Perfect simulation of stochastic hybrid systems with an application to peer to peer systems

Bruno Gaujal, Florence Perronnin, Remi Bertin

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Perfect simulation of stochastic hybrid systems with an application to peer to peer systems

Bruno Gaujal — Florence Perronnin — Rémi Bertin

N°6019

October 2006

Thème NUM

ISSN 0249-6399

Perfect simulation of stochastic hybrid systems with an application to peer to peer systems

Bruno Gaujal*†, Florence Perronnin‡† , Rémi Bertin§

Thème NUM — Systèmes numériques
Projet Mescal

Rapport de recherche n° ???? — October 2006 — 24 pages

Abstract: In this paper we present a rather general hybrid system made of deterministic differential equations and random discrete jumps. We then show how to construct a simulation of such a stochastic hybrid system that provides perfect samples of its asymptotic behavior. The applicability of the method is illustrated by showing how this framework can be used to model the Squirrel peer to peer system and by reporting a simulation study based on this approach. This paper provides both a general framework on how to carry simulation based experimental studies of large and complex hybrid systems and its application in the Squirrel model demonstrating how versatile and powerful this approach can be over a typical example.

Key-words: backward coupling, stochastic hybrid systems, simulation, P2P systems, fluid models

§ Supelec, Metz. This work was done while this author was visiting the Laboratory ID-IMAG under an INRIA internship
* INRIA. Email: Bruno.Gaujal@imag.fr
† Lab. ID-IMAG (CNRS, INPG, INRIA, UJF) 51, Av. J. Kunztmann, Montbonnot, France
‡ Université Joseph Fourier. Email: Florence.Perronnin@imag.fr

Unité de recherche INRIA Rhône-Alpes
655, avenue de l'Europe, 38334 Montbonnot Saint Ismier (France)
Téléphone : +33 4 76 61 52 00 — Télécopie +33 4 76 61 52 52

Simulation Parfaite de systèmes hybrides stochastiques avec application à un système pair-à-pair

Résumé : Dans cet article nous définissons un système hybride assez général composé d'équations différentielles déterministes et de saut aléatoires discrets. Nous montrons ensuite comment construire une simulation parfaite d'un tel système hybride, qui fournit des échantillons distribués exactement selon la mesure stationnaire. Nous illustrons ensuite cette méthode en proposant une évaluation de performances du système pair-à-pair Squirrel modélisé par un système hybride.

Mots-clés : Couplage vers l'arrière, systèmes hybrides stochastiques, simulation, systèmes pair-à-pair, modèles fluides

1 Introduction

Hybrid systems have become popular to model systems which dynamics is an interplay between a continuous one and a discrete one.

The main difficulty in dealing with them is that continuous systems are often modeled by differential equations while discrete ones are in general seen as automata. The mathematical tools to analyze the two parts (calculus versus algebra) as well as computer issues (numerical precision versus complexity) from both worlds are hardly compatible. However several recent breakthroughs such as [Alur et al., 2003, Tomlin et al., 2003, Girard, 2005] have demonstrated the usefulness and the interest from both theoretical and practical points of view, of mixing the continuous and the discrete paradigms.

For example 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 [Dai, 1995], continuous Petri nets [David and Alla, 2004], or timed automata [Asarin et al., 1995]. 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.

While hybrid systems often provide compact and elegant models for complex systems, the analysis is often difficult on a mathematical as well as on a numerical point of view and large hybrid systems are often considered computationally untractable. Simulation approaches are efficient alternative methods to estimate the behavior of such systems by providing samples distributed according to its asymptotic distribution, even when it is impossible to compute this distribution numerically. However, simulation has several drawbacks. First, simulations do not make any sense unless the system has ergodicity properties, which are sometimes difficult to assert. Second, even under 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 used backward coupling techniques ([Propp and Wilson, 1996]) to design a simulation algorithm to get *perfect sampling* (*i.e.* which distribution is not an approximation but the exact asymptotic distribution) of discrete time finite Markov chains. Their technique has been extended to Markov chains over continuous state spaces under uniform ergodicity conditions in [Foss and Tweedy, 1998] and we show how this is suitable for hybrid systems by using the embedded chain at jump instants.

More precisely, in this paper, we show that backward coupling technique can be used to simulate perfectly an *embedded Markov chain* in a stochastic hybrid systems. Moreover, whenever monotonicity of the evolution equations can be verified, the backward coupling can be done by simulating only a small number of trajectories, starting with the maximal and the minimal states. These simulations provide, in finite time, trajectories of an hybrid system which follow its asymptotic behavior. Our simulation algorithm is presented under a very general framework of an hybrid system made of ordinary differential equations coupled with a stochastic discrete process. An example of an hybrid model of the squirrel peer to peer system [Iyer et al., 2002] is developed in full details throughout the paper.

The goal of this paper is two-fold. First it provides a general framework on how to carry experimental studies based on perfect simulations, of large and complex hybrid systems. Second, it shows how fast, versatile, practical and powerful this approach can be over a typical example, namely the Squirrel peer to peer system, which exhibits many features which make hybrid systems difficult to study: a large state space, highly non-linear dynamics and an intricate interplay between its discrete and continuous parts.

The paper is organized as follows. In Section 2, we present the general hybrid system for which a perfect simulation algorithm will be constructed and constructs the corresponding embedded Markov chain. Section 3 describes the Squirrel model in details and provides the corresponding

hybrid model fitting the general framework. Section 4 shows how hybrid systems are amenable to perfect simulations. In the case where the dynamics are monotone, the general simulation algorithm is provided. Section 5 shows how to adapt the general algorithm to the Squirrel instance. In particular, monotonicity of the embedded Markov chains is not verified. This problem is bypassed by simulating lower and upper envelopes instead, which have the wanted monotonicity property and couple in finite time. Finally Section 6 reports a complete study of the Squirrel model based on a the perfect simulation tool implementing the envelope algorithm and Section 7 provides several improvements in terms of simulation time and fast approximations.

2 Stochastic hybrid systems

In this paper we consider a dynamic system $\mathcal{S}$ which state is a couple $(N(t), x(t))$ where $N(t)$ is the discrete component of $\mathcal{S}$ and can only take discrete values while x is the continuous part of $\mathcal{S}$. The variables $(N(t), x(t))$ will be left-continuous functions in the following.

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), $T_1, \dots T_n$. At time T_n,

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

where $\{\xi_n\}_{n\in\mathbb{N}}$ is a random process of *innovations* and ϕ describes the dynamics of N at jump instants.

As for the continuous part, $x(t)$ is governed by a stochastic process at jump instants

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

and by a deterministic differential equation between jump instants. In $[T_n, T_{n+1})$,

$$\frac{dx}{dt} = f(N(T_n), x, t). \tag{3}$$

Such dynamical equations are quite general and cover a wide range of cases. Most hybrid systems studied in the literature can be fitted within this framework. There is one major class that cannot be described in this framework, that with continuous stochastic dynamics, such as Brownian motions.

2.1 Embedded Markov chains

In the following, we will restrict our study to cases with an integer discrete part $N(t) \in \mathbb{N}$, a real continuous part $x(t)$ which is mono-dimensional ($x(t) \in \mathbb{R}$) and a differential equation $\frac{dx}{dt} = f(N(T_n), x, t)$ that admits a unique solution for all possible initial states (N_0, x_0), denoted $F(N_0, x_0, t)$.

Furthermore, the stochastic innovations ξ_n are iid and the jump process T_i is a Poisson process (its rate may vary and may depend on the state at the previous jump). We denote by $\mathcal{D} \subset \mathbb{N} \times \mathbb{R}$ the set of all reachable states for the couple (N, x).

Lemma 1. *Under the foregoing assumptions $S_n = (N(T_n), x(T_n))$ is a homogeneous Markov chain over the continuous domain, $\mathcal{D} \subset \mathbb{N} \times \mathbb{R}$.*

Proof. Using the previous notations,

$$N(T_n) = \varphi(N(T_{n-1}), F(N(T_{n-1}), x(T_{n-1}), T_n - T_{n-1}), \xi_n), \tag{4}$$
$$x(T_n) = h(N(T_{n-1}), F(N(T_{n-1}), x(T_{n-1}), T_n - T_{n-1}), \xi_n). \tag{5}$$

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 $T_n - T_{n-1}$, 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 the Markov chain as $S_n = \Phi(S_{n-1}, \xi_n)$. For simplicity, we also denote $\Phi(S, \xi_1, \ldots, \xi_n) \stackrel{\text{def}}{=} \Phi(\cdots \Phi(\Phi(S, \xi_1), \xi_2), \cdots)$.

Figure 1 provides an illustration of the evolution of such a stochastic hybrid system Φ.

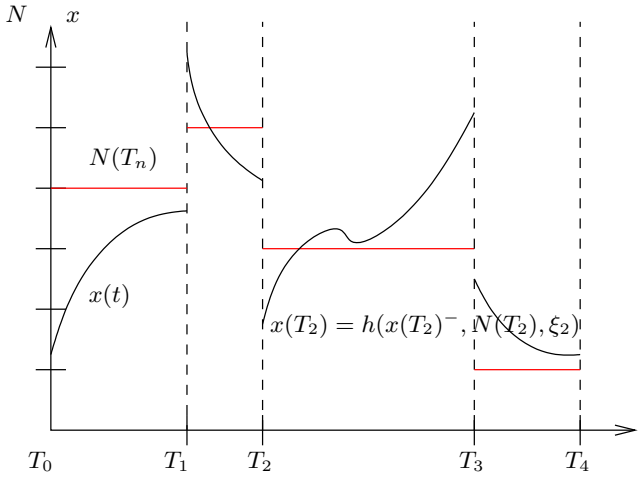

Figure 1: Hybrid system Φ

3 Peer to peer systems

Throughout the paper, we will use the example of an hybrid system to illustrate and to apply the general framework. The example models a peer to peer system. We believe that the example provides a good illustration of the applicability of the general framework, but also presents a significant interest of its own, providing interesting insights on the evolution of peer to peer systems. Indeed, in Section 6 we use our approach to compute detailed performance metrics of complex P2P systems, such as the extinction probability or the probability that files are present in the system.

In this section we briefly introduce peer-to-peer (P2P) systems, then we describe in more detail the running example, namely Squirrel and finally we show how it can be modeled as a hybrid system.

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 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 $\mathcal{F}$. (the notion of "closeness" depends on the desired performance metric, typically A and B may be on the same local network, or are geographically close). 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,
- 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.

In the sequel we study one specific peer-to-peer system : a P2P cooperative Web cache called Squirrel.

3.1 The Squirrel model

The P2P paradigm is similar to the caching paradigm : keep copies of popular files in "servers" that are closer to the end-users than the origin servers. In the context of Web caching, the "servers" may be a local proxy or even a local cache on each client's machine. The Squirrel system [Iyer et al., 2002] was designed to enable users to share their local Web caches to make a distributed P2P caching system.

Squirrel system description

We now briefly describe how Squirrel works. The interested reader can refer to [Iyer et al., 2002] for a complete description of the protocol. In the following, a *node* will denote a user's machine. Squirrel is based on the Pastry routing protocol [Rowstron and Druschel, 2001]. Nodes emit *requests* for web objects (typically files). In the following we assume that the request rate σ is constant and identical for each node. When a node issues a request for a Web object, the local cache is checked first. If the object is not found ("cache miss") the client tries to locate the file on another node of the Squirrel system as follows. The URL of the object is mapped to a number called "object-Id" using a hash function. The request is then forwarded using Pastry to the node whose node-Id is numerically closest to the object-Id. This latter node is called "home node" for this object. The home node acts as a classical proxy cache for this object : it checks its own local cache for the object. In case of a cache hit the object is sent to the requesting node. In case of a cache miss, the home node downloads the object from the origin Web server, stores a copy locally and sends it to the requesting node. The hash function is chosen so as to ensure load balancing in the system.

Finally, nodes can join and leave the system at anytime, for instance due to peer disconnection or to software crash. When a node A leaves the system, the objects that were cached in A are lost for the whole system. When a node B joins the system, it becomes *de facto* an home node for a number of objects. These objects may have been previously requested and cached in their previous home nodes which are neighbors of B in the node-Id space. To avoid subsequent requests for these files to result in a cache miss, B can download these files from its neighbors when joining the system. We assume that B does not bring any exogenous content to the global system when joining in. This assumption is discussed in [Clévenot-Perronnin, 2005].

Modeling Squirrel with a hybrid system

We now introduce a fluid-discrete model for Squirrel. This model was introduced and experimentally validated in [Clévenot-Perronnin, 2005]. Let us now describe the model and show that it belongs to the class of hybrid systems defined in Section 2.

We assume that nodes join and leave independently of each other. This process is the discrete component of the model. Let $N(t)$ denote the number of connected nodes. We assume the system is closed, i.e., there may exist only N_{max} nodes. Each node leaves the system with a constant rate μ. Each node joins the system with a constant rate λ. Note that this process is a classical continuous time Markov chain corresponding to the Erhenfest urn model displayed in Figure 2.

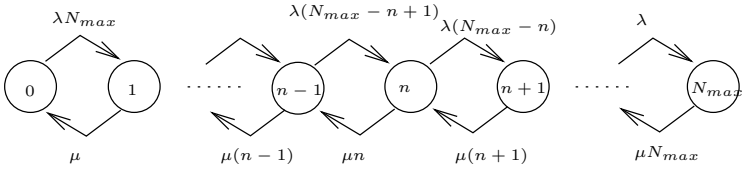

Figure 2: The infinitesimal generator for the Markov process N

We now model the files dynamics in the Squirrel system. The total amount of cached documents is modeled with fluid. The justification for this continuous model is that files may be partially transferred and also that these transfers occur on a much faster time scale than node events. More

details on this fluid assumption can be found in [Clévenot-Perronnin, 2005]. Specifically, let $x(t)$ denote the total amount of fluid, i.e. the total number of files in the Squirrel system. This process is the continuous component of our hybrid system. Again, we assume a closed set of objects, i.e. $x(t)$ is bounded by the maximum number of existing files C.

We now describe the dynamics of $x(t)$ and show how they can be modeled by Equations (2)-(3). Let us first determine the differential equation which describes the evolution of fluid between two consecutive jumps of $N(t)$. The amount of fluid increases whenever a client brings a new file in the system. That is, upon a cache miss, the home node downloads the requested file and stores it for future requests, which increases $x(t)$. This process is proportional to the total request rate $\sigma \times N(t)$ and to the miss probability. Using the well-known Zipf-like popularity distribution of Web objects [Breslau et al., 1999], we show in Appendix A.1 that the miss probability is of the form $1 - \left(\frac{x(t)}{C}\right)^{\alpha}$ with $\alpha > 0$ close to 1. Finally the fluid decreases when copies become stale or obsolete, typically after an average time-to-live $1/\theta$. As a result, the function f in differential equation (3) becomes homogeneous in time and writes:

$$\frac{dx}{dt} = f(N(T_n), x(t)) = \sigma N(T_n) \left(1 - \left(\frac{x(t)}{C}\right)^{\alpha}\right) - \theta x(t). \tag{6}$$

When $\alpha = 1$ this ODE admits a closed-form solution and the stationary hit probability of the Squirrel system may be analytically expressed in closed form [Clévenot and Nain, 2004]. However, if $\alpha \neq 1$ this equation admits a unique solution (see Appendix A.3) with no known closed-form and a numerical resolution is needed.

We now turn to the evolution of the fluid at jump instants T_n. When a node leaves the system it takes away all the documents it was responsible for. With the load balancing property of Pastry, the number of lost documents is a proportional fraction of the total fluid. Therefore if the event at T_n is a departure then

$$x(T_n) = \frac{N(T_n^-) - 1}{N(T_n^-)}\, x(T_n^-). \tag{7}$$

When a node joins the system, it does not bring exogenous files with him upon its arrival. The increase of the number of files will come from the future downloads of the newcomer:

$$x(T_n) = x(T_n^-). \tag{8}$$

Figure 3 shows a typical trajectory of the evolution of both N and x over time. By looking closely to the evolution of $x(t)$, one may notice down jumps corresponding to departures of customers and cusps (instants where x remains continuous but is not differentiable), corresponding to customer arrivals.

3.2 Constructive Markov Chain

In order to deal with the continuous time Markov chain driving N, we will use a uniformized discrete version of it using the uniformization constant $\Lambda = (\lambda + \mu)N_{max}$. The transition matrix of this embedded discrete time Markov chain is given in Figure 4, where the rates are normalized: $\lambda' = \lambda/\Lambda$ and $\mu' = \mu/\Lambda$.

Note that in this example, N does not depend on x so that its evolution can be computed in isolation. Also note that N being a one-dimensional random walk, it is an ergodic chain. Its stationary distribution π_E is easy to compute using the close form formula:

$$\pi_E(N = k) = \frac{\left(\frac{\lambda}{\mu}\right)^k C_k^{N_{max}}}{\left(1 + \frac{\lambda}{\mu}\right)^{N_{max}}} \quad \forall\, 0 \leq k \leq N_{max}. \tag{9}$$

However, the asymptotic behavior of the continuous part, x is more difficult to compute for several reasons. First, the differential equation (6) does not have a closed form solution when α is not an

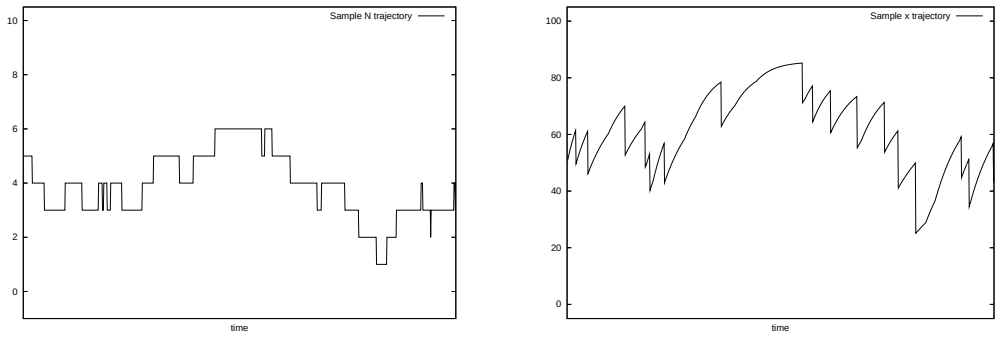

Figure 3: A trajectory of N and x respectively for the Squirrel model, with $C = 100, N_{max} = 10, \lambda = \mu = 1, \theta = \sigma = 1, \alpha = 1$.

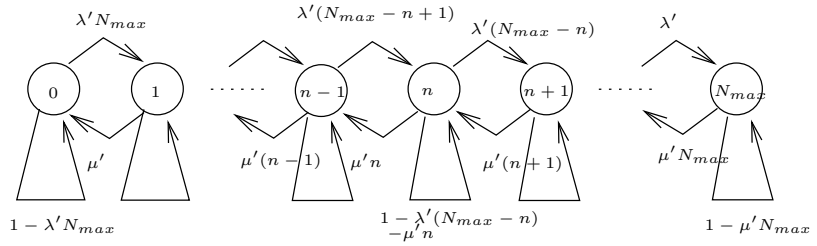

Figure 4: The transition kernel of the uniformized Markov chain

integer. Second, the stochastic jumps and the dependence with respect to N make the asymptotic behavior of x hard to grasp, even numerically.

In this paper, we will construct a perfect simulation algorithm based in the general framework presented in Section 2. Here is the functional construction of the next state $N(T_n), x(T_n)$ of the chain using $N(T_{n-1})$ and $x(T_n^-)$. This corresponds to defining the functions φ and h.

1. Generate ξ uniformly over $[0, 1]$.
2. Generate τ exponentially distributed with parameter Λ.
3. - If $\xi > 1 - \lambda'(N_{max} - N(T_{n-1}))$, then the next event is a *customer arrival:* $N(T_n) = N(T_{n-1}) + 1$ and $x(T_n) = x(T_n^-)$
   - If $\xi < \mu N(T_{n-1})$, then the next event is a *customer departure:* $N(T_n) = N(T_{n-1}) - 1$ and $x(T_n) = x(T_n^-) - x(T_n^-)/N(T_{n-1})$;
   - Otherwise, this is a *null event*, *i.e.* no customer arrives nor leaves and the system is left unchanged at time T_n: $N(T_n) = N(T_{n-1})$ and $x(T_n) = x(T_n^-)$.
4. Integrate the differential equation over span τ: $x(T_{n+1}^-) = F(x(T_n), N(T_n), \tau)$.

This defines Φ as $(N(T_n), x(T_n)) = \Phi(N(T_{n-1}), x(T_n^-), \xi, \tau)$.

In most cases, the integral F of the differential equation cannot be computed under closed form. In this case, we use a classical Runge-Kutta numerical integrator [Press et al., 1992] to compute $x(T_{n+1}^-)$.

4 Perfect simulation of Markov chains over continuous state spaces

As mentioned in the introduction, the goal of this paper is to get guaranteed estimates of the asymptotic behavior of the hybrid system $\mathcal{S}$.

In order to do so, we will provide a *perfect simulation algorithm* of the Markov chain $S_n = (N(T_n), x(T_n))$. Let us first recall the main ingredients for perfect sampling of finite Markov chains.

Let $P^n(s, A)$ denote the transition law of n steps of the chain S_n (This is the probability $\mathbb{P}(S_n \in A | S_0 = s)$). When it exists, its stationary measure satisfies $\Pi(A) = \int_{\mathcal{D}} P^1(s, A)\Pi(ds)$, for all measurable set A in $\mathcal{D}$.

Of course, 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 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 (4),(5) 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. Therefore, it is difficult to estimate the bias between $P^n(s_0, A)$ and $\Pi(A)$ in general.

Here, we will show how to compute samples 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 [Propp and Wilson, 1996] 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 and look at what happens at time 0. If all the trajectories coincide at time 0, 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.

This idea was extended to Markov chains over continuous state spaces in [Foss and Tweedy, 1998]. Here is their main theorem, adapted to our notations.

Theorem 2 (Vertical backward coupling [Foss and Tweedy, 1998]). *If the Markov chain is uniformly ergodic, i.e. if there exists a non-trivial measure φ over $\mathcal{D}$, some $m > 1$ and $0 \leq \beta \leq 1$ such that*

$$\forall x \in \mathcal{D} \quad P^m(x, \cdot) \geq \beta\varphi(\cdot),$$

then, 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 $s \in \mathcal{D}$, $\Phi(s, \xi_{-K}, \ldots, \xi_{-1}, \xi_0) \sim \Pi$.

Note that K is defined by using *backward* iterations of function Φ. Instead of starting from one initial state at time 0 and make the chain evolve from that point on, here, the iterations start at time -K using all possible initial states and evolve up to time 0 where they should all coincide (or couple). This is the reason why this technique is also called "coupling from the past".

Also note that Theorem 2 does not provide directly an effective algorithm for computing perfect samples $\Phi(s, \xi_{-K}, \ldots, \xi_{-1}, \xi_0)$ since the definition of K is not constructive (the state space is continuous) and because m may be too large to be amenable to any efficient computation.

4.1 Monotonicity issues

From now on, we assume that the domain $\mathcal{D}$ admits a component-wise ordering: $(N, x) \leq (N', x')$ if $N \leq N'$ and $x \leq x'$. If the construction Φ of the chain $S_n = \Phi(S_{n-1}, \xi_n)$ is monotone in its first coordinate, ($\forall y, \quad \Phi(u, y) \leq \Phi(v, y)$ whenever $u \leq v$) then, one can characterize the backward coupling time in terms of maximal and minimal states.

Since Φ is a combination of several functions, Φ is non-decreasing if F, h and φ are all non-decreasing in their first two coordinates N and x. Finally note that F is monotone in its first two coordinates as soon as f is monotone in N and the solution of the differential equation is monotone in its initial condition.

Theorem 3 ([Foss and Tweedy, 1998]). *Let MAX (resp. MIN) be the set of all maximal (resp. minimal) elements in $\mathcal{D}$ for the order $\leq$. Then,*

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

is a vertical backward coupling time of the chain.

Note that if MAX and MIN are finite sets, then this definition of K is constructive so that one can use backward coupling to design a perfect simulation algorithm of the Markov chain. The general algorithm for perfect simulation of a monotone hybrid system is given in Algorithm 1.

Algorithm 1 Backward-coupling simulation (monotone version)

```
n = 1;
ξ[n]:=Random_event;{array ξ stores the backward sequence of events }
repeat
  n = 2n;
  for all s ∈ MAX ∪ MIN do
    y(s) := s {Initialize all trajectories at time −n}
  end for
  for i = n downto n/2 + 1 do
    ξ[i]=Random_event; {generates all events from time −n + 1 to n/2 + 1}
  end for
  for i = n downto 1 do
    for all s ∈ MAX ∪ MIN do
      y(s) := Φ(y(s), ξ[i])
      {y(s) is the state at time −i of the trajectory starting in s at time −n}
    end for
  end for
until All y(s) are equal
return y(s)
```

The algorithm 1 uses a time doubling technique for reducing the total computation time, as suggested in [Propp and Wilson, 1996]. Its time complexity is $O((2K + c_\Phi)(|MAX| + |MIN|))$, where c_Φ is the total complexity of computing Φ over the whole simulation.

4.2 Implementation issues

The perfect simulation theorem of Propp and Wilson for finite Markov chain can be almost directly translated into an implemented algorithm for sampling the stationary distribution of the chain. This is not really the case for Theorem 2 nor for Theorem 3 for two reasons. First, the vertical coupling times may be too large for a computer program to ever converge in a reasonable time. Second, the definition of the vertical coupling times is based on testing the equality of real numbers, which is always difficult to do in a computer program.

Both issues can be addressed by choosing an arbitrary precision ε and by stopping the algorithm as soon as all the trajectories are contained in a ball of size ε.

The output of the algorithm, in the monotone version is an upper and a lower bound on the cumulative distribution function of Π, with precision ε.

In the experiments provided below (Section 6), this method improves the convergence time by several orders of magnitude, even when ε is very small. An explanation of this behavior is also provided.

5 Application to Squirrel

In this section, we show how the general monotone algorithm 1 can be adapted to our example.

5.1 The Squirrel MC is uniformly ergodic

In the Squirrel example, N follows a birth and death process. If a large number of consecutive departures occurs, then the number of customers becomes $N = 0$ and the number of files must also be null in that case: $x = 0$. Moreover, for all n, $\mathbb{P}(N(T_n) = 0|N(0) = N_0) \geq \mathbb{P}(N(T_n) = 0|N(0) = N_{max})$. Now, $\mathbb{P}(N(T_n) = 0|N(0) = N_{max}) \geq \frac{\mu^n N_{max}!}{\Lambda^n}$ if $n \geq N_{max}$. It suffices to choose $m = N_{max}$, $\varphi = \mathbb{1}_{(0,0)}$ and $\beta = (\mu/\Lambda)^{N_{max}}$ in the definition of uniform ergodicity to satisfy the condition of uniform ergodicity required in Theorem 2. Moreover, the Squirrel MC is irreducible on its final class given in Appendix B. Therefore, the Squirrel MC, $S_n = (N(T_n), x(T_n))$ admits a vertical backward coupling time.

5.2 The Squirrel MC is not monotone

The next thing is to check whether the embedded Markov chain is monotone. Actually, as shown below, the Markov chain is not monotone.

We recall that the Markov chain is such that :

1. if $\xi > 1-\lambda'(N_{max}-N(T_{n-1}))$, then the next event is a customer arrival: $N(T_n) = N(T_{n-1})+1$ and $x(T_n) = x(T_n^-)$;
2. if $\xi < \mu N(T_{n-1})$, then the next event is a customer departure $N(T_n) = N(T_{n-1}) - 1$ and $x(T_n) = x(T_n^-) - x(T_n^-)/N(T_{n-1})$;
3. otherwise, this is a null event, *i.e.* no customer arrives not leaves and the system is left unchanged at time T_n.

As for the evolution between jumps, it follows the differential equation (6). To test if the chain is monotone in its two components, 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 (6) 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 $\xi > 1 - \lambda' N_{max}$, which corresponds to arrivals or null events, whatever the value of N. In that case, if $N_1(T_{n-1}) > N_2(T_{n-1})$ then $N_1(T_n) \geq N_1(T_{n-1}) \geq N_2(T_{n-1}) + 1 \geq N_2(T_n)$ since it may be that ξ corresponds to an arrival for S_2 and a null event for N_1. As for the files, such events do not change their values: $x_1(T_n) = x_1(T_n^-) \geq x_2(T_n) = x_2(T_n^-)$.

When $\xi < 1 - \lambda' N_{max}$, 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.

5.3 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. These envelopes are constructed to keep monotonicity along all trajectories. However, they are not Markovian in isolation.

In the following, we will consider an upper and a lower envelopes, $\overline{S}_1 = (\overline{N_1}, \overline{x_1})$ and $\underline{S}_2 = (\underline{N_2}, \underline{x_2})$, respectively, of all trajectories of the Markov chain, starting from all possible states in $\mathcal{D}$. 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 over all events which cannot cause a swap of the ordering between the two envelopes. Whenever such an event occurs, then, here is the way both envelopes evolve.

Let $N_3 = \lfloor \xi/\mu' \rfloor$, 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. Note that since ξ is a swapping event, then $\overline{N_1}(T_{n-1}) > N_3 \geq \overline{N_2}(T_{n-1})$. Now, the smallest value of N after event ξ is larger than $\underline{N_2}(T_{n-1})$, the smallest value of x is larger than $\underline{x_2}(T_n^-)_{\frac{N_3}{N_3+1}}$, while the largest possible value of N after event ξ is smaller than $\overline{N_1}(T_{n-1}) - 1$ and the largest possible value of x after event ξ is smaller than $\overline{x}_1$. Therefore, we set $\overline{S}_1(T_n) = (\overline{N_1}(T_{n-1}) - 1, \overline{x}_1)$ and $\underline{S}_2(T_n) = (\underline{N_2}(T_{n-1}), \underline{x_2}(T_n^-)_{\frac{N_3}{N_3+1}})$. Therefore, an event that would correspond to a departure in $\overline{S_1}$ and a null event in S_2 becomes a "dummy" departure for $\underline{S_2}$ and a "dummy" arrival in S_1.

The upper envelope is not a Markov chain, neither is the lower one. However, the couple $(\overline{S}_1(T_n), \underline{S}_2(T_n))$ is a Markov chain over the continuous state space $\mathcal{D} \times \mathcal{D}$. The construction of the envelopes $(\overline{S}_1(T_n), \underline{S}_2(T_n))$ 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.

$$\overline{S}_1(T_n) = \Gamma_1\Big(\overline{S}_1(T_{n-1}), \underline{S_2}(T_{n-1}), \xi_n, \tau_n\Big), \tag{10}$$

$$\underline{S_2}(T_n) = \Gamma_2\Big(\overline{S}_1(T_{n-1}), \underline{S_2}(T_{n-1}), \xi_n, \tau_n\Big). \tag{11}$$

Note that by construction of Γ_1, Γ_2, The envelopes have been built such that $\overline{S_1}(t)$ stays above $\underline{S_2}(t)$ for all time $t \geq 0$ as soon as $\overline{S_1}(0)$ is above $\underline{S_2}(0)$. So the envelopes have a monotone behavior.

Also note that by construction, for all initial state $S(0) = (N, x)$ and all time t,

$$\overline{S_1}(t) \geq S(t) \geq \underline{S_2}(t).$$

5.4 Perfect Simulation Algorithm for Squirrel

The following theorem is the theoretical foundation of the perfect simulation algorithm for the Squirrel model.

Theorem 4. *Envelopes $\overline{S_1}(T_n)$ and $\underline{S_2}(T_n)$ admit a coupling time, K'':*

$$K'' = \min\left\{n : \Gamma_1\big((N_{max}, C), (0,0), \xi_{-K}, \ldots, \xi_{-1}, \xi_0\big) = \Gamma_2\big((N_{max}, C), (0,0), \xi_{-K}, \ldots, \xi_{-1}, \xi_0\big)\right\}.$$

Furthermore, K'' 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 $(\overline{S}_1(T_{n-1}), \underline{S}_2(T_{n-1}))$ 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 $\overline{S_1}(t) \geq S(t) \geq \underline{S_2}(t)$ 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 $\overline{S_1}(0) = S(0) = \underline{S_2}(0)$ by definition of K''.

□

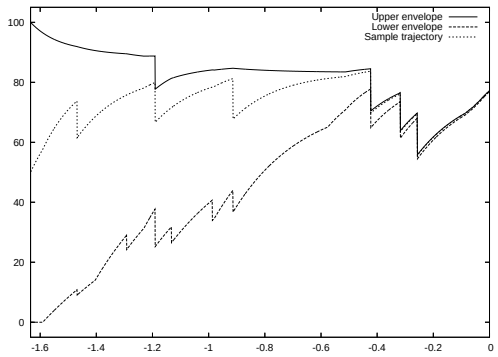

Figure 5: Trajectories of the two envelopes $\overline{S_1}, \underline{S_2}$ and a trajectory of the Markov chain S. One can notice several "dummy" departures on S_2 and several "dummy" arrivals on S_1. The sample of S is a real trajectory, which stays within the upper and lower envelopes.

Algorithm 2 is the adaptation of the general algorithm 1 to the Squirrel case. The main difference is the stopping test. Here we decide to stop as soon as the upper and lower 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 6). This means that the average vertical coupling time is huge which makes the backward coupling technique of no practical use. Testing for a small gap between both envelopes provides a strong control on the simulation time (see Section 7) and guarantees on the intervals on the measures as seen in the following.

Algorithm 2 Backward simulation for Squirrel

$n = 1$;
repeat
  $n = 2n$;
  $\overline{S_1} := (N_{max}, C)$,$\underline{S_2} := (0,0)$ {Initialize the two envelopes at time $-n$}
  for $i = n$ downto $n/2 + 1$ **do**
    $\xi[i]$:=Random(Uniform over $[0,1]$); $\tau[i]$:=Random(exponential with rate Λ) {generates all events from time $-n+1$ to $\frac{n}{2}+1$ and the inter-jump times}
  end for
  for $i = n$ downto 1 **do**
    $\overline{S_1} := \Gamma_1(\overline{S_1}, \underline{S_2}, \xi[i], \tau[i])$, $\underline{S_2} := \Gamma_2(\overline{S_1}, \underline{S_2}, \xi[i], \tau[i])$
  end for
until $\overline{S_1} \to N = \underline{S_2} \to N$ and $\overline{S_1} \to x - \underline{S_2} \to x \leq \varepsilon/3$
return $\overline{S_1}$

In Figure 5, we display a perfect simulation of $\overline{S_1}, \underline{S_2}$ together with an arbitrary trajectory of the Squirrel Markov chain S, starting in $N_{max}/2, C/2$. First note that S stays within $[\underline{S_2}, \overline{S_1}]$ at all times. Also note that several dummy events on $\overline{S_1}, \underline{S_2}$ are visible. They all correspond to discontinuous jumps of $\underline{S_2}$ and cusps of $\overline{S_1}$. At those instants, S may or may not jump, depending on the value of ξ.

Algorithm 2 is used to simulate the hybrid squirrel model. The outputs of the i-th run of the algorithm are numerical approximations of the couples $(\overline{S}_1^i = (N_1^i, x_1^i), \underline{S_2}^i = (N_2^i, x_2^i))$. Using a small step h for numerical integration of the differential equations (*i.e.* such that the error

$\epsilon = h\sigma N + \alpha\tau \frac{\sigma^{\alpha+1} g N^{\alpha+1}}{C^\alpha} h^\alpha \leq \varepsilon/3$, see Appendix A.5), then the outputs of the algorithm are such that the real values $x_1^i - x_2^i \leq \varepsilon$.

Let us denote by π_N and π_x the marginal distribution of N and x respectively, for the unknown distribution Π. According to Theorem 4, they satisfy for all runs i:

$$P(N_1^i = k) = P(N_2^i = k) = \pi_N(k), \tag{12}$$

$$P(x_1^i \leq z | N_1^i) \leq \pi_x([0,z]|N_1^i) \leq P(x_2^i \leq z | N_2^i) \leq P(x_1^i - \varepsilon \leq z | N_1) = P(x_1^i \leq z + \varepsilon | N_1^i). \tag{13}$$

From these equations, one can compute confidence intervals using the following procedure. Let E be an interval $E = [a,b] \subset [0,C]$ for which we want to compute $\pi_x(E)$ with confidence c. The problem is to find an interval I such that $P(\pi_x(E) \in I) \geq c$.

We denote by E_ε the augmented interval , $E_\varepsilon \stackrel{\text{def}}{=} [a-\varepsilon, b+\varepsilon]$. For a total of M runs, let $\hat{p_1} = 1/M \sum_{i=1}^M \mathbf{1}\{x_1^i \in E \wedge x_2^i \in E\}$ and $\hat{p_2} = 1/M \sum_{i=1}^M \mathbf{1}\{x_1^i \in E_\varepsilon \vee x_2^i \in E_\varepsilon\}$. Using Equation 13, There exists an unbiased estimator $\hat{p}$ of $\pi_x(E)$ such that $\hat{p_1} \leq \hat{p} \leq \hat{p_2}$.

Now, the central limit theorem gives the following confidence interval

$$I = \left[\hat{p_1} - \frac{\beta_c v}{\sqrt{M}}, \hat{p_2} + \frac{\beta_c v}{\sqrt{M}} \right],$$

where β_c is half the c-percentile for the normal distribution $\mathcal{N}(0,1)$ and v is the variance of a Bernoulli distribution with probability $\pi_x(E)$: $v = \sqrt{\pi_x(E)(1-\pi_x(E))} \leq 1/2$.

Note that the size of the confidence interval can be bounded by $\frac{\beta_c}{\sqrt{M}} + e$ where the error e is made by substituting $\hat{p}$ by the empirical measures $\hat{p_1}$ and $\hat{p_2}$. If we denote the cumulative distributions $\pi_1([u,v]) \stackrel{\text{def}}{=} P(u \leq x_1^i \leq v | N_1^i)$, $\pi_2([u,v]) \stackrel{\text{def}}{=} P(u \leq x_2^i \leq v | N_2^i)$, that do not depend on i, then

$$e \leq \sup_{z>0} \pi_1([z, z+\varepsilon]) + \sup_{z>0} \pi_2([z, z+\varepsilon]).$$

It should be noted that since the only atom of all the distributions is in 0 and the supremum is taken over all $z > 0$, The error e goes to 0 when ε goes to 0.

So the size of the confidence interval decreases with $1/\sqrt{M}$, as usual with unbiased estimates and with the error e which is an empirical function of the precision ε.

6 Numerical experiments

In this section we report the results of several simulations made on the Squirrel model. We report the computation times as well as the performance indexes measured over the system. All experiments are carried out on a 2GHz Pentium 4 with 1GB of RAM.

We choose a realistic instance of the Squirrel model with

- $C = 10000$, which is the maximal number of files that can be present simultaneously over the system;
- $N_{max} = 1000$, which is the maximal number of participants to our Squirrel system;
- $\mu = \lambda = 10^{-4}$ (on average, each customer stays connected/disconnected for 2.7 hours);
- $\sigma = 10^{-2}$ (on average, each connected customer emits a request every 1.6 minutes);
- $\theta = 10^{-2}$, which corresponds to an 1.6 minutes time-to-live;
- $\alpha = 0.4$, which corresponds to a Zipf-like popularity distribution with parameter 0.8.

On Figures 6 and 7 we show the density of N and x, respectively. To obtain these figures we ran algorithm 2 or equivalently the "N-first" algorithm (see section 7 10000 times in approximatively 4 minutes and 30 seconds. The ten thousand resulting samples of N and x are distributed according to the stationary distribution. On Figure 6 we observe the density of the Engset model, which is highly centered around its mean value. On Figure 7 we observe that the distribution of x also exhibit a very narrow peak around its mean value. This shows that x has very little variance and is well described by its mean. For this reason in the following experiment we restrict our attention to the mean value of x.

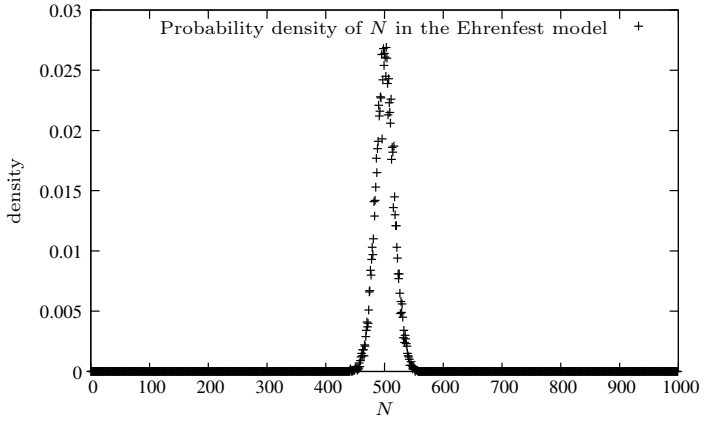

Figure 6: The empirical density of the distribution of N, as given by 10000 simulations.

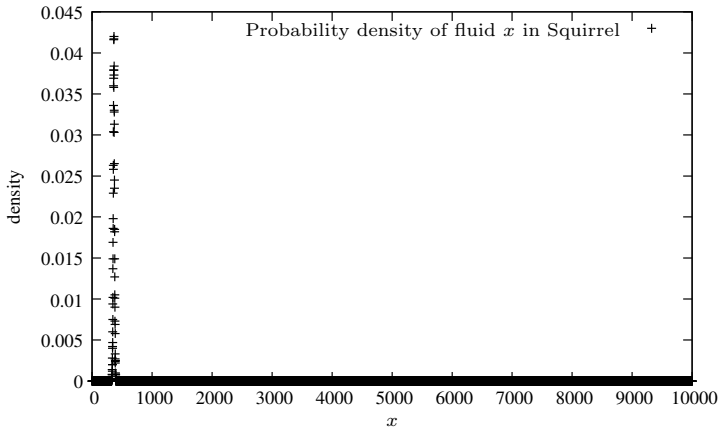

Figure 7: The empirical density of the distribution of x, as given by 10000 simulations.

We then chose to measure H, the asymptotic *hit probability*. This is the probability that a given file (chosen uniformly over all files) is present in the system, under its stationary regime. As shown in Appendix A.1,

$$H = \lim_{t\to\infty} \left(\frac{x(t)}{C}\right)^{\alpha}.$$

This quantity is the typical performance measure of a caching system like Squirrel.

From Figure 7 shows that for the Squirrel system described above, there are on average 500 simultaneously connected nodes and the average number of cached documents is 363.78, which corresponds to an average hit probability of 0.26.

We now show how the hit probability varies with regard to its design parameters. Note that θ is the only parameter over which the system designer has some control (the time-to-live of cached

objects may be set to an optimal default value). Figure 8 displays the hit probability (and not the amount of fluid x) as θ varies from 10^{-7} to 10 for the simulation setup described above.

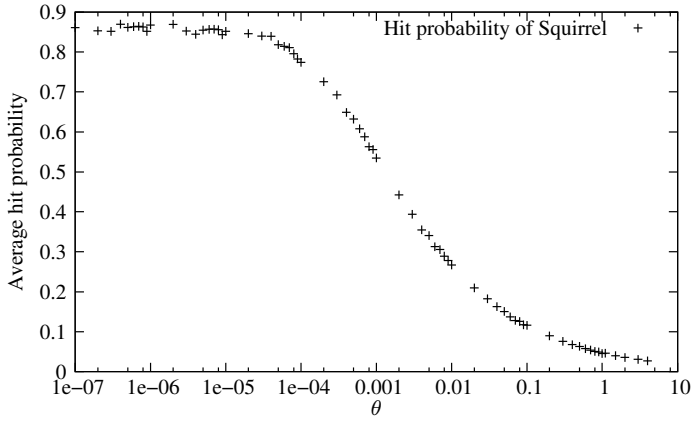

Figure 8: The hit probability as θ varies (over a log scale).

Note that θ evolves over a logarithmic scale. We observe that the hit probability can vary greatly with θ (from 0 to 0.9). However, only the $[10^{-4}, 10^{-2}]$ range of θ has a real impact on the hit probability: the hit probability drops sharply as soon as $\theta > 10^{-4}$ and is below 0.2 as soon as $\theta > 10^{-2}$. A 0.5 hit probability is achieved with $\theta > 10^{-3}$. This means that for the system described above, the time-to-live of objects should not be of the order of 15 minutes. We also note that the smaller θ the better hit probability. However, a small θ means a increased risk of using out-of-date cached copies. To limit this drawback most caching systems use a default 24 hours time-to-live which corresponds to $\theta \approx 10^{-5}$. This values seems to be a minimum value to ensure acceptable average staleness.

7 Coupling time improvements in Squirrel

The most important feature of perfect simulation is the coupling time. Indeed, perfect simulation is not usable for sampling the stationary distribution if the coupling time is too large. Two factors play a major role in the coupling time.

The first factor is the convergence rate of the stochastic discrete part, N. Since, the underlying Markov Chain is the Erhenfest model, the stationary distribution is not spread and both the upper and lower envelopes have their respective discrete part converge fast towards middle values. The second factor is the ratio between the convergence rate of the differential equation to its asymptotic value and the jump rate of the discrete part. The next two sections address these two points.

7.1 Let N converge first

One can take advantage of the fact that the evolution of N is independent of the evolution of x, so that its stationary distribution can be computed in isolation.

The idea is to first compute a sample N_∞ for N, distributed according to its marginal stationary measure, and then to simulate for the global system (N, x) starting with the initial states (N_∞, C) and $(N_\infty, 0)$, instead of (N_{max}, C) and $(0, 0)$. This is valid using the next lemma.

Theorem 5. *If N_∞ is a sample of N with the stationary distribution of the Markov chain $N(T_n)$, then,*

$$(N_0, x_0) = \Phi\Big((N_\infty, C), \xi_{-K}, \ldots, \xi_0\Big),$$

is a sample with the stationary distribution of the Markov chain $(N(T_n), x(T_n))$ *if* K''' *is the vertical backward coupling time of the trajectories starting with the initial states* (N_∞, C) *and* $(N_\infty, 0)$*:*

$$K''' \stackrel{\text{def}}{=} \min\{n : \Phi\Big((N_\infty, C), \xi_{-n}, \ldots, \xi_0\Big) = \Phi\Big((N_\infty, 0), \xi_{-n}, \ldots, \xi_0\Big).$$

Proof. Starting from a stationary state (N_∞, x_∞) at time τ, the state at time 0 must be (N_0, x_0) by monotonicity of the chain with respect to x. The state at time $-K'''$ being stationary, so is the state at time 0, therefore, (N_0, x_0) has the stationary distribution of the chain $(N(T_n), x(T_n))$. □

This construction has several advantages.

First, N_∞ can be constructed without using a simulation since the stationary distribution of the Erhenfest birth and death process $N(T_n)$ is well known (see Equation (9)). Generating a sample according to this distribution is rather simple and can be done in constant time using an aliasing technique [Walker, 1974].

A second advantage is that since both trajectories always have the same value for N, the associated Markov chain becomes monotone and there is no need to use envelopes. This improves the coupling time since dummy events which tend to widen the gap between the upper and lower trajectories never occur here.

We have used this approach and have evaluated its benefit in terms of coupling time as well as total computation time. In the following experiment, we ran both algorithm 2 and this N_∞ enhancement. The experimental setup is the following. We chose the realistic parameters $\alpha = 0.4, \theta = 10^{-4}, \sigma = 10^{-3}, C = 10000, N_{max} = 5000, \lambda = \mu = 10^{-4}$. For each experiment using this setup, the coupling time was 65536 (2^{16}) steps for algorithm 2 and 32768 (2^{15}) for the N_∞ algorithm (note that the coupling time is necessarily a power of two because of the time doubling technique). Regarding the real execution time (measured with the *date* command), the N_∞ algorithm takes less than 45 minutes (for 1000 runs) while the original algorithm 2 ended after more than 2 hours and 40 minutes, over a Pentium 4 PC. The fact that the N_∞ algorithm is more than twice faster than the original algorithm while its coupling time is just halfed, comes from the fact that there is no need to generate envelopes in that case, which saves times in the inner loop of the algorithm.

Finally, this approach provides a better understanding on the simulation duration. When starting from states (N_∞, C) and $(N_\infty, 0)$, the rates of convergence over both trajectories (see Lemma 9 in Appendix A) is uniformly bounded by $\gamma \stackrel{\text{def}}{=} \max_{N=1}^{N_{max}} \gamma_N$.

From Lemma 9, the distances $|x_1(t) - \ell_N|$ and $|x_2(t) - \ell_N|$ are both bounded by $e^{-\gamma t}$ over all the inter-jump intervals. Therefore, $|x_1(t) - x_2(t)| \leq 2Ce^{-\gamma t}$. Note that this bound does not depend on N anymore. Also note that $|x_1(t) - x_2(t)|$ never increases at jump times: $|x_1(T_n) - x_2(T_n)| \leq |x_1(T_n^-) - x_2(T_n^-)|$.

The duration before coupling T verifies $|x_1(T) - x_2(T)| \leq \varepsilon/3$, so that

$$T \leq -\gamma \log(\varepsilon/6C) = \gamma(\log(6C) - \log(\varepsilon)). \tag{14}$$

This bound is rather tight since the rate of convergence is never much larger than γ, over the whole range of N and x.

As for the vertical coupling time K''' of the simulation, K''' corresponds to a sum of independent exponential variables with rate Λ such that $\sum_{i=1}^{K'''} \tau_i \geq T \geq \sum_{i=1}^{K'''-1} \tau_i$. Therefore, $K''' - 1$ has a Poisson distribution with rate ΛT. Its mean verifies $\mathbb{E}K''' \leq -\Lambda\gamma \log(\varepsilon/6C) + 1$ and its variance $varK''' \leq -\Lambda\gamma \log(\varepsilon/6C) + 1$.

This explains why most simulations have the same number of steps (which is a power of 2, such that $2^{n-1} \leq K''' \leq 2^n$), since T is almost a deterministic value.

Theorem 6. *The time complexity of the* N_∞ *simulation algorithm is*

$$O\left(N_{max}(\log C - \log \varepsilon)\left(1 + \frac{(N \log(C/\varepsilon))^{1/\alpha}}{C\varepsilon^{1/\alpha}}\right)\right),$$

and its space complexity is $O(N_{max}(\log C - \log \varepsilon))$*.*

Proof. The complexity of the algorithm is $O((K''' + c_\Phi)(|MAX| + |MIN|))$ where K''' is the vertical coupling time, c_Φ is the total cost of computing Φ and $(|MAX| + |MIN|)$ is the number of simulations to be run in parallel. Here , $(|MAX| + |MIN|) = 2$ and we have shown above that $K''' = O(N_{max}(log(C) - log(\varepsilon))$. As for c_ϕ, it is proportional to the number of steps for integrating the differential equation over time T. If each step is of size h such that $h\sigma N + \alpha\tau \frac{\sigma^{\alpha+1} g N^{\alpha+1}}{C^\alpha} h^\alpha \leq \varepsilon/3$ (see Appendix A.5), then $c_\Phi = O(T/h) = O(\frac{N^{1+1/\alpha}}{C\varepsilon^\alpha}(\log C - \log \varepsilon)^{1+1/\alpha}$. Therefore, the total complexity is $O\left(N_{max}(\log C - \log \varepsilon)\left(1 + \frac{(N \log(C/\varepsilon))^{1/\alpha}}{C\varepsilon^{1/\alpha}}\right)\right)$.

The space complexity comes from the fact that the random innovations ξ_n and τ_n need to be stored for the total duration K of the simulation. □

It is quite remarquable that the complexity is linear in N_{max} and in $\log C$, when C is of order N^2 and $\alpha > 1/2$, which is typical in Squirrel systems.

7.2 Asymptotic solutions

The solutions of the differential equations converge to a unique asymptotic value, ℓ_N if no jumps ever occur (or equivalently, if N remains constant) (see Lemma 7 in Appendix A for more on this). This is the number of copies present in the system on average when N customers are present and when nobody ever leaves or joins in.

If the rate of convergence γ_N of x towards its asymptotic value is larger than the jump rate Λ, then is most cases, x will actually be very close to ℓ_N before the next change occurs. Therefore, one may disregard the transient behavior of x and let x jump from ℓ_N to $\ell_{N'}$ whenever a jump from N to N' occurs. This actually makes the system discrete since both x and N only take discrete values.

Computing the asymptotic value, ℓ_N is much faster than solving the differential equation numerically, and can be done by solving numerically in x (using Newton's method).

$$\sigma N \left(1 - \left(\frac{x}{C}\right)^\alpha\right) - \theta x = 0. \tag{15}$$

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

$$\Pi'(N = k, x = \ell_k) = \frac{\left(\frac{\lambda}{\mu}\right)^k C_k^{N_{max}}}{\left(1 + \frac{\lambda}{\mu}\right)^{N_{max}}}.$$

Generating samples according to this distribution is rather simple and can be done in constant time using aliasing techniques.

We have compared this approximation with the exact samples (N_1, x_1) computed by the simulation algorithm presented in Section 5 (using the absolute value $|\ell_{N_1} - x_1|$). As seen in Figure 9, the approximation using the asymptote is very good as soon as γ_{N_1}, the rate of convergence of the solution of the differential equation to its asymptote it larger than 5% of the jump rate Λ.

As a rule of thumb, for large squirrel systems (with a large number of customers (N_{max}) and a very large file system (C)), the asymptotic approximation is valid as soon as the inverse of the life time of files , θ is of the same order as $C(\lambda + \mu)/N_{max}$.

8 Conclusion

In this paper we have presented a general simulation framework for stochastic hybrid systems providing guaranteed samples. We demonstrated the applicability of this technique by carrying out an experimental study of a complex peer to peer system modeled by hybrid equations.

Simulations based on backward coupling techniques are particularly well adapted to the simulation of stochastic hybrid systems for several reasons.

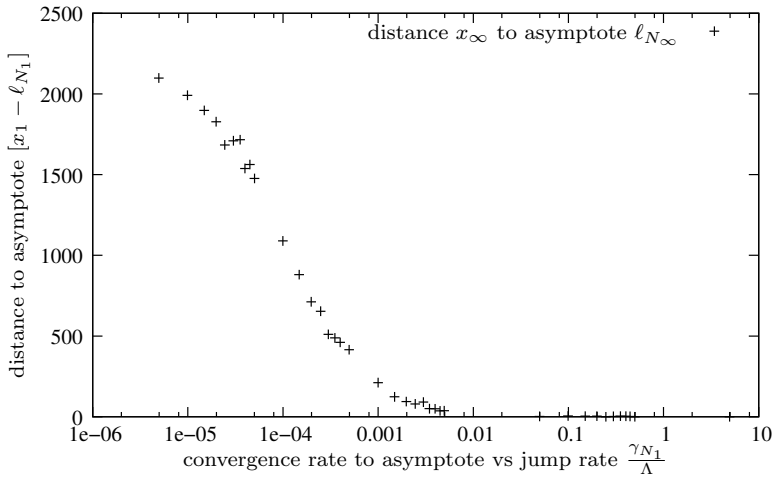

Figure 9: Comparing the samples of perfect simulations with their asymptotic approximation.

- They provide numerical guarantees,
- and they get rid of the difficulty to manipulate continuous variables.
- Finally, hybrid systems often mean large state spaces. These large state spaces often make numerical computations of the behavior of the system impossible. However, the complexity of our backward coupling hardly depends on the size of the state space. It is remarquable that in the Squirrel example, the complexity is in $O(N_{max}(\log(C) - \log(\varepsilon))$ in most realistic cases.

A More on the Squirrel differential equation

The appendix is used to explain the construction of the differential equation

$$\frac{dx}{dt} = f_N(x(t)) = \sigma N \left(1 - \left(\frac{x(t)}{C}\right)^\alpha\right) - \theta x(t), \tag{16}$$

and to assess its properties (uniqueness of the solution, asymptotic behavior, rate of convergence).

A.1 Modeling the miss probability

We now explain how the miss probability of Squirrel can be modeled by

$$1 - \left(\frac{x}{C}\right)^\alpha. \tag{17}$$

This model was first proposed in [Clévenot and Nain, 2004, Clévenot-Perronnin, 2005]. In this section we will not only recall the theoretical grounds for this model, but we will also provide experimental results to validate the model and estimate α.

A uniform-popularity model would straightforwardly give a linear model $P(\text{hit}|N(t), x(t)) = \frac{x(t)}{C}$ (the hit probability being 1 - the miss probability). However, it was shown in [Breslau et al., 1999] that Web objects exhibit a Zipf-like popularity distribution, i.e., the k-th most popular object is requested with probability $\frac{\Omega}{k^\beta}$, where β is the skew parameter of the distribution and $\Omega = 1/\sum_{i=1}^{C} i^{-\beta}$ is a normalization factor. In [Breslau et al., 1999, Dunn et al., 2003] it was shown that β is close to one. Assuming that the $x(t)$ present objects in cache are the $x(t)$ most

popular ones (since they are more likely to be requested), a straightforward approximation for the hit probability is

$$P(\text{hit}|x(t)) = \sum_{i=1}^{\lfloor x(t) \rfloor} \frac{\Omega}{i^\beta} \approx \frac{x(t)^{1-\beta} - 1}{C^{1-\beta} - 1}, \tag{18}$$

by using the approximation $\sum_{i=1}^{\lfloor x \rfloor} i^{-\beta} \approx \int_1^x t^{-\beta} dt = (x^{1-\beta} - 1)/(1-\beta)$ for $x \geq 1$. From (18) we get the rough approximation $P(\text{hit}|x(t)) = (\frac{x(t)}{C})^\alpha$ with $\alpha = 1 - \beta$.

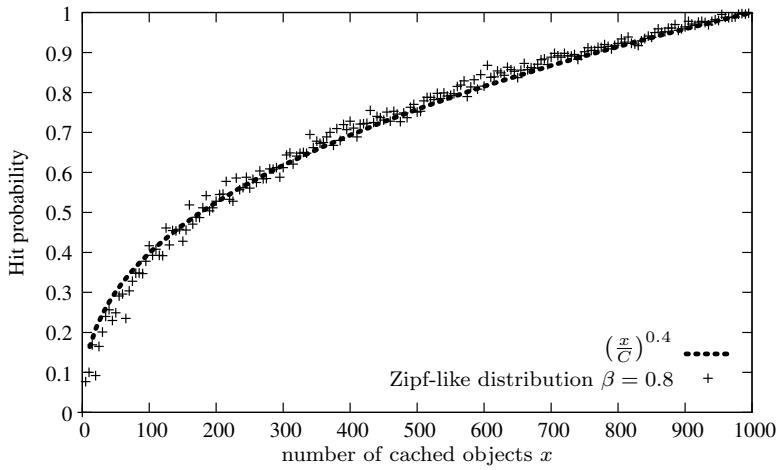

Figure 10: Hit probability as a function of the number of cached objects x for the Zipf-like distribution with parameter $\beta = 0.8$ and with $C = 1000$ possible objects.

To validate this model we have conducted the following experiment. We have initially drawn x samples from a Zipf-like distribution. Then we drew an additional sample z. This sample z results in a hit if it belongs to the set of x initially drawn samples, in a miss otherwise. This latter step is repeated 1000 times to get a 95% confidence interval no larger than 0.03 on the hit probability. Then we compared the experimental hit probability to the model given by (17) and we computed the value of α that best fitted the numerical results. Figure 10 show how perfectly (17) follows the experimental hit probability.

However, the value of α used to get this accuracy slightly differs from $1-\beta$. Actually for most values of β between 0.1 and 0.9 we found experimentally that $\alpha \approx 1.2 - \beta$ for $\beta < 1$. We conjecture that this discrepancy is partly due to the fact that the x cached objects are not necessarily the x most popular ones. It may also be due to the approximations used to get (18).

A.2 Asymptote

Lemma 7. *Any solution $x(t)$ of the differential equation converges to a unique limit ℓ_N, for all initial conditions.*

Proof. Setting $\frac{dx}{dt} = 0$ in (16) we get $f_N(x) = 0$. This equation is verified by potential asymptotes for $x(t)$.

Since $\frac{df_N}{dx} < 0$ for all $x \geq 0$ and for all $N > 0$, and since $f_N(0) = \sigma N > 0$ and $f_N(C) = -\theta C < 0$, equation $f_N(x) = 0$ admits a unique positive solution denoted by ℓ_N:

$$f_N(\ell_N) = 0. \tag{19}$$

Therefore any solution $x_N(t)$ of (16) admits a unique asymptote ℓ_N. □

Computing ℓ_N can be done by using a classical Newton method [Press et al., 1992], which converges very fast here.

A.3 Uniqueness of the solution

The differential equation (16) is not Lipschitz in 0 as soon as $\alpha < 1$. Therefore, the classical techniques used to prove uniqueness do not apply here. However, uniqueness is required for our simulation method to work.

Lemma 8. *The equation (16) admits a unique solution for any initial condition* $x^0 \stackrel{\text{def}}{=} x(0) \geq 0$.

Proof. Let us consider first the case $x^0 < \ell_N$. In that case, if $x_1(t)$ is one solution, $\frac{dx_1}{dt} = f_N(x_1(t)) > 0$ for all $t \geq 0$, so that $x_1(t)$ is strictly increasing from x^0 to ℓ_N. Therefore, the inverse function $x_1^{-1}(u) : [x^0, \ell_N) \to \mathbb{R}$ is well defined. If we consider another solution $x_2(t)$ then the derivative of the inverse functions $x_1^{-1}(u)$ and $x_2^{-1}(u)$ verify $\frac{dx_1^{-1}}{du}(u) = \frac{1}{f_N(u)} = \frac{dx_1^{-1}}{du}(u)$ for all $u \in [x^0, \ell_N)$. So that $x_1^{-1}(u) - x_2^{-1}(u)$ is constant and does not depend on u. When $u = x^0$, then $x_1^{-1}(x^0) = 0$ and $x_2^{-1}(x^0) = 0$. This implies that $x_1^{-1}(u) = x_2^{-1}(u)$ for all $u \in [x^0, \ell_N)$, so that $x_1(t) = x_2(t)$ for all $t \geq 0$.
If $x^0 = \ell_N$, then the derivative of any solution remains equal to zero, so that there is a unique solution $x(t) \equiv \ell_N$.
Finally, the case $x^0 > \ell_N$ is similar to the case $x^0 < \ell_N$, with decreasing solutions instead of increasing ones. □

A.4 Rate of convergence

Let us define the *rate of convergence* $\gamma_N(t)$ at time t of the ODE solution $x_N(t)$ as follows: if $x_N(t) < \ell_N$,

$$\gamma_N(t) \stackrel{\text{def}}{=} \lim_{d \to 0} \frac{(x_N(t+d) - \ell_N) - (x_N(t) - \ell_N)}{d(\ell_N - x_N(t))}, \tag{20}$$

and $\gamma_N \stackrel{\text{def}}{=} \lim_{d \to 0} \frac{(x_N(t+dt) - \ell_N) - (x_N(t) - \ell_N)}{dt(x_N(t) - \ell_N)}$ otherwise.

We now simplify (20) when $x_N(t) < \ell_N$. The case $x_N(t) > \ell_N$ is exactly similar and will be omitted.

Taking the limit as $d \to 0$ we get

$$\begin{aligned}
\gamma_N(t) &= \lim_{d \to 0} \frac{x_N(t+d) - x_N(t)}{d(\ell_N - x_N(t))}, & (21)\\
&= \frac{\frac{dx_N}{dt}(t)}{\ell_N - x_N(t)}, & (22)\\
&= \frac{\frac{dx_N}{dt}(t) - f_N(\ell_N)}{\ell_N - x_N(t)}, & (23)
\end{aligned}$$

where (23) is obtained using (19). Then when $x_N(t) \to \ell_N$ we get the asymptotic rate of convergence,

$$\begin{aligned}
\gamma_N &= \lim_{x \to \ell_N} \frac{f(x) - f_N(\ell_N)}{\ell_N - x_N(t)}, & (24)\\
&= -\frac{df}{dx}(\ell_N) = \frac{\sigma\alpha N + \theta(1-\alpha)\ell_N}{\ell_N}. & (25)
\end{aligned}$$

Lemma 9. *If* $x_N(0) < \ell_N$, *then for all* $t > 0$, $\gamma_N(t) \geq \gamma_N$ *and for all* $t > 0$, $x_N(t) \geq \ell_N(1 - \exp(-\gamma_N t))$.

Proof. Note that if $x_N(0) < \ell_N$ then (in the absence of jumps) we have $x_N(t) < \ell_N$ for all $t > 0$. From (22) and (25) we have for all N:

$$\begin{aligned}\gamma_N(t) - \gamma_N &= \frac{\frac{dx_N}{dt}(t)}{\ell_N - x_N(t)} + \frac{df}{dx}(\ell_N),\\ &= \frac{f_N(x_N(t)) + (\ell_N - x_N(t))\frac{df}{dx}(\ell_N)}{\ell_N - x_N(t)},\\ &= \frac{\sigma N\left(1 - \left(\frac{x_N(t)}{C}\right)^\alpha\right) - \theta x_N(t) - (\ell_N - x_N(t))(-\frac{\sigma N\alpha(\frac{\ell_N}{C})^\alpha}{\ell_N} - \theta)}{\ell_N - x_N(t)}.\end{aligned}$$

The numerator $g(x(t)) \stackrel{\text{def}}{=} \sigma N\left(1 - \left(\frac{x_N(t)}{C}\right)^\alpha\right) - \theta x_N(t) - (\ell_N - x_N(t))(-\frac{\sigma N\alpha(\frac{\ell_N}{C})^\alpha}{\ell_N} - \theta)$ is positive since:

$$\frac{dg}{dx}(x(t)) = \frac{\sigma N\alpha\ell_N^\alpha x_N(t)^\alpha(-\ell_N^{1-\alpha} + x_N(t)^{1-\alpha})}{C^\alpha x_N(t)\ell_N} < 0,$$

and $\lim_{t\to\infty} g(x(t)) = 0$. This proves the first inequality of the lemma.

Using (22) this first inequality gives

$$\begin{aligned}\frac{\frac{dx_N}{dt}(t)}{\ell_N - x_N(t)} &\geq \gamma_N\\ \frac{dx_N}{dt}(t) &\geq \gamma_N(\ell_N - x_N(t)).\end{aligned}$$

Let $y(t)$ be the solution of the first-order linear ODE $\frac{dy}{dt}(t) = \gamma_N(\ell_N - y(t))$ with $y(0) = 0$. This solution writes $y(t) = \ell_N(1 - \exp(-\gamma_N t))$. We now prove the second inequality, i.e., that for all $t > 0$, $x_N(t) \geq y(t)$. Let us consider the derivative of the inverse functions (which are always well defined here):

$$\frac{dy^{-1}}{du}(u) = \frac{1}{\gamma_N(\ell_N - u)} \geq \frac{dx_N^{-1}}{du}(u).$$

As a result we have $x_N^{-1}(u) \leq y^{-1}(u)$ for all $u \geq 0$ since $x_N^{-1}(0) = y^{-1}(0) = 0$. This implies that $x_N(t) \geq y(t)$ for all $t \geq 0$. □

A.5 Numerical integration

First, note that in the case $\alpha = 1$ the ODE (16) admits a closed-form solution on $[T_n, T_{n+1})$:

$$x(t) = \frac{\sigma N(T_n)}{\frac{\sigma N(T_n)}{C} + \theta} + \left(x(T_n) - \frac{\sigma N(T_n)}{\frac{\sigma N(T_n)}{C} + \theta}\right) e^{-(t-T_n)(\theta + \frac{\sigma N(T_n)}{C})}. \tag{26}$$

However, when $\alpha \neq 1$, the equation does not have a closed form solution and must be solved numerically. In this paper, we only consider a first order Runge-Kutta method. More sophisticated integration is possible, yielding a better precision with fewer discretization steps.

The fact that the equation may not be Lipschitz in 0, makes the computation of the error a little tricky. Here, we simply compute a bound on the error made by using the classical Euler integration method: $x_n = x_{n-1} + hf_N(_{n-1})$. One has to consider the first step of integration apart. The error on the first step is bounded by $hf_N(0) \leq h\sigma N$. As for all subsequent steps, n, the error is bounded by $|h^2 x''(a)|$ for some $nh \leq a \leq (n+1)h$.

$$\begin{aligned}h^2|x''(a)| &\leq |h^2\left[\sigma N\left(1 - \left(\frac{x_n}{C}\right)^\alpha\right) - \theta x_n\right]\left[-\sigma N\alpha\frac{x_n^{\alpha-1}}{C^\alpha} - \theta\right]|,\\ &\leq h^2\frac{\sigma^{\alpha+1}N^{\alpha+1}}{C^\alpha}\alpha h^{\alpha-1},\\ &\leq \frac{\sigma^{\alpha+1}N^{\alpha+1}}{C^\alpha}\alpha h^{\alpha+1}.\end{aligned}$$

Therefore, the total error by integrating over a duration T is bounded by

$$e = h\sigma N + \alpha T \frac{\sigma^{\alpha+1} N^{\alpha+1}}{C^\alpha} h^\alpha.$$

B More on the Squirrel Markov chain

The squirrel Markov chain $S_n = (N(T_n), x(T_n))$ is proved uniformly ergodic in Section 5 and $(0,0)$ is an atom. However, its final class (the set of states reachable with positive probability) is not clearly stated. We define $\mathcal{C} = \cup\{A : \exists m | P^m(s, A) > 0\}$ for all $s \in \mathcal{D}$

Lemma 10. *The final reachable class* $\mathcal{C} = \{(N,x) | N \in \{1, \ldots, N_{max}\}\ ,\ 0 < x < \ell_N\} \cup \{(0,0)\}$.

Proof. First note that state $(0,0)$ is reachable from any state $(N,x) \in \{0, \ldots N_{max}\} \times [0, C]$ (see Section 5). Now, from state $(0,0)$, one may reach state N, x for any $0 \leq N \leq N_{max}$ and any x such that $0 < x < \ell_N$ by letting N arrivals come almost simultaneously and let the time run, so that x may grow from 0 to ℓ_N.

This means that $\mathcal{D} \stackrel{\text{def}}{=} \{(N,x) | N \in \{1, \ldots, N_{max}\}, 0 \leq x < \ell_N\} \subset \mathcal{C}$.

Next, we show that during the evolution of the Markov chain starting in $(0,0)$, no state with $x > \ell_N$ may ever be reached. Starting in state (N,x) with $x < \ell_N$, one may only reach (N', x') with $N' = N$, or $N' = N-1$ or $N' = N+1$.

1. If $N' = N$ then $x' < \ell_N$ and the state stays in $\mathcal{D}$.
2. If $N' = N+1$ then $x \leq x' < \ell_{N+1}$ and the state stays in $\mathcal{D}$.
3. If $N' = N-1$ then x' may reach any value between $x(N-1)/N$ and ℓ_{N-1}. The proof is over if one can show that $(N-1)x/N < \ell_{N-1}$.

It remains to show that $(N-1)x/N < \ell_{N-1}$ for any $0 < x < \ell_N$. It is enough to show that $\ell_N N'/N < \ell_{N'}$, with $N' = N-1$.

Let us consider $f_{N'}(\ell_N N'/N)$:

$$\begin{aligned}
f_{N'}\left(\ell_N \frac{N'}{N}\right) &= \sigma N' \left(1 - \left(\frac{N'\ell_N}{NC}\right)^\alpha\right) - \theta \frac{N'}{N}\ell_N, \\
&= \sigma N' \left(\left(\frac{N'}{N}\right)^\alpha \left(1 - \left(\frac{\ell_N}{C}\right)^\alpha\right) + 1 - \left(\frac{N'}{N}\right)^\alpha\right) - \theta \frac{N'}{N}\ell_N, \\
&= \left(1 - \left(\frac{N'}{N}\right)^\alpha\right)\left(\sigma - \frac{\theta \ell_N}{N}\right), \\
&= \left(1 - \left(\frac{N'}{N}\right)^\alpha\right)\sigma\left(\frac{\ell_N}{C}\right)^\alpha, \\
&\geq 0.
\end{aligned}$$

$f_{N'}(\ell_N N'/N) \geq 0$ means that $\ell_N N'/N < \ell_{N'}$ because $f_{N'}$ is decreasing and $f_{N'}(\ell'_N) = 0$ (see Appendix A.2). □

References

[Alur et al., 2003] Alur, R., Dang, T., Esposito, J., Hur, Y., Ivancic, F., Kumar, V., Lee, I., Mishra, P., Pappas, G. J., and Sokolsky, O. (2003). Hierarchical modeling and analysis of embedded systems. *Proceedings of the IEEE*, 91(1):11–28.

[Asarin et al., 1995] Asarin, E., Maler, O., and Pnueli, A. (1995). Reachability analysis of dynamical systems having piecewise-constant derivatives. *Theoretical Computer Science*, 138:35–65.

[Breslau et al., 1999] Breslau, L., Cao, P., Fan, L., Phillips, G., and Shenker, S. (1999). Web caching and Zipf-like distributions: Evidence and implications. In *Proceedings of IEEE INFOCOM '99*, pages 126–134, New York.

[Clévenot and Nain, 2004] Clévenot, F. and Nain, P. (2004). A simple model for the analysis of the Squirrel peer-to-peer caching system. In *Proceedings of IEEE INFOCOM 2004*, Hong Kong.

[Clévenot-Perronnin, 2005] Clévenot-Perronnin, F. (2005). *Fluid Models for Content Distribution Systems.* PhD thesis, University of Nice-Sophia Antipolis. http://www-sop.inria.fr/dias/Theses/phd-9.php.

[Dai, 1995] Dai, J. (1995). On positive harris recurrence of multiclass queueing networks: a unified approach via fluid limit models. *Annals of Applied Probability*, 5:49–77.

[David and Alla, 2004] David, R. and Alla, H. (2004). *Discrete, Continuous, and Hybrid Petri Nets.* Springer-Verlag.

[Dunn et al., 2003] Dunn, K. G. R., Saroiu, S., Gribble, S., Levy, H., and Zahorjan, J. (2003). Measurement, modeling and analysis of a peer-to-peer file-sharing workload. In *Proceedings of the 19th ACM Symposium of Operating Systems Principles (SOSP)*, NY.

[Foss and Tweedy, 1998] Foss, S. and Tweedy, R. (1998). Perfect simulation and backward coupling. *Stochastic Models*, 14:187–204.

[Girard, 2005] Girard, A. (2005). Reachability of uncertain linear systems using zonotopes. In *Hybrid Systems : Computation and Control*, volume 3414 of *LNCS*, pages 291–305.

[Iyer et al., 2002] Iyer, S., Rowstron, A., and Druschel, P. (2002). Squirrel: A decentralized, peer-to-peer Web cache. In *Proceedings of of ACM Symposium on Principles of Distributed Computing (PODC 2002)*, pages 213–222, Monterey, California.

[Press et al., 1992] Press, W. H., Teukolsky, S. A., Vetterling, W. T., and Flannery, B. P. (1992). *Numerical Recipes in C.* Cambridge University Press.

[Propp and Wilson, 1996] Propp, D. and Wilson, J. (1996). Exact sampling with coupled Markov chains and application to statistical mechanics. *Random Structures and Algorithms*, 9(1):223–252.

[Rowstron and Druschel, 2001] Rowstron, A. and Druschel, P. (2001). Pastry: Scalable distributed object location and routing for large-scale peer-to-peer systems. In *Proceedings of Int. Conf. on Distributed Systems Platforms (Middleware)*, Heideberger, Germany.

[Tomlin et al., 2003] Tomlin, C., Mitchell, I., Bayen, A., and Oishi, M. (2003). Computational techniques for the verification and control of hybrid systems,. *Proceedings of the IEEE*, 91(7):986–1001.

[Walker, 1974] Walker, A. (1974). An efficient method for generating random variables with general distributions. *ACM Trans. Math. Software*, pages 253–256.

Unité de recherche INRIA Rhône-Alpes
655, avenue de l'Europe - 38334 Montbonnot Saint-Ismier (France)

Unité de recherche INRIA Futurs : Parc Club Orsay Université - ZAC des Vignes
4, rue Jacques Monod - 91893 ORSAY Cedex (France)
Unité de recherche INRIA Lorraine : LORIA, Technopôle de Nancy-Brabois - Campus scientifique
615, rue du Jardin Botanique - BP 101 - 54602 Villers-lès-Nancy Cedex (France)
Unité de recherche INRIA Rennes : IRISA, Campus universitaire de Beaulieu - 35042 Rennes Cedex (France)
Unité de recherche INRIA Rocquencourt : Domaine de Voluceau - Rocquencourt - BP 105 - 78153 Le Chesnay Cedex (France)
Unité de recherche INRIA Sophia Antipolis : 2004, route des Lucioles - BP 93 - 06902 Sophia Antipolis Cedex (France)

Éditeur
INRIA - Domaine de Voluceau - Rocquencourt, BP 105 - 78153 Le Chesnay Cedex (France)
http://www.inria.fr
ISSN 0249-6399