Draft Submission to... **XX,** X–XX *Cambridge University Press, 2016*
doi:doi:10.1017/XXXX

On Data Access Latency of Network Codes

Suayb S. Arslan* and Elif Haytaoglu†

As the requirements of distributed networks have changed dramatically in the past decade, novel code designs, also referred as network codes, due to their constructions based on network flow models have been proven to be useful for better bandwidth and I/O utilizations. One of the performance metrics that describes suitability of erasure codes for storage applications is known as average read overhead which represents the average number of extra whole device readings as an overhead in order to access any unavailable (failed) coded data block. This is also known as degraded read in the storage literature. Network codes are typically analyzed in terms of occupied storage capacity and repair bandwidth and hence the very well known trade-off between these two important resources of storage networks. Contrary to that, in this study, we analyze the data access latency of various modern network codes by introducing a common framework for which all linear erasure codes can be analyzed and compared.

Introduction and Motivation

TODO: We need a good literature survey. What has been done and what is being done in this field of research. In what parts, we differ and contribute to the literature. Advantages/disadvantages etc.

Background

Let us assume we store a file of F bits. This file is partitioned into k equal size fragments, usually called data *chunks*, each of size F/k bits. A total of k fragments are grouped together and encoded into n equal size (F/k bits each) coded chunks, $n \geq 2$, using an (n, k) linear erasure correcting code of rate $R = k/n < 1$. Encoded data (n chunks) is split equally and is stored in m distinct nodes, $2 \leq m \leq n$ implying that each node can store multiple chunks and $m|n$. Therefore, each network node in this setting stores α bits where

$$\alpha = \frac{nF}{mk} = \frac{F}{mR} \geq \frac{F}{k} \tag{1}$$

Furthermore, we assume that the original file can be reconstructed from any subset of h nodes, $1 \leq h < m$, which is known as the *download locality* in the literature. In other words, we use $\alpha h \geq F$ bits to perfectly reconstruct the original file where the inequality is due to well known pigeonhole principle. Whenever a coded chunk is lost, the degraded read (the same way the repair process is handled) can be performed by contacting any r network nodes, $1 \leq r < m$, and downloading $\beta \leq \alpha$ bits from each node. In other words, we need to download a total of $r\beta \geq \alpha$ bits where r is known as *repair locality*. However, although this is the accepted norm in literature for balanced communication cost, some network codes only allow unbalanced communication i.e., different amounts of downloads from each node. Thus, we need to generalize this operation

*Dr. Arslan is with MEF University, MAslak, Turkey. 2018
†Dr. Haytaoglu is with Pamukkale University.

as follows. If the network node downloads $\beta_{s_1}, \ldots, \beta_{s_r}$ bits from each of the r helper nodes, respectively, then β is defined to be the average i.e., $\beta = \frac{1}{r}\sum_i \beta_{s_i} \geq \alpha/r$.

For the rest of the paper, we will refer to a generic network code as $[m, h, r, \alpha, \beta]$ erasure code satisfying above definitions. Using the presented context, we note that one way to visualize a classical MDS codes is to treat it as a $[n, k, k, F/k, F/k]$ network code. A very well known n-replication scheme can also be expressed using the same formulation and it can be considered as an $[n, 1, 1, F, F]$ network code. It is important to observe that these treatments are not unique but quite conventional.

An exact-repair Minimum Storage Regenerating (MSR) code is a $[m, h, r, F/h, F/k]$ network code with $k = h(r - h + 1)$, $n = m(r - h + 1)$ and $r = 2(h - 1), \ldots, m - 1$. Note that if $r = h \in \{1, 2\}$[1], then we have a $[n, k, k, F/k, F/k]$ network code due to $r = h = k$ and $n = m$ reducing it to a standard RS-type MDS code. Also, if we want to minimize the repair bandwidth $r\beta$ we should choose $r = m - 1$, which then results in the following communication cost for a single node repair,

$$r\beta = \frac{F(m-1)}{k} = \frac{F(m-1)}{h(m-h)} \leq F \tag{2}$$

An exact-repair Minimum Bandwidth Regenerating (MBR) code is a $[m, h, r, rF/k, F/k]$ network code with $k = hr - h(h-1)/2$, $n = mr$ for $r = h, \ldots, m - 1$. Similarly, the repair bandwdith of an MBR code is minimized when $r = m - 1$ which results in the following communication cost for a single node repair,

$$r\beta = \frac{rF}{k} = \frac{2F(m-1)}{h(2m-h-1)} \leq F \tag{3}$$

from which it is not hard to show that for $h \geq 2$, MBR codes allows better repair bandwidth compared to MSR codes.

On the other hand, a lower repair locality can be achieved using Locally Repairable Codes (LRCs). Using the earlier notation, a $[m, h, r, F(r+1)/k, F(r+1)/k]$ LRC has $k = rh$ and $n = m(r + 1)$, where $r < h$ and $(r + 1)$ divides the number of nodes m. In LRC, helper nodes share all the information they store whenever a repair process is initiated. In other words,

$$r\beta = r\alpha = \frac{F(r+1)}{h} = \frac{Fr(r+1)}{k} \leq F \tag{4}$$

where m storage nodes are aranged in $m/(r + 1)$ disjoint repair groups where each local group contains $r + 1$ nodes. Any failure in each local groups can be recovered using the contents of the other r neighbour nodes.

In a very similar manner, functional-repair MBR and MSR codes can be analyzed using the same notation and context. This formation will help us derive a common analysis framework and hence the developed expressions will be applicable to all types of linear erasure codes under the same umbrella.

Average Read Overhead

Average read overhead is defined to be the average number of bits read to resurrect any particular file chunk in a coded system. An example is shown in Fig. 1 for a single failure case using

[1]Other values are not permissable for both r and h by construction.

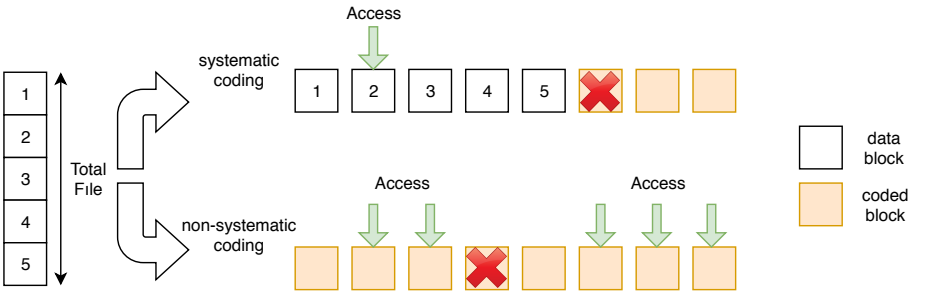

Figure 1: A file is chopped into 5 equal size segments and gets encoded into 8 segements using either systematic or non-systematic MDS codes. Ths figure shows the number of accesses made in each case if there is a single segment/node failure ($n = m$).

(8,5) MDS code for illustration purposes. A file is borken into 5 equal-size chunks and either a systematic or non-systematic MDS code is applied to generate coded chunks as shown in orange. For non-systematic MDS codes, in case of a single chunk failure/unavailability, to be able to access any chunk, we need to access α bits unless it is the failed chunk (5α bits in that case) for reconstruction. In fact, this is always the case for any coded chunk access making the average read access overhead to be $(5\alpha + 7\alpha)/8 = 1.5\alpha$.

On the other hand for systematic MDS codes, depending on where the failure is, read access latency might be different. If the failure is on coded blocks, then the data is readily available and the read overhead would be α bits. However, if the failure is in data blocks, we need to access 5 other blocks to access the failed block and one block for the non-failed ones making an average of $9\alpha/5$ bits access. Since the probability of failure to land on any block is the same, the average read overhead is given by $\frac{5}{8} \times \frac{9\alpha}{5} + \frac{3}{8} \times \alpha = 1.5\alpha$ which is the same as the overhead given for non-systematic version. For the rest of our discussion, we shall focus on the non-systematic codes as systematic versions of network codes may not be readily available.

In general, multiple node content might be accessed by the users in case of multiple failures. So we need to generalize the repair locality concept to multiple simultaneous node failures. Whenever i coded chunks are lost, the repair can be performed by contacting any (at maximum) r_i network nodes, $1 \leq r_i < m$, and downloading $\beta \leq \alpha$ bits from each node. In other words, we need to download $r_i\beta \geq r\beta = r_1\beta \geq \alpha$ at worst case where r_i is known as i-node repair locality upper bound.

Upper and Lower Bound

For a given (n, k) erasure block code (not necessarily MDS), let $S_i^{(m)}$ be the set of nodes that we need to contact at minimum to reconstruct i^{th} node. Let us assume we have $0 \leq j < m$ failed/unresponsive nodes. The repair overhead (and its size) conditioned on j failures depends in general on which j blocks are failed, because $S_i^{(m)}$s might be different for each i. In fact if for example we have 1^{st}, 2^{nd} and 3^{rd} blocks are failed, then we need to access all the nodes in the system defined by the set of nodes if we repair failures independent of eachother

$$S_1^{(m)} \cup S_2^{(m)} \cup S_3^{(m)} \tag{5}$$

Let us consider the class of network codes in general. For each j failure combinations, let us sum the total number of accesses using the above logic. This expression shall be given by

$$\binom{m}{j} r\beta \leq \sum_{s=1}^{\binom{m}{j}} \left| \bigcup_{c \in C_s} S_c^{(m)} \right| \leq \binom{m}{j} r_j \beta \tag{6}$$

where C_s is the set of indexes corresponding to s-th combination of all $\binom{m}{j}$ combinations. On the other hand, for unfailed $m - j$ nodes, we have access overhead of α because they are readily readable without further operation. This is the case since the amount of bits that needs to be communicated is α to be able to access α available bits. Since we sum all the different combinations, we finally have $\binom{m}{j}(m-j)\alpha$ access read overhead. Finally, we have the following upper bound on the total number of accesses (using inequality (6))

$$\sum_{s=1}^{\binom{m}{j}} \left| \bigcup_{c \in C_s} S_c^{(m)} \right| + \binom{m}{j}(m-j)\alpha \leq \binom{m}{j} \left(r_j \beta + (m-j)\alpha \right) \tag{7}$$

Assuming independent failure and repair processes with the failure probability p in the network, and let X be the random variable characterizing the number of failures. Using the bound given in (7), we have the following upper bound on the average read access rate per node

$$\begin{aligned}
\frac{1}{m} \sum_{j=0}^{m-1} Pr\{X = j\} \left(\frac{1}{\binom{m}{j}} \sum_{s=1}^{\binom{m}{j}} \left| \bigcup_{c \in C_s} S_c^{(n)} \right| + (m-j)\alpha \right) &\leq \frac{1}{m} \sum_{j=0}^{m-1} \binom{m}{j} p^j (1-p)^{m-j} \left(r_j \beta + (m-j)\alpha \right) \\
&\leq \frac{1}{m} \sum_{j=0}^{m-1} \binom{m}{j} p^j (1-p)^{m-j} \left(jr\beta + (m-j)\alpha \right) \\
&= \frac{1}{m} \sum_{j=0}^{m} \binom{m}{j} p^j (1-p)^{m-j} \left(jr\beta + (m-j)\alpha \right) - r\beta p^m \\
&= p(r\beta - \alpha) + \alpha - r\beta p^m \\
&= r\beta p(1 - p^{m-1}) + \alpha(1-p)
\end{aligned} \tag{8}$$

where we have used the union bound as well as the expectation of binomial distribution. Note that we have summed for $m - 1$ number of failures because for $j = m$, there remains nothing for a successful recovery operation. Similarly, using equation (6), we can find the following lower bound (i.e., it is enough to access r network nodes in case of all possible j combinations of unavailable or lost nodes),

$$\frac{r\beta}{m} \left(1 - p^m - (1-p)^m \right) + \alpha(1-p) \tag{9}$$

which equals to the upper bound if $p \to 0$ which is quite intutive and in that case both bounds are equal to α bits. Note that for classical MDS block codes, we strictly satisfy the equality $r_j\beta = r\beta = k\alpha$ with an average read access rate per node of $\alpha = F/k$. One of the things we notice is that both bounds achieve their maximum at different p values and as $p \to 1$, they both tend to zero. This does not make much sense since as $p \to 1$, the access overhead should not be defined due to irrecoverability. Thus, we need to make sure that we have at least r available (not lost) network nodes to make read access possible.

TODO list now includes:

- Compare various codes MSR, MBR (their functional repair versions as well), LRC, and classical MDS codes using these expressions. Perhaps MATLAB plots will suffice? .

  EH: I assume that we should consider cooperative regenerating codes because of denoting repair cost as $r_j\beta$ not $jr\beta$. Is it true? I've searched for a functional cooperative regenerating codes. However, there is not much study about this code settings. All the cooperative repair models are based on the exact repair. Nevertheless, I'll add some results based on the cooperative exact regenerating codes by Shum and Chen 2016.

  SSA: Hocam, You are absolutely right. I have just given a crude upper bound on $r_j\beta$ not even considered such modern cooperative techniques. Definitely we can improve our bound using the bounds of their paper. Let me give more attention to what they have got to offer.

  EH: Apart from this, if the cooperative regeneration is not considered in functional regenerating codes, there are few functional regenerating code models (repair by transfer, different specific d values, specific n values etc.). Should we investigate them one by one or consider in a more general manner?

  SSA: I'll prefer general methodology if possible. Do you mean functional cooperative regenerating codes is not something studied?

  EH: Hocam, I could not find any study related with functional cooperative regeneration.

  SSA: Hocam, I think cut-set bounds in the literature are applicable to both functional and exact repairs, but most constructions are exact. Why? because they are deemed more useful for storage access point of view. On the other hand, since random linear network coding achieves the cut-set bound in a single-source multi-cast network, the functional regeneration problem is usually considered solved. Maybe that's why we do not see explicit functional repair constructions. This simply means we can directly use bounds.

  EH: Moreover, in practice when there are lots of node failures, should we think that each lost node is repaired by their newcomers separately -not being cooperatively- or after the number of node failures reach a some threshold a newcomer may repair other newcomer node's content and send the repaired data (α) bits directly to it?

  SSA: What is the motivation behind assuming a threshold? This might be an interesting cooperative strategy though if we find a good motivation.

  EH: I mean, if there are many node failures and the bandwidth consumption occurred while repairing these lost nodes one by one and is too high. The newcomer node (or the node employing the repair procedure) may repair other nodes by gathering more than $r\beta$ symbols.

  SSA: In the case of multiple failures, the problem of accessing the contents of these erased nodes comes in two variations similar to "multiple-repair" literature. The first is the centralized model, where a single new comer (client or requester) is responsible for the access of all the failed nodes and hence there needs to communicate nothing with other potential newcomers. The other is the cooperative model, where we have multiple requesters (new comers) and each requests one of the failed node content. In that case, requesters (newcomers) may communicate (share information), and the amount of data exchanged between them needs to be included in the access latency calculations.

  EH: If we try to access the data on distributed storage and then realized that some of the connected nodes are not alive, we can reconstruct the file after repair the unavailable nodes.

Isn't it degraded read? If so, in the Equation (6), the second term may be $(h-j)\alpha$ not $(m-j)$? Am I missing sth?

SSA: Hocam, consider a single node failure. If we would like to read the data in that node, we perform degraded read. The difference between a degreaded read and repair is that in the degraded read, we communicate minimum data from other nodes and with minimum effort we reconstruct the original data but we keep it in the RAM (because usually the data request comes from a client rather than internal system). The important thing is to communicate the data residing on that node to the requester as fast as possible. In the other hand, repairs are initiated by the system to recover the lost reliability and it involves similar process and I/O because we need persistent storage. Well we can first repair the original data with a newcomer and then provide the repaired data to the requester...but that would delay the requested operation? Also, requesters can communicate with eachother? (we need it to be able to use cooperative regeneration) Maybe? Maybe not always?

On the other hand, $(m-j)$ term is to count all possible single node accesses. In other words, if we would like to access any one of $(m-j)$ unfailed nodes we need to communicate α bits total. That's why for each one of these unlos nodes, we multiply it by α to get $(m-j)\alpha$. Does that explain your question? or maybe I could not understand the question well?

EH: I eventually figured out :). I had considered reading the file rather than reading node content. But, among m nodes some nodes are systematic and some nodes are not systematic, so the access times of nodes may change, access rate of systematic ones may be high. The average degraded read may change regarding to these unbalanced accesses.

SSA: Hocam, you are right the access latency is a function of access pattern (balanced/unbalanced). Here we just assumed the rate of accesses for each node content is equally likely. What is the real pattern? I do not know. We just consider an average read access latency with uniform conditionals.

- Improve this bound. For instance in the upper bound, we have $j\beta$ which cannot be greater than m because we have m nodes in the system. So we can replace $j\beta$ with $\min\{j\beta, m\}$ in the expression and bound it with $(j\beta)^s m^{1-s}$ for any $0 < s < 1$. Furthermore, $j <= m - r$ implies that we have at least r nodes available to help repair the failed node contents. Hence in (8), the sum's upper limit should be changed to $m - r$ to make the upper and lower bounds meaningful. In that case, closed form expressions might not be possible?

  EH: Maybe, after the number of node failures (j) becomes higher than h, we may use the term $h * \alpha$ rather than $r_j\beta$.

  SSA: That is right, in other words, we can switch back to conventional repair just like in classical RS codes. Let me think about all these possibilities (including cooperative regeneration) to suggest improvements on the upper bound and present it to your approval.

- Consider non-regular constructions having different S_c^m, β for different failure combinations. Such codes includes fountain codes, raptor codes, etc. and let us compare it with the previous item.

References

Shum, Kenneth W., and Junyu Chen. 2016. "Cooperative Repair of Multiple Node Failures in Distributed Storage Systems." *IJICoT* 3:299–323.