HAL

archives-ouvertes.fr

Analysis of Failure Correlation in Peer-to-Peer Storage Systems

Olivier Dalle, Frédéric Giroire, Julian Monteiro, Stéphane Pérennes

▶ To cite this version:

Olivier Dalle, Frédéric Giroire, Julian Monteiro, Stéphane Pérennes. Analysis of Failure Correlation in Peer-to-Peer Storage Systems. [Research Report] RR-6771, INRIA. 2008, pp.35. <inria-00346857>

HAL Id: inria-00346857
https://hal.inria.fr/inria-00346857

Submitted on 12 Dec 2008

![INRIA logo]

Analysis of Failure Correlation in Peer-to-Peer Storage Systems

Olivier Dalle — Frédéric Giroire — Julian Monteiro — Stéphane Pérennes

N° 6771

December 2008

Rapport de recherche

Analysis of Failure Correlation in Peer-to-Peer Storage Systems

Olivier Dalle* , Frédéric Giroire* , Julian Monteiro* , Stéphane Pérennes*

Thème COM — Systèmes communicants
Projets Mascotte

Abstract: In this paper, we propose and study analytical models of self-repairing peer-to-peer storage systems subject to failures. The failures correspond to the simultaneous loss of multiple data blocks due to the definitive loss of a peer (or following a disk crash). In the system we consider that such failures happen continuously, hence the necessity of a self-repairing mechanism (data are written once for ever). We show that, whereas stochastic models of independent failures similar to those found in the literature give a correct approximation of the average behavior of real systems, they fail to capture their variations (e.g. in bandwidth needs). We propose to solve this problem using a new stochastic model based on a fluid approximation and we give a characterization of the behavior of the system according to this model (expectation and standard deviation). This new model is validated using comparisons between its theoretical behavior and computer simulations.

Key-words: P2P storage system, failure correlation, performance evaluation, data durability, Markov chain models, fluid models.

Analyse de la Corrélation des Pannes dans les Systèmes de Stockage Pair-à-Pair

Résumé : Dans ce papier, nous proposons et étudions des modèles analytiques pour des systèmes de stockage pair-à-pair sujet à des pannes. Ces pannes correspondent à la perte simultanée de multiples blocs de données en raison de la perte définitive d'un des pairs (ou à la suite d'une défaillance de disque dur). Comme de telles pannes ont lieu en continu dans le système, un mécanisme de reconstruction est nécessaire pour ne pas perdre de données. Nous montrons que si les modèles stochastiques proposés précédemment dans la littérature donnent une approximation correcte du comportement moyen des systèmes réels, ils ne capturent pas leur variations (par exemple en utilisation de bande passante) parce qu'ils supposent des pannes de bloc indépendantes. Nous proposons un nouveau modèle stochastique utilisant une approximation fluide pour résoudre ce problème et nous caractérisons le comportement du système (espérance et déviation) gâce à ce modèle. Ce dernier est validé par la comparaison des résultats théoriques avec ceux d'un jeu complet de simulations.

Mots-clés : système de stockage P2P, corrélation des pannes, évaluation de performance, durabilité des données, modeèles en chaînes de Markov, modèles fluides.

1 Introduction

Traditional means to store data are dedicated servers or magnetic tapes. These solutions are reliable but expensive. Recently, hard disks and bandwidth have become cheaper and widely available, allowing new forms of data storage on distributed, peer-to-peer (P2P) architectures. A large number of such systems have been built: Intermemory [7], Ocean Store [8], Freenet [5], CFS [6], Total Recall [3]. These systems are cheap to operate, but their highly distributed nature raises questions about reliability, durability, availability, confidentiality, and routing of the data.

Storage systems are prone to disk failures. Hence, to ensure the reliability and durability of the storage, some redundancy needs to be added to the system. Two common methods to add redundancy are replication and Erasure Correcting Codes (ECCs). It has been shown that ECCs usually are much more efficient than replication [17]. However, both methods require an additional self-repairing mechanism to maintain the level of redundancy after multiple failures.

In this paper, we primarily focus on P2P-based storage systems that have high durability requirements, such as backup systems. We assume that each subscriber of the backup service hosts a networked hard disk that runs the P2P backup protocol. As long as no failure occurs, the peers are assumed to be fully cooperative, always running and to be permanently connected to a common network or the Internet. The data of each customer are replicated and spread among numerous peers. The questions we address here are the following: given a certain level of redundancy and system size, under continuous disk failures, what is the probability that the system looses some data, or equivalently what is the data lifetime in the system? How much resources (bandwidth and storage space) are needed to ensure a given level of reliability (i.e. to ensure that the data loss probability is kept under a given value)?

First we study a Markov Chain Model (MCM) to model the system neglecting correlations that may exist between multiple data losses. In other words, we consider that each failure event applies to one data block and is independent from other failure events. Using simulations of a realistic system, we show that the average behavior of the system is accurately given by the model. However, we observe that the variations around this average behavior are much higher in the simulations than in the MCM. This variance is caused by the strong correlation between block failures, that are implemented in simulations but neglected in the MCM. As a matter of fact, the main cause of failure in such a system is a disk crash (or a peer leaving the system) which impacts tens of thousands of blocks at the same time. In this paper, we point out the strong impact of this correlation on the system behavior. Even for large systems with tens of thousands of peers these variations cannot be neglected. We show that a bandwidth provisioning decision not taking into account these variations, would lead to a significant loss of data. We then propose a new stochastic Fluid Model that does not model the behavior of a single block anymore but of the whole system, i.e. with peers containing many blocks. We provide a mathematical analysis of this model and validate it by simulations.

Along with faults correlation, we also point out the impact of peer age heterogeneity on the system. When a disk crashes, it gets replaced by a new disk with no data. This disk

then gets filled up gradually by the self repairing process. Hence, a young disk has less data and the system is much more affected by the fault of an old disk. We propose a Refined Fluid Model that takes into account this phenomenon and its corresponding mathematical analysis.

To the best of our knowledge, this paper is the first study to propose a model that takes into account the correlations between data blocks failures resulting from the fact that many data blocks are lost at the same time when a peer is definitely lost (or its disk crashes). We propose a complete analysis of this model and we show that the results of extensive simulations of a real system are close to what expected from the analysis.

More precisely, our contributions in this paper are the following:

- We study a Markov Chain Model explaining the average behavior of the bandwidth usage and giving the probability to loose data in the case of independent failures (Section 3).

- We underline the impact of fault correlation even on large systems and show that bad provisioning policy that do not take this phenomena into account could lead to an increased loss rate (Section 4).

- We propose a new Fluid Model to take into account this correlation. We give a full mathematical analysis of the model (Section 5).

- We bring to light the impact of disk age on the variations of the system and propose a new shuffling policy and a biased reconstruction policy to reduce this impact (Section 6).

- We validate our models using an extensive set of simulations.

Related work

An abundant literature exists on the topic of P2P storage systems. Several efforts to build large-scale self-managing distributed systems have been done, among others, Intermemory, Ocean Store, Freenet, PASTRY, CFS, Total Recall [7, 8, 5, 6, 3]. However, few analytical models have been proposed to estimate the behavior of the system (the data durability, resource usage, e.g., bandwidth) and understand the trade-offs between the system parameters.

In [17], the authors show that erasure code use an order of magnitude less bandwidth and storage than replication to provide similar system durability. In [16], a stochastic model is used to prove that, when there is a low peer availability, a replication scheme may be more efficient than erasure codes. Conditions where replication has to be preferred are provided in [10] , along with a discussion on the optimal number of fragments to use.

In [4], the authors show that a scalable system with high peer dynamicity and high availability is non feasible due to bandwidth limitations. Note that in the backup system

that we study here, we do not consider churn as the disks are almost continuously connected to the network. The behavior of a storage system using full replication is studied in [14]. A Markov Chain Model is used to derive the lifetime of the system, and other practical metrics like storage and bandwidth limits on the system. In [1], the authors also use a Markovian analysis, but for a system using ECCs. They analyze the performance of two different schemes (a centralized and a distributed) to recover the data and estimate the data lifetime and availability.

In all these models, the behavior of a single block is modeled and the block failures are considered independent. In this paper, we show that this assumption can lead to severe errors of estimation on the behavior of a system subject to peer failures. We propose new analytical models for the context of correlated block failures.

2 System Description

Assumptions on system dynamics. We consider here a P2P storage system designed for back-up. The data of a user are saved once for ever in the system. Since we mainly focus on the durability of the storage in presence of hardware failures (disk crashes), we choose to neglect the processes of new data insertions and new peer arrivals in the system and only focus on the process of maintaining the existing data initially saved in the system: our system has a fixed amount of user data and a constant number of peers. When a disk fails, the company replaces it after a short period of time. Reintroduced disks are empty. To keep the amount of saved data at a constant level, each time a saved user data is lost, it is reintroduced and dispatched randomly among peers. This purely formal assumption does not affect the system behavior because dead blocks practically do not happen.

Saved data maintenance process. The data are spread and stored among all peers in the system according to the following process: Each peer of this system has a disk and is continuously connected to the network. Users data are saved in units called the *initial blocks*. Each initial block is divided into s equally sized *fragments* to which are added r fragments of redundancy using erasure code encoding (see [11]) and resulting in a unit of storage called a *system block* or more simply a *block* (in the remaining of the paper, the unqualified term "block" will always refers to a "system block"). Each block has then $n = s + r$ fragments, verifying the property that *any* set of s fragments chosen among the $s + r$ initial fragments are sufficient to recover (reconstruct) the block. Thanks to this redundancy each block tolerate up to r faults (i.e. fragment loss). However, partial recovery is not possible: at least s fragments of the same block need to be concentrated at single place to make the computation and recover the block. The newly computed fragments are then spread at random in the system in a way that no peer receives more than one fragment of the same block.

Peer Failures. Peer are subject to failures, mainly disk crashes. Such events result in the loss of all the data, i.e., all the fragments present on the disk. Peers are assumed to

be all similar and to be fully cooperative. They get *faulty independently* according to a memoryless process. Given a *peer fault rate*, α, the probability for a peer to be alive after a time T is given by $a(T) = e^{-\alpha T}$. We note f the probability for a disk to experience a failure over a period T. We have $f = 1 - e^{-\alpha T}$. It is important to note that if, for a single disk, this event is rare, a system with thousands of disks continuously experiences failures. Different estimations of disk mean time to failure can be found in literature [13], roughly spanning from 2 years to 20 years. Hence, for example, a system with $20,000$ peers will experience more than one disk crash per hour assuming 2 years as an average lifetime. As a consequence, it is vital for the system to *monitor* the state of all the blocks and to be able to *reconstruct* the lost fragments.

Reconstruction. The system implements a *threshold-based policy*. When a block has less than r_0 fragments of redundancy left in the system, the recovery begins. r_0 is called the reconstruction threshold value. Note that the higher the threshold value, the lower the probability to loose data, but the higher the bandwidth usage of the system. The case $r_0 = r - 1$ is a special case called the *eager policy* where a block is reconstructed as soon as it lost a fragment. Note that studying different reconstruction policies (like, for example, rebuilding the most damaged blocks first) would be interesting.

When a block b has to be rebuilt, a peer is chosen uniformly at random to carry out the reconstruction. First, it has to download s of the remaining fragments. Then it rebuilds the block. Lastly, it sends the $r - r(b)$ (corresponding to the number of redundancy fragments missing in the system) to $r - r(b)$ in the system.

A reconstruction uses *system resources*. The fragments have first to be gathered and then spread again. The *bandwidth usage* is then sS_f (with S_f the size of a fragment) before reconstruction and $(r - r(b))S_f$ after, hence $(s + r - r(b))S_f$ in total. CPU time is consumed as well (the amount depends on the used ECCs). Our study allows to evaluate it, but it is not a critical resource with the current computers. So we only present results for the bandwidth. The *reconstruction time* needed by the system to rebuild a block is the sum of the time to detection, the time for a peer to download the remaining fragments, to rebuild the blocks, to cut it into fragments and to send them to random peers.

System state monitoring and resultant control traffic. Monitoring the state of the system is needed to decide which blocks have lost a critical number of their fragments. A method to do so is to record on which peers the fragments of each block are located and to detect peer failures. Different architectures are possible to monitor the state of the system. A centralized architecture with dedicated servers to store the fragment location information or a P2P architecture using a Distributed Hash Table (DHT) where peers are responsible for a subset of the other peers. Then, in an environment with collaborative peers and where disk crashes are the main cause of failures, knowing which peers are alive is enough. The *control traffic* is composed of:

Table 1: Summary of main notations.

N	# of peers
s	# of fragments in the initial block
r	# of redundancy fragments
$n = s + r$	# of fragments in a system block
$p(b)$	number of remaining fragments of block b
$r(b)$	# of remaining redundancy fragments of block b
r_0	reconstruction threshold value
S_f	size of a fragment in bytes
B	total number of blocks in the system
α	peer fault rate
τ	time step of the model
f	probability for a disk to experience a fault during a time step
$\delta(i)$	probability for a block at level i to loose one fragment

- Periodic pings to test if the machines are up and running. The time scale (periodicity) here is in the order of few hours. Indeed, we want to detect real failures and not temporary events, like a temporary electricity cut. The generated traffic is in the order of N bytes per hour;

- Assignments sent to a peer to reconstruct a block. This assignment may contains the identity of the peers that will receive the rebuild fragments. For each block reconstruction, this traffic is of the order of $(s + r) \log(NB)$ bits ($\log N$ being the size of a peer id and $\log B$ the size of a block id. Note that for a network of 10^{10} peers or blocks, this value is below 34.).

As we saw that the data traffic generated for the reconstruction roughly equals the size of a block, the control traffic is very limited compared to this reconstruction traffic ($S_f >> \log N + \log B$ is needed, which is true here as $S_f = 400$kB). Hence, we will not consider it in this paper. As so, our following analysis apply to any kind of control infrastructure.

3 Markov Chain Models

We propose here a Markov Chain Model (MCM) for our specific peer-to-peer storage system.

For the sake of clarity of presentation, we do not present here the most accurate and complex chains, but rather simplified versions (where unlikely transitions are ignored). Explicit expressions can be derived for these chains. They give very good approximations and provide the intuition of the system behavior. We actually used more sophisticated chains for our computations. We also use a poisson reconstruction time for mathematical tractability and because we think that it takes well into account the random nature of network delays.

Note that most other types of reconstruction could be captured by MCMs. For example, a reconstruction lasting a deterministic time can be modeled by labeling the states of a critical block by the progress of the reconstruction.

The Markov Chain approach allows to model more complex systems or behaviors. For example several classes of peers, with different disk sizes, fault rates, availabilities, or bandwidths, can easily be introduced. For k families of peers, the state of a block would be a k-uple $\{n_i\}, i \in \{0, k\}$ where n_i is the number of fragments on peers of family i. This would leads to Markov chains with n^k states that could be solved as long as k remains small.

These various chains all have a polynomial (in n) number of states. Hence stationary distributions and speed of convergence can be computed in less than a second on a computer.

3.1   Description

The behavior of a single block is modeled by a finite Markov Chain with discrete time of time step τ.

Markov Chain States. The chain has $r + 1$ states that each represent one of the possible levels of redundancy of the block and a *Dead* state. Three different kinds of states can be distinguished:

- **Non critical:** when the block, b, is such that $r_0 + 1 \leq r(b) \leq r$;

- **Critical:** when the block, b, is such that $0 \leq r(b) \leq r_0$;

- **Dead:** when the block has less than s fragments left and cannot be reconstructed. Its data are lost.

Markov Chain Transitions. A block can be affected by two different kinds of events: peer failures and reconstructions.

During the evolution of the system, some peers are failing. All the data on the hard disk is lost in these cases, leading to losses of fragments. Recall that f is the probability for a peer to get faulty during the timestep τ, $f = 1 - e^{-\alpha\tau}$. Note that we can choose the transition time of the system, τ, small enough to ensure that only one disk failure can happen per time step. Since a block has at most one fragment on a disk, disk failures will result in the loss of at most one fragment per time step in this case. The average number of fragments lost by a block is less than $(s + r)f$. A choice of τ such that $(s + r)f << 1$ is sufficient. Note also that the concurrent loss of several fragments will then be emulated by successive fragment losses.

The probability for a block at level i to loose one fragment during a time step is denoted by $\delta(i)$ and is given by

$$\delta(i) := (s + i)f(1 - f)^{s+i-1}.$$

A block at level 0 may die with probability $\delta(0)$. When a block becomes critical ($r(b) \leq r_0$), the reconstruction starts. The reconstruction is modeled as follows: the average duration of a reconstruction being noted θ, at each timestep, a critical block has a probability $\gamma := 1/\theta$
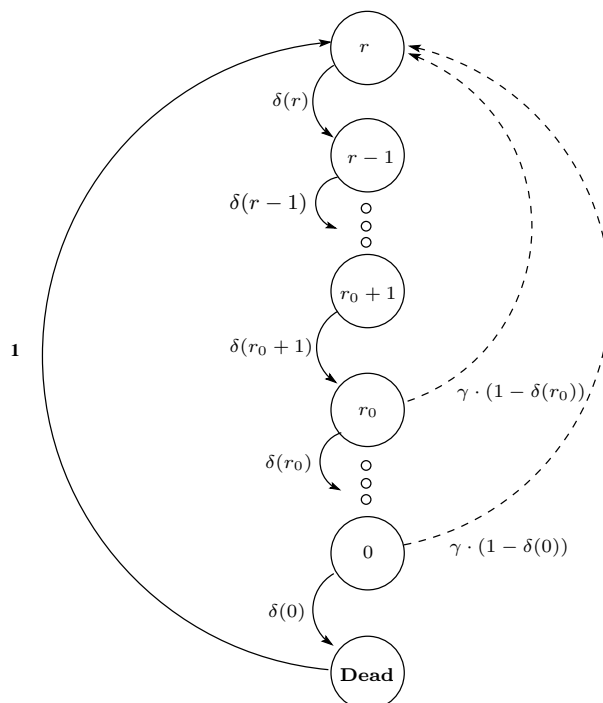
Figure 1: Markov chain modeling the behavior of one block. Solid and dashed lines respectively represent fault and reconstruction impacts.

to be rebuilt. Note, that we also assume that the blocks that lose a fragment during a time step can not be reconstructed during the same time step.

If a block looses all its redundancy fragments before being reconstructed ($p(b) < s$), the block gets lost: it goes to the *dead* state. Otherwise at the end of the reconstruction, the block switches to the top level of redundancy. In our model, due to the stability assumption (the number of blocks is constant in the system), each dead block immediatly get reborn as a full redundancy block. This assumption does not change the behavior of the system noticeably (dead blocks are very rare).

Hence, the Markov Chain transitions are given hereafter and depicted in Figure 1.

$$
\begin{array}{lll}
0 < i \leq r & i \to i-1 & \delta(i) \\
 & 0 \to Dead & \delta(0) \\
0 \leq i \leq r_0 & i \to r & \gamma \cdot (1 - \delta(i)) \\
 & Dead \to r & 1
\end{array}
$$

The transitions that leave the block in the same state should be added. These transitions, of course, are such that the outgoing transition probabilities sum up to 1. Note that in the system, we choose to implement the reconstruction after the impact of failures during the time step. Both choices are possible, it changes only slightly the transitions.

3.2 Analysis

Stationary distribution. The above finite Markov chain is irreducible and aperiodic. Hence, it admits a unique stationary distribution denoted by P. $P(x)$ is the stationary probability to be in state x. The stationary distribution of the Markov chain can be computed exactly in time polynomial in n by finding the eigenvector with eigenvalue 1. The complexity is independent of the number of blocks B, or of the number of peers N. Note that, because of the simple form of the system, we can compute the stationary distribution, as seen hereafter.

Remark on the initialization phase. Initially all the blocks have all their fragments ($r(b) = r$). As the system evolves and failures happen, some blocks drop to a lower level of redundancy. Hence, during a long transient phase the distribution tends to be extremely concentrated toward the maximum levels of redundancy. To avoid this phenomena, it could be good to introduce in the real system heterogeneous levels of redundancy at the start according to the stationary distribution. Note also that in a storage system where the peers appears gradually, the transient phase will be very limited.

Expression of the Stationary Distribution.
Let P denote the stationary distribution. $P(x)$ is the stationary probability to be in state x. As already claimed, for fixed values of the system parameters (θ, n, r_0), the stationary distribution can be computed in polynomial time by finding the eigenvector with eigenvalue 1 or simply by a Howard Perron Frobenius iteration. Note that, because of the simple form of the system (the system without the first row is lower triangular), we can compute the

stationary distribution more efficiently as follows:

$$
\begin{aligned}
P(r-1) &= \frac{\delta(r)}{\delta(r-1)}P(r) \\
P(r-2) &= \frac{\delta(r-1)}{\delta(r-2)}P(r-1) = \frac{\delta(r)}{\delta(r-2)}P(r) \\
\\
.. &= .. \\
r_0 < i < r \quad P(i) &= \frac{\delta(r)}{\delta(i)}P(r) \\
0 < i \le r_0 \quad P(i) &= \frac{\delta(i+1)}{\delta(i)+\gamma(1-\delta(i))}P(i+1) \\
P(Dead) &= \delta(0)P(0)
\end{aligned}
$$

All probabilities are expressed in function of $P(r)$ which can computed by the normalization:

$$
\sum_{i=0}^{r} P(i) + P(Dead) = 1.
$$

Expression of the Bounds for a Simplified Chain. To get a simpler expression of the stationary distribution, we present and analyze here a simplified chain. The goal is to give the intuition of how behaves the system in function of the parameters and to give explicit closed formulas.

The simplified chain assumes the following simplifying hypothesis: the probability to loose a fragment is $g = e(s+r)f$, the same for all levels. Note that the simplified chain is "pessimistic". Pessimistic here means that the bandwidth needed by the simpler chain is an upper bound of the bandwidth needed by the real system. The probability to loose a block is higher or, equivalently, the average block lifetime is lower. The transitions of the chain are then given by

$$
\begin{array}{lcc}
0 < i \le r & i \to i-1 & g \\
& 0 \to Dead & g \\
0 \le i \le r_0 & i \to r & \gamma(1-g) \\
& Dead \to r & 1
\end{array}
$$

Let us note $P^* = P(r)$ and express all the stationary probabilities in function of P^*. For all states i, such that $r_0 < i \le r$, we have $P(i) = gP(i+1) + (1-g)P(i)$. It gives

$$
P(i) = P(i+1) = P^*.
$$

For all states i such that $0 \le i \le r_0$, we have $P(i) = gP(i+1) + (1-g-\gamma(1-g))P(i)$. It gives

$$
P(i) = \frac{g}{g+\gamma(1-g)}P(i+1).
$$

That is, if we note $\rho := \frac{g}{g+\gamma(1-g)}$,

$$
P(i) = \rho^{r_0-i+1}P^*.
$$

Last, $P(dead) = gP(0)$, giving

$$P(dead) = g\rho^{r_0+1}P^*.$$

P^* is then evaluated using $\sum_{i=0}^{r} P(i) + P(dead) = 1$.

$$P^*\left(r - r_0 + (\rho + \ldots + \rho^{r_0+1} + g\rho^{r_0+1})\right) = 1.$$

That is

$$P^*\left(r - r_0 + \left(\frac{\rho - \rho^{r_0+2}}{1-\rho} + g\rho^{r_0+1}\right)\right) = 1.$$

We get the following approximations:

$$P^* \approx \frac{1}{r - r_0 + \frac{\rho}{1-\rho}}.$$

But we have $\frac{\rho}{1-\rho} = \frac{g}{\gamma(1-g)}$, simplifying the expression to

$$P^* \approx \frac{1}{r - r_0 + \frac{g}{\gamma(1-g)}}.$$

It directly gives the average number of blocks lost during a time step:

$$\boxed{\#\text{deads} \approx \rho^{r_0+1}\frac{g}{r - r_0 + \frac{g}{\gamma(1-g)}}B,}$$

with B the total number of blocks in the system. Note that the number of deads decreases exponentially with the threshold value r_0. Notice also that the base of the exponential $\rho := \frac{1}{1+\frac{\gamma}{g}(1-g)}$ depends of the proportion γ/g. As expected, the more the reconstruction ratio is greater than the failure ratio the less deads there are.

Similarly we get the average number of blocks under reconstructions during a time step (it is the sum of the blocks at level $0 \le i \le r_0$):

$$\boxed{\#\text{reconstructions} \approx \frac{\gamma(1-g)}{g}\frac{1}{r - r_0 + \frac{g}{\gamma(1-g)}}B.}$$

We see that the number of reconstruction is almost proportional to the inverse of $r - r_0$. As a matter of fact, dividing r_0 by 2 roughly leads to reconstruct the block after twice as many fragment losses. Note that it gives an estimation of the bandwidth need of the system.

We get the average number of blocks lost during a time step:

$$\boxed{\#\text{deads} \approx \rho^{r_0+1}\frac{g}{r - r_0 + \frac{g}{\gamma(1-g)}}B,}$$

with B the total number of blocks in the system and $\rho := \frac{1}{1+\frac{\gamma}{g}(1-g)}$. Note that the number of deads decreases exponentially with the threshold value r_0. Notice also that the base of the exponential ρ depends of the proportion γ/g. As expected, the more the reconstruction ratio is greater than the failure ratio the less deads there are.

Similarly we get the average number of blocks under reconstructions during a time step (it is the sum of the blocks at level $0 \leq i \leq r_0$):

$$\#\text{reconstructions} \approx \frac{\gamma(1-g)}{g} \frac{1}{r - r_0 + \frac{g}{\gamma(1-g)}} B.$$

We see that the number of reconstruction is almost proportional to the inverse of $r - r_0$. As a matter of fact, dividing r_0 by 2 roughly leads to reconstruct the block after twice as many fragment losses. Note that it gives an estimation of the bandwidth need of the system.

4 Study of correlation effects

As always, the MCM for a single entity allows to compute all the expectancies in system of B correlated entities (from the linearity of the expectation). However, this approach usually does not tell anything about the deviation from the mean. This is the case here, because of the very strong correlation that exists between block failures, as, when a disk crashes, tens of thousands of blocks are affected at the same time. Indeed the deviations are far bigger than the ones that would appear in an independent system (which are standard to compute).

We ran extensive experiments: first to check the consistency of the MCM expectation with the simulation and to validate the model simplifications (in Section 4.2); but mainly to observe the deviations from the mean. We show in Section 4.4 the significant impact of correlation on the variations of bandwidth usage, even for a large system. In Section 4.5, we examine a provisioning scenario based on a model assuming block independence. It shows that, when not taken into account, this variation could lead to a very high loss rate (or equivalently a very low data lifetime).

4.1 Simulations

We implemented a custom cycle-based simulator to evaluate the several characteristics of a real system. The simulator does not aim at capturing the low level details of the network (such as packet level communication, traffic congestion or latency), but it focus on the global evolution of the blocks of data in the system in the presence of peer failures and reconstructions.

The simulator uses a detailed view of the system, as it monitors each fragment individually. The number of fragments is $B(s + r)$. As an example, for 5000 peers with a disk of only 5GB, there are already 25 millions of fragments. Hence real life large systems can not be simulated and we need to rely on formal analysis.

Simulation Model (SM). The simulator simulates precisely the evolution of the blocks in the system, that is their state at each time. For each disk, it stores a list of all the blocks having a fragment on it. When a disk fault occurs, the simulator updates the state of the blocks having lost a fragment. Precisely, during each cycle the simulator performs three phases:

1. generate disk failures;

2. handle the reconstruction of critical blocks:

   (a) for each block, test if its reconstruction is over;

   (b) if so, choose randomly $r - r(b)$ peers and add a fragment to their disk;

3. ensure the stability of the system:

   - by reintroducing fragments of the dead blocks in the system;
   - by reintroducing empty disks in replacement of crashed disks.

Initialization Phase. At the beginning each block is at full redundancy and has $n = s + r$ fragments. The fragments of the B blocks are uniformly distributed at random among nodes. Thus, each node starts with an average of $B(s+r)/N$ fragments. Note that during the first phase of the simulation, the system is in a *transient phase*. The cycles corresponding to this phase are not considered when we give results in the following section. We focus on the properties of the stationary phase of the system.

Monitored metrics. The simulator monitors *different metrics* of the system. The main ones are the number of reconstructions (hence the bandwidth usage), the number of dead blocks and the redundancy level of blocks (hence the number of available fragments). But it stores also other metrics like the disk lifetimes and the disk occupancy.

At end of each cycle, a *trace* is generated containing all this information. The *bandwidth consumption BW* is calculated using the number of on-going reconstructions during each cycle. The reconstruction of a block b uses a bandwidth corresponding to the s fragments that have to be gathered by the peer in charge of the reconstruction and to the $r - r(b)$ fragments that have to be redistributed among peers at the end of the reconstruction. In total, that is $(s + r - r(b))S_f$. For the total bandwidth consumption, we sum over all blocks in reconstruction B_{rec} and we divide by the timestep τ and by θ, as the bandwidth is distributed over the whole time of reconstruction. We get

$$BW = \frac{S_f \cdot \sum_{b \in B_{\text{rec}}} (s + r - r(b))}{\tau \cdot \theta}.$$

Average bandwidth usage (in Mbits/s)

r_0	1	3	6	9
MCM	0.75	1.23	2.90	14.74
SM	0.75	1.23	2.91	14.80

Proportion of dead blocks

r_0	1	3	6	9
MCM	$4.98e-03$	$1.23e-06$	$9.55e-12$	$3.0e-16$
SM	$5.1e-03$	0	0	0

θ	1	12	24	36
MCM	0.0009	0.071	0.29	0.60
SM	0.0003	0.073	0.26	0.61

Table 2: Average bandwidth usage and proportion of dead blocks of the SM versus the expectations of the MCM for different values of the parameters.

Simulation suite and default parameters. We did a large number of simulations for different set of parameters. When not otherwise explicitly indicated, the simulations are given for a system with the following default parameters.

We chose a medium sized system with $N = 5000$ peers. However, note that, as stated, we simulated different scenarios with different values of the parameters: we also simulated system with N spanning from 25 to several millions of peers. We use an initial block size of 3.6MB. This initial block is divided into $s = 9$ fragments of size $S_f = 400$KB and $r = 6$ redundancy fragments are added. This leads to a (system) block size of 6MB. The reconstruction threshold value is chose to be $r_0 = 3$. The system-wide number of blocks is $B = 5 \cdot 10^5$ (i.e. 2.86TB), which leads to an average of 600MB per disk. This value was chosen so that the memory of our machines is sufficient to run the simulations in a reasonable time. However, we show in the following that an increased amount of data per disk would only intensify the impact of block correlation. The average time to reconstruct a block is $\theta = 10$ hours. It represents the delay to discover critical blocks (this time is in the order of few hours as we do not want to consider temporary disappearance of a peer for a failure), the time to collect the remaining fragments, to recalculate the erasure code and to redistribute the missing fragments. The average lifetime of a disk or Mean Time Between Failures (MTBF) is assumed to be 5 years (see e.g. [13, 15] for a discussion). In general, the simulation time T_{sim} was chosen to be 50 years, with a time step of one hour, which leads to 438000 cycles.

4.2   Average Behavior of the System

We compare here the results for the simulations of SM with what is predicted from the analysis of the MCM. First, we study the average behavior of the system for different values of the parameters. We present here a representative subset of all our experiments. Table 2-TOP compares the average bandwidth usage of the Simulation Model (SM) with the corresponding expectations derived from the Markov Chain Model (MCM). We varied the threshold value r_0 from 1 to 9 ($r = 10$ is chosen in this experiment) and θ, the mean time to finish a block reconstruction between 1 and 36 hours. We observe that the values are extremely close for both models: the difference only consists in few tens of percents for all the cases. The MCM gives precisely the average bandwidth usage of the system.

The second fundamental statistics of the system is the amount of data lost by the system. Table 2-BOTTOM gives the average proportion of dead blocks among all the blocks. Note that the parameters of the system have to be chosen in such a way that this proportion is extremely low (e.g. in the order of 10^{-15}). Therefore it is non feasible to study such rare events using this simulation technique. Accordingly, we see that for $r_0 \geq 3$ the simulation does not experience any loss (up to 200 years were simulated in this experiment with disk lifetime (MTBF) as low as 1 year). The theoretical proportion given by the MCM is $3 \cdot 10^{-16}$ in the case $r_0 = 9$. To be able to compare the theory and the simulation, we chose less realistic parameters, so that the simulated systems experiences losses during our simulation duration. As an example, we set some long reconstruction time of 36 hours with a very short disk MTBF of 180 days. We observed that both sets of values are very close.

To summarize, we see that the MCM models very closely the average behavior of the system. However, we show in the following that the MCM fails to capture the variations around this average behavior displayed by the simulation results.

4.3   The problem of correlation

One disk failure impacts all the blocks that have fragments stored on the faulty disk. Hence block trajectories are very dependent since the events of our system are disk crashes which can be seen as correlated fragment losses.

An example of system. We simulated a system of medium size with 5000 peers for 50 years. Figure 2 gives the experimental distribution of the bandwidth usage in two cases: the simulation of the real system SM (top plot) and the simulation of a system with independent block faults, modeled by the MCM (bottom plot) and named *Sim. Model Indep.* in the plots. As previously stated the average value of both systems are very close (5.55 versus 5.50 Mbits/s). On the contrary the variations around this average are totally different. The standard deviation is 2.23 Mbits/s in the first case, to compare with only 0.1 Mbits/s in the second one. Hence, we see that the impact of fault correlation is very strong on the behavior of the system.

Note that the standard deviation of the independent model can be deduced directly from the MCM. Each block has a probability $p = \sum_{i=0}^{r_0} P(i)$ of being in reconstruction, with P
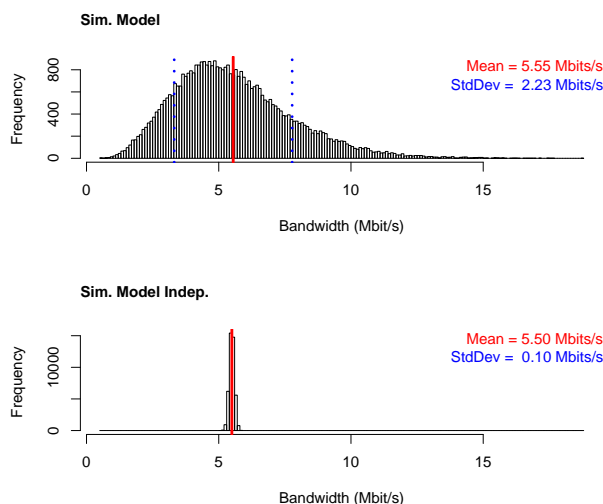
Figure 2: Histogram of the bandwidth used by the system. Top: System with disk failures. Bottom: system with independent block faults.

the stationary distribution of the MCM. Hence, the total number of blocks in reconstruction is the sum of independent variables and follows a binomial distribution of parameters B and p. This distribution is very concentrated around its mean Bp and the standard deviation is given by $\sqrt{Bp(1-p)}$.

4.4 Correlation and the System Size

The impact of fault correlation depends on the system size. A somewhat extreme case is when the number of peers is equal to the number of fragments of a block at full redundancy, that is $N = n$. In such a system all the blocks loose one fragment whenever a disk crashes and all the blocks follow the same trajectory. Almost at the opposite, when the disk contains few fragments (the extreme being each disk contains at most one fragment), trajectories do get independents and the system does not deviate from its mean. These two extreme examples illustrate the fact, that the impact of correlation depends on the ratio between the number of fragments per disk and the number of peers (a peer failure simultaneously impacts about $(s + r)B/N$ fragments). In an extremely large system, the dynamic gets closer to the independent case. The following simulations confirm this intuition.

To illustrate this and show the dependence of correlation and the number of peers in the system, we simulated experiments with the fixed amount of data in the system (same number of blocks), but with varying numbers of peers and disk sizes. The number of blocks is $2.5 \cdot 10^5$. It corresponds to $3.75 \cdot 10^6$ fragments. We varied the number of peers between 25
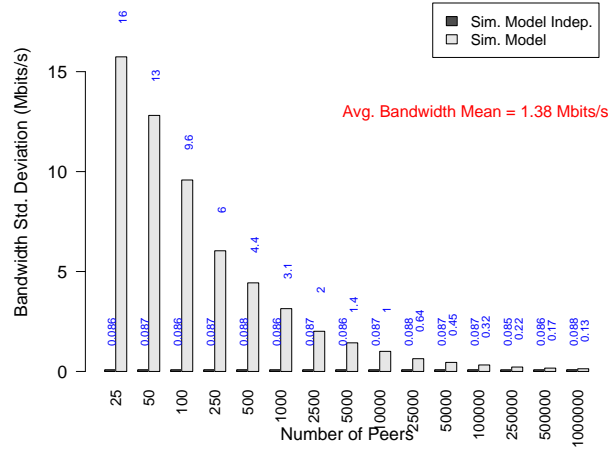
Figure 3: Standard deviation of bandwidth used versus the size of the system.

and 1 million. The standard deviation of the system with dependent and independant block failures are plotted in Figure 3. As expected, the standard deviation of the independent system does not change much, as it does not depend on the system size, but on the number of blocks which is constant here. Two important facts have to be noticed here:

1. The standard deviation of the dependent system is very far from the one of the independent system. This is obvious for small systems: 9.6 vs. 0.09 for 100 peers. But this is true even for large systems: the deviation is still 5 times higher for a system with 50,000 peers.

2. The deviation of the dependent system decreases monotonically with the system size toward the limit obtained for the independent system. In this example, when the number of peers reaches 3 millions, both standard deviation are the same.

   Note that the occupation of the hard disks in this experiment is only 600 MB. For a larger amount of data, the correlation would be even larger, and still strong for system with millions of peers.

4.5   Bandwidth Provisioning and Loss of Data

We showed that the fault correlation has a strong impact on the variations of the bandwidth usage. But will these variations really affect the system? What will happen if the amount of bandwidth available is limited? When a spike occurs in the number of blocks to be rebuilt,
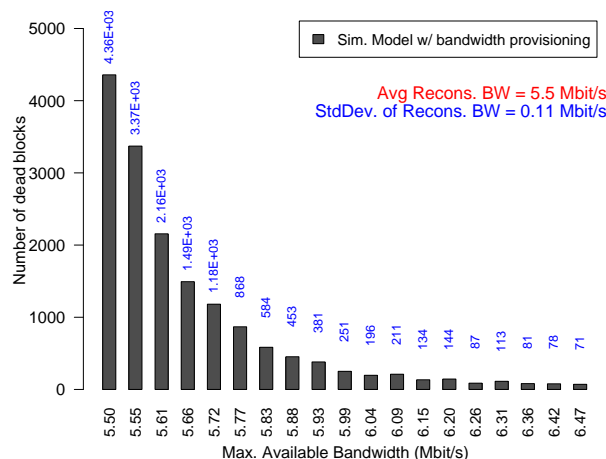
Figure 4: Number of block losses for different provisioning scenarios using the SM.

some reconstructions will be delayed. Will it impact the probability to loose data or will the reconstructions be done when more bandwidth is available without negative effects?

To answer these questions, we simulate different scenarios with bandwidth limitation. This limit varies from μ to $\mu + 10\sigma$, with μ and σ respectively the expectation and the standard deviation given by the MCM. In these experiments, when the bandwidth is not sufficient to carry out all the needed reconstructions, a queue is used to store the blocks to be rebuilt. The reconstructions then start in FIFO order when bandwidth is available.

Figure 4 shows the cumulative number of dead blocks (i.e. total number of blocks whose remaining fragments have fallen below s) for different limits of bandwidth. This system is similar to the one in Figure 2, but here we simulated 400 years. We see that limiting the bandwidth has very strong impact. Between μ (5.5Mbits/s) and $\mu + 5\sigma$ (6.04), the number of dead blocks dropped from $4.3 \cdot 10^3$ in the former to 196 in the later. If we have no limit on the bandwidth, we also computed that the cumulative number of deads drops to 11, which is respectively 400 and 18 times less than the former cases.

Note that for all these experiments, the available bandwidth is greater than the average bandwidth of the MCM. Hence, it only is *the fact of delaying some block reconstructions, that increases the probability to loose fragments.* As a consequence, provisioning the system based on a model assuming block independence, as the MCM, could lead to disastrous effects. As a matter of fact, in the MCM the bandwidth usage is very concentrated around its mean. For example, the probability to exceed $\mu + 5\sigma$ is less than 5.810^{-7}. A provisioning of this amount of bandwidth seems a very safe one. But as we see in Figure 4, such a dimensioning

of the system would lead to large losses of data. Therefore, it is be very important to have a model that takes fault correlation into account.

5 A New Stochastic Model

In this section we address the problem of block correlations. We propose a Fluid Model (FM) that does not represent the behavior of a single block anymore but of the whole system. We give a theoretical analysis in Section 5.2, giving its average behavior, the variation from its mean and a way to compute any of its moments. In Section 5.4, we show by mean of simulations that the FM models very closely the variations of a realistic system.

5.1 The New Model

The key question that araises from the observations made in Section 4 discussion is to provide a way to analyze such a strongly correlated system. Block states could be fully described by a vector encoding the location of its fragments. This leads to a gigantic Markov Chain (with around $N^{(s+R)B}$ states) that indeed represents our simulator SM. However intuition tells us that what matter for a block is its number of fragments not their location. Indeed, at each time step, everything seems to happen as if the fragments were distributed uniformly over the disks at random. So to analyze the system we use this hypothesis (we discuss it in Section 6). This assumption implies that the blocks at the same level are equivalent. Hence a Markov Chain that counts how many blocks are at each level can be used. This discrete chain can be formally described, but it is still too large for practical use (it has $(r+1)^B$ states). However since many blocks are in the same state, we use a fluid approximation for that chain (see [9, 2] for references on fluid models).

We present here this stochastic Fluid Model, with discrete time of time step τ. Generally, the system is completely described by the vector $Y(t)$ of dimension r in which Y_i counts the percentage of blocks that are in state i. Then the evolution of the state vector is modeled as follows. At each time step, either no disk fault occurs and we only account for the effects of the reconstructions or a fault occurs and the block losses are added. The model makes the following assumptions:

- At most one disk gets faulty during a cycle. (Note that it is sufficient to choose τ small enough to ensure that multiple failures almost never happen).

- During a time step, a fault happens with probability $l = \alpha N \tau$. Recall that α is the peer fault rate (see Section 2).

- A fault induces the loss of an exact proportion $1/N$ of the fragments present in the system. This hypothesis equivalently says that there is a uniform repartition of the fragments per disk. If we also assume that the repartition of the blocks into different states is the same for each disk, it gives that, when a fault occurs, $\frac{(s+i)Y_{s+i}}{N}$ fragments are lost. Let us note $\mu_i = \frac{(s+i)}{N}$. For the model, it means that $\mu_i Y_{s+i}$ blocks go from

the state $s + i$ to the state $s + i - 1$. The two hypothesis on the fragment repartition will be later discussed in Section 6 where we propose a refined model.

Let us summarize the new notations that will be used throughout this section:

l	probability to have a disk fault during a time step
μ_i	fraction of blocks in state i affected by a failure
γ	fraction of blocks reconstructed after a time step $(\gamma = 1/\theta)$

The fluid approximation of the system is then:

$$Y_i(t+1) = Y_i(t) \quad for \quad r_0 + 1 \leq i \leq s - 1,$$

$$Y_i(t+1) = (1 - \gamma)Y_i(t) \quad for \quad 1 \leq i \leq r_0,$$

$$Y_s(t+1) = Y_s(t) \sum_{i=1}^{r_0} \gamma Y_i(t),$$

when no fault occurs (reconstruction only). The effects of a fault on a state vector $X(t)$ are:

$$X_i(t+1) = (1 - \mu_i)X_i(t) + \mu_{i+1}X_{i+1}(t) \quad for \quad 1 \leq i \leq s - 1,$$

$$X_s(t+1) = (1 - \mu_s)X_s(t) + \mu_1 X_1(t).$$

When a fault occurs, both the effects of the reconstruction and of the fault act on the state vector. Note that we consider in our model that the fault occurs after the reconstruction. Hence the fluid approximation of the system is:

$$Y_i(t+1) = (1 - \mu_i)Y_i(t) + \mu_{i+1}Y_{i+1}(t) \quad for \quad r_0 + 1 \leq i \leq s - 1,$$

$$Y_{r_0}(t+1) = (1 - \mu_{r_0})(1 - \gamma)Y_{r_0}(t) + \mu_{r_0+1}Y_{r_0+1}(t),$$

$$Y_i(t+1) = (1 - \mu_i)(1 - \gamma)Y_i(t) + \mu_{i+1}(1 - \gamma)Y_{i+1}(t) \quad for \quad 1 \leq i \leq r_0 - 1,$$

$$Y_s(t+1) = (1 - \mu_s)Y_s(t) \sum_{i=1}^{r_0} \gamma Y_i(t) + \mu_1(1 - \gamma)Y_1(t).$$

when a fault occurs.

5.2   Analysis

The analysis of the Fluid Model boils down to the analysis of a random matrix (or matrix distribution), $M(t, \omega)$. Note that it is a difficult problem: there exist no general theory to get the distribution of a random product of two matrices. It is not surprising since, for

example, only determining if the infinite product of two matrices is null is an undecidable problem [12].

A transition of the system transforms the state vector $X = (X_1, \cdots, X_n)$ according to

$$X(t+1, \omega) = M(t, \omega) X(t, \omega).$$

$M(t, \omega)$ depends of two matrices, R and F' respectively representing the effects of a reconstruction and of a disk fault on the state vector. They are defined as:

$$F' = \begin{pmatrix} 1-\mu & & & & \mu \\ \mu & \ddots & & & \\ & \ddots & \ddots & & \\ & & \ddots & \ddots & \\ & & & \mu & 1-\mu \end{pmatrix} \quad R = \begin{pmatrix} 1 & & & \gamma & \cdots & \gamma \\ & \ddots & & & & \\ & & 1 & & & \\ & & & 1-\gamma & & \\ & & & & \ddots & \\ & & & & & 1-\gamma \end{pmatrix}$$

At the time t, the random matrice is equal to PR if there is a failure and to R otherwise. We have

$$M(t, \omega) = \begin{cases} F'R & \text{with probability } l \text{ (disk fault)}; \\ R & \text{with probability } 1-l \text{ (reconstruction only).} \end{cases}$$

In the following, we note $F = F'R$ for simplicity.

Expression of the expectation of the Fluid Model.

$$\mathbb{E}[X(t+1, \omega)] = \mathbb{E}[M(t, \omega)]\mathbb{E}[X(t, \omega)].$$

We clearly have

$$\mathbb{E}[M(t, \omega)] = lF + (1-l)R.$$

The linear operator $\mathbb{E}[M(t, \omega)]$ is a probability matrix and one can check with the computer that it has no eigenvalue with norm one other than 1. Hence we have $\mathbb{E}[X(t, \omega)]$ converges to E_0, solution of the equation

$$E_0 = (lF + (1-l)R)E_0.$$

Note, that we get the same expectation than with the Markov Chain model, indeed $(lF + (1-l)R)$ is roughly equivalent to the MCM transition matrice.

Expression of the standard deviation of the Fluid Model.

We want to compute the standard deviation of the state vector X, meaning the standard deviation of each of its coordinates. We recall that each coordinate corresponds to the number of blocks in a given state.

Let start by computing $\mathbb{E}[X^2]$.

$$X(t+1,\omega)^2 = (M(t,\omega)X(t,\omega))^2.$$

That is

$$X_i^2 = \left(\sum_{j_1=1}^n m_{ij_1} X_{j_1}\right)\left(\sum_{j_2} m_{ij_2} X_{j_2}\right).$$

We get

$$X_i^2 = \sum_{j_1,j_2} m_{ij_1} m_{ij_2} X_{j_1} X_{j_2}.$$

Note that, as X^2 depends of all the cross-products of X_i and X_j, we have to compute all their expectations.

Expression of the expectations of the cross-products.
We have

$$X_i X_j = \left(\sum_{k_1=1}^n m_{ik_1} X_{k_1}\right)\left(\sum_{k_2=1}^n m_{jk_2} X_{k_2}\right).$$

Hence

$$X_i X_j = \sum_{k_1,k_2} m_{ik_1} m_{jk_2} X_{k_1} X_{k_2}.$$

It gives for the expectations:

$$\mathbb{E}[X_i X_j] = \mathbb{E}[\sum_{k_1,k_2} m_{ik_1} m_{jk_2} X_{k_1} X_{k_2}].$$

By linearity and independence (of m_{ij} and X_i), we obtain

$$\mathbb{E}[X_i X_j] = \sum_{k_1,k_2} \mathbb{E}[m_{ik_1} m_{jk_2}]\mathbb{E}[X_{k_1} X_{k_2}].$$

The method is to write a linear system of equations linking the cross-product expectations at time $t+1$ with the expectations at time t. Let ind be the function $[1,n]\times[1,n]\to[1,n^2]$, $\mathrm{ind}(i,j)=(i-1)n+j$. Let us define the matrix N by

$$N_{i'j'} = \mathbb{E}[m_{i,k_1} m_{j,k_2}],$$

with $i'=\mathrm{ind}(i,j)$ and $j'=\mathrm{ind}(k_1,k_2)$. Note that this matrix is of dimensions $n^2\times n^2$. By definition of the matrice of transition $M(t,\omega)$, we have

$$\mathbb{E}[m_{ik_1} m_{jk_2}] = l(F_{ik_1} F_{jk_2}) + (1-l)(R_{ik_1} R_{jk_2}).$$

Hence, we get

$$N_{i'j'} = l(F_{ik_1} F_{jk_2}) + (1-l)(R_{ik_1} R_{jk_2}).$$

Now, if we note Z the vector of the cross-products $(Z_{\text{ind}(i,j)} = X_i X_j)$, we have

$$\mathbb{E}[Z(t+1, \omega)] = N(t, \omega)\mathbb{E}[Z(t, \omega)]$$

Again, as the linear operator $\mathbb{E}[Z(t, \omega)]$ is a probability matrix and that it can be checked that it has no eigenvalue with norm one other than 1, we have $\mathbb{E}[Z(t, \omega)]$ converges to E_0, solution of the equation

$$E_0 = N(t, \omega)E_0.$$

When Z is computed (by a resolution of a linear system with n^2 variables and equations), we can extract the coefficients $\mathbb{E}[X_i^2]$ and compute the standard deviations with

$$\sigma(X_i) = \sqrt{\mathbb{E}[X_i^2] - \mathbb{E}[X_i]^2}.$$

Conclusions for the number of reconstructions and the bandwidth. The number of blocks in reconstruction ξ is equal to the sums of the number of blocks in the states from 0 to r_0. So $\xi = \sum_{i=0}^{r_0} X_i$. We have

$$\mathbb{E}[\xi] = \sum_{i=0}^{r_0} \mathbb{E}[X_i] \qquad \text{and} \qquad \mathbb{V}[\xi] = \sum_{i=0}^{r_0} \sum_{j=0}^{r_0} \text{cov}[X_i X_j].$$

The covariances can be extracted from the previous computations ($\text{cov}[X_i, X_j] = \mathbb{E}[X_i X_j] - \mathbb{E}[X_i]\mathbb{E}[X_j]$).

Each reconstruction lasts in average $1/\gamma$, translated in the model by a probability γ to be reconstructed. Hence the bandwidth, BW, used by the system during on time step is

$$BW = \gamma \xi s b / s.$$

Recall that b is the size of a block and s the number of fragments needed to reconstruct the block. Hence we get the expectation and the variance of the bandwidth by

$$\mathbb{E}[BW] = \gamma b \mathbb{E}[\xi] \qquad \text{and} \qquad \mathbb{V}[BW] = \gamma^2 b^2 \mathbb{V}[\xi].$$

Remark: Any of the moments can be computed with the same method.

5.3 Explicit expressions for the eager policy

Numerical computations for the expectation and the standard deviation of the number of reconstructions are given in Section 5.4. But before, explicit expressions can be deduced for the eager reconstruction policy.

In the eager policy, a block is reconstructed as soon as one of its fragment is lost. Hence the number of reconstructions just is a fraction γ of the number of blocks which lost at least one fragment (meaning all blocks in states $< s + r$). So, if we only want to analyze the
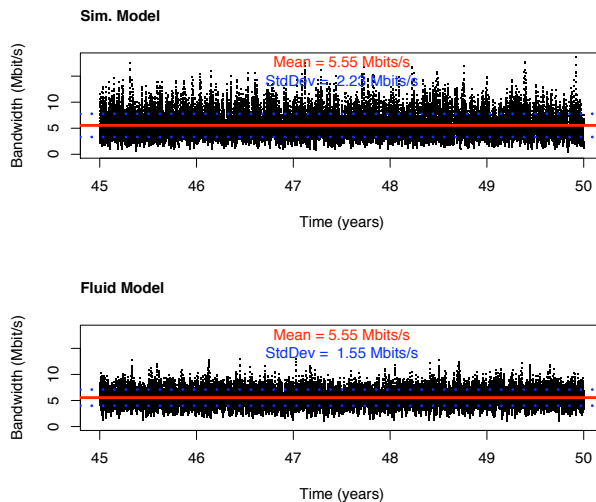
Figure 5: Timeseries of the bandwidth used by SM and FM for 5 years.

use of bandwidth (there are no dead in the system), the model is equivalent to a two state model: the states $s + r$ with the blocks with all their fragments and the state $< s + r$ with the blocks with some fragments missing.

The state vector is $X = (X_{s+r}, X_{<s+r})$, with $X_{<s+r}$ the number of blocks in reconstruction. The intermediate calculuses are omitted here

$$\mathbb{E}[X_{<s+r}] = \frac{B}{1 - (1 - \frac{1}{l\mu})\gamma}.$$

$$\mathbb{E}[X_{<s+r}^2] = \frac{c_1 B^2 + B(c_2 - 2c_1)\mathbb{E}[X_{<s+r}]}{1 - c_1 + c_2 - c_3},$$

with $c_1 = l\mu^2$, $c_2 = 2l\mu(\mu - 1)\gamma + 2l\mu$ and $c_3 = l(\mu\gamma + 1 - \gamma)^2 + (1 - l)(1 - \gamma)^2$.

5.4   Validation of the model

We ran simulations to see how close the Fluid Model (FM) is from the Simulation Model (SM). The behavior of the FM is given by two ways: the FM is simulated for Figure 5 (meaning that at each cycle we randomly choose the transition matrix of the state vector between F and R), but numeric computations from the mathematical analysis are given in Table 5.4.

Figure 5 presents the timeseries of the use of bandwidth. The top plot is for the system simulation and the bottom one for the Fluid Model. As with the MCM, the expectations are

N	25	50	100	250	500	1000	5000	$5 \cdot 10^4$
SM	15.66	11.93	8.82	5.68	2.88	1.30	0.58	0.41
FM	13.38	9.11	6.39	3.99	2.01	0.88	0.39	0.28

r_0	1	3	5	7	9
SM	1.07	1.02	1.00	0.98	0.93
FM	0.67	0.67	0.67	0.67	0.67

$MTBF$	1	2	3	4	5	6	8	10
SM	0.40	0.58	0.69	0.80	0.90	0.97	1.11	1.54
FM	0.28	0.39	0.48	0.55	0.61	0.67	0.77	1.19

Table 3: Standard error (StdDev/Mean) for SM, FM and MCM (MTBF is in years).

pretty close (few tenths of percent). But in addition, the variations have now the same order of magnitude. However the standard errors (the standard error is defined as the standard deviation divided by the mean) still differ by around 20 to 40 percents in most cases, as shown in the summary of simulation with different sets of parameter given in Table 5.4.

It gives the need for a reexamination of the hypothesis of the model. Hence, we propose a Refined Fluid Model in the next section.

6   Impact of disk age: a refined model

6.1   Problem

When a disk fails, it is replaced by a new *empty* disk. This disk fills up with reconstructions, like the other disks in the system. Recall that when a block is reconstructed, each of the fragment rebuilt is sent on a random peer (in a way no two fragments are sent to the same peer). Hence, at each timestep, the partition of the rebuilt fragments between the peers follows a multinomial distribution of parameters N and $1/N$. As the multinomial distribution is very concentrated around its mean, the filling up process can be approximated by a affine process of its age, where, at each timestop, each disk gets in average the number of reconstructed fragments divided by the number of peers.

It is now important to point out that at each moment *disks with very heterogeneous number of fragments* are present in the system. As the process to fill up a disk is close to an affine function of its age, it is equivalent up to a constant to look at disk age or disk filling up. The age of death follows a geometric law of parameter f, as at each timestep a disk has a probability f to experience a fault. That is,

$$\Pr[\text{death age} = k] = (1-f)^{k-1} f.$$

Note that in our simulations of the real system, the disk is introduced at the beginning of the time step but may crash only at the end of a time step. Hence the age of death of a disk
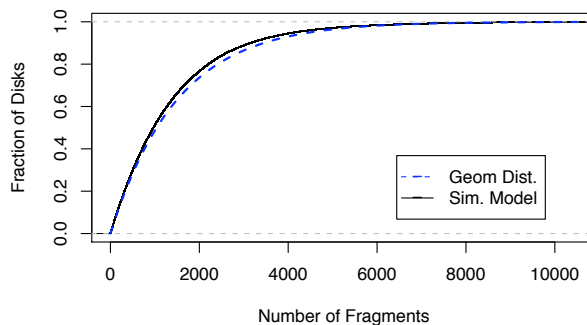
Figure 6: CDF of the number of fragments per disk in the system.

is greater or equal than one in our model. Our formal expression for the distribution of the disk occupancy is tight

Figure 6 shows the cumulative distribution function (CDF) of the number of pieces per disk in the system during a simulation of SM. It is compared with the CDF of the geometric law of parameter f ($f = 1/(24 * 365)$ here) normalized by the average disk occupancy $B * (s + r)/N$. We see that the distribution of disk occupancy is pretty close from what expected from the previous discussion.

6.2 Refinement of the Fluid Model

This strong heterogeneity of the number of fragments per disk may have a deep impact on the variations of the system. As a matter of fact, when the system experiences a disk failure, we may loose a lot of fragments if the disk was almost full, but a lot less for a young disk.

Hence we propose a refinement of the Fluid Model. For each disk failure, we randomly choose the number of fragments present on the disk. More precisely, we add to the model a random variable $\Omega(t, \omega)$, which represents the filling ratio of the disk that crashed. When a disk crashes, we choose $\Omega(t, \omega)$ according to the following distribution:

$$
\begin{aligned}
Pr[\Omega(t, \omega) = \tfrac{k}{f}] &= (1 - f)^{k-1} f \qquad \text{for } 1 \leq k < k_{\max} \\
Pr[\Omega(t, \omega) = \tfrac{k_{\max}}{f}] &= 1 - (1 - f)^{k_{\max}},
\end{aligned}
$$

the principle being:

- to choose the disk age according to the geometric;

- to normalize it to obtain a average filling ratio of 1 ($\mathbb{E}[\Omega(t,\omega)] = 1$). It is enough to multiply the values of Ω by f as $\mathbb{E}[\text{death age}] = 1/f$;

- to truncate the distribution, as a disk does not receive any fragments when it is full. $fk_{\max}$ is equal to the disk maximum size.

Hence, the percentage of blocks concerned by a disk fault is no more μ_i, but $\mu_i \cdot \Omega(t,\omega)$. In the matrician formalism, the matrix F, that represents the impact of a fault on the state vector X becomes a random matrix $F(t,\omega)$, defined as

$$
F(t,\omega) = \begin{pmatrix}
1 - \mu \cdot \Omega(t,\omega) & & & & & \mu \cdot \Omega(t,\omega) \\
\mu \cdot \Omega(t,\omega) & \ddots & & & & \\
& \ddots & \ddots & & & \\
& & \ddots & \ddots & & \\
& & & & \mu \cdot \Omega(t,\omega) & 1 - \mu \cdot \Omega(t,\omega)
\end{pmatrix}
$$

6.3   Analysis of the Refined Fluid Model

The same kind of analysis can be done than in Section 5, albeit the difficulties introduced by the fact that we have now a stochastic failure matrix $F(t,\omega)$. The structure of the proof is the same, but the computation of the moments of the transition matrix, $M(t,\omega)$, differs.

Expectation.
For the computation of the average behavior of the system, only the expectation of the transition matrix changes.

$$
\mathbb{E}[M(t,\omega)] = \mathbb{E}[lF(t,\omega) + (1-l)R] = l\mathbb{E}[F(t,\omega)] + (1-l)R.
$$

$$
\mathbb{E}[F(t,\omega)] = \mathbb{E}[F'(t,\omega)R] = \mathbb{E}[F'(t,\omega)]R,
$$

as F' is independent of R. Hence, we have $\mathbb{E}[F(t,\omega)] = \mathbb{E}[F]$. Therefore, we obtain the same expectation than in the previous Fluid Model.

Standard Deviation.
For the computation of the standard deviation, one have to recompute the matrix N, of the cross-products of elements of $M(t,\omega)$. More precisely, recall that we have

$$
N_{i'j'} = \mathbb{E}[m_{i,k_1} mj, k_2],
$$

with $i' = \mathrm{ind}(i,j)$ and $j' = \mathrm{ind}(k_1, k_2)$. As the matrix F is now stochastic, we have to sum over all possible disk fillings $\Omega(t,\omega)$. We note $F^{(k)}$ the matrix F for a filling ratio equal to k. It gives

$$
N_{i'j'} = \sum_{k=1}^{k_{\max}} \Pr[\Omega(t,\omega) = fk](lF_{ik_1}^{(k)} F_{jk_2}^{(k)}) + (1-l)R_{ik_1}R_{jk_2}.
$$

6.4 Validation of the model

We compare here the behavior of the Refined Fluid Model (RFM) given by theory and the one of the Simulation Model (SM). Figure 7 shows the standard deviation of the bandwidth use in both cases (results for the FM are also given to show the improvements provided by the RFM). We see that the values are very close and differ by only few percents. Note that the average bandwidth use is the about the same in all these experiments and is close to 1.37Mbits/s. We ran extensive sets of simulations to validate the model for different values of the parameters. A summary is given in Table 4. We see that when making the threshold value r_0 vary between 1 and 9 (r is 10 in this experiment) or the average disk lifetime L_D, the standard errors differ from 2 to 4 percents. We can conclude the system is modeled very closely by the RFM.
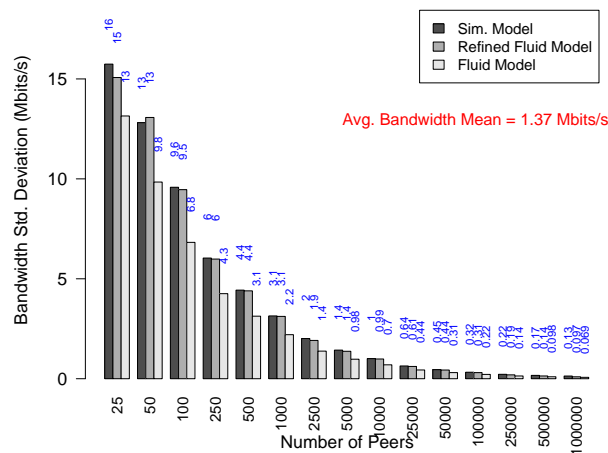

Figure 7: Bandwidth std. deviation vs number of peers for SM, RFM (FM also given for comparison).

6.5 Model Discussions - Future Directions

We showed that the Refined Fluid Modelclosely models the behavior of the real system. In fact, we see that the non uniform repartition of the fragments between the different disks increases the standard deviation of the bandwidth use. To lower the impact of disk age and have more uniform disk fillings, we propose two new policies:

- **Shuffling algorithms.** At each time step, a proportion of the fragments in the system are chosen at random and sent each to a random disk. If all fragments are concerned,

r_0	1	2	3	4	5	7	8	9
SM	1.07	1.05	1.02	1.02	1.00	0.98	0.97	0.93
RFM	0.96	0.96	0.96	0.96	0.96	0.96	0.96	0.95

$MTBF$	1	2	3	4	5	6	8	10
SM	0.40	0.58	0.69	0.80	0.90	0.97	1.11	1.54
RFM	0.38	0.55	0.67	0.78	0.87	0.96	1.09	1.66

Table 4: Standard error for SM and RFM (MTBF is in years).

we obtain an *ideal system* with perfectly uniform repartition of the fragments among the disks. Note that in fact, *the first Fluid Model of Section 5 models almost exactly this ideal system.* The advantages of such policy are that it lowers the differences in number of fragments of the disks, but also decreases the correlation between old blocks that were more present on old disks. However, a drawback is the introduction of more network traffic in the system to transfer the fragments.

- **Biased reconstruction policy.** An other way to have more uniform disk fillings is to change the reconstruction policy. During the last phase of the reconstruction, the rebuilt fragments are sent to random peers. We propose to choose these peers not uniformly, but *to select with higher probability disks with less data.* Do so, the new disks will be filled faster. One drawback of this policy, is that it reinforces the correlation between blocks rebuilt at the same time. But it has the advantage of not changing the bandwidth needs.

We are currently studying the impact of this new policies on the storage system: failure correlation, bandwidth...

7   Conclusion

In this paper, we studied the behavior of a peer-to-peer system storing data blocks. We showed using simulations and formal analysis that modeling such a system by independent blocks, each following its own Markov Chain was very far from reality: If expectations are perfectly captured, deviations from the mean are extremely underestimated. This is due to correlation: in this system a disk crash affects tens of thousand of blocks. We also proved by simulation that these variations (e.g.in bandwidth usage) can have a severe impact on the reliability (probability to loose data).

We then introduced two tractable fluid models capturing most of the system dynamic. Simulations show that these models give very tight results. We believe that the methods proposed in this paper can be applied in other contexts where correlation phenomena occur. This also raises a more theoretical question. The fluid models have a simple dynamic, since it is defined as a random product of two small dimension matrices. Determining the behavior of such a product is known to be untractable, but in our specific case we succeeded to get

exact formulas and compute the moments of the distribution. It would be interesting to find general non trivial conditions (other than commutability) under which the dynamic can be computed.

Acknowledgments

References

[1] S. Alouf, A. Dandoush, and P. Nain. Performance analysis of peer-to-peer storage systems. *Lecture Notes in Computer Science*, 4516:642, 2007.

[2] D. Anick, D. Mitra, and M. Sondhi. Stochastic theory of a data handling system with multiple sources. In *International Conference on Communications (ICC'80)*, volume 1, Seattle, USA, 1980.

[3] R. Bhagwan, K. Tati, Y. chung Cheng, S. Savage, and G. M. Voelker. Total recall: System support for automated availability management. In *Proc. of the conference on Symposium on Networked Systems Design & Implementation (NSDI'04)*, pages 337–350, 2004.

[4] C. Blake and R. Rodrigues. High availability, scalable storage, dynamic peer networks: pick two. In *Proc. of the 9th conference on Hot Topics in Operating Systems (HO-TOS'03)*, 2003.

[5] I. Clarke, O. Sandberg, B. Wiley, and T. W. Hong. Freenet: A distributed anonymous information storage and retrieval system. *Lecture Notes in Computer Science*, 2009:46–66, 2001.

[6] F. Dabek, M. F. Kaashoek, D. Karger, R. Morris, and I. Stoica. Wide-area cooperative storage with CFS. In *Proc. of the 18th ACM SOSP*, pages 202–215, Canada, October 2001.

[7] A. V. Goldberg and P. N. Yianilos. Towards an archival intermemory. In *Proc. Advances in Digital Libraries Conference ADL '98:*, page 147, Washington, DC, USA, 1998.

[8] J. Kubiatowicz, D. Bindel, Y. Chen, S. Czerwinski, P. Eaton, D. Geels, R. Gummadi, S. Rhea, H. Weatherspoon, C. Wells, et al. Oceanstore: an architecture for global-scale persistent storage. *ACM SIGARCH Computer Architecture News*, 28(5):190–201, 2000.

[9] T. Kurtz. *Approximation of Population Processes*. Society for Industrial Mathematics, 1981.

[10] W. Lin, D. Chiu, and Y. Lee. Erasure code replication revisited. In *Proc. of 4th International Conference on Peer-to-Peer Computing (P2P'04)*, pages 90–97, 2004.

[11] M. Luby, M. Mitzenmacher, M. Shokrollahi, D. Spielman, and V. Stemann. Practical loss-resilient codes. In *Proc. of the 29th annual ACM symposium on Theory of computing*, pages 150–159. ACM New York, NY, USA, 1997.

[12] A. Markov. On the problem of representability of matrices. *Z. Math. Logik Grundlagen Math*, 4:157–168, 1958.

[13] E. Pinheiro, W. Weber, and L. Barroso. Failure trends in a large disk drive population. In *Proc. of the conference on File and Storage Technologies (FAST'07)*, 2007.

[14] S. Ramabhadran and J. Pasquale. Analysis of long-running replicated systems. In *Proc. of IEEE Conference on Computer Communications (INFOCOM'06)*, 2006.

[15] B. Schroeder and G. Gibson. Disk failures in the real world: What does an MTTF of 1,000,000 hours mean to you? In *Proc. of the conference on File and Storage Technologies (FAST'07)*, 2007.

[16] G. Utard and A. Vernois. Data durability in peer to peer storage systems. In *Proc. of the 2004 IEEE International Symposium on Cluster Computing and the Grid (CC-GRID'04)*, pages 90–97, USA, 2004.

[17] H. Weatherspoon and J. Kubiatowicz. Erasure coding vs. replication: A quantitative comparison. In *Proc. of International Workshop on Peer-to-Peer Systems (IPTPS'02)*, volume 2, pages 328–338. Springer, 2002.