HAL
open science

# Repairing Multiple Failures with Coordinated and Adaptive Regenerating Codes

Anne-Marie Kermarrec, Nicolas Le Scouarnec, Gilles Straub

## To cite this version:

Anne-Marie Kermarrec, Nicolas Le Scouarnec, Gilles Straub. Repairing Multiple Failures with Coordinated and Adaptive Regenerating Codes. [Research Report] RR-7375, INRIA. 2010. inria-00516647v4

**HAL Id: inria-00516647**

**https://inria.hal.science/inria-00516647v4**

Submitted on 7 Jul 2011

**HAL** is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

# *Repairing Multiple Failures with Coordinated and Adaptive Regenerating Codes*

Anne-Marie Kermarrec — Nicolas Le Scouarnec — Gilles Straub

**N° 7375 — version 4**

initial version 10 September 2010 — revised version 27 June 2011

Distributed Systems and Services

Rapport de recherche

ISSN 0249-6399 ISRN INRIA/RR--7375--FR+ENG

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

INRIA

centre de recherche
RENNES - BRETAGNE ATLANTIQUE

# Repairing Multiple Failures with Coordinated and Adaptive Regenerating Codes

Anne-Marie Kermarrec*, Nicolas Le Scouarnec†, Gilles Straub†

Theme : Distributed Systems and Services
Networks, Systems and Services, Distributed Computing
Équipe-Projet Asap

Rapport de recherche n° 7375 — version 4 — initial version 10 September 2010 — revised version 27 June 2011 — 15 pages

**Abstract:** Erasure correcting codes are widely used to ensure data persistence in distributed storage systems. This paper addresses the repair of such codes in the presence of simultaneous failures. It is crucial to maintain the required redundancy over time to prevent permanent data losses. We go beyond existing work (i.e., regenerating codes by Dimakis *et al.*) and propose *coordinated regenerating codes* allowing devices to coordinate during simultaneous repairs thus reducing the costs further. We provide closed form expressions of the communication costs of our new codes depending on the number of live devices and the number of devices being repaired. We prove that deliberately delaying repairs does not bring additional gains in itself. This means that regenerating codes are optimal as long as each failure can be repaired before a second one occurs. Yet, when multiple failures are detected simultaneously, we prove that our coordinated regenerating codes are optimal and outperform uncoordinated repairs (with respect to communication and storage costs). Finally, we define *adaptive regenerating codes* that self-adapt to the system state and prove they are optimal.

**Key-words:** distributed storage, regenerating codes, multiple failures, self-healing, network coding

* INRIA Rennes - Bretagne Atlantique
† Technicolor

Centre de recherche INRIA Rennes – Bretagne Atlantique
IRISA, Campus universitaire de Beaulieu, 35042 Rennes Cedex
Téléphone : +33 2 99 84 71 00 — Télécopie : +33 2 99 84 71 71

# Réparation de défaillances multiples avec des codes régénérants coordonnés et adaptatifs

**Résumé :** Les codes correcteurs d'effacements sont largement utilisés pour assurer la persistance des données dans les systèmes de stockage distribués. Ce rapport s'intéresse à la réparation de tels codes dans le cas de défaillances simultanées. Cette maintenance est cruciale afin de prévenir les pertes de données permanentes. Nous étendons les travaux existants (codes régénérants par Dimakis *et al.*) et proposons des *codes régénérants coordonnés* qui permettent aux éléments du systèmes de se coordonner durant les réparations de défaillances simultanées afin de réduire les coûts de réparation. Nous fournissons une forme close des coûts de communications de nos codes en fonction du nombre d'équipements vivants et du nombre d'équipements en cours de réparation. Nous prouvons, par ailleurs, que retarder les réparations de façon délibérée n'apporte pas de gains additionnels. Cela signifie que les codes régénérants sont optimaux tant qu'une première défaillance peut être réparée avant une seconde. Cependant, quand de multiples défaillances sont détectés simultanément, nous prouvons que nos codes régénérants coordonnés sont optimaux et dépasse les réparations non coordonnées (vis à vis des coûts de stockage et de réparation). Enfin, nous définissons des codes régénérants adaptatifs qui s'auto-adaptent à l'état du système et prouvons qu'ils sont optimaux.

**Mots-clés :** stockage distribué, codes régénérants, défaillances multiples, auto-réparation, codage réseau

# Repairing Multiple Failures with Coordinated and Adaptive Regenerating Codes

Anne-Marie Kermarrec*, Nicolas Le Scouarnec† and Gilles Straub†
* INRIA Rennes - Bretagne-Atlantique, Rennes, France
† Technicolor, Rennes, France
Anne-Marie.Kermarrec@inria.fr, Nicolas.Le-Scouarnec@technicolor.com, Gilles.Straub@technicolor.com

arXiv:1102.0204v2 [cs.IT] 27 Jun 2011

***Abstract*—Erasure correcting codes are widely used to ensure data persistence in distributed storage systems. This paper addresses the repair of such codes in the presence of simultaneous failures. It is crucial to maintain the required redundancy over time to prevent permanent data losses. We go beyond existing work (i.e., regenerating codes by Dimakis *et al.*) and propose *coordinated regenerating codes* allowing devices to coordinate during simultaneous repairs thus reducing the costs further. We provide closed form expressions of the communication costs of our new codes depending on the number of live devices and the number of devices being repaired. We prove that deliberately delaying repairs does not bring additional gains in itself. This means that regenerating codes are optimal as long as each failure can be repaired before a second one occurs. Yet, when multiple failures are detected simultaneously, we prove that our coordinated regenerating codes are optimal and outperform uncoordinated repairs (with respect to communication and storage costs). Finally, we define *adaptive regenerating codes* that self-adapt to the system state and prove they are optimal.**

***Keywords*—erasure correcting codes, regenerating codes, distributed storage, repair, multiple failures**

## I. Introduction

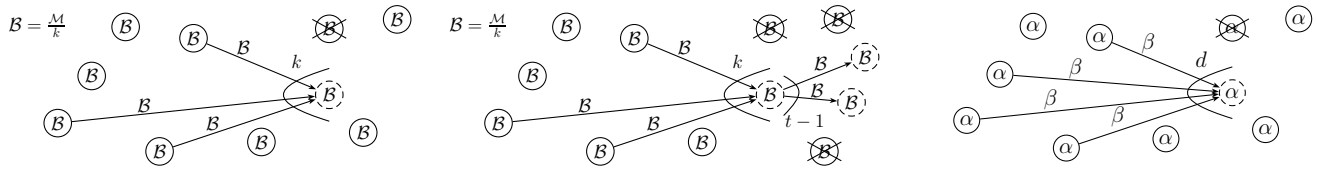
Over the last decade, digital information to be stored, be it scientific data, photos, videos, etc, has grown exponentially. Meanwhile, the widespread access to the Internet has changed behaviors: users now expect reliable storage and seamless access to their data. The combination of these factors dramatically increases the demand for large-scale distributed storage systems. Such systems are used as back-ends by cloud service providers or as a basis for P2P systems to provide users with storage, backup or sharing capabilities. This is traditionally achieved by aggregating numerous physical devices to provide large and resilient storage [2]–[5]. In such systems, which are prone to disk and network failures, redundancy is the natural solution to prevent permanent data losses. However, as failures occur, the level of redundancy decreases, potentially jeopardizing the ability to recover the original data. This requires the storage system to self-repair to go back to its healthy state (i.e., keep redundancy above a minimum level).

Repairing redundancy is of paramount importance for the design and implementation of distributed storage systems. A self-healing mechanism is usually composed of three phases. First, the system must self-monitor to detect failures. Second, the system must trigger a repair on a set of spare devices. Finally, the system must regenerate the lost redundancy from the remaining one. In this paper, we focus on this last phase. Redundancy in storage systems has been extensively implemented using erasure correcting codes [6]–[8] because they enable tolerance to failures at a low storage overhead. In this context, the repair used to induce a large communication overhead, as it required to download and decode the whole file. Yet, Dimakis *et al.* recently showed [9], [10] that the repair cost can be significantly reduced by avoiding decoding in the so-called regenerating codes.

In this paper, we go beyond these works by considering simultaneous repairs in regenerating-like codes. We propose *coordinated regenerating codes* allowing devices to leverage simultaneous repairs: each of the $t$ devices being repaired contacts $d$ live (i.e., non-failed) devices and then coordinates with the $t-1$ others. Our contribution is threefold:

- As deliberately delaying repairs in erasure correcting codes leads to savings [5], [11], [12], it is natural to wonder if the same additional savings can be expected when delaying repairs for regenerating codes. By defining *coordinated regenerating codes*, we prove that, when relying on regenerating-like codes (MSR or MBR) [10], deliberately delaying repairs (so that $t > 1$) cannot lead to further savings.
- Yet in practical systems, it might be difficult to detect every single failure and fix it before a second one occurs (i.e., ensure $t = 1$). We establish the optimal quantities of information to be transferred when $t$ devices must be repaired simultaneously from $d$ live devices. Our *coordinated regenerating codes* consistently outperform existing approaches.
- In addition, most practical systems are highly dynamic. Therefore, assuming that $t$ and $d$ remain constant across repairs is unrealistic. To address this issue, we define *adaptive regenerating codes* achieving optimal repairs according to $t$ and $d$. Under a constant system size $d+t$, their performance does not depend on $t$ (i.e., the cost of

This is an extended version of an article published as a regular paper at NetCod 2011 available on IEEE Xplore (please cite the conference article [1]). It initially appeared (September 2010) as an INRIA Research Report (http://hal.inria.fr/inria-00516647) entitled *Beyond Regenerating Codes*. The following notice apply to the conference article published at NetCod 2011. ©2011 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works.




(a) Erasure correcting codes (immediate repair) (b) Erasure correcting codes (delayed repair) (c) Regenerating codes

Figure 1. Repairing failures with codes. In a $n$ device network, failed devices are replaced by new ones. The new devices get a given amount of data from live devices to repair the redundancy. In our examples, $k = 3$, $d = 4$, $\mathcal{B} = 1$, $\alpha = 1$, and $\beta = 1/2$. Some example of gains for $k = 32$ are given in Table I

a repair does not increase if the number of failed devices to repair increases).

Previous approaches are either known for not supporting simultaneous coordinated repairs [10] or known for assuming that repairing implies decoding which requires to download the whole file [5]–[7], [11], [12]. Hence, we define *coordinated regenerating codes* that fill the gap between the two aforementioned approaches by achieving simultaneous repairs without decoding (Figure 4). Furthermore, as existing regenerating codes assume a static set of parameters, we define *adaptive regenerating codes* that can self-adapt. Two recent works have been interested in these same problems. MCR Codes [13] define MSR-like codes that support multiple repair at once and MFR [14] codes show that MSR codes can be turned into adaptive codes. Yet, MCR Codes only consider the Minimum Storage point and assume that all transfers are equal without proving it (i.e., $\beta = \beta'$); and MFR [14] codes are not optimal when repairing more than one failure. Finally, since, for erasure correcting codes, simultaneous repairs reduce the communication overhead, we study the impact of deliberately delaying repairs with regenerating-like codes.

The paper is organized as follows. Section II describes the background on codes. Section III presents our *coordinated regenerating codes*. In this section, we prove the optimality of our codes, we derive closed-form expressions for specific subsets of codes, and we show that deliberately delaying repairs does not reduce the costs further as long as each failure can be fixed before a second one occurs. In Section IV, we propose *adaptive regenerating codes* that self-adapt to the dynamic of the system and we prove their optimality. Section V briefly reviews some related work. Section VI concludes.

## II. Background

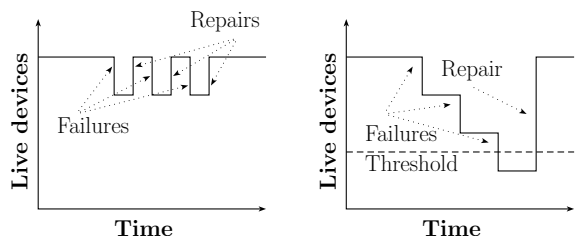
We consider a $n$ device system storing a file of $\mathcal{M}$ bits split in $k$ blocks of size $\mathcal{B} = \mathcal{M}/k$. To cope with device failures, blocks are stored with some redundancy so that a single failure cannot lead to permanent data losses. We consider code-based approaches to redundancy since they have been proven to be more efficient than replication with respect to both storage and repair costs [6]. We focus on self-healing systems (i.e., systems that automatically repair themselves upon failure). Distributed storage systems are required to be self-healing so as to ensure that the system does not gradually lose its ability to recover the initial file. Self-healing systems are equipped with a self-monitoring component that detects failed devices and triggers repairs [15] on new spare devices. To repair, the new spare devices regenerate the lost redundancy from data downloaded from live devices. The repair is constrained by the communication costs [16]. In the rest of this section, we describe the main code-based approaches for generating and repairing redundancy.

### A. Erasure correcting codes (immediate/eager repairs)

Erasure correcting codes have been widely used to provide redundancy in distributed storage systems [6]–[8]. Devices store $n$ encoded blocks of size $\mathcal{B}$, which are computed from the $k$ original blocks. As erasure correcting codes are optimal with respect to recovery (i.e., they allow recovering the $k$ original blocks from any subset of $k$ encoded blocks), storing encoded blocks at $n = k + f$ devices is sufficient to tolerate $f$ failures. This approach is efficient with respect to storage. Yet, repairing a lost encoded block is very expensive since the devices must fully decode the initial file. Hence, repairing a single lost block implies downloading $k$ encoded blocks as shown on Figure 1a.

### B. Erasure correcting codes (delayed/lazy repairs)

A first approach to limit the repair cost of erasure correcting codes is to delay repairs and factor downloading costs [5], [11], [12]. When a device has downloaded $k$ blocks, it can produce as many new encoded blocks as wanted without any additional cost. Therefore, instead of performing a repair upon every single failure (Figure 2a), repairs are deliberately delayed until $t$ (threshold) failures are detected (Figure 2b). This repair strategy is depicted on Figure 1b. One of the new devices downloads $k$ blocks, regenerates $t$ blocks and dispatches them to the $t - 1$ other spare new devices.




(a) Immediate repairs (b) Delayed repairs

Figure 2. Delaying repairs allows performing multiple repairs at once.

### C. Network coding and regenerating codes

A second approach to increase the efficiency of repairs relies on network coding. Network coding differs from erasure correcting codes as it allows devices to generate new blocks with only partial knowledge (i.e., with less than $\mathcal{M}$ bits). Network coding was initially applied to multicast, for which it has been proved that linear codes achieve the maxflow in a communication graph [17]–[19]. Network coding has latter been applied to distributed storage and data persistence [20]–[23]. A key contribution in this area are regenerating codes [9], [10] introduced by Dimakis *et al*. They infer the minimum amount of information to be transferred to repair lost redundancy.

The idea behind regenerating codes [9] is that, when compared to erasure correcting codes, more devices are contacted upon repair but much less data is downloaded from them thus offering low repair cost for each single failure. Similarly to erasure correcting codes, $n$ encoded blocks of size $\alpha$ bits are computed from the $k$ original blocks. Each device stores $\alpha \approx \frac{\mathcal{M}}{k}$ bits. During a repair, the new device contacts $d > k$ other devices to get $\beta \ll \mathcal{M}/k$ bits from each and stores $\alpha$ bits as shown on Figure 1c.

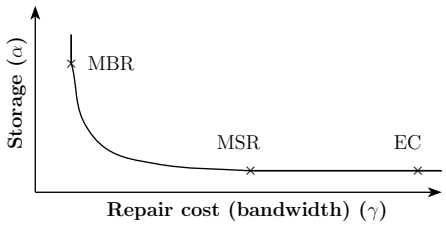



Figure 3. Regenerating codes (MSR or MBR) offer improved performances when compared to erasure correcting codes (EC)

Regenerating codes achieve an optimal trade-off between the storage $\alpha$ and the repair cost $\gamma = d\beta$. The graph on Figure 3 depicts the performance of the optimal regenerating codes. Points $(\alpha, \gamma)$ above the curve correspond to correct (but non-optimal) regenerating codes. Two specific regenerating codes are interesting: MSR (Minimum Storage Regenerating codes) offer optimal repair cost $\gamma = \frac{\mathcal{M}}{k}\frac{d}{d-k+1}$ for a minimum storage cost $\alpha = \frac{\mathcal{M}}{k}$, and MBR (Minimum Bandwidth Regenerating codes) offer optimal storage cost $\alpha = \frac{\mathcal{M}}{k}\frac{2}{2d-k+1}$ for a minimum repair cost $\gamma = \frac{\mathcal{M}}{k}\frac{2d}{2d-k+1}$. Regenerating codes can be implemented using linear codes [18], [19] be they random [24], [25] (i.e., random linear network codes), or deterministic [26]–[30]. Similarly to regenerating codes, our codes can be implemented using random linear network codes.

Table I gives some examples of storage cost $\alpha$ and repair cost $\gamma$ for the codes we describe including the coordinated codes we propose. These costs depend on the number $k$ of devices needed to recover the file, the number $d$ of contacted live devices, and the number $t$ of devices being repaired simultaneously. Regenerating codes by Dimakis *et al.* [10] represent a clear improvement over erasure correcting codes, and our coordinated regenerating codes allow reducing further the costs.

### D. Design rationale

Regenerating codes transfer the minimal quantity of information needed to repair one device storing $\alpha$ bits. Dimakis *et al.* assume fully independent repairs: simultaneous failures are fixed independently. The cost of repairs increase linearly in $t$.

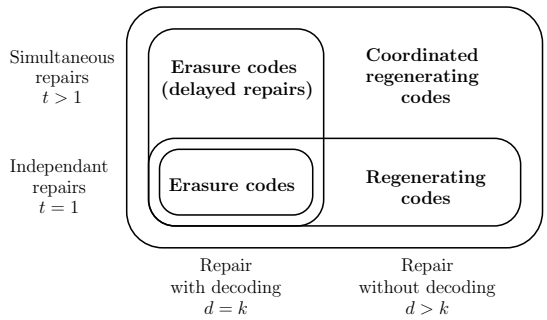



Figure 4. Our coordinated regenerating codes encompass all existing codes. Moreover, they allow the repair of multiple devices at once without decoding.

In this work, we investigate coordinately repairing simultaneous failures in an attempt to reduce the cost, along the lines of delayed repair in erasure correcting codes. Contrary to regenerating codes that repair only using data from live devices, our coordinated regenerating codes also allow the use of data from other devices being repaired. By coordinating the repair, we show that it is possible to repair multiple failures at once with an average cost of $\tilde{\gamma} < \gamma$. As depicted on Figure 4, our new codes encompass both erasure correcting codes ($d = k$) and regenerating codes ($t = 1$). In the next section, we detail our coordinated regenerating codes supporting coordinated repairs.

## III. Coordinated regenerating codes

We consider a situation where $t$ devices fail and, repairs are performed simultaneously. We assume that an underlying monitoring service triggers the repair and contacts all involved devices (i.e., the $t$ spare devices that join the system to replace failed ones). Directly applying erasure correcting codes delayed repairs (Fig. 1b) (i.e., one device repairing for many other devices) to regenerating codes (Fig. 1c) is not appropriate. In short, the fact that all the data goes through the device that regenerates for others induce more network transfers than needed: as repairing does not require decoding, gathering all the information at a single device is not necessary.

Table I
Some examples of repairs of codes for a file of 32 MB

| | $k$ | $d$ | $t$ | $\alpha$ | $\gamma$ |
|---|---|---|---|---|---|
| Erasure codes | 32 | *NA* | *NA* | 1 MB | 32 MB |
| Erasure codes (delayed repair) | 32 | *NA* | 4 | 1 MB | 8.8 MB |
| Dimakis *et al.*'s MSR | 32 | 36 | *NA* | 1 MB | 7.2 MB |
| Dimakis *et al.*'s MBR | 32 | 36 | *NA* | 1.8 MB | 1.8 MB |
| Our MSCR (cf. Sec. III-D2) | 32 | 36 | 4 | 1 MB | 4.9 MB |
| Our MBCR (cf. Sec. III-D1) | 32 | 36 | 4 | 1.7 MB | 1.7 MB |

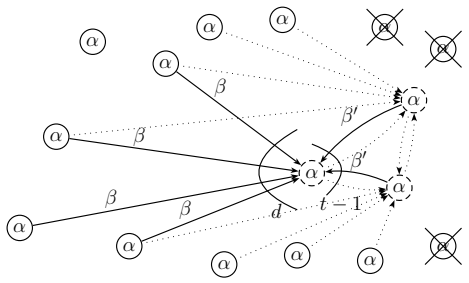



Figure 5. Repairing failures in coordinated regenerating codes. In a network of $n$ devices storing $\alpha$ bits, when $t$ devices have failed, $t$ new devices collect $\beta$ bits from $d$ live devices. They coordinate by exchanging $(t-1)\beta'$ bits with other new devices and store $\alpha$ bits.

### A. Repair algorithm

We introduce new *coordinated regenerating codes* allowing devices to repair simultaneously at the optimal cost (with respect to network communication). The repair is illustrated on Figures 5 and 7 showing the amounts of information transfered, and on Figure 6 showing the computations and exchanges of sub-blocks of data. A device being repaired performs the three following tasks, jointly with all other devices being repaired:

**1. Collect.** It downloads a set of sub-blocks (size $\beta$) from each of the $d$ live devices. The union of the sets is stored as $W_1$.

**2. Coordinate.** It uploads a set of sub-blocks (size $\beta'$) to each of the $t-1$ other devices being repaired. These sets are computed from $W_1$. During this step, sub-blocks received from the $t-1$ other devices being repaired are stored as $W_2$. The data exchanged during this step can be considered as a digest of what each has received during the collecting step.

**3. Store.** It stores a set $W_3$ of sub-blocks (size $\alpha$) computed from $W_1 \cup W_2$. $W_1$ and $W_2$ can be erased upon completion.

Interestingly, thanks to the explicit coordination step involving all devices, this approach evenly balances the load on all devices. Hence, coordinated regenerating codes avoid the bottleneck existing in erasure correcting codes delayed repairs (i.e., the device gathering all the information) (cf. Fig. 1b).

In the rest of this section, we present our main results: we investigate the optimal tradeoffs between storage and repair costs (i.e., optimal values for $\alpha$ (data to store), $\beta$ and $\beta'$ (data to transfer)). As we consider the problem from an information theoretic point of view, we can ignore the nature of the information and only consider the amounts of information that must be exchanged to repair the redundancy. Our results define the fundamental tradeoffs that can be achieved (i.e., lower bounds on amounts of information to be transfered to repair).

Our proof is inspired from the proof by Dimakis *et al.* found in [10]. We represent the system as an information flow graph. For we add a coordination step, our graph differs from the one proposed in [10]. However, Lemmas 2 and 3 are identical to the ones proposed in [10] as they still apply to our information flow graph. When compared to [10], we allow the coordination of multiple repairs while they assume fully independant repairs.

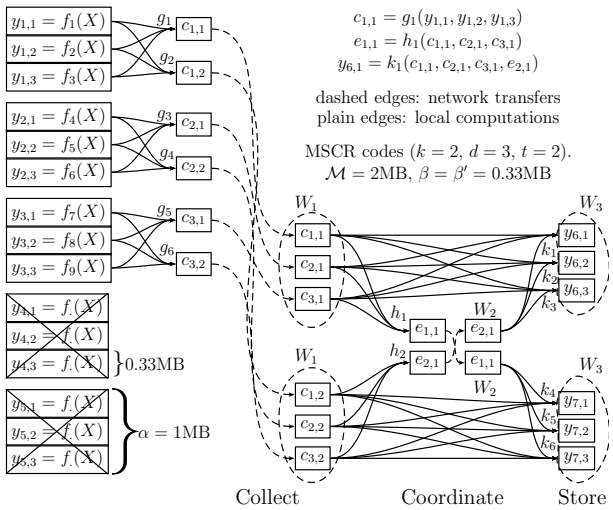



Figure 6. Coordinated regenerating codes based on linear codes. The system stores a file $X$ and is compound of 5 devices. Device $i$ stores 3 sub-blocks $\{y_{i,1}, y_{i,3}, y_{i,3}\}$. Devices 4 and 5 fail and are replaced by devices 6 and 7.

Table II
NOTATION USED IN SECTION III

| | | |
|---|---|---|
| $k$ | Constant (Integer) | Number of devices needed to recover |
| $t$ | Constant (Integer) | Number of devices being repaired |
| $d$ | Constant (Integer) | Number of live devices ($d \geq k$) |
| $\alpha$ | Variable (Real) | Quantity stored |
| $\beta$ | Variable (Real) | Quantity transferred (collect) |
| $\beta'$ | Variable (Real) | Quantity transferred (coordinate) |
| $\gamma$ | Expression (Real) | Quantity transferred over the network |

We determine the optimal codes (i.e., we minimize the storage and the repair cost under some constraints obtained by studying information flow graphs). We give the expressions for optimal values of $\alpha$ (storage at each node), $\gamma$ (repair cost), $\beta$ (data transferred during collecting step) and $\beta'$ (data transferred during coordinating step) as a function of $d$, $k$ and $t$ (parameters of the system). We also examine the influence of deliberately delaying repairs (i.e., increasing $t$ while decreasing $d$). Our notations are summarized in Table II.

### B. Information flow graphs

Our study is based on information flow graphs similar to the ones defined in [10]. An information flow graph is a representation of a distributed storage system that describes how the information about the file stored is communicated through the network. The information flow graph $\mathcal{G}$ is a directed acyclic graph composed of a source S, intermediary nodes $x^{i,j}_{\text{in}}$, $x^{i,j}_{\text{coor}}$ and $x^{i,j}_{\text{out}}$ ($i$ corresponds to a time step while $j$ corresponds to the index of a device introduced at time step $i$), and data collectors $\text{DC}_i$ (data collectors try to contact $k$ devices to decode and recover the file). The capacities of the edges ($\alpha$, $\beta$, $\beta'$) correspond to the amounts of information that can be stored or transferred.

In our approach, a real device $x^{i,j}$ is represented in the graph by 3 nodes ($x^{i,j}_{\text{in}}$, $x^{i,j}_{\text{coor}}$ and $x^{i,j}_{\text{out}}$) corresponding to

its successive states. The graph of a repair of $t$ devices is shown on Figure 7 (assume $t$ divides $k$.). First, devices perform a collecting step represented by edges $x_{\text{out}}^{k,j} \to x_{\text{in}}^{i,j'}$ ($k < i$) of capacity $\beta$ ($d$ such edges). Second, devices undergo a coordinating step represented by edges $x_{\text{in}}^{i,j} \to x_{\text{coor}}^{i,j'}$ of capacity $\beta'$ for $j \neq j'$ ($t-1$ such edges). Devices keep everything they obtained during the first step justifying the infinite capacities of edges $x_{\text{in}}^{i,j} \to x_{\text{coor}}^{i,j}$. Third, they store $\alpha$ using edges $x_{\text{coor}}^{i,j} \to x_{\text{out}}^{i,j}$. Figure 8 gives other examples of information flow graphs.

The graph $\mathcal{G}$ evolves as repairs are performed. When a repair is performed, a set of nodes is added to the graph and the nodes corresponding to failed devices become inactive (i.e., subsequently added intermediary nodes or data collectors cannot be connected to these nodes).

The rest of the article relies on the concept of minimum cuts in information flow graphs. A cut $\mathcal{C}$ between S and $\text{DC}_i$ is a subset of edges of $\mathcal{G}$ such that there is no path from S to $\text{DC}_i$ that does not have at least one edge in $\mathcal{C}$. The minimum cut is the cut that has the smallest sum of edge capacities.

### C. Achievable codes

We define two important properties on codes:

- Correctness: A code $(n, k, d, t, \alpha, \gamma)$ is correct iff, for any succession of repairs, a data collector can recover the file by connecting to any $k$ devices.
- Optimality: A code $(n, k, d, t, \alpha, \gamma)$ is optimal iff it is correct and any code $(n, k, d, t, \bar{\alpha}, \bar{\gamma})$ with $(\bar{\alpha}, \bar{\gamma}) < (\alpha, \gamma)$ is not correct[1].

The following theorem is an important result of our work.

**Theorem 1.** *A coordinated regenerating code $(n, k, d, t, \alpha, \gamma)$ is correct[2] if and only if there exists $\beta$ and $\beta'$ such that the constraints of* (1) *and* (2) *are satisfied. A code minimizing the repair cost $\gamma$* (1)*, along constraints of* (2) *is optimal.*

$$\gamma = d\beta + (t-1)\beta' \tag{1}$$

[1]In this paper, we always consider that $(\bar{a}, \bar{b}) < (a, b)$ means that either $\bar{a} \leq a$ and $\bar{b} < b$, or $\bar{a} < a$ and $\bar{b} \leq b$

[2]We assume that $t$ divides $k$, no result is known if $t$ does not divide $k$.

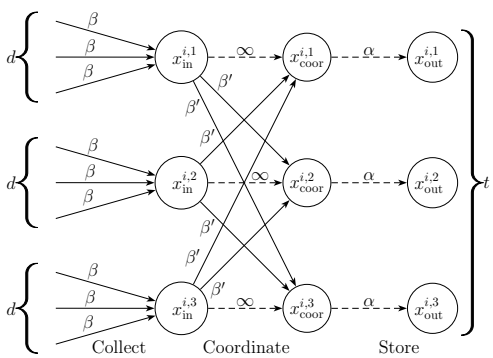



Figure 7. Information flow graph of a repair of $t = 3$ devices. The internal nodes of the graph represent intermediary steps in the repair. First, each device collects $\beta$ from $d$ live devices. Second, devices coordinate by exchanging $\beta'$ with each other. Third, they store $\alpha$. Plain edges correspond to network communication and dashed edges correspond to local communication.

$$\forall \mathbf{u},\ \textit{such that} \sum_{i=0}^{g-1} u_i = k \textit{ and } 1 \leq u_i \leq t,$$
$$\sum_{i=0}^{g-1} u_i \min\{\alpha, (d - \sum_{j=0}^{i-1} u_j)\beta + (t - u_i)\beta'\} \geq \mathcal{M} \tag{2}$$

These constraints, proven in the rest of this subsection, mean that the sum of the amounts of information that can be downloaded from each of the $k$ devices contacted by a data collector must be greater than the file size (amount of information needed to recover the original file). To this end, we consider, for a given coordinated regenerating code $(n, k, d, t, \alpha, \gamma)$, all (infinite) corresponding information flow graphs and evaluate the flow of information that can go from the source to any data collector in such graphs. The vector $\mathbf{u} = (u_i)_{0 \leq i < g}$ represents a possible recovery scenario (i.e., the number $u_i$ of devices contacted in each of the $g$ repair group of size $t$ during the recovery). Since the recovery must be possible for any scenario, we consider all possible $\mathbf{u}$. We show that (2) is satisfied if and only if decoding is possible at any time (i.e., as long as the aforementioned constraints are satisfied, no data is lost).

**Lemma 2.** *For any information flow graph $\mathcal{G}$, no data collector* DC *can recover the initial file if the minimum cut in $\mathcal{G}$ between* S *and* DC *is smaller than the initial file size $\mathcal{M}$.*

*Proof:* Similarly to the proof in [10], since each edge in the information flow graph can be used at most once, and since source to data collector capacity is less than the file size $\mathcal{M}$, the recovery of the file is impossible. ■

**Lemma 3.** *For any finite information flow graph $\mathcal{G}$, if the minimum of the min-cuts separating the source and each data collector is larger than or equal to the file size $\mathcal{M}$, then there exists a linear network code such that all data collectors can recover the file. Furthermore, randomized network coding allows all collectors to recover the file with high probability.*

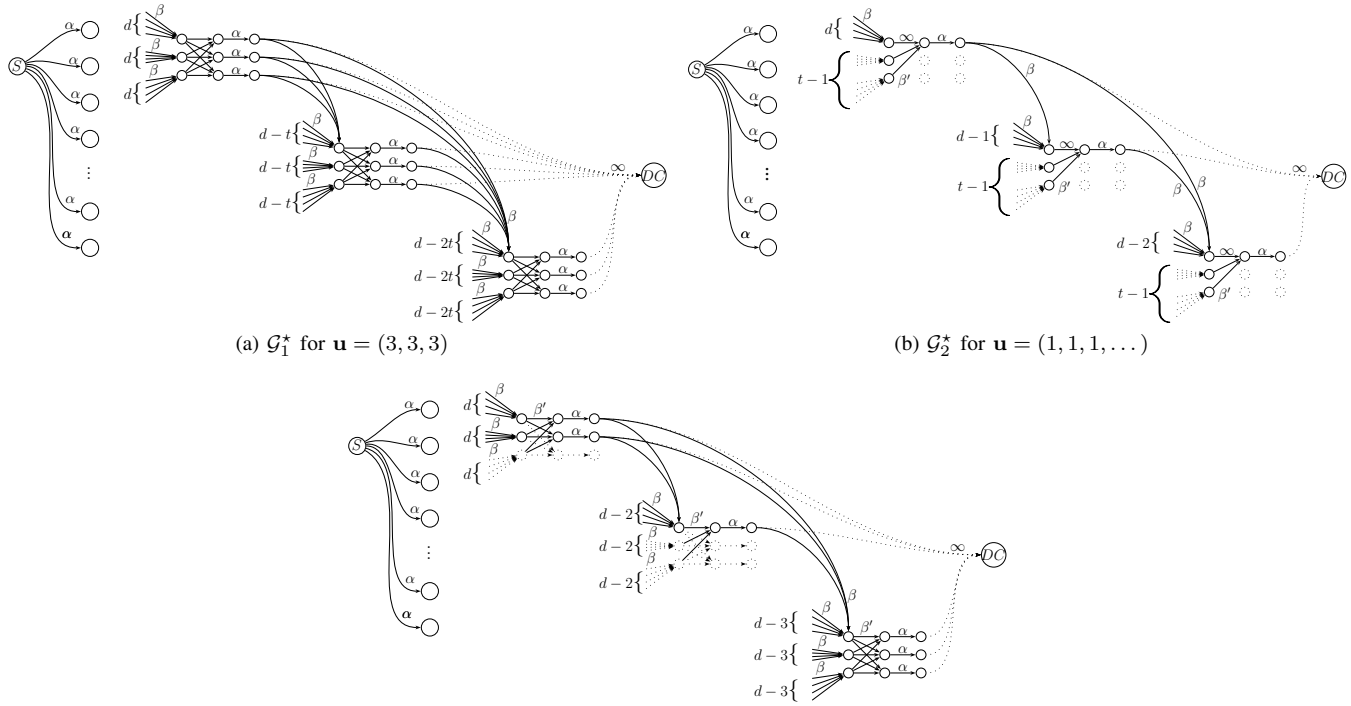
*Proof:* Similarly to the proof in [10], since the reconstruction problem reduces to multicasting on all possible data collectors, the result follows from the results in network coding theory which are briefly discussed in Section II. ■

**Lemma 4.** *For any information flow graph $\mathcal{G}$ compounded of initial devices that obtain $\alpha$ bits directly from the source* S *and of additional devices that join the graph in groups of $t$ devices obtaining $\beta$ from $d$ existing devices and $\beta'$ from each of the other $t-1$ joining devices, any data collector* DC *that connects to a subset of $k$ out-nodes of $\mathcal{G}$ satisfies:*

$$\text{mincut}(S, DC) \geq \min_{\mathbf{u} \in P} \left( \sum_{i=0}^{g-1} u_i \min\{\alpha, (d - \sum_{j=0}^{i-1} u_j)\beta + (t - u_i)\beta'\} \right) \tag{3}$$

*with* $P = \{\mathbf{u} : 1 \leq u_i \leq t \wedge \sum_{i=0}^{g-1} u_i = k\}$.

*Proof:* Let us consider some graph $\mathcal{G}$ (see examples in Figure 8) formed by adding devices according to the repair




(a) $\mathcal{G}_1^\star$ for $\mathbf{u} = (3, 3, 3)$

(b) $\mathcal{G}_2^\star$ for $\mathbf{u} = (1, 1, 1, \dots)$

(c) $\mathcal{G}_3^\star$ for $\mathbf{u} = (2, 1, 3, \dots)$

Figure 8. Information flow graphs for which bounds in (2) are matched with equality for some $\mathbf{u}$.

process described above. Consider a recovery scenario $\mathbf{u} \in P$ in which, a data collector $DC$ connects to a subset of $k$ nodes $\{x_{\text{out}}^{i,j} : (i,j) \in I\}$, where $I$ is the set of contacted devices.

As all incoming edges of DC have infinite capacity, we only examine cuts $(U, \bar{U})$ with $\text{S} \in U$ and $\{x_{\text{out}}^{i,j} : (i,j) \in I\} \subset \bar{U}$. Moreover some additional cases cannot happen since there is an order between $x_{\text{in}}^{i,j}$, $x_{\text{coor}}^{i,j}$ and $x_{\text{out}}^{i,j}$ (*e.g.*, $x_{\text{in}}^{i,j} \in \bar{U}$ and $x_{\text{coor}}^{i,j} \in U$ needs not be considered). Therefore, we only need to examine three cases in the rest of this proof.

Let $\mathcal{C}$ denote the edges in the cut (*i.e*, the set of edges going from $U$ to $\bar{U}$). Every directed acyclic graph has a topological sorting [31], which is an ordering of its vertices such that the existence of an edge $x \to y$ implies $x < y$. In the rest of the analysis, we group nodes that were repaired simultaneously. Since nodes are sorted, nodes considered at one step cannot depend on nodes considered at the following steps.

**First group.** Let $J_0$ be a set of indexes such that $\{x_{\text{out}}^{0,j} : j \in J_0\}$ are the topologically first output nodes in $\bar{U}$ corresponding to a first (same) repair. The set contains $\#\{x_{\text{out}}^{0,j} : j \in J_0\} = u_0$ nodes. Consider a subset $M_0 \subset J_0$ of size $m$ such that $\{x_{\text{in}}^{0,j} : j \in M_0\} \subset U$ and $\{x_{\text{in}}^{0,j} : j \in J_0 - M_0\} \subset \bar{U}$. $m$ can take any value between 0 and $u_0$.

First, consider the $m$ nodes $\{x_{\text{in}}^{0,j} : j \in M_0\}$. For each node, $x_{\text{in}}^{0,j} \in U$. We consider two cases.

- If $x_{\text{coor}}^{0,j} \in U$, then $x_{\text{coor}}^{0,j} \to x_{\text{out}}^{0,j} \in \mathcal{C}$. The contribution to the cut is $\alpha$.
- If $x_{\text{coor}}^{0,j} \in \bar{U}$, then $x_{\text{in}}^{0,j} \to x_{\text{coor}}^{0,j} \in \mathcal{C}$. The contribution to the cut is $\infty$.

Second, consider the $u_0 - m$ other nodes $\{x_{\text{in}}^{0,j} : j \in J_0 - M_0\}$ (third and last case: $x_{\text{in}}^{i,j}$, $x_{\text{coor}}^{i,j}$ and $x_{\text{out}}^{i,j}$ all belong to $\bar{U}$). For each node, the contribution comes from multiple sources.

- The cut contains $d$ edges carrying $\beta$: since $x_{\text{out}}^{0,j}$ are the topologically first output nodes in $\bar{U}$, edges come from output nodes in $U$.
- The cut contains $t - u_0 + m$ edges carrying $\beta'$ thanks to the coordination step. The node $x_{\text{coor}}^{0,j}$ has $t$ incoming edges $x_{\text{in}}^{0,k} \to x_{\text{coor}}^{0,j}$. However, since $\#(\{x_{\text{in}}^{0,k}\} \cap \bar{U}) = u_0 - m$, the cut contains only $t - (u_0 - m)$ such edges.

Therefore, the total contribution of these nodes is

$$c_0(m) \geq m \min(\alpha, \infty) + (u_0 - m)(d\beta + (t - u_0 + m)\beta')$$

Since the function $c_0$ is concave on the interval $[0 : u_0]$, the contribution can be bounded thanks to Jensen's inequality.

$$c_0(m) \geq u_0 \min\{\alpha, d\beta + (t - u_0)\beta'\}$$

**Second group.** Let $\{x_{\text{out}}^{1,j} : j \in J_1\}$ be the topologically second output nodes in $\bar{U}$ corresponding to a second (same) repair. We follow a similar reasoning.

First, consider the $m$ nodes $\{x_{\text{in}}^{1,j} : j \in M_1\} \subset U$. Similarly to the above, the contribution of each node is $\min(\alpha, \infty)$.

Second, consider the $u_0 - m$ nodes $\{x_{\text{in}}^{1,j} : j \in J_1 - M_1\} \subset \bar{U}$. For each node, the contribution comes from multiple sources.

- The cut contains at least $d - u_0$ edges carrying $\beta$: since $x_{\text{out}}^{1,j}$ are the topologically second output nodes in $\bar{U}$, at most $u_0$ edges come from output nodes in $\bar{U}$, and at least

$d-u_0$ other edges come from output nodes in $U$ and are in the cut.
- Similarly to the above, the cut contains $t-u_1+m$ edges carrying $\beta'$ thanks to the coordination step.

Therefore, the total contribution of these nodes is

$$c_1(m) \geq u_1 \min\{\alpha, (d-u_0)\beta + (t-u_1)\beta'\}$$

**$i$-th group.** Following the same reasoning, we find that the $i$-th group of nodes $(i = 0, \ldots, g-1)$ in the sorted set $\bar{U}$ contributes

$$c_i(m) \geq u_i \min\{\alpha, (d - \sum_{j=0}^{i-1} u_j)\beta + (t-u_i)\beta'\}$$

Summing these contributions for all $i$, and considering the worst case for $\mathbf{u} \in P$ leads to (4). ■

*Proof of Theorem 1:* From Lemmas 3 and 4, a code is correct if it satisfies (1) and (2).

$$\gamma = d\beta + (t-1)\beta' \tag{1}$$

$$\forall \mathbf{u}, \text{ such that } \sum_{i=0}^{g-1} u_i = k \text{ and } 1 \leq u_i \leq t,$$
$$\sum_{i=0}^{g-1} u_i \min\{\alpha, (d - \sum_{j=0}^{i-1} u_j)\beta + (t-u_i)\beta'\} \geq \mathcal{M} \tag{2}$$

From Lemma 2, a code is correct only if $\mathrm{mincut}(\mathrm{S}, \mathrm{DC}) \geq \mathcal{M}$. Moreover, for any set of parameter $(n, k, d, t, \alpha, \beta, \beta')$ and any scenario $\mathbf{u}$, we can find a graph $\mathcal{G}_{\mathbf{u}}$ such that

$$\mathrm{mincut}(\mathrm{S}, \mathrm{DC}) = \sum u_i \min\{\alpha, (d - \sum u_j)\beta + (t-u_i)\beta'\} \tag{4}$$

The graph $\mathcal{G}_{\mathbf{u}}$ is built using the process described here after.

- The data collector gets all bits from a set $U$ of $k$ devices.
- The contacted devices repaired simultaneously are grouped in subsets $U_i$ of size $u_i$ such that $U = \bigcup_{i=0}^{g-1} U_i$.
- Each device $x \in U_i$ gets $\beta$ bits from all devices in $\bigcup_{j=0}^{i-1} U_j$, $\beta'$ from $u_i - 1$ devices taking part to the reconstruction, $\beta$ from $d - \sum_{j=0}^{i-1} u_j$ devices not in $U$, $\beta'$ from $t - u_i$ devices not taking part to the reconstruction.

Hence, a code is correct if and only if (1) and (2) are satisfied.

A code minimizing $(\alpha, \gamma)$ under constraints of (1) and (2) is optimal as any code with $(\bar{\alpha}, \bar{\gamma}) < (\alpha, \gamma)$ would not satisfy at least one constraint and hence would not be correct. ■

### D. Optimal tradeoffs

Determining the optimal tradeoffs boils down to minimizing storage cost $\alpha$ and repair cost $\gamma$, as defined by (1), under constraints of (2). $k$, $d$ and $t$ are constants and $\alpha$, $\beta$ and $\beta'$ are parameters to be optimized. In this subsection, we provide optimal tradeoffs $(\alpha, \gamma)$ between storage cost and repair cost (bandwidth).

*1) MBCR codes:* Minimum Bandwidth Coordinated Regenerating Codes correspond to optimal codes that provide the lowest possible repair cost (bandwidth consumption) $\gamma$ while minimizing the storage cost $\alpha$. Figure 10 compares MBCR codes to both Dimakis *et al.* 's MBR [10] and erasure correcting codes with delayed repairs (ECC). Note that similarly to MBR, there is no *expansion* since every bit received is stored (i.e., $\alpha = \gamma$).

$$\alpha = \frac{\mathcal{M}}{k} \frac{2d+t-1}{2d-k+t}$$

$$\beta = \frac{\mathcal{M}}{k} \frac{2}{2d-k+t} \qquad \beta' = \frac{\mathcal{M}}{k} \frac{1}{2d-k+t}$$

We determine the values for MBCR codes in two step. We study two particular cuts and find the minimal values required to ensure that the quantity of information going through the cuts is at least equal to the file size (thus proving the optimality of the solution if it is correct). We then prove that these quantities are enough for all possible cuts.

*Proof of MBCR (Optimality):* Let us consider two particular successions of repairs leading to the graphs shown on Figure 8. The repairs corresponding to such graphs are described in the Proof of Theorem 1. As we want to minimize $\gamma$ before $\alpha$, we assume $\alpha \geq d\beta + (t-1)\beta'$ (i.e., the stored amount is always larger than the downloaded amount).

When $\forall i, u_i = t$ (Fig. 8a), it is required that

$$\sum_{i=0}^{k/t-1} t \left( (d - \sum_{j=0}^{i-1} t)\beta \right) \geq \mathcal{M}$$

which is equivalent to

$$\beta \geq \frac{\mathcal{M}}{k} \frac{2}{2d-k+t}$$

When $\forall i, u_i = 1$ (Fig. 8b), it is required that

$$\sum_{i=0}^{k-1} \left( (d - \sum_{j=0}^{i-1} 1)\beta + (t-1)\beta' \right) \geq \mathcal{M}$$

which is equivalent to

$$\beta' \geq \frac{1}{t-1} \left( \frac{\mathcal{M}}{k} - \beta \frac{2d-k+1}{2} \right)$$

If we consider the smallest possible value $\beta' = \frac{1}{t-1}\left(\frac{\mathcal{M}}{k} - \beta\frac{2d-k+1}{2}\right)$, the associated repair cost is $\gamma = \frac{M}{k} + \frac{k-1}{2}\beta$. The repair cost hence grows linearly with $\beta$. Hence, we should minimize $\beta$. The minimum value for $\beta$ is $\beta = \frac{\mathcal{M}}{k}\frac{2}{2d-k+t}$.

Since lower values would lead to incorrect repair for the graphs depicted on Figure 8, these values are optimal. ■

*Proof of MBCR (Correctness):* We have proved that the aforementioned values are required for two particular cuts. We now prove that such values ensure that enough information flow through every cut and, hence, is correct. According to Theorem 1, the following condition is sufficient for the code

to be correct. We show that the constraint is satisfied when $\alpha$, $\beta$ and $\beta'$ take the values defined for MBCR codes.

$$\sum_{i=0}^{g-1}\left(u_i \min\{(d-\sum_{j=0}^{i-1}u_j)\beta+(t-u_i)\beta',\alpha\}\right)\geq\mathcal{M}$$

since $\alpha$ (the stored part) is always larger than or equal to the transmitted data,

$$\sum_{i=0}^{g-1}u_i\left((d-\sum_{j=0}^{i-1}u_j)\beta+(t-u_i)\beta'\right)\geq\mathcal{M}$$

replacing $\alpha$, $\beta$ and $\beta'$ by their values,

$$\sum_{i=0}^{g-1}u_i\left((d-\sum_{j=0}^{i-1}u_j)2+(t-u_i)\right)\geq k(2d-k+t)$$

which is equivalent to,

$$(2d+t)\sum_{i=0}^{g-1}u_i-2\sum_{i=0}^{g-1}u_i\sum_{j=0}^{i-1}u_j-\sum_{i=0}^{g-1}u_i^2\geq k(2d-k+t)$$

$$(2d+t)\sum_{i=0}^{g-1}u_i-\left(\sum_{i=0}^{g-1}u_i\right)^2\geq k(2d-k+t)$$

as $k=\sum_{i=0}^{g-1}u_i$, it simplifies to

$$(2d+t)k-k^2\geq k(2d-k+t)$$

which is always true. Hence, minimum bandwidth regenerating codes are correct. ■

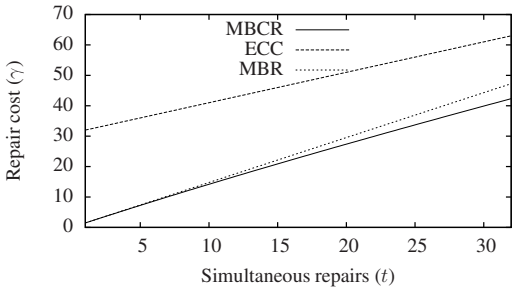



Figure 9. Total repair cost $t\gamma$ for $d=48$ and $k=32$. MBCR codes permanently outperform both erasure correcting codes and regenerating codes

*2) MSCR codes:* Minimum Storage Coordinated Regenerating Codes correspond to optimal codes that provide the lowest possible storage cost $\alpha$ while minimizing the repair cost $\gamma$. This point has been independently characterized by Hu *et al.* in [13]; however, they assume that $\beta=\beta'$ without proving it. We present a simple derivation from Theorem 1 allowing to characterize this point. Figure 10 compares MSCR codes to both Dimakis *et al.*'s MSR [10] and erasure correcting codes with delayed repairs (ECC). Note that for $d=k$, our MSCR codes share the same repair cost as erasure correcting codes delayed repair. Yet, in this case, our codes still have the advantage that they balance the load evenly thus avoiding bottlenecks.

$$\alpha=\frac{\mathcal{M}}{k}\qquad\beta=\frac{\mathcal{M}}{k}\frac{1}{d-k+t}\qquad\beta'=\frac{\mathcal{M}}{k}\frac{1}{d-k+t}$$

*Proof of MSCR (Optimality):* Let us consider two particular successions of repairs leading to the graphs shown on Figure 8. The repairs corresponding to such graphs are described in the Proof of Theorem 1.

We minimize $\alpha$ first. It is clear that $\alpha=\frac{\mathcal{M}}{k}$ is minimal since $\alpha<\frac{\mathcal{M}}{k}$ makes impossible to reconstruct a file of size $\mathcal{M}$ using only $k$ blocks. Hence, what is important is now that each element of the sum is at least equal to $\frac{\mathcal{M}}{k}$.

$$\forall i\in 0\ldots g-1,(d-\sum_{j=0}^{i-1}u_i)\beta+(t-u_i)\beta'\geq\frac{\mathcal{M}}{k}$$

When $\forall i,u_i=t$ (Fig. 8a), it is required that

$$\forall i\in 0\ldots k/t-1,(d-\sum_{j=0}^{i-1}t)\beta\geq\frac{\mathcal{M}}{k}$$

which is equivalent to

$$\beta\geq\frac{\mathcal{M}}{k}\frac{1}{d-k+t}$$

When $\forall i,u_i=1$ (Fig. 8b), it is required that

$$\forall i\in 0\ldots k-1,(d-\sum_{j=0}^{i-1}1)\beta+(t-1)\beta'\geq\frac{\mathcal{M}}{k}$$

which is equivalent to

$$\beta'\geq\frac{1}{(t-1)}(\frac{\mathcal{M}}{k}-\beta(d-k+1))$$

If we consider the smallest possible value $\beta'=\frac{1}{(t-1)}(\frac{\mathcal{M}}{k}-\beta(d-k+1))$, the associated repair cost is $\gamma=\frac{\mathcal{M}}{k}+(k-1)\beta$. The repair cost hence grows linearly with $\beta$. Hence, we should minimize $\beta$. The minimum value for $\beta$ is $\beta=\frac{\mathcal{M}}{k}\frac{1}{d-k+t}$.

Since lower values would lead to incorrect repair for the graphs depicted on Figure 8, these values are optimal. ■

*Proof of MSCR (Correctness):* The proof of correctness is quite similar to the previous one. It consists in proving that

$$\sum_{i=0}^{g-1}u_i\min\{\alpha,(d-\sum_{j=0}^{i-1}u_j)\beta+(t-u_i)\beta'\}\geq\mathcal{M}$$

is always verified when $\alpha$, $\beta$ and $\beta'$ take the aforementioned values.

Since each element of the sum is at most $u_i\frac{\mathcal{M}}{k}$, each element of the sum must satisfy the following constraint.

$$\forall i<g,\min\{\frac{\mathcal{M}}{k},(d-\sum_{j=0}^{i-1}u_j)\beta+(t-u_i)\beta'\}\geq\frac{\mathcal{M}}{k}$$

which simplifies to

$$\forall i<g,\ (d-\sum_{j=0}^{i-1}u_j)\beta+(t-u_i)\beta'\geq\frac{\mathcal{M}}{k}$$

Applying values for MSCR codes,

$$\forall i < g,\ \frac{1}{d-k+t}(d_i - \sum_{j=0}^{i-1} u_j + (t-u_i)) \geq 1$$

$$\forall i < g,\ d - \sum_{j=0}^{i} u_j + t_i \geq d-k+t$$

which is satisfied if

$$\forall i \leq g,\ \sum_{j=0}^{i} u_j \leq k$$

which is true since $\sum_{j=0}^{g-1} u_j = k$ and $u_j > 0$. Therefore, MSCR codes are correct. ■

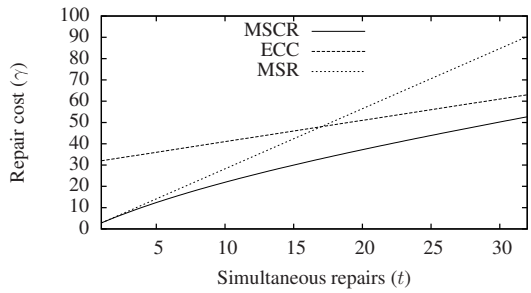



Figure 10. Total repair cost $t\gamma$ for $d = 48$ and $k = 32$. MSCR codes permanently outperform both erasure correcting codes and regenerating codes

*3) General CR codes:* The general case corresponds to all possible trade-offs in between MSCR and MBCR. Valid points $(\alpha, \beta, \beta')$ can be determined by performing a numerical optimization of the objective function. Figure 11 shows the optimal tradeoffs $(\alpha, \gamma)$: coordinated regenerating codes ($t > 1$) can go beyond the optimal tradeoffs for independent repairs ($t = 1$) defined by regenerating codes by Dimakis *et al.* [10].

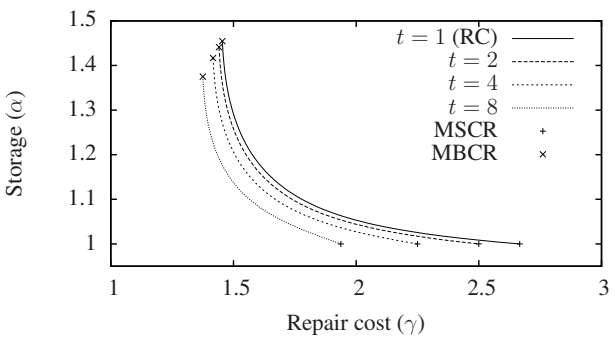



Figure 11. Optimal tradeoffs between storage and repair costs for $k = 16$ and $d = 24$. Regenerating codes (RC) [10] are depicted as $t = 1$. For each $t$, both MSCR and MBCR are shown. Costs are normalized by $\mathcal{M}/k$.

### *E. Optimal threshold*

As previously explained, in regenerating codes, the higher the number of devices being contacted $d$, the higher the savings on the repair cost $\gamma$. Moreover, when repairs are delayed, higher values for the number of devices being repaired $t$ lead to higher savings on the repair cost $\gamma$. If we consider a system of constant size $n = d+t$, these two objectives are contradictory: the longer the delay, the lower the number of live devices $d$. An interesting question is what is the optimal threshold $t$ for triggering repairs assuming that $d + t$ is constant (i.e., is it useful to deliberately delay repairs?). This question is addressed hereafter by studying how MBCR codes and MSCR codes behave as $t$ changes in a system of constant size.

**Theorem 5.** *If we consider a system of size $n = d + t$, for MBCR codes, the optimal value is $t = 1$ while for MSCR codes any value $t \in \{1 \ldots n-k\}$ is optimal.*

*Proof:* Let us consider the repair cost assuming that $n = d + t$ is constant. For MBCR codes, the cost $\gamma = \frac{\mathcal{M}}{k}\frac{2n-t-1}{2n-k-t}$ increases when $t$ increases. The optimal value of $t$ for MBCR codes is the lowest possible value (i.e., $t = 1$). For MSCR codes, the cost $\gamma = \frac{\mathcal{M}}{k}\frac{n-1}{n-k}$ does not depend on $t$. The repair cost of MSCR remains constant, and $t$ can be set to any value as there is no optimum. Neither MSCR nor MBCR allow additional gains when deliberately delaying repairs (i.e., deliberately setting $t > 1$). ■

**Corollary 6.** *If we consider a system of size $n = d+t$ where $t$ can be freely chosen (i.e., the value of $t$ is not constrained by the system) both MSR and MBR regenerating codes [10] are optimal. Hence deliberately delaying repairs to force high values for $t$ does not bring additional savings.*

However, if several failures are detected simultaneously, coordinated regenerating codes remain more efficient as they leverage simultaneous failures by coordinating during repairs.

## IV. Adaptive Regenerating Codes

In the previous section, we presented coordinated regenerating codes that assume $t$ and $d$ to remain constant across repairs. This is similar to regenerating codes [10], in which $d$ remains constant across repairs. Yet, in real systems, it is not realistic to assume that the failure rate remains constant over time, and to assume that every single failure can be repaired before a second one occurs.

In the particular case of Minimum Storage ($\alpha = \frac{\mathcal{M}}{k}$), such strong assumptions are not needed. Indeed, when minimizing the sum in (2), we can minimize the different elements of the sum, which correspond to repairs, independently. Therefore, repairs are independent. We propose to adapt the quantities to transfer $\beta$ and $\beta'$ to the system state which is defined by the number $t$ of devices being repaired and the number $d$ of live devices.

Adaptive regenerating codes simplify the design of a system based on regenerating codes. Indeed, when designing such a system, at least two parameters must be fixed: the number $k$ of devices needed to recover the file and the number $d$ of devices needed to repair a file. The higher the $d$ is, the lower the repair cost is. Hence, $d$ should be chosen as high as possible. Yet, if less than $d$ devices are alive, regenerating

codes cannot perform the repair, and the only proven but costly way to repair is to decode (gathering the whole file). Therefore, $d$ should be fixed to reasonably low values (sub-optimal) so that there are always at least $d$ live devices. On the contrary, adaptive regenerating codes self-adapt by choosing, at each repair, the highest possible $d$ (the lowest repair cost $\gamma$ is achieved for the highest $d$). Hence, designing a system using adaptive regenerating codes implies setting only the right value for the parameter $k$ thus leading to simpler designs.

### A. Our approach

**Theorem 7.** *Adaptive regenerating codes $(k, \Gamma)$ are both correct and optimal. $\Gamma$ is a function $(t, d) \to (\beta_{t,d}, \beta'_{t,d})$ that maps a particular repair setting to the amounts of information to be transferred during a repair.*

$$\beta_{t,d} = \frac{\mathcal{M}}{k}\frac{1}{d-k+t} \qquad \beta'_{t,d} = \frac{\mathcal{M}}{k}\frac{1}{d-k+t} \tag{5}$$

In this subsection, we prove they are correct and optimal.

**Lemma 8.** *For any information flow graph $\mathcal{G}$ compounded of initial devices that obtain $\alpha$ bits directly from the source* S *and of additional devices that join the graph in groups of $t_i$ devices obtaining $\beta_{t_i,d_i}$ from $d_i$ existing devices and $\beta'_{t_i,d_i}$ from each of the other $t_i - 1$ joining devices, any data collector* DC *that connects to a subset of $k$ out-nodes of $\mathcal{G}$ satisfies:*

$$\text{mincut}(S, DC) \geq \tag{6}$$
$$\min_{\mathbf{u}\in P}\left(\sum_{i=0}^{g-1} u_i \min\{\alpha, (d_i - \sum_{j=0}^{i-1} u_j)\beta_{t_i,d_i} + (t_i - u_i)\beta'_{t_i,d_i}\}\right)$$

*with $P = \{\mathbf{u} : 1 \leq u_i \leq t_i \wedge \sum_{i=0}^{g-1} u_i = k\}$.*

*Proof:* The proof is similar to the proof of Lemma 4. ■

*Proof of Theorem 7 (Correctness):* Using Lemmas 2, 3 and 8, we can define the following sufficient condition for the code to be correct. The condition is satisfied when $\beta$ and $\beta'$ take the values defined in (5).

$$\forall \mathbf{u}, \text{ such that } \sum_{i=0}^{g-1} u_i = k \text{ and } 1 \leq u_i \leq t,$$
$$\sum_{i=0}^{g-1} u_i \min\{\frac{\mathcal{M}}{k}, (d_i - \sum_{j=0}^{i-1} u_j)\beta_{t_i,d_i} + (t_i - u_i)\beta'_{t_i,d_i}\} \geq \mathcal{M}$$

The condition must be satisfied for every $\mathbf{u}$. For any $\mathbf{u}$, since each element of the sum is at most $u_i\frac{\mathcal{M}}{k}$, each element of the sum must satisfy the following constraint.

$$\forall i < g, \min\{\frac{\mathcal{M}}{k}, (d_i - \sum_{j=0}^{i-1} u_j)\beta_{t_i,d_i} + (t_i - u_i)\beta'_{t_i,d_i}\} \geq \frac{\mathcal{M}}{k}$$

which simplifies to

$$\forall i < g, \ (d_i - \sum_{j=0}^{i-1} u_j)\beta_{t_i,d_i} + (t_i - u_i)\beta'_{t_i,d_i} \geq \frac{\mathcal{M}}{k}$$

Applying formulas of (5),

$$\forall i < g, \ \frac{1}{d_i - k + t_i}(d_i - \sum_{j=0}^{i-1} u_j + (t_i - u_i)) \geq 1$$

$$\forall i < g, \ d_i - \sum_{j=0}^{i} u_j + t_i \geq d_i - k + t_i$$

which is satisfied if

$$\forall i \leq g-1, \ \sum_{j=0}^{i} u_j \leq k$$

which is true since $\sum_{j=0}^{g-1} u_j = k$ and $u_j > 0$. Therefore, adaptive regenerating codes are correct. ■

*Proof of Theorem 7 (Optimality):* We prove by contradiction that the adaptive regenerating codes are optimal. Let us assume that there exists a correct code $(k, \bar{\Gamma})$ such that $\bar{\Gamma} < \Gamma$ (i.e., for some $(t, d)$, $\bar{\Gamma}(t, d) < \Gamma(t, d)$). This is equivalent to $(k, \Gamma)$ not being optimal.

Consider a set of failures such that all repairs are performed by groups of $t$ devices downloading data from $d$ devices. Consider the corresponding information flow graph. Assuming repairs are performed with a correct code $(k, \bar{\Gamma})$, the information flow graph also corresponds to a correct code $(d + t, k, t, d, \alpha, \bar{\beta}_{t,d}, \bar{\beta}'_{t,d})$.

Moreover, according to the previous section, these failures can be repaired optimally using the MSCR code $(d + t, k, t, d, \alpha, \beta_{t,d}, \beta'_{t,d})$. Therefore, there is a contradiction since the code $(d + t, k, t, d, \alpha, \bar{\beta}_{t,d}, \bar{\beta}'_{t,d})$ cannot be correct if the code $(d + t, k, t, d, \alpha, \beta_{t,d}, \beta'_{t,d})$ is optimal. A correct code $(k, \bar{\Gamma})$ cannot exist, and the adaptive regenerating code $(k, \Gamma)$ defined in this section is optimal. ■

Building on results from coordinated regenerating codes (especially MSCR), we have defined adaptive regenerating codes and proved that they are both correct and optimal. These codes are of particular interest for dynamic systems where failures may occur randomly and simultaneously.

### B. Performance

We compare our approach to MFR codes defined in [14]. This approach is built upon MSR codes defined by Dimakis *et al.* in [10]. The coding scheme can be described as $(k, \Gamma')$ where $\Gamma'$ is a function $d \to \beta_d$. The $t$ repairs needed are performed independently.

$$\beta_d = \frac{\mathcal{M}}{k}\frac{1}{d-k+1} \tag{7}$$

Let us consider the particular case where $d + t = n$. The average cost per repair of our codes remains constant $\gamma = \frac{\mathcal{M}}{k}\frac{n-1}{n-k}$. In the MFR approach, which requires repairs to be performed independently, the average repair cost $\gamma' = \frac{\mathcal{M}}{k}\frac{n-t}{n-t-k+1}$ increases with $t$. Therefore, the performance of our adaptive regenerating codes does not degrade as the number of failures increases, as opposed to the MFR constructed upon Dimakis *et al.* 's codes. This is also shown on Figure 12

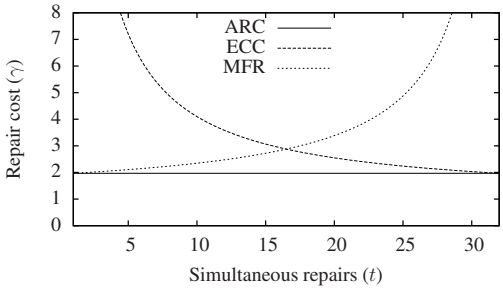



Figure 12. Average repair cost $\gamma$ for $n = 64$ and $k = 32$. Adaptive Regenerating Codes (ARC) permanently outperform both erasure correcting codes (ECC) and the MFR codes.

### C. Implementation

Our approach also has significant advantages over the MFR approach with respect to the implementation. The implementations are similar in principle to the one described in Subsection III-A and Figure 6. The only difference is that the values $d$ and $t$ may differ from one repair to the other. Each device stores sub-blocks of data and combines them to send the appropriate quantities of information. To be able to send $\beta = \frac{2}{3}\frac{\mathcal{M}}{k}$, each device must store $z = 3$ sub-blocks. To be able to send $\beta = \frac{1}{3}\frac{\mathcal{M}}{k}$ or $\beta = \frac{1}{4}\frac{\mathcal{M}}{k}$, each device must store $z = \operatorname{lcm}\{3, 4\} = 12$ sub-blocks. Hence, the length of any random linear code used to implement such a system is $l = zk$ where $z$ is the number of sub-blocks stored by each device. We now consider a system of constant size $n = d + t$ and compare both implementations.

The implementation of the MFR approach implies that to support $d \in \{k \dots n-1\}$, each device must be able to send all quantities $\beta \in \{\frac{1}{1}\frac{\mathcal{M}}{k} \dots \frac{1}{n-k}\frac{\mathcal{M}}{k}\}$. Hence, $z = \operatorname{lcm}\{1 \dots n-k\}$. It can be shown that $2^{n-k} \leq z \leq 3^{n-k}$. Hence, the length of the codes required to implement such codes grows exponentially with $n - k$.

The implementation of our approach implies that to support $d \in \{k \dots n-1\}$, each device must be able to send quantities $\beta = \frac{1}{n-k}\frac{\mathcal{M}}{k}$ and $\beta' = \frac{1}{n-k}\frac{\mathcal{M}}{k}$. Hence, $z = n - k$. Hence, the length of the codes required to implement such codes grows linearly with $n - k$.

Even though simple adaptive regenerating codes (MFR) can be built from Dimakis *et al.*'s MSR codes, our adaptive regenerating codes have two clear advantages. First, their repair cost is always lower than any other approach (i.e., optimal) and is constant on systems of constant size. Second, an optimal implementation of our adaptive regenerating codes requires much shorter codes (i.e. smaller $l$). This is very important since $l$ has a direct impact on the complexity of all operations (encoding, recoding, and decoding).

## V. Related Work

As already explained, our coordinated and adaptive regenerating codes, which allow optimal simultaneous repairs, outperform both regenerating codes [10] and erasure correcting codes with lazy repairs [5], [11], [12]. Another approach aiming at reducing the repair costs consists in contacting less devices while still downloading the same amount from each device [32], [33]. Such approach requires structuring the code so that encoded blocks can be recovered without decoding by combining a few other encoded blocks. Even though this approach offers a reasonably low repair cost, it is neither optimal in storage nor optimal in repair cost. Moreover, these studies provide storage and repair costs that apply only to the specific codes proposed.

Regenerating codes [10] and coordinated regenerating codes presented in this paper assume a symmetric role for all devices (i.e., they all transfer the same amounts of information). Since network connections between every device may not be equivalent, it is interesting to adapt the repair strategy to take into account the underlying network topology. A first study [34] has focused on structuring the repair as a tree instead of a star. Indeed, with regenerating codes, repairs are performed by having the repaired device download directly from live devices. In [34], a first live device can receive information from a second live device so as to forward this information to the repaired device. This is of interest in systems where devices are connected in a mesh network and cannot contact all other devices directly. In such case, this can avoid a potential bottleneck link between a live device and the repaired device. However, in our study, we assumed the most frequent case where all devices are connected to a regular network (*e.g.*, a peer-to-peer system) allowing direct connection from one device to all other devices. Another study [29] has focused on downloading unequal amounts of information from other devices during repairs. They define the total amount of information that must be downloaded depending on the maximum amount of information that can be downloaded from each device. They show that the lowest repair cost is offered when all devices download the same amount of data (i.e., regular regenerating codes). This last study can also be applied to our codes and would show that allowing unequal downloads (i.e., non symmetric system) would also increase the global repair cost.

Independently from our result, the work [13] addresses a subset of the problem we consider. They notice that regenerating codes can only repair single failures and come up with a solution that can handle multiple failures. They naturally define a similar repair method (i.e., they add a coordination step to the information flow graph). Yet, their solution is much more limited than ours as they only study the Minimum Storage case (MSR). Not only, we also study the Minimum Bandwidth (MBR) point, but this cannot be covered by their model since they assume all transfers are equal (i.e., $\beta = \beta'$). Finally, we also determine numerically the general case (i.e., points between Minimum Storage (MSR) and Minimum Bandwidth (MBR) points). Their paper is also restrictive with respect to system they consider as, they assume a system of constant size where all devices are involved (i.e., $n = d + t$). Finally, we do build upon our result to define an adaptive form of regenerating codes that is more flexible

to use in practical systems while they do not consider such constructions. Hence, the previously published paper [13], which is yet another proof of the importance of the considered problem, covers only a subset of our results even if it shares both the problem and some tools used (an adaptation of Information Flow Graphs from Dimakis et *al.* [9], [10]) .

## VI. Conclusion

In this paper, we have proposed novel and optimal *coordinated regenerating codes* by considering simultaneous repairs in regenerating codes. This work has both theoretical and practical impacts. From a theoretical standpoint, we proved that deliberately delaying repairs cannot provide further gain in term of communication costs. In practical systems, however, where several failures are detected simultaneously, our coordinated regenerating codes outperform regenerating codes [10] and are optimal. We also proposed *adaptive regenerating codes* that allow adapting the repair strategy to the current state of the system so that it always performs repairs optimally. For both codes, we provided and proved the minimum amounts of information that need to be transfered during the repair thus defining the fundamental tradeoffs between storage and repair costs for coordinately repairing multiple failures. Finally, our *coordinated regenerating codes* can also be viewed as a global class of codes that encompass erasure correcting codes with delayed repairs ($d = k$), regenerating codes ($t = 1$), erasure correcting codes ($t = 1$ and $d = k$), and new codes ($t > 1$ and $d > k$) that combine, previously incompatible, existing approaches of regenerating codes and delayed repairs (cf. Figure 4).

So far, our study has focused on the theoretical framework for performing multiple coordinated repairs in distributed storage. A straightforward implementation of such codes relies on random linear network codes (cf. Figure 6). Coordinated regenerating codes can therefore be used as they are for self-healing distributed storage. Yet, the seminal paper on regenerating codes [9] has been followed by studies of more constrained repairs (deterministic repairs known as exact repairs by opposition with functional repairs studied in this paper). In short, in functional repairs, the regenerated redundancy is not necessarily the same as the lost one while, in exact repairs, the regenerated redundancy is exactly the same as the lost one. A recent survey [26] details the various works on exact repairs. Since our coordinated regenerating codes only consider the problem of functional repair (i.e., they leave open the problem of coordinated exact repair), an interesting extension of this work would be to study deterministic coordinated regenerating codes performing exact repair. Indeed, they would have appealing properties (simpler integrity checking, lower decoding complexity) when compared to coordinated regenerating codes based upon random linear network codes.

## References


[1] A. Kermarrec, N. Le Scouarnec, and G. Straub, "Repairing Multiple Failures with Coordinated and Adaptive Regenerating Codes," in *NetCod'2011: The 2011 International Symposium on Network Coding*, July 2011.

[2] F. Dabek, F. Kaashoek, D. Karger, R. Morris, and I. Stoica, "Wide-area Cooperative Storage with CFS," in *SOSP*, 2001.

[3] S. Rhea, P. Eaton, D. Geels, H. Weatherspoon, B. Zhao, and J. Kubiatowicz, "Pond: the OceanStore Prototype," in *FAST*, 2003.

[4] S. Ghemawat, H. Gobioff, and S.-T. Leung, "The Google File System," in *SOSP*, 2003.

[5] R. Bhagwan, K. Tati, Y.-C. Cheng, S. Savage, and G. M. Voelker, "Total Recall: System Support for Automated Availability Management," in *NSDI*, 2004.

[6] H. Weatherspoon and J. Kubiatowicz, "Erasure Coding Vs. Replication: A Quantitative Comparison," in *IPTPS*, 2002.

[7] W. K. Lin, D. M. Chiu, and Y. B. Lee, "Erasure Code Replication Revisited," in *P2P*, 2004.

[8] R. Rodrigues and B. Liskov, "High Availability in DHTs: Erasure Coding vs. Replication," in *IPTPS*, 2005.

[9] A. G. Dimakis, P. B. Godfrey, Y. Wu, M. O. Wainwright, and K. Ramchandran, "Network Coding for Distributed Storage Systems," in *INFOCOM*, 2007.

[10] ——, "Network Coding for Distributed Storage Systems," *IEEE Transactions On Information Theory*, vol. 56, pp. 4539–4551, 2010.

[11] A. Datta and K. Aberer, "Internet-scale storage systems under churn – A Study of steady-state using Markov models," in *P2P*, 2006.

[12] O. Dalle, F. Giroire, J. Monteiro, and S. Pérennes, "Analysis of Failure Correlation Impact on Peer-to-Peer Storage Systems," in *P2P*, 2009.

[13] Y. Hu, Y. Xu, X. Wang, C. Zhan, and P. Li, "Cooperative Recovery of Distributed Storage Systems from Multiple Losses with Network Coding," *IEEE Journal on Selected Areas in Communications*, vol. 28, pp. 268–276, 2010.

[14] X. Wang, Y. Xu, Y. Hu, and K. Ou, "MFR: Multi-Loss Flexible Recovery in Distributed Storage Systems," in *ICC*, 2010.

[15] R. Morales and I. Gupta, "AVMON: Optimal and Scalable Discovery of Consistant Availability Monitoring Overlays for Distributed Systems," *IEEE Transaction on Parallel and Distributed Systems*, vol. 20, pp. 446–459, 2009.

[16] C. Blake and R. Rodrigues, "High Availability, Scalable Storage, Dynamic Peer Networks: Pick Two," in *HotOS*, 2003.

[17] R. Ahlswede, N. Cai, S.-Y. Li, and R. Yeung, "Network Information Flow," *IEEE Transactions On Information Theory*, vol. 46, pp. 1204–1216, 2000.

[18] S.-Y. Li, R. Yeung, and N. Cai, "Linear Network Coding," *IEEE Transactions On Information Theory*, vol. 49, pp. 371–381, 2003.

[19] R. Koetter and M. Médard, "An Algebraic Approach to Network Coding," *IEEE/ACM Transactions on Networking*, vol. 11, pp. 782–795, 2003.

[20] A. G. Dimakis, V. Prabhakaran, and K. Ramchandran, "Ubiquitous Access to Distributed Data in Large-Scale Sensor Networks Through Decentralized Erasure Codes," in *IPSN*, 2005.

[21] ——, "Decentralized Erasure Codes for Distributed Networked Storage," in *Joint special issue, IEEE/ACM Transactions on Networking and IEEE Transactions on Information Theory*, 2006.

[22] A. Kamra, V. Misra, J. Feldman, and D. Rubenstein, "Growth Codes: Maximizing Sensor Network Data Persistence," in *SIGCOMM*, 2006.

[23] Y. Lin, B. Li, and B. Liang, "Differentiated Data Persistence with Priority Random Linear Codes," in *ICDCS*, 2007.

[24] T. Ho, M. Médard, R. Koetter, D. Karger, M. Effros, J. Shi, and B. Leong, "A Random Linear Network Coding Approach to Multicast," *IEEE Transaction on Information Theory*, vol. 52, pp. 4413–4430, 2006.

[25] A. Duminuco and E. Biersack, "A Pratical Study of Regenerating Codes for Peer-to-Peer Backup Systems," in *ICDCS*, 2009.

[26] A. G. Dimakis, K. Ramchandran, Y. Wu, and C. Suh, "A Survey on Network Codes for Distributed Storage," *The Proceedings of the IEEE*, vol. 99, pp. 476–489, 2010.

[27] Y. Wu and A. G. Dimakis, "Reducing Repair Traffic for Erasure Coding-based Storage via Interference Alignement," in *ISIT*, 2009.

[28] K. V. Rashmi, N. B. Shah, P. V. Kumar, and K. Ramchandran, "Explicit Construction of Optimal Exact Regenerating Codes for Distributed Storage," in *Allerton Conference on Control, Computing, and Communication*, 2009.

[29] N. B. Shah, K. Rashmi, P. V. Kumar, and K. Ramchandran, "Explicit Codes Minimizing Repair Bandwidth for Distributed Storage," in *ITW*, 2010.

[30] C. Suh and K. Ramchandran, "Exact Regeneration Codes for Distributed Storage Repair Using Interference Alignment," in *ISIT*, 2010.

[31] J. Bang-Jensen and G. Gutin, *Digraphs: Theory, Algorithms and Applications*. Springer-Verlag, 2001.
[32] A. Duminuco and E. Biersack, "Hierarchical Codes: How to Make Erasure Codes Attractive for Peer-to-Peer Systems," in *P2P*, 2008.
[33] F. Oggier and A. Datta, "Self-repairing Homomorphic Codes for Distributed Storage Systems," in *INFOCOM*, 2011.
[34] J. Li, S. Yang, X. Wang, and B. Li, "Tree-structured Data Regeneration in Distributed Storage Systems with Regenerating Codes," in *INFOCOM*, 2010.

Centre de recherche INRIA Rennes – Bretagne Atlantique
IRISA, Campus universitaire de Beaulieu - 35042 Rennes Cedex (France)

Centre de recherche INRIA Bordeaux – Sud Ouest : Domaine Universitaire - 351, cours de la Libération - 33405 Talence Cedex
Centre de recherche INRIA Grenoble – Rhône-Alpes : 655, avenue de l'Europe - 38334 Montbonnot Saint-Ismier
Centre de recherche INRIA Lille – Nord Europe : Parc Scientifique de la Haute Borne - 40, avenue Halley - 59650 Villeneuve d'Ascq
Centre de recherche INRIA Nancy – Grand Est : LORIA, Technopôle de Nancy-Brabois - Campus scientifique
615, rue du Jardin Botanique - BP 101 - 54602 Villers-lès-Nancy Cedex
Centre de recherche INRIA Paris – Rocquencourt : Domaine de Voluceau - Rocquencourt - BP 105 - 78153 Le Chesnay Cedex
Centre de recherche INRIA Saclay – Île-de-France : Parc Orsay Université - ZAC des Vignes : 4, rue Jacques Monod - 91893 Orsay Cedex
Centre de recherche INRIA Sophia Antipolis – Méditerranée : 2004, route des Lucioles - BP 93 - 06902 Sophia Antipolis Cedex

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