Wireless Pers Commun
DOI 10.1007/s11277-010-0121-6

A Cross-Layer Multi-Hop Simulator for IEEE 802.11e

João M. Ferro · Orlando Cabral · Fernando J. Velez

Abstract In this work we simulate the ad hoc mode of IEEE 802.11e for routing optimisation. We simulate the behaviour of routing algorithms at the network layer by using a custom-made cross-layer network simulator developed by our team, which simultaneously considers the physical and Medium Access Control (MAC) layers. Although the simulator also supports the infrastructure mode, in this paper we focus on the ad hoc feature which was introduced by the authors. We opted for the simulator approach over the theoretical analysis, but we also present a mathematical model for IEEE 802.11 ad hoc networks. Some initial tests were performed by using a simple routing algorithm (to evaluate the behaviour of the system in terms of selection of the path between a source and a destination, and the correctness of the calculated metrics, which include end-to-end delay, packets lost, packets delivered), but more advanced cross-layer design solutions were also tested. When information from the physical and MAC layers is used as an input to the routing algorithm, improvements are achieved in the performance of the network. Several functions were compared and the algorithm that privileges shorter links accounting with the metric "collision rate" achieves the best results. When compared with a standard routing solution, this cross-layer approach allows to increase the number of packets delivered, while not significantly affecting the end-to-end delay of the packets.

Keywords Cross-layer · IEEE 802.11e · Routing · Ad hoc

J. M. Ferro (✉) · O. Cabral · F. J. Velez
Instituto de Telecomunicações-Departamento de Engenharia Electromecânica, Faculdade de Engenharia, Universidade da Beira Interior, Calçada Fonte do Lameiro, 6201-001 Covilhã, Portugal
e-mail: ferro@lx.it.pt

O. Cabral
e-mail: orlandoc@ubi.pt

F. J. Velez
Centre for Telecommunications Research, King's College London, Strand, London, WC2R 2LS, UK
e-mail: fjv@ubi.pt

Published online: 03 September 2010

1 Introduction

The IEEE 802.11 is a family of protocols designed to standardise the transmission of packets in a Wireless Local Area Network (WLAN). It provides regulation for the Medium Access Control (MAC) and Physical (PHY) layers, only describing how the direct communication is made between two stations. The IEEE 802.11e is an amendment to the original standard, and is built for the support of Quality-of-Service (QoS). Since the original IEEE 802.11 standard equally treats all the packets, time sensitive traffic—such as video and voice—could not arrive on time at destination. The improvement brought by IEEE 802.11e is the classification of the packets in four different classes—video (VI), voice (VO), best effort (BE) and background (BK), and the assignment of different priorities to each class or access category (AC). This is accomplished at the MAC layer, where a new function is introduced.

The IEEE 802.11 defines two types of networks, one called Basic Service Set (BSS), in which there is an Access Point (AP) that acts like the coordinator of the network, all stations in the network are controlled by it and can communicate only with the AP; and another, called Independent Basic Service Set (IBSS), in which there is no AP and all the stations can communicate among them, as long as they are within the range of their radio device. In this work we consider only the latter, which can also be called an ad hoc network.

Packet forwarding is a very important aspect in an ad hoc network. Stations that generate the packets must know to where they should forward their data, and they do this by consulting their routing table. Each routing table contains an entry for all the members of the network. Associated with that entry is a station that is called the "next hop". When a station receives a packet that has another station as the destination, it simply checks who the next hop is and sends the packet to it. Hence, it is very important to have a routing table as much accurate as possible, in order to send the packet through the best path possible. Parameters that can be considered for this purpose are the end-to-end delay (period of time that passed since the packet was generated until it was successfully received at the destination), throughput (rate at which packets are delivered, in bit/s) and the number of packets delivered or lost (a path where many packets are lost is not a good one). Since IEEE 802.11 only regulates the MAC and PHY layers, the routing principles are not covered by the standard. Several proposals have been made in this field. For example, the authors from [13] have demonstrated that a cross-layer approach can reach better results than a standard one. Reference [4] presents the Expected Transmission Count (ETX) metric, which attributes to every link a cost related to the packet delivery ratio. Derived from this one, the Expected Transmission Time (ETT) was presented by [6] and is a function of the loss rate, the bandwidth of the link, and the size of the packet to be transmitted. All these proposals have been proved to improve the performance of the routing protocols to which they are applied to. In our work, we will use a different cross-layer approach looking for an optimal network performance.

The remainder of this paper is organised as follows. Section 2 presents a small overview of the standard IEEE 802.11e and its implementation on our simulator. Section 3 describes an analytical model for an IEEE 802.11 ad hoc network. Section 4 describes the routing techniques based on cross-layer design, used in this paper. Section 5 presents the results of our simulations, obtained for different cost functions. The discussion of the improvement is based on a comparison with a basic routing algorithm, which does not consider cross-layer design at all. Finally, in Section 6 we make the final considerations about the work, presenting the conclusions and proposing suggestions for further work.

2 IEEE 802.11e Simulator

In order to address the performance of IEEE 802.11e multi-hop ad hoc networks, two different approaches may be taken. One is the analytical modelling of the problem, and the other is event-driven simulation. Analytical frameworks to model a multi-hop network have been previously proposed by [2,7] and [15], but the results are neither based on a realistic approach to the problem, nor account for the achievable QoS that IEEE 802.11e may provide. As the EDCA features of IEEE 802.11e are not addressed, the use of several service classes with different priorities (the base for EDCA) is not considered at all. The existing theoretical approaches previously referred do not consider the hidden terminal problem and do not assume a multi-rate scenario and non ideal physical channel. [7] and [15] assume that the buffer is always full. The hidden/exposed terminal problem is one fundamental issue in WLANs, but most of the existing analytical models assume it does not exist. The multi-rate feature in the same environment was also not considered. Furthermore, the packet loss due to errors in the channel is not taken in consideration. Secondly, we can use a simulation tool more suitable for investigating the dynamic behaviour (for example the end-to-end delay). Although an initial model is proposed here, cross-layer results are obtained via simulation.

The original IEEE 802.11 uses a Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA) methodology, denominated Distribution Coordination Function (DCF), to coordinate the access of the stations to the shared medium. This amendment introduces a new function called Hybrid Coordination Function (HCF), which uses the Enhanced Distributed Channel Access (EDCA) and the HCF Controlled Channel Access (HCCA). The EDCA provides a contention-based access to the channel, while HCCA provides access to the channel in a contention-free manner. In both methods each station tries to gain a Transmission Opportunity (TXOP) which will allow it to transmit a packet. The EDCA provides differentiated, distributed access to the medium for Quality Stations (terminals that support IEEE 802.11e) using four access categories (ACs): voice (VO), video (VI), best effort (BE), and background (BK). This differentiation is achieved by mapping the traffic to four queues that correspond to the four ACs. The traffic prioritisation is performed by varying: the amount of time a station queue senses the channel to be idle before backoff or transmission; the length of the contention window to be used for the backoff; the duration a station queue may transmit after it acquires the channel. Each AC contends to access the medium with a single CSMA instance [9,10]. Each queue has an Arbitration Inter-frame Spacing (AIFS) period preceding the next backoff contention window, Fig. 1.

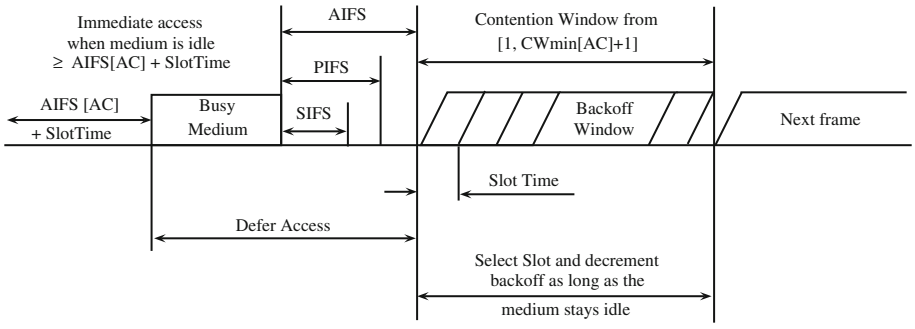

Fig. 1 Timing relationship in EDCA

The tool used to extract the results is a discrete event simulator that simulates IEEE 802.11a [9] standard with the enhancement that IEEE 802.11e [10] adds to the IEEE 802.11 networks.

The stations/machines go through all the possible states that an active machine in a Wi-Fi network may go through. The following states were considered:

- *Idle*—the machine has nothing to transmit;
- *Waiting an AIFS*—the machine is contending during an AIFS;
- *TX*—the machine is transmitting;
- *RX*—the machine is receiving;
- *Waiting an Inter-Frame Space* (IFS)—the machine is contending during an IFS;
- *Waiting an acknowledgement* (ACK)—the machine is waiting for an acknowledgement;
- *Backoff timer*—the machine is decreasing its backoff counter;
- *TX ACK*—the machine is sending an acknowledgement;
- *Frozen*—the machine has frozen its backoff counter because someone is transmitting.

A simple case that may happen is the following:

1. The machines start in the *idle* state when they do not have anything to transmit;
2. If the traffic generator causes the packets to arrive into the MAC buffer, it changes the state to *Waiting an AIFS*;
3. By the end of the AIFS time, if no machine has tried to transmit, the machine goes into *TX* state;
4. The receiving machine goes into *RX* state;
5. When the transmission time ends, the transmitting and receiving machines go into *Waiting an IFS* state;
6. By the end of the IFS time, the transmitting machine changes its state to *Waiting an ACK*;
7. By the end of the IFS time, if the receiving machine has received the packets correctly, it goes into *TX ACK* state;
8. When the transmission of the acknowledgement has finished, both the transmitting and receiving machines, either go into *Waiting an AIFS* state if there is a packet in the buffer, or go into *idle* state if there are no packets in the buffer.

Many other cases may happen, when collisions occur, when the request to send/clear to send (RTS/CTS) procedure is used, etc. The following events cause transition/change of the machine state:

- NEW_PCK_BK—a new packet of BK is generated;
- NEW_PCK_BE—a new packet of BE is generated;
- NEW_PCK_VI—a new packet of VI is generated;
- NEW_PCK_VO—a new packet of VO is generated;
- STOP_LTN_DIFS—end of the AIFS period for sensing the medium;
- STOP_LTN_SIFS—end of the SIFS period for sensing the medium ;
- TIME_SLOT—the station decrements the backoff value;
- START_TX—the station starts to transmit;
- STOP_TX—end of the transmission;
- START_RX—the station starts receiving;
- STOP_RX—end of the reception;
- ACK_OK—the ACK was received;
- ACK_NOK—the ACK was not received.

The physical layer specification used in this work is the IEEE 802.11a standard that defines an Orthogonal Frequency Division Multiplexing (OFDM) based PHY layer operating in the

5 GHz frequency band, being able to achieve bit-rates as high as 54 Mbit/s. IEEE 802.11a specifies eight different transmission modes, obtained with different combinations of modulation and coding schemes. Each transmission mode corresponds to a different bit-rate.

Our simulation tool considers an event-driven engine that was developed in C++, and is able to run in Windows or Linux environments without any change to the source code (only a compiler is required). This custom-made approach allows to flexibly alter it at our own will. The output is built per application, meaning all the variables are accessible globally and it is easy to select the ones desired for the output. It does not have any mobility model implemented and does not include any visual interface yet. Hence, the user cannot see the simulation nor interact with it, and it has only access to the desired output at the end of the simulations. It was shown that simulations running in Linux were faster than the ones running on a similar machine with Windows.

A more comprehensive description of the simulation variables, possible states for the machines, the simulation entities, the functions for events can be found in [3], as well as more details about the simulator.

3 Analytical Model

3.1 Notation and Formulation

An analytical model for IEEE 802.11 networks was developed in [11] for non-saturated conditions. Several types of traffic were simultaneously implemented in this model. Here, we upgrade this work to model an ad hoc network. The model presented in [11] characterises the behaviour for each station. The following probabilities were used in our model:

- q—the probability that at least one packet is waiting transmission at the start of a counter decrement;
- p—the probability of collision given that the station is attempting transmission;
- b—the Markov chain stationary distribution;
- τ—the stationary distribution's probability that the station transmits in a slot;
- P_{idle}—the probability that the next slot is sensed idle.

P_{idle}, τ, and b depend on p and q.

$$
\begin{aligned}
b(p,q) = (1-q) &+ \frac{q^2 W_0(W_0+1)}{2(1-(1-q)^{W_0})} + \frac{q(W_0+1)}{2(1-q)} \\
&\cdot \left(\frac{q^2 W_0}{1-(1-q)^{W_0}} + (1-P_{idle})(1-q) - q P_{idle}(1-p) \right) \\
&+ \frac{pq^2}{2(1-q)(1-p)} \cdot \left(\frac{W_0}{1-(1-q)^{W_0}} - (1-p)P_{idle} \right) \\
&\cdot \left(2W_0 \frac{1-p-p(2p)^{m-1}}{1-2p} + 1 \right)
\end{aligned} \tag{1}
$$

where

$$
\tau(p,q) = \frac{1}{b(p,q)} \left(\frac{q^2 W_0}{(1-p)(1-q)(1-(1-q)^{W_0})} - \frac{q^2 P_{idle}}{1-q} \right) \tag{2}
$$

W_0 is the minimum contention window, and P_{idle} will be explained ahead.

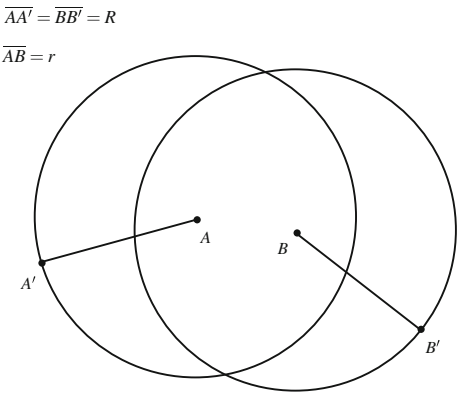

Fig. 2 One hop transmission example

Equations (1) and (2) model the behaviour of a terminal in a Wi-Fi network. To model an ad hoc network more concepts and assumptions have to be introduced. We assume that stations are distributed using a two-dimensional Poisson point process with density λ, which means that the probability of finding i terminals in an area A is given by:

$$P(i|A) = \frac{(\lambda A)^i e^{-\lambda A}}{i!}. \tag{3}$$

Figure 2 presents an example of a transmission (Tx) between station A and B. By considering the same Tx power for each station, and the same scenario regarding propagation, the hearing/Tx range of each station is the same, therefore $\overline{AA'} = \overline{BB'} = R$, where R is the hearing/Tx range. Some notations that will be needed ahead are defined as follows:

- $N(A)$ is the hearing area of station A;
- $N = \lambda\pi R^2$ is the number of stations within the hearing zone of a given station Tx;
- r is the distance between A and B;
- $U(r) = N(B) - N(A)$ is the non-overlapping area, which is the hearing area for one station but not for the other, i.e. B can listen but not A, which is equal to the one that A can listen but not B (which is easily proven since the areas are the same);
- $C(r) = N(B) \cap N(A)$ is the common area that A and B can listen.

3.2 Network Model

It is assumed that the entire network uses the same channel. Therefore, the actions of the neighbouring terminals depend on their neighbours. We assume that there are n terminals in the scenario, then the probability of no collision in a given area is defined by

$$1 - p = (1 - \tau)^{N-1}, \tag{4}$$

where $N = \lambda\pi R^2$. It means that there is no collision since at the most there is only one station transmitting. We need to know how much time is spent on each of the states to, for example, calculate the throughput. If the average time spent in a successful transmission is determined, the throughput may be obtained. The medium in a given area, as our Markov chain, can spend time in an idle state, a successful transmission, or a collision. Equation 5 establishes the average time spent in these three states.

$$E_s = (1 - P_{Tx})\sigma + \sum_{i=1}^{n} P_{s_i} T_s + \sum_{j=2}^{n} C_j^n P_{c_j} T_c \tag{5}$$

where

$$P_{s_i} = \tau \cdot (1 - \tau)^{|C(r)|} \cdot (1 - \tau)^{|U(r)|} \tag{6}$$

is the probability for station i to successfully transmit, T_s is the expected time taken for a successful transmission, and T_c is the expected time taken for a collision. Besides

$$P_{c_j} = \prod_{i=1}^{j} \tau (1 - \tau)^{j-i} \cdot \frac{(\lambda \pi R^2)^i e^{-\lambda \pi R^2}}{i!} \tag{7}$$

gives the probability for all the j stations within a given area to have a collision, σ is the slot time, and $|X|$ is the number of elements of X. The probability that at least one station attempts to transmit is given by

$$P_{Tx} = 1 - \prod_{i=1}^{N} (1 - \tau). \tag{8}$$

After the average duration of each state is found, the throughput can be determined by estimating the amount of time a station is successfully transmitting data

$$S_i = \frac{P_{s_i} L_i}{E_s} \tag{9}$$

where L_i is the expected time spent transmitting data from source i. The normalised throughput of the system is then $S = \sum_{i=1}^{n} S_i$.

P_{idle} is the probability for the channel to be idle, when no station is transmitting, and is given by

$$P_{idle} = (1 - \tau)^N = 1 - p. \tag{10}$$

The offered load is represented by q, the probability that a packet becomes ready to be transmitted. As the traffic for Wi-Fi in [12] has a constant distribution, we consider that packet arrivals are uniformly distributed across slots and $q = \min(E_s / inter\ arrival\ time, 1)$.

The validation of this model is left for future work.

4 Routing Techniques

The routing calculation is performed by running Dijkstra's algorithm [5] in a table containing the cost of the links between all the stations in the network. Dijkstra's algorithm is a shortest path finding algorithm widely used in routing calculations. For a certain node, it computes the path to all the remaining nodes of the network with the global lower cost. Thus, by assigning different costs to the links between the stations, it originates different paths. Since Dijkstra's algorithm has been proven to work well in a centralised routing system [14], which is the case, we will only optimise the link costs and leave the routing calculation unchanged for all our simulations. In our simulator, once the search for the least cost path is finished, the algorithm generates a table in which for each pair source/destination there is the indication of the next hop.

Table 1 Maximum throughput for each transmission mode

Mode	Modulation	Code rate	Min. SINR	Link throughput [Mbit/s]
1	BPSK	1/2	4.1	6
2	BPSK	3/4	–	9
3	QPSK	1/2	7.9	12
4	QPSK	3/4	11.0	18
5	16-QAM	1/2	14.8	24
6	16-QAM	3/4	17.8	36
7	64-QAM	2/3	22.8	48
8	64-QAM	3/4	24.2	54

Table 2 Cost calculation formula for the first tests

Alg.	Link cost
A	$\sqrt{(20 \times 10^6 - link\_throughput)^2}$
B	$\sqrt{(21 \times 10^6 - link\_throughput)^2}$
C	$\sqrt{(30 \times 10^6 - link\_throughput)^2}$
D	$\sqrt{(56 \times 10^6 - link\_throughput)^2}$

In the first simulations we assume that the cost of the link between the stations that are able to directly communicate is 1, otherwise the cost is set to infinity. Later, we use information gathered from the physical layer to compute the cost of each link, and we change the formula to calculate the cost in order to achieve the best performance possible.

Thus, all the results identified as "no cross-layer" (*No*) were calculated by using the following formula:

$$link\_cost = 1 \quad \text{(if stations are in range)}. \tag{11}$$

Regarding the tests for the cross-layer approach, we determine the signal-to-interference-plus-noise ratio (SINR) sensed at the sender, and use it to determine the link throughput according to Table 1, adapted from [8].

For the next set of tests, we use the information on the link throughput to compute the paths according to the formulas from Table 2.

Since with the algorithm *No* our intention is to minimise the number of hops (notice that since all link costs is 1, Dijkstra's algorithm produces paths with a minimal number of hops), with these new metrics we are looking for a compromise between the total number of hops and their quality. With the *D* algorithm, it is our intention to force the network to use links with the highest throughput to route the traffic (the cost will decrease with the increase of the throughput). This may increase the number of hops, but we expect to reduce the number of retransmissions and lost packets since the stations are closer to each other and communicate with higher SINR. Metrics *A*, *B* and *C* situate between the *No* and *D* ones, as they attribute a lower cost to paths that have mean link throughput.

Up to this point, the tests were very static since the routing calculation was fully performed prior to the simulation start. The simulator would deploy the stations in the study field and, by

Table 3 Cost calculation formula for our second proposal

Alg.	Link cost
A'	$\sqrt{(20 \times 10^6 - link\_throughput)^2} + (1 \times 10^5 \times collisions)$
B'	$\sqrt{(21 \times 10^6 - link\_throughput)^2} + (1 \times 10^5 \times collisions)$
C'	$\sqrt{(30 \times 10^6 - link\_throughput)^2} + (1 \times 10^5 \times collisions)$
D'	$\sqrt{(56 \times 10^6 - link\_throughput)^2} + (1 \times 10^5 \times collisions)$

Table 4 Cost calculation formula for our third proposal

Alg.	Link cost
A''	$\sqrt{(20 \times 10^6 - link\_throughput)^2} \times (1 + collision\_rate)$
B''	$\sqrt{(21 \times 10^6 - link\_throughput)^2} \times (1 + collision\_rate)$
C''	$\sqrt{(30 \times 10^6 - link\_throughput)^2} \times (1 + collision\_rate)$
D''	$\sqrt{(56 \times 10^6 - link\_throughput)^2} \times (1 + collision\_rate)$

taking readings of the SINR from every station (at the physical layer), it would run Dijkstra's algorithm for each station. After that, the simulation started and the routing table was kept unchanged. This could lead to situations in which a path was being used exhaustively, overloading the stations in it, while an alternative path could exist and be available, but not being used because it was not the shortest one.

To avoid this situation, and trying to achieve better performance, we update the routing table by looping back the number of collisions that occur at a certain station (for each link we use the number of collisions that have occurred at the receiver side). The intention behind the formulas presented in Table 3 is, firstly, to choose the shortest path. Then, if the stations on the shortest path are getting congested, the cost of the associated links increases up to a point that another path will arise as the best one.

We also use the metric collision rate, which is computed as the ratio between the number of collisions and the number of packets successfully transmitted by a station. The formulas for this approach are shown in Table 4.

5 Results

5.1 Simulator Test

To test the simulator we first tried to verify how the presence of multiple video streams would affect the end-to-end delay and packet losses. For this test, and the subsequent ones, we used the stations deployment shown in Fig. 3.

Each one of the dashed squares has an area of $30 \times 30\,m^2$, the entire field has $120 \times 120\,m^2$. In the first set of tests, we added sequentially video streams as follows:

I: From station 1 to station 3;
II: From station 4 to station 2;
III: From station 5 to station 6;
IV: From station 8 to station 9.

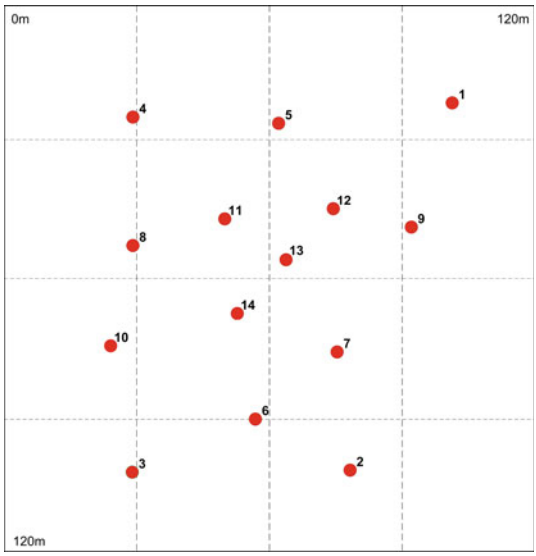

Fig. 3 Stations deployment for the tests

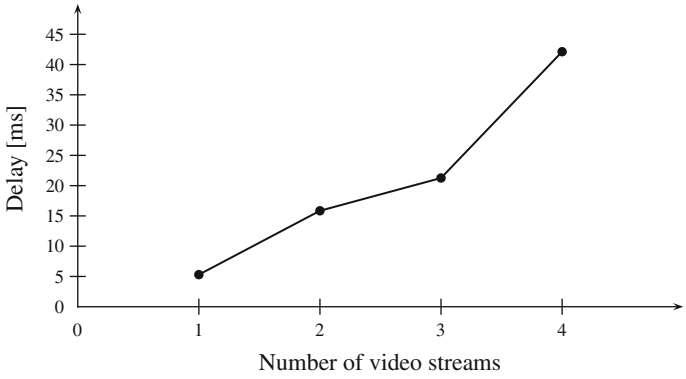

Fig. 4 End-to-end delay for stream I

For each of these streams, the routing algorithm has selected the following paths:

I: $1 \rightarrow 9 \rightarrow 13 \rightarrow 14 \rightarrow 10 \rightarrow 3$;
II: $4 \rightarrow 11 \rightarrow 14 \rightarrow 7 \rightarrow 2$;
III: $5 \rightarrow 13 \rightarrow 14 \rightarrow 6$;
IV: $8 \rightarrow 14 \rightarrow 13 \rightarrow 9$.

Figure 4 presents the end-to-end delay for the video packets flowing from station 1 to station 3 (stream I). Like one should expect, by increasing the number of stations that generate packets (i.e., adding stream II, then stream III, and then stream IV), the end-to-end delay is increased.

Another interesting metric is the number of lost packets, which is presented in Fig. 5. When the number of packets to be transmitted increases, there are more collisions and packet losses. This is due to the fact that more packets are being pushed onto the shared medium at the same time.

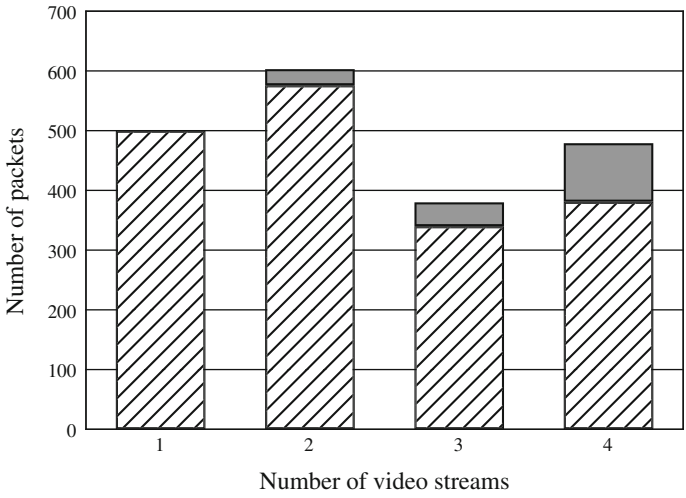

Fig. 5 Total of packets received (*hatched bar*) and lost (*gray bar*)

Table 5 Characterisation of each traffic type

Type	Packet size [bits]	Generation Period [ms]
Video	10,240	10
Voice	1,280	20
Background	18,430	12.5

Being intrinsically built for QoS support, our IEEE 802.11e simulator works with four different traffic types (video, voice, background, and best effort; the latter is not considered in this work). We produced a new set of tests by using this feature, keeping the same topology, but modifying the type of packets in each stream:

I: Video stream from station 1 to station 3;
II: Background traffic from station 4 to station 2;
III: Voice traffic from station 5 to station 6.

We modelled the traffic according to the following criteria: for video, a new packet is deterministically generated every 10 ms; for voice, packets are generated in both directions, and the receiver generates a packet 5 ms after the sender has generated its own, this is repeated every 20 ms; for background traffic, a new packet is queued every 12.5 ms. These values were extracted from [12] and are summarised in Table 5. The results for the number of packets successfully delivered and lost are presented in Figs. 6 and 7.

While Fig. 6 presents the results for each packet, Fig. 7 addresses each stream, i.e, in the latter case a successful packet is only counted when it arrives at the final destination, while the former case a packet is counted as received whenever it arrives at any station.

For this reason, and looking at the results for the video stream (VI) alone, in the latter one only 100 packets arrived at the final destination, while in the former case 500 packets were successfully delivered. Note that this stream is being transmitted in a 5-hop path, and that the most relevant result is the 100 packets successfully delivered.

With only one stream the system behaves perfectly, all packets are delivered at destination. Since the rate at which packets are generated corresponds to a period longer than

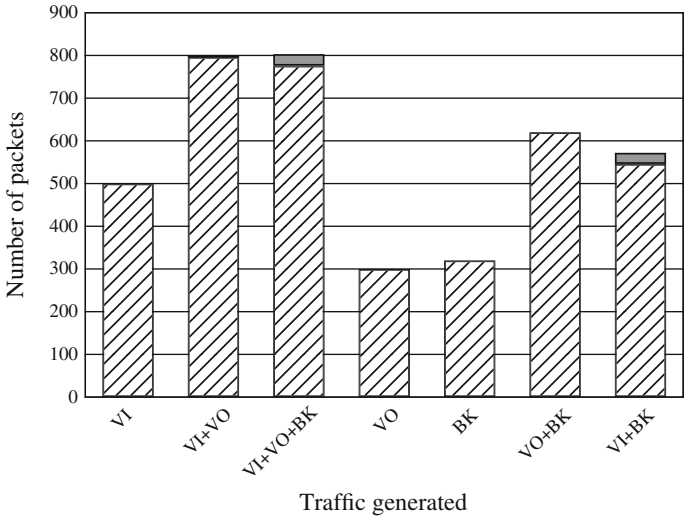

Fig. 6 Total of packets received (*hatched bar*) and lost (*gray bar*)

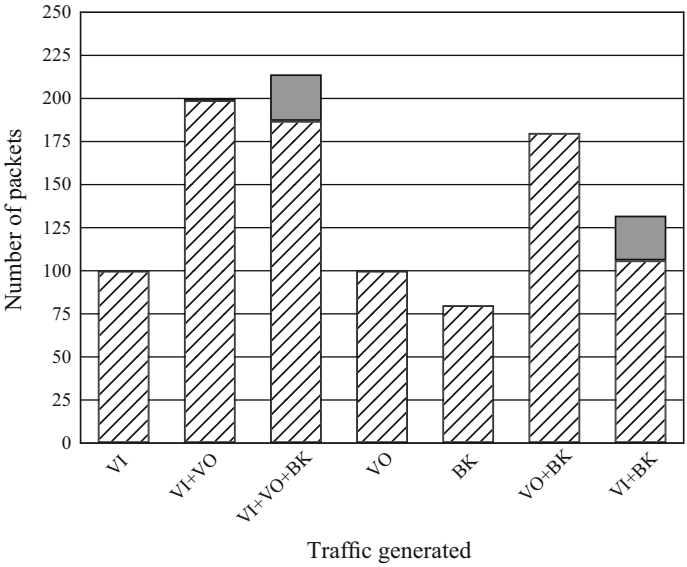

Fig. 7 Total of packets received at destination (*hatched bar*) or lost (*gray bar*)

the end-to-end delay, there are no collisions. However, when more than one type of traffic is being generated simultaneously, the packets start to collide and some of them are lost (current policy establishes that a station will drop a packet after it experiences eight collisions). From Fig. 7, one can conclude that background traffic is the one that causes more collisions, which can be explained by the larger size of each packet. Due to the limitation of the maximum packet size that can be transmitted at a time, each background packet will be fragmented into four fragments, increasing the number of packets in circulation. For voice traffic, the delay

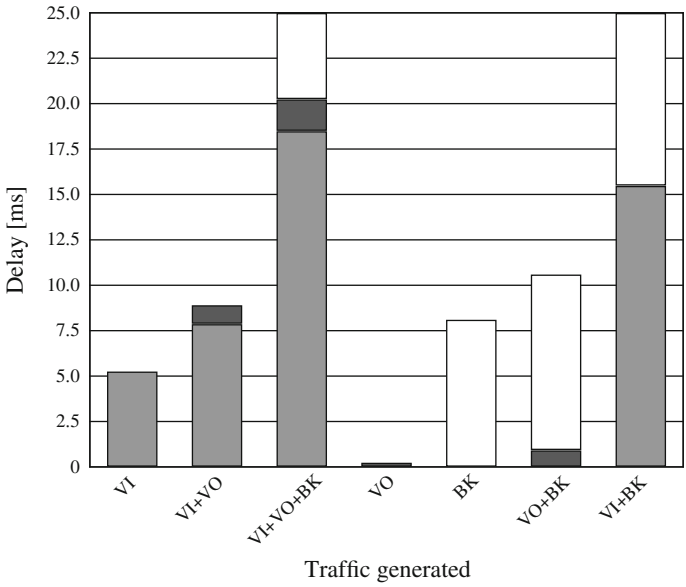

Fig. 8 End-to-end delay for VI (*gray*), VO (*dark gray*) and BK (*white*)

is very low, and it seems that it does not affect the remaining traffic at all. This is essentially due to the small packets generated by this type of traffic (no fragmentation is required).

This behaviour can also be noticed by looking at the end-to-end delay, Fig. 8. When only one stream is flowing the delay is minimum for each traffic class. However, if other streams are added the delay starts to increase. Background traffic seems to be the traffic class that more negatively affects the other streams. For example, with only one video stream (VI) the end-to-end delay is 5.3 ms, adding a voice stream (VI+VO) will increase this delay (i.e., end-to-end delay for video stream) to 7.9 ms, while adding just a background stream (VI+BK) will increase it to 15.5 ms. Of course, this delay is maximum when the three streams (VI+VO+BK) are flowing (18.5 ms). In order to keep the important part of the chart visible, the delay for the background traffic is not entirely shown at the VI + VO + BK and VI + BK bars (it corresponds to 266.5 and 250.7 ms, respectively).

5.2 Results for Our Cross-Layer Proposal

As mentioned in Section 4, we performed several tests with different formulas for calculating the cost of each link. In order to determine the best approach, we directly compare them in the same conditions. Table 6 summarises the considered parameters.

We monitor a video stream among stations 3 and 10, and inserted 11 background traffic flows and 9 voice flows. We then run a 10 s simulation for the first cost functions we created, like stated in the previous section. The results for these tests for ten randomly generated networks are presented in Figs. 9 and 10.

Proposal *D*, which assigned a cost that increases as the available link throughput decreases, achieves better performance than the others since generally delivers more packets (the difference can reach up to 10 times more packets—case 2 for example) and faster (the average end-to-end delay is lower). This can be explained by the fact that despite using paths with

Table 6 Simulation parameters

Parameter	Value
Inter arrival time for VI	10 ms
Inter arrival time for VO	20 ms
Inter arrival time for BK	12.5 ms
VI payload	10,240 bit
VO payload	1,280 bit
BK payload	18,430 bit
Simulation time	10 s
Field of test	$120 \times 120\,m^2$
Stations deployed	30
SIFS	16 μs
DIFS	34 μs
Slot time	9 μs
ACK payload	112 bit

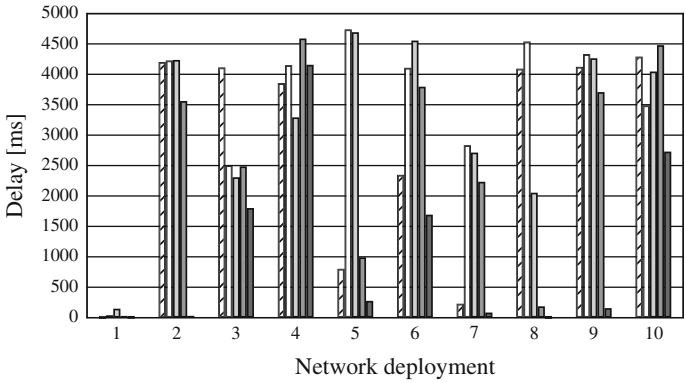

Fig. 9 End-to-end delay for video traffic for each algorithm: *No* (*hatched bar*), *A* (*white bar*), *B* (*light gray bar*), *C* (*gray bar*) and *D* (*dark gray bar*)

more hops, it uses paths with a higher SINR and raw bit rate, which increases the probability of a packet being successfully transmitted over the channel.

For the next set of tests, we modified some aspects of the simulation. While on the previous ones the routing table was only modified at the start of the simulation, we began to modify it periodically. We have chosen to update it every second, and as so we have extended our simulated time to 30 s. We decreased the traffic being generated to just one video stream, three background streams and three voice streams (generated according to Table 7).

In order to achieve statistical relevance, each simulation was repeated 10 times, and the final output is an average of the results obtained. We obtained the performance for the deployment of the stations shown in Fig. 11. Table 8 summarises the output for our simulations.

The first conclusion is that, similar to what have been previously observed in the preceding tests with different networks and a heavier load, proposal *D* outperforms the *No*, *A*, *B* and *C* ones, both in terms of end-to-end delay and in packets delivered at destination. If the

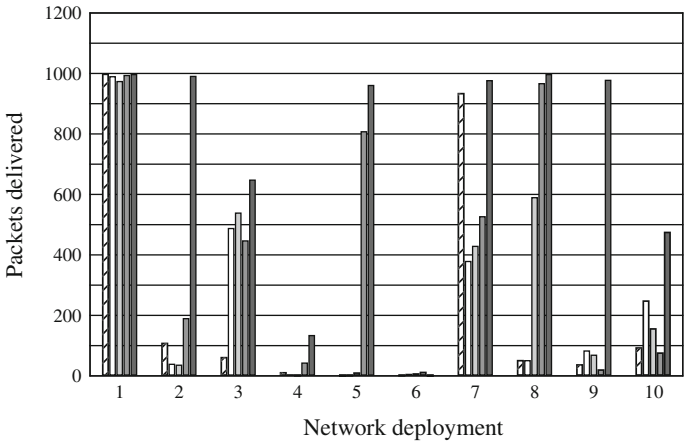

Fig. 10 Number of video packets delivered by each algorithm: *No* (*hatched bar*), *A* (*white bar*), *B* (*light gray bar*), *C* (*gray bar*) and *D* (*dark gray bar*)

Table 7 The traffic generated for the tests

Traffic type	Source	Destination
VI	3	10
VO	11	12
VO	12	13
VO	13	14
BK	3	4
BK	4	5
BK	5	6

algorithms that loop back the collisions occurred at the destination machine of a link are used (algorithms X', where X stands for either A, B, C or D), there is no improvement on the system. In fact, the packets suffer less delay, but the delivery rate is almost half of the one for the previous metrics (which only use static routing).

Nevertheless, when we consider the ratio "lost packets over packets sent successfully" (case X''), a great improvement on the number of packets delivered can be noticed, while the delay is kept the same (of the static solution—case X). We can compare directly solutions X, X', and X'', and get to the conclusion that the last one delivers more packets than the first one (X), and that this one delivers more packets than X'. Globally speaking, solution D'', which considers the following cost:

$$link\_cost = \sqrt{\left(56 \times 10^6 - link\_throughput\right)^2} \times (1 + collision\_rate) \qquad (12)$$

can deliver more than three times the number of packets delivered by the traditional solution that does not use any kind of cross-layer. Algorithm D presents a lower end-to-end delay, but it delivers less packets.

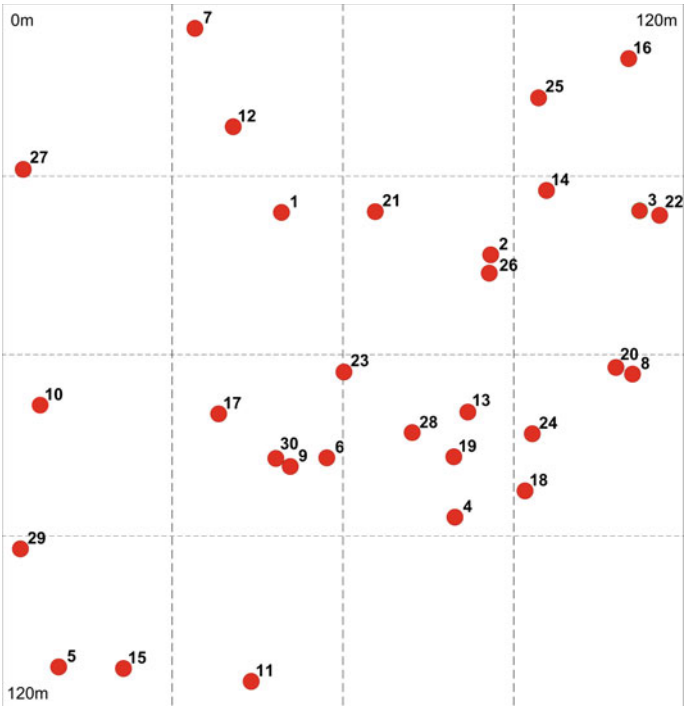

Fig. 11 The network that is being optimised

Table 8 The results of the proposed algorithms

Algorithm	Delay [ms]	Packets delivered
No	12514.54	157
A	10967.93	378
B	11106.00	377
C	12329.77	171
D	10543.86	516
A'	8190.77	144
B'	7831.30	116
C'	10975.38	120
D'	9738.40	138
A''	12500.82	381
B''	10693.08	402
C''	11215.53	186
D''	12179.88	540

6 Conclusion

In this work, experiments using different routing techniques were performed in a custom-made IEEE 802.11e simulator. The objective of this comparison was to understand which one would be more efficient. In our initial simulations, we verified if our version of the IEEE 802.11e simulator was behaving like expected, and if the output was accurate. This

preliminary step was necessary because the simulator is (still) being developed at our research group, Instituto de Telecomunicações, to allow the simulation of ad hoc networks (before our work on it, the simulator allowed only infrastructure mode).

In our routing experiments, first we used a static routing approach, in which the routing table was created at the beginning of the simulation and was not modified during the network lifetime. As a result of this experiment, the approach that privileges stations that are closer to each other achieves better performance than the others. Despite using a larger number of hops, the proximity of the stations allows the transmission to take place shortly (the link throughput is higher) and strongly (SINR is higher), being less vulnerable to collisions. Collisions are the major cause of delay because they require retransmission of packets and the necessary exponential backoff period. Its reduction decreases the delay and packet losses, which are the most important aspects in this study. Then, we inserted a dynamic routing calculation, in which periodically the routing table was updated in order to try to avoid stations that are being unable to transmit all the packets that they receive. We introduced the number of collisions in the stations as an input to the path computation. However, the obtained results were worse than the ones for the first approach. Nevertheless, the use of the metric "collision rate" in the computation of the path costs enabled to increase the number of packets delivered at the destination, while not significantly affecting the end-to-end delay.

Besides the simulation approach, we also presented an analytical approach for modelling an IEEE 802.11 ad hoc network. This model enables to determine the expected throughput of the network by using Markov chains. As a future work, we intend to validate and expand this theoretical approach to include additional metrics. We also plan to modify our simulator by adding a new function for transmitting broadcast packets. These are special packets that do not require an acknowledgement, and are used in the ETX and ETT to evaluate the quality of the links. With this broadcast packet functionality implemented, we will be able to compare those proposals with our ones, in order to evaluate which achieves better results.

How to combine several parameters, like we did, data rate and collision rate was a challenge. Although several formulations were attempted, the combinations that present better results are the ones presented here. However, this may not be an optimal solution to determine the cost of a link; for this purpose, a genetic programming algorithm is being implemented to combine, besides the two metrics (data rate and collision rate), also packet loss and packets in the buffer. Using an algorithm other than Dijkstra's one is also being considered. As the load changes in one node, Dijkstra's algorithm based on the weights of the links does not have the capability to provide a different path to deliver the required traffic. The genetic approach seems promising, as it can optimise the network in very few iterations with real time optimisation. The coding of the network and the genetic operation parameters are a challenge, but the authors from [1] proved that the genetic procedure converges to the Dijkstra's algorithm when simple fixed measures like Euclidean distances are used.

Acknowledgments The authors would like to acknowledge the following projects who provided financial support: IST-UNITE (a Specific Targeted Research Project supported by the European 6th Framework Programme, Contract number IST-FP6-STREP-026906), CROSSNET (a Portuguese Foundation for Science and Technology, FCT, POSC project with FEDER funding), Marie Curie Intra-European Fellowship OPTIMOBILE (Cross-layer Optimization for the Coexistence of Mobile and Wireless Networks Beyond 3G, FP7-PEOPLE-2007-2-1-IEF), UbiquiMesh, PLANOPTI, and Projecto de Re-equipamento Científico REEQ/1201/EEI/2005 (an FCT project). João Ferro and Orlando Cabral acknowledge the Ph.D. grants from FCT ref.

SFRH/BD/36742/2007 and SFRH/BD/28517/2006, respectively. Authors also acknowledge the COST Action 2100—Pervasive Mobile & Ambient Wireless Communications, and the Portuguese project Smart-Clothing.

References

1. Ahn, C. W., & Ramakrishna, R. (2002). A genetic algorithm for shortest path routing problem and the sizing of populations. *Evolutionary Computation, IEEE Transactions, 6*(6), 566–579. doi:10.1109/TEVC.2002.804323.
2. Barowski, Y., Biaz, S., & Agrawal, P. (2005). Towards the performance analysis of ieee 802.11 in multi-hop ad-hoc networks. In *Wireless Communications and Networking Conference, 2005 IEEE, vol. 1* (Vol. 1, pp. 100–106). doi:10.1109/WCNC.2005.1424483.
3. Cabral, O., Segarra, A., & Velez, F. J. (2008). Event-driven simulation for ieee 802.11e optimization. *IAENG International Journal of Computer Science, 35*(1), 161–173.
4. De Couto, D. S. J., Aguayo, D., Bicket, J., & Morris, R. (2003). A high-throughput path metric for multi-hop wireless routing. In *MobiCom '03: Proceedings of the 9th annual international conference on Mobile computing and networking* (pp. 134–146). New York, NY, USA: ACM. doi:10.1145/938985.939000.
5. Dijkstra, E. (1959). A note on two problems in connexion with graphs. *Numerische Mathematik, 1*, 269–271.
6. Draves, R., Padhye, J., & Zill, B. (2004). Routing in multi-radio, multi-hop wireless mesh networks. In *MobiCom '04: Proceedings of the 10th annual international conference on Mobile computing and networking* (pp. 114–128). New York, NY, USA: ACM. doi:10.1145/1023720.1023732.
7. Duffy, K., Leith, D., Li, T., & Malone, D. (2006). Modeling 802.11 mesh networks. *Communications Letters, IEEE, 10*(8), 635–637. doi:10.1109/LCOMM.2006.1665135.
8. Grilo, A., & Nunes, M. (2003). Link-adaptation and transmit power control for unicast and multicast in ieee 802.11 a/h/e wlans. In *LCN '03: Proceedings of 28th annual IEEE international conference on local computer networks* (pp. 334–345). doi:10.1109/LCN.2003.1243159.
9. IEEE. (1999). IEEE Std. 802.11; Wireless LAN Media Access Control (MAC) and Physical Layer (PHY) Specifications.
10. IEEE. (2005). IEEE Std. 802.11e; Wireless LAN Media Access Control (MAC) and Physical Layer (PHY) Specifications.
11. Malone, D. D. K., & Leith, D. (2007). Modeling the 802.11 distributed coordination function in nonsaturated heterogeneous conditions. *Transactions on Networking, IEEE 15*(1). doi:10.1109/TNET.2006.890136.
12. Ni, Q. (2005). Performance analysis and enhancements for ieee 802.11e wireless networks. *Network, IEEE, 19*(4), 21–27. doi:10.1109/MNET.2005.1470679.
13. Romdhani, L., & Bonnet, C. (2005). A cross-layer on-demand routing protocol for delay-sensitive applications. In *Personal, indoor and mobile radio communications, 2005. PIMRC 2005. IEEE 16th international symposium on, vol. 2* (Vol. 2, pp. 994–998). doi:10.1109/PIMRC.2005.1651590.
14. Stallings, W. (2006). *Data and computer Communications* (8th ed.). Upper Saddle River, NJ, USA: Prentice-Hall Inc.
15. Wang, Y., & Garcia-Luna-Aceves, J. J. (2004). Modeling of collision avoidance protocols in single-channel multihop wireless networks. *Wireleless Networks, 10*(5), 495–506. doi:10.1023/B:WINE.0000036453.53208.2d.

Author Biographies

João M. Ferro received the Licenciado degree in Electromechanical Engineering from University of Beira Interior, Covilhã, Portugal, in 2005. He is research assistant in Instituto de Telecomunicações, where is pursuing a Ph.D. in Electrical Engineering. His main research areas are routing in ad-hoc networks, routing in Wireless Sensor Networks, and cross-layer design. He is a member of Institute of Electrical and Electronics Engineers (IEEE), Vehicular Technology Society of the Institute of Electrical and Electronics Engineers (IEEE-VTS), Ordem dos Engenheiros, and International Association of Engineers (IAENG).

Orlando Cabral was born in Viseu, Portugal, on September 1978. He received the Licenciado degree in Mathematics from University of Trás os Montes e Alto Douro, Vila Real, Portugal. He finished his M.Sc. in 2006 in Informatic Engineering at University of Beira Interior, Covilhã, Portugal, where he is research assistant in Instituto de Telecomunicações. He is doing his Ph.D. in Electrical Engineering. He makes part of the teams UNITE, COST 290, COST2100 and CROSSNET. His main research areas are network capacity and coverage planning, multi-service traffic, heterogeneous networks, and cross-layer design.

Fernando J. Velez received the Licenciado, M.Sc. and Ph.D. degrees in Electrical and Computer Engineering from Instituto Superior Técnico, Technical University of Lisbon in 1993, 1996 and 2001, respectively. Since 1995 he has been with the Department of Electromechanical Engineering of University of Beira Interior, Covilhã, Portugal, where he is Assistant Professor. He is also researcher at Instituto de Telecomunicações, Lisbon, and was a Marie Curie Fellow in King's College London. He made or makes part of the teams of RACE/MBS, ACTS/SAMBA, COST 259, COST 273, COST 290, ISTSEACORN, IST-UNITE, and COST 2100 European projects, he participated or is participating in SEMENTE, SMART-CLOTHING, and UbiquiMesh Portuguese projects, and he was the coordinator of four Portuguese projects: SAMURAI, MULTIPLAN, CROSSNET, and MobileMAN. He has authored five book chapters, around seventy five papers and communications in international journals and conferences, plus twenty five in national conferences, and is a senior member of IEEE and Ordem dos Engenheiros (EUREL), and a member of IET and IAENG. His main research areas are cellular planning tools, traffic from mobility, simulation of wireless networks, cross-layer design, inter-working, spectrum sharing/aggregation, multi-service traffic and cost/revenue performance of advanced mobile communication systems.