

Reliability indices can be used also to enforce cooperation. The basic idea is to reverse the service we obtain from each neighbor. For example, an incoming packet is accepted according to the reliability index associated to the incoming link. Thus we can define the probability to accept a packet from a neighbor I as:

$$P\{accept\ from\ I\} = R_i \tag{3}$$

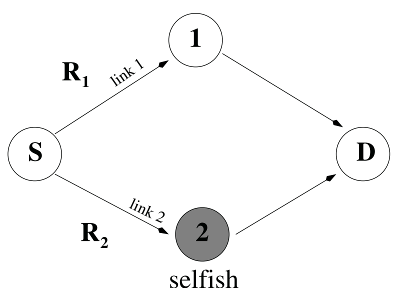
If a node accepts an incoming packet according to such a probability, then the sender will see its packets accepted in a way that agrees with its reliability level. In fact, as we said in section 2, there is a strong relation between a reliability value and the behavior of the node pointed out by such a value. If a node has a very low reliability value for a neighbor, it is quite likely that the pointed node is misbehaving. This mechanism stimulates nodes cooperation: a good reliability value will allow them to see their traffic forwarded through the network.

The example above represents a first possible approach coping with cooperation enforcing. Further work will investigate an efficient utilization of reliability indices. In particular, we aim at finding out refined policies able to control incoming traffic, without penalizing too much sender nodes.

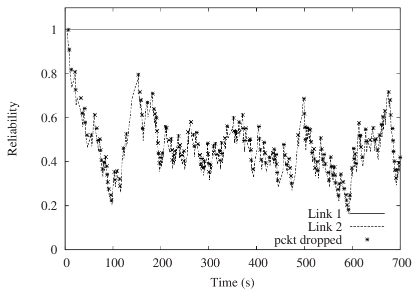
So far we considered only forwarding misbehavior as it is the topic we address in this paper. Let us see what happens in case of a routing misbehavior, caused by a malicious or a selfish node. A typical example for the former case is the Black Hole attack [10]. A node uses the routing protocol to advertise itself as having the shorter path to the node whose packets it wants to intercept. Once it has created the forged route, it can drop packets passing by it. As the malicious node drops packets, destination node will never receive such packets. Consequently, source node will not receive any acknoledgement from destination, and will decrease reliability in that direction. In the future, the outgoing traffic will avoid it and packets incoming from that zone will be refused with a high probability. On the other hand, in case of a selfish disruption of the routing protocol, the node does not partecipate to the routing function to save energy. In this situation, the selfish node will not appear in any route. This will not affect our system, that will continue working correctly, but it will not be able to detect the routing misbehavior. We intend to approach routing misbehavior in the future work.

4 Evaluation

The objective of this evaluation is to test the effectiveness of the reliability formula, given by (1), and of the policies for route selection, defined in Section 3.2. Specifically, we want to check if the defined reliability correctly reflects links behavior. In addition we want to investigate the throughput achieved by the different policies.



(a)



(b)

Fig. 3. Test of reliability formula

4.1 Simulation Setup

To test the reliability formula we simulated a double path between two source and destination nodes (see Figure 3(a)). On the first path, Node 1 is cooperative while, on the second path, there is a selfish node discarding packets with probability $p = 0.5$. Every time the node sends a packet on one of two paths, it updates the reliability value associated to the neighbor through which the packet was forwarded[3]. Figure 3(b) shows the reliability function for both neighbors. Both reliability values, R_1 and R_2 are inizialized to 1. On Link 1 reliability keeps constant the value 1 because node on that path never discards packets. On Link 2 reliability decreases every time a packet is dropped (a cross on the plotting indicates a packet dropping on Link 2), assuming a mean value of 0.5.

To evaluate network performance for the forwarding policies we considered a similar context (see Figure 4). Source node S and destination node D are more than one hop away (we denote intermediate nodes between source and destination as *zone*). Let us call Node 1 and Node 2, the neighbors of source node respectively on Route 1 and Route 2. We associate a reliability value R_1 to link towards Node 1 and, R_2 to link towards Node 2. We also suppose that all nodes in Zone 1 are cooperative, while in Zone 2 there is a selfish node discarding packets with a probability p.

We investigated throughput for different route-selection policies, by varying the traffic loads and the transmission speeds. We defined four different scenarios and conducted experiments applying the different policies to such scenarios.

Scenario 1. All links have the same transmission speed (1 Mbit). Nodes on *zone 1* are all cooperative, so that there is no packets dropping on Route 1, while there is a selfish node in *zone 2*, that provokes packets' losses with probability $p = 0.5$.

[3] For the sake of semplicity, simulations implement immediate acknowledgments of delivered packets and no loss of ACKs. This choise does not affect the meaning of obtained results and effects of ACK delay on reliability estimates will be studied in further work.

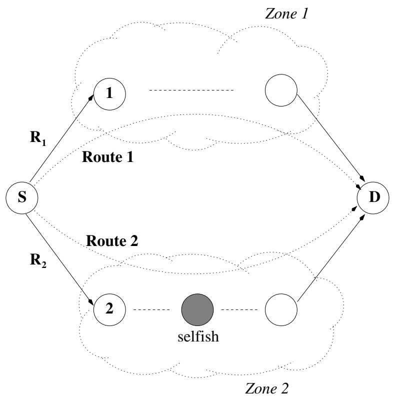


Fig. 4. Network simulated

Scenario 2. We inserted in *zone 1* a node with an outgoing link with transmission speed of 0.75 Mbit. Node N has a buffer capacity of 10 packets. Consequently, N can get congested, and drops packets arriving while the buffer is full.

Scenario 3. We considered the same parameters of the second scenario, by increasing buffer capacity to 100 for Node N.

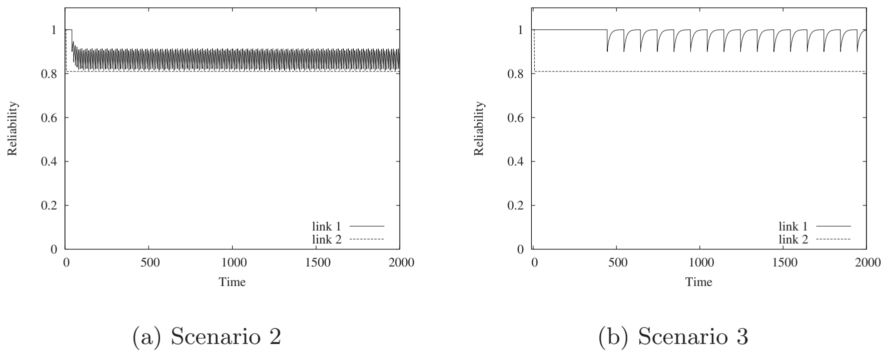
Scenario 4. We increase the transmission speed of sender link to 2 Mbit, keeping the other parameters the same as Scenario 3.

4.2 Simulation Results

Experiments have been performed in each scenario previously defined, by applying the different route selection policies. We simulated also the conventional case in which the forwarding node does not consider reliability values, and distribute packets equally on both routes. Choosing both routes with the same probability represents the best compromise when no selection criterion is used, because it minimizes congestion events. In such a case the forwarding node would probably choose always the same route (e.g., the shorter one), by overloading nodes on that route. In the following, we call such a criterion of traffic distribution *load-balancing*, to indicate that reliability values are not considered, but traffic is equally spread on both routes.

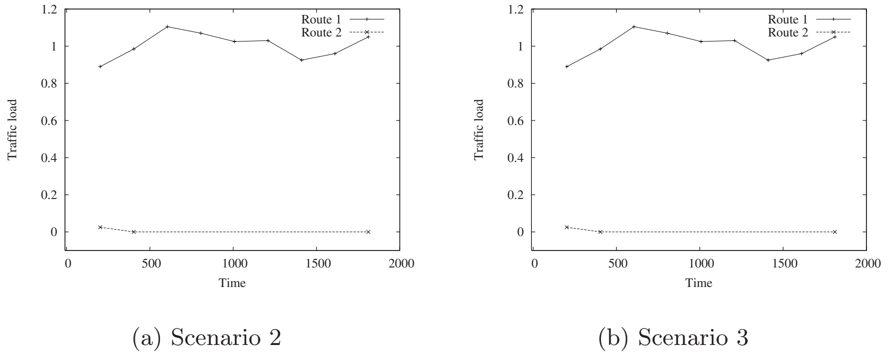
We evaluated the model with the same traffic load. We observed the reliability function and network throughput in time[4]. In particular, we studied situations of congestion. In experiments concerning Scenario 2 and 3, Zone 1 contains a node that may get congested in case of high traffic load, as it has limited buffer capacity and link speed passes from 1 Mbit (incoming link) to 0.75 Mbit (outgoing link). Figures 5 and 7 show the reliability function obtained by applying policy-1 and policy-2 to scenario 2 (Figures 5(a) and 7(a)) and to scenario 3 (Figures 5(b) and 7(b)).

[4] Time is always measured in seconds.



(a) Scenario 2 (b) Scenario 3

Fig. 5. Reliability function for link 1 and link 2, by applying policy-1

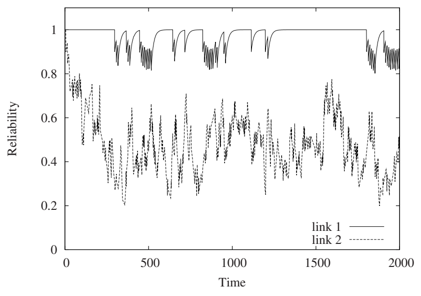


(a) Scenario 2 (b) Scenario 3

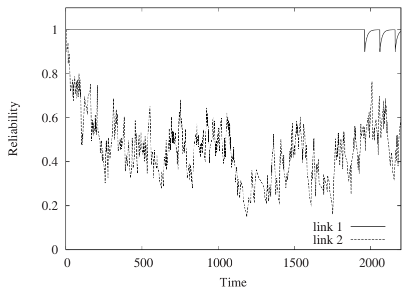
Fig. 6. Traffic sent on both routes by policy-1

When we apply policy-1, we choose always the most reliable route. Figure 5(a) shows the reliability function for Scenario 2. As reliability R_2 on Link 2 descreases because of the selfish node in Zone 2, source node S starts forwarding packets always on Route 1 (Fig. 6(a)), as soon as R_2 slopes down the value 0.81. In fact, R_1 is always greater than such a value. On Link 1 reliability decreases whenever Node N get congested. Such events are very frequent because Node N cannot buffer more that 10 packets. By increasing the buffer capacity (figure 5(b)), congestion events become rarer. This phenomenon is due to burstiness in the traffic. A longer buffer space allows to absorb temporary overload conditions. Alike the previous case, all traffic is sent on Route 1 as soon as R_2 slopes down the value 0.81 (Fig. 6(b)).

While applying policy-2, packets are distributed on both links, even if most of traffic is sent on Route 1 (Fig. 8(a)). In fact, when reliability value for Link 2 decreases greatly, probability to choose Route 1 increases, leading to some packets dropping due to congestion. Figure 7(a) shows that when the reliability value on Link 2 drops, the consequent deviation of most of traffic on Route 1 causes several events of congestion. Again, increasing buffer capacity for Node

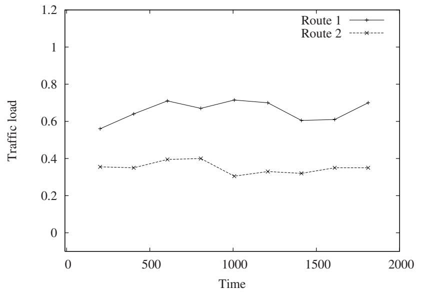


(a) Scenario 2

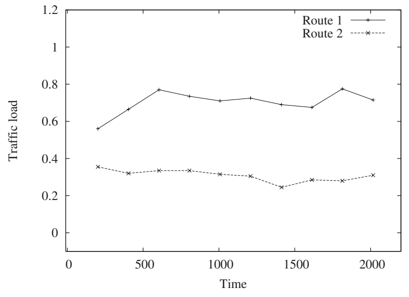


(b) Scenario 3

Fig. 7. Reliability function for link 1 and link 2, by applying policy-2



(a) Scenario 2



(b) Scenario 3

Fig. 8. Traffic sent on both routes by policy-2

Table 1. Throughput comparison

	Scenario 1	Scenario 2	Scenario 3	Scenario 4
Load-balancing	0.77	0.76	0.77	1.16
Policy-1	1.00	0.66	0.66	0.66
Policy-2	0.86	0.82	0.79	0.98

N, congestion becomes rarer (Figure 7(b)). Traffic sent on both routes is shown in Figure 8(b).

We want now to investigate the efficiency of policies previously defined by comparing their throughputs. Table 1 compares policies based on reliability (policy-1 and -2) with the load-balancing case, and shows the throughputs observed in each scenario. It is quite obvious that in Scenario 1 policies based on reliability values prevail over load-balancing. In fact, as Zone 1 never fails in forwarding packets, Route 1 is always chosen in case of policy-1, and very often chosen when policy-2 is applied.

Table 2. Throughput comparison with selfish nodes both in Zone 1 and 2

	Scenario 1
Load-balancing	0.67
Policy-1	0.81
Policy-2	0.70

If we insert a bottleneck in Zone 1, due to a slower link and/or to a node with a smaller receiving capacity (or processing power), then in case of high traffic load, Zone 1 can get congested, and drops packets. In such cases (Scenario 2 and 3), policy-1 is not very successful as it deviates all the traffic on Route 1, by causing frequent congestion events. Policy-2 appears better as it allows a more balanced traffic distribution on both routes, selecting more frequently the most reliable one.

Both reliability-based policies increase network throughput: policy-1 is better in case of no congestion, there policy-2 is more successful. This is not true if Zone 1 becomes completely congested (see Scenario 4). In such a situation, the load-balancing solution seems the better one, even if policy-2 is still very efficient. The reason is the deviation of most of traffic on Route 1 that continuously get congested and thus drops packets, while the load-balancing solution avoids any congestion events by equally distributing traffic on both routes.

Finally, we observed the throughput in case of selfishness in both zones. We considered a model analogous to that one shown in Figure 4, where both zones have a selfish node, with different level of selfishness: probability to discard packets is 0.2 in Zone 1 and 0.5 in Zone 2. Even in this situation reliability-based policies improve network performance in packet forwarding. Values reported in Table 2 show the efficiency of such policies, highlighting again the success of policy-1 in case of no congestion.

5 Policy Refinement

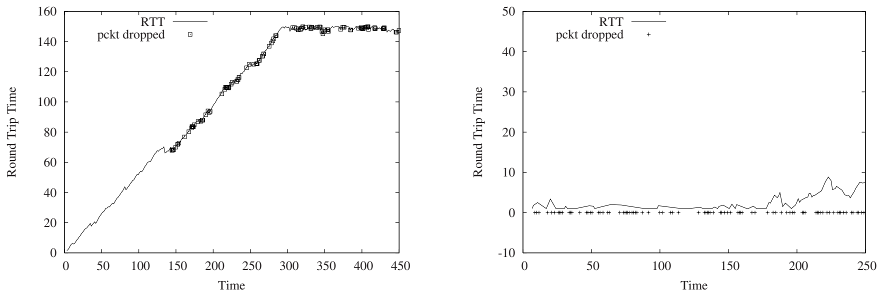
Results obtained by applying our policies to scenario 4 led us to a new possible route-selection policy, based on congestion control. In fact, in case of high traffic load and frequent congestion, reliability-based policies become ineffective. Policy-1 especially aims at choosing always the most reliable route, without taking into account the amount of traffic sent on it. To improve network performance, the forwarding node should stop sending packets on a route that is becoming overloaded, even if it has the bigger reliability value. Thus a good policy would be:

Choose the most reliable link until it does not get congested.

We repeated Scenario 4 experiments, inserting such a restriction on packet forwarding. For the sake of semplicity, we simulated the ability to understand if a node has reached the maximum of its receiving capacity by checking the occupancy of its buffer before sending a packet to it. Results obtained through

Table 3. Throughput comparison for congestion-control based policies.

	Scenario 4
Load-balancing	1.14
Policy-1	1.19
Policy-2	1.16



(a) Route 1 (b) Route 2

Fig. 9. Round Trip Time

such a simulation are shown in Table 3. Both policy-1 and policy-2 are better than load-balancing and improve network throughput in comparison with results obtained without taking congestion into account.

In reality, the simulated method is not applicable because a node cannot know other nodes' capacity. Instead, it is possible to understand if a zone on a path is becoming congested by observing the Round Trip Time (RTT). A RTT increase indicates a possible congestion condition. In fact, if we relate forwarding failures and RTT values measured in the experiment performed for policy-2 in Scenario 4, we note that, on Route 1, nodes start dropping packets when RTT reaches value 68 (see fig. 9(a)). After that value we have frequent packets dropping due to the high level of congestion. On the other route, there is no relation between RTT and packets dropping, as it is caused by a selfish node. Figure 9(b) shows how packets dropping on Route 2 happens even if RTT value keeps very low. We intend to face in a further study the identification of congestion conditions by observing the RTT value, its derivative and variation.

This led us to observe that the reliability index provides a good link characterization for selfish nodes that discard packets intentionally. On the other hand, to better cope with congestion situations, reliability can be characterized by a two states index R, to distinguish between congestion events and intentional misbehavior. We will address this new kind of reliability index in the future work.

References

1. L.M. Feeney, B. Ahlgren and A. Westerlund, Spontaneous Networking: An Application-oriented Approach to Ad Hoc Networking. In *IEEE Communications Magazine*, June 2001.
2. I. Chisalita and N. Shahmehri, A novel architecture for supporting vehicular communication. In *IEEE 56th Vehicular Technology Conference* pp. 1002–1006, September 2002.
3. P. Michiardi and R. Molva, Simulation-based Analysis of Security Exposures in Mobile Ad Hoc Networks. In *Proceedings of European Wireless 2002 Conference*, February 2002.
4. S. Marti, T. Giuli, K. Lai and M. Baker, Mitigating Routing Misbehavior in Mobile Ad Hoc Networks, In *Proceedings of the Sixth annual ACM/IEEE International Conference on Mobile Computing and Networking*, pages 255–265, 2000.
5. S. Buchegger and J. Y. Le Boudec, Performance analysis of the CONFIDANT protocol. In *Proceedings of the* 3^{rd} *ACM International Sysmposium on Mobile Ad Hoc Networking and Computing (MobiHoc)*, pages 226–236, June 2002.
6. P. Michiardi and R. Molva, CORE: A COllaborative REputation mechanism to enforce node cooperation in Mobile Ad Hoc Networks. In *Proceedings of Communication and Multimedia Security 2002 Conference*, September 2002.
7. D. B. Johnson and D. A. Maltz, The Dynamic Source Routing for mobile Ad Hoc Networks. Internet Draft, Mobile Ad Hoc Network (MANET) Working Group, IETF, October 1999.
8. E.M. Royer and C-K Toh, A Review of Current Routing Protocols for Ad-Hoc Mobile Wireless Networks. *IEEE Personal Communications*, Apr. 1999.
9. V. Jacobson, Congestion Avoidance and Control. In *Proceedings of SIGCOMM '88* Stanford, CA, Aug. 1988, ACM.
10. Y.-C. Hu, A. Perrig and D.B. Johnson, Ariadne: A secure on-demand routing protocol for ad hoc networks. In *The 8th ACM International Conference on Mobile Computing and Networking*, September 2002.