Computer Communications 35 (2012) 1707–1714

Contents lists available at SciVerse ScienceDirect

Computer Communications

journal homepage: www.elsevier.com/locate/comcom

Can cooperation improve energy efficiency in ad hoc wireless networks?

Maurizio D'Arienzo [a,*], Francesco Oliviero [b], Simon Pietro Romano [b]

[a] Seconda Università di Napoli, Italy
[b] Università di Napoli Federico II, Italy

ARTICLE INFO

Article history:
Available online 16 May 2012

Keywords:
Cooperation
Ad hoc routing
Energy efficiency
Game theory

ABSTRACT

To make an ad hoc network work properly, wireless nodes are usually requested to cooperate in routing operations. However, there is currently a lack of behavior-tracking mechanisms, so certain nodes can freely play a selfish role at the detriment of altruistic ones. In this paper we try to answer the question in the title, by showing how cooperation can definitely help reduce the overall energy consumed in an ad hoc network. By exploiting a behavior-tracking algorithm mutuated from game theory, we allow traffic to be forwarded only towards cooperative nodes. We hence prove that we can reduce power wastage at the same time maximizing the delivery rate. With the mentioned approach, selfish nodes are isolated from the network unless they decide to start cooperating. Our experimental tests aim at verifying the quick reaction time in response to variable nodes' behaviors as well as presenting a comparative analysis of the actual energy spent to successfully send traffic towards destinations.

1. Introduction

Ad hoc networks are composed of several nodes with wireless connection capability. Differently from wired networks, in an ad hoc environment each node is an end system and a router at the same time. A transmission between a sender and a receiver happens with the help of one or more intermediate nodes that are requested to relay packets according to routing protocols designed for this kind of networks. Message forwarding hence relies on a blind trust agreement among nodes. However, wireless nodes have often limited power resources, and some of them spend most of their time relaying other nodes' packets rather than sending their own data. Thus, a good percentage of power is wasted to serve other nodes. Besides, the open nature of the current ad hoc network protocols raises a number of security concerns. In fact, although in the recent past there have been proposals [1,2] of protocol modifications to enhance security, at present the aggregation of new nodes is usually uncontrolled and open to potential malicious users. In such a situation, a generic node of the network has to decide whether to trust or not to trust the other nodes. This obviously calls for a capability of each single node to somehow interpret (or, even better, predict) the behavior of the other nodes, since they represent fundamental *allies* in the data transmission process.

In this paper, we show how cooperation can be perceived by nodes as an incentive, thanks to the fact that it helps save the overall amount of energy needed for data transmissions. Differently from recent works proposed in the literature, which aim at making the routing process become *natively* aware of the energy-related parameters, we herein propose a different approach, by leveraging cooperation in order to improve the overall energy efficiency of an ad hoc network without modifying the existing routing protocol. Our work is indeed complementary to the above mentioned proposals, in that it can co-exist with any routing protocol, be it legacy or energy-aware. We try and exploit a different perspective on energy efficiency, which is much more related to the behavioral patterns of the nodes rather than to the specific mechanisms and protocols adopted in the network.

Delving into some of the details of how we deal with the behavioral aspects of the problem at hand, we present in the paper an algorithm to identify and isolate defecting nodes. The algorithm takes inspiration from the results of game theory and keeps a local trace of the behavior of the other nodes. At the beginning the behavior of all the other nodes is unknown, but as soon as the first flows of traffic are exchanged among them, each node becomes gradually aware of the past behavior of the others, which can be either cooperative or defecting. Once the defecting nodes are identified, different countermeasures can be adopted. The current version of the algorithm makes the decision of not relaying packets coming from defecting nodes as long as they do not cooperate, but other, less disruptive policies can be considered and included. The algorithm is implemented in an existing ad hoc routing protocol and is validated in the ns-2 simulator. The current experimental results highlight the induced reduction of throughput of defecting nodes.

The paper is organized in seven Sections. Section 2 deals with related work. Some background about game theory basics is

* Corresponding author. Tel.: +39 0823 302649.
E-mail addresses: maudarie@unina.it (M. D'Arienzo), folivier@unina.it (F. Oliviero), spromano@unina.it (S.P. Romano).

0140-3664/$ - see front matter © 2012 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.comcom.2012.05.004

provided in Section 3. Section 4 presents the algorithm we designed to infer behavioral information about the network nodes, whose implementation is described in Section 5. Results of the experimental simulations we carried out are presented in Section 6, while Section 7 provides concluding remarks and proposes some directions of future work.

2. Related work

Energy efficiency represents a crucial challenge in any wireless infrastructure. In [15] a classification of the different strategies for reducing the energy consumption in wireless networks has been provided. In particular, the authors identify four categories: (i) *energy efficient routing*, which minimizes the power consumed to transmit a packet by selecting an appropriated "low-energy" path; (ii) *node sleeping state scheduling*, which is capable to save energy by continuously selecting the nodes whose configuration has to be forced in sleeping mode; (iii) *topology control by tuning node transmission power*, which optimizes the transmission power of the single nodes in order to guarantee data delivery, network connectivity and energy efficiency; (iv) *volume information reduction*, which consists in data aggregation, wasteful transmission limitation, and control messages reduction. As concerns energy efficiency routing, in particular, a great amount of energy-aware protocols have been proposed [8,12,11,10,9]. They can be roughly classified based on their specific goals. They can in fact try to: (i) minimize the total power needed to transmit packets; (ii) maximize the lifetime of every single node; (iii) minimize the total power needed to transmit packets at the same time maximizing the lifetime of every single node. Some interesting energy-efficient route selection schemes, falling in one of the previous categories, are presented in [8] and briefly described in the following.

Minimum Total Transmission Power Routing (MTPR) is a routing protocol aimed at minimizing overall power consumption in ad hoc networks. Given a source s and a destination d, we denote with P_r the total transmission power for a generic route r from s to d. P_r is the sum of the power consumed for the transmission between each pair of adjacent nodes belonging to r. MTPR selects the route r^* such that $r^* = min_{r\in R}P_r$, where R is the set containing all possible routes from s to d. A simple shortest path algorithm can be used to find this route.

Minimum Battery Cost Routing (MBCR) associates each node n_i in the network with a weight $f_i(c_i(t)) = 1/c_i(t)$, where $c_i(t)$ is the battery capacity level of n_i at time t. Given a source s and a destination d, if we say E_r the sum of the nodes weights of a generic route r from s to d, MBCR selects the route r^* such that $r^* = min_{r\in R}E_r$, where R is the set containing all possible routes from s to d. Such a scheme will always choose routes with maximum total residual energy.

With *Min–Max Battery Cost Routing* (MMBCR), starting from the above definition of $f_i(c_i(t))$, for each route r from a source s to a destination d, a cost is defined as $C_r(t) = max_{i\in r}f_i(c_i(t))$. The chosen route r^* verifies the relation $C_{r^*}(t) = min_{r\in R}C_r(t)$. MMBCR safeguards nodes with low energy level because it selects the route in which the node with minimum energy has more energy, compared to the nodes with minimum energies of the other routes.

Conditional Max–Min Battery Capacity Routing (CMMBCR) proposes an approach based on both MTPR and MMBCR. Let us consider the node of a generic route r from a source s to a destination d, with lowest energy. Let also $m_r(t)$ be its energy, and R the set of all the routes from s to d. If some paths with $m_r(t)$ over a specific threshold exist in R, one of these will be chosen using the MTPR scheme. Otherwise, the route r^* satisfying the relation $m_{r^*}(t) = max_{r\in R}m_r(t)$ will be selected. This scheme suffers from an unfair increment of the forwarding traffic towards nodes with more energy [10].

Minimum Drain Rate (MDR [11]) proposes a mechanism which takes into account node energy dissipation rate, thus avoiding the above problem. MDR defines for each node n_i a weight $C_i = RBP_i/DR_i$, where RBP_i is the residual battery power and DR_i the drain rate of n_i. Intuitively, DR_i represents the consumed energy per second in a specified time interval. Now, let C_r be the minimum weight of a generic route r from a source s to a destination d. MDR selects the route r^* such that $C_{r^*} = max_{r\in R}C_r$. In this way, residual energy level, as well as the energy consumption rate due to the incoming traffic to be forwarded, are jointly taken into account.

Generally, stimulating cooperation among network nodes might contrast the need for the nodes to preserve lifetime of their batteries. For example, a node close to a gateway can mainly consume its battery to forward neighbors' data rather than to send its own packets, thus reducing its overall achieved throughput. However, this statement seems to contradict the results of several recent works. Although cooperation entails an additional power consumption, it nonetheless seems to favor effective average energy consumption of the overall network. Several solutions have in fact been proposed in the last years in order to stimulate cooperation with the final aim of increasing energy savings. In [16] the authors provide a scheme for encouraging nodes to cooperate in order to save energy. By implementing a reward/punishment scheme at the physical layer with a Decode-Forward (DF) algorithm and by using a game theoretic model, the work shows how the presence of altruistic nodes can stimulate the selfish nodes to cooperate in order to improve energy efficiency. Similarly, in [18] the authors consider the effect of selfishness on energy efficiency by using a non-cooperative game approach to implement a relaying transmission scheme. The work verifies that a cooperative equilibrium is achieved in the presence of both a fading and a non-fading channel. On the other hand, cooperation in multi-channel coordination schemes is introduced in [17]. The authors propose a strategy called *altruistic cooperation* for cooperative multi-channel MAC protocols. Thanks to the presence of some "altruistic" nodes disseminated in the network, whose role is to acquire and share information about the channel's utilization, it is possible to limit the energy consumption of the other nodes, which do not have to stay awake in order to gather such information. Wireless routing algorithms can also improve energy efficiency by stimulating cooperation. For example in [19] the authors propose a new routing algorithm, which is also based on a game theoretic model. The main goal in this case is to reduce energy consumption of the overall network.

Similarly to the above mentioned works, we propose a new solution to increase nodes cooperation in an ad hoc network, and we verify how this approach actually induces advantages in terms of energy consumption reduction. In order to implement such a mechanism, an existing routing protocol, the AH-CPN (Ad-Hoc Cognitive Packet Network) [6], has been adopted and enhanced with new functionality specifically conceived in order to stimulate nodes' cooperation. Unlike routing protocols which are intrinsically energy efficient, in this paper we do not embrace an approach aimed at modifying routing in order to let it become energy-aware. We rather propose to induce network nodes to cooperate in return for a significant improvement of the achievable performance. The choice of the AH-CPN protocol has been suggested by its specific peculiarities which meet the requirements of our solution, and hence simplify its implementation. However, the approach can be extended to any other protocol, provided that one implements the required mechanisms.

In our case, cooperation incentives have been designed based on the results of game theory applied to wireless networks. Some

basic information about game theory will hence be provided in the following section.

3. Game theory basics

Game theory is a branch of applied mathematics which witnessed a great success thanks to the application of its results to a wide range of fields, including social sciences, biology, engineering and economics. Game theory covers different situations of conflicts regarding, in a first attempt, two agents (or *players*), and in the generalized version, a population of players. Each of these players expects to receive a reward, usually named *payoff*, at the end of the game. The basic assumption is that all the players are self interested and rational: given a utility function with the complete vector of payoffs associated with all possible combinations, a rational player is always able to place these values in order of preference even in case they are not numerically comparable (e.g., an amount of money and an air ticket). This not necessarily means that the best value will be selected, since the final reward of each player is strongly dependent on the decision of the other players. Each player is then pushed to plan a *strategy*, that is a set of actions aimed at total *payoff* maximization, provided that he is aware that the other players will try to do the same.

Games are classified in several categories according to various properties. If the players tend to be selfish in the achievement of the best payoff, the game is classified as *non cooperative*. When there is a common knowledge of the utility function for all players, the game is with complete information, otherwise it is considered incomplete. There are several real world examples that fall in one of these categories. Here we are mainly interested in the difference between strategic and extensive games. In strategic (also known as *static*) games the players make their decision simultaneously, without any knowledge of the others' intention. Even if the game is repeated, the players are still unaware of others' plans and do not have the chance to react to a previous action. This last opportunity is instead available in case of extensive games. Such games are played more than once and the players can evaluate what the others did at least during the last tournament, so that they can potentially decide to modify their strategy for the next move. Also, the payoff is cumulated at the end of each round rather than accounted for only once.

One of the fundamental problems of game theory is known as *prisoner's dilemma*, which can be represented in the matrix format of Fig. 1: two suspects of a crime are arrested and jailed in different cells with no chance to communicate between each other. They are questioned by the police and receive the same deal: if one confesses (*defect*) and the other stays silent (*cooperate*), the first is released, the second is convicted and goes to prison with a sentence of 10 years, the worst; if both stay silent (*cooperate*), they go to prison for only 1 year; if both testify against the other (*defect*) they go to prison with a sentence of 5 years. The situation in which they both stay silent (*cooperate*) is the more convenient to both of them; however, it was demonstrated that a rational behavior is to confess (*defect*) and receive the sentence of 5 years, and this situation represents the only equilibrium, as first introduced by Nash [14,13]. Hence, the prisoner's dilemma falls in the field of strategic non-cooperative games.

In its basic form the prisoner's dilemma is played only once and has been applied to many real life situations of conflict, even comprising thorny issues of state diplomacy. Another version of the prisoner's dilemma is played repeatedly rather than a just once and is known as iterated prisoner's dilemma (IPD), which turned out to be a cooperative game under certain circumstances [3,4]. The goal of both players still is the maximization of their payoff, as the cumulated payoff earned at each stage. If the number of

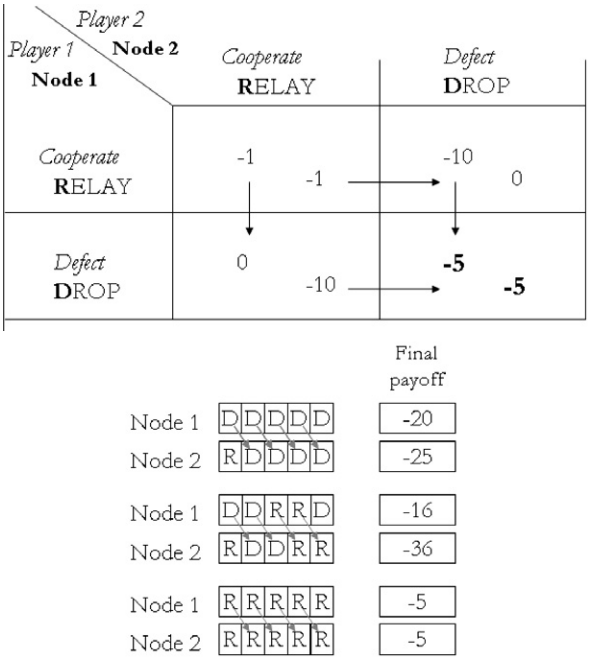


Fig. 1. The tit-for-tat strategy in action.

rounds is finite and known in advance, the strategy of always defecting is still the only situation of equilibrium and the game is still non-cooperative. However, in case the number of repetitions is infinite, it was demonstrated that the choice to always defect is not the only equilibrium as even the choice of cooperating may be an equilibrium. In this case, one of the strategies that let players maximize their payoff is the so-called *tit for tat* game, in which each player repeats the past behavior of the other player: a player is keen to cooperate if the other node behaved correctly the last time, otherwise it defects. We base our algorithm to mitigate node selfishness on the results of this version of the game.

3.1. A game theory model applied to ad hoc networks

Game theory is mainly represented in two *forms*: the *extensive* form, first introduced by Von Neumann in 1928 and then improved by Kuhn in 1953 [20], which illustrates the time sequence of moves on graphic trees with each leaf representing a point of decision; the *normal* (or *strategic* form), due to Von Neumann and Morgenstern [21] and Shubik (1982), models the game as $G = (N, U, S)$ with N the set of players, S the strategy space, and U the payoff (or utility) linked to a particular set of strategies. The players, their strategies and related payoffs are often schematically represented in a matrix. For each player, a set of strategies $S_i = (s', s'', \ldots)$ is available. We define $s = (s_i^*, s_{-i})$ as the strategy $*$ chosen by the ith player in combination with those selected by all the remaining players, indicated with s_{-i}. U is then equal to $(u_1(s), \ldots, u_i(s), \ldots, u_n(s))$, being $u_i(s)$ the payoff of player i when strategy s is played.

Although the operations in an ad hoc network involve multiple nodes, and should then be represented as an *n-player* game, the step underlying any transmission is concerned with two nodes, the former which requests the relay of a packet, the latter which is in charge of making a decision about that request. This situation iterates, ideally an infinite number of times, with players acting either the same or the reverse role. Given such assumptions, the single hops in an ad hoc network can be modeled as an iterated prisoner's dilemma with

$$S_1 = S_2 = (R, D)$$

meaning that the strategy of each node can be to either relay (R) or drop (D) packets. The set of available strategies is then limited to

$$s = (R,D),(R,R),(D,R),(D,D)$$

and the utility function is

$$U = (u_1(s), u_2(s))$$

being $u_i(s)$ an index that is in direct ratio to the energy spent by node i to ship packets to the final destination and in inverse ratio to the total number of bytes successfully delivered. The expected goal is the minimization of function U, as this maximizes the delivery ratio at the same time minimizing the energy consumption of involved nodes. As mentioned above, players would not reach this result in a single tournament, or when the number of tournaments is known in advance, because the dominant strategy is stuck on $s = (D,D)$ and yields a sub optimal payoff to both players. However, one of the repeated versions of prisoner's dilemma, known as *tit for tat*, can lead to a second equilibrium with an optimal payoff if players (or nodes) decide to cooperate. As an example, if we consider the first five tournaments of a *tit for tat* game in the bottom part of Fig. 1, a node that always defects would play (D,D,D,D,D) and earn $(0,-5,-5,-5,-5) = -20$ against the opponent (R,D,D,D,D) that would earn -25. If the first node decides to relay packets two times out of five (D,D,R,R,D), he would earn $(0,-5,-10,-1,0) = -16$ against (R,D,D,R,R) that sums up to -36. In case the first node decides to always relay packets, both its own payoff and the opponent's payoff will amount to $(-1,-1,-1,-1,-1) = -5$, which is the best they can both achieve. So, continued cooperation in terms of packet relaying for the iterated prisoner's dilemma also yields the best payoff as long as nodes do not defect. However, in lack of a *tit for tat* mechanism, defecting nodes would have the chance to freely increase their payoff at the expense of unaware cooperative ones. Thus a side effect of the tit for tat introduction, which is of great interest to us, is that it stimulates cooperation since it becomes no longer convenient to defect. In next sections we aim at experimentally proving this outline, even comprising the evaluation of payoff in terms of energy consumption with respect to the bytes delivered.

4. A novel behavior-tracking approach

In an ad hoc network, the number of nodes and links can change over time, so we consider (see Fig. 2) the number of nodes $N(t)$ as a function of time t. We also define a dynamic array $C(t)$ of $N(t)$ elements for each node of the network. The generic element $c_i(t)$ of $C(t)$ assumes the values (UNKNOWN, COOPERATE, DEFECT), meaning that the behavior of node i at time t is respectively unknown, cooperative or non cooperative. At time $t = 0$ all the values are set to UNKNOWN, since at the beginning each node is not aware of the behavior of the other nodes.

We herein propose to introduce a routing control strategy at each node based on the game theory framework described earlier. Basically, the control algorithm first identifies the cooperative nodes (Detection Phase) and then reacts in the most appropriate way in order to give priority to packets generated by cooperative nodes. Specifically, the control algorithm is composed of the two following phases:

Detection phase: Suppose the generic node s of the network needs to send some traffic to the destination d. The first task is to discover an available path, if it exists, to reach the destination. To this purpose, we consider a source based routing protocol capable of discovering a list $A(t)_{(s,d)i}$, $\forall i : 0 < i < P$, of P multiple paths. All the nodes in the list $A(t)_{(s,d)i}$ are considered under observation and marked as probably defecting in the array $C(t)$ unless a positive feedback is received before a timeout expires. The sender s starts sending his traffic along all the discovered paths. If the destination node generates D acknowledgment messages containing the list of all the nodes $L_{(s,d)i}$ (with $0 < i < D$) traversed, as it happens in some source based routing protocols, the sender s is informed about the behavior of intermediate nodes. For each acknowledgment message received, the sender s can make a final update of the array $C(t)$ by setting the matching elements $c_i(t)$ contained in the list $L_{(s,d)i}$ as cooperative. Notice that the last update overwrites the previous stored values and represents the most recent information concerning the behavior of a node.

Reaction phase: Once done with the detection phase, each node is aware of the behavior of other nodes and can react in the most appropriate way. For example, a node can refuse to relay packets of defecting nodes, or operate a selective operation like queuing their packets and serving them only if idle and not busy with the service requested by cooperative nodes. In this first proposal, we rely on the harsh policy of packet discarding, and this brings to the isolation of defecting nodes. However, a defecting node can even gain trust of other nodes if it starts to cooperate. The array $C(t)$ is not static over time and its values are continuously updated. In fact, due to the dynamic evolution of ad hoc networks, the search of available paths is frequently repeated, and the list $A_{(s,d)}$ consequently updated. Hence, if a defecting node decides to cooperate, its identification address will be included in one of the acknowledgment messages $L_{(s,d)i}$ sent to the sender s and its aim to cooperate will be stored in the array $C(t)$.

The situation described here for the pair (s,d) is replicated for all possible pairs of nodes that try to interact, but each node stores only one array $C(t)$ that is updated upon reception of any acknowledgment message, wherever it comes from. Furthermore, not all relayed packets are checked in order to verify the nodes' behaviors,

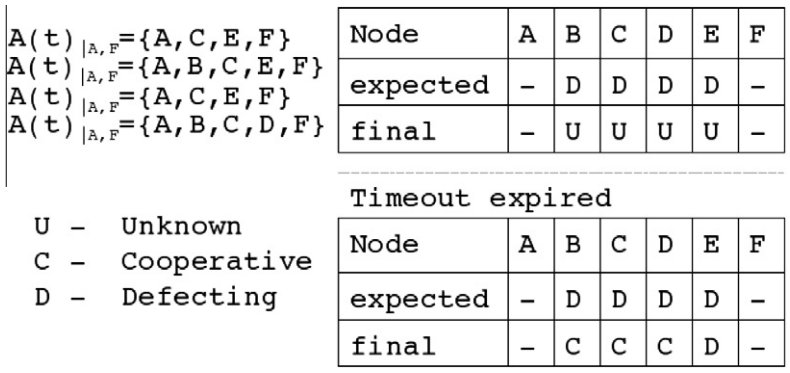


Fig. 2. Algorithm description.

but only a sample of them, thus keeping the total overhead under control.

5. Algorithm implementation

Although several dedicated protocols have been designed and implemented to properly manage wireless ad hoc networks, they still do not support any mechanism to keep trace of nodes' past behavior. We introduced the algorithm presented in the previous section in an existing source based routing protocol for ad hoc networks. We first modified this protocol to support the search of multiple paths, and then included the new algorithm for the identification of non cooperative nodes, as described in the next two subsections.

5.1. Multipath source based routing in ad hoc networks

The proposed behavior-tracking algorithm relies on an acknowledgment (or, more precisely, on a "missed acknowledgment") technique. Among the many ad hoc routing protocols, AH-CPN (Ad Hoc Cognitive Packet Network) [5] is designed to support QoS and make an intense use of acknowledgment messages independently from the transport protocol in use. AH-CPN is the wireless version of CPN (Cognitive Packet Network) [6], a proposal for a self aware network architecture with native support for QoS. Both in AH-CPN and CPN, the presence of a neural network engine is able to undertake dynamic and fast routing decisions as soon as a condition, like for example a congested link or a different user's requirement, has changed. An always active traffic of smart packets (SP) discovers new paths according to specific QoS goals, e.g., discovering paths that minimize the delay or maximize the throughput. This information is made available to the interested nodes that can send traffic along the defined path on the basis of a source based routing technique. Smart packets keeps on looking for the specific goals, and in case a better path is found, the sender is informed and can update its routing path. SPs are only in charge to discover new paths. They are initially sent via flooding to discover a first usable path. An identification number is used to prevent loops, so that SPs with the same ID touching a node for the second time are discarded.

Besides the smart packets, AH-CPN provides for three other kinds of packets: Smart Acknowledgment (SA), Dumb Packets (DP), and Dumb Acknowledgment (DA), whose role is hereafter described.

SA packets are generated every time a SP successfully reaches a target destination and are routed along the reverse path up to the source node to inform it about the last discovered path in compliance with the source based routing theory. CPN differs from other similar protocols in the way each hop is selected, since the decision is made by properly weighing both the current and past performance of each outgoing link [7].

Once a SA is at destination, the source node has finally the chance to prepare one or more DPs to send actual data across the network. DPs contain the whole discovered path in a dedicated field. Internal nodes relay DPs to the next hop excerpted from such field, while also adding timestamp information useful to evaluate the round trip time (RTT) between each pair of nodes along the path. RTT data are stored in special *mailboxes* available at each node and provide the routing algorithm with precious information concerning the past behavior of a link. Finally, DAs are generated upon reception of DPs. Notice that differently from other protocols, in CPN these acknowledgments are generated for every single DP received, independently from the transport protocol employed. This feature proves helpful in the deployment of our algorithm for the identification of defecting nodes, as better explained in the next subsection.

The basic CPN version looks for one available path, representing the best in terms of the requested QoS goal. We modified this protocol to search for multiple paths. To this purpose, we slightly changed the SPs original flooding prevention mechanism to collect up to four different paths for each source/destination pair. Each sender can then collect the different SAs to be stored in its routing table. DPs are initially sent along all of the discovered paths to detect at least a cooperative one, which will be used for the subsequent transmissions. The other paths are periodically checked to verify the cooperation state of other nodes. Notice that even after a successful collection of multiple paths, the transmission of SPs is not terminated; it is rather repeated periodically, both for path maintenance and to check whether the topology has changed. As soon as new paths are discovered, these will also update the multiple paths table.

5.2. Identification and isolation of defecting nodes

We provide the multipath source based routing protocol with the support for identification and isolation of defecting nodes. The array $C(t)$ is constructed and stored at each node and its dimension can change according to the number of nodes active in the ad hoc area. When node a needs to send traffic to node b, SPs are immediately sent in flooding. We make the assumption that non cooperative nodes try to cheat by forwarding inexpensive SPs, that do not carry any payload, while they do not relay DPs containing the real data. In fact, in case the non cooperative nodes should naively decide to simply block the SPs forwarding process, they would be immediately discovered as non cooperative and have no chance to keep on cheating.

Thanks to the multi-path extension, at the end of the process node a becomes aware of one or more paths, if they exist, and these are all stored in the array $A(t)_{(a,b)}$. At the time of the first transmission, the real data are packed in as many DPs as the number of discovered paths and sent towards the interested nodes. Since in CPN a destination b must send an acknowledgment message DA whatever the transport protocol is, node a will receive only the DAs from those paths composed of all cooperative nodes, while DAs from paths with at least a cheating node will not be generated. This information, as described before, helps finalize the array $C(t)$ with the list of both cooperative and defecting nodes. Traffic will hence be sent only along one cooperative path, if it exists, rather than towards all the available paths. When one of the cheating nodes requests the relaying of a message to node a, it is aware of his past behavior and can decide to drop all its packets, while it can regularly relay packets coming from cooperative nodes.

The situation just described concerning both the cooperation and the selection of paths is not static; it can rather change over time, so isolated nodes are not banned forever from the network. Although the traffic from a node is delivered only along paths composed of cooperative nodes, sending nodes continue to perform periodically the multiple paths collection process, even along the paths containing the defecting nodes. Should a defecting node decide to change its behavior and begin to cooperate, our algorithm soon detects this change and admits again the node to the transmission of flows. In this way, a node reacts following a *tit for tat* strategy. The first series of experiments presented in Section 6.1 has been specifically conceived to verify the responsiveness of the algorithm (i.e., its quick reaction to changes in the nodes' behavior).

6. Experimental results

We implemented and tested the proposed system in the ns-2 simulator. The implementation of the algorithm relies on a

dedicated multi-path ad hoc routing protocol that supports the explicit acknowledgment of packets regularly received at the final destinations. The refresh rate of array $C(t)$ is currently set to 100 msec.

To highlight the robustness of the algorithm, we designed two different scenarios aiming at: (i) evaluating the nodes' responsiveness against swinging nodes' behaviors; (ii) comparing the average network performance achieved by cooperative nodes with respect to defective ones.

6.1. Tit for tat in action

The first scenario is based on a simple wireless testbed composed of 8 nodes (see Fig. 3), labeled from 0 to 7. In such a network we set up the following conditions: (i) node 3 defects all the time; (ii) the behavior of node 4 dynamically changes over time; (iii) all the other nodes are cooperative. The duration of the experiments is set to 12 min. The defection of a node means that the relay of traffic to serve other nodes is totally stopped, so the percentage of node 3's cooperation is 0% (of the total time). As far as node 4 is concerned, five situations are considered, most of them offering the other nodes the chance to reply with a *tit for tat* strategy:

1. Node 4 never cooperates. Requests of relay are never forwarded, so the percentage of cooperation is 0%;
2. Node 4 follows a switching behavior: each 3 min interval, node 4 defects for the first 2 min and then cooperates for the remaining minute; the total percentage of cooperation is hence 25%;
3. Node 4 still switches its behavior: each 2 min interval, it defects and cooperates in equal parts; in this case the total percentage of cooperation is thus 50%;
4. Node 4 switches its behavior in a way that is opposite to the one described in the second item of this list: each 3 min interval, node 4 defects for the first minute and cooperates for the last 2 min; the overall percentage of cooperation is, in this case, 75%;
5. Node 4 always cooperates; all relay requests are served, for a final percentage of cooperation of 100%.

Two equal sessions of constant bit rate traffic are activated between node 4 and node 0 and node 1 and node 7, respectively at time 1.0 and at time 2.0. In the ideal situation of all cooperating nodes, the shortest paths would be (4,3,0) and (1,2,4,7). However, node 3 is always defecting, so the path (4,3,0) turns out to be unavailable and the traffic coming from node 4 is forced along the other available path (4,2,1,0). As long as node 1 does not generate traffic, it does not have the chance to track the behavior of node 4, so the relay requests coming from node 4 are regularly served. At time 2.0 node 1 begins the discovery of paths to reach node 7. Besides the other choices, the best path (1,2,4,7) is soon discovered and selected to immediately generate traffic. If node 4 follows a switching behavior, then node 1 has the chance to react in compliance with the *tit for tat* strategy. Notice that in case node 4 is in a defecting state, node 1 still can send traffic to the destination along the path (1,2,5,6,7).

In Fig. 4a we evaluate the delivery ratio of node i as the ratio $dr_i = r_i/s_i$ between the total number of bytes correctly received at the destination and the total number of bytes sent during the experiment. The x axis represents the percentage of node 4's cooperation, the y axis is the final delivery ratio dr_i. On the left part of the x axis in Fig. 4a, node 4 is fully defecting; the same applies to node 3. Traffic from node 4 towards node 0 is regularly sent between time 1.0 and time 2.0 because node 1 did not generate any request and did not yet check the behavior of the other nodes. At time 2.0, however, node 1 tries to send traffic to node 7 and hence has the chance to verify the behavior of the other nodes. Among the other discovered paths, it realizes that paths comprising nodes 4 and 3 are not working, so as soon as the timeout expires it marks nodes 3 and 4 as defecting and immediately thereafter stops relaying traffic coming from node 4. The final delivery ratio dr_1 of node 1 almost reaches the ideal value because the alternative path (1,2,5,6,7) is soon discovered and used for the entire duration of the experiment. Delivery ratio dr_4 is instead severely impacted.

As node 4's percentage of cooperation increases up to 100%, delivery ratio dr_4 increases until it reaches a value close to delivery ratio dr_1 when there is full cooperation. Although node 3's defection makes the path (4,3,0) unavailable, the routing protocol discovers the alternative path (4,2,1,0) composed of cooperative nodes, while the shortest path (1,2,4,7) is regularly available in this case. This is the only situation in which node 4 maximizes its delivery ratio. In the intermediate cases the trend is linear and clearly demonstrates the correct implementation of the *tit for tat* reaction mechanism, as node 1 cooperates only when node 4 does the same. Delivery ratio dr_1 remains more or less unaltered independently of node 4's behavior, thanks to the fact that node 1 has a chance to discover alternative cooperative paths.

We compared these results with the situation in which the nodes are unable to detect the defecting behavior. We mark these sessions with *NT* (with *NT* standing for *No Tracking* of nodes' behaviors) in the same figure (Fig. 4a). The situation is now opposite to the previously analyzed case because delivery ratio dr_4 outperforms dr_1 in the case of node 4's full defection. Node 1 is now unaware of node 4's defection; hence, while its traffic is not relayed, it regularly relays the incoming packets having node 4 as source. Anyway, both delivery ratios dr_1 and dr_4 are lower than those measured in the previous case. This time the lack of tracing of nodes defection affects even node 4's performance, since such node tries to forward traffic not only along the working path (4,2,1,0) but

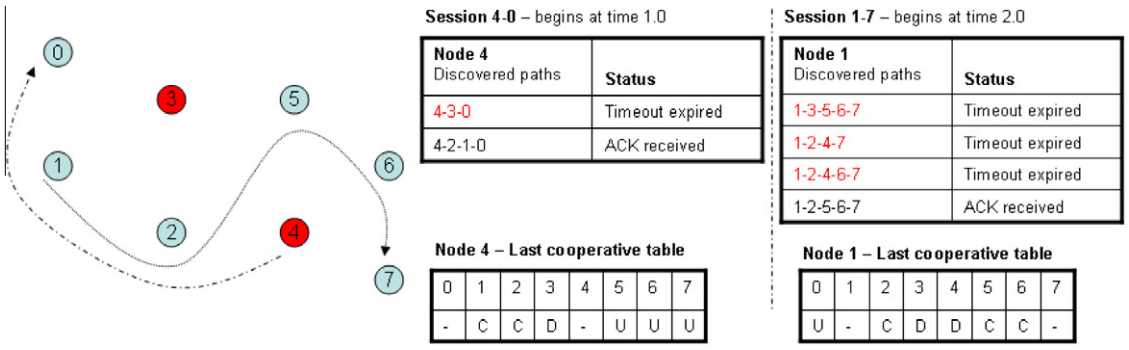
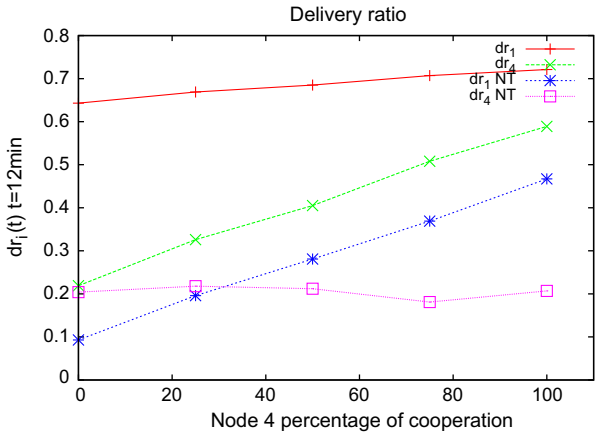
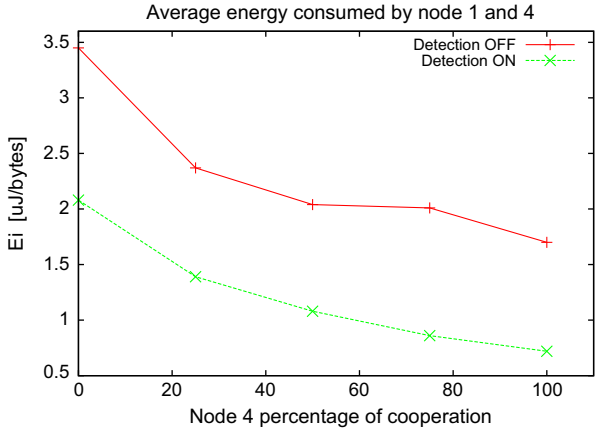


Fig. 3. The first testbed.



(a)



(b)

Fig. 4. Node 1 and Node 4 network performance.

also along the uncooperative path (4,3,0), which explains the halved final delivery ratio.

Finally, notice that if we consider the average (cumulative) energy consumed by both node 1 and node 4 in case the detection of defecting nodes is enabled, such value is always lower compared to the case when detection is disabled, as showed in Fig. 4b.

6.2. Overall network performance comparison

In this second series of experiments we aim at proving how the introduction of a system able to detect defecting nodes composing a wireless ad hoc network can lead to a better distribution of the energy consumed by each node. To this purpose we introduce a deeper energy analysis of the system besides the final delivery ratio evaluation. The experiments are executed on a 24 nodes testbed, depicted in Fig. 5, to compare the average network performance as the percentage of nodes' cooperation increases. In this scenario the nodes' behavior is selected at the beginning of each experiment and does not change over time. The different configurations are (i) all nodes cooperate (100%), (ii) nodes 5 and 13 defect (about 90% of nodes cooperate), (iii) nodes 5, 13, 10, and 18 defect (about 85% of nodes cooperate), (iv) nodes 5, 6, 13, 10, 17 and 18 defect (75% of nodes cooperate).

Sixteen sessions of constant bit rate traffic are generated between several pairs of nodes, as indicated by the arrows in the picture (e.g., node 5 generates traffic towards both node 3 and node 7, node 14 sends data to nodes 12 and 16, etc.). To avoid traffic overlaps, just a single session is active per time slot, with time slots (each lasting 0.25 sec) alternating over time in a cyclic fashion. At the end of each experiment, which has a total duration of 6 min, we measure the average delivery ratio achieved by the cooperative nodes set against the same average ratio attained by the set of defective ones. We repeat the same comparison with respect to the average energy spent by a node to successfully deliver a byte to the destination, which we calculate for each node as:

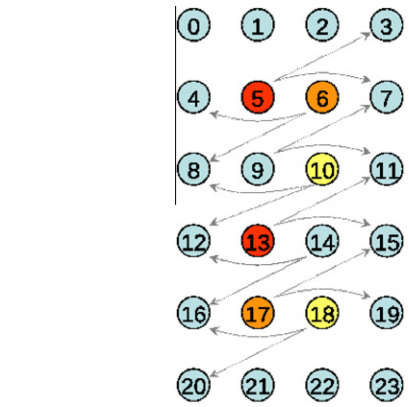
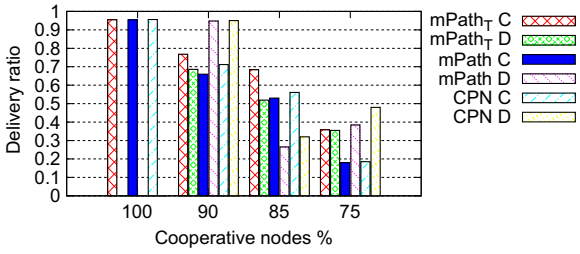
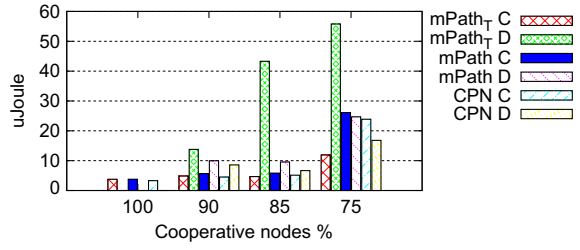


Fig. 5. The second testbed.



(a) Delivery ratio



(b) Energy per byte

Fig. 6. Comparison between average network performance.

$$S_i = \frac{Ec_i}{(s_i + rl_i)} * \frac{s_i}{r_i}$$

being Ec_i the total energy consumed by node i, s_i the total number of bytes sent to the destination, rl_i the number of bytes relayed from node i, and r_i the bytes correctly received at destination. S_i has a dimension of [Joule/bytes].

As a further comparison, the experiments are repeated on three different testbed configurations, which we mark with $mPath_T$ when the tracing algorithm is enabled, *mPath* when only the multi-path algorithm is available, and finally with *AH-CPN* in the presence of the basic CPN protocol.

All the results of this series of experiments are collected in the histograms presented in Fig. 6, with values related to cooperative and defective nodes marked respectively with letters *C* and *D*.

By observing Fig. 6a, we can notice how the final delivery ratio value is independent from the routing algorithm when all the nodes are cooperative. As soon as some of the nodes start to defect, the boxes in the histogram indicate the split between the average delivery ratio of cooperative nodes and the average ratio of defective ones. In all the three considered situations of nodes' defection, the final delivery ratio gives advantage to defective nodes at the detriment of cooperative ones except in the case of the $mPath_T$ algorithm, which keeps the delivery ratio of cooperative nodes at a value which is always greater than that attained by defective ones.

Coming to the average energy spent to successfully deliver a single byte to the destination (reported in Fig. 6b), we once again observe that the overall performance figures are in favor of the $mPath_T$ algorithm, which preserves the energy level of cooperative nodes in all the considered situations with respect to both the percentage of nodes' cooperation and the available routing algorithm.

The evaluation of the average energy consumption indicator S_i also provides an indirect evaluation of the overhead introduced by our tracing algorithm. If we look closely at the experiment results in case of full cooperation (Fig. 6b, vertical bars associated with a cooperation percentage of 100%), these values are: 3.78 μJ for $mPath_T$, 3.78 μJ for *mPath*, 3.30 μJ for AH-CPN. The difference among the three reported energy consumption levels gives an indication of the overhead due to the introduction of our algorithm. We can notice how $mPath_T$ exhibits a slightly greater consumption compared to the basic AH-CPN protocol. We can also derive that the main reason behind such energy consumption increase is ascribable to the multi-path nature of the tracing algorithm, since both $mPath_T$ and *mPath* reveal the same final value.

7. Conclusions

In this paper we showed how cooperation positively affects the performance of an ad hoc network, by helping reduce the overall energy consumption associated with data transmissions. We demonstrated through simulations that cooperation actually acts as an incentive for nodes, since it allows for a lower average energy expenditure per byte transmitted. We also studied the positive impact of cooperation on delivery ratio, which is considered a key performance indicator for any networked environment. Namely, we proved that the resulting network equilibrium achieved in the presence of cooperative nodes increases fairness in terms of energy consumed per unit of successfully delivered packets.

We do believe that the behavior-based approach that we presented in this work can be effectively exploited in a number of alternative scenarios, since it actually works along a dimension which turns out to be complementary to other potential approaches, like, for example, ad hoc designed energy-efficient routing paradigms or link layer strategies.

This work is clearly just a first attempt at studying the many facets of cooperation in ad hoc networks. Among the numerous improvements that we identified and which represent directions of our future work, we firstly mention a more detailed analysis of the dependence of the performance improvements deriving from cooperation on the specific network topology taken into account. Apart from this, we also intend to study how the specific location of a node in the ad hoc network topology affects its performance and consequently its willingness to cooperate. This requires that a thorough analysis of the tradeoff between relaying other nodes' packets and sending one's own data is conducted.

References

[1] F. Olivero, S.P. Romano, A reputation-based metric for secure routing in wireless mesh networks, in: IEEE GLOBECOM 2008, December 2008, pp. 1–5.
[2] K. Mandalas, D. Flitzanis, G.F. Marias, P. Georgiadis, A survey of several cooperation enforcement schemes for MANETs, in: IEEE International Symposium on DOI, 2005, pp. 466–471.
[3] R. Axelrod, The Evolution of Cooperation, Basic Books, 1988.
[4] R. Axelrod, D. Dion, The further evolution of cooperation, Science 242 (4884) (1988) 1385–1390.
[5] E. Gelenbe, R. Lent, Power-aware ad hoc cognitive packet networks, Ad Hoc Networks 2 (3) (2004) 205–216.
[6] E. Gelenbe, R. Lent, Z. Xu, Design and performance of cognitive packet networks, Performance Evaluation 46 (2–3) (2001) 155–176.
[7] E. Gelenbe, Learning in the recurrent random neural network, Neural Computing 5 (1) (1993) 154–164.
[8] C.K. Toh, Maximum battery life routing to support ubiquitous mobile computing in wireless ad hoc networks, IEEE Communications Magazine (2001).
[9] S. Mahfoudh, P. Minet, Survey of energy efficient strategies in wireless ad hoc and sensor networks, in: IEEE International Conference on Networking, Cancun, Mexico, 2008, pp. 1–7.
[10] Floriano De Rango, Marco Fortino, Energy efficient OLSR performance evaluation under energy aware metrics, in: Symposium on Performance Evaluation of Computer and Telecommunication Systems, 2009, pp. 193–198.
[11] D. Kim, J.J. Garçia Luna Aceves, K. Obraczka, J. Cano, P. Manzoni, Power-aware routing based on the energy drain rate for mobile ad hoc networks, in: Proceedings of IEEE 11th International Conference on Computer Communications and Networks, 2002, pp. 562–569.
[12] P. Sondi, D. Gantsou, Voice communication over mobile ad hoc networks: evaluation of a QoS extension of OLSR using OPNET, in: Proceedings of AINTEC'09, Bangkok.
[13] J. Nash, Non-cooperative games, The Annals of Mathematics 54 (2) (1951) 286–295.
[14] J.F. Nash, Equilibrium points in n-person games, Proceedings of the National Academy of Sciences of the United States of America 36 (1) (1950) 48–49.
[15] S. Mahfoudh, P. Minet, Survey of energy efficient strategies in wireless ad hoc and sensor networks, in: Proceedings of Seventh International Conference on Networking, 2008, pp. 1–7.
[16] L. Lai, H. El Gamal, On cooperation in energy efficient wireless networks: the role of altruistic nodes, IEEE Transactions on Wireless Communications 7 (5) (2008) 1868–1878.
[17] T. Luo, M. Motani, V. Srinivasan, Altruistic cooperation for energy-efficient multi-channel MAC protocols, in: Proceedings of the 13th Annual ACM International Conference on Mobile Computing and Networking, 2007, pp. 322–325.
[18] J. Yang, D.R. Brown III, Energy efficient relaying games in cooperative wireless transmission systems, in: Proceedings of Asilomar Conference on Signals Systems and Computers, 2007, pp. 835–839.
[19] S. Kim, Cooperative game theoretic routing algorithm based on Shapley-value approach, in: Proceedings of International Conference on Information Networking, 2009, pp. 1–3.
[20] H.W. Kuhn, Contributions to the Theory of Games II (AM-28), Princeton University Press, 1953, ISBN 978-0-691-07935-6.
[21] John von Neumann, Oskar Morgenstern, Theory of Games and Economic Behavior, Princeton University Press, 1944, ISBN 978-0-691-13061-3. 625 p..