HMM-Based Monitoring of Packet Channels

Pierluigi Salvo Rossi[1], Francesco Palmieri[2], and Giulio Iannello[1,3]

[1] Dipartimento di Informatica e Sistemistica, Universotá di Napoli "Federico II"
{salvoros,iannello}@unina.it
[2] Dipartimento di Ingegneria dell'Informazione, Seconda Universitá di Napoli
frapalmi@unina.it
[3] Università Campus Bio-Medico di Roma

Abstract. Performance of real-time applications on network communication channels are strongly related to losses and temporal delays. Several studies showed that these network features may be correlated and exhibit a certain degree of memory such as bursty losses and delays. The memory and the statistical dependence between losses and temporal delays suggest that the channel may be well modelled by a Hidden Markov Model (HMM) with appropriate hidden variables that capture the current state of the network. In this paper we discuss on the effectiveness of using an HMM to model jointly loss and delay behavior of real communication channel. Excellent performance in modelling typical channel behavior in a set of real packet links are observed. The system parameters are found via a modified version of the EM algorithm. Hidden state analysis shows how the state variables characterize channel dynamics. State-sequence estimation is obtained by use of the Viterbi algorithm. Real-time modelling of the channel is the first step to implement adaptive communication strategies.

1 Introduction

Gilbert and Elliott works [1][2] on modelling burst-error channels for bit transmission showed how a simple 2-state Hidden Markov Model (HMM) was effective in characterizing some real communication channels. As in the case of bit-transmission channels, end-to-end packet channels show bursty loss behavior. Jiang and Schulzrinne [10] investigated lossy behavior of packet channels finding that a Markov model is not able to describe appropriately channels inter-loss behavior. They also found that delays manifest temporal dependency, i.e. they should not be assumed to be a memoryless phenomenon. Salamatian and Vaton [11] found that an HMM trained with experimental data seems to capture channel loss behavior and found that an HMM with 2 to 4 hidden states fits well experimental data. Liu, Matta and Crovella [12] used an HMM-based loss-delay modelling in the contest of TCP traffic in order to infer loss nature in hybrid wired/wireless environments. They found that such a kind of modelling can be used to control TCP congestion avoidance mechanism. Similar works have been done by Zorzi [7] on wireless fading links.

Z. Mammeri and P. Lorenz (Eds.): HSNMC 2004, LNCS 3079, pp. 144–154, 2004.

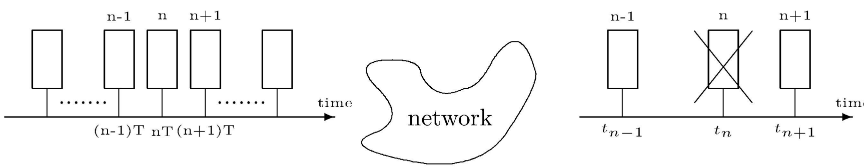


Fig. 1. End-to-end packet channel.

These works suggested us that a Bayesian model, or an HMM, should be effective in capturing the dynamic behavior of losses and delays on end-to-end packet channels [13][14]. The definition of such a model is highly desirable for designing and evaluating coding strategies. Furthermore, the possibility of learning on-line the model parameters opens the way to design efficient content-adaptive services.

In this paper we propose a comprehensive model that jointly describes losses and delays. The model is an HMM trained with an adapted version of the EM algorithm to capture channel dynamics. Then we discuss about the meaning of the hidden states of the trained model. Hidden states of the model represent different working conditions of the channel. Current state knowledge and prediction of state transitions enable a powerful characterization of future channel behavior, which could be used to implement content-adaptive strategies for coding (e.g. Multiple Description Coding) and scheduling (e.g. traffic shaping).

2 The Model

The model we are referring to is shown in Fig. 1. A periodic source transmits a packet of size N_b bits every T seconds, i.e. at rate $R = N_b/T$ *bits/s*. The network randomly cancels and delays packets according to current congestion. Transmitted packets are numbered, $n = 1, 2, \ldots$; t_n and τ_n are the arrival time and the accumulated delay of the *n-th* packet respectively, i.e. $\tau_n = t_n - nT$.

Memory and correlation presence in losses and delays dynamic of the communication channels suggest the introduction of a hidden state variable, namely x_n, carrying information about link congestion. An observable variable, y_n, is introduced in order to describe jointly losses and delays. Let us denote $\{s_i\}_{i\in\{1,2,\ldots,N\}}$ the possible states of the channel, and let define

$$y_n = \begin{cases} \tau_n & if\ packet\ was\ NOT\ lost \\ -1 & if\ packet\ was\ lost \end{cases} . \tag{1}$$

The state and the observable variables are related according to the HMM structure showed in Fig. 2. The channel dynamics are characterized by
$\Lambda = \{\mathbf{A}, \mathbf{p}, f_1(\tau), f_2(\tau), \ldots, f_N(\tau)\}$, where $\mathbf{A} = [a_{ij}]_{i,j=1}^{N}$ is the state transition matrix,i.e.

$$a_{ij} = Pr(x_{n+1} = s_j | x_n = s_i)\ _{i,j\in\{1,2,\ldots,N\}}\ , \tag{2}$$

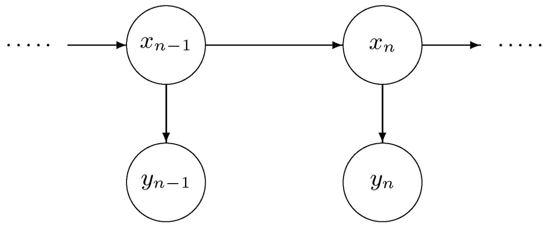


Fig. 2. Hidden Markov Model.

while $\mathbf{p} = [p_i]_{i=1}^N$ is the loss probability vector, and $\{f_i\}_{i=1}^N$ are the delay conditional pdfs,i.e.

$$\begin{cases} p_i = Pr(y_n = \tau_n | x_n = s_i) \\ 1 - p_i = Pr(y_n = -1 | x_n = s_i) \end{cases}_{i \in \{1,2,\ldots,N\}} , \tag{3}$$

$$Pr(\tau_n > t | x_n = s_i, y_n = \tau_n) = \int_t^{+\infty} f_i(\tau) d\tau \ . \tag{4}$$

The hybrid random variable y_n is characterized given $\{x_n = s_i\}$ by the following conditional pdf,

$$b_i(t) = p_i f_i(t) + (1 - p_i)\delta(t + 1) \ . \tag{5}$$

If $\pi = [\pi_i]_{i=1}^N$ is the steady-state probability distribution,

$$\pi_i = \lim_{n \to \infty} \{Pr(x_n = s_i)\}_{\ i \in \{1,2,\ldots,N\}} , \tag{6}$$

the average loss probability and the average delay of the model are

$$P_{loss} = \sum_{i=1}^N \pi_i (1 - p_i) \ ; \ D_{mean} = \sum_{i=1}^N \pi_i d_i \ , \tag{7}$$

where $d_i = \gamma_i \vartheta_i$ is the conditional-average delay.

Parameters Λ for the model are estimated by the Forward-Backward algorithm [3][4][5]. For HMMs it is a form of the Expectation-Maximization (EM) algorithm [8], an optimization procedure searching for the set of parameters according to maximization of likelihood of an observable sequence. Given a training sequence $\mathbf{y} = [y_i]_{i=1}^K$, compute iteratively the following equations,

$$\hat{a}_{ij} = \frac{\sum_{k=1}^{K-1} \alpha_k(i) a_{ij} b_j(y_{k+1}) \beta_{k+1}(j)}{\sum_{k=1}^{K-1} \alpha_k(i) \beta_k(i)} \ _{i,j \in \{1,2,\ldots,N\}} , \tag{8}$$

$$\hat{p}_i = \frac{\sum_{k=1}^{K} \rho_k(i) \beta_k(i)}{\sum_{k=1}^{K-1} \alpha_k(i) \beta_k(i)} \ _{i \in \{1,2,\ldots,N\}} , \tag{9}$$

$$\hat{\mu}_i = \frac{\sum_{k=1}^{K} \rho_k(i) \beta_k(i) y_k}{\sum_{k=1}^{K-1} \rho_k(i) \beta_k(i)} \ _{i \in \{1,2,\ldots,N\}} , \tag{10}$$

$$\hat{\sigma}_i^2 = \frac{\sum_{k=1}^{K} \rho_k(i)\beta_k(i)(y_k - \mu_i)^2}{\sum_{k=1}^{K-1} \rho_k(i)\beta_k(i)} \quad {}_{i\in\{1,2,\dots,N\}} , \tag{11}$$

where

$$\alpha_k(j) = \sum_{i=1}^{N} \alpha_{k-1}(i) a_{ij} b_j(y_k) \quad {}_{\substack{k\in\{2,3,\dots,K\}\\ j\in\{1,2,\dots,N\}}} , \tag{12}$$

$$\beta_k(i) = \sum_{j=1}^{N} a_{ij} b_j(y_{k+1}) \beta_{k+1}(j) \quad {}_{\substack{k\in\{1,2,\dots,K-1\}\\ i\in\{1,2,\dots,N\}}} , \tag{13}$$

are the forward and backward partial likelihood, and where

$$\rho_k(j) = \sum_{i=1}^{N} \alpha_{k-1}(i) a_{ij} p_j \left. \frac{\partial b_j(t)}{\partial p_j} \right|_{t=y_k} \quad {}_{\substack{k\in\{2,3,\dots,K\}\\ j\in\{1,2,\dots,N\}}} . \tag{14}$$

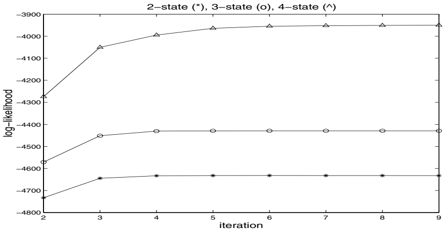
Our choice of conditional pdf's for modelling delays is a classical Gamma distribution, as suggested by several works [6][9],

$$f_i(t) = \frac{(t/\vartheta_i)^{\gamma_i - 1} e^{-(t/\vartheta_i)}}{\vartheta_i \Gamma(\gamma_i)} u(t) . \tag{15}$$

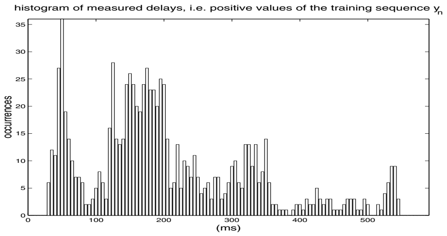
3 Hidden States Analysis

Measures of losses and delays have been performed on real Internet channels using the software Distributed-Internet Traffic Generator (D-ITG) [15][18]. D-ITG was used to obtain loss-delay sequences of UDP traffic. A little portion of the sequences was used as a training sequence to learn model parameters. Performance of the trained model have been tested on the remaining portions of the sequences. The model showed good modelling properties, i.e. the training procedure well captures loss-delay statistics of the channel and the trained model exhibits generalization capacity. Fig. 3 and Table 1 summarize the results we obtained in terms of channel modelling. More specifically, they show the results concerning a typical data set: the log-likelihood trend during the learning procedure (Fig. 3(a)); the histogram of delays in the training sequence (Fig. 3(b)); the continuous term of the pdf (5) of the observable variable before and after the learning procedure (Fig. 3(c)); the log-likelihood of the models before and after the learning procedure evaluated for sequences not used during the training (Fig. 3(d)); the average loss probability and average delay of the model (7) before and after the learning procedure (Table 1). More details can be found in [13][14].

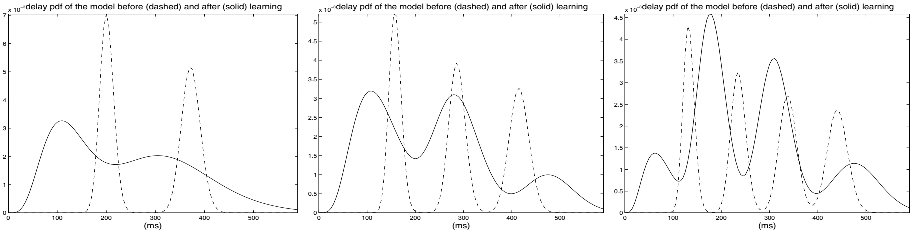
To verify how the hidden state variable x_n captures the current channel congestion state, a Viterbi algorithm [5] has been applied to the training sequence. We remind that the Viterbi algorithm furnishes the most likely state sequence $\mathbf{x} = (x_1, x_2, \dots, x_K)$, i.e. the state sequence such that the *a posteriori* probability $Pr(\mathbf{x}/\mathbf{y}, \Lambda)$ is maximum. Fig. 4 shows the temporal evolution of the training sequence $\mathbf{y} = (y_1, y_2, \dots, y_K)$ and the state sequences obtained by use



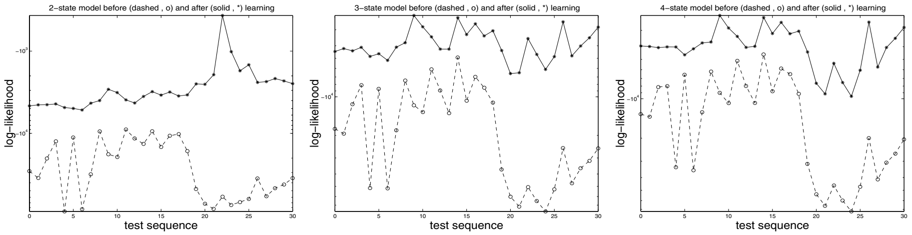
(a) log-likelihood trend in the learning procedure.



(b) Histogram of measured delays, i.e. of the positive value of training sequence y_n.



(c) Continuous term of pdf before (dashed) and after (solid) learning procedure, for 2−,3− and 4− state models.



(d) Log-likelihood before (dashed, ∘) and after (solid, ∗) learning procedure for the test set, for 2−,3− and 4− state models.

Fig. 3. Joint loss-delay modeling using a HMM.

of the Viterbi algorithm on the previous 2−, 3−, and 4−state trained models. The trained models give state sequences that look to capture loss-delay network behavior quite well.

Now we want, with reference to Fig. 4, to furnish a qualitative interpretation of states automatically found. If $s_i^{(k)}$ is the *i-th* state of the k−state model, then

- the 2−state model emerges to distinguish 2 situations: $s_1^{(2)}$ for lower delays and fewer losses, $s_2^{(2)}$ for large delays and many losses, (Fig. 4(b));
- the 3−state model seems to use its states to distinguish the same 2 situations as the previous model with $s_1^{(3)}$ resembling $s_1^{(2)}$, while $s_2^{(2)}$ is now split in 2 states: $s_2^{(3)}$ for many losses and $s_3^{(3)}$ describing very high-delays situation, (Fig. 4(c));

Table 1. Loss probability and average delay before and after the learning procedure compared to statistics of the training sequence.

	P_{loss}	D_{mean} (ms)
training sequence	0.117	206.10
starting model	0.500	286.90
$2-$ *state model*	0.132	246.95
$3-$ *state model*	0.132	253.64
$4-$ *state model*	0.133	272.42

Table 2. Steady-state probabilities, State-Conditioned loss probabilities and State-Conditioned average delay before and after the learning procedure for a 2−state model.

	starting	*trained*
π_1	0.500	0.450
$1-p_1$	0.500	0.085
d_1 (ms)	201.14	131.14
π_2	0.500	0.550
$1-p_2$	0.500	0.170
d_2 (ms)	372.65	341.69

– the 4−state model distinguishes the same 2 situations as the 2−state model, but now each one of them is described with 2 states: $s_3^{(4)}$ and $s_4^{(4)}$ respectively corresponding to $s_2^{(3)}$ and $s_3^{(3)}$ for the high-delays/many-losses situation, while for the low-delays situation $s_1^{(2)}$ or $s_1^{(3)}$ too is split: $s_1^{(4)}$ describes low-delays and losses and $s_2^{(4)}$ describes low delays and very few losses, (Fig. 4(d)).

Fig. 5 synthesizes the correspondences we noted about states of the trained models. Let we denote $\pi_i^{(k)}$, $1-p_i^{(k)}$ and $d_i^{(k)}$, the steady-state probability, the loss probability and the average delay of the state $s_i^{(k)}$, respectively. Let $P_{loss,a}^{(k)}$, $D_{mean,a}^{(k)}$ and $P_{loss,b}^{(k)}$, $D_{mean,b}^{(k)}$ be the loss probabilities and the average delays in the two situations previously evinced (low delays or few losses, and larger delays or many losses) for the $k-$state model; $\pi_a^{(k)}$ and $\pi_b^{(k)}$ are the corresponding steady probabilities. From Tables 2, 3, and 4, the following equalities strengthen the effectiveness of the various models, confirming the correspondence, previously described, among hidden states as well as the significance of the state variable x_n.

$$\begin{cases} \pi_a^{(2)} = \pi_1^{(2)} = 0.450 \\ \pi_a^{(3)} = \pi_1^{(3)} = 0.441 \\ \pi_a^{(4)} = \pi_1^{(4)} + \pi_2^{(4)} = 0.467 \end{cases} , \begin{cases} \pi_b^{(2)} = \pi_2^{(2)} = 0.550 \\ \pi_b^{(3)} = \pi_2^{(3)} + \pi_3^{(3)} = 0.559 \\ \pi_b^{(4)} = \pi_3^{(4)} + \pi_4^{(4)} = 0.533 \end{cases} , \tag{16}$$

Table 3. Steady-state probabilities, State-Conditioned loss probabilities and State-Conditioned average delay before and after the learning procedure for a 3−state model.

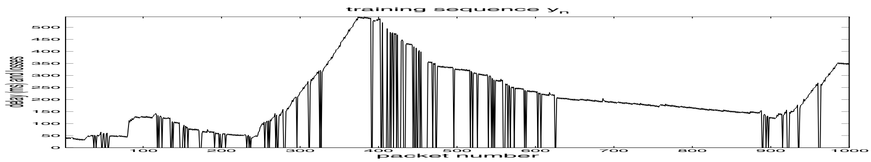
	starting	*trained*
π_1	0.333	0.441
$1-p_1$	0.500	0.085
d_1 (ms)	158.27	130.72
π_2	0.333	0.383
$1-p_2$	0.500	0.111
d_2 (ms)	286.90	290.62
π_3	0.333	0.176
$1-p_3$	0.500	0.298
d_3 (ms)	415.53	481.13

Table 4. Steady-state probabilities, State-Conditioned loss probabilities and State-Conditioned average delay before and after the learning procedure for a 4−state model.

	starting	*trained*
π_1	0.250	0.128
$1-p_1$	0.500	0.139
d_1 (ms)	132.54	77.98
π_2	0.250	0.339
$1-p_2$	0.500	0.045
d_2 (ms)	235.45	181.98
π_3	0.250	0.330
$1-p_3$	0.500	0.120
d_3 (ms)	338.35	312.53
π_4	0.250	0.203
$1-p_4$	0.500	0.297
d_4 (ms)	441.26	480.87

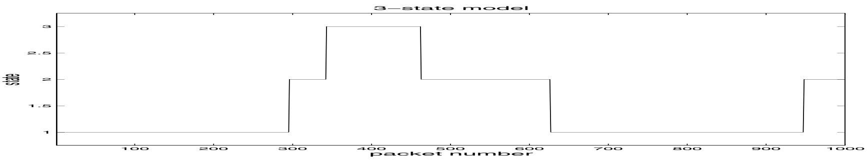
$$\begin{cases} P^{(2)}_{loss,a} = \pi^{(2)}_1(1-p^{(2)}_1) = 0.038 \\ P^{(3)}_{loss,a} = \pi^{(3)}_1(1-p^{(3)}_1) = 0.037 \\ P^{(4)}_{loss,a} = \pi^{(4)}_1(1-p^{(4)}_1) + \pi^{(4)}_2(1-p^{(4)}_2) = 0.033 \end{cases} \begin{cases} P^{(2)}_{loss,b} = \pi^{(2)}_2(1-p^{(2)}_2) = 0.094 \\ P^{(3)}_{loss,b} = \pi^{(3)}_2(1-p^{(3)}_2) + \pi^{(3)}_3(1-p^{(3)}_3) = 0.095 \\ P^{(4)}_{loss,b} = \pi^{(4)}_3(1-p^{(4)}_3) + \pi^{(4)}_4(1-p^{(4)}_4) = 0.100 \end{cases}, \tag{17}$$

$$\begin{cases} D^{(2)}_{mean,a} = d^{(2)}_1 = 131.14\,ms \\ D^{(3)}_{mean,a} = d^{(3)}_1 = 130.72\,ms \\ D^{(4)}_{mean,a} = \frac{\pi^{(4)}_1}{\pi^{(4)}_1+\pi^{(4)}_2} d^{(4)}_1 + \frac{\pi^{(4)}_2}{\pi^{(4)}_1+\pi^{(4)}_2} d^{(4)}_2 = 153.52\,ms \end{cases} \begin{cases} D^{(2)}_{mean,b} = d^{(2)}_2 = 341.69\,ms \\ D^{(3)}_{mean,b} = \frac{\pi^{(3)}_2}{\pi^{(3)}_2+\pi^{(3)}_3} d^{(3)}_2 + \frac{\pi^{(3)}_3}{\pi^{(3)}_2+\pi^{(3)}_3} d^{(3)}_3 = 350.65\,ms \\ D^{(4)}_{mean,b} = \frac{\pi^{(4)}_3}{\pi^{(4)}_3+\pi^{(4)}_4} d^{(4)}_3 + \frac{\pi^{(4)}_4}{\pi^{(4)}_3+\pi^{(4)}_4} d^{(4)}_4 = 376.55\,ms \end{cases}, \tag{18}$$

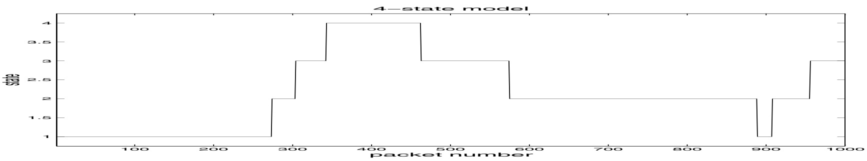


(a) Training sequence y_n.

(b) State-sequence estimation for a 2−state model.



(c) State-sequence estimation for a 3−state model.



(d) State-sequence estimation for a 4−state model.

Fig. 4. Example of state-sequence estimation based on Viterbi algorithm.

where

$$\begin{aligned} D^{(k)}_{mean,situation} &= \textstyle\sum_{i \in situation} Pr(s_i^{(k)} | situation) d_i^{(k)} \\ &= \textstyle\sum_{i \in situation} \frac{\pi_i^{(k)}}{\pi^{(k)}_{situation}} d_i^{(k)} \ , \end{aligned} \tag{19}$$

and where $situation = \{a, b\}$ and $k = \{2, 3, 4\}$.

Moreover, Figs. 6 and 7 show the hidden states of the starting and trained models on a test sequence. Comparing state-sequences from starting and trained models it can be noted how they assume a very different behavior. In case of starting models, hidden states are strictly dependent from instantaneous behavior of the channel, showing a rapidly oscillating trend; while hidden states for the trained models seem to capture well the state of the network on a larger time-scale, exhibiting a more stable trend.

All this rises up the following considerations. A state is associated to a particular loss probability (depending on parameters $\mathbf{p}$), to a particular average delay (depending on parameters $\{\gamma, \vartheta\}$), to a particular duration in the state

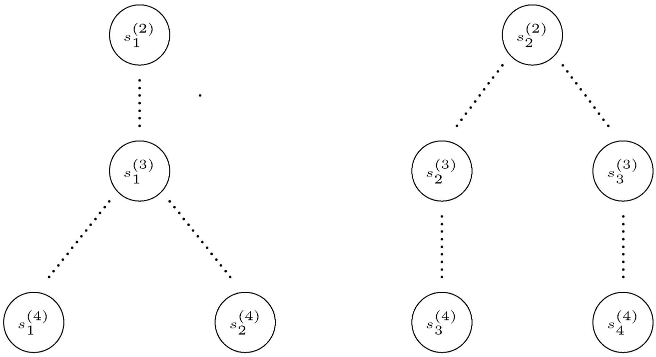


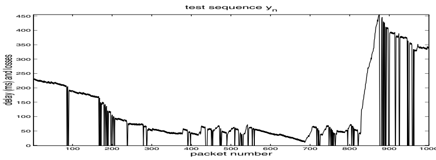
Fig. 5. Correspondences among states of the previous trained models.

itself (depending on parameters **A**) and to a particular transition probability into another state (depending on parameters **A**).

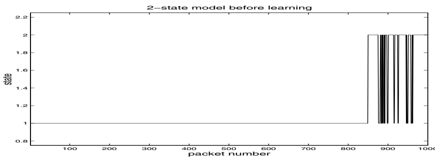
Stable behavior of the states of a trained model suggests to investigate on the possibility of supporting adaptive services mechanisms. Such on-line modelling features can be exploited to support device independent services as defined by the corresponding W3C working group [17], according to the scheme showed in Fig. 8. Such a scheme needs losses and delays to be monitored to train an HMM-model like the one previously described. Then state-sequence estimation is used to foresee the short-term future behavior of the channel. This information could be sent back to the sender in order to adapt transmission. This strategy would clearly require that sufficient stationarity of the channel exists to make adaptive coding strategies worth the effort. We believe this possible in many practical situations and we are currently pursuing such an effort. When adaptive coding is not possible, or not worth the effort, good channel modelling can be very useful to evaluate performance of existing coders.

4 Conclusion

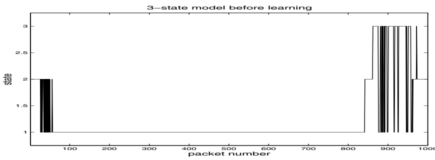
In this paper we presented an HMM used to model end-to-end packet channels behavior capturing jointly loss and delay characteristics. A training procedure to learn model parameters, based on the EM algorithm, was derived. Tests ran on real packet links showed very encouraging preliminary results. Trained models exhibit very good generalization capacities. We also discussed on the significance of hidden states automatically found by the training algorithm. We showed how the states can be associated to particular congestion levels of the network. Monitoring or even prediction of hidden states should be very effective in implementation of content-adaptive communication strategies. Future works will be directed towards model improvements and development of content-adaptive strategies based on hidden state knowledge.



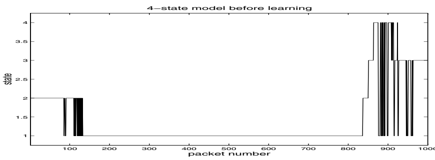
(a) Example of test sequence.



(b) State-sequence estimation for a 2−state model before learning.

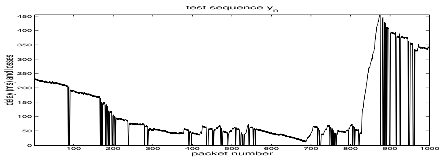


(c) State-sequence estimation for a 3−state model before learning.

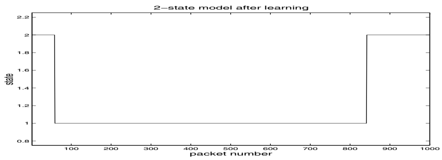


(d) State-sequence estimation for a 4−state model before learning.

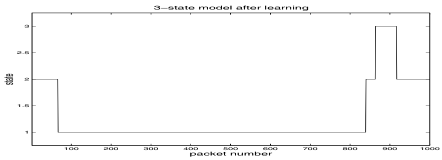
Fig. 6. Significance of the state variable before learning procedure.



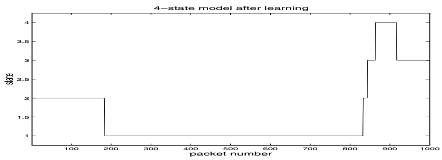
(a) Example of test sequence.



(b) State-sequence estimation for a 2−state model after learning.



(c) State-sequence estimation for a 3−state model after learning.



(d) State-sequence estimation for a 4−state model after learning.

Fig. 7. Significance of the state variable after learning procedure.

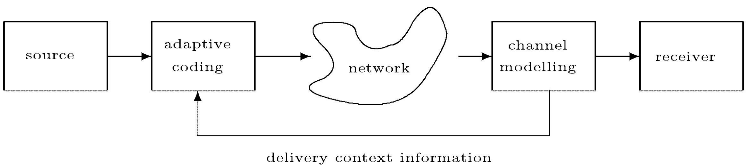


Fig. 8. Scheme for an adaptive communication protocol using HMM-based channel modelling.

Acknowledgement

This work has been carried out partially under the financial support of the Ministero dell'Istruzione, dell'Università e della Ricerca (MIUR) in the framework of the FIRB Project “Middleware for advanced services over large-scale, wired-wireless distributed systems (WEB-MINDS)”.

References

1. E.N. Gilbert, *Capacity of a burst-noise channel*. Bell System Technical Journal, Vol. 39, pp. 1253-1265, Sept. 1960.
2. E.O. Elliott, *Estimates of error-rate for codes on burst-noise channels*. Bell System Technical Journal, Vol. 42, pp. 1977-1997, Sept. 1963.
3. L.A. Liporace, *Maximum Likelihood Estimation for Multivariate Observations of Markov Sources*. IEEE Transactions on Information Theory, Vol. IT-28(5), pp. 729-734, Sept. 1982.
4. B.H. Juang, S.E. Levinson, and M.M. Sondhi, *Maximum Likelihood Estimation for Multivariate Mixture Observations of Markov Chains*. IEEE Transactions on Information Theory, Vol. IT-32(2), pp. 307-309, Mar. 1986.
5. L.R. Rabiner, *A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition*. Proceedings of the IEEE, Vol. 77(2), pp. 257-285, Feb. 1989.
6. J.C. Bolot, *Characterizing End-to-End Packet Delay and Loss in the Internet*. Journal of High-Speed Networks, Vol. 2(3), pp. 305-323, Dec. 1993.
7. M. Zorzi, R.R. Rao, and L.B. Milstein, *On the Accuracy of a First-Order Markov Model for Data Block Transmission on Fading Channels*. IEEE International Conference on Personal Communications, pp. 211-215, Nov. 1995.
8. J.A. Bilmes, *A Gentle Tutorial of the EM Algorithm and its Application to Parameter Estimation for Gaussian Mixture and Hidden Markov Models*. Technical Report, ICSI-TR-97-021, University of Berkeley, CA, 1998.
9. V. Paxson, *End-to-End Internet Packet Dynamics*. IEEE Transactions on Networking, Vol. 7(3), pp. 277-292, June 1999.
10. W. Jiang, and H. Schulzrinne, *Modeling of Packet Loss and Delay and Their Effect on Real-Time Mulrimedia Service Quality*. International Workshop on Network and Operating System Support for Digital Audio and Video, June 2000.
11. K. Salamatian, and S. Vaton, *Hidden Markov Modeling for Network Communication Channels*. ACM Sigmetrics/Performance, Vol. 29, pp. 92-101, 2001.
12. J. Liu, I. Matta, and M. Crovella *End-to-End Inference of Loss Nature in a Hybrid Wired/Wireless Environment*. Technical Report, Boston University, MA, Mar. 2002.
13. P. Salvo Rossi, G. Romano, F. Palmieri, and G. Iannello, *Bayesian Modelling for Packet Channels*. Italian Workshop on Neural Nets, Springer Verlag, pp. 285-292, June 2003.
14. P. Salvo Rossi, G. Romano, F. Palmieri, and G. Iannello, *A Hidden Markov Model for Internet Channels*. IEEE International Symposium on Signal Processing and Information Technology, Dec. 2003.
15. S. Avallone, A. Pescapé, and G. Ventre, *Analysis and Experimentation of Internet Traffic Generator*. International Conference on Next Generation Teletraffic and Wired/Wireless Advanced Networking, Feb. 2004.
16. P. Salvo Rossi, G. Romano, F. Palmieri, and G. Iannello, *Inteleaving for Packet Channels*. Conference on Information Sciences and Systems, pp. 1560-1564, Mar. 2004.
17. W3C - Device Independence Working Group. `http://www.w3c.org/2001/di/`
18. Distributed Internet Traffic Generator. `http://www.grid.unina.it/software/ITG/`