# Modeling P2P-TV Traffic*

**Maria Antonieta Garcia**[1], **Ana Paula Couto da Silva**[2], **Michela Meo**[1]

[1]Dipartimento di Elettronica - Politecnico di Torino - Torino, Italy

antonieta@tlc.polito.it,michela.meo@tlc.polito.it

[2]Computer Science Department - Federal University of Juiz de Fora - Juiz de Fora, Brazil

anapaula.silva@ufjf.edu.br

***Abstract.*** *The increasing success of P2P-TV applications calls for the need of new traffic models that can effectively represent the traffic generated by these applications. In this paper, we consider PPLive; we study and model the traffic generated by PPLive clients. From the analysis of some real traffic traces, we recognize that the peer may follow three typical behaviors, depending on the peer degree of contribution to the video content distribution. We then propose two simple models of the data received or transmitted by a peer: i) the* Memoryless *model that fits the distribution of traffic exchanged during short time windows; ii) the* Hidden-Markov *model that introduces also some memory in the traffic generation process so that the autocorrelation function typical of real traces can be matched. The accuracy of the models for all three classes of peer behavior is validated by both directly comparing synthetic traces generated by the models with real traces and considering the performance of a queue fed by these traces. Our results show that the models are quite accurate and can be effectively used as synthetic traffic generators.*

## 1. Introduction and Motivation

Service providers, network operators and designers are very much interested in understanding the characteristics of Internet traffic. It is indeed clear now that the design of any network element, including traffic engineering mechanisms, component dimensioning, resource management strategies, must rely on a careful characterization of the actual traffic that the network should carry. This task is made particularly challenging by the fast evolution of Internet services. Pushed by many recent successful business stories and relying on the increase of both computational power and bandwidth, new services are continuously offered and the pattern of applications the users are fond of changes quickly. In ten years, traffic transformed from being mainly generated by web browsing applications to being composed, for a good fraction, of data exchanges between clients of peer-to-peer (P2P) file sharing applications. Recently, the success of streaming applications has also introduced a large amount of UDP traffic, that is delay-sensitive rather than loss-sensitive and is often loosely controlled with respect to network conditions.

Of these streaming applications, peer-to-peer TV (P2P-TV) systems seem to be particularly critical, due to the large volume of generated traffic. Several commercial

*This work has been supported by the European Commission through NAPA-WINE Project (Network-Aware P2P-TV Application over Wise Network), ICT Call 1 FP7-ICT-2007-1 and by FAPEMIG-BRAZIL through APQ-00734-09.

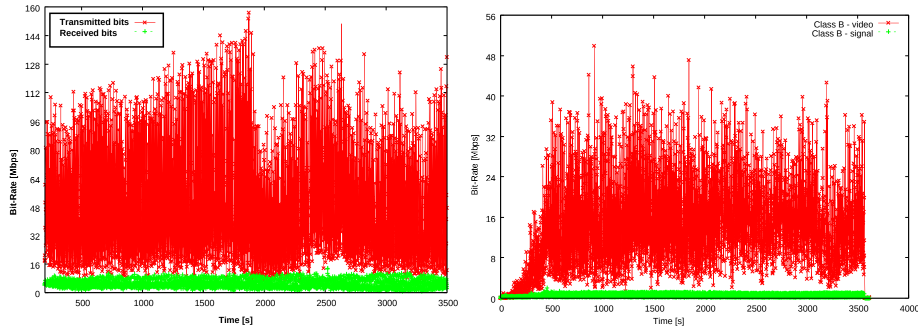
P2P-TV systems such as PPLive [ppl ], TVAnts [tva ], SOPCast [sop ], among the most popular ones, are already available and have reported a great success with an increasing number of users watching their channels at rates that can reach 1 Mbps. These applications have the potential to completely transform traffic pattern and even push network usage to critical regions; some researchers claim that P2P-TV may overload backbone links [Choffnes and Bustamante 2008]. Indeed, their impact may be disruptive due to the large volume of generated traffic combined with huge number of users. For instance, PPLive had over 200 distinct on-line channels, a daily average of $400,000$ aggregated users, and most of its channels had several thousands of users at their peaks [Vu et al. 2007]. Investigating the characteristics of this new traffic is therefore an extremely important task.

However, knowing and characterizing the traffic alone are not sufficient. There is also the need for simple and flexible traffic models that can be effectively introduced in network simulators or complex dimensioning and planning tools. Traffic models should trade-off at best the opposite needs of being simple and accurate. In this paper, we contribute at both the knowledge of P2P-TV traffic and the proposal of simple traffic models. We focus on PPLive, one of the most popular P2P-TV application. We start by characterizing, through real experiments, the traffic that is received and generated by the application peers focusing on the amount of information that is exchanged in small time windows. We organize the observed traces in three classes, based on their average bit-rate, and we study the traffic classes in terms of the bit-rate distribution and autocorrelation function. We then propose two simple traffic models: a *Memoryless* model, that just fits the distribution of the amount of exchanged traffic in the considered time windows; and a *Hidden-Markov* model, that catches also time correlations. We evaluate the accuracy of the proposed models as synthetic traffic generators, by both directly comparing real traces with the generated synthetic traces, and evaluating the performance of a simple queue fed by either the real traces or the synthetic ones.

The main contributions of our work are twofold. First, our analysis enlightens some interesting properties of the traffic generated by the considered P2P-TV application. We would like to mention two of these properties. First, time correlations are relatively short (they basically span over a few seconds). Second, while real traffic traces taken from different peers can be quite different one from the other, the aspect that mainly influences them is probably the mean bit-rate. This makes traffic traces classification extremely simple. Second, traffic seems simpler to engineer than other kinds of Internet traffic. A simple probabilistic traffic source already emulates pretty well real traffic traces; the further introduction of a short memory allows us to even catch the dominating time correlations. Such simplicity is probably due to two reasons: first, the nature of the video-stream makes incoming traffic profile quite regular; second, these applications have a certain degree of "aggressiveness" towards the network that simplifies traffic characteristics: no congestion control mechanism seems to be applied and generated traffic loosely adapts to network conditions.

## 2. Traffic measurement and analysis

Our work is based on real traces collected during an experiment performed in the context of the NAPA-WINE Project, funded by the EU [nap ]. From the experiment we extracted a number of traces corresponding to peers in a Campus, with potentially high bandwidth in transmission. The collected traces involved 27 controlled-peers from 4 NAPA-WINE partners' Institutions, in 3 European Countries and in 3 different Autonomous Systems.




(a) Transmitted and received bits (b) Transmitted video and signaling

**Figure 1. Characteristics of a larger contributor peer (Class C peer).**

The diverse geographic locations in which traces were collected allow us to define the main relevant P2P-TV traffic characteristics for our modeling purposes, under a wide variety of network conditions. The experiment lasted one hour during which packet-level traces were collected and later, offline, analyzed. The experiment was performed on February 29th, 2008; all peers were watching at the same popular PPLive channel, i.e., “CCTV-1” selected at 14:00-CET.

From the packet level traces we measure the bit-rate during time intervals of 1s separating incoming (received) and outgoing (transmitted) traffic. PPLive has an average streaming rate of about 5 Mbps in transmission and of about 600 kbps in reception. From the $40,000$ peers that each PPLive client contacts, we observed that only $3,000$ peers really contributes to the stream. Incoming traffic corresponds basically to the video content, and it is pretty much the same in all the traces. On the contrary, when the transmission side is considered, there can be dramatic differences in the peers’ behavior. Some peers act as strong distributor of the traffic, and retransmit several times the video content they have received, some others retransmit only a limited portion of it, if any. As an example, Fig.1 reports the transmitted and received traffic traces of a peer which contributes much. The received traffic, even if composed of many packets coming from different peers, results in a video stream that is relatively regular. The transmitted traffic, on the contrary, may be quite bursty depending on the number of neighbors to which the peer is distributing the video flow. For this reason, in what follows we focus on transmitted traffic only which is typically larger in volume, more variable, and more challenging to model.

As observed in [Ciullo et al. 2008], the packet size distribution is bi-modal. Packets are either large, almost constant around 700 B, or smaller than a few hundreds bytes; this distribution suggests that a simple criterion to distinguish signaling from video packets is based on their size. In this set of traces, we consider packets smaller than 1000 B as pure signaling packets; while packets longer than 1024 B are carrying video information. Given the packet size distribution presented in [Ciullo et al. 2008], our choice do not disagree with that one used previously. Clearly, this simple criterion is not perfect, it cannot account for video packets piggy-backing signaling information or small video packets possibly deriving from fragmentation of larger packets; however, it allows us to have a first view of traffic composition. Also in Fig. 1 we report the signaling and data bit-rate for a large contributor peer. The signaling bit-rate is almost negligible with respect to the data bit-rate.

In what follows, we represent a trace by a series $\{Z_t\}_{t=1}^{T}$ taking values in the $\Re$ set, where the $t-$th element represents the total number of bits transmitted by the probing peer, over 1s time intervals. We then derive a number of characteristics of the series, such as the distribution of the values and the autocorrelation. We denote by $F_Z(z)$ and $f_Z(z)$ the empirical cumulative distribution function (CDF) and probability density function (pdf) of a collected series; the autocorrelation is $A(\tau)$ and is computed as

$$A(\tau) = \frac{E[Z_t Z_{t+\tau}] - E[Z_t]^2}{E[Z_t^2] - E[Z_t]^2}. \tag{1}$$

As previously mentioned, the traces can have quite different characteristics one from the other. Finding a way to categorize PPLive traces is not that easy. However, we choose a simple approach and grouped traces according to their mean bit-rate. We identify 3 classes of traces that have quite homogeneous characteristics:

- Class A, small contributors: the mean bit-rate is in the range [0,1]Mbps.
- Class B, medium contributors: the mean bit-rate is in the range [1,4]Mbps.
- Class C, large contributors: the mean bit-rate is in the range [4,9]Mbps.

Table I reports for each trace (named after the university short name and the client identification number inside the institution) the mean and maximum bit-rate observed during the trace. Notice that most of the clients of the same institution falls in the same class, with only a few exceptions. Most of POLITO (*Politecnico di Torino*) and UNITN (*Universita di Trento*) traces belong to the same class, class C, while WUT (*Warsaw University of Technology*) and BME (*Budapest University of Technology and Economics*) are on separate classes (classes A and B, respectively). Let us now focus on the pdf of the transmitted traffic. Fig.2 shows the transmitted traffic pdf for all traces, grouped in different plots according to the criterium reported above. The pdf are created considering bins that are 500 kbps wide. Reflecting the different mean values of the transmission bit-rate, the three classes have different shapes that correspond to: an L-shape for class A, a bell-shape for class B and a more uniform like distribution over a wide support for class C traces. Despite the simplicity of the grouping criterion, the traces of the same class have quite the same behavior: there is some gap between curves, but limited in size and the overall shape is preserved.

Table 1 reports also, in the last columns, the mean and maximum number of neighbors with which the peer has exchanged traffic during a time window. As expected, this number reflects the overall bit-rate; the larger the bit-rate is, the larger the number of neighbors is. Fig.3(a) shows the distribution of the observed neighbors per time window for a class C peer. Two peaks are clearly visible: one around 150 and one around 270 peers. To investigate further the relationship between neighbors and traffic, we plot also, in the same figure ((b)), the number of neighbors versus the bit-rate for each time window: the x-coordinate of the points is the bit-rate observed during a time window, the y-coordinate is the number of neighbors. Three points are reported for each time window (with the same x-coordinate but different y-coordinate); they are the number of neighbors that exchanged signaling information with the peer, the number of those that exchanged video content and the total. Signaling and video information are, as before, discriminated based on the packet size. The plot shows that the occurrence of low bit-rates is associated to time windows in which signaling information is the solely exchanged information (no video is delivered). In general, the bit-rate mainly depends on the number of neighbors

**Table 1. Video traces characteristics.**

| Class | Peer | Transmitted Mb | | Neighbors | |
|---|---|---|---|---|---|
| | | Mean | Max | Mean | Max |
| 9*A | WUT-1 | 4.64 | 52.96 | 266 | 87 |
| | WUT-2 | 3.28 | 32.32 | 36 | 92 |
| | WUT-3 | 1.12 | 11.04 | 43 | 100 |
| | WUT-4 | 1.2 | 10.08 | 55 | 140 |
| | WUT-5 | 1.84 | 16.56 | 81 | 267 |
| | WUT-6 | 3.84 | 53.2 | 35 | 71 |
| | WUT-7 | 0.72 | 8.08 | 34 | 71 |
| | WUT-8 | 0.48 | 9.84 | 38 | 74 |
| | POLITO-1 | 0.08 | 0.96 | 197 | 328 |
| 5*B | BME-1 | 13.76 | 49.84 | 135 | 422 |
| | BME-2 | 15.28 | 77.36 | 147 | 425 |
| | BME-3 | 17.04 | 49.12 | 153 | 325 |
| | BME-4 | 23.28 | 63.2 | 168 | 346 |
| | UNITN-3 | 10.56 | 28.96 | 152 | 352 |
| 13*C | POLITO-2 | 69.36 | 181.68 | 220 | 433 |
| | POLITO-3 | 51.84 | 141.84 | 216 | 402 |
| | POLITO-4 | 54.40 | 169.12 | 208 | 469 |
| | POLITO-5 | 46.88 | 109.60 | 211 | 563 |
| | POLITO-6 | 58.00 | 189.20 | 208 | 494 |
| | POLITO-7 | 72.40 | 170.96 | 230 | 418 |
| | POLITO-8 | 48.80 | 164.24 | 200 | 343 |
| | UNITN-1 | 46.80 | 159.52 | 204 | 556 |
| | UNITN-2 | 50.00 | 160.00 | 205 | 314 |
| | UNITN-4 | 50.32 | 144.48 | 207 | 316 |
| | UNITN-5 | 43.04 | 130.96 | 213 | 367 |
| | UNITN-6 | 60.64 | 187.12 | 207 | 341 |
| | UNITN-7 | 50.24 | 163.12 | 210 | 356 |

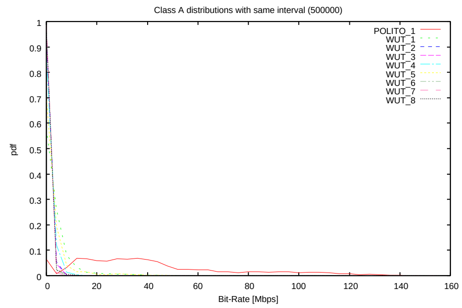
to which some video was transmitted (the green points form a linear trend); while the number of neighbors that receive signaling information is large and does not significantly influences the bit-rate. Signaling information from time to time involves twice the number of neighbors that are usually involved; this is probably due to the periodic signaling information sent to a number of peers much larger than the number of those the video is sent to. Similar observations can be drawn from the results obtained for a class B peer, whose values are reported in Fig. 4.

## 3. Models

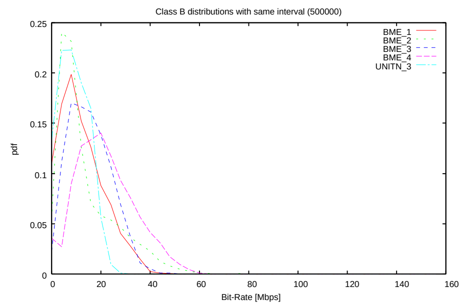
Based on the preliminary analysis of the traffic traces reported above, we can now proceed to develop some traffic models. The two models we propose, called *Memoryless* (ML) and *Hidden-Markov* (HM) models, have an increasing degree of complexity. The objective is to provide simple and efficient synthetic traffic generators; in other terms, the traffic generated by synthetic traces, $\{X_t\}$, should be as much as possible "similar" to the real traces, $\{Z_t\}$. As we will show, while both models match pretty well the distribution of the real trace, autocorrelation is matched only by the HM model.

### 3.1. Memoryless Model

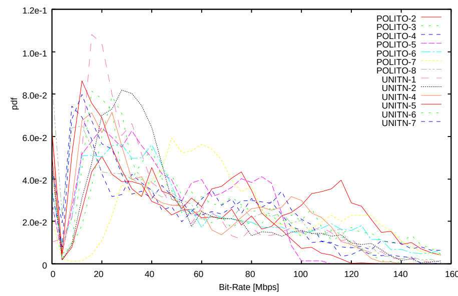
The ML model is based on the assumption that elements of the series $\{Z_t\}$ are independent and identically distributed random variables. Thus, ML model setting means




(a) Class A peers

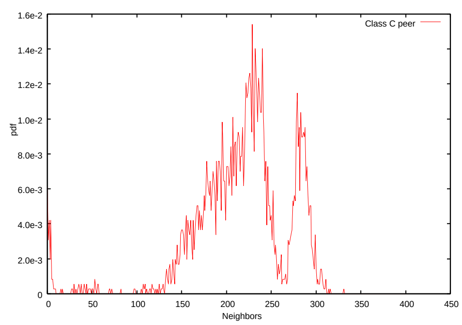



(b) Class B peers

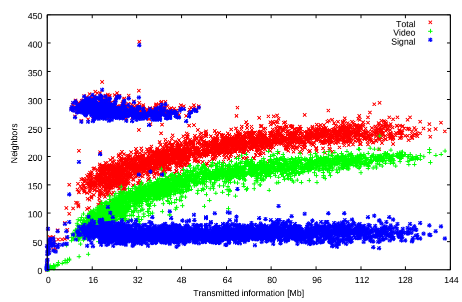



(c) Class C peers

**Figure 2. Transmitted video pdf.**




(a) Neighbors distribution

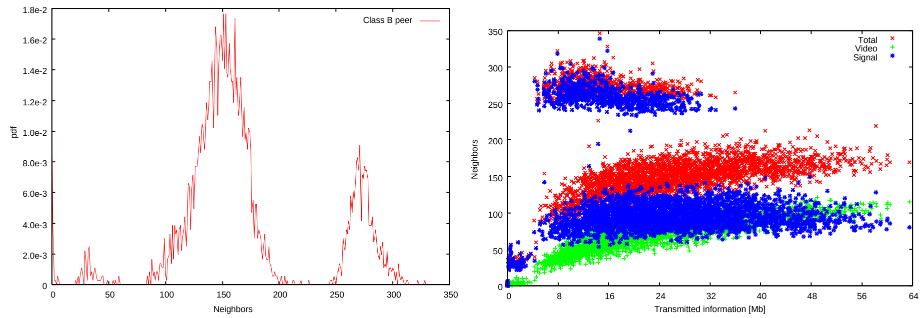



(b) Neighbors bit-rate

**Figure 3. Class C peer.**

defining a distribution, $G_X(x)$ (and the pdf $g_X(x)$), that well matches the empirical distribution $F_Z(z)$. We operate as follows. We choose $L$ *discretization levels* of $F_Z(z)$ that are equally spaced over the finite set of possible values of $\{Z_t\}$; if the values of $\{Z_t\}$ span over the interval $[Z_m, Z_M]$, the $L$ levels have size $\Delta = (Z_M - Z_m)/L$, and the $i-$th level corresponds to the interval $[Z_m + (i-1)\Delta, Z_m + i\Delta]$. The series that is obtained after discretization is denoted by $\{Z_t^{(\mathrm{d})}\}$. Then, the ML model is defined by the probabilities $p_i$ that the synthetic trace $\{X_t\}$ takes a given value $x_i$: the probability $p_i$ is set to $n_i/T$ where $n_i$ is the number of samples of $\{Z_t\}$ that fall in the $i-$th discretization level and $T$ is the total number of samples; the value $x_i$ is computed as the mean of these $n_i$ values. The probabilities $p_i$ form the pdf $g_X(x)$.

Fig.5 reports the QQ plot of the synthetic versus real traces for a class C peer. Two different values of the number $L$ of discretization levels are considered: 10 and 40. Clearly, the larger the number of levels is, the better the QQ plot is. However, the plot is




(a) Neighbors distribution (b) Neighbors bit-rate

**Figure 4. Class B peer .**

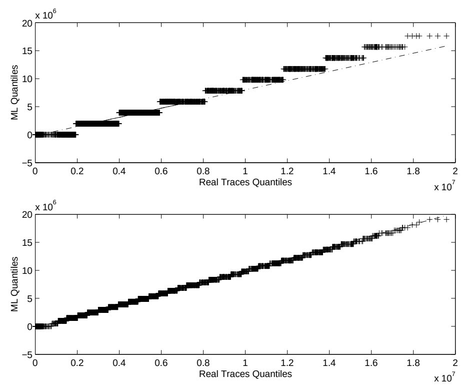



**Figure 5. ML model: QQ plot of the synthetic versus real traces of a class C peer, 10 discretization levels (top plot) and 40 discretization levels (bottom plot).**

pretty good in both cases. Fig.6 shows that the pdf $g_X(x)$ of the ML model well matches the pdf $f_Z(x)$ of the $\{Z_t\}$ series.

Consider now the autocorrelation function shown in Fig.7. The real trace presents several regular peaks that reflect the periodic pattern of the traffic. Periodicity is around 5s and peaks are present for values of the time lag $\tau = \{5, 10, 15, \cdots\}$. This periodicity is caused by the fact that peer activity appears bursty, so that every 5 seconds new bulks of data are exchanged with other peers. Synthetic traffic traces generated by the ML model cannot catch this correlation, due to the memoryless property. Indeed, autocorrelation is approximately equal to zero for all time lags greater than 0.

### 3.2. *Hidden-Markov* Model

As Fig.7 shows, the ML model cannot catch the typical autocorrelation exhibited by traffic. To also capture time correlation, some memory needs to be introduced. As a modeling approach we use a discrete time *Hidden Markov* Model (DT-HMM). Following the definition presented in [Rabiner 1989], a *Hidden-Markov* model is a doubly embedded stochastic process with an underlying stochastic process that is not observable (it is hidden), but can only be observed through another set of stochastic process that produces a sequence of observations. The basic idea is to use a discrete time Hidden Markov chain to introduce time correlation; in each state of the chain there is a different pattern of bit-rate generation. The Hidden Markov chain is derived by means of a training phase, during which the best fitting with the real trace is found.

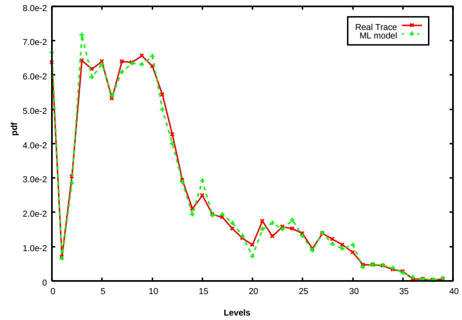



**Figure 6. ML model: Pdf real and discretized values with 40 levels for a class C peer.**

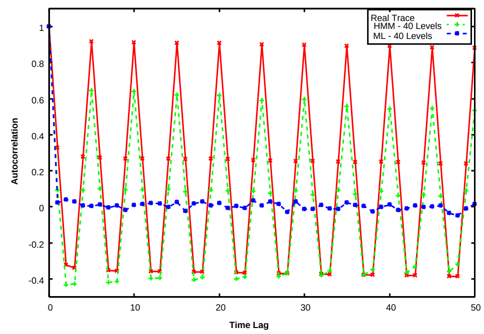



**Figure 7. ML and HM models: Autocorrelation for a class C peer.**

Let $\{H_t\}$ be the DT-HMM that we wish to learn and to use as a synthetic traffic generator. Each element of $\{H_t\}$ contains two components: the hidden state $S_t \in \mathcal{S} = \{s_i : i = 1, 2, \cdots, N\}$, which provides the process time memory, and the observation symbol $O_t \in \mathcal{O} = \{o_k : k = 1, 2, \cdots, M\}$, that represents the amount of bits that are transmitted in one time slot. Following the notation in [Rabiner 1989], the one step transition probability between any two hidden states $(i, j)$ is given by the $(i, j)-$th element of a matrix denoted by $\boldsymbol{A}$. The observation symbol probability distribution in state $j$, is given by the $j-$th row of matrix $\boldsymbol{B}$, denoted by $\boldsymbol{B}_j$, and such that $\boldsymbol{B}_j = \{b_j(o_k)\}$ where $b_j(o_k) = P[O_t = o_k|S_t = s_j]$. The initial state distribution is given by $\boldsymbol{\pi} = \{\pi_i\}$, with $\pi_i = P[S_1 = s_i]\ \forall\, i$. We use the compact notation $\boldsymbol{\Lambda} = (\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{\pi})$ to indicate the complete model parameter set. In what follows, we describe how the main DT-HMM elements are defined in our application case.

*a) The number of hidden states.* We set the number of states $N$ to 5, since the autocorrelation function, as shown in Fig.7, suggests that traffic patterns occur with periodicity of 5s (i.e., 5 time windows). Indeed, although the states are hidden, for many practical applications there is often some physical meaning attached to them. In our case, the set of hidden states gives the process memory of 5s that is shown by the autocorrelation function in Fig.7. The same figure also shows that the HM model can catch autocorrelation of the real traces pretty well (observe the dashed green line).

*b) The number of distinct observation symbols per state.* As we are interested in defining how much traffic is generated or received by a P2P-TV client, the set of observation symbols is the set of values of the bit-rate in the real trace (corresponding to observations in the DT-HMM model). These are real values that we discretize into $L$ val-

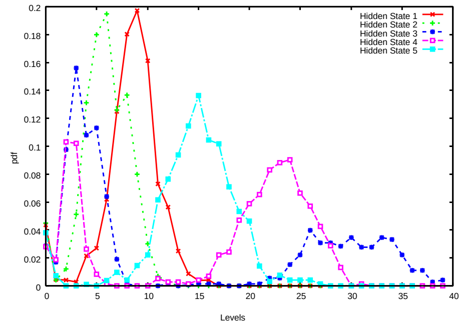



**Figure 8. HM model: Bit-rate pdf for each state of the DT-HMM.**

ues, in the same way as for the ML model. Thus, the number of observation symbols is equal to the discretization levels, $M = L$. For all results reported in Sections 4 and 5, $L$ was set to $40$. (We have also experimented other values for $L$, larger than 40, but we did not achieve any significant improvement of the accuracy. )

*c) Model parameters:* $\boldsymbol{\Lambda} = (\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{\pi})$. The definition of model parameters is the answer of the following question: *How do we adjust the model parameters* $\boldsymbol{\Lambda}$ *in order to maximize* $P(\{O_t\} = \{Z_t^{(\mathrm{d})}\}|\boldsymbol{\Lambda})$*?* In other words, the model parameters are chosen so as to best represent how a given observation sequence comes about. The sequence $\{Z_t^{(\mathrm{d})}\}$ used to adjust the model parameters is called *training sequence* since it is used to "train" the DT-HMM. The training algorithm is the *Baum-Welch algorithm* which main ideas of parameter estimation are similar to the *Expectation-Maximization* approach. We refer the reader to [Rabiner 1989] for further details on the training algorithm. To give some highlights into the model parameter definition, observe the $\boldsymbol{A}$ matrix as it comes out after the training phase of a class C peer that we use in this section as a sample case:

$$\boldsymbol{A} = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ 1.70e\text{–}2 & 0 & 9.83e\text{–}1 & 0 & 0 \\ 8.00e\text{–}3 & 0 & 0 & 9.92e\text{–}1 & 0 \\ 3.10e\text{–}2 & 0 & 0 & 0 & 9.69e\text{–}1 \\ 1 & 0 & 0 & 0 & 0 \end{pmatrix}.$$

As we can see from the matrix structure, the training phase preserves the temporal correlation. With high probability, the hidden process moves through 4 states before returning to the initial state. For completing the model parameters definition, Fig.8 shows the pdf of the observation symbols in each hidden state, i.e., the probabilities $\boldsymbol{B}_j$.

Let us now focus on the accuracy of the HM model. Fig.9 shows the comparison of the pdf obtained from the HM model against the pdf of the real trace. Similar to the ML model, the pdf matches very well the pdf of the real trace. However, as already mentioned, Fig.7 states that the autocorrelation is only captured by the HM model. These results show that the traces generated by the HM model can well represent the real traces, when basic stochastic characteristics are considered.

## 4. Queuing Analysis

The objective of this section is to assess the goodness of the proposed models as synthetic traffic generators. We consider the simple case of a single bottleneck queue. We feed the queue with real and synthetic traces and evaluate queue performance in terms of loss

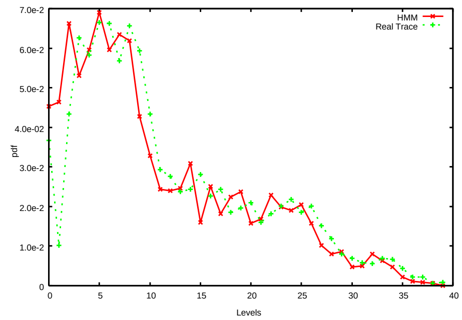


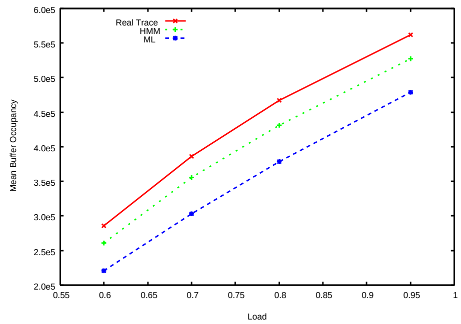

**Figure 9. HM model: Pdf: real trace and 40 discretization levels.**

probability mean buffer occupancy and the CDF of buffer occupancy. We define the relative error as the difference between the measure value given by proposed model and that one given by the real trace, divided by the measure given by the real trace.

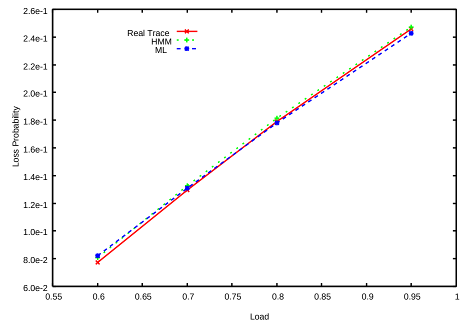
The queue is fluid and modeled as follows. Time is divided into 1s long slots. In each slot, the queue receives an amount of bits specified by a given trace (either a real trace or a synthetic one). The output rate is constant and set so as to obtain a desired load, defined as the ratio of the mean input bit-rate and the output rate. Input bit-rate is constant during a time slot. For performing the fluid simulation, TANGRAM-II tool was used [de Souza e Silva and et al 2006].

In the set of results reported here, we compare the queue performance obtained feeding the queue with a real trace and with synthetic traces derived from the associated models. We consider buffer size equal to 128Kb and a class C peer. Fig.10 reports the mean buffer occupancy versus load obtained by the real trace, the ML and HM models. For the loss probability measure, the models are almost indistinguishable. However, for the mean buffer occupancy, HM model is more accurate: the maximum relative error is 8% against 22% of error provided by ML model. The HM model also outperforms the ML model for the CDF of buffer occupancy, providing a maximum relative error equals 7% against 16% error of the ML model. The apparent contradiction of a not perfect buffer occupancy and distribution of the buffer occupancy estimations and very accurate prediction of loss probability is due to the fact that losses are dominated by the length of heavy traffic periods. For the the other analysed traces, the HM model captures better the behavior of the real trace, even being a simple model.

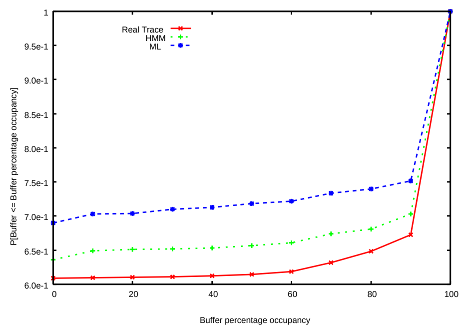
Interestingly, the results presented in this section suggest that P2P-TV traffic is not that difficult to model, and this traffic is probably easier to engineer than other kinds of more traditional Internet traffic. The reason is probably twofold. First, the streaming nature of the traffic makes the stream bit-rate quite smooth, continuous, and not that bursty and time variable as other traffic is. Second, these applications are quite aggressive towards the network, they barely apply any congestion control mechanism, and they tend not to adapt to network conditions. Indeed, many of the difficulties in modeling traditional Internet traffic are due to the combination of the long file sizes and the closed-loop nature of TCP. Similar results were obtained with other traces in the same class or belonging to other classes.




(a) Mean buffer occupancy




(b) Loss Probability




(c) CDF of buffer percentageoccupancy

**Figure 10. A class C trace and their HM and ML models. Load equal to $0.8$.**

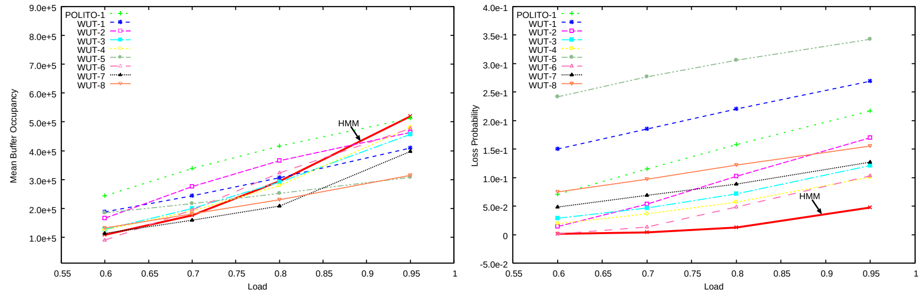
## 5. Representativeness of model

In previous section, we have evaluated the accuracy of the models as synthetic traffic generators by comparing the impact on a queue of a real trace and of the synthetic traces generated by models that were fit on the trace itself. In this section, instead, we aim at assessing how representative the model derived from a given trace is of other similar P2P-TV traffic, i.e., of other traces belonging to the same class. This is interesting because it can be considered as a measure of the degree of “homogeneity” of classes; and, if traces of a class are similar, it is enough to derive the model of one trace to have a traffic generator that represents the whole class. For example, for testing some traffic engineering solution, it would be enough to test the solution with three models (one per class) to consider the solution tested under a wide range of possible traffic patterns generated by PPLive clients. Since the HM model seems more accurate and promising, while still being simple, in this section we focus only on it.

We consider again the performance of a fluid queue, modeled and simulated as described in previous section. Of all the traces of a class, we select one trace, and call it the *base* trace of the class. We then check the queue performance fed by synthetic traces derived from the model of the base trace, and from all the real traces belonging to the class.

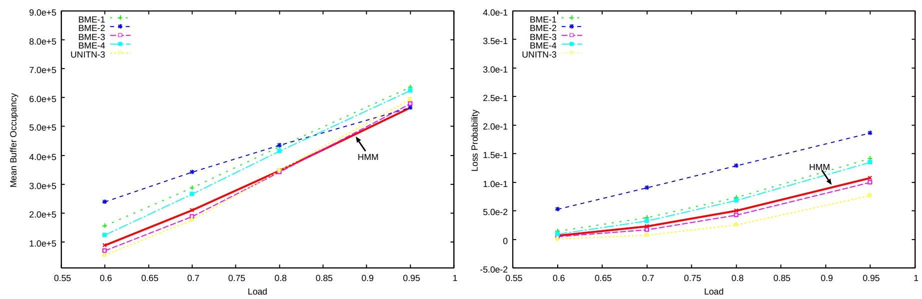
### 5.1. *Class A* Peers

Let us start our analysis by the peers that belong to *Class A*. The results for mean buffer and loss probability measures are show in Fig.11. The HM model is not that representative for all traces of this class, as we will see on sections 5.2 and 5.3. Analysing the traces and the bit-rate distribution reported on Fig.2, although the mean is almost the same for all




(a) Mean buffer occupancy (b) Loss Probability

**Figure 11. Class A traces and HM Model. Load equal to** $0.8$**.**




(a) Mean buffer occupancy (b) Loss Probability

**Figure 12. Class B traces and HM Model. Load equal to** $0.8$**.**

traces, the their bit generation shape is not that uniform. For instance, the WUT-1 trace has a high transmission peak at the beginning of the video generation. The base trace, however, has this peak at the end of the video generation (a similar behavior for almost all traces in this Class). As a last remark, these peaks on the first minutes of the trace or on the last minutes of it can explain the poor approximation for the loss probability measure.

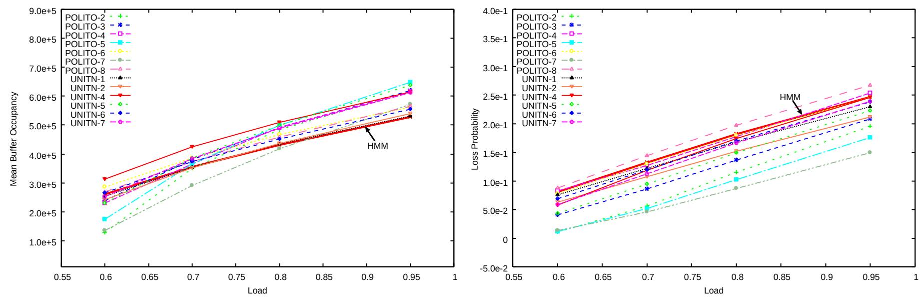
### 5.2. *Class B* Peers

In this section we analyse how the model performs considering the *Class B* peers. Fig.12 shows the results for the mean buffer occupancy and loss probability. The HM model provides satisfactory results for the loss probability, being more accurate than the mean buffer occupancy. For both measures, only the BME-2 results diverge from the model results, with high maximum relative errors (64% for the mean buffer measure in a low load scenario and 43% for the loss probability measure).

### 5.3. *Class C* Peers

Let us focus our discussion in the peers that belong to *Class C*. Fig.13 shows the set of measures performed using the HM model and the real traces. The HM model captures quite well the overall *Class C* behavior, confirming that it is possible to parametrize a very simple model that can be used as a synthetic trace for a set of real traces. For the mean buffer occupancy, the maximum relative error reached is around 17%, excepted for three peers in which the approximation is not that good for low buffer load: if the losses are resulted by the the long range dependence autocorrelation, in some cases, the measure for the low load can be less accurate. For the buffer distribution, not shown here due to space limitation, the maximum relative error is around 15%; for the loss probability measure the error of almost all traces fluctuates around 10%.

Concluding the analysis of the model representativeness, we can see that, even for the case in which the model does not represent very well the traces in the same Class (*Class A*) the HM model provides highlights about the behavior of different real traces, with almost the same mean bit-rate generation. The results presented here are very important for a first step on the direction of understanding and modeling P2P-TV traffic.




(a) Mean buffer occupancy (b) Loss Probability

**Figure 13. Class C versus HM Model. Load equal to** $0.8$**.**

## 6. Related Works

Despite the only recent deployment of P2P-TV applications, there already are in the literature a number of studies about the impact of P2P-TV traffic on the Internet.

Most of these works present results from measurements taken on a single P2P-TV system, as well as its characteristics. For instance, Hei et al. [Hei et al. 2007a] develop a dedicated crawler to measure and characterizes PPLive traffic. The author of [Vu et al. 2007] focus in on the node degrees of popular versus unpopular channels in PPLive. Instead, the results presented in [Hei et al. 2007b] are focused in inferring QoS metrics. Another set of works compare different P2P-TV systems. Authors in [S.Ali et al. 2006] analyze and compare PPLive and SOPCast, presenting metrics like transmitted/received bytes, etc. Silverston and Formaux in [Silverston and Fourmaux 2007] present statistics from a large-scale live event broadcasted on P2P networks by PPLive, PPStream, SOPCast and TVAnts. To the best of our knowledge, no synthetic P2P-TV traffic generator has been proposed so far in the literature.

Finally, over the years, HM models have proved to be useful for a number of applications, given their simplicity and nice mathematical characteristics. Traditionally used for speech recognition [Rabiner 1989], one can cite different examples of their application in the field of telecommunications, including modeling of network channels [Salamatian and Vaton 2001], network performance evaluation [Wei et al. 2002], development of adaptive FEC algorithms [Duarte et al. 2003] and modeling short-term dynamics of packet losses [Silveira and de Souza e Silva 2006].

## 7. Conclusions

In this paper, we focused on the traffic generated by PPLive. Starting from the analysis of real traces collected by running the application clients from several peers spread over the Internet, we focused on the bit-rate generated by a client in short time windows of

1s. We proposed two traffic models: a *Memoryless* model that fits the distribution of the bit-rate observed in real traces, and a slightly more complex model based on *Hidden Markov* chains which fits also the autocorrelation function of the bit-rate. We tested the proposed models as synthetic traffic generators by considering the impact of the synthetic traffic traces and real traces on a queue in terms of the mean buffer occupancy and loss probability. Despite their simplicity, our models are pretty accurate.

## References


Network-aware p2p-tv application over wise networks. http://www.napa-wine.eu.

Pplive. http://www.pplive.com.

Sopcast. http://www.sopcast.com.

Tvants. http://www.tvants.com.

Choffnes, D. R. and Bustamante, F. E. (2008). Taming the torrent: A practical approach to reducing cross-isp traffic in peer-to-peer systems. In *SIGCOMM*.

Ciullo, D., Mellia, M., Meo, M., and Leonardi, E. (2008). Understanding p2p-tv systems through real measurements. In *GLOBECOM*.

de Souza e Silva, E. and et al, R. M. M. L. (2006). Modeling, analysis, measurement and experimentation with the tangram-ii integrated environment. In *ValueTools*.

Duarte, F. P., de Souza e Silva, E., and Towsley, D. (2003). An adaptive fec algorithm using hidden markov chains. *Performance Evaluation Review*.

Hei, X., Liang, C., Jian Liang, Y. L., and Ross, K. W. (2007a). A measurement study of a large-scale p2p iptv system. *IEEE Transactions on Multimedia*.

Hei, X., Liu, Y., and Ross, K. (2007b). Inferring network-wide quality in p2p live streaming systems. *IEEE JSAC*, 25(9):1640–1654.

Rabiner, L. R. (1989). *A Tutorial on Hidden-Markov Models and Selected Applications in Speech Recognition*. In *Proceedings of the IEEE*, volume 77.

Salamatian, K. and Vaton, S. (2001). *Hidden-Markov* modeling for network communication channels. In *ACM SIGMETRICS*.

S.Ali, A.Mathur, and H.Zhang (2006). Measurements of commercial peer-to-peer live video streaming. In *Workshop on Recent Advances in Peer-to-Peer Streaming*.

Silveira, F. and de Souza e Silva, E. (2006). Modeling the short-term dynamics of packet losses. *Performance Evaluation Review*, pages 27–29.

Silverston, T. and Fourmaux, O. (2007). Measuring p2p iptv systems. In *ACM NOSSDAV*.

Vu, L., Gupta, I., Liang, J., and Nahrstedt, K. (2007). Measurement of a large-scale overlay for multimedia streaming. In *16th Int. Symposium on High Performance Distributed Computing*.

Wei, W., Wang, B., and Towsley, D. (2002). Continuous-time *hidden-markov* models for network performance evaluation. *Performance Evaluation*, pages 129–146.