Stochastic Fluid Model for P2P Caching Evaluation

Florence Clévenot-Perronnin, Philippe Nain

INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Stochastic Fluid Model for P2P Caching Evaluation

Florence Clévenot-Perronnin — Philippe Nain

N° 5533

March 2005

Thème COM

ISSN 0249-6399 ISRN INRIA/RR--5533--FR+ENG

INRIA
SOPHIA ANTIPOLIS

Stochastic Fluid Model for P2P Caching Evaluation

Florence Clévenot-Perronnin*, Philippe Nain†

Thème COM — Systèmes communicants
Projet Maestro

Rapport de recherche n° 5533 — March 2005 — 25 pages

Abstract: In this paper we propose a stochastic fluid model to analyze the performance of Squirrel: a P2P cooperative Web cache. This work provides a scalable and insightful extension of our previous analysis of Squirrel.

This new model provides a closed-form expression for the hit probability when documents are equally popular. Realistic object popularity is also addressed through a clustering approximation. The accuracy of this model is validated by a comparison with discrete-event simulations.

Our model allows us to study the impact of various parameters on the performance of the Squirrel system. In particular, we emphasize the importance of taking object popularity into account. We also investigate the utility of clients announcing their departure on the resulting hit probability.

Key-words: Web caching, peer-to-peer, Zipf-like popularity, performance analysis, fluid models.

* INRIA Sophia Antipolis. E-mail: Florence.Clevenot@sophia.inria.fr
† INRIA Sophia Antipolis. E-mail: Philippe.Nain@sophia.inria.fr

Unité de recherche INRIA Sophia Antipolis
2004, route des Lucioles, BP 93, 06902 Sophia Antipolis Cedex (France)
Téléphone : +33 4 92 38 77 77 — Télécopie : +33 4 92 38 77 65

Un Modèle Stochastique Fluide pour l'Analyse d'un Système P2P de Cache Web

Résumé : Nous proposons un modèle fluide stochastique pour analyser les performance d'un système de cache Web coopératif pair-à-pair appelé Squirrel. Ce travail est une extension d'une précédente analyse de Squirrel.

Le nouveau modèle proposé permet notamment le passage à l'échelle pour les systèmes de grande taille, grâce à une formule close. Nous montrons également comment incorporer la popularité des documents dans l'analyse grâce à une approximation multiclasses. Nous validons cette approche par comparaison avec une simulation à événements discrets.

Ce modèle permet d'étudier l'influence de différents paramètres sur la probabilité de hit du système Squirrel. En particulier, nous nous intéressons à l'importance de la distribution de popularité des documents. Nous évaluons également l'utilité d'un mécanisme permettant aux usagers d'annoncer leur déconnexion.

Mots-clés : Systèmes peer-to-peer, caches Web, évaluation de performances, modèles fluides stochastiques, loi de Zipf.

Contents

1 Introduction

P2P systems capitalize on individual user resources to build up self-organizing, scalable file-sharing systems. The Squirrel system [10] was recently introduced by Iyer, Rowstron and Druschel as a cooperative P2P web cache. Squirrel leverages the individual storage capabilities of the members of an institutional network to build a shared, decentralized web cache. In Squirrel each document (or object) is mapped to a unique node in the network, called the *home node*, identified by a node-Id. The home node of an object is responsible for delivering cached copies of this object to any user in the Squirrel network. Two schemes have been proposed and analyzed in [10], the *home store* scheme and the *directory* scheme. In the home store scheme, the home node stores a copy of each document it is responsible for. In the directory scheme the home node only stores a directory of pointers to the last nodes which have requested the document and thus may keep a local copy. More details on the behavior of these schemes, and more generally on the behavior of Squirrel, can be found in [10]. It has been reported in [10] that the home store scheme is overall more attractive than the directory scheme. Therefore, we will only focus on the home node scheme from now on.

Because of its peer-to-peer structure, Squirrel is expected to be scalable and cost-effective. While this promising system is still in the test phase it is important for dimensioning and optimization purposes to study its expected performance (hit probability, document transfer time, etc.) and its scalability. In [4] we developed a quantitative analysis of Squirrel in which we derived the hit probability (i.e. the probability that a requested document is found in the Squirrel network). The analysis was based on the observation that Squirrel basically evolves on two different time-scales: a slow time-scale corresponding to the node dynamics, namely the process at which nodes join and leave Squirrel, and a fast time-scale corresponding to the frequency at which documents are requested. In [4] the node dynamics were modeled by a Engset process, a N-state Markov process, with N the number of nodes in the Squirrel network. As to the request process, we assumed that each active Squirrel node generated requests at a constant rate. We then showed that the total number of available documents in the Squirrel network was accurately modeled by a piecewise deterministic fluid process.

The aim of the present work is to extend the analysis in [4] in two main directions. First, we replace the Engset model by an infinite-state Markov process (the $M/M/\infty$ queuing model – see Section 2.3), which yields a dramatic decrease in the complexity of computing the hit probability. Indeed, the Engset model solution involves binomial coefficients and exponentials in the size of the network. This restricted the performance analysis to the order of 10,000 nodes. Our new $M/M/\infty$ model allows us to easily handle real size networks (e.g., 100,000 nodes for a large corporate network or even larger). Second, we relax the assumption made in [4] that all documents are equally popular, and provide an efficient method for computing the hit probability in realistic situations (i.e. with Zipf-like document popularity distribution). Numerical comparisons with discrete-event simulations validate these extensions.

The rest of the paper is organized as follows. In Section 2 we give an overview of Squirrel, discuss related studies, recall the fluid model defined in [4] for the number of available documents in the Squirrel networks, and introduce the model for the node dynamics. We then show in Section 3 how to compute the hit probability at a constant cost in the number of nodes under the assumption that all objects are equally popular. The latter assumption is relaxed in Section 4, where we

incorporate a Zipf-like object popularity distribution in our model, and show how to compute the hit probability in this more general setting. These models are used in Section 5 to make a number of qualitative observations on Squirrel, related to the impact of unequal document popularity and to the impact of announced/unannounced departures (see 5.3) on Squirrel performance. Section 6 is devoted to the experimental validation of our approach, and concluding remarks are given in Section 7.

2 Modeling a P2P Cooperative Cache

2.1 Overview of Squirrel

Squirrel [10] is a decentralized, peer-to-peer Web cache that uses Pastry [15] as a location and routing protocol. When a client requests an object it first sends a request to the Squirrel proxy running on the client's machine. If the object is uncacheable then the proxy forwards the request directly to the origin Web server. Otherwise it checks the local cache, like every Web browser would do, in order to exploit locality and reuse. If a fresh copy of the object is not found in this cache, then Squirrel tries to locate one on some other node. To do so, it uses the distributed hash-table and the routing functionalities provided by Pastry. First, the URL of the object is hashed to give a 128-bit object identity (a number called *object-Id*) from a circular list; then the routing procedure of Pastry forwards the request to the node with the identity (called *node-Id*; this number is assigned randomly by Pastry to a participating node) the closest to *object-Id*. This node then becomes the *home node* for this object. Squirrel then proposes two schemes from this point on: *home-store* and *directory* schemes.

In the home-store scheme, objects are stored both at client caches and at its home node. The client cache may either have no copy of the requested object or a stale copy. In the former case the client issues a GET request to its home-node, and it issues a *conditional* GET (cGET) request in the latter case. If the home-node has a fresh copy of an object then it forwards it to the client or it sends the client a not-modified message depending on which action is appropriate. If the home-node has no copy of the object or has a stale copy in its cache, then it issues a GET or a cGET request, respectively, to the origin server. The origin server then either forwards a cacheable copy of the object or sends a not-modified message to the home-node. Then, the home-node takes the appropriate action with respect to the client (i.e. send a not-modified message or a copy of the object).

In the directory scheme the home-node for an object maintains a small directory of pointers to nodes that have recently accessed the object. A request for this object is sent randomly to one of these nodes. We will not go deeper into the description of this scheme since from now on we will only focus on the home-store scheme. We do so mainly because the latter scheme has been shown to be overall more attractive than the directory scheme [10]. In addition, the home-store scheme is more amenable to a fluid analysis than the directory scheme.

In a Squirrel network (a corporate network, a university network, etc.), like in any peer-to-peer system, clients arrive and depart the system at random times. There are two kinds of failures (or departures): abrupt and announced failures. Each failure has a different impact on the performance

of Squirrel. An abrupt failure will result in a loss of objects. To see this, assume that node i is the home-node for object O. If node i fails, then a new home-node for object O has to be found by Pastry, as explained above, the next time object O is requested. Assume that the copy of object O was fresh when node i failed and consider the first GET request issued for O after the failure of node i. The GET request is therefore forwarded to the new home-node for object O (say node j); this request will result in a miss if j has no copy of O or if its copy is stale. In this case, the failure of node i will yield a degradation in the performance since node j will have to contact the origin server to get a new copy of object O or a not-modified message, as appropriate. If a node is able to announce its departure and to transfer its content to its immediate neighbors in the node-Id space before leaving Squirrel (announced failure), then no content is lost when the node leaves.

When a node joins Squirrel then it automatically becomes the home node for some objects but does not store those objects yet (see details in [10]). In case a request for one of those objects is issued, then its two neighbors in the node-Id space transfer a copy of the object, if any. Therefore, we can consider that there is no performance degradation in Squirrel due to a node arrival, since the transfer time between two nodes is supposed to be at least one order of magnitude smaller than the transfer time between any given node and the origin server.

From now on the terms "node" and "client", "active" and "up", and "inactive" and "down" will be used interchangeably.

2.2 Related Work

Performance evaluation of P2P systems has recently attracted a lot of attention. Because of the complexity of these systems (in terms of number of documents, number of users, etc.), many traditional models such as queuing models, Markovian models and branching processes require numerical methods [7, 17] or model simulations [18].

A fluid approach allows one to simplify these systems and may offer meaningful analysis at a low computational cost. We already used this approach in [5] to study cache clusters then in [4] for a first study of Squirrel. Here we propose an enhanced version of this model and relax some major assumptions. This fluid approach was introduced by [1] and successfully used in packet network modeling. It has also been used in [13] for an analysis of emerging BitTorrent-like file sharing protocols.

2.3 The Fluid Model

The basic idea is to model HTTP requests as a fluid flow modulated by the random arrivals and departures of the Squirrel nodes. In [4] we modeled the node dynamics by a finite-state birth and death process, with birth (resp. death) rate $\lambda(N-i)$ (resp. μi) when there are $i = 0, 1, \ldots, N$ nodes up, where parameters N (the number of nodes), λ and μ are given. In the literature this Markov process is referred to as the Engset model. This model has two main problems. First, it requires the existence of a bound on the number of nodes which can simultaneously be active (the parameter N). In general there does not exist such a bound and, if it did, then it would be very difficult to find. Second, the calculation of the hit rate induced by the Engset model poses serious computational complexity issues as N becomes large. As an illustration, it took more than one

day to compute the hit rate (given in [4, Prop. 4.1], using a realistic value of $\rho = 100$) on a Intel 4 2000GHz/768Mo workstation for 10,000 nodes, a small population for a corporate network.

To overcome the shortcomings of using the Engset model (i.e. need to have a bound on the number of users and scalability issue), in this paper we model the node dynamics by a M/M/∞ queuing system [12, p. 101].

Remark 2.1 *The M/M/∞ queuing system can be seen as a limiting case of the Engset model, in the sense that their steady-state distributions are equivalent when the mean number of nodes goes to infinity (cf. Appendix C).*

In the M/M/∞ setting, nodes become active according to a Poisson process with intensity λ (referred to as the arrival process) and each node remains active for an exponentially distributed amount of time, with mean $1/\mu$. It is a natural model since it assumes nodes join the system at arbitrary times, independently of each other. At the end of its activity period a node disappears, an event which corresponds to a departure in the M/M/∞ queue. Node activity periods are assumed to be mutually independent, and further independent of the arrival process.

The number of active nodes at time t is denoted by $N(t)$. Let $0 \leq T_1 < T_2 < \cdots$ be the successive jump times of the process $\{N(t), t \geq 0\}$, where a jump corresponds either to a node becoming active (arrival) or to a node becoming inactive (departure). We assume that the sample paths of the process $\{N(t), t \geq 0\}$ are right-continuous, so that $N_n := N(T_n)$ gives the number of active nodes just after the n-th jump (i.e. at time T_n+). By convention we set $T_0 = 0$ and we denote by N_0 the number of active nodes at time $t = 0$.

Let N^∞ denote the number of nodes which are active in steady-state. It is known that N^∞ has a Poisson distribution with parameter $\rho := \lambda/\mu$ [12, p. 101], namely

$$\mathbb{P}[N^\infty = i] = \frac{\rho^i}{i!} e^{-\rho}, \quad i \geq 0. \tag{1}$$

In particular, the expected number of active nodes in steady-state is given by

$$\mathbb{E}[N^\infty] = \rho. \tag{2}$$

Let $\pi_i := \lim_{n\uparrow\infty} \mathbb{P}[N_t = i | t = T_n]$ be the steady-state probability that there are i nodes active just after a jump. We show in appendix B that

$$\pi_0 = \frac{e^{-\rho}}{2} \tag{3}$$

$$\pi_i = \frac{i+\rho}{i!}\, \rho^{i-1} \frac{e^{-\rho}}{2}, \quad i \geq 1. \tag{4}$$

Observe that the probability distributions in (1) and in (3)-(4) are different.

We assume that each node can store an unlimited number of objects. (Currently, disk storage capacity is abundant for most caching systems, and capacity misses are very rare compared to misses due to stale objects.)

The fluid object model is the same as in [4]. It is based on the observation that the process of nodes joining and leaving Squirrel evolves on a much larger time-scale than the rate at which

documents are requested. Like in [4] we assume that each active node produces a continuous and deterministic stream of requests with rate σ, so that σs requests are generated by each active node throughout the time-interval $[t, t+s)$. Each miss (i.e., unsatisfied request) brings a new document into the Squirrel cache. This follows from the fact that a missing document has to be retrieved from the origin server and brought back into the Squirrel cache, thereby increasing the number of stored documents. It may happen that two concurrent requests for the same document will generate two misses but only one cached copy. This event is assumed rare enough to be neglected. (We validate this claim in Section 6.) The total number of documents in Squirrel at any time is represented by a fluid, with $X(t)$ the amount of fluid at time $t \geq 0$. We assume the sample paths of the process $\{X(t),\, t \geq 0\}$ are right-continuous. In particular, $X(T_n)$ is the amount of fluid just after the n-th jump.

During the time-interval (T_n, T_{n+1}) $(n \geq 0)$, each active node generates requests at the constant rate σ, so that σN_n is the total request rate in (T_n, T_{n+1}). Thus $X(t)$ increases at the constant rate σN_n in (T_n, T_{n+1}), multiplied by the probability that a requested document is not found in the Squirrel cache. On the other hand, we assume that each stored document has a Time-To-Leave (TTL) equal to $1/\theta$, so that $\theta X(t)$ is the total expiration rate at time t.

Hence, in the time-interval (T_n, T_{n+1}), $X(t)$ satisfies the following first-order differential equation

$$\frac{d}{dt}X(t) = \sigma N_n(1 - \mathbb{P}[\text{hit}|N_n, X(t)]) - \theta X(t) \tag{5}$$

if $N_n > 0$, where $\mathbb{P}[\text{hit}|N_n, X(t)]$ is the hit probability (and therefore $1 - \mathbb{P}[\text{hit}|N_n, X(t)]$ is the miss probability) at time t given that there are N_n nodes active at time T_n+, and that the amount of fluid (or cached documents) at time t is $X(t)$. Different expressions for $\mathbb{P}[\text{hit}|N_n, X(t)]$ will be given in Sections 3 and 4 depending on whether or not documents are equally popular.

When $N_n = 0$ (all nodes are inactive in (T_n, T_{n+1})), then the amount of fluid remains constant and equal to zero in this time-interval, namely $X(t) = 0$ for $T_n < t < T_{n+1}$ when $N_n = 0$. In particular, the hit probability is equal to zero during such a time-interval.

Let us now examine the behavior of $X(t)$ at jump times T_n, $n \geq 1$. To this end, we introduce two mappings, $\Delta_u : \{0, 1, \ldots\} \to [0, 1]$ and $\Delta_d : \{1, 2, \ldots\} \to [0, 1]$.

If the jump occurring at time T_n corresponds to a new node joining Squirrel, then the amount of fluid in the Squirrel cache at time T_n drops by a factor $\Delta_u(N(T_n-))$, that is $X(T_n) = \Delta_u(N(T_n-))X(T_n-)$. Similarly, if the jump occurring at time T_n corresponds to a departure, then the amount of fluid drops by a factor $\Delta_d(N(T_n-))$, so that $X(T_n) = \Delta_d(N(T_n-))X(T_n-)$. (The subscripts u and d in Δ_u and Δ_d refer to "up" and "down", respectively.)

As discussed in Section 2.1 there is no loss of content when a new node joins Squirrel, since its neighbors (in the node-Id space) can transfer the documents for which the new node is now the home node. Moreover, we assume that a node joining Squirrel does not bring any document with it. Though this assumption can be regarded as restrictive, it has the following motivation: the minimum time during which a node is inactive is orders of magnitude higher than the request inter-arrival time. Therefore, most of the documents stored in an inactive node i, especially the most popular ones, are very likely to be requested and added to their new home nodes, while node i is down. As a result, when node i joins the system with its own set of documents, it will not *add*

any content to the Squirrel system, but merely become the new home node for these objects. The combination of these two properties gives that $\Delta_u(i) = 1$ for $i \geq 1$.

On the other hand, there is no loss of content if a departure is announced, so that $\Delta_d(i) = 1$ $(i \geq 2)$ when such an event occurs. In the case of an abrupt departure the content of the departing node is totally lost. Since it has been observed that Squirrel reaches a good load balancing among the active nodes [10], we can assume that a fraction $1/i$ of the content is lost if a node departs without warning when there are i nodes active. Hence, $\Delta_d(i) = (i-1)/i$ $(i \geq 1)$ in this case.

In the following we will analyze both the situations where $\Delta_d(i) = 1$ and $\Delta_d(i) = (i-1)/i$, with $\Delta_u(i) = 1$ in both cases.

We denote by $c < \infty$ the total number of objects that Squirrel clients may request.

3 Hit Probability: Uniform Popularity Case

We consider the stationary hit probability as a performance measure of the system. A precise definition of this metric will be given shortly (see (9)). We first assume that all objects are equally popular, which implies that the probability that a given object o is requested is $1/c$. This assumption is relaxed in Section 4, where a more realistic Zipf-like popularity distribution is considered.

Under the uniform document popularity assumption, the (conditional) hit probability at time t, $\mathbb{P}[\text{hit}|N_n, X(t)]$, is a simple linear function of $X(t)$, given by

$$\mathbb{P}[\text{hit}|N_n, X(t)]) = \frac{X(t)}{c}. \tag{6}$$

Plugging this value of $\mathbb{P}[\text{hit}|N_n, X(t)]$ into (5) yields

$$\frac{d}{dt}X(t) = \sigma N_n - Z_n X(t), \quad t \in (T_n, T_{n+1}). \tag{7}$$

where $Z_n := \frac{\sigma N_n}{c} + \theta$.

Given N_n, T_n and T_{n+1}, and given the amount of fluid $X_n := X(T_n)$ in the Squirrel cache at time T_n+, (7) is a first-order linear ODE, whose solution is given by

$$X(t) = \frac{\sigma N_n}{Z_n} + \left(X_n - \frac{\sigma N_n}{Z_n}\right) e^{-(t-T_n)Z_n} \tag{8}$$

for $t \in [T_n, T_{n+1})$.

It is worth noting from (8) that the ratio $X(t)/c$, which gives the hit probability at time t (see (6)), is less than 1 for all $t > 0$ as long as $X(0)$, the amount of fluid at time $t = 0$, is less than c. We will assume from now on that $0 < X(0) < c$, although this assumption can be relaxed since we will only be interested in the steady-state behavior of the system. More specifically, we will show (see proof of Proposition 3.1) that the stationary expected amount of fluid in the system – denoted by $\mathbb{E}[X]$ – exists and is independent of the initial state $(N(0), X(0))$.

Hence, we define the *stationary hit probability* p_H (or simply the hit probability) as the ratio of the stationary expected amount of fluid in the system to the number of available documents, that is

$$p_H = \frac{\mathbb{E}[X]}{c}. \tag{9}$$

For the sake of convenience we introduce the new parameters

$$\gamma := \frac{\sigma}{\mu c} \quad \text{and} \quad \alpha := \frac{\theta c}{\sigma}\ . \tag{10}$$

A first expression for the hit probability p_H is derived in the following proposition.

Proposition 3.1 *The hit probability is given by*

$$p_H = e^{-\rho} \sum_{i=1}^{\infty} \frac{\rho^i}{i!} v_i \tag{11}$$

where the constants $v_1, v_2, \ldots$ satisfy the infinite linear recursion

$$(\rho + \alpha\gamma + (\gamma+1)i)\, v_i = \gamma i + i\Delta_u(i-1)v_{i-1} + \rho\Delta_d(i+1)v_{i+1}, \quad i \geq 1, \tag{12}$$

with $v_0 := 0$.

The proof of Proposition 3.1 is given in appendix A. It is shown in this proof that v_i is the stationary hit probability just before a *jump epoch* given that i nodes are active. The expression in (11) is not amenable to efficient computation, since it involves the solution of an infinite system of linear equations and the computation of an infinite series.

Building on Proposition 3.1, the next result provides an alternative expression for the hit probability, which will turn out to be more amenable to numerical computation than (11). This is done for the cases (i) $\Delta_d(i) = (i-1)/i$, $\Delta_u(i) = 1$ and (ii) $\Delta_d(i) = \Delta_u(i) = 1$.

Proposition 3.2 *Assume that $\Delta_u(i) = 1$ (no loss of content at node arrival).*

If node departures are not announced (i.e. $\Delta_d(i) = (i-1)/i$) then

$$p_H = e^{-\frac{\gamma\rho}{\gamma+1}} \gamma^{-(1+\kappa)} \int_{\frac{1}{\gamma+1}}^{1} \gamma\rho e^{\frac{\gamma\rho t}{\gamma+1}} \left(t(\gamma+1)-1\right)^{\kappa} dt \tag{13}$$

where $\kappa := \gamma(\alpha(\gamma+1)+\rho)/(\gamma+1)^2$.

If node departures are announced (i.e. $\Delta_d(i) = 1$) then

$$p_H = \rho e^{-\frac{\rho\gamma}{\gamma+1}} \gamma^{-\nu} \int_{\frac{1}{\gamma+1}}^{1} (\gamma t e^{\rho t} - v_1) e^{-\frac{\rho t}{\gamma+1}} \left((\gamma+1)t-1\right)^{\nu-1} dt \tag{14}$$

with $v_1 := \dfrac{\int_0^{1/(\gamma+1)} \gamma t e^{\frac{\rho\gamma t}{\gamma+1}} \left(1-(\gamma+1)t\right)^{\nu-1} dt}{\int_0^{1/(\gamma+1)} e^{\frac{\rho t}{\gamma+1}} \left(1-(\gamma+1)t\right)^{\nu-1} dt}$ *and* $\nu := \frac{\alpha\gamma(\gamma+1)+\rho}{(\gamma+1)^2}$.

Proof. For $0 \leq z \leq 1$ introduce the generating function

$$F(z) = \sum_{i=1}^{\infty} \frac{\rho^i}{i!} v_i z^i. \tag{15}$$

Observe that $0 \leq F(z) \leq \exp(\rho z)$ since $0 \leq v_i \leq 1$ for all $i \geq 1$. With (15) the hit probability p_H given in (11) rewrites

$$p_H = e^{-\rho} F(1). \tag{16}$$

It remains to determine $F(1)$, which is done below.

Assume first that $\Delta_d(i) = (i-1)/i$. Multiplying both sides of (12) by $\rho^i z^i / i!$ and summing the resulting equation over all values of $i \geq 1$ yields, after easy algebra,

$$\left(\rho + \alpha\gamma - \rho z + \frac{1}{z}\right) F(z) + ((\gamma+1)z - 1) \frac{d}{dz} F(z) = \gamma\rho z e^{\rho z}, \quad z \in (0,1). \tag{17}$$

Equation (17) defines an ODE for $F(z)$, with the initial condition $F(0) = 0$. Letting $z = 1/(\gamma+1)$ in (17) we see that necessarily

$$F\left(\frac{1}{\gamma+1}\right) = \frac{\gamma\rho e^{\rho/(\gamma+1)}}{(\gamma+1)(\gamma+1+\alpha\gamma+\gamma\rho/(\gamma+1))}. \tag{18}$$

Since we only need to compute $F(1)$ (see (16)), it is enough to solve (17) for $z \in (1/(\gamma+1), 1]$, with the initial condition (18), and then to use the continuity of the function $F(z)$ at point $z = 1/(\gamma+1)$.

We first solve the standard homogeneous equation, then use the method of variation of constant. The homogeneous equation writes

$$\begin{aligned} \frac{d}{dz} F(z) &= \frac{\rho + \alpha\gamma - \rho z + \frac{1}{z}}{1 - (\gamma+1)z} F(z) \\ &= \left[\frac{\rho}{\gamma+1} + \frac{1}{z} - \frac{(\gamma+1)^2 + \alpha\gamma(\gamma+1) + \gamma\rho}{(\gamma+1)\left(z(\gamma+1)-1\right)}\right] F(z). \end{aligned} \tag{19}$$

Its solution is

$$F(z) = C e^{\frac{\rho z}{\gamma+1}} z \left(z(\gamma+1) - 1\right)^{-(1+\kappa)} \tag{20}$$

where κ is defined in the statement of the proposition, and where C is an integration constant. Considering now C as a function of z, we routinely find from (17) and (20) that $C = C(z)$ satisfies the equation

$$\frac{d}{dz} C(z) = \gamma\rho e^{\frac{\gamma\rho}{\gamma+1} z} \left(z(\gamma+1) - 1\right)^{\kappa}.$$

Solving for $C(z)$ gives

$$C(z) = \int_{\frac{1}{\gamma+1}}^{z} \gamma\rho e^{\frac{\gamma\rho t}{\gamma+1}} \left(t(\gamma+1) - 1\right)^{\kappa} dt + C_0 \tag{21}$$

where C_0 is a constant to be determined from the initial condition (18).

Since the exponent $-(1+\kappa)$ of $(z(\gamma+1)-1)$ in (22) is strictly negative, and since $F(1/(\gamma+1))$ is finite from (18), we conclude that necessarily $C(1/\gamma+1))=0$, which implies that the constant C_0 in (21) must be equal to zero. Therefore, cf. (20), (21),

$$F(z) = e^{\frac{\rho z}{\gamma+1}} z\,(z(\gamma+1)-1)^{-(1+\kappa)} \int_{\frac{1}{\gamma+1}}^{z} \gamma\rho e^{\frac{\gamma\rho t}{\gamma+1}}\,(t(\gamma+1)-1)^{\kappa}\,dt \tag{22}$$

for $z \in (1/(\gamma+1),1)$.

Letting $z \to 1$ in (22) and using (16) finally gives (13).

Assume now that $\Delta_d(i)=1$. In this case $F(z)$ satisfies the ODE

$$(\rho(1-z)+\alpha\gamma)F(z) + ((\gamma+1)z-1)\frac{d}{dz}F(z) = \rho(ze^{\rho z}-v_1), \quad z\in(0,1). \tag{23}$$

We only sketch the derivation of $F(z)$ as it does not offer any difficulty. The first step is to solve (23) separately for $z \in (0,1/(\gamma+1))$ and for $z\in(1/(\gamma+1),1)$, with the initial condition $F(0)=0$ and $F(1/(\gamma+1)) = \rho(e^{-\rho/(\gamma+1)}/(\gamma+1)-v_1)/(\rho+\alpha\gamma)$, respectively (the latter condition is obtained by setting $z=1/(\gamma+1)$ in (23)). The second and last step is to use the continuity of $F(z)$ at point $z=1/(\gamma+1)$, which gives a linear equation to be satisfied by v_1, from which we find v_1 and ultimately (14). This concludes the proof.

■

Proposition 3.2 provides a low-complexity formula for the computation of p_H. The only difficulty lies in the evaluation of the various exponentials, especially when ρ is large or equivalently (see (2)) when the expected number of active nodes is large. In this case, a good accuracy can be achieved by rewriting p_H in the form $p_H = \int_{1/\gamma+1}^{1} e^{f(t,\rho,\alpha,\kappa)}dt$, where the mapping f can easily be identified from (13) (resp. (14)). Using this method, the average CPU time needed to compute the hit probability using (13) or (14) is typically less than a second with a Intel 4 2GHz/768Mo workstation, even for networks as large as a million nodes.

Next, we address the situation where documents may have a different popularity.

4 Hit Probability: Zipf-like Popularity Case

Following [3] we assume that the popularity of the documents follows a Zipf-like distribution. This implies that the probability ψ_n that the n-th most popular object is requested, is given by

$$\psi_n = \frac{\Omega}{n^\beta} \quad \text{for } n=1,\ldots,c \tag{24}$$

with $0<\beta\le 1$, where $\Omega := 1/\sum_{i=1}^{c} i^{-\beta}$ is a normalization factor. When $\beta=1$ then we have the Zipf's law. (For sake of comparison, note that $\psi_n = 1/c$ under the equally likely popularity assumption – see analysis in Section 3.)

The next step is to replace (6) by an expression that takes into account the popularity of the documents, as defined in (24). If we assume that the $X(t)$ cached objects at time t are the most

popular ones, a natural choice for $\mathbb{P}[\text{hit}|N_n, X(t)]$ is (with $\lfloor x \rfloor$ the largest integer less than or equal to c)

$$\mathbb{P}[\text{hit}|N_n, X(t)] = \sum_{i=1}^{\lfloor X(t) \rfloor} \frac{\Omega}{i^\beta} \approx \frac{X(t)^{1-\beta} - 1}{c^{1-\beta} - 1} \tag{25}$$

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 $x \geq 1$.

Unfortunately, with this hit probability function equation (5) has no closed-form solution, which does not allow us to develop the same kind of analysis as in Section 3.

Instead, we suggest to approximate the hit probability by dividing the set of c documents into K popularity classes of size c_k, $1 \leq k \leq K$ ($\sum_{k=1}^{K} c_k = c$) and to assume that documents belonging to the same class have the same popularity. By doing this, the hit probability within each class can be computed by using Proposition 3.2.

More specifically, assume that the K classes are ordered according to the popularity of their documents, with class 1 containing the most popular documents, class 2 the second most popular documents, etc.

We define the global hit rate p_H as a weighted sum of the intra-class hit probabilities, that is,

$$p_H = \sum_{k=1}^{K} q_k \, p_H^k \tag{26}$$

with p_H^k the hit rate for documents of class k, and q_k the probability that a document of class k is requested. From (24) we see that

$$q_k = \mathbb{P}[\text{request for class } k] = \sum_{i=1}^{c_K} \frac{\Omega}{(\sum_{l=1}^{k-1} c_l + i)^\beta}, \quad k = 1, 2, \ldots, K. \tag{27}$$

This formula is obtained by summing up the popularities of all documents in class k, with $\Omega/(\sum_{l=1}^{k-1} c_l + i)^\beta$ the popularity of the i-th most popular document of class k.

The intra-class hit probability p_H^k is obtained from Proposition 3.1 by replacing the parameters α and γ in (13) and (14) by $\alpha_k = \theta c_k/(\sigma q_k)$ and $\gamma_k = \sigma q_k/(\mu c_k)$, respectively.

It remains to specify how to choose the number of classes K and the number of objects assigned to each class.

We first select the number of classes K. This number has to be low enough for computational efficiency, but large enough to capture the effect of the skew factor β on the hit probability. Clearly, the accuracy of this approximation will only increase with the number of classes. As a result, we simply choose the highest value of K that leads to an affordable computation. In Section 6.2 we will search for an acceptable number of classes through a comparison with a simulation of the real system.

Once K is chosen, we need to calculate the number of objects c_k assigned to each class k, $1 \leq k \leq K$. This is a classical clustering problem that can be solved with a scalar quantization algorithm (see e.g. [8]), which also readily provides the q_k coefficients. Given the initial popularity

vector $(\psi_1, ..., \psi_c)$, the vector quantization algorithm aims at finding the class vector $(\phi_1, ..., \phi_K)$ that minimizes

$$E = \sum_{n=1}^{c} d(\psi_n, Q(\psi_n)),$$

where $d(.)$ is a distance measure (in our case the Euclidean distance) and $Q(\psi_n)$ the quantified version of ψ_n in the set $\{\phi_1, ..., \phi_K\}$, namely,

$$Q(\psi_n) = \arg\min_{\phi_k} d(\psi_n, \phi_k). \tag{28}$$

The quantity ϕ_k can be understood as the average popularity of documents in class k. Therefore the q_k coefficients are given by

$$q_k = c_k \phi_k, \quad 1 \leq k \leq K.$$

In order to determine the set $\{\phi_1, ..., \phi_K\}$ we used the Lloyd algorithm [8, page 189] that can be seen as an application of the Expectation-Maximization (EM) algorithm. This algorithm is composed of the following four steps:

S1 Initialize $(\phi, \ldots, \phi_K)$ (for example by using random sampling);

S2 For $n = 1, ...N$, estimate $Q(\psi_n)$ from (28): for each ϕ_k we obtain c_k corresponding objects;

S3 For $k = 1, 2, \ldots, K$, re-estimate ϕ_k: $\phi_k = (1/c_k) \sum_{n:Q(\psi_n)=\phi_k} \psi_n$;

S4 Go back to step 2 (S2) until convergence.

Since this algorithm is based on EM, the error will decrease at each iteration so that the set $\{\phi_1, ..., \phi_K\}$ will converge to a local optimum. In practice, this algorithm provides the optimal vector $(\phi_1, ..., \phi_K)$ along with the corresponding $(c_1, ..., c_K)$ values.

This approximation is validated in Section 6.2.

5 Qualitative Observations

In this section we investigate the impact on the hit probability of the document popularity distribution (Section 5.2) and of announced/unannounced departures (Section 5.3).

5.1 Experimental Setup

We used Matlab to compute the hit probability from (13), (14) and (26) with the following parameters

$$\begin{cases} c &= 10^7 \text{ files} \\ \sigma &= 10^{-3} \text{ requests per second and per user} \\ \theta &= 10^{-6} \text{ s}^{-1} \text{ (corresponding to a 11-day TTL)} \\ \mu &= 10^{-7} \text{ s}^{-1} \text{ (corresponding to 3 failures/departures per year and per user)} \end{cases}$$

With the above values, we see from (10) that

$$\begin{cases} \gamma &= 10^{-3} \\ \alpha &= 10^{4}. \end{cases}$$

For the Zipf-like distribution we used $\beta = 0.7$ (cf. [3]) and an approximation of $K = 10$ classes for 10^7 documents (cf. Section 6.2 for a discussion on the choice of K).

We also investigate the role of the mean online time on the hit probability in Section 5.3 by setting $\rho = 10^5$ and varying μ instead. This case will be explicitly mentioned.

5.2 Impact of the Zipf-like Popularity

Figure 1 displays the hit probability for the Squirrel system with unannounced departures as a function of the expected number of active nodes ρ, for uniform and Zipf-like document popularity distributions. In both cases, the hit probability is an increasing function of the size (i.e. ρ) of the network, which reflects the self-scaling nature (and therefore the scalability) of the Squirrel system.

We can see on Figure 1 that the document probability distribution has as an important impact on the hit probability. More specifically, the Zipf-like document popularity distribution generates higher hit probability than the uniform popularity for small and medium-sized networks (say up to 10^4-10^5 active nodes on average). This is a rather intuitive result since when the popularity is skewed, many requests can be served with only a few popular cached documents.

From this, we conclude that the document probability distribution is a crucial performance factor, which must be carefully modeled.

One can also use Figure 1 to determine the minimum network size necessary for an acceptable performance. For instance, with the experimental setting in Section 5.1, 8000 nodes must be active on the average with the Zipf-like distribution if one wants the hit probability to exceed $1/2$.

5.3 Utility of Announced Departures

In this section we evaluate the benefit of announcing departures on Squirrel performance. We compare the hit probability of the Squirrel system in the case of abrupt failures and announced departures. We do this for the uniform popularity case, using (13)-(14).

In Figure 2 we show the hit probability as a function the the network size. As expected (cf. Section 2.1 and 2.3), the hit probability is improved when users are able to announce their departure. However, the improvement that this feature brings is rather small, typically a 5% improvement over the abrupt failure case. Therefore, the interest of announcing departures has to be balanced with the overhead cost that this feature will induce, and which is due to departing nodes transferring their content to their neighbors.

We can expect this tradeoff to depend strongly on the mean online time of peers $1/\mu$. In particular, if peers disconnect much more often than 3 times a year as assumed in Figure 2, the cost of not announcing departures may be much more important - as well as the overhead cost. In Figure 3 we compare the hit probability of the Squirrel system for announced departures and abrupt failures for various departure rates, ranging from 10^{-7} (3 departures per year) to 1^{-5}

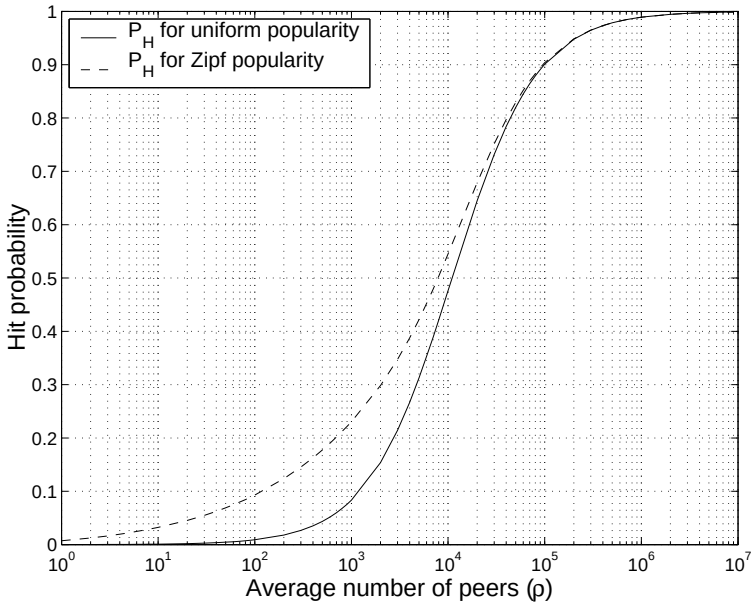

Figure 1: Hit probability of Squirrel for various document popularity distributions.

(around 1 departure per day). For this experiment we used a network size of $\rho = 10^5$ nodes and $\theta = 10^{-5}$ (24 hours TTL).

We observe that the performance of the system does not depend on γ (i.e., on μ in this experiment) when nodes are able to announce their departure. While this property is not directly visible from the expression in (14), it is fairly intuitive since an announced departure does not generate performance degradation, unlike abrupt failures. We also observe that the performance degradation due to abrupt failures only becomes significant for $\gamma \leq 10^{-4}$, corresponding to $\mu \geq 10^{-6}$, or a mean online time of 11 days at most.

6 Experimental Validation

The goal of this section is to validate the fluid model approximation of requests, as well as the clustering approximation of document popularity, against a discrete-event simulation of the Squirrel system.

6.1 Uniform Popularity Case

We compared the hit probability when the node dynamics are modeled as the number of customers in a M/M/∞ queuing system, given in (13), to the corresponding formula in [4, Prop. 4.1] where

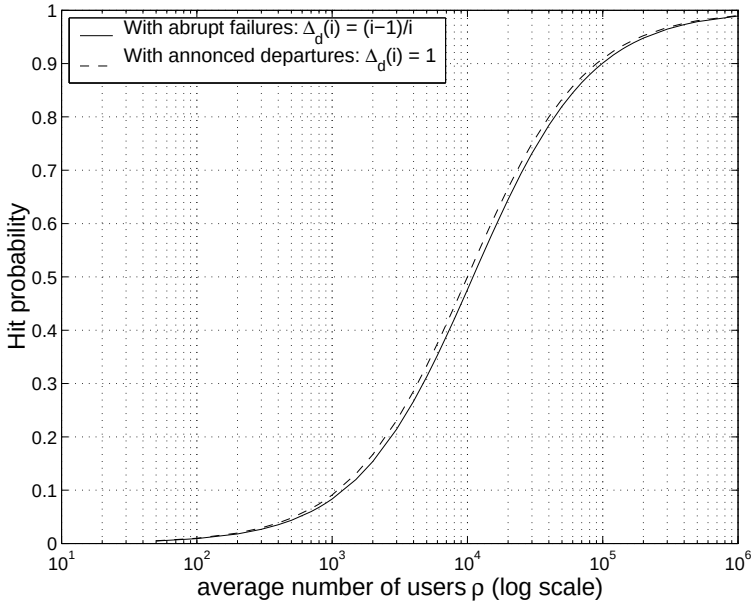

Figure 2: Hit probability of Squirrel for announced and unannounced node departures as a function of the network size.

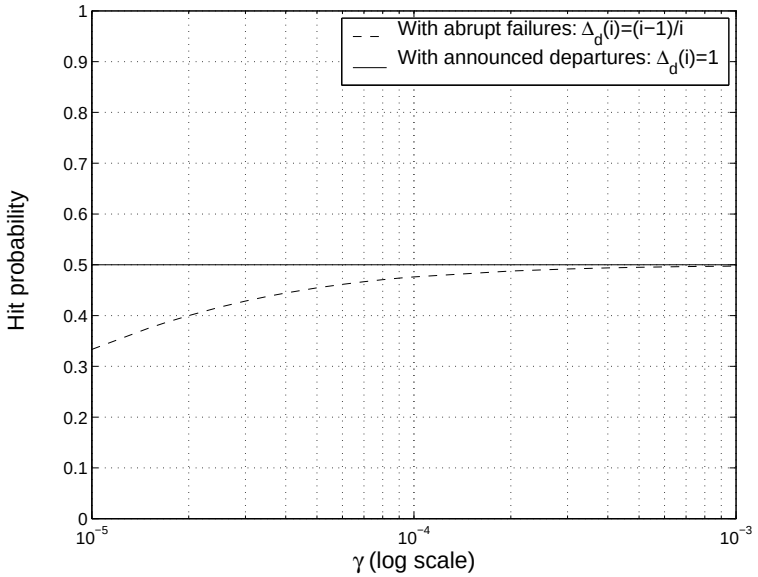

Figure 3: Hit probability of Squirrel for announced and unannounced node departures as a function of the mean online time. (network of 100,000 nodes.)

node dynamics are represented by the Engset model. To do so we fixed the mean number of active nodes to the same value in both models and varied it between 1 and 11000 nodes (range of tractability of the hit probability obtained via the Engset model). The hit probability with the Engset model was computed using Maple V.

We found that both models consistently predict the same hit probability over all range of loads (i.e. mean number of active nodes), even for very small networks. The relative error was always smaller than 10^{-4}. Therefore, we can expect both models to describe the Squirrel system with the same accuracy.

In [4], we compared the theoretical results obtained via the Engset model to a discrete-event simulation of the Squirrel system with uniform popularity distribution. The simulation validates the fluid model approximation by using Poisson arrivals for requests and by allowing concurrent requests. We found that the theoretical hit probability was remarkably close to the hit probability obtained through simulations (see [4] for details).

We can therefore safely conclude from the above that the hit probability computed via the $M/M/\infty$ model offers the same accuracy as the one obtained via the Engset model, at least in the uniform popularity case (the analysis in [4] was only carried out for uniformly popular objects). In particular, we can reasonably extrapolate that the model developed in this paper is a good approximation of Squirrel's behavior when deployed on large networks (say larger than 10,000 users), a situation where both discrete-event simulations and the model in [4] fail to work.

6.2 Zipf-like Popularity Case

In Figure 4 we compare our multiclass approach (see 4) to a discrete-event simulation of the Squirrel system with a Zipf-like popularity distribution. The parameters were

$$c = 40,000 \text{ files}, \quad \rho = 9.99 \text{ nodes}, \quad \theta = 10^{-3} \text{ s},^{-1} \quad \mu = 10^{-7} \text{ s},^{-1} \quad \beta = 0.7$$

and we varied the request rate σ. Simulation results are accurate at 0.2% with 99% confidence.

Figure 4 shows that our multiclass model is able to approximate very closely the hit probability of the simulated system: with 10 classes the curve follows already closely the same shape as the curve obtained by simulation, and with 100 classes the relative error amounts to 1%. We conclude that the combination of the M/M/∞ model for node dynamics and of the multiclass approach for modeling the different document popularities provides a very accurate estimation of Squirrel behavior and performance.

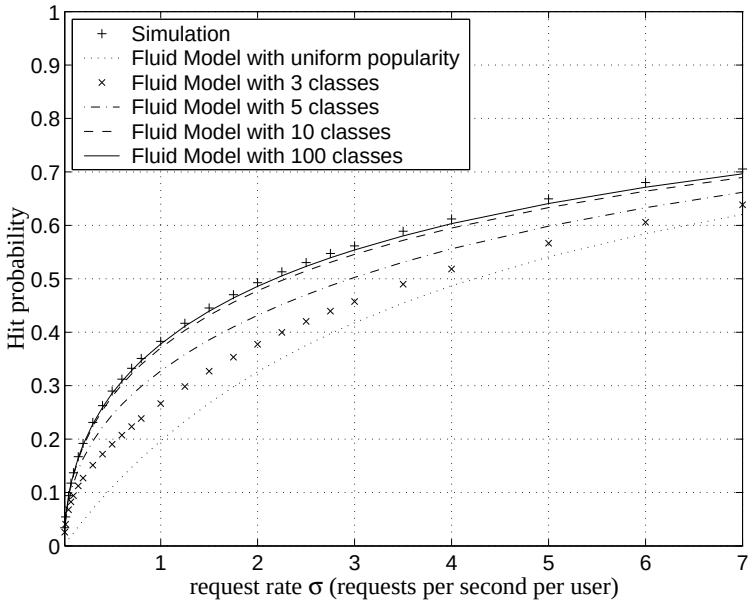

Figure 4: Comparison of the multiclass M/M/∞ model with a discrete-event simulation of Squirrel under a Zipf-like popularity distribution.

7 Conclusion

We proposed a stochastic fluid model for the Squirrel peer-to-peer cooperative caching system. This model, based on M/M/∞ node dynamics, approximates the request streams of the individual

nodes by a fluid flow. The model can be viewed as a scalable extension of our previous Engset-based fluid model. The new model turns out to be tractable for any network size and is also more convenient than our previous model. In addition, the model also allowed us to study the effect of nodes announcing their departure on the resulting hit probability.

Furthermore, we proposed, implemented and evaluated a multiclass approach to take variable object popularity into account. We found that this method gives accurate results even with a small number of classes.

Finally, this paper illustrates a stochastic fluid model approach through the example of the Squirrel system. This approach can also be straightforwardly applied to other systems based on distributed hash tables such as Chord or CAN ([16,14]) for example, since it mainly relies on the load balancing, provided by Pastry, and the absence of replication that characterizes the home-store scheme of Squirrel.

Future work will intend to adapt the fluid model approach to analyze other types of content distribution systems which require to take into account document replication.

A Proof of Proposition 3.1

We compute the stationary hit probability under the linear model (with uniform popularity):

$$p_H = \frac{\mathbb{E}[X]}{c} \tag{29}$$

The idea of the proof is to first compute the expected amount of cached fluid just before a jump in the process $\{N(t)\}_t$ conditioned on the value of $N(t)$ just before this jump, and then to invoke Palm calculus to deduce the expected amount of cached fluid at *any time*.

Let Y_n be the amount of cached fluid just before the $(n+1)$-st jump in the process $\{N(t)\}_t$ (i.e. $Y_n = X(T_{n+1}-)$). We first compute $v_i \stackrel{\text{def}}{=} \lim_{n\to\infty}(1/c)\mathbb{E}[Y_n \mid N_n = i]$ for $i \geq 1$.

$$\begin{aligned}
\lim_{n\uparrow\infty} \mathbb{E}[Y_n | N_n = i] &= \lim_{n\uparrow\infty} \mathbb{E}\left[\frac{\sigma N_n}{\frac{\sigma N_n}{c}+\theta} + \left(X_n - \frac{\sigma N_n}{\frac{\sigma N_n}{c}+\theta}\right) e^{-(T_n+1-T_n)(\theta+\sigma N_n/c)} | N_n = i\right] \\
&= \lim_{n\uparrow\infty} \frac{\sigma i}{\frac{\sigma i}{c}+\theta} \times \frac{\frac{\theta}{\mu}+\frac{\sigma i}{\mu c}}{\frac{\theta}{\mu}+\frac{\sigma i}{\mu c}+\rho+i} + \frac{(\rho+i)\,\mathbb{E}[X_n \mid N_n = i]}{\rho+i+\frac{\theta}{\mu}+\frac{\sigma i}{\mu c}}.
\end{aligned} \tag{30}$$

To derive (30) we have used the fact that, given $N_n = i$, the random variables X_n and $T_{n+1} - T_n$ are independent, and $T_{n+1} - T_n$ is exponentially distributed with parameter $\lambda + \mu i$.

Let us now evaluate $\lim_{n\uparrow\infty} \mathbb{E}[X_n \,|\, N_n = i]$ for $i \geq 1$. Conditioning on N_{n-1} we have

$$\begin{aligned}
\lim_{n\uparrow\infty} \mathbb{E}[X_n \,|\, N_n = i] &= \lim_{n\uparrow\infty} \mathbb{E}\left[X_n \,|\, N_n = i, N_{n-1} = i-1\right] P\left(N_{n-1} = i-1 \,|\, N_n = i\right) \\
&\quad + \lim_{n\uparrow\infty} \mathbb{E}\left[X_n \,|\, N_n = i, N_{n-1} = i+1\right] P\left(N_{n-1} = i+1 \,|\, N_n = i\right) \\
&= \Delta_u(i-1) \lim_{n\uparrow\infty} \mathbb{E}[Y_{n-1} \,|\, N_{n-1} = i-1] \frac{\pi_{i-1}}{\pi_i} \frac{\rho}{\rho+i-1} \\
&\quad + \Delta_d(i+1) \lim_{n\uparrow\infty} \mathbb{E}[Y_{n-1}, \,|\, N_{n-1} = i+1] \frac{\pi_{i+1}}{\pi_i} \frac{i+1}{\rho+i+1} \\
&= c \frac{i\Delta_u(i-1)v_{i-1} + \Delta_d(i+1)v_{i+1}\,\rho}{\rho+i} \qquad (31)
\end{aligned}$$

by using (4) and the definition of v_i. Finally, dividing both sides by c and introducing (31) into (30) yields

$$(\rho + i + \frac{\theta}{\mu} + \frac{\sigma i}{\mu c})\, v_i = \frac{\sigma i}{c\mu} + i\Delta_u(i-1)v_{i-1} + \rho\Delta_d(i+1)v_{i+1} \qquad (32)$$

for $i \geq 1$, or equivalently (12) by using (10).

The v_i coefficients in (12) gives the conditional stationary expected amount of fluid correctly cached just before jump epochs (up to a multiplicative constant). However, the hit probability p_H in (29) is defined in terms of the stationary expected amount of fluid correctly cached at *arbitrary* epochs. The latter metric can be deduced from the former one by using Palm calculus, through the identity (see e.g. [2, Formula (4.3.2)])

$$\mathbb{E}\left[X\right] = \Lambda \mathbb{E}^0 \left[\int_0^{T_1} X(t)dt\right] \qquad (33)$$

where $\mathbb{E}^0$ denotes the expectation with respect to the Palm distribution (i.e. assuming that a jump occurs at time 0 and that the system is in steady-state at time 0), T_1 denotes the time of the first jump after 0, and where Λ denotes the global rate of the M/M/∞ model, i.e.

$$\Lambda = \frac{1}{\mathbb{E}^0[T_1]}. \qquad (34)$$

From now on we assume that the system is in steady-state at time 0. Under the Palm distribution we denote by N_{-1} and Y_{-1} the number of up caches and the amount of correctly cached fluid respectively, just before time 0 (i.e. just before the jump to occur at time 0).

We first compute $1/\Lambda$. We have

$$\frac{1}{\Lambda} = \sum_{i=0}^{\infty} \pi_i\, \mathbb{E}^0[T_1 \,|\, N_0 = i] = \frac{1}{\mu} \sum_{i=0}^{\infty} \frac{\pi_i}{\rho+i} = \frac{1}{2\rho\mu} \qquad (35)$$

by using (3)-(4).

Let us now determine $\mathbb{E}[X]$. From (33), (8), (35) we find

$$\begin{aligned}
\mathbb{E}[X] &= \Lambda \sum_{i=1}^{N} \pi_i \, \mathbb{E}^0 \left[\int_0^{T_1} \left(\frac{\sigma N_0}{\frac{\sigma N_0}{c} + \theta} + \left(X_0 - \frac{\sigma N_0}{\frac{\sigma N_0}{c} + \theta} \right) e^{-t(\theta + \sigma N_0/c)} \right) dt \mid N_0 = i \right] \\
&= \Lambda \left[\sum_{i=1}^{\infty} \pi_i \frac{\sigma i}{\frac{\sigma i}{c} + \theta} \mathbb{E}^0[T_1 \mid N_0 = i] \right. \\
&\quad \left. + \sum_{i=1}^{\infty} \frac{\pi_i}{\theta + \sigma i/c} \mathbb{E}^0 \left[\left(X_0 - \frac{\sigma i}{\frac{\sigma i}{c} + \theta} \right) \left(1 - e^{-T_1(\theta + \sigma i/c)} \right) \mid N_0 = i \right] \right] \\
&= \Lambda \left[\sum_{i=1}^{\infty} \pi_i \frac{\sigma i}{\theta + \frac{\sigma i}{c}} \frac{1}{\lambda + \mu i} + \sum_{i=1}^{\infty} \frac{\pi_i}{\theta + \sigma i/c} \left(\mathbb{E}^0[X_0 \mid N_0 = i] - \frac{\sigma i}{\frac{\sigma i}{c} + \theta} \right) \right. \\
&\quad \left. \times \left(1 - \mathbb{E}^0 \left[e^{-T_1(\theta + \sigma i/c)} \mid N_0 = i \right] \right) \right] \qquad (36) \\
&= \frac{\Lambda}{\mu} \left[\sum_{i=1}^{\infty} \pi_i \frac{\sigma i}{\theta + \frac{\sigma i}{c}} \frac{1}{\rho + i} + \sum_{i=1}^{\infty} \pi_i \frac{\mathbb{E}^0[X_0 \mid N_0 = i] - \frac{\sigma i}{\theta + \frac{\sigma i}{c}}}{\rho + i + \frac{\theta}{\mu} + \frac{\sigma i}{\mu c}} \right] \\
&= 2\rho \sum_{i=1}^{\infty} \pi_i \left[\frac{\sigma i}{\theta + \frac{\sigma i}{c}} \frac{1}{\rho + i} + \frac{\mathbb{E}^0[X_0 \mid N_0 = i] - \frac{\sigma i}{\theta + \frac{\sigma i}{c}}}{\rho + i + \frac{\theta}{\mu} + \frac{\sigma i}{\mu c}} \right]. \qquad (37)
\end{aligned}$$

By definition, $\mathbb{E}^0[X_0 \mid N_0 = i] = \lim_{n \uparrow \infty} \mathbb{E}[X_n \mid N_n = i]$, which has been computed in (31). By combining (31) and (12) we obtain

$$\mathbb{E}^0[X_0 \mid N_0 = i] = \frac{(\rho + i + \frac{\theta}{\mu} + i\frac{\sigma}{\mu c}) c v_i - i\frac{\sigma}{\mu}}{\rho + i}.$$

Plugging this value of $\mathbb{E}^0[X_0 \mid N_0 = i]$ into the r.h.s. of (37), and using (4), yields after some straightforward algebra

$$\mathbb{E}[X] = c e^{-\rho} \sum_{i=1}^{\infty} \frac{\rho^i}{i!} v_i. \qquad (38)$$

According to (29) it remains to divide both sides of (38) by c to get (11). This concludes the proof.

B Stationary Distribution of the M/M/∞ Model at Jump Times

In this section we compute the invariant distribution of the Markov chain $\{N_n, n \geq 1\}$. Let $P = [p_{i,j}]_{i,j \geq 0}$ be its transition probability matrix. We have : $p_{i,i+1} = \rho/(\rho + i)$ for $i \geq 0$, $p_{i,i-1} = i/(\rho + i)$ for $i \geq 1$ and $p_{i,j} = 0$ if $|j - i| \neq 1$.

Note that this chain is periodic with period 2. Since this Markov chain has a finite-state space and is irreducible, it is positive recurrent [6, Cor. 5.3.19, 5.3.22]. Therefore, it possesses a unique stationary distribution $\pi = (\pi_0, \pi_1 \cdots)$ given by the (unique) solution of the equation $\pi P = \pi$ such that $\sum_{i=0}^{\infty} \pi_i = 1$ [9, page 208].

We proceed by induction to compute π. From the equation $\pi P = \pi$ we find that $\pi_1 = \frac{1}{1+\rho}\pi_0$ and $\pi_2 = \frac{\rho+2}{2}\rho\pi_0$. This suggests that

$$\pi_j = \frac{\rho + j}{j!}\,\rho^{j-1}\,\pi_0 \tag{39}$$

for $j = 1, 2, \ldots, N$. Assume that (39) holds for $j = 1, 2, \ldots, i$, with $i \geq 2$. Let us show that it still holds for $j = i + 1$. We have

$$\begin{aligned}
\pi_{i+1} &= \frac{\rho+i+1}{i+1}\left(\pi_i - \frac{\rho}{\rho+i-1}\pi_{i-1}\right) \\
&= \frac{\rho+i+1}{i+1}\left(\frac{\rho+i}{i!}\rho^{i-1} - \frac{\rho}{\rho+i-1} \times \frac{i-1+\rho}{(i-1)!}\rho^{i-2}\right)\pi_0 \qquad (40) \\
&= \frac{\rho+i+1}{i+1}\,\frac{\rho^i}{i!}\,\pi_0
\end{aligned}$$

where (40) follows from the induction hypothesis. The constant π_0 is computed by using the normalizing condition $\sum_{i=0}^{N} \pi_i = 1$; we find $\pi_0 = e^{-\rho}/2$ as announced in (3). Plugging this value of π_0 into (39) gives (4). ■

C Engset and M/M/∞ Models

In the Engset model, every user independently goes down (resp. up) after an exponentially distributed time with parameter μ (resp. λ_E). The total number of users (connected or not) is N. Let us denote by $N_E(t)$ the state of the Engset model at time t. Observe that this Engset model and the M/M/∞ model introduced in Section 2.3 have the same death rate in state $i \in \mathbb{N}$, given by $i\mu$. Let us define $\rho_E = \lambda_E/\mu$.

We have for the Engset model [11]:

$$P(N_E^{\infty} = i) = \binom{N}{i}\frac{\rho_E^i}{(1+\rho_E)^N}, \quad 1 \leq i \leq N \tag{41}$$

$$\mathbb{E}\left[N_E^{\infty}\right] = N\frac{\rho_E}{1+\rho_E} \tag{42}$$

as opposed to (1) and (2), respectively, for the M/M/∞ model.

In the following, the shorthand $f(N) \sim g(N)$ will stand for $\lim_{N\to\infty} f(N)/g(N) = 1$ for any mappings f and g.

Proposition C.1 *Assume that the mean number of active users in the $M/M/\infty$ model and in the Engset model with N nodes are the same, namely,*

$$\rho = \frac{N\rho_E}{1+\rho_E}. \tag{43}$$

Then, as $N \to \infty$, the stationary distribution of the Engset model defined in (41) is equivalent to that of the $M/M/\infty$ model given in (1), namely

$$P(N_E^\infty = i) \sim \frac{\rho^i}{i!} e^{-\rho} \tag{44}$$

for any $i \in \mathbb{N}$.

Proof. For any fixed $i \in \mathbb{N}$ we have from (41) and (43)

$$P(N_E^\infty = i) = \frac{N!}{i!(N-i)!} \frac{(\frac{\rho}{N-\rho})^i}{(1+\frac{\rho}{N-\rho})^N}. \tag{45}$$

Using the Stirling formula $N! \underset{N\to\infty}{\sim} (N/e)^N \sqrt{2\pi N}$ in (45) yields

$$\begin{aligned} P(N_E^\infty = i) &\underset{N\to\infty}{\sim} e^{-i} \left(\frac{N}{N-i}\right)^N (N-i)^i \, \frac{\rho^i}{i!} \frac{1}{(N-\rho)^i} \left(\frac{N-\rho}{N}\right)^N \\ &\underset{N\to\infty}{\sim} e^{-i} \frac{\rho^i}{i!} \frac{1}{(1-\frac{i}{N})^N} \left(1-\frac{\rho}{N}\right)^N. \end{aligned}$$

With the identity $\lim_{N\to\infty} (1+x/N)^N = e^x$ applied to the last equation, we find (44), which completes the proof. ■

References

[1] Anick, D., Mitra, D., and Sondhi, M. M. Stochastic theory of data-handling systems with multiple sources. *Bell Systems Technical Journal 61* (1982), 1871–1894.

[2] Baccelli, F., and Brémaud, P. *Elements of Queueing Theory: Palm-Martingale Calculus and Stochastic Recurrences.* Springer Verlag, 1994.

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

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

[5] Clévenot, F., Nain, P., and Ross, K. W. Stochastic fluid models for cache clusters. *Performance Evaluation 59*, 1 (January 2005), 1–18.

[6] E.Çinlar. *Introduction to Stochastic Processes.* Prentice Hall, 1975.

[7] Ge, Z., Figueiredo, D., Jaiswal, S., Kurose, J., and Towsley, D. Modeling peer-peer file sharing systems. In *Proc. IEEE INFOCOM 2003* (San Francisco, California, April 2003).

[8] Gersho, A., and Gray, R. M. *Vector Quantization and Signal Compression.* Kluwer Academic Publishers, 1992.

[9] Grimmett, G., and Stirzaker, D. *Probability and Random Processes.* Oxford University Press, 1992.

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

[11] Kelly, F. P. *Reversibility and Stochastic Networks.* Wiley, Chichester, 1979.

[12] Kleinrock, L. *Queueing systems*, vol. 1. J. Wiley and sons, 1975.

[13] Qiu, D., and Srikant, R. Modeling and performance analysis of bittorrent-like peer-to-peer networks. In *Proceedings of ACM Sigcomm* (Portland, OR, Aug 2004).

[14] Ratnasamy, S., Francis, P., Handley, M., Karp, R., and Shenker, S. A scalable content-addressable network. In *Proc. ACM SIGCOMM 2001* (San Diego, California, August 2001).

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

[16] Stoica, I., Morris, R., Karger, D., Kaashoek, M. F., and Balakrishnan, H. Chord: A scalable peer-to-peer lookup service for internet applications. In *Proc. ACM SIGCOMM 2001* (San Diego, California, August 2001), pp. 149–160.

[17] Yang, X., and Veciana, G. D. Service capacity of peer-to-peer networks. In *Proc. INFOCOM 2004* (Hong Kong, March 2004).

[18] Zou, L., and Ammar, M. A file-centric model for peer-to-peer file sharing systems. In *Proc. ICNP 03* (Atlanta, Georgia, USA, November 2003).

Unité de recherche INRIA Sophia Antipolis
2004, route des Lucioles - BP 93 - 06902 Sophia Antipolis Cedex (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 Rhône-Alpes : 655, avenue de l'Europe - 38334 Montbonnot Saint-Ismier (France)
Unité de recherche INRIA Rocquencourt : Domaine de Voluceau - Rocquencourt - BP 105 - 78153 Le Chesnay Cedex (France)

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