Noname manuscript No.
(will be inserted by the editor)

On the use of Hidden Markov Processes and Auto-Regressive Filters to Incorporate Indoor *Bursty* Wireless Channels into Network Simulation Platforms

David Gómez · Ramón Agüero · Marta García-Arranz · Luis Muñoz

Received: date / Accepted: date

Abstract In this paper we thoroughly analyze two alternatives to replicate the bursty behavior that characterizes real indoor wireless channels within Network Simulation platforms. First, we study the performance of an improved *Hidden Markov Process* (*HMP*) model, based on a time-wise configuration so as to decouple its operation from any particular traffic pattern. We compare it with the behavior of the *Bursty Error Model Based on an Auto-Regressive filter* (*BEAR*), a previous proposal of ours that emulates the received *Signal to Noise Ratio* (*SNR*) by means of an auto-regressive filter that captures the "memory" assessed in real measurements. The study is based on an extensive simulation campaign carried out over the `ns-3` platform, and it also looks at the computational complexity of the two approaches (trade-off between accuracy and required simulation time).

Keywords Simulation · Wireless Channel Models · Hidden Markov Processes · Bursty behavior · ns-3

1 Introduction

Wireless technologies are constantly evolving and have become an essential part of everyday life. In particular, the birth and rise of the IEEE 802.11-compliant technologies has led to a remarkable increase of the popularity of wireless local area networks. As a consequence, the research community needs to address the multiple challenges posed by this particular type of networks. Although the empirical experimentation using off-the-shelf technologies seems to be the natural way to analyze the performance of such technologies, it also has clear limitations (for instance scalability and repetitiveness). These drawbacks, amongst others, bring about the need of simulation methodologies.

On the other hand, one of the strongest arguments against simulation techniques advocates the low accuracy of the most widespread propagation models, whose operation is usually "simplified". Although there exist advanced (and complex) approaches that could provide a higher level of accuracy (for instance ray modelling or electromagnetic theory), they require a rather long simulation time. This prevents their use within network simulation platforms, which are focused on the upper layer protocols, algorithms and mechanisms. Hence, the main reason behind the *abstraction* of the physical layer complexity in a network simulator is to ensure the scalability of the scenarios that are prone to be deployed, in which the number of elements might considerably grow. Hence, it is deemed necessary providing physical-level mechanisms able to reflect a behavior close to the one exhibited by real channels, in a reasonable amount of time, showing a good trade-off between accuracy and simulation complexity.

Although we can find IEEE 802.11 networks almost everywhere (even in open areas), *indoor environments* appear as the the most sensible scenario for this type of technologies. In this sense, many works have assessed the performance over this type of environments, being one of the most remarkable findings the hostility of the wireless channel, due to the presence of walls, furniture, people moving, etc. Several of these studies highlight that there is a clear memory effect within these particular scenarios, since consecutive frame error events are not independent, and tend to happen in bursts.

Besides, the ever-increasing computing capacity of devices has led to the development of advanced techniques to

D. Gómez (☎), R. Agüero, M. García-Arranz and L. Muñoz
Communications Engineering Department,
Edificio José Luis García García,
Plaza de la Ciencia, Avda. de Los Castros S/N
Santander, Cantabria (Spain)
Tel.: +34-942-200 919 ext.504
Fax: +34-942-201 488
E-mail: dgomez, ramon, marta, luis@tlmat.unican.es

replicate the behavior of real networks. As a consequence, novel and more powerful simulators, like `ns-3` [1], the natural successor to the popular `ns-2`, have become available. However, the mainstream propagation models are still far from being realistic, keeping the same drawbacks than its predecessors.

In this work we aim at mimicking the behavior of a real indoor channel, which was thoroughly studied by means of an empirical campaign. The results obtained from this analysis will be used to tune the performance of two novel wireless channel models, whose operation is rather different; whilst the former one relies on a *Hidden Markov Process (HMP)*, the second one, the *Bursty Error model based on an Auto-Regressive filter (BEAR)* model, employs an *Auto-Regressive (AR)* filter to estimate the received signal strength and to reflect the memory effect observed over real channels. Both models are able to replicate the bursty behavior exhibited over real indoor scenarios. We also compare them with one of the legacy `ns-3` simulator alternatives that, although providing accurate error rate and throughput average values, is not able to replicate the memory effect that was assessed over a real scenario. We also evaluate the impact of these models over the TCP performance, provided that it is severely jeopardized when it is used over wireless channels. Last, we also compare the computational complexity of the three approaches since, as mentioned earlier, there must be a trade-off between accuracy and complexity, especially when the number of wireless links is large.

The remainder of this paper has been structured as follows: Sections 2 and 3 describe the two models that mimic the bursty behavior exhibited by real indoor wireless channels (*HMP* and *BEAR*, respectively). Section 4 introduces an alternative that is originally provided by the `ns-3` simulator. Section 5 outlines the simulation campaign carried out to compare the performance of the three different solutions, whose main findings are also discussed. Afterwards, Section 6 positions this work with other contributions that have tackled the modeling of indoor wireless channels. Finally, Section 7 concludes the paper, advocating some issues to be addressed in our future work.

2 Channel model based on a Hidden Markov Process

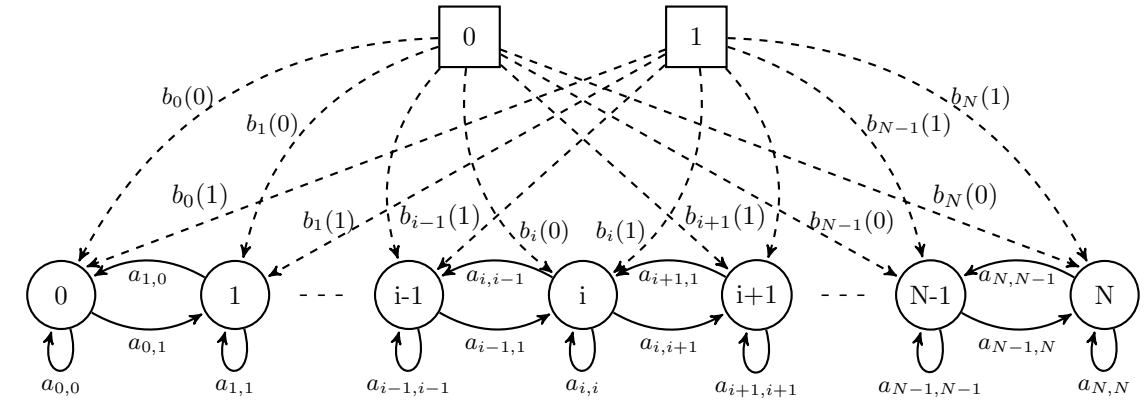
The use of *HMP* techniques to mimic real processes has gained popularity since their spring in the 60's decade. Countless research lines have loomed since then: we can find *HMPs* within speech recognition applications, to predict the location of people based on their habits or even to be used for novel bio-informatics studies, such as the analysis of bio-segments (for example gene prediction or protein folding).

We can define a *HMP* as a discrete system with N independent states (S_i, where i is the index of each of the states). The transitions between them follow a set of stochastic probabilities, which are referred to as transition probabilities, and are represented as $a_{i,j}$, probability of moving from i (current state) to j. Another set of probabilities is used to characterize the decisions within the states, mapping them with the possible output values of the system (*observables*); they are defined as $b_i(k)$, where i refers to the particular state, and k establishes the corresponding output symbol. It is worth mentioning that in *HMP*, unlike legacy Markov models, the states are "hidden", since each of them does not yield only one single output value, but there are various possibilities (each of them with a probability given by $b_i(k)$). Last, we also need to establish the initial state of the system; for that purpose, the vector $\Pi = \{\pi_i\}$ defines the probability of being at the i^{th} state when the system gets started.

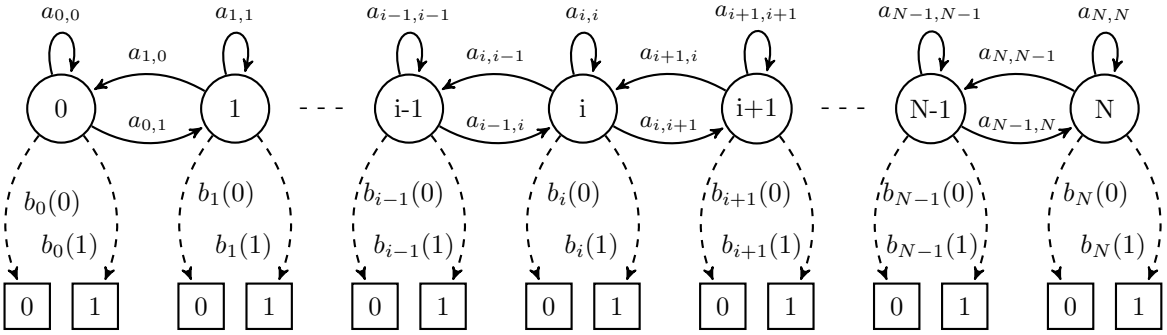
Taking into account how the model is implemented, we are able to define a complete *HMP* channel by means of the following elements.

1. Number of states in the model, N.
2. Number of possible output values, M; in this work, there will be only two: correct o erroneous frame.
3. Transition matrix (A), with dimension $N \times N$, containing all the state change probabilities, $a_{i,j}$.
4. Emission matrix (B), with dimension $N \times M$. Each element represents the probability of having output k at state i, $b_i(k)$.
5. The initial probability distribution of being at each state, $\Pi = \{\pi_i\}$. For the sake of simplicity, we will assume that $\pi_i = \frac{1}{N}$, and therefore the initial state will be randomly selected.

In order to configure this model, we used some real traces, obtained over a real indoor channel. The corresponding experimental setup used *WaveLAN 11 Mbps Lucent/Orinoco PCMCIA cards*, configured in a proprietary *Ad Hoc* (pseudo-IBSS) mode which did not use management frames; we fixed the maximum data rate of 11 *Mbps* during all the experiments. The corresponding wireless card driver was modified so as to be able to track whether each incoming frame was corrupted (CRC failed) as well as the received *SNR*. The maximum number of transmissions for an IEEE 802.11 frame was fixed to 4 and the RTS/CTS mechanism was disabled during the experiments. The transmitter and the receiver were separated by ≈ 15 meters, without line of sight, and with both metallic obstacles and people moving within the scenario (typical office environment). Last, but not least, we ensured that the presence of IEEE 802.11 traffic from other networks was negligible during the whole campaign. In the UDP case, we sent 10000 UDP/IP unicast datagrams, with 1472 bytes of payload, saturating the wireless link; to generate TCP traffic we used FTP to transfer a file of 10 *MBytes*. TCP Reno was used, with the Selective Acknowledgment and Timestamp options enabled; the Maxi-



(a) HMP general overview



(b) HMP simulator's insight

Fig. 1: Graphical interpretation of an HMP (birth-and-death process)

mum Segment Size was 1448 bytes, to maintain the same frame length that was used in the UDP case. The reader might refer to [2] for a more thorough description of the overall process.

For each of the 15 experiments (whose complete statistics are gathered in Table 1 in [2]), we generated a trace file, having, for every received frame, a 0 value if it was corrupted or an 1 if it was correctly received. Afterwards, with the resulting binary vector, as Figure 1a shows, the corresponding Markov chain is "trained" using the `hmmtrain` *Matlab's* function[1], establishing (as an additional constraint) that the resulting chain shall be a birth-and-death process. As can be seen, the chain itself is "hidden", whilst the "discrete observations" (reception events) are the main input arguments. This function returns both the transition and the decision matrices, which will be afterwards used by the `ns-3` environment.

On the other hand, the simulator-driven operation, depicted in Figure 1b, shows a different operation. In this case, the "visible" part of the process is the Markov chain itself, and its transition probabilities define the behavior of the model. Furthermore, the "hidden part" of the process corresponds to the frame reception decisions: error (0) or success (1). In other words, during the execution flow, the channel will be changing its current state, according to the subjacent transition probabilities ($a_{i,j}$) and, when a node receives a frame, the corresponding decision probability will establish whether the frame was correct or not, by comparing a random value with the corresponding $b_i(0)$ coefficient.

Cardoso *et al.* [4] also used a set of real traces to configure the different parameters of the *HMP*. However, there is a rather relevant difference between their approach and ours: in our case the measurements were done by saturating the wireless channel, ensuring that there were always frames to be sent at the transmitter, while that the authors of [4] fixed an interval of 10 ms between consecutive frames. With the corresponding traces, the model presented in this paper was configured by changing the duration of bursts from frames to time. In this sense, we are not bound to use any particular time between consecutive transmissions at the source node, and the behavior of the channel model is orthogonal to the traffic characteristics; this would bring about the possibility of using it with different types of applications (including TCP-based ones, in which the time between consecutive segments heavily depends on the dynamics of the corresponding congestion algorithms).

In order to complete this *time-based* model configuration, we need to estimate the *probability density function (pdf)* of the time spent at a particular state i, which follows

[1] This function uses the Baum-Welch algorithm [3] to estimate the chain parameters (transition and emission matrices, as well as the initial probabilities).

Table 1: UDP performance over a real indoor wireless channel. This set of results represents an illustrative sample of the 15 measurements that can be found in [2], whose trace files are used to "train" the *HMP* that defines the channel model's behavior

#	*Thput [Mbps]*	*FER*	*PER*	*EFB*	
				Avg.	*Max.*
Bad	2.33	0.517	0.179	6.21	821
Avg.	3.80	0.298	0.127	4.83	219
Good	4.79	0.163	0.025	2.63	144

a negative exponential distribution[2], $f_{T_i}(t_i) = \lambda_i \cdot e^{-\lambda_i \cdot t_i}$, with $\overline{T_i} = \frac{1}{\lambda_i}$, being the average time spent at state i. $\overline{T_i}$ can be calculated using the average number of consecutive frames at each of the states, $\overline{F_i}$, which can be derived using Eq. (1) where $p_i(j)$ is the probability of having j consecutive frames at state i.

Average consecutive frames at state $i = \overline{F_i} =$

$$= \sum_{j=0}^{\infty} j \cdot p_i(j) = \sum_{j=0}^{\infty} j \cdot a_{i,i}^{j-1} \cdot (1 - a_{i,i}) = \frac{1}{1 - a_{i,i}} \quad (1)$$

Eq. (2) can be used to determine the value of $\overline{T_i}$, the average sojourn time at state i, where ψ denotes the average inter-frame duration, if it is assumed to be constant.

$$\overline{T_i} = \psi \cdot \overline{N_i} = \frac{\psi}{1 - a_{i,i}} \quad (2)$$

From all the results obtained over the real-scenario testbed [2], we have trained the corresponding *HMP* configurations with three illustrative behaviors (selected from the 15 measurements), ranging from a *Bad* channel, characterized by rather negative transmission conditions, to a *Good* channel, whose operation gets closer to that which we could expect over an error-free link; an *Average* channel, representing an average behavior of the channel, was also selected. Table 1 summarizes the main performance values for each of these measurements.

2.1 Dynamic time-basis analysis

As mentioned earlier, the authors of [4] used a *frame-based HMP* able to mimic the behavior of a wireless channel for a very particular traffic pattern. In this sense, if the data rate generated by the source node was different, the behavior of the channel model would not be appropriate. The corresponding chain would not accurately mimic the real performance, since there is a tight relationship between such behavior and the configuration of the subjacent model.

[2] The legacy *frame-based* operation uses a geometric distribution, discrete "version" of the exponential random variable.

Since we aim at a more generic solution, we modeled the average transmission time per frame. In a first approach, we simply assumed that the time between two consecutive frames was constant for all states, ψ in Eq. (2), no matter the channel quality or the erroneous frame bursts. In order to have a more accurate solution, we modeled the average time between two consecutive frames depending on the current state and the corresponding number of retransmissions. For that purpose, we need to calculate how long it takes (in average) for a frame to be delivered to a receiver node, $\overline{\Delta_k}$; considering k retransmissions attempts, as shown in Eq. (3):

$$\overline{\Delta_k} = (k+1) \cdot \delta_c + \left[\left(16 \cdot \sum_{j=0}^{k} 2^j \right) - \frac{k+1}{2} \right] \cdot \sigma \quad (3)$$

where the first term corresponds to the deterministic contribution of the IEEE 802.11 DCF scheme, while the second term models the average value of the random time caused by the CSMA/CA procedure, which doubles the contention window for every retransmission (binary exponential backoff procedure). In particular, the following parameters are used:

- k. This value indicates the number of retransmissions that were sent for a particular datagram. A value of $k = 0$ indicates that a frame was correctly received at the first transmission. Since the maximum number of retransmissions was 3, $k \leq 3$. Besides, $\overline{\Delta_3}$ does not necessarily imply that the frame was correctly received after the third retransmission, although the overall time (for the four attempts) would be alike.
- δ_c. It is defined as the fixed time per frame, which accounts for the deterministic contributions (*Distributed Inter Frame Space - DIFS*, transmission time of the data frame, together with the physical header and preamble, *Short Inter Frame Space - SIFS* and transmission time of the IEEE 802.11 ACK, including the physical header and preamble). For the particular configuration that was used during the measurement campaign, the value of δ_c is $\approx 1.7\ ms$.
- σ. This parameter reflects the slot time of the contention window used by the IEEE 802.11 DCF mechanism.

Each of the states is characterized by the probability for a frame to be correct $(1 - p_i)$ or erroneous (p_i), being i the current state at the Markov chain and $p_i = b_i(0)$. From these two values we could easily derive the probability that frame requires k retransmissions[3]. Hence, the average time per frame that shall be used to translate the state duration to time units can be derived as shown in Eq. (4), where R is

[3] For example, the probability that a frame requires two retransmissions at state i can be calculated as $p_i^2 \cdot (1 - p_i)$, i.e. there are two consecutive erroneous transmissions and then a correct one.

Table 2: Statistics without saturating the link (*Bad* channel)

Channel	*FER*	*EFB*		
		Avg	*Var*	*Max*
Real (synthetic)	0.5191	3.76	22.18	19
Frame-based	0.4876	5.375	237.97	166
Time-based	0.5398	3.774	16.045	28

the maximum number of retransmission attempts per frame, which was set to 3 in our case.

$$E\left[\Delta_k|S_i\right] = \psi_i = \sum_{k=0}^{R}(1-p_i)\cdot p_i^k \cdot \overline{\Delta_k} + p_i^{R+1}\cdot\overline{\Delta_R} \quad (4)$$

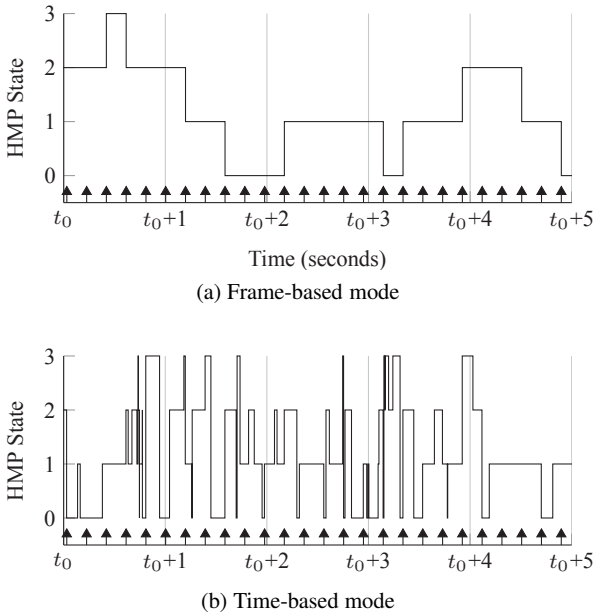
Finally, and considering that the traces that were used to train the subjacent chains were obtained under saturation conditions, we can model the average sojourn time per state $\overline{T_i}$ as shown in Eq. (5).

$$\overline{T_i} = \frac{\psi_i}{1-a_{i,i}} \quad (5)$$

2.2 Time-based Vs. frame-based modeling

One of the most relevant aspects of the configuration proposed herewith lies on the fact that it has been done by means of a *time-based* characterization. As was said before, most of the existing works use the traditional frame-based approach, which is sensible only when the simulation conditions are *exactly* the same as the ones that characterized the real traces used to train the *HMP* model (i.e. the 10 ms between consecutive transmissions in [4]); otherwise, the model would not be valid. On the other hand, if the configuration followed a time-based operation, the dependency on the traffic pattern would not longer be a problem.

In order to assert that a *frame-based* operation is not able to correctly capture a change of the traffic pattern, we carried out a complementary analysis. With the *Bad* channel trace, we configured the *HMP* using with both the frame-based and the time-based configurations. Then, we reduced the application data rate (in the simulator) to 600 Kbps, i.e. without saturating the channel. The average delay between two consecutive receptions would correspond to $\approx$ 20 ms, opposed to $\approx$ 2 ms that characterizes saturated IEEE 802.11b transmission. In order to assess the goodness of the results, we synthetically created a trace, by decimating the one corresponding to the *Bad* channel instance; in this sense we extracted one every 10 frames, roughly corresponding to an interval of 20 ms between consecutive frames. Furthermore, as this synthetic trace (only one every ten frames) does not include any 802.11 retransmissions, we disabled the corresponding scheme in the simulator, so as to enable a fair comparison. Table 2 shows the behavior exhibited by the two possible configurations and the statistics of the synthetic trace. As can be seen, the *frame-based* approach resembles the FER quite well, but keeps the memory behavior of the trace it was originally trained with, and therefore the bursts are much longer. On the other hand, the *time-based* model also mimics quite appropriately the EFB statistics, showing the greater flexibility of this approach.



(a) Frame-based mode

(b) Time-based mode

Fig. 2: Accuracy loss of the frame-based mode upon non-trained traffic conditions

As an illustrative example, we represent in Figure 2 the temporal evolution of the *HMP* state transitions, as well as the frame reception events (plotted as arrows). We can observe that the *time-based* mode (Figure 2b) keeps the state change rate along the time, independently of the traffic pattern; transition events are decoupled from the reception of frames and there are cases in which the channel visits and leaves a state within the interval between two consecutive receptions. Hence, the memory effect that was seen over a saturated channel is reduced (as was observed in Table 2), since the reception event of an arbitrary frame might be independent of the previous ones (the "bursty effect" disappears). On the other hand, the *frame-based* operation is tightly coupled on these physical receptions; Figure 2a shows that, even if the average time between transmissions was modified, the average time per state would be scaled likewise, since transitions are triggered by reception events and the "*mean number of frame receptions the channel will remain at the* i^{th} *state*" can be expressed as shown in Eq. (1), independent of the time. In other words, a *frame-based HMP* channel model would be tightly linked to the particular conditions that were used to train the model and to obtain the transition and emission matrices, so this approach will always yield a

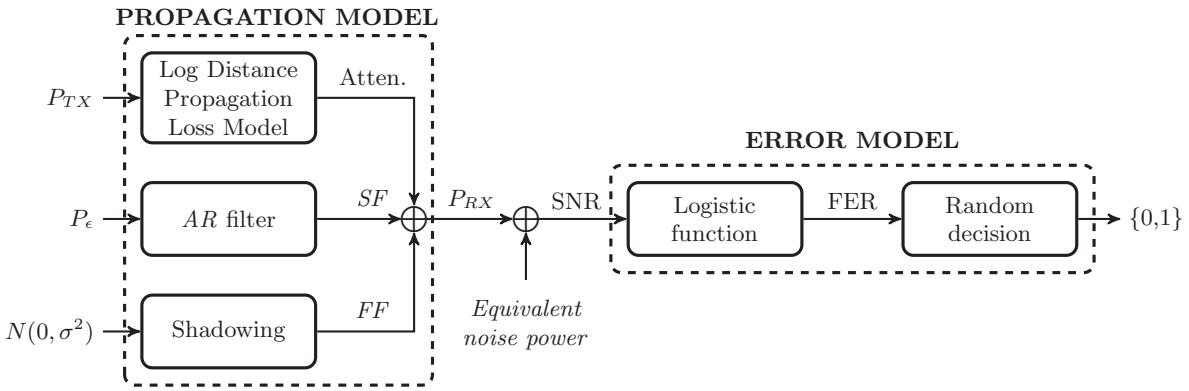


Fig. 3: BEAR channel model implementation

similar output as the one corresponding to the trace file used to "train" it.

3 Channel model based on an auto-regressive filter

Another approach to replicate the behavior of indoor wireless channels is to estimate the signal power at the receiver node. The *BEAR* model, originally proposed by Agüero *et al.* [2] follows this concept. In this work, in order to compare its performance with the one shown by the *HMP* models discussed in the previous version, we have ported its implementation to the `ns-3` network simulator. Figure 3 depicts the *BEAR* operation, from the physical transmission of a packet to the point at which the receiver entity decides whether that particular frame is correct or not.

The cornerstone of *BEAR* consists in estimating the received link quality by considering three different contributions. We abridge below their main features:

- The first one depends on the distance between the transmitter and the receiver nodes; it is normally characterized by a factor $d^{-\nu}$, where d represents the separation between the two nodes and ν is tuned according to the propagation loss model (we could refer to this parameter as "pathloss exponent"). Within this work we use a simple log distance propagation loss model, which is originally provided by the simulator.
- The second component reflects the slow variations on the received signal (*Slow Variation - SV*) which could be ascribed to the presence of physical obstacles within the path. In order to mimic such effect, BEAR uses an *auto-regressive* filter as shown in (6). The corresponding coefficients, $a[i]$ were tuned from the results obtained during the empirical campaign, using the *Yule-Walker* algorithm. As can be seen, the next value of the *SV* contribution, $SV[i]$, is "predicted" from the previous stored samples, $SV[i-j]$, limited by the *AR* filter order, T; $a[j]$ correspond to the filter coefficients. It is worth highlighting that each of the samples reflects a received frame (the time step would be $\approx 2\ ms$ in the particular configuration we used); in order to decouple the channel model from the traffic pattern (i.e. without saturation conditions), we included a timer, whose expiration would delete the previously stored samples, so that they not longer impact the SNR of new frames. Finally, ϵ is a white noise contribution with average power P_ϵ. The reader might refer to [2] for a more thorough discussion of the operation of this model.

$$SV[i] = \sum_{j=0}^{T} a[j] \cdot SV[i-j] + \varepsilon[i] \tag{6}$$

- The latter contribution reflects the multi-path wireless channel nature, leading to fast signal variations. The literature refers to this phenomena as *Fast Variation* (*FV*) or *shadowing* effect. In this work, it will be modeled as a random (i.e. Gaussian) variable with a mean zero and a variance of $\sigma^2\ dB^2$.

The sum of all these contributions, which are as well combined with an equivalent noise power to calculate the *Signal to Noise Ratio (SNR)*, is depicted in Figure 4: first, the deterministic propagation loss model returns the *Received Signal Strength (RSS)* as a function of the distance; besides, we can get the noise floor level from a legacy interference model, which is out of the scope of this work (Figure 4a). The second contribution is the result of the *AR* filter, yielding a signal with slow variations along the time (Figure 4b). The last component reflects a typical shadowing effect, showing a completely random nature (Figure 4c). As can be seen in Figure 5, which shows the *pdf* of the received SNR, the BEAR clearly shows that a different behavior is reflected for correct and erroneous frames. The average SNR for the erroneous frames is around 3/4 dB lower than for the correct ones; this reflects what was observed over real channels [2].

Afterwards, the overall *SNR* (Figure 4d) will be the input of a decision entity, responsible of establishing whether the received frame is correct or not. Its operation is detailed below.

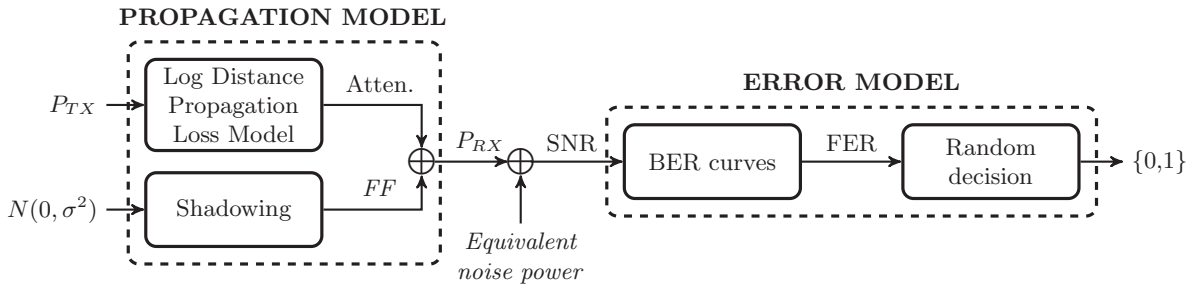


Fig. 6: ns-3's default channel model implementation

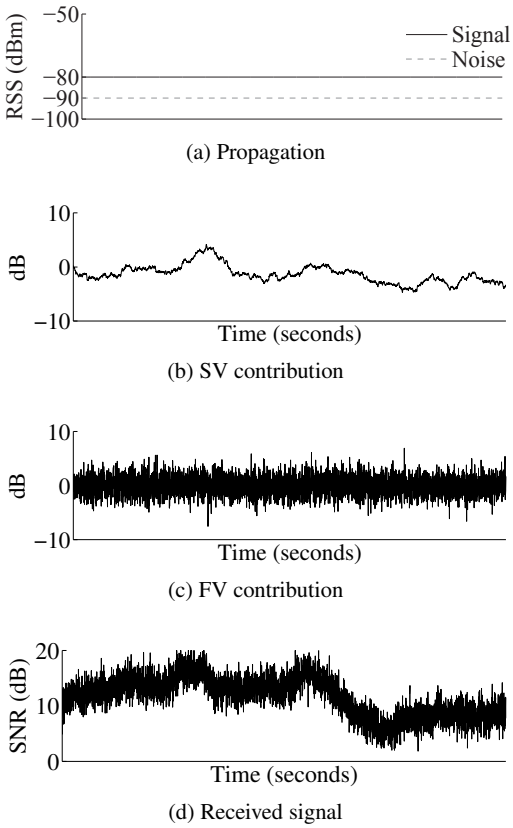


Fig. 4: Illustrative example of how *BEAR* estimates the received signal quality by means of three different contributions

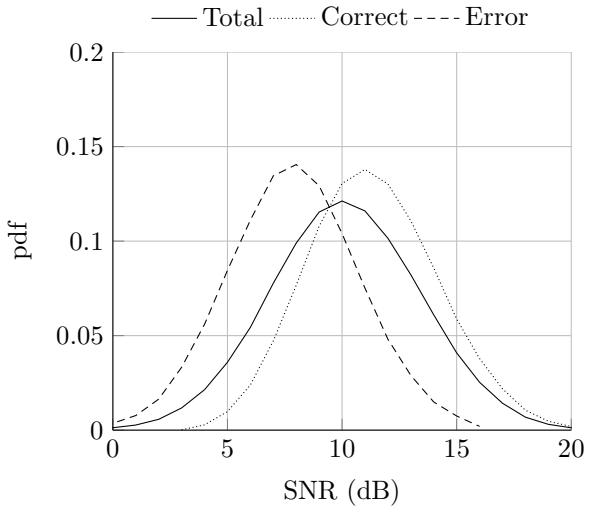


Fig. 5: BEAR's SNR *pdf*

– If the *RSS* is higher than the energy reception threshold, the frame delivery to the upper layers relies on the operation of the new error model, silently passing through the original physical reception.
– Instead of using the *Bit Error Rate (BER)* curves supported by the simulator, we have incorporated a logistic function, Eq. (7), which determines the *Frame Error Rate (FER)* as a function of the *SNR*; this relationship, as well as all its parameters (a, b, c and the two thresholds: LT and HT) was obtained using a curve fitting tool with the relationship that was empirically observed over the real channel.

$$FER = \begin{cases} 1, & SNR < LT \\ \dfrac{a}{1 + e^{b \cdot (SNR - c)}}, & SNR\,\epsilon\,[LT, HT] \\ 0, & SNR > HT \end{cases} \quad (7)$$

– The previous expression is only valid for 1500 Byte frames (worst case), and different relationships should be found for different lengths. For instance, the model considers that all IEEE 802.11 ACKs are always correct, since the probability of losing them is much lower than the one seen for data frames.

For a more thorough description of how the real traces are used to identify the *AR* filter coefficients and the logistic function parameters the reader can refer to [2].

4 Legacy model supported by the simulator

The last channel analyzed in this work corresponds to one of the mainstream IEEE 802.11 models originally supported by the `ns-3` simulator. In this particular case, as shown in Figure 6, we have configured the lower layer as follows: on the

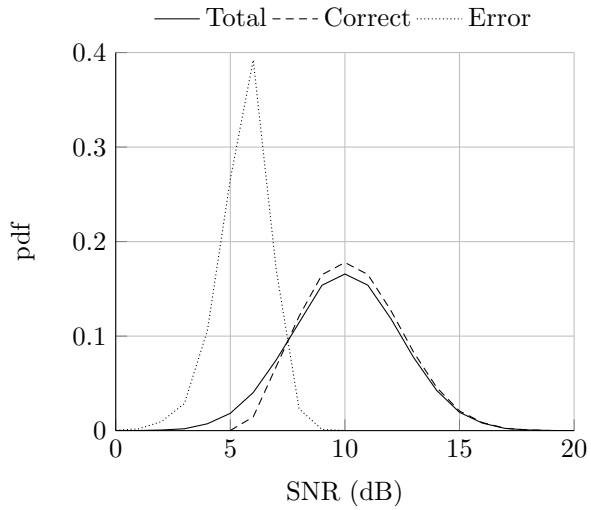


Fig. 7: Default model SNR *pdf* of an arbitrary transmission

first hand, we have two propagation elements: the former is based on a *log distance propagation loss model*, and returns a deterministic value as a function of the distance between the nodes. The attenuation factor L is obtained as shown in Eq.(8), being L_0 the path loss reference (in dB), ν the path loss distance exponent, d the distance between the source and sink nodes and d_0 the reference distance (in meters). Besides, the second contribution mimics a shadowing - *FV* - effect, which is modeled with a normal random process N $(0, \sigma^2)$.

$$L = L_0 + 10 \cdot \nu \cdot log_{10}\left(\frac{d}{d_0}\right) \tag{8}$$

As was done with *BEAR*, we derive the *SNR* of the received frame by adding the two aforementioned contributions, and the equivalent noise power (provided by an interference model). This value will be used to find the corresponding *FER*, using *BER* curves that are chosen according to the binary rate and the modulation. Once the *FER* is obtained, it is compared with a random value to decide whether a frame is correct or not.

It is worth mentioning that the operation of this so-called *Default* model has some similarities with *BEAR*; the main difference between them is the use of the *AR* filter in the latter one to emulate the slow variation of the channel. The overall behavior, in terms of the *SNR*, of the *Default* approach can be seen therefore as the sum of the deterministic propagation contribution (Figure 4a) and the shadowing component (Figure 4c). In addition, Figure 7 shows that, unlike the BEAR case, there is not a relevant correlation between the SNR and the presence of errors within the frame.

5 Simulation setup and results

After describing the different approaches that we have used to mimic the behavior of indoor wireless channels, we describe the setup of the simulation procedure we carried out to study the performance of the three alternatives. A source node sends 10000 data packets with an MTU of 1500 bytes, to a receiver entity. We also assume that there is always traffic to be sent at the transmitter, thus saturating the wireless channel. Besides, the transmission power (*txPowerDbm*) was tweaked to reduce the corresponding coverage, using a value of 0 *dBm*.

Below we depict the configuration for each of the different channel models.

- *BEAR* uses an order three auto-regressive filter ($T = 3$) with a white noise power $P_\epsilon = 5 \cdot 10^{-3} W/Hz$; besides, the *FV* contribution will be modeled by means of a normal random variable N(0, 2.8 dB^2).
- Regarding the *HMP* model, a 4-state hidden Markov chain will be used[4], which was trained with the three traces corresponding to the measurements depicted in Table 1.
- Finally, the so-called *Default model* uses the same shadowing contribution as *BEAR*: N(0, 2.8 dB^2). Furthermore, in order to compare its results to the ones observed with the *HMP* model, we have mimicked the same channel conditions (i.e. *Good*, *Average* and *Bad*), by changing the distance between the nodes so as to get a similar output in terms of the average *FER*.

We have carried out four phases: first, we characterize the behavior of the three different solutions using UDP traffic so as to reduce as much as possible the interplay of different upper layer mechanisms; the second one focuses on the addition of a distance-dependent functionality to the *HMP* model, tuned from the output obtained by *BEAR* in [5]; the third stage studies the impact of these channels over the TCP performance, assessing its sensitivity to indoor wireless environments. Finally, we compare the three models in terms of their computational cost, studying the corresponding *computational time-accuracy* tradeoff.

5.1 Raw channel characterization

We use UDP, because it is is the most appropriate transport protocol to evaluate the "raw" behavior of the lower layers, since it does not use any technique that might alter their intrinsic mechanisms, ensuring that the wireless channel stands as the actual system bottleneck.

Figure 8 shows the most relevant statistics to understand the performance of the various models, namely *FER*, *Packet*

[4] We also studied configurations with higher number of states (i.e. 8 and 16) and the resulting performance was alike.

Error Rate (PER) and *throughput*. We represent the cumulative distribution functions (cdfs) of the aforementioned parameters, after carrying out 500 independent experiments per configuration. First, the limited variability exhibited by the *Default* model can be easily seen (Figures 8a, 8d and 8g), with an almost deterministic behavior. On the other hand, all the *HMP* configurations show a similar performance (see Figures 8b, 8e and 8h), although the PER for the *Bad* configuration shows a higher variance, due to the impact of the error bursts, as will be discussed below. The third model, *BEAR* (Figures 8c, 8f and 8i), covers, with just a single configuration, almost the whole range seen over the real testbed. The modification of *BEAR's* configuration parameters (P_ϵ for the *SV* and $N\left(0,\sigma^2\right)$ for the *FV* contribution) could yield an even higher variability.

Figure 9 shows the relationship between the *FER* and the *PER* for each of the channel models, to illustrate their memory factor. Each figure includes, in addition to the simulation-based results, the values observed over the real wireless channel [2], as well as a curve that represents the behavior of a memoryless channel. In this sense, there was no memory (the random process associated to a frame reception is independent from the previous ones), we could say that $PER = FER^{R+1}$, since a packet will get lost after the consecutive reception of $R+1$ erroneous frames, being R the total number of retransmission attempts set by the IEEE 802.11 entity (3 in this work). In general, we can state that the *PER* could be calculated from the *FER* using Eq. 9, where γ gives an idea of the channel's memory impact (the lower γ, the higher the memory).

$$PER = FER^{\gamma} \tag{9}$$

Figure 9 yields the predictability exhibited by the *Default* model. As can be seen, all simulations lie within the memoryless behavior, $\gamma \approx 4$. On the other hand, *BEAR* shows a higher variability, spanning (for a single configuration) the whole set of values that were observed during the real measurement campaign. However, there are some particular measurements that are not properly replicated by BEAR, since its γ parameter is slightly higher that the one of the real channel. Finally, the *HMP* offers, considering its three configurations altogether, a reliable modeling of the memory assessed over a real scenario, although the variability is (for each of them) much lower than *BEAR's*. It can be also seen that the value of γ is, for all the *HMP* configurations, lower than the one seen for the *BEAR* channel.

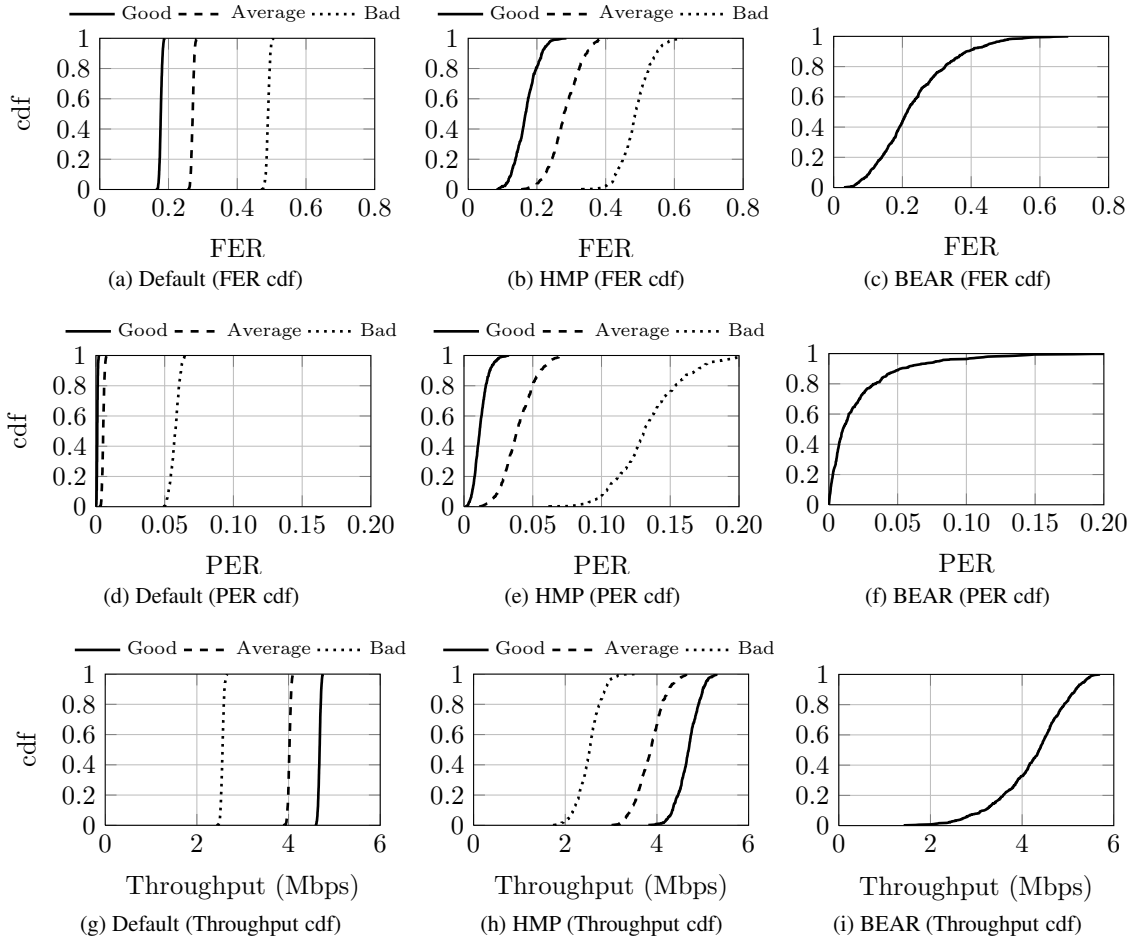
As was discussed before, the behavior of the different models in terms of *FER* and *PER* has a direct relationship with the "bursty" response of the channel. Figure 10 shows the *EFB's pdf* and *complementary cumulative distribution function (ccdf)* for the three models. It is worth highlighting that a burst longer than 4 frames would lead to a packet loss. We can again observe the poor bursty behavior offered by the *Default* model (Figures 10a and 10d), where the vast majority of *EFBs* are shorter than ten frames; in fact, only the *Bad* configuration was able to replicate the appearance of bursts longer than 5 frames. As for the *HMP* model, its *pdf* (Figure 10a) shows that bursts are much higher, having a non-negligible probability for *EFBs* > 10 frames, even for the *Good* configuration. On the other hand, *BEAR* (Figures 10c and 10f) is able to cover (even though for a single configuration) a broader range of behaviors, from short *EFBs* ($\approx 85\%$ are shorter than four frames) to long ones ($\approx 5\%$ are longer than 10 frames).

On the other hand, it has been shown [2] that the probability of having an *EFB* of 100 or more frames over the real channel is actually lower than 0.7%, and therefore we can conclude that both *HMP* and *BEAR* reflect this behavior with a reasonable level of accuracy. Although they yield bursts longer than 100 corrupted frames, their probability is very low: $\Pr\{EFB > 100\} \leq 10^{-3}$, for both models.

5.2 HMP distance-aware operation

Unlike the legacy wireless channel simulation models, one important shortcoming of the *HMP* model basic configuration is that it cannot provide any dependency to the received signal strength[5] by itself. We have to choose a duple of transmission-emission matrices (A and B) during the scenario setup, keeping those settings throughout the simulation. However, we can improve the operation of this model by exploiting the results achieved by *BEAR* in [5], which will be used to tune the performance as a function of the distance between nodes. We first "discretize" the response along the distance at a finite number of points. By using a distance-based study of the *BEAR* model, and the corresponding real behavior, we have chosen up to seven different measurements from Table 1 in [2] to configure the various *HMP* instances. Figure 11 shows the performance, in terms of *FER*, *PER* and *throughput* while we vary the distance between transmitter and receiver. Provided that the distance thresholds were configured according to the *FER*, we can observe in Figure 11a that all these values are reliably mimicked, as well as the resulting throughput (Figure 11c), which covers all the range showcased by *BEAR*. On the other hand, despite the *PER* shows an appropriate behavior, as shown in Figure 11b, there are two small "misalignments": first, since *HMP's* memory factor γ is greater than *BEAR's*, the *PER* is slightly higher, especially for low *FER* values; on the other hand, the current implementation is not able to cover *PER* values higher than 0.4.

[5] There is no relationship between the distance between nodes and the erroneous performance of the transmission.

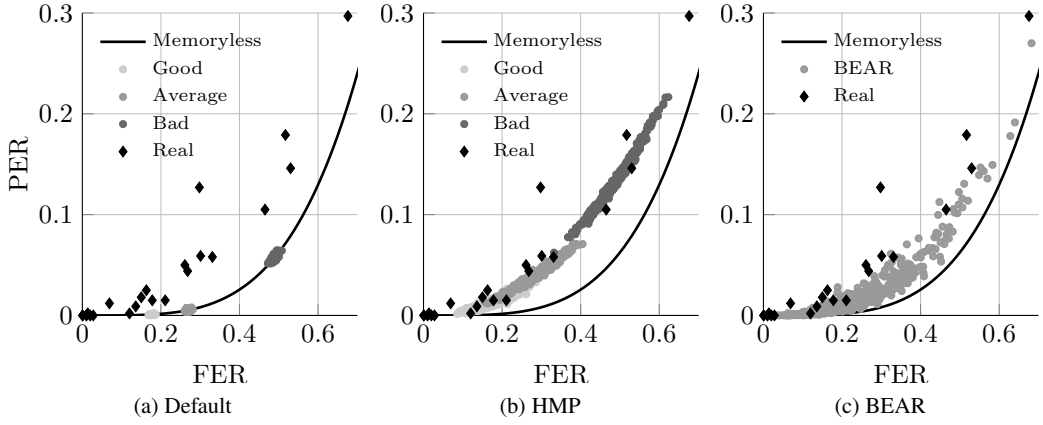


(a) Default (FER cdf) (b) HMP (FER cdf) (c) BEAR (FER cdf)

(d) Default (PER cdf) (e) HMP (PER cdf) (f) BEAR (PER cdf)

(g) Default (Throughput cdf) (h) HMP (Throughput cdf) (i) BEAR (Throughput cdf)

Fig. 8: Performance indicators of the different channel models

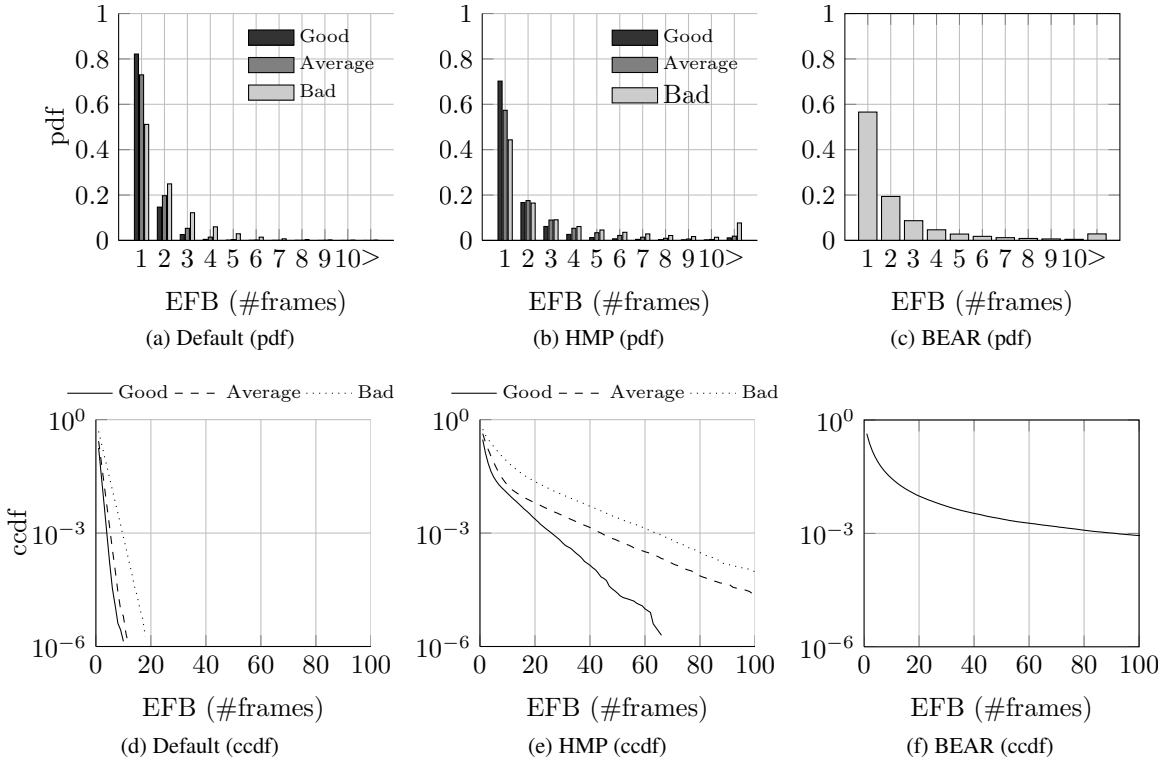


(a) Default (b) HMP (c) BEAR

Fig. 9: Relationship between FER and PER for the different channel models

5.3 Impact of a packet erasure channel over TCP

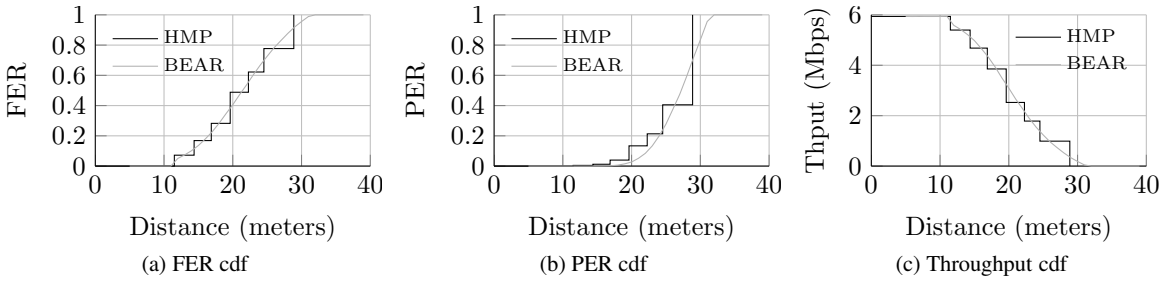
Although the results obtained with UDP traffic are appropriate to reflect the "raw" behavior of indoor channels, it is also interesting to assess the impact of these environments over rather different transport protocols, like TCP. As mentioned before, it is not able to determine the cause of a segment loss, either brought about by the congestion of intermediate routers' buffers or as a consequence of the hostile conditions of the wireless channel. The default TCP interpretation is always the same: when a segment gets lost, the TCP entity associates this event to a congestion situation, hence the congestion control mechanisms will act accordingly, by reducing its sending congestion window. Furthermore, this loss of information might lead to the reception of an out-of-order segment, event that triggers a *Dup ACK* backwards



(a) Default (pdf) (b) HMP (pdf) (c) BEAR (pdf)

(d) Default (ccdf) (e) HMP (ccdf) (f) BEAR (ccdf)

Fig. 10: *EFB* characterization of the different channel models



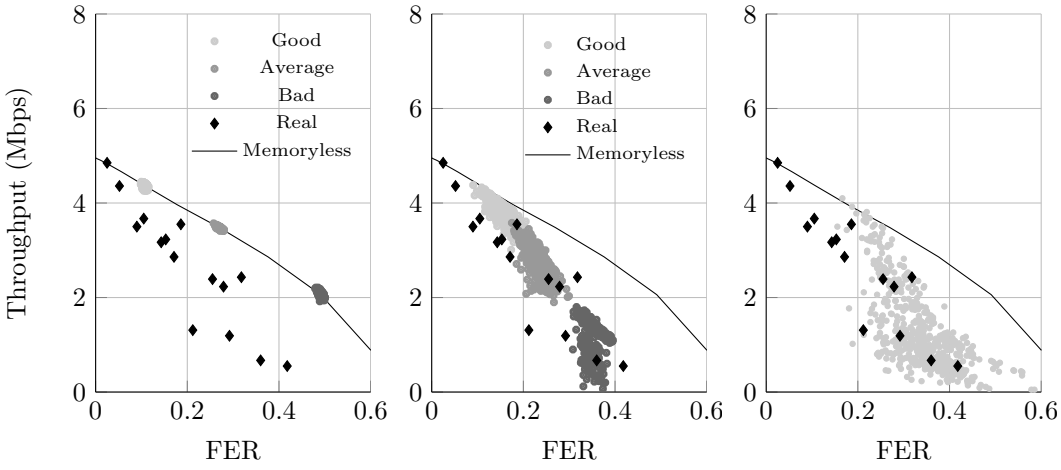
(a) FER cdf (b) PER cdf (c) Throughput cdf

Fig. 11: HMP behavior as a function of the distance

delivery. After the consecutive reception of three of them (*Triple Duplicate ACK*), the *Fast Retransmit* algorithm will immediately trigger the retransmission of the corresponding segment. In few words, the presence of long error bursts (as observed over the real channel) during a TCP transmission has a huge impact over the system performance. For these reasons, it is essential to provide channel models that accurately capture this behavior, in order to evaluate the performance of such type of protocols over wireless networks.

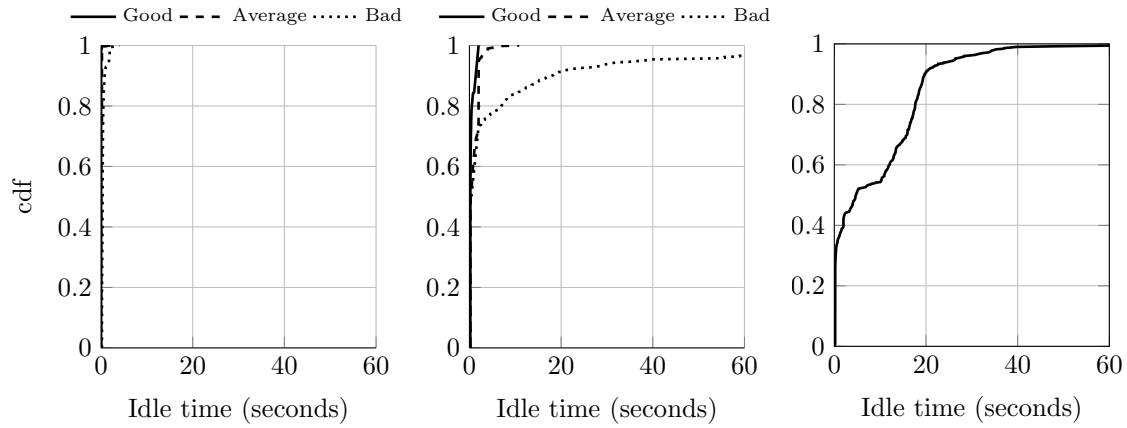
Figure 12 shows the relationship between the *FER* and the *throughput* for the three studied models, as well as the values observed during the empirical campaign and an upper bound, which is established by means of a memoryless channel. First of all, we can observe the poor performance exhibited by the *Default* model (Figure 12a), with a clear memoryless behavior. On the other hand, the different *HMP* configurations present an acceptable level of variability (Figure 12b). Finally, *BEAR* offers again the broadest range of possible outputs, mimicking quite well the memory effect shown by the real measurements.

As said before, TCP uses two retransmission triggers: the first one, the so-called *Fast Retransmit* algorithm, establishes that a segment must be immediately retransmitted after the reception of a *triple duplicate ACK*; on the other hand, if, after sending a segment, the transmitter does not receive an acknowledgement within a time interval (the *Retransmission TimeOut, RTO*), it would be retransmitted. The latter one causes a stronger impact over the TCP performance, since it might lead to long inactivity periods (during which the channel is not used). Provided that the *RTO* gradually increases (following a binary exponential backoff algorithm) after any timer-triggered retransmission, these idle times might reach rather long values. Hence, they could severely jeopardize the overall TCP performance. Figure 13 shows



(a) Default (b) HMP (c) BEAR ($P_\epsilon = 5\ mW/Hz$)

Fig. 12: Relationship between FER and throughput for the different channel models (TCP)



(a) Default (b) HMP (c) BEAR

Fig. 13: Idle time *cdf* of the different channel models

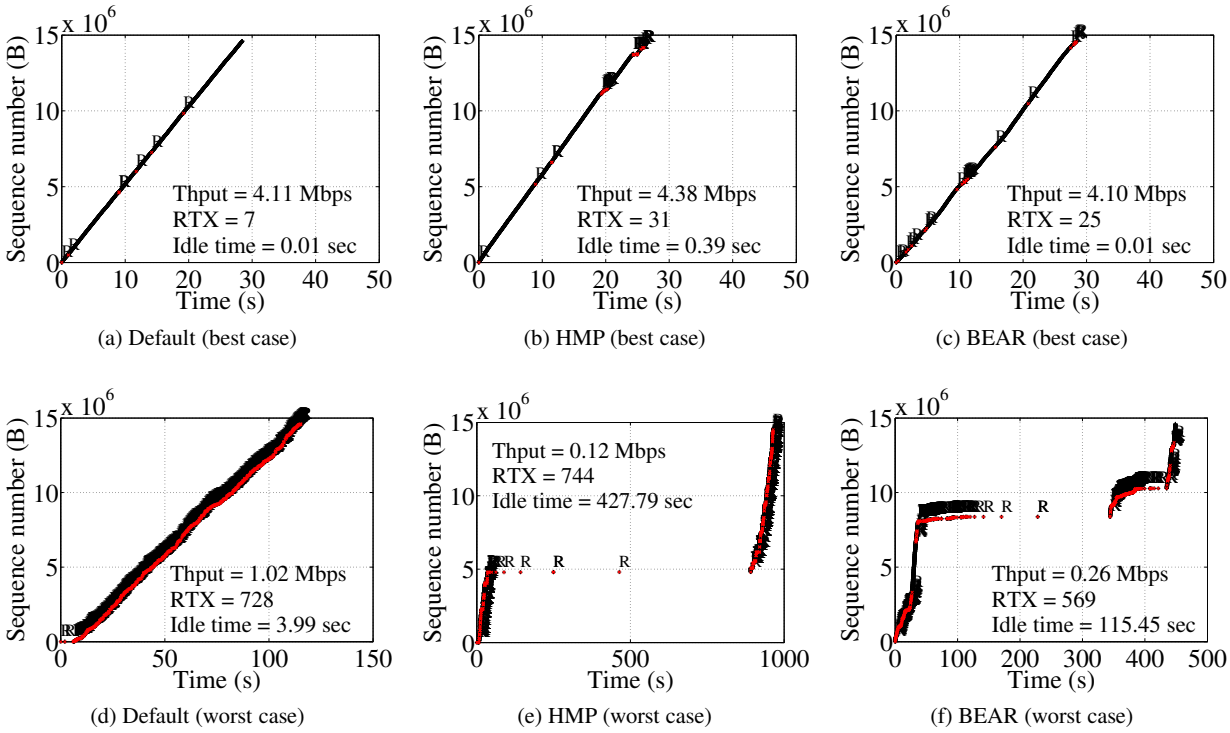
the maximum idle time *cdf* for the different channel models. As can be seen, the *Default* approach shows a complete lack of bursty behavior, and the idle times always stay below 2 seconds (even for the *Bad* configuration); on the other hand, *HMP* yields a rather predictable behavior for the *Good* and *Average* configurations (with values much lower than the ones observed over the real channel), and only the *Bad* instance leads to idle times greater than 5 seconds, but at the expense of causing inactivity periods greater than 60 seconds in around 4 % of the cases, which do not accurately reflect the real channel behavior. Besides, *BEAR* appropriately mimics the variability assessed during the characterization carried out over the real channel.

Finally, it is interesting to analyze the temporary evolution of some illustrative individual measurements. For that we will represent the "Time-TCP Sequence Number" graph of some particular experiments. Figure 14 shows (per channel model) one good and bad example. At first sight we can conclude that the *Default* model does not provide any variability at all (as was also discussed earlier), since the behavior remains almost alike for the 500 independent simulations. On the other hand, we can see that the other two models are actually able to lead to TCP connections with a rather opposite behavior, as was the case for the real channel.

5.4 Computational cost assessment

In this last phase we aim to characterize the computational cost (in terms of simulation time) for each of the wireless channel models studied in this work. For this purpose, we need to make various changes on the scenario, as described below.

1. The parameter that is modified for this particular analysis is the number of nodes deployed along a line topology. The first node is the source and the last one takes the receiver role. We increase the number of nodes from 2 to 32.
2. All the configurations present a common aspect: besides the particular operation of the proposed models, we have



(a) Default (best case) (b) HMP (best case) (c) BEAR (best case)

(d) Default (worst case) (e) HMP (worst case) (f) BEAR (worst case)

Fig. 14: Time - Sequence Number for arbitrary measurements of the three channel models

added another propagation loss entity, a range distance propagation loss model. It defines two different thresholds: the first one limits the radius within which every frame will be successfully received (when $P_{RX} > RX_{threshold}$); on the other hand, a second one will be used to establish the distance that limits the Carrier Sense (CS) threshold, zone in which the node will sense that another transmissions is happening and will therefore deter its own transmission until it is finished ($RX_{threshold} > P_{RX} > CS_{threshold}$). Out of these zones, frames will always be corrupted ($P_{RX} < CS_{threshold}$), to ensure that a packet needs $N-1$ hops to reach the destination and at the same time to avoid the "hidden-terminal" effect, since the i^{th} node will be able to overhear the transmissions carried out by up to its two-hop neighbors.

3. During the simulations, 5000 UDP datagrams are sent between the source and sink nodes, using a Constant Bit Rate (CBR) application that delivers packets at a rate of 100 Kbps (non-saturating conditions).
4. The remaining configuration parameters keep the values chosen in the previous simulation campaigns.
5. Finally, we carried out a total of 25 independent runs for each of the scenarios.

Figure 15 represents the normalized simulation time of each of the runs (using the lowest value as the reference: two nodes, *Default* channel), as well as the 95% confidence intervals (as can be seen, the variability of the results is almost negligible), as a function of the number of nodes deployed

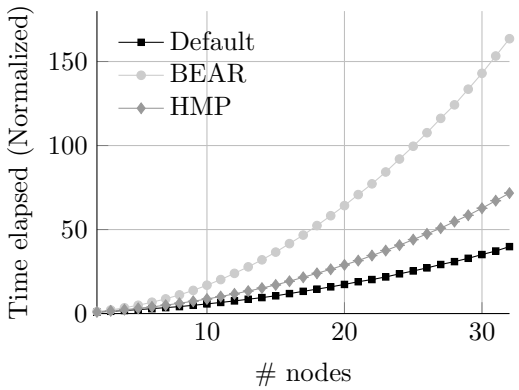


Fig. 15: Computational cost comparison between the studied channel models as a function of the number of nodes over a line topology

along the line topology. According to the obtained results, it is easily inferred that the *Default* model requires less time to decide whether a received frame is correct or not, since its complexity is rather low: after calculating the received signal power and the SNR from the propagation and interference models, respectively, the error model maps the SNR into a FER value, based on the BER curve associated to the appropriate modulation scheme (11 Mbps - Complementary Code Keying (CCK) in our work) [6,7]. The *HMP* model has a slightly higher cost although the simulation time keeps an acceptable growth rate with the number of nodes. The complexity of this model lies on the matrices (transition and emission) that define the behavior of the wireless channel.

Upon the reception of a frame by a node, the first step consists in looking up the current state of the Markov chain at the receiver node; after that, the error model takes the emission error value $b_i(0)$ belonging to such state and decides whether the frame is correct or not. It is worth mentioning that the change-of-state process is completely orthogonal to the reception of frames, since a transition will be triggered by a negative exponential random variable, whose mean value is obtained as shown in Eq. (5). Finally, *BEAR* is the model that shows the longest simulation time, being penalized as the number of nodes gets higher. In a nutshell, besides the operation carried out by the *Default* model, BEAR needs another signal contribution (i.e. the *SV* one), obtained by means of the AR filter, which stores the last T samples of the SNR values and operates with them following Eq.(6). Upon the calculation of the overall SNR value, a logistic function is used to decide whether the received frame is correct or not.

5.5 Discussion

At this point, we can abridge the main conclusions extracted throughout this simulation campaign: on the first hand, we have checked that, albeit the *Default* model captures quite appropriately the behavior of a wireless channel in terms of average *FER* and *throughput*, it is not able to reflect the memory factor observed over real scenarios, leading to an almost deterministic output, far from being realistic. Although its computational complexity is very low, its use is not recommended, because it does not provide the minimum level of accuracy. On the other hand, the *BEAR* model leverages a remarkable varying performance, covering the whole range of results observed over the real channel with a single setup (i.e. the *AR* filter noise input P_ϵ and the variance of the shadowing model); however, this model is complex, and its simulation time is long, especially when the number of nodes gets higher. The *HMP* model provides an interesting trade-off, since it offers a certainly good degree of variability, showing an appropriate memory behavior, while keeping a reasonable computational growth rate (as a function of the number of nodes deployed over the scenario). That, together with the feature proposed in this work that provides the model with distance-dependent behavior makes the *HMP* model a very appealing alternative.

Another aspect that should be looked at, especially considering the *empirical* nature of both models, is how *reproducible* they are and how they can be configured for different conditions (other IEEE 802.11 variants, different frame sizes, etc). *BEAR* has two main parts: the modeling of the SNR and the dependency between this and the probability for a frame to be erroneous; it can be said that the SNR is independent from the frame size and the type of IEEE 802.11 modulation scheme, as it is estimated during the PLCP [8]; hence, in order to include these into the model we should just find an appropriate match between the SNR and the FER (this could be done, for instance, by means of lookup tables). In this sense, Lertpratchya *et al.* have recently used the *BEAR* model to study the *bursty* behavior of wireless channels [9]. On the other hand, the HMP could be more complex to be updated so as to consider different frame lengths and would probably require additional measurement campaigns; the training of the subjacent markov chains is rather systematic, though, and therefore the same methodology could be used to consider different frame lengths or modulation schemes. Nonetheless, the time-based configuration would still be of outer relevance so as to decouple its operation from the particularities of the traffic patterns.

6 Related work

The first works within this research line focused on the empirical characterization of IP protocols performance over *Wireless Local Area Network (WLAN)*, using the *AT&T's WaveLAN* wireless network adapters, which appeared even before the approval of *IEEE 802.11* standard, in 1997. Within this group we can find the one carried by by Eckhart *et al.* [10], where, from a set of packet traces captured at a receiver entity, including information of both the signal and noise levels, as well as the presence of errors, they assessed the influence of different interference and attenuation sources in terms of both packet and bit error rates. At the same time, Nguyen *et al.* [11], following a similar methodology, aimed at finding a realistic model to emulate the behavior of a wireless channel. Starting from both the error rates and the burst lengths, they proposed an enhanced *2-state Markov model*, in which they substituted the traditional geometrical distributions used to model the time spent at each state with other approaches, which mimicked more accurately the empirically observed ones.

Other works focused on the characterization of an 11 Mbps *IEEE 802.11b channel*; it is worth highlighting the research done by Ikkurthy and Labrador [12], in which they studied the effect of errors over a coded video (using the widespread *MPEG-4* video compressor) transmission. They carried out an experimental campaign in which they modified the packet size, and analyzed the erroneous and correct packet bursts, and their probability distributions, for a scenario where both nodes were separated a distance of approximately 22 meters. Comparing these results with the ones gathered by Nguyen *et al.* [11] at 2 Mbps, they came to the same conclusion: *a simple geometric model does not precisely reflect the real behavior*. They also concluded that for 1500 bytes packets, 90% of the error bursts are shorter than 4 packets. Nonetheless, the authors did not specify the number of MAC retransmissions which were used during their mea-

surements. In fact, they coined the term *error bursts*, whilst a more precise expression would have been *packet losses*.

Furthermore, there are a number of works which state that traditional channel models based on Markov chains (*Gilbert-Elliot*) are not able to reliably reflect the behavior observed over real indoor wireless environments. In [13] the authors show that this model is not able to reflect time periods with a high frame loss rate, which might have a strong effect over video transmission, in terms of the quality perceived by the end user as well as to streamline the design of appropriate error control procedures. In [14] the *Gilbert-Elliot* model was used to establish the most adequate parameters to reflect the perceived quality of a voice transmission using an adaptive *Frame Error Correction (FEC)* scheme, setting out the need of a research effort to come with more realistic channel models.

More recently, the authors in [15] rely on the results gathered from an experimental campaign carried out over an outdoor rural environment, in order to propose a model able to mimic the observed frame error rate; they also identified the need of conducting a similar analysis over indoor scenarios. Finally, Cardoso *et al.* [4] question the appropriateness of a 2-state Markov chain to reflect frame loss processes which are seen over real indoor IEEE 802.11 channels. They propose and evaluate a novel model based on an *HMP*. Although the use of *HMP* to model wireless channels was already discussed by Turin and van Nobelen [16] and Zhu and Garcia-Frias [17], to our best knowledge, one of the first works proposing their use within *Network Simulation* platforms was that of Cardoso *et al.* Their results are compared with a batch of traces obtained from a set of experiments carried out at a constant bit rate, and without considering the IEEE 802.11 retransmission scheme. In addition, they established the traffic pattern at the source node, having a fixed interval (at the application layer) of 10 *ms* between consecutive packets, which is rather high if compared with the average *IEEE 802.11* time gap between two consecutive transmissions (i.e. $\approx$ 2 *ms* for an IEEE 802.11b saturated channel at 11 Mbps). Besides, they did not include any reference to the scenario topology (i.e. distance between the two nodes) nor to the received *SNR*. On the other hand, both the frame error rate and the burst lengths they obtained are considerably lower than those we aim at modeling herewith. They conclude that an 11-state *HMP-based* model, with a birth-death structure was able to reflect (quite accurately) the first and second statistics of the packet losses measured over a real testbed. However, as already discussed in Section 2, this model is not able to reflect a realistic behavior under different traffic conditions than those that were used to configure the *HMP*.

Besides, with the main goal of overcoming some of the main wireless modeling drawbacks, we proposed a new channel model: *BEAR* [2]. As was already mentioned throughout this document, it is based on the modeling of the SNR, resembling a set of traces obtained during an extensive measurement campaign carried out over a real indoor scenario. Its most distinguishing feature is that it aims to reflect the memory effect shown over a real channel, using an auto-regressive filter. We compared its performance with other alternatives, widely used in the literature (all of them showing a memoryless behavior) as well as the traditional *Gilbert-Elliot* model. *BEAR* outperformed the rest of the channels studied by the authors, but none of them was characterized by offering a *memory* behavior.

In what respects to `ns-3` [1], the simulator framework that is likely to be prominent in the near/mid term, the mainstream available models for wireless channels [6, 7] are based on the usage of *BER* curves, as a function of the *RSS*. Although they perform quite well in terms of *FER* and *throughput*, they are not able to appropriately reflect the *bursty* nature of real indoor wireless channels. There have been a number of proposals to overcome the limitations of these *legacy* wireless channel models. Papanastasiou *et al.* [18] challenge the suitability of this particular simulator and other alternatives (i.e. *QualNet*), advocating that, despite the upper layers are accurately implemented, little attention has been usually paid to the physical behavior, taking many abstractions and simplifications. Hence, they propose a clean-slate alternative to the legacy models supported by `ns-3`. The authors present a fully-fledged bit-level physical layer emulator tuned for the *Orthogonal Frequency Division Multiplex (OFDM)* based IEEE 802.11 transmissions. Although this approach could get closer to the real behavior of wireless channels, its intrinsic complexity penalizes the time required to perform the simulations, which might be prohibitive over scenarios with a large number of deployed nodes and traffic flows.

Last, but not least, Al-Bado *et al.* [19] used an extensive empirical campaign over a real indoor scenario to propose a new `ns-3` wireless channel model, tailored from the Frame Detection Rate (FDR), as well as the capture and interference patterns observed over the real measurements, for different physical rates (i.e. IEEE 801.11g at 6, 24 and 54 Mbps). Although they share the same *empirical* approach that we exploited to configure the two channel models, they focus on other aspects, rather than the channel *bursty* behavior. In particular they pay attention to both the interference and the capture effects and their model is tightly related to their testbed. It should not be to complicated including those effects (especially the interference) within *BEAR* and this might be an interesting point to tackle in our future research.

7 Conclusions

In this work we presented two different wireless channel models, tailored from the results obtained over a real in-

door testbed, following two complementary approaches: the first one (*BEAR*) estimates the *RSS* by means of an auto-regressive filter, whilst the second one (*HMP*) "discretizes" the error response of the wireless channel in a finite number of states, building a hidden Markov chain. The cornerstone of both models is that they aim at reflecting the *bursty* nature that characterizes real indoor wireless channels, whose memory behavior is usually disregarded by the vast majority of simulators, usually providing a rather predictable behavior.

We have carried out an extensive simulation workout to characterize the behavior of these models using a naive transport-layer protocol, UDP, to study the raw performance leveraged by the lower layers. Under these assumptions, we have observed that, although the *Default* model provides acceptable results in terms of the average *FER* and *throughput*, it exhibits an almost deterministic behavior. This demonstrates that this sort of models fail to capture the memory effect, and thus every frame reception can be considered as an independent event. On the other hand, *HMP* and *BEAR*, besides being able to mimic the average performance (*FER* and *throughput*), they yield a much broader range of outputs. Regarding their *bursty* behavior, we have seen that both of them adequately reflect the results observed during the real measurements, capturing as well the expected behavior in terms of *EFBs*.

One of the most obvious limitations of the legacy *HMP* models is their lack of dependency to the received signal quality, since the corresponding matrices (A and B) are chosen *offline* (before simulation starts). To overcome this limitation, we have added the possibility to dynamically change the *HMP* coefficients according to the distance between source and destination. We have taken the *BEAR* performance to tailor the thresholds between which the *distance/HMP coefficients* bindings are done. Regarding the obtained results we can assert that, although we are limited to a low finite number of configurations, the broad range of behaviors brings about the possibility of providing a dynamic-range model.

After the analysis of the lower layers (and the channel itself) raw performance, we have assessed the impact that these *bursty* channel models have over connection-oriented protocols, in particular TCP. Its performance is severely damaged, since its intrinsic congestion control mechanisms are extremely sensitive to consecutive segment losses, thus jeopardizing the overall throughput. In this study, we have asserted the almost null variability and *bursty* effect provided by the *Default* model, making it completely unsuitable to study the performance of TCP-based applications over indoor wireless channels. On the contrary, with both *BEAR* and *HMP*, the simulation results showed a broad range of outputs, as well as an appropriate memory behavior. Besides, these models are also able to reflect the harmful effect brought about by long *EFBs*, leading to remarkable idle times at the transmitter.

Finally, we have also analyzed the computational complexity of the three different channels. The *Default* model shows the lowest simulation time, but this does not compensate its lack of accuracy. On the other hand, we found out that *BEAR* requires the longest time, standing the *HMP* model as an intermediate solution which, together with its rather realistic performance, shows a reasonable complexity, making it attractive on scenarios with a large number of nodes.

Regarding the future work, the most straightforward aspect to be mentioned is the fact that the analyzed channel models can be exploited to evaluate various techniques, algorithms and protocols, including cross-layer techniques. In particular, we plan to use them so as to study the performance of Network Coding techniques, focusing on the impact of errors bursts over the performance gain that those techniques might bring about. Furthermore, there are still a number of open issues that could be tackled in the future in order to enhance and extend the functionalities of the proposed channel models, as described below.

- First we would like to adapt our models to more recent IEEE 802.11 physical specifications (i.e. g/n/ac). In order to be able to appropriately tune their different configuration parameters, such as *HMP's* matrices and *BEAR's AR* filter coefficients and logistic functions, we first need to carry out a measurement campaign over a real indoor scenario for each of the IEEE 802.11 recommendations to be mimicked.
- Another interesting aspect would be to evaluate the performance of these models over different conditions, such as number of nodes or traffic patterns.
- It is worth highlighting that our models disregard the interference contribution produced by contention (and collisions) with other IEEE 802.11 stations, coexisting 2.4 GHz radio technologies over the coverage area, etc. Actually, they rely on the legacy interference model helpers provided by the simulator, whose operation is currently under development in [20]. The interaction between these physical-level solutions shall be addressed in order to create a holistic solution in the future.

Last, but not least, all the information regarding the two proposed models (both *HMP* and *BEAR*) have been made available to the scientific community [21]. We strongly encourage the interested readers to download the code, assess the suitability of the models, and use them for their own research, as this would help us to improve them by means of an active feedback.

Acknowledgements

The authors would like to express their gratitude to the Spanish government for its funding in the project "Connectivity as a Service: Access for the Internet of the Future", COSAIF (TEC2012-38574-C02-01). Besides, we would like to thank as well Mr. Juan Ramón Santana for his help during the development of this work. This work has been partially presented in part in the following conferences: IEEE International Conference on Wireless and Mobile Computing, Networking and Communications (*WiMob 2012*), workshop on ns-3 (*WNS3 2013*), within the International ICST Conference on Simulation Tools and Techniques (*Simutools 2013*) and IEEE *Wireless Days* 2013.

References

1. "The ns-3 network simulator," http://www.nsnam.org/.
2. R. Agüero, M. García-Arranz, and L. Muñoz, "Accurate simulation of 802.11 indoor links: a bursty channel model based on real measurements," *EURASIP J. Wirel. Commun. Netw.*, vol. 2010, pp. 16:1–16:12, April 2010.
3. L. E. Baum, "An inequality and associated maximization technique in statistical estimation for probabilistic functions of Markov processes," in *Inequalities III: Proceedings of the Third Symposium on Inequalities*, O. Shisha, Ed. University of California, Los Angeles: Academic Press, 1972, pp. 1–8.
4. K. Cardoso and J. De Rezende, "Accurate hidden markov modeling of packet losses in indoor 802.11 networks," *IEEE Communications Letters*, vol. 13, no. 6, pp. 417 –419, June 2009.
5. D. Gómez, R. Agüero, M. García-Arranz, and L. M. noz, "Replication of the Bursty Behavior of Indoor WLAN Channels," in *Workshop on NS3 (WNS3)*, Cannes, France, 2013. [Online]. Available: http://hal.inria.fr/hal-00781591
6. M. Lacage and T. R. Henderson, "Yet another network simulator," in *Proceeding from the 2006 workshop on ns-2: the IP network simulator*, ser. WNS2 '06. New York, NY, USA: ACM, 2006. [Online]. Available: http://doi.acm.org/10.1145/1190455.1190467
7. G. Pei and T. Henderson, "Validation of ns-3 802.11b PHY model," May 2009, http://www.nsnam.org/~pei/80211b.pdf.
8. A. Vlavianos, L. Law, I. Broustis, S. Krishnamurthy, and M. Faloutsos, "Assessing link quality in IEEE 802.11 wireless networks: Which is the right metric?" in *Personal, Indoor and Mobile Radio Communications, 2008. PIMRC 2008. IEEE 19th International Symposium on*, sept. 2008, pp. 1 –6.
9. D. Lertpratchya, G. F. Riley, and D. M. Blough, "Simulating Frame-Level Bursty Links in Wireless Networks," in *Proceedings of the 7th International Conference on Simulation Tools and Techniques, SIMUTOOLS'14*, March 2014.
10. D. Eckhardt and P. Steenkiste, "Measurement and analysis of the error characteristics of an in-building wireless network," *SIGCOMM Comput. Commun. Rev.*, vol. 26, no. 4, pp. 243–254, Aug. 1996. [Online]. Available: http://doi.acm.org/10.1145/248157.248178
11. G. Nguyen, B. Noble, R. Katz, and M. Satyanarayanan, "A trace-based approach for modeling wireless channel behavior," in *Simulation Conference, 1996. Proceedings. Winter*, 1996, pp. 597 – 604.
12. P. Ikkurthy and M. Labrador, "Characterization of MPEG-4 traffic over IEEE 802.11b wireless LANs," in *Local Computer Networks, 2002. Proceedings. LCN 2002. 27th Annual IEEE Conference on*, nov. 2002, pp. 421 – 427.
13. G. Convertino, S. Oliva, F. Sigona, and L. Anchora, "An Adaptive FEC Scheme to Reduce Bursty Losses in a 802.11 Network," in *IEEE Global Telecommunications Conference, 2006. GLOBECOM '06*, 27 2006-Dec. 1 2006, pp. 1 –6.
14. V. R. Gandikota, B. R. Tamma, and C. S. R. Murthy, "Adaptive FEC-based packet loss resilience scheme for supporting voice communication over Ad hoc Wireless Networks," *IEEE Transactions on Mobile Computing*, vol. 7, no. 10, pp. 1184–1199, Oct. 2008. [Online]. Available: http://dx.doi.org/10.1109/TMC.2008.42
15. P. Barsocchi, G. Oligeri, and F. Potorti, "Measurement-based frame error model for simulating outdoor Wi-Fi networks," *IEEE Transactions on Wireless Communications*, vol. 8, no. 3, pp. 1154 –1158, March 2009.
16. W. Turin and R. Van Nobelen, "Hidden markov modeling of flat fading channels," *Selected Areas in Communications, IEEE Journal on*, vol. 16, no. 9, pp. 1809–1817, Dec 1998.
17. W. Zhu and J. Garcia-Frias, "Stochastic context-free grammars and hidden markov models for modeling of bursty channels," *Vehicular Technology, IEEE Transactions on*, vol. 53, no. 3, pp. 666–676, May 2004.
18. S. Papanastasiou, J. Mittag, E. Strom, and H. Hartenstein, "Bridging the gap between physical layer emulation and network simulation," in *IEEE Wireless Communications and Networking Conference (WCNC)*, 2010, pp. 1–6.
19. M. Al-Bado, C. Sengul, and R. Merz, "What details are needed for wireless simulations? - A study of a site-specific indoor wireless model." in *INFOCOM*. IEEE, 2012, pp. 289–297.
20. "802.11b Interference Modeling in NS-3 Simulator," http://www.ee.washington.edu/research/funlab/802_11_b_intf_model/index.html.
21. D. Gómez and R. Agüero, "GitHub's repository for HMP and BEAR IEEE 802.11b indoor channel models (source code and documentation)," https://github.com/dgomezunican/ns3-wifi-memory-channel.