Coupling from the past in hybrid models for file sharing peer to peer systems

Bruno Gaujal, Florence Perronnin

Coupling from the Past in Hybrid Models for File Sharing Peer to Peer Systems

Bruno Gaujal[1] and Florence Perronnin[2]

[1] INRIA and Lab. ID-IMAG (CNRS, INPG, INRIA, UJF) 51, Av. J. Kunztmann, Montbonnot, France
Bruno.Gaujal@imag.fr
[2] UJF and Lab. ID-IMAG (CNRS, INPG, INRIA, UJF) 51, Av. J. Kunztmann, Montbonnot, France
Florence.Perronnin@imag.fr

Abstract. In this paper we show how file sharing peer to peer systems can be modeled by hybrid systems with a continuous part corresponding to a fluid limit of files and a discrete part corresponding to customers. Then we show that this hybrid system is amenable to perfect simulations (*i.e.* simulations providing samples of the system states which distributions have no bias from the asymptotic distribution of the system). An experimental study is carried to show the respective influence that the different parameters (such as time-to-live, rate of requests, connection time) play on the behavior of large peer to peer systems, and also to show the effectiveness of this approach for numerical solutions of stochastic hybrid systems.

1 Introduction

Hybrid systems are very useful to model discrete systems with several time and space scales. In that case, one typically uses *fluid limits* for the parts of the system with fastest and largest scales. These models have been introduced in various domains under the form of fluid queues [1], continuous Petri nets [2], or timed automata [3]. In this paper, we will consider one such example, namely peer to peer systems, where two types of dynamics are superimposed. The slow dynamics concerns the customers, who join and leave the system. The fast dynamics concerns the files and their transfers between the customers. A natural model for file sharing peer to peer systems mixes a discrete stochastic system to model the behavior of the customers and a deterministic differential equation for the mean behavior of the files, seen as a fluid quantity.

The analysis of such stochastic hybrid systems is often difficult on a mathematical as well as on a numerical point of view and such large hybrid systems are often considered computationally untractable. Simulation approaches are efficient alternatives to estimate their behavior by providing samples distributed according to their asymptotic distribution. However, simulation has several drawbacks. First, simulations do not make any sense unless the system has ergodicity properties, which are sometimes difficult to check. Second, even under

A. Bemporad, A. Bicchi, and G. Buttazzo (Eds.): HSCC 2007, LNCS 4416, pp. 217–230, 2007.

the right ergodicity conditions, classical simulation techniques only provide approximations of the asymptotic behavior. The longer the simulation the more accurate the result, but it is usually hard or impossible to be more precise than this general statement. Recently, Propp and Wilson have used backward coupling techniques ([4]) to design a simulation algorithm to get *perfect samplings* (*i.e.* whose distributions are not approximations but the exact asymptotic distributions) of *discrete time, finite* Markov chains.

In this paper we show how their idea can be adapted to the *infinite* and *continuous* case at hand, by using *regeneration points*. We use this property to design a perfect simulation algorithm of our peer to peer model, by using additional monotonicity properties. Finally, we carry an experimental study of the performances of the model based on this sampling technique. A similar approach has been used in [5] for decoupled hybrid systems (where the discrete part does not depend on the continuous one). In that case, the coupling of the perfect simulation occurs in a very controlled manner because the system is uniformly contracting. Here however, the interplay between the discrete and the continuous parts are more intricate so that the coupling time might have a larger variance.

Actually, the goal of this paper is two-fold. First, we propose a new hybrid model for file sharing P2P systems, combining the effects of popularity and age decays of files on the customers behavior. Second, we show how this kind of stochastic hybrid systems can be solved numerically by using a new and powerful simulation method based on coupling from the past properties. In practice, this approach has proved to be very fast and we have been able to treat very large cases (with state spaces of size larger than 10^7) within one or two minutes over a standard PC.

2 A Hybrid Model of Peer to Peer File Sharing Systems

Downloading popular multimedia content from the Internet can take a long time due to bandwidth bottlenecks and to Web server overload. The central idea of peer to peer (P2P) systems is to leverage the downloaders' own (often unused) resources to provide a globally better service. In the context of file download, the resource is upload bandwidth and the service is a faster diffusion of popular files. For instance, a popular file $\mathcal{F}$ downloaded by a user A may also be of interest to a user B which is "closer" to A than to the origin Web server hosting file $\mathcal{F}$. Then if B downloads file $\mathcal{F}$ from A instead of the origin server, the benefit is threefold : B downloads the file at a high rate on the local network; B doesn't contact the origin server, which reduces the load on this server and finally, bandwidth is saved on the wide-area network since the data is transferred locally.

An important aspect of P2P systems is that clients (downloaders) are also servers (uploaders). For this reason, these systems are said "self-scaling" since the resources increase with the demand.

Among P2P systems, one of the most popular applications is file sharing. The most famous systems such as KaZaA or Gnutella [6,7] belong to this category. They can mainly be described by the copies of files that all the users make

available for download (typically, copies of files previously downloaded). These systems may be intricate and are extremely difficult to model in full details. Here are the main simplifications used in this paper. First, the peers are assumed to be statistically homogeneous and we only consider two variables, $N(t)$, the number of customers (or nodes) connected to the system at time t and $x(t)$, the number of copies of all the files which are available globally in the system. We will see below that $x(t)$ can also be viewed as the popularity index of the system. The second assumption is that downloads always succeed and download times are neglected since files can be split into fragments of "unit" size for download.

2.1 Modelling with a Hybrid System

The variable $N(t)$ being discrete, it only changes values at discrete times. Thus $N(t)$ is driven by *jump instants*, which forms a point process $T_0 = 0$ (time origin), $T_1, \dots T_n$. Let $(\tau_n)_{n\in\mathbb{N}}$ be the sequence of inter jump times: $\tau_n \stackrel{\text{def}}{=} T_n - T_{n-1}$. At each time T_n, each customer has the opportunity to see the global state of the system $(N(T)n), x(T_n))$. Based on this information, each customer decides either to join, or to leave or even to remain as is. Therefore, the total number of connected customers $N(T_n)$ is a continuous time Markov chain, which infinitesimal generator is given in Figure 1. This infinitesimal generator describes

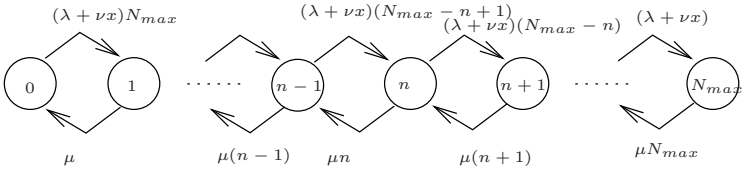

Fig. 1. The infinitesimal generator for the Markov process N

the fact that each customer leaves the system with a constant rate μ. Also, each customer joins the system with a rate proportional to the popularity of the system (which is proportional to x), plus a "blind" rate λ, independent of x. The behavior of $N(T_n)$ can be written under a constructive form as

$$N(T_n) = \varphi(N(T_{n-1}), x(T_{n-1}), \xi_n, \tau_n), \tag{1}$$

where $\{\xi_n\}_{n\in\mathbb{N}}$ is a random process of *innovations*, uniformly distributed over $[0,1]$, and φ describes the dynamics of N at jump instants:

1. If $\xi > 1-\lambda' + \nu' x(T_{n-1})(N_{max} - N(T_{n-1}))$, then the next event is a *customer arrival:* $N(T_n) = N(T_{n-1}) + 1$.
2. If $\xi < \mu' N(T_{n-1})$, then the next event is a *customer departure:* $N(T_n) = N(T_{n-1}) - 1$.
3. Otherwise, this is a *null event*, *i.e.* no customer arrives nor leaves and the system is left unchanged at time T_n: $N(T_n) = N(T_{n-1})$.

As for the continuous part, $x(t)$ is governed by a discrete process at jump instants, T_n. Upon a customer departure, it takes away all the documents it was responsible for. Since our model is symmetric over all customers and all files, the number of lost documents is uniform over all customers so that it corresponds to a proportional fraction of the total fluid. Therefore if the event at T_n is a departure then $x(T_n) = x(T_n^-)\frac{N(T_n^-)-1}{N(T_n^-)}$. This fraction of files lost may also be a random variable with mean $\frac{N(T_n^-)-1}{N(T_n^-)}$ to account for user heterogeneity as shown in Section 3.8. When a node joins the system, it does not bring exogenous files with him upon its arrival. This assumption can easily be relaxed as shown in Section 3.8. The increase of the number of files will come from the future downloads of the newcomer: $x(T_n) = x(T_n^-)$. Upon a null event, x is also left unchanged, $x(T_n) = x(T_n^-)$. We denote this evolution by

$$x(T_n) = h(N(T_n^-), x(T_n^-), \xi_n), \tag{2}$$

As for the behavior of $x(t)$ between jump times, it is given by a deterministic differential equation, In $[T_n, T_{n+1})$,

$$\frac{dx}{dt} = f(N(T_n), x, t) = \sigma N(T_n)e^{-\alpha x} + \sigma N(T_n)x\frac{N(T_n)C - x(t)}{N(T_n)C} - \theta x. \tag{3}$$

Here is a brief account on the dynamics of x between jumps. The first term $\sigma N e^{-\alpha x(t)}$ corresponds to the rate at which new files are introduced in the system. First, σ is the rate at which each customer requests files. As for $e^{-\alpha x(t)}$, it corresponds to the probability that the requested file is not yet present in the system (called the miss probability in the following) so that a download from outside is needed. The form of this term has been derived experimentally as follows.

The number of requests for each file typically follows a Zipf law with parameter β. Therefore, the number of copies per file also follows a Zipf law, since each request results in the creation of a new copy. In our model, we only know x, the total number of *copies* while the miss probability depends on the average number of distinct *files*, $d(x)$. Actually, the miss probability is uniform over all missing files (new files do not have any popularity yet) and is equal to $1-d(x)/C$. Computing $d(x)$ exactly is intricate and we could not find a close-form formula for it. However, we found empirically that $d(x)$ is very close to $C(1 - exp(-\alpha x))$ as seen in Figure 2. Parameter α was found to be equal to $e^{-\beta}/C$. Finally, we can assume that new files are introduced in the system with rate $\sigma N(t)e^{-\alpha x(t)}$.

The second term $\sigma N(T_n)x(t)\frac{N(T_n)C-x(t)}{N(T_n)C}$ corresponds to the increase of the number of copies linearly in the popularity $(x(t))$, provided that the copy is not yet present locally (the probability of local absence being $\frac{N(T_n)C-x(t)}{N(T_n)C}$).

The last term $-\theta x(t)$ corresponds to the decrease of the number of copies due to obsolescence (each copy is removed from the system by the system administrator at rate θ).

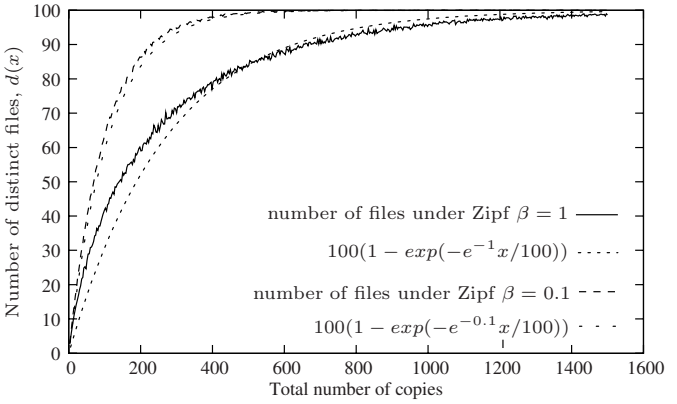

Fig. 2. Matching of the exponential with $d(x)$ under Zipf laws for popularity, with at most $C = 100$ different files and using Zipf parameters $\beta = 0.1$ and $\beta = 1$ respectively

This differential equation is Lipschitz everywhere, therefore, it admits a unique solution, once an initial condition x_0 is given, denoted $F(x_0, t)$. No closed form of the solution is known, so that one needs to use a numerical method for integration. This equation is prone to numerical instability because the second derivative of $x(t)$ is large when t goes to infinity. In our programs, we have used a finely tuned ad-hoc Runge-Kutta integrator with adaptative step sizes [8]. In any case, the step size is always larger than $\varepsilon/(T||f'||)$ where $||f'|| = \sup_{N,x} \frac{df(N,x)}{dx}$, for integrating with precision ε over an interval of length T. No further details will be given in this technical issue.

3 Coupling from the Past

Given the stochastic hybrid model for file sharing systems, we now show how to compute its steady state behavior.

3.1 Embedded Markov Chain

In this part we only study the system at its jump instants. The behavior at arbitrary instants can be easily derived as shown in Section 3.7.

Lemma 1. *The embedded sequence at jump times , $S_n \stackrel{\text{def}}{=} (N(T_n), x(T_n))$ is a homogeneous continuous time Markov chain over a continuous domain, $\mathcal{D} \subset \mathbb{N} \times \mathbb{R}$.*

Proof (sketch). The state of the process at time T_n only depends on the state at time T_{n-1}, the innovation ξ_n and the n-th inter-arrival of the jump process $\tau_n = T_n - T_{n-1}$, which value only depends on the state at time T_{n-1}. This means that $(N(T_n), x(T_n))$ is a homogeneous Markov chain over the domain of all reachable states, $\mathcal{D} \subset \mathbb{N} \times \mathbb{R}$.

In the following we will denote this global construction of this peer to peer Markov chain (P2P MC) as $S_n = \Phi(S_{n-1}, \xi_n)$. For simplicity, we also denote $\Phi(S, \xi_1, \ldots, \xi_n) \stackrel{\text{def}}{=} \Phi(\cdots \Phi(\Phi(S, \xi_1), \xi_2), \cdots)$.

The Markov chain S_n can be uniformized into a discrete time Markov chain (useful for simulation purposes) using the constant $\Lambda = N_{max}(\lambda + \nu C N_{max} + \mu)$.

Lemma 2. *The chain S_n is uniformly ergodic over $\mathcal{D}$: there exists a non-trivial measure φ over $\mathcal{D}$, some $m > 1$ and $0 \leq \beta \leq 1$ such that $\forall x \in \mathcal{D}$, $P^m(x, \cdot) \geq \beta\varphi(\cdot)$. Moreover, it has an atom in $(0,0)$ (i.e. (0,0) can be reached with a positive probability, starting from any other state).*

Proof. The discrete part, $N(T_n)$ follows a birth and death process. Therefore, $\mathbb{P}(N(T_n) = 0 | N(0) = N_0, x = x_0) \geq \mathbb{P}(N(T_n) = 0 | N(0) = N_{max}, x_0 = CN_{max}) \geq \mu^n N_{max}!/\Lambda^n$ if $n \geq N_{max}$. Again, $m = N_{max}$, $\varphi = \mathbb{1}_{(0,0)}$ and $\beta = N_{max}!(\mu/\Lambda)^{N_{max}}$ verify the definition of uniform ergodicity. The fact that $\varphi = \mathbb{1}_{(0,0)}$ means that $(0,0)$ is an atom.

Lemma 2 implies directly that S_n admits a unique stationary distribution (using general results for continuous Markov chains, see for example, [9]). Let $P^n(s, A)$ denote the transition law of n steps of the chain S_n (This is the probability $\mathbb{P}(S_n \in A | S_0 = s)$). The stationary measure Π of S satisfies $\Pi(A) = \int_{\mathcal{D}} P^1(s, A)\Pi(ds)$, for all measurable set A in $\mathcal{D}$.

3.2 Computing the Stationary State

If Π can be computed explicitly, there are many ways to draw samples from it. However, in most cases, analytical or even numerical computations of Π are impossible to obtain, either because the domain $\mathcal{D}$ is huge (in finite cases) or because the structure of the transition kernel $P^1(s, A)$ is too complex.

Without analytical or numerical knowledge of Π the most popular method for sampling from Π is simulation. The classical Monte-Carlo simulation consists in choosing an arbitrary initial value $S_0 = s_0$ in $\mathcal{D}$ and to use the constructive equations given in Equations (1), (2),(3) to generate $S_1, \ldots, S_n$ by using a random number generator for ξ. This technique works asymptotically because the sequence of samples converges in law, in the sup-norm, to the stationary distribution: $\lim_{n\to\infty} \sup_A |P^n(s_0, A) - \Pi(A)| = 0$. However, for a given finite n, the gap with the exact distribution depends on the convergence rate to the stationary distribution which is unknown in general. Here, we will show how to compute a sample in finite time which distribution is *exactly* Π (hence the name perfect), using a backward coupling technique. This technique was proposed for the first time in [4] for Markov chains over *finite* state spaces. The main idea is to run several simulations in parallel *starting in the past* from all possible initial states. If all the trajectories coincide at time 0 (meaning that they have all coupled at some point in the past) then the simulation stops and outputs the common value of all the trajectories at time 0, which happens to be a perfect sample of the chain.

Obviously, simulating trajectories from all initial states cannot be done for continuous state spaces. However, it was shown in [10] that backward coupling can also be defined for continuous state Markov chains. Here, we do not need the general theorems in [10] and we show how the backward coupling idea can be applied for the P2P MC using directly its uniform ergodicity.

Theorem 1. *The vertical backward coupling time*

$$K \stackrel{\text{def}}{=} \min\{n \geq 0 : \Phi(s, \xi_{-n}, \ldots, \xi_{-1}, \xi_0) = \Phi(r, \xi_{-n}, \ldots, \xi_{-1}, \xi_0), \forall r, s \in \mathcal{D}\},$$

is a well defined random variable. Furthermore, for all $s \in \mathcal{D}$, the random variable $\Phi(s, \xi_{-K}, \ldots, \xi_{-1}, \xi_0)$ is distributed according to measure Π (denoted $\Phi(s, \xi_{-K}, \ldots, \xi_{-1}, \xi_0) \sim \Pi$).

Proof. Let us consider the sub-sequence S_{mn} that makes m steps of the transition Kernel where $m = N_{max}$. As shown in Lemma 2, with probability $p_c > \mu^{N_{max}} N_{max}! / \Lambda^{N_{max}}$, then $S_{mn} = (0, 0)$. Therefore, K is a finite random variable, by Borel-Cantelli. Now, if s_∞ is any state distributed according to the stationary measure Π, then $\Phi(s_\infty, \xi_{-n}, \ldots, \xi_{-1}, \xi_0)$ is also stationary for all $n > 0$, by definition of Π. By choosing $n = K$, we get for an arbitrary state s, $\Phi(s, \xi_{-K}, \ldots, \xi_{-1}, \xi_0) = \Phi(s_\infty, \xi_{-K}, \ldots, \xi_{-1}, \xi_0) \sim \Pi$.

From Theorem 1, one may construct an algorithm providing a sample of the chain which has the stationary distribution Π: simulate backward in time starting from an arbitrary state and using steps of size m, until $\xi_{-n} < p_c$, where S_{-n+1} is set to $(0, 0)$. This algorithm has two major drawbacks: the kernel P^m is very difficult to compute and the run may last for long (because p_c is very small). They might make it impossible to use in practice. One way to deal with these two problems is to use structural properties of the P2P chain S_n, such as monotonicity.

3.3 Monotonicity Issues

The final class $\mathcal{D}$ admits the natural component-wise ordering: $(N, x) \leq (N', x')$ if $N \leq N'$ and $x \leq x'$. The chain S is said to be *monotone* if $(N, x) \leq (N', x')$ implies that $\Phi(N, x, \xi, \tau) \leq \Phi(N', x', \xi, \tau)$ for all ξ and all τ.

To test if the chain is monotone, one considers two chains S_1 and S_2 starting with ordered values, $N_1(0) \geq N_2(0)$ and $x_1(0) \geq x_2(0)$.

One must first consider the evolution between jumps. It should be clear that the differential equation (3) is monotone in N as well as in its initial condition. Therefore, if $N_1(0) \geq N_2(0)$ and $x_1(0) \geq x_2(0)$, then for all time $t \geq 0$, $x_1(t) \geq x_2(t)$.

As for the behavior of the chain at jump times, it is monotone as long as $\xi > 1 - \lambda' N_{max}$,However if $\xi < 1 - \lambda' + \nu' x(T_{n-1}) N_{max}$, the event may either be a departure or a null event for both chains. The following tricky situation can occur: $\mu' N_2(T_{n-1}) < \xi < \mu' N_1(T_{n-1})$. This corresponds to a departure for S_1 and a null event for S_2. Now, if $x_1(T_n^-)$ and $x_2(T_n^-)$ are too close, the following can happen: $x_1(T_n) = x_1(T_n^-) - x_1(T_n^-)/N_1(T_{n-1}) \leq x_2(T_n^-) = x_2(T_n)$. So that the chain is actually not monotone under such events.

3.4 Upper and Lower Envelopes

In order to deal with monotone systems, we introduce upper and lower envelopes of the trajectories of the Markov chain S, $S_1 = (N_1, x_1)$ and $S_2 = (N_2, x_2)$, respectively. The upper (resp. lower) envelope start at time $t = 0$ in state (N_{max}, C) (resp. $(0,0)$). The upper (resp. lower) envelope evolve exactly as the Markov chain for all events which cannot cause a swap of the ordering between the two trajectories. Whenever a potential swapping event occurs, then, let $N_3 = \lfloor \xi/\mu' \rfloor$ be the largest value of N for which ξ is the null event. For N_3+1 and larger values of N, ξ would be a departure. We set $S_1(T_n) = (N_1(T_{n-1}) - 1, x_1)$ and $S_2(T_n) = (N_2(T_{n-1}), x_2(T_n^-)\frac{N_3}{N_3+1})$ so that the order remains unchanged between S_1 and S_2 Such an event can be seen as a "dummy" departure for S_2 and a "dummy" arrival in S_1. The construction of the envelopes $(S_1(T_n), S_2(T_n))$ given above can be written under the form of two new functions Γ_1, Γ_2 that describes the Markovian evolution of both envelopes at jump times: for $j = 1, 2$,

$$S_j(T_n) = \Gamma_j\Big(S_1(T_{n-1}), S_2(T_{n-1}), \xi_n, \tau_n\Big).$$

Note that by construction of Γ_1, Γ_2, the envelopes have been built such that $S_1(t)$ stays above $S_2(t)$ for all time $t \geq 0$ as soon as $S_1(0)$ is above $S_2(0)$. So the envelopes have a monotone behavior. Also note that by construction, for all initial state of the initial P2P MC, $S(0) = (N, x)$ and all time t: $S_1(t) \geq S(t) \geq S_2(t)$.

3.5 Perfect Simulation Algorithm for Peer to Peer Systems

The following theorem is the theoretical foundation of our perfect simulation algorithm for peer to peer systems.

Theorem 2. *The Markov chain $(S_1(T_n), S_2(T_n))$ hits the diagonal (i.e. states of the from $\big((N,x),(N,x)\big)$) in finite time a.s.. The hitting time $K'' =$*

$$\min\big\{n : \Gamma_1\big((N_{max}, C), (0,0), \xi_{-K}, \ldots, \xi_0\big) = \Gamma_2\big((N_{max}, C), (0,0), \xi_{-K}, \ldots, \xi_0\big)\big\}$$

is a vertical backward coupling time of the Markov chain S, so that for all initial state s, $\Phi(s, \xi_{-K''}, \ldots, \xi_{-1}, \xi_0) \sim \Pi$.

Proof. The first part of the proof is similar to the proof of the uniform ergodicity of chain S. Indeed, if a large number of departures occur, both envelopes will eventually reach $(0,0)$. Since this happens with a positive probability, the Markov chain $(S_1(T_{n-1}), S_2(T_{n-1}))$ is uniformly ergodic and K'' is a finite random variable with finite expectation.

As for the second part of the proof, it simply uses the fact that $S_1(t) \geq S(t) \geq S_2(t)$ for all initial conditions for the chain S. Consider a stationary initial condition $S(-K'') \sim \Pi$. Then, $S(0) = \Phi(S(-K''), \xi_{-K''}, \ldots, \xi_{-1}, \xi_0) \sim \Pi$ by stationarity and $S_1(0) = S(0) = S_2(0)$ by definition of K''.

From Theorem 2, it is possible to derive an algorithm to compute both an upper bound on K'' and a sample from Π. Here is the outline of such an algorithm. Start at time $-k$ (at the beginning, $k = -1$) and simulate in parallel the lower and upper envelopes, starting respectively in states $(0, 0)$ and (N_{max}, C), using the same random variables, $(\tau_{-k}, \xi_{-k}), \cdots, (\tau_{-1}, \xi_{-1}), (\tau_0, \xi_0)$ for both of them. If at time 0, both envelopes reach the same state (N_0, x_0), then this means (using theorem 2) that the system has coupled, $K'' \leq k$ and $(N_0, x_0) \sim \Pi$. Otherwise, generate a new random innovation $(\tau_{-k-1}, \xi_{-k-1})$, and restart at time $-k-1$ with initial states $(0, 0)$ and (N_{max}, C).

A classical improvement is to double the number of steps backward at each iteration, so that the simulation time becomes linear in the total number of steps (κ) which is most twice the coupling time K''.

Another improvement is to stop when both envelopes are close enough. A stopping test on equality is theoretically possible since both envelopes couple in $(0, 0)$ with positive probability, and remain exactly equal from this point on. However, the probability that N ever reaches 0 is extremely small (less than 10^{-300} in the examples of Section 4). This means that the average vertical coupling time is huge. Testing for a small gap between both envelopes reduces the simulation time drastically and keep guarantees on the intervals on the measures as seen in the following. The complete algorithm is given as Algorithm 1.

Algorithm 1. Backward simulation for P2P MC

$\kappa = 1$;
repeat
  $\kappa = 2\kappa$;
  $S_1 := (N_{max}, C)$,$S_2 := (0, 0)$ {Initialize the two envelopes at time $-\kappa$}
  for $i = \kappa$ downto $\kappa/2 + 1$ **do**
    $\xi[i]$:=Random(Uniform over $[0, 1]$); $\tau[i]$:=Random(exponential with rate Λ)
  end for
  for $i = \kappa$ downto 1 **do**
    $S_1 := \Gamma_1(S_1, S_2, \xi[i], \tau[i])$, $S_2 := \Gamma_2(S_1, S_2, \xi[i], \tau[i])$
  end for
until $S_1\dot{N} = S_2\dot{N}$ and $S_1\dot{x} - S_2\dot{x} \leq \varepsilon/3$
return S_1

In Figure 3, we display a perfect simulation of S_1, S_2. Note that several dummy events on S_1, S_2 are visible. They all correspond to discontinuous jumps of S_2 and cusps of S_1. Also note that the trajectories of x and N are a very high positive correlation: x and N both increase and decrease at the same time. This shows the effect of popularity: when the popularity is high, more customers get connected and they download more files, thus increasing the popularity. The total complexity of the program is $\kappa(c(\Gamma_1) + c(\Gamma_2)) + 2p$, where $c(\Gamma_i)$ is a constant corresponding to the time to compute Γ_i and p is the total number of steps needed to integrate the differential equation over the total simulated time, $T = \sum_{i=1}^{\kappa} \tau[i]$.

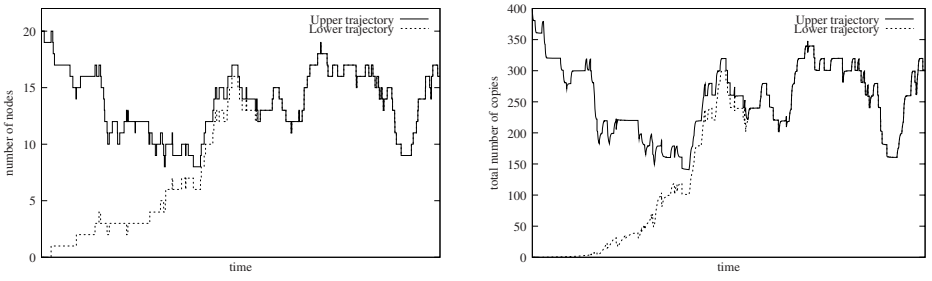

Fig. 3. Coupling of the trajectories of the two envelopes S_1, S_2 for N (left) and x (right) as generated by Algorithm 1

Since $\kappa \leq 2K''$ and $p \leq \frac{T^2}{\varepsilon} \sup_{N,x} \frac{df(N,x)}{dx}$, the complexity of the algorithm is given on average by the following lemma.

Lemma 3. *The average complexity of Algorithm 1, with precision ε is*

$$O\left(\mathbb{E}K'' + \frac{1}{\varepsilon}\left(\frac{\mathbb{E}K''}{\Lambda} + \frac{\mathbb{E}(K''^2)}{\Lambda^2}\right)\right). \tag{4}$$

Computing $\mathbb{E}K''$ is a difficult is out of the scope of this paper. We carried out several experiments to measure the number of backward steps $\kappa \leq 2K''$ in Figure 6.

3.6 Confidence Intervals

The outputs of the i-th run of the algorithm are numerical approximations of the couples $(S_1^i = (N_1^i, x_1^i), S_2{}^i = (N_2^i, x_2^i))$. The numerical integration of the differential equation yields a global error ϵ. The integration steps h are chosen small enough so that $\epsilon \leq \varepsilon/3$. Then, the outputs of the algorithm are such that the exact values verify $x_1^i - x_2^i \leq \varepsilon$.

Let E be an interval $E = [a, b] \subset [0, C]$ for which we want to compute $\pi_x(E)$ with confidence c. The central limit theorem gives the following confidence interval $I = \left[\hat{p_1} - \frac{\beta_c v}{2\sqrt{M}}, \hat{p_2} + \frac{\beta_c v}{2\sqrt{M}}\right]$, where β_c the c-percentile for the normal law and $v = \sqrt{\pi_x(E)(1-\pi_x(E))} \leq 1/2$ and $\hat{p_1} = 1/M \sum_{i=1}^{M} \mathbf{1}\{x_1^i \in E \wedge x_2^i \in E\}$, $\hat{p_2} = 1/M \sum_{i=1}^{M} \mathbf{1}\{x_1^i \in E_\varepsilon \vee x_2^i \in E_\varepsilon\}$.

3.7 Stationary State at Arbitrary Instants

Algorithm 1 provides samples of the stationary distribution at jump instants. However, this distribution may significantly differ from the distribution of the system at arbitrary instants. From the PASTA [11] property, this latter distribution can be sampled simply by pursuing each trajectory during a random time independent of the system, distributed according to the jump process.

3.8 Adding Files at Jump Times

In this section we show how the model can be modified to take into account additional features of a P2P system. First, the fraction of files lost when a user leaves the system need not be a constant. It can be a random fraction η of the files present before the jump, with η following an arbitrary distribution with mean $\frac{N(T_n^-)-1}{N(T_n^-)}$. Also, some users may connect the system to inject new files in the system. These files may be *intrinsically* popular. This can easily be incorporated in the model by injecting a random number of copies δ when a node brings one of these files: upon a join event,

$$x(T_n) = x(T_n) + \begin{cases} 0 & \text{with probability } 1-p \\ \delta(x(T_n), N(T_n)) & \text{with probability } p \end{cases}$$

A possible model for δ is the following. Let $M = C - d(x)$ be the total number of missing files. When a file D is injected it is chosen according to the popularity Zipf distribution $P_M(D)$. The number of injected copies is then a fixed number proportional to the number of free places $NC-x$: $\delta = P(D)\omega(NC-x)$, where $0 < \omega < 1$ is a constant. This jump preserves the ordering between both trajectories and can be incorporated in the algorithm simply by modifying function h. This illustrates the flexibility and simplicity of the perfect simulation algorithm.

4 Numerical Experiments

The Algorithm 1 has been implemented in Java. All experiments reported here are carried out on a 2GHz Pentium 4 with 1GB of RAM. We have chosen realistic parameters for a rather large P2P model resembling a typical file sharing system. Here, $C = 1000$, $N_{max} = 1000$, $\mu = \lambda = 10^{-5}$ (an average customer stays connected/disconnected for 24 hours); $\nu = \lambda/(CN_{max})$, so that the popularity plays the same role as the exogenous process for the connection rate of customers; $\sigma = 10^{-4}$ (an average customer emits a request every 3 hours); $\theta = 10^{-4}$, (3 hour time-to-live for copies of files) $\alpha = e^{-1}/C$ (the Zipf-like popularity distribution

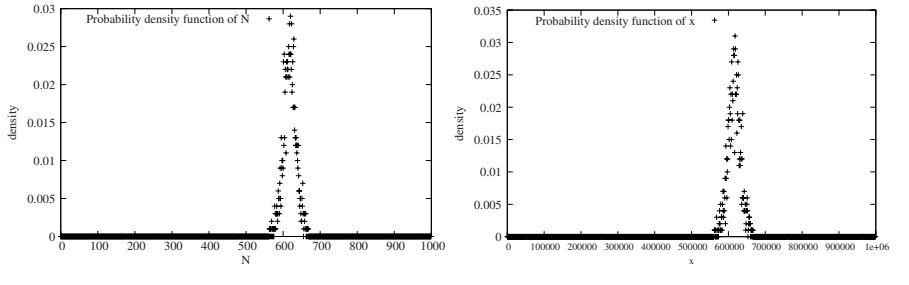

Fig. 4. The density of the stationary distribution of the number of customers (right) being connected and of the number of copies (left) in the hybrid model of a P2P system, as computed by perfect simulation

has parameter 1). The total computational time to get the asymptotic density of copies is about 2 hours, for a total of 10000 independent runs. Note that in Figure 4, the distributions of x and of N are centered on their average values and have rather small variances. In the following experiments, we will report only the average value of x or N (with confidence intervals).

Since θ is the only parameter that the system designer can control, we have computed the average value of the number of copies when θ varies. Figure 5 shows an interesting cut-off behavior: the number of files remains more or less constant until θ becomes very large ($\theta \approx 0.1$, meaning copies are removed after 10 seconds on average), where the number of copies drops to 0.

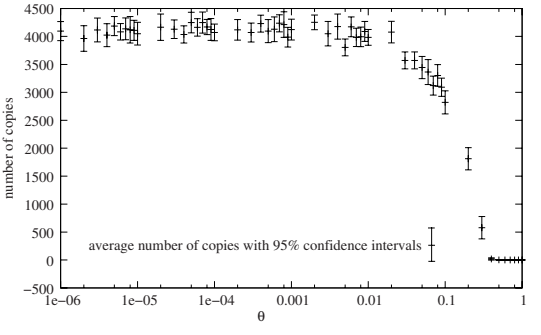

Fig. 5. Average number of copies as a function of θ

An important criterion for applicability of perfect simulation is the simulation time, which is random. The simulation time is directly related to the number of backward steps κ, which is doubled at each iteration in the algorithm. In order to evaluate experimentally how κ behaves when the state space of the system increases, we have run several simulation using the parameters $\lambda = \mu = \nu = \sigma = \theta = 1$ and we let the state space $N_{max} \times C$ grow from 0 to one million. As seen in Figure 6, the number of iterations κ of the simulation is almost deterministic and may only take two values as N and C grow. Also, κ is sub-linear in $N_{max} \times C$, which is quite good for perfect simulations.

5 Asymptotic Approximations

The solution x_N of the differential equation (3) admits an asymptote ℓ_N when t goes to infinity. The asymptote is the smallest solution of $f_N(x) = 0$. We consider the asymptotic *rate of convergence* of x_N to its asymptote, ℓ_N as

$$\gamma_N = \lim_{t \to \infty} \frac{f(x_N(t)) - f_N(\ell_N)}{x_N(t) - \ell_N} = \frac{df}{dx}(\ell_N) = -\alpha\sigma N e^{-\alpha \ell_N} + \sigma N - \frac{2\sigma \ell_N}{C} - \theta.$$

When the rate of convergence γ_N of $x_N(t)$ to its asymptotic value, is larger than the jump rate Λ, then in most cases, x will actually be very close to ℓ_N before

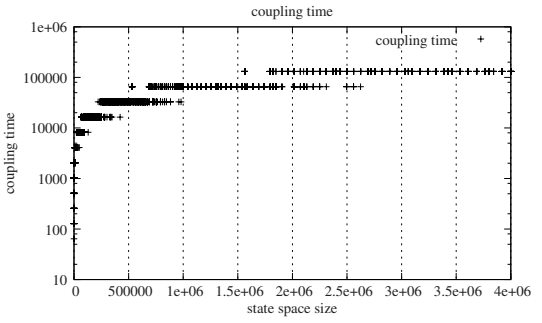

Fig. 6. Number of backward steps κ (in log scale for the number of iterations) as $N_{max} \times C$ vary

the next jump occurs. Therefore, one may disregard the transient behavior of x and let x jump directly from ℓ_{N_1} to ℓ_{N_2} whenever a jump from N_1 to N_2 occurs. This actually makes the system discrete since both x and N only take discrete values.

A visual illustration of this behavior is given in Figure 7 where two trajectories of the variable x are given with $\sigma = \Lambda$ and $\sigma = 10\Lambda$ respectively. In the second case, the rate of convergence of x to its asymptotic value is much larger than the jump rate and the trajectory of x looks like a staircase.

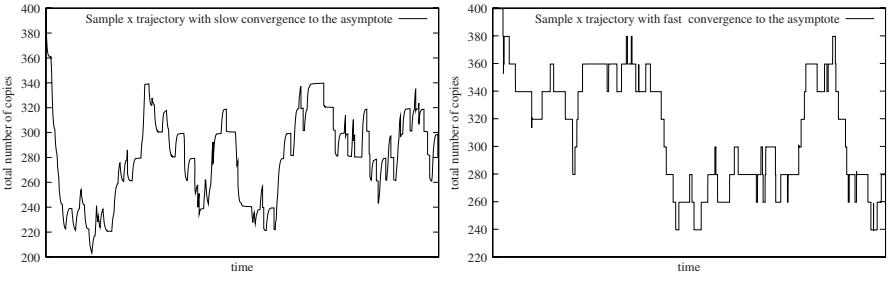

Fig. 7. Two behaviors of x with two rates of convergence

In that case, the stationary distribution Π of (N, x) can be approximated by

$$\Pi'(N = k, x = \ell_k) = \frac{1}{\Omega} \binom{N_{max}}{k} \frac{1}{\mu^k} \prod_{i=0}^{k-1} (\lambda + \nu \ell_i),$$

where the normalization constant Ω is such that all probabilities sum to one. Computing ℓ_k for each k is numerically easy using a Newton method. Here, the derivative of f at ℓ_k is large so that the computation can be done with a large precision quite fast. Then, generating samples according to the distribution Π' is rather simple and can be done using aliasing techniques [12]. We have

compared this approximation with the exact samples (N_1, x_1) computed by the simulation algorithm presented in Section 3. Numerical evidence show that when $\gamma_N > \Lambda/10$, the approximation becomes very good and can be used instead of our perfect sampling method. If $\gamma_N < \Lambda/100$, then the approximation is not valid any longer.

6 Conclusion

In this paper we have presented a simulation study of a stochastic hybrid systems providing guaranteed samples of a complex peer to peer system modeled by hybrid equations.

Our simulations are based on backward coupling techniques that provide statistical guarantees on its samples. We believe this technique is well adapted to stochastic hybrid systems because they enable us to manipulate continuous variables in a coherent way and because they make numerical computations of the behavior of the system possible because the memory space required and the coupling time are small with respect to the size of the state space.

Acknowledgement. We thank Rémi Bertin who implemented Algorithm 1 and provided insights on monotonicity issues.

References

1. Dai, J.: On positive harris recurrence of multiclass queueing networks: a unified approach via fluid limit models. Annals of Applied Probability **5** (1995) 49–77
2. David, R., Alla, H.: Discrete, Continuous, and Hybrid Petri Nets. Springer-Verlag (2004)
3. Asarin, E., Maler, O., Pnueli, A.: Reachability analysis of dynamical systems having piecewise-constant derivatives. Theoretical Computer Science **138** (1995) 35–65
4. Propp, D., Wilson, J.: Exact sampling with coupled Markov chains and application to statistical mechanics. Random Structures and Algorithms **9**(1) (1996) 223–252
5. Gaujal, B., Perronnin, F., Bertin, R.: Perfect simulation of stochastic hybrid systems with an application to peer to peer systems. Technical report, INRIA (2006)
6. KaZaA: (http://www.kazaa.com/)
7. Gnutella: (http://www.gnutella.com/)
8. Press, W.H., Teukolsky, S.A., Vetterling, W.T., Flannery, B.P.: Numerical Recipes in C. Cambridge University Press (1992)
9. Meyn, S.P., Tweedy, R.L.: Markov chains and stochastic stability. Springer Verlag (1993)
10. Foss, S., Tweedy, R.: Perfect simulation and backward coupling. Stochastic Models **14** (1998) 187–204
11. Baccelli, F., Brémaud, P.: Elements of queueing theory. Springer-Verlag (1994)
12. Walker, A.: An efficient method for generating random variables with general distributions. ACM Trans. Math. Software (1974) 253–256