EAI Endorsed Transactions
on Mobile Communications and Applications
Research Article ICST.ORG

A cooperation-based approach to energy optimization in wireless ad hoc networks

Maurizio D'Arienzo[1,*], Simon Pietro Romano[2]

[1]Dipartimento di Scienze Politiche, Seconda Università di Napoli - Italy
[2]DIETI, Università di Napoli Federico II - Italy

Abstract

A well known and still open issue for wireless ad hoc networks is the unfair energy consumption among the nodes. The specific position of certain nodes composing an ad hoc network makes them more involved in network operations than others, causing a faster drain of their energy. To better distribute the energy level and increase the lifetime of the whole network, we propose to periodically force the cooperation of less cooperative nodes while overwhelmed ones deliberately stop their service. A dedicated ad hoc network routing protocol is introduced to discover alternative paths without degradation in the overall network performance.

Received on 13 January 2015; accepted on 06 August 2015; published on 11 August 2015

Keywords: Cooperation, Energy efficiency, Wireless Ad Hoc Networks

doi:10.4108/eai.11-8-2015.150092

1. Introduction

Ad hoc networks are composed of wireless nodes with limited power resources usually provided by accumulators. In these networks each node is an end system and a router at the same time. The limited energy is then not only used to deliver one's own packets to the destinations but also to serve other nodes as message relayers [1] [2] [3]. Current routing protocols do not implement any mechanism to verify if other nodes are participating in relay operations. Thus, certain nodes have the chance not to cooperate [4]. This typically happens either because their resources are going to expire, and then they can decide to temporarily stop their service to avoid a premature shutdown, or because of a malicious behavior that turns them into cheater nodes that save power for their own transmissions rather than wasting energy to serve the others.

Both these situations can affect the final performance achieved by single nodes as well as that of the entire network. For instance, the presence of malicious nodes in the network can have a negative impact on the final delivery ratio achieved by well behaving nodes compared to that achieved by uncooperative ones, as we measured in the simple testbed represented in Figure 1 and composed of 25 nodes arranged in a $5x5$ grid topology. In the testbed, the distance between each pair of nodes is such that a transmitting node is obliged to use one of its adjacent nodes as the next hop rather than relying on further, non adjacent ones. Several sessions of traffic are generated between pairs of nodes, as we will detail in Section 5. Nodes $1, 3, 6, 8, 12, 16, 18, 21, 23$ can adopt a defective behavior, that is, they can refuse to relay packets for other nodes. The remaining nodes are instead always cooperative. All the nodes start from an initial energy level of $1000J$; at the end of the experiments we measure both the delivery ratio dr_i and the average residual energy of all the nodes.

As a preliminary observation, Figure 2(a) shows how the presence of malicious nodes affects the performance of well behaving ones, since the final delivery ratio is in favor of defective nodes. As the number of malicious nodes increases, the performance of both cooperative and uncooperative nodes is affected compared to the ideal case of full cooperation, albeit still in favor of uncooperative ones. These results are consistent with the lack of a system that tracks the behavior of other nodes.

At the opposite, too much cooperation can lead to an unfair energy consumption because certain nodes, usually the inner nodes of a topology, are more involved

*Corresponding author - e-mail: maudarie@unina.it

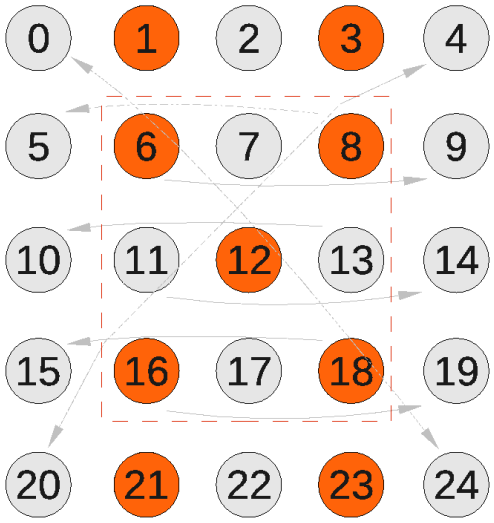
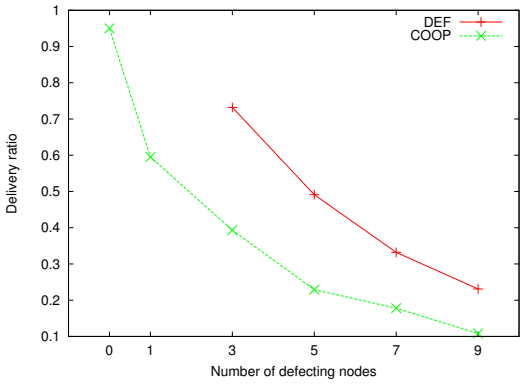
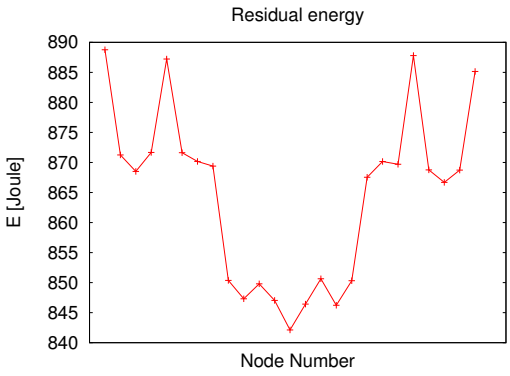


Figure 1. The simulated testbed



(a) Final delivery ratio in presence of uncooperative nodes



(b) Final residual energy in case of full nodes cooperation

Figure 2. Performance unfairness in a simple 25 nodes testbed

than others in relay operations and this causes a greater drain of their energy compared to that spent by border nodes. In Figure 2(b) we measured the residual nodes energy at the end of experiments with low energy levels associated with the inner nodes (those rounded by the dashed rectangular box in Figure 1). To deal with such unfairness in performance we propose to introduce a system for the detection of nodes' behavior in ad hoc routing protocols.

In the presence of a behavior detection system, well behaving nodes can deal with uncooperative ones, for instance by refusing to relay packets coming from them. In this way, the malicious nodes experience a delivery ratio reduction and are in fact rather pushed towards a more cooperative attitude. We also aim at minimizing the residual energy variance, that means to improve the fairness among nodes and avoid an irregular shut down of single nodes. To this purpose, overloaded nodes can periodically switch to a defection state to preserve their own energy, hence inducing a better energy balance among the nodes. By leveraging our behavior detection system, the transmitting nodes can soon realize that some paths are temporarily unavailable and have the chance to search for alternative paths, thus keeping the delivery ratio unaltered.

Rather than introducing new energy aware features into the routing protocols, we rely on a decentralized algorithm to track down the behavior of network nodes in order to quickly find alternative paths. We demonstrate that the proposed approach allows for a better balance of energy among the nodes. The algorithm takes inspiration from the results of game theory and enhances an existing ad hoc routing protocol. It has been implemented and tested in a simulated environment. Next sections introduce, respectively, the proposed novel algorithm, the modifications applied to an existing routing protocol, the experimental results, as well as relevant related work. Finally, the last section provides concluding remarks and gives some insights into future investigations.

2. Game theory applied to ad hoc networks

Although game theory is a branch of applied mathematics, it witnessed a great success thanks to the application of its results to a wide selection of fields, including social sciences, biology, engineering and economics, as welll as the study of ad hoc networks [4]. 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 aiming at total *payoff* maximization, provided that he is aware that the other players will try to do the same. Games can be classified according to various properties. Here we are mainly interested in the difference between *cooperative* and *non-cooperative* games as well as the difference between *strategic games* (played once) and *extensive games* (played many times).

One of the fundamental problems of game theory is known as *prisoner's dilemma*, which can be represented in the matrix format of Figure 3: 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 [7] [6]. 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. A different version of the prisoner's dilemma is played repeatedly rather than just once and is known as iterated prisoner's dilemma (ITD), which turned out to be a cooperative game under certain circumstances [8][9]. The goal of both players still is the maximization of their payoff, as the cumulated payoff earned at each stage. If the number of 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 also 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. If we consider the first five tournaments of a two players game, a player who defects (D) against a cooperative (C) player adopting the tit for tat strategy would play (D,D,D,D,D) and earn $(0,-5,-5,-5,-5) = -20$. If

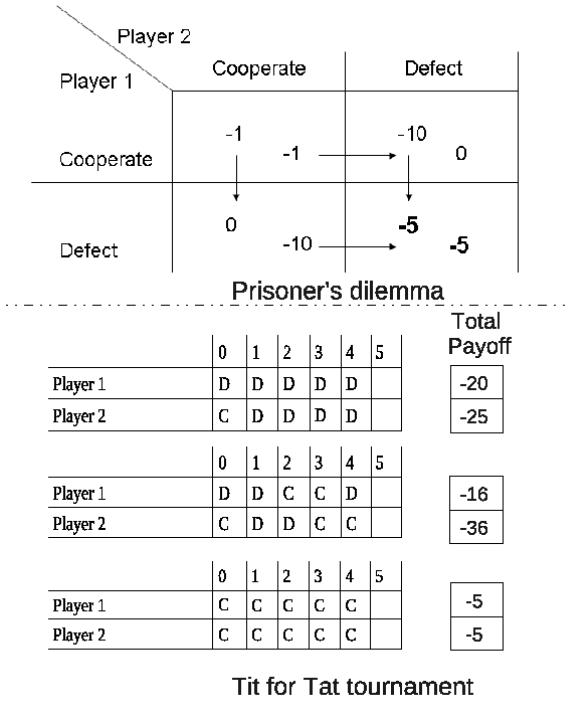


Figure 3. The tit-for-tat strategy in action

the first player decides to cooperate two times out of five (D,D,C,C,D), he would earn $(0,-5,-10,-1,0) = -16$. In case he always cooperates, his payoff would be $(-1,-1,-1,-1,-1) = -5$, which is the best he can achieve. So, continued cooperation for the iterated prisoner's dilemma also yields the best payoff. Despite this benefit, the main result of the tit for tat strategy is that it stimulates cooperation. We base our algorithm to mitigate the node selfishness on the results of this version of the game.

3. Tracking nodes behavior

In an ad hoc network, the number of nodes and links can change over time, so we consider 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.

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 acknowledgement messages containing the list of all the nodes $L_{(s,d)i}$ $0 < i < D$ traversed, as it happens in some source based routing protocols, the sender s is informed about the behavior of intermediate nodes. For each acknowledgement message received, the sender s can make a final update of the array $C(t)$ by setting the matching elements between the list $L_{(s,d)i}$ and list $A(t)_{(s,d)i}$ as *cooperative*. The mismatches are instead set to *defective* (Figure 4). Notice that the last update overwrites the previous stored values and represents the most recent information concerning the behavior of a node.

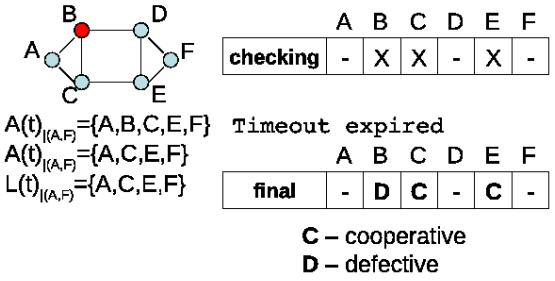


Figure 4. Algorithm description

At the same time, intermediate nodes (those not generating or receiving traffic but still involved in a path) can keep trace of other nodes' behaviors. As soon as a packet to be forwarded (and containing the complete routing list) is received, all the nodes on the path *preceding* the current node are marked as cooperative. Similarly, when an acknowledgement packet is received, all the nodes on the path *following* the current node are marked as cooperative. The remaining nodes in the path list are instead marked according to the ratio between the packets *relayed* and those *requested to relay* so far. If this ratio is above a fixed threshold (set for instance to 0.5), then the node is marked as cooperative, else it is marked as defective. Naturally, this ratio is neglected in case the node actively becomes a sender and starts the algorithm described above with the collection of acknowledgements $L_{(s,d)i}$.

Given this algorithm, 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. As soon as a cooperative path has been discovered, the following transmissions make use of it, thus limiting the multiple paths feature only to the search of new cooperative paths [5]. However, due to the highly dynamic evolution of ad hoc network topologies, the search for 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 acknowledgement 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 the possible pairs of nodes that try to interact, but all nodes store only an array $C(t)$ that is updated upon reception of any acknowledgement message, wherever it comes from, also including those intercepted from other traffic sessions. Furthermore, not all the packets relayed are checked in order to verify the nodes' behaviors, but only a fraction of them, so to keep the total overhead under control.

4. A protocol for the discovery of defective nodes

The algorithm introduced in the previous section has been implemented 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. In the next subsections we present the existing protocol, its basic features and the newly added ones.

4.1. Multipath source based routing in ad hoc networks

The dynamic configuration of an ad hoc network topology makes the routing protocols used in wired networks unsuitable for this kind of networks. Hence, several new protocols have been designed and made available to manage this collection of wireless nodes. To the purpose of identifying defecting nodes, an acknowledgement (or missed acknowledgement) technique is needed. Among the many ad hoc routing protocols, AH-CPN (Ad Hoc Cognitive Packet Network) [10] is designed to support QoS (Quality of Service) and make an intense use of acknowledgement messages independently from the transport protocol in use. AH-CPN is the wireless version of CPN (Cognitive Packet Network) [11], a proposal for a self aware network [24] architecture with native support for QoS. Both in AH-CPN and CPN, the presence of a neural network engine enables 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 discovers new paths according to specific QoS goals, e.g. 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 a source based routing basis. The smart traffic

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.

There are four different kinds of packets in AH-CPN, all sharing the same header: Smart Packets (SP), Smart Acknowledgements (SA), Dumb Packets (DP), and Dumb Acknowledgements (DA).

Smart packets are those described at the beginning of this section. They are lightweight packets containing a QoS goal sent by a sender to a destination. These packets are routed with the Random Neural Network (RNN) [12] algorithm that runs on each node and which selects the next hop by taking into account the past behavior of the link. Every time a SP traverses a node, a specific 'route map' (RM) field is updated with the node's address. Once at the destination, a SA is generated and sent backwards according to the RM received in the SP. Finally, the actual data can be sent across the network in a DP, which is prepared with the whole path copied in the RM field. Internal nodes relay DPs to the next hop excerpted from the RM field, and they add timestamp information useful to evaluate the round trip time (RTT) between each pair of nodes along the path. These RTT data are stored in special *mailboxes* present in each node and provide the RNN algorithm with precious information concerning the past behavior of a link. Once the DP reaches its destination, a DA is sent along the reverse path. Notice that differently from IP networks, in CPN the acknowledgements are generated upon reception of each single packet, whatever the transport protocol is. This feature is helpful in the deployment of our algorithm to identify defecting nodes, as we will soon explain.

The basic CPN version looks for one available path, the best in terms of the requested QoS goal. We modified this protocol to search for multiple paths. To this purpose, SPs are initially sent via flooding to collect all the available paths. To prevent loops, SPs are marked with an identification number, and those with the same such number touching a node for the second time are discarded. SPs reaching the same destinations with different contents for what concerns the routing map are considered valid, and SAs are sent backward to inform the sender. The sender collects the different SAs and updates its own routing table. DPs are sent on a round robin basis. Once the available paths are discovered, the transmission of SPs is not terminated; it is rather repeated periodically for path maintenance, to check if the topology has changed, and in our case also to verify if there is a different configuration concerning the behavior of nodes.

4.2. Identification and isolation of defecting nodes

We provide the multipath source based routing protocol with the capability to identify and isolate defecting nodes. The array $C(t)$ is computed and stored at each node. 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 by flooding. We make the assumption that non-cooperative nodes try to cheat by forwarding inexpensive SPs, which do not carry any payload, while they do not relay DPs containing the real data. In case the non-cooperative nodes decide to block the SPs forwarding, they are immediately discovered as non-cooperative and have no chance to cheat. In this scenario, every time a SP traverses a node, its cognitive map is extended with the label of the visited node. Once at the destination, the complete cognitive map is copied into the DA and sent back to the sender along the reverse path. Obviously, this is repeated for all the discovered paths. Thus, at the end of this process node a has a complete knowledge of all the available paths, including those comprising cheating nodes, and these are all stored in $A(t)_{(a,b)}$. At the time of the first transmission, the real data are packed in multiple DPs and sent along all the available paths on a round robin basis, but the interested cheating nodes will not relay them. Since in CPN a destination b must send an acknowledgement message DA whatever the transport protocol is, node a will receive only the DAs associated with the successful paths, i.e., those without cheating nodes. This information, as described before, helps finalize the array $C(t)$ with the list of cooperative and defecting nodes; traffic is sent only along the path or the paths composed of cooperative nodes rather than along all the available paths. When one of the cheating nodes requests the relaying of a message to node a, node a is aware of the past behavior of such nodes and can thus appropriately react, while it can regularly relay packets coming from cooperative users.

The situation concerning the detection of cooperation, as well as the selection of routing paths, is not static but can rather change over time. Then, 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 keep on checking periodically the paths containing the defecting nodes. Should a defecting node decide to change its behavior and begin to cooperate, the routing protocol soon detects this change and admits again the node to the transmission of flows. The current update time is set at 100ms.

5. Testbed and experiments

We tested the proposed routing protocol on a simulated testbed in the ns-2 simulator under different working

conditions. As already depicted in Figure 1, the testbed is composed of 25 nodes arranged in a $5x5$ grid topology.

At the end of each of the experiments run on the testbed, we measure the delivery ratio dr_i, the average residual energy of all the nodes μ, their variance ν, as well as the energy e_i spent by node i to successfully deliver one single byte to the destination, which is calculated as:

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

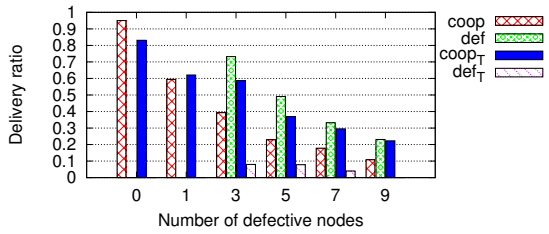
where Ec_i is the energy consumed by node i, s_i is the total number of bytes sent to the destination, rl_i is the number of bytes relayed from node i, and r_i is the bytes correctly received at destination. e_i has a dimension of $[Joule/bytes]$.

5.1. Dealing with unfair delivery ratio

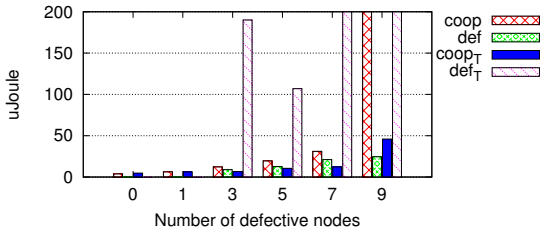
The first series of experiments aims at checking the algorithm responsiveness against sudden changes in nodes' behaviors. Ten sessions of UDP constant bit rate traffic are generated between the node pairs $(1,4)$, $(3,0)$, $(6,9)$, $(8,5)$, $(11,14)$, $(13,10)$, $(16,19)$, $(18,15)$, $(21,24)$ and $(23,20)$. The experiment lasts $300s$. Nodes $1, 8, 11, 18, 21$ start with a defecting behavior for the first $60s$. Then, they begin to switch between a cooperation status and a defection status in alternate time slots of $30s$ and $60s$, respectively. The remaining nodes are always cooperative.

In this first series of experiments we consider the presence of malicious nodes, which clearly have a negative impact on the final performance of cooperative nodes. As anticipated, in the 25 nodes testbed, nodes $1, 3, 6, 8, 12, 16, 18, 21, 23$ decide to either cooperate or defect for the entire duration of the experiment. The remaining nodes are instead always cooperative. We consider six cases: i) all nodes cooperate; ii) node 12 does not cooperate; iii) nodes $6, 12, 18$ do not cooperate; iv) nodes $6, 8, 12, 16, 18$ defect; v) nodes $1, 3, 6, 8, 12, 16, 18$ defect; vi) nodes $1, 3, 6, 8, 12, 16, 18, 21, 23$ do not cooperate. A total number of 10 CBR (Constant Bit Rate) UDP traffic sessions are generated between each pair of nodes $(0,24)$, $(24,0)$, $(4,20)$, $(20,4)$, $(6,9)$, $(8,5)$, $(13,10)$, $(11,14)$, $(16,19)$, and $(18,15)$. Each experiment lasts $720s$.

By looking at the first histogram in Figure 5(a), we can observe how the presence of malicious nodes affects the final performance. Also, in lack of a tracing algorithm, the delivery ratio of defective nodes (marked as *def*) outperforms that achieved by the cooperative ones (*coop*), whatever the number of defective nodes is. At the opposite, the introduction of the tracing algorithm always favors cooperative nodes ($coop_T$), whose delivery ratio is constantly kept higher than that of defective ones (def_T).



(a) Delivery ratio



(b) Energy per byte

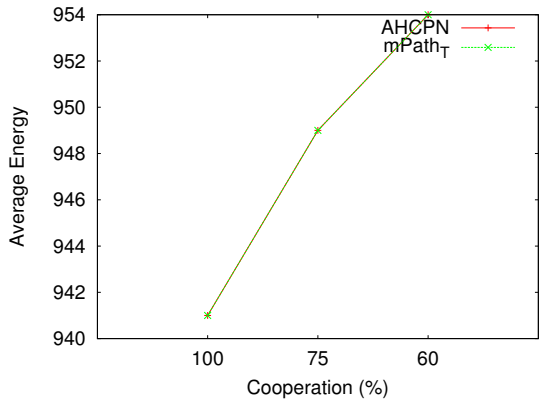
Figure 5. Fairing the delivery ratio

We then evaluate the energy e_i spent to successfully deliver a single byte to the destination in the second row of histograms in Figure5(b). Again, the behavior tracking algorithm *mPath-T* is able to reverse the values achieved with the plain version of the *AH-CPN* protocol. While in the basic protocol version the value e_i of cooperative nodes increases at a pace which closely mirrors the increase in the number of defecting nodes, when the tracing algorithm is enabled such a value is kept low as the number of defecting nodes increases. Also, the energy e_i of defecting nodes shows a raising exponential slope.

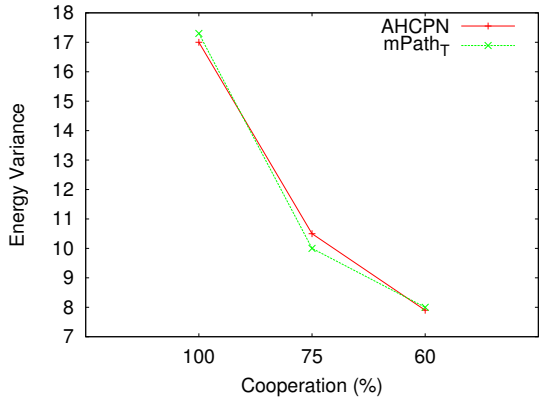
5.2. Dealing with unfair energy consumption

To balance the unfair energy consumption among the nodes we propose to deliberately push overwhelmed nodes in a defection mode for short time intervals. During this second series of experiments the percentage of cooperation of the inner nodes $6, 7, 8-11, 12, 13-16, 17, 18$ changes over time. In the first experiment, all nodes cooperate. In the second experiment nodes $6, 8, 12, 16, 18$ do not cooperate during the time interval $[340s, 520s]$, while nodes $7, 11, 13, 17$ do not cooperate during the time interval $[520s, 700s]$. These time intervals correspond to a percentage of 75% of nodes cooperation with respect to the overall duration of the experiment. In the third and last experiment, the defection intervals are extended, respectively, to $[100s, 400s]$ for the first group of nodes and to $[400s, 700s]$ for the second one. These time intervals correspond to a percentage of about 60% of

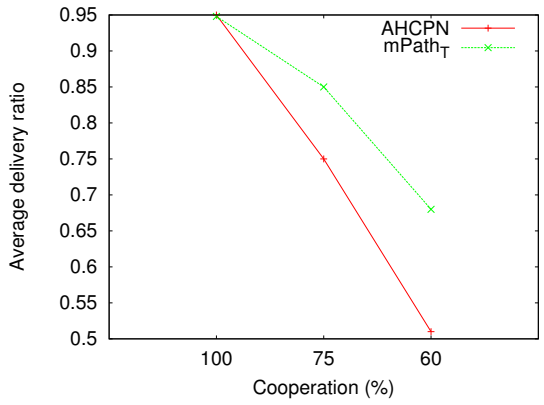
cooperation with respect to the overall duration of the experiment. Furthermore and differently from the previous series of experiments, defective nodes decide not to send traffic during the above mentioned intervals since their action aims at preserving the energy rather than cheating the other nodes.



(a) Energy per byte



(b) Energy variance



(c) Delivery ratio

Figure 6. Average results

Results are reported in Figure 6 (one graph for each of the metrics considered). Each graph reports the final values of a single performance indicator related to the three inner nodes' cooperation cases described above (total cooperation of 100%, 75%, and 60%). We compared the proposed algorithm with the basic AH-CPN version. They are marked in the following graphs as, respectively, *basic* (AH-CPN) and *mPath-T* (the version with the proposed tracking algorithm).

The first thing we can observe from the figures is how the average residual energy (Figure 6(a)) increases as the percentage of cooperation decreases. Also, the variance (Figure 6(b)) is significantly reduced when the defection interval of inner nodes increases, which means a fairer distribution of the energy consumed at each node. These results were quite expected and do not show significant improvements deriving from the introduction of the proposed tracking algorithm. Notice however that if we look at the comparison of delivery ratio achieved by the two versions of the protocol in Figure 6(c), the behavior tracking algorithm clearly outperforms the counterpart when the percentage of cooperation of inner nodes goes down. The basic protocol is in fact unable to detect that certain nodes (and hence certain paths) have become unavailable. Thus, it does not try to discover any alternative paths. The *mPath-T* protocol is instead capable to promptly detect both nodes and paths unavailability, and thus react by discovering alternative paths allowing to still reach the destination.

5.3. Network lifetime

We conducted a further experiment to analyze network nodes lifetime. To this purpose we make the testbed start from a situation of low energy and we count the number of nodes that shut down before the natural end of the experiment. As witnessed by the comparative results presented in Figure 7, the introduction of the behavior tracking system again helps reduce the number of node shutdowns as long as the initial energy level is higher than $54J$. Notice also how the slope of the basic protocol is linear, while that of the tracing algorithm looks like an impulse, which again indicates a fairer distribution of residual energy among nodes and then a longer lifetime for the whole network.

5.4. Discussion

Many applications of ad hoc networks are conditioned by the limited energy provided by accumulators and then rely on an implicit cooperation agreement among the nodes. To address some of these issues we propose to introduce a dedicated mechanism in ad hoc routing protocols. Such a mechanism provides ad hoc network nodes with the capability to react to unfair working conditions, in terms of both delivery ratio and energy consumption. Our experimental results show that we can in such a way detect uncooperative nodes, keep

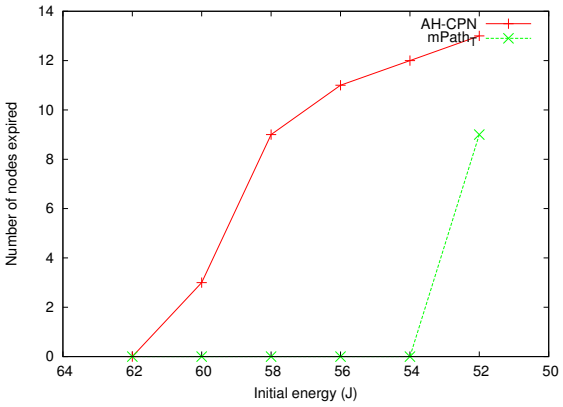


Figure 7. Network lifetime evaluation

the delivery ratio of cooperative nodes both high and favorable to them, and provide a prolonged network lifetime. The introduction of a tracking algorithm certainly introduces additional overhead. The overhead due to the generation of signaling packets to serve the CPN protocol is limited to 5% out of the total traffic [13]. The multiple path functionality is limited to the search for available paths, as the real traffic flows along a single path composed of cooperative nodes. This search is renewed periodically, and can be triggered according to the size of the network. Finally, the evaluation of nodes behavior is fully distributed and extrapolated from the actual traffic, so additional signaling traffic is not needed. Although we do not have a detailed evaluation of this load, we consider the limited energy consumption in relation with the preserved delivery ratio measured during our experiments an indirect estimation of the reasonable overhead introduced by the tracking algorithm. We indeed believe that the introduction of a tracking algorithm can be beneficial to all those battery constrained ad hoc network applications like, for example, sensor networks for environmental monitoring, vehicular networks and health care monitoring networks.

6. Related work

Most of the proposals to mitigate the unfair consumption of energy rely on energy aware routing protocols. The authors of [14], [15] propose a power-based routing metric rather than a distance metric. In the former, a min-max approach selects the shortest path if all nodes in all possible routes have sufficient battery capacity. In the latter, an extension to an existing protocol allows some nodes to intelligently switch off when they are not actively transmitting or receiving packets, thus yielding a reduction in battery consumption. The presence of a separate control channel supports the decision upon the sleeping time slots, as well as their duration. However, this last proposal lacks an acknowledgement mechanism and therefore cannot recover from data collisions. A substantial enhancement is then presented in [16], which introduces a new MAC protocol with control signals placed into control frame slots having the capability to recover from data collisions. These modifications provide a higher data throughput and a lower energy consumption in a truly mobile environment. Similarly, [17] is based on the observation that when a wireless network has a sufficient density of nodes, only a small number of them are involved in forwarding operations, so most of these nodes can be placed in a sleep mode. When nodes are awaken, they join a forwarding backbone acting as coordinator. Local decisions set the node status on the basis of both the amount of energy still available and the benefit given to the neighbors. The authors of [18] also embrace an approach based on network topology. They propose to check energy consumption in each discovered route and further monitor its updated level in subsequent maintenance cycles.

Rather than considering an adaptation at MAC or network layer, the work in [19] formulates the fair energy distribution problem with the same objective functions as our proposal: minimizing the residual energy variance at the same time maximizing the minimum residual node energy. To this purpose, [19] suggests to partition the connections in segments: when nodes are close enough to be reached in a single hop, part of the connection's data are sent by direct transmission, whereas the remainder is sent along regular network routes. A further solution is reported in [20], which proposes an organization in clusters of wireless nodes so that the load of the network is balanced and energy dissipation is reduced.

To address the fair energy consumption issue, in [21] a fair cooperative protocol (FAP) is proposed to improve the overall performance of the whole network. Each node calculates a *power reward* to evaluate the power contributed to and by the others. The adoption of such a fair scheme can also lead to substantial throughput gains over both the direct transmission and the full cooperation case. There exists however a trade off between the fairness and the throughput achieved. An analytical study helps in the selection of the working point associated with the best trade off between fairness and throughput.

The work done in [22] also relies on cooperation among nodes. However, it makes the assumption that only a subset of the nodes belonging to the same group can be interested in mutual cooperation. This hypothesis replaces the one which assumes full cooperation among all nodes. If nodes belonging to a group form a *coalition* and route packets without intervention of nodes outside the coalition, then the routing shows some benefits with respect to the

fully selfish case. In [23], an alternative game theory approach to increase efficiency in sensor networks is presented. Following a duopoly Stackelberg game, sensor nodes are modeled in a hierarchical tree with parents, acting as leaders, and children, acting as followers. Parents cooperate with other parents, at the opposite of children that behave selfishly. This approach yields a better distribution of the overall energy. Although these solutions prove effective, we prefer the fully distributed approach (with neither groups nor leaders) in which each of the nodes takes on responsibility for its own behavior.

7. Conclusions

In ad hoc networks each node, besides sending and receiving its own traffic, is also requested to serve the other nodes. Current ad hoc routing protocols do not take into account the amount of work done to relay other nodes' traffic, which impacts the residual energy available for a node's own transmissions. This inevitably brings to a situation of unfair energy consumption for certain nodes in favor of other, less involved, ones. In this paper we proposed to introduce some modifications in ad hoc routing protocols to support the identification of nodes' behaviors. Behavioral information gives the busiest nodes a chance to temporarily stop serving the others, while the routing protocol helps discover alternative paths allowing to keep both the residual energy and the overall network performance at fair levels. The improved distribution of energy consumption also prevents the irregular shut down of overloaded nodes, thus increasing the overall network lifetime.

References

[1] G. Anastasi, M. Conti, M. Di Francesco, A. Passarella Energy conservation in wireless sensor networks: A survey *Ad Hoc Networks* Volume 7, Issue 3, May 2009, Pages 537-568, ISSN 1570-8705, http://dx.doi.org/10.1016/j.adhoc.2008.06.003.

[2] J. Zhu, C. Qiao, X. Wang On Accurate Energy Consumption Models for Wireless Ad Hoc Networks *Wireless Communications, IEEE Transactions on*, vol.5, no.11, pp.3077,3086, November 2006 doi: 10.1109/TWC.2006.04566

[3] D. Feng, C. Jiang, G. Lim, Cimini L.J. Jr., G. Feng, Li G.Y A survey of energy-efficient wireless communications In *Communications Surveys and Tutorials, IEEE*, vol. 15, no.1, pp.167-178, First Quarter 2013 doi: 10.1109/SURV.2012.020212.00049

[4] V. Srinivasan, P. Nuggehalli, C. F. Chiasserini, and R. R. Rao. Cooperation in wireless ad hoc networks. In *INFOCOM 2003.*, vol. 2, pp. 808–817, April 2003.

[5] M. D'Arienzo, F. Oliviero, and S.P. Romano. Smoothing selfishness by isolating non-cooperative nodes in ad hoc wireless networks. In *Proceedings of the 2010 Second International Conference on Advances in Future Internet*, AFIN '10, pages 11–16, Washington, DC, USA, 2010. IEEE Computer Society.

[6] J. Nash. Non-Cooperative Games. *The Annals of Mathematics.*, 54(2), pp. 286–295, 1951.

[7] J. F. Nash. Equilibrium Points in n-Person Games. In *Proceedings of the National Academy of Sciences of the United States fo America.*, 36(1), pp. 48–49, 1950.

[8] R. Axelrod. *The Evolution of Cooperation*. Basic Books, 1988.

[9] R. Axelrod and D. Dion. The further evolution of cooperation. *Science*, 242(4884), pp. 1385–1390, December 1988.

[10] E. Gelenbe and R. Lent. Power-aware ad hoc cognitive packet networks. *Ad Hoc Networks*, 2(3):205–216, July 2004 (ISN: 1570-8705).

[11] E. Gelenbe, R. Lent, and Z. Xu. Design and performance of cognitive packet networks. *Perform. Eval.*, 46(2-3):155–176, 2001.

[12] E. Gelenbe. Learning in the recurrent random neural network. *Neural Comput.*, 5(1):154–164, 1993.

[13] Gellman, Michael and Liu, Peixiang Random Neural Networks for the Adaptive Control of Packet Networks Artificial Neural Networks, 4131:313–320, ICANN 2006

[14] C.-K. Toh. Maximum battery life routing to support ubiquitous mobile computing in wireless ad hoc networks. *Communications Magazine, IEEE*, 39(6):138 – 147, June 2001.

[15] Suresh Singh and C. S. Raghavendra. Pamas: power aware multi-access protocol with signalling for ad hoc networks. *SIGCOMM Comput. Commun. Rev.*, 28:5–26, July 1998.

[16] S. Mehta, Tamanna, M. Kumar. A Comparative Study of Power Aware Routing Protocols of Ad Hoc Network IJCA Proceedings on National Workshop-Cum-Conference on Recent Trends in Mathematics and Computing 2011 RTMC(15):3-7, May 2012. Published by Foundation of Computer Science, New York, USA.

[17] B. Chen, K. Jamieson, H. Balakrishnan, and R. Morris. Span: An energy-efficient coordination algorithm for topology maintenance in ad hoc wireless networks. *Wireless Networks*, 8(5):481–494, 2002. cited By (since 1996) 385.

[18] Asha, T. S.; Muniraj, N. J. R. Energy efficient topology control approach for mobile ad hoc network *International Journal of Computer Science Issues (IJCSI)*, Vol. 10 Issue 4, p289, Jul2013.

[19] N.K. Singh, R. Simha, and B. Narahari. Energy balance in wireless networks using connection segmentation and range control. In *Wireless Communications and Networking, 2003. WCNC 2003. 2003 IEEE*, volume 3, pages 1871 –1876 vol.3, march 2003.

[20] Wen-Wen Huang, Ya-Li Peng, Jian Wen, and Min Yu Energy-Efficient Multi-hop Hierarchical Routing Protocol for Wireless Sensor Networks In Proceedings of the 2009 International Conference on Networks Security, Wireless Communications and Trusted Computing (NSWCTC '09), Vol. 2. IEEE Computer Society, http://dx.doi.org/10.1109/NSWCTC.2009.352

[21] Lin Dai, Wei Chen, L.J. Cimini, and K.B. Letaief. Fairness improves throughput in energy-constrained

cooperative ad-hoc networks. *Wireless Communications, IEEE Transactions on*, 8(7):3679 –3691, july 2009.

[22] R.K. Guha, C.A. Gunter, and S. Sarkar. Fair coalitions for power-aware routing in wireless networks. *Mobile Computing, IEEE Transactions on*, 6(2):206 –220, feb. 2007.

[23] Behzadan, Afshin and Anpalagan, Alagan and Woungang, Isaac and Ma, Bobby and Chao, Han-Chieh An energy-efficient utility-based distributed data routing scheme for heterogenous sensor networks *Wireless Communications and Mobile Computing*, 2014

[24] E. Gelenbe "Steps toward self-aware networks", *Communications ACM*, 52(7):66-75, July 2009.