Selective Regenerating Codes

Abbas Kiani and Soroush Akhlaghi

***Abstract*— Regenerating codes are mainly justified due to their ability to reduce the repair bandwidth incurred by a newcomer node. This happens when a node fails or leaves the network, thus a new node is initiated, attempting to connect to existing nodes to reconstruct the data. This paper aims to investigate the case in which the newcomer can wisely select some of existing nodes to connect to, so as to reduce the repair bandwidth. Accordingly, selective regenerating codes are proposed, showing the corresponding repair bandwidth is dramatically reduced as compared to that of existing codes.**

***Index Terms*—Network coding, information flow graph, regenerating codes.**

I. Introduction

DATA in distributed storage systems should be stored reliably over unreliable nodes. This requires to distribute some redundancy information over storage nodes. Moreover, to have a long term durability, the network should have the possibility of self-repairing, meaning when a node is damaged, it is replaced with a newcomer node. This, in turn, requires a great deal of data transferring over the network, dubbed the repair bandwidth. Dimakis et al. in [1] proposed a new coding strategy, called regenerating code (RC), to make a balance between the minimum achievable repair bandwidth and the storage capacity per node. This problem is further investigated in [2], [3] when there is a download cost associated with each node. Accordingly, in [3] a new type of codes, called Generalized Regenerating Codes (GRCs) is proposed, showing the download cost of GRC is dramatically reduced as compared to that of RC.

Distributed storage systems can be modeled as an information flow graph [1], a directed acyclic graph, with three kinds of nodes: (i) A single source (S), (ii) Some storage nodes, (iii) Data collectors (DCs). The source node is the source of original data file. The storage nodes store α bits and according to each request of reconstructing the original data file, a DC is added to the network and connects to k out of existing n nodes. The edges departing the storage nodes and arriving at a DC node are assumed to have an infinite capacity. When a node is damaged, a newcomer connects to $k \leq d \leq n-1$ nodes and downloads β bits from each. So the repair bandwidth is $\gamma = d\beta$ bits [1].

The authors in [1] showed that the repair problem can be translated to a multicast problem over the corresponding information flow graph. More importantly, it is demonstrated that as long as the minimum cut set between S and the affiliated DC is greater than the size of original data file, the entire data file can be successfully retrieved. Accordingly, based on the minimum cut set bound, an optimal tradeoff curve between the storage per node and repair bandwidth is identified and is shown that any point on this curve can be achieved through the use of network coding [4]. This curve has two extremal points; one end of this curve corresponds to the minimum storage per node and the other end corresponds to minimum bandwidth point. These two extremal points can be achieved by the use of Minimum Storage Regenerating (MSR) and Minimum Bandwidth Regenerating (MBR) codes, respectively. Accordingly, the storage per node (α) and repair bandwidth (γ) for MSR and MBR codes are computed, respectively, as [1], $(\alpha_{MSR}, \gamma_{MSR}) = (\frac{M}{k}, \frac{Md}{k(d-k+1)})$ and $(\alpha_{MBR}, \gamma_{MBR}) = (\frac{2Md}{2kd-k^2+k}, \frac{2Md}{2kd-k^2+k})$, where M represents data size and d denotes the number of storage nodes a newcomer is connected to. Moreover, k is the total number of storage nodes to which a DC connects. Note that as $\alpha(.)$ and $\gamma(.)$ are decreasing functions with respect to d, their minimum values are achieved when d takes its maximum value, i.e., $d = n-1$.

Note that from the network management perspective, it is desirable to keep the same number of storage nodes across time, say n nodes, where some of which are new comers which are connected to other nodes. Regenerating codes introduced in [1] are motivated by the assumption that a newcomer does not have the capability of selection d out of $n-1$ existing nodes to connect to. However, it is interesting to explore the case in which the new comer can wisely select these d nodes to reduce the resulting repair bandwidth. This motivated us to investigate the aforementioned issue when a newcomer can choose its own surviving nodes. Accordingly, a new type of code, called Selective Regenerating Code (SRC), is proposed and its performance is compared to that of RC codes.

The rest of this paper is organized as follows: Section II presents system model. Sections III and IV propose SRC. Finally, Section V concludes the paper.

II. System Model

Figures 1 and 2 depict information flow graphs associated with two distributed storage networks investigated in the current work. Referring to these figures, the i^{th} node is depicted with two distinct nodes X^i_{in} and X^i_{out}, representing input and output ports, respectively, which are connected through a directed edge with capacity α, indicating the storage capacity of this node. Moreover, there are two different types of nodes: (i) solid-line circles (initial nodes) and (ii) dashed-line circles (repaired/under construction nodes). Referring to the aforementioned figures, S and DC nodes are connected with infinite capacities to storage nodes, while newcomers connect to existing nodes with edges of capacity β, indicating

Manuscript received November 22, 2010. The associate editor coordinating the review of this letter and approving it for publication was I. Maric.

The authors are with Shahed University, Tehran, Iran (e-mail: {akiani, akhlaghi}@shahed.ac.ir).

Digital Object Identifier 10.1109/LCOMM.2011.061611.102271

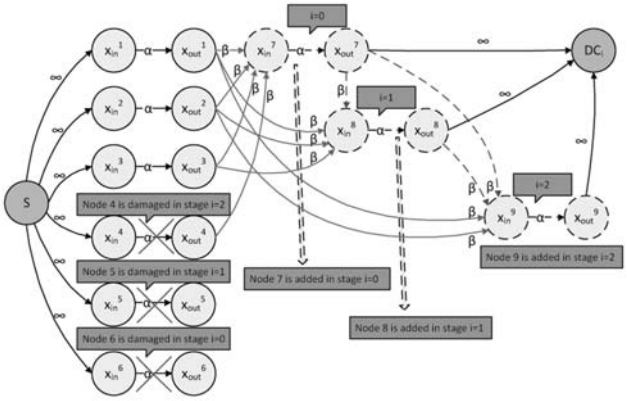

Fig. 1. The graph $\mathcal{G}^*$ for (6,3) regenerating code, where the newcomer should connect to $d = 4$ surviving nodes.

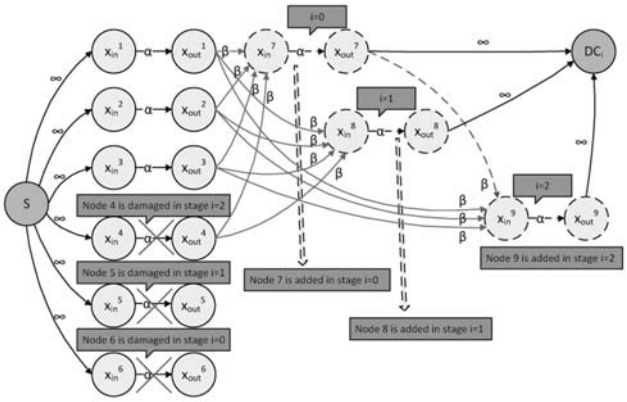

Fig. 2. The graph $\mathcal{G}^*$ for (6,3) selective regenerating code, where the newcomer should select $d = 4$ surviving nodes to connect to.

they can merely download β bits from each surviving node. In [1], it is argued that "If the minimum of the min-cuts separating the source with each data collector is larger or equal to the data object size M, then there exists a linear network code defined over a sufficiently large finite field F (whose size depends on the graph size) such that all data collectors can recover the data object".

Note that the graph changes through the so called failure/repair stage in which an active node leaves the system, thus a newcomer is added to the system. Noting this, one can divide current n active nodes in two parts: (i) the initial nodes, called the first-type nodes, and (ii) the repaired/under construction nodes, called the second-type nodes. Thus, depending on the number of first-type and second-type nodes, we have sort of graphs $\mathcal{G}$ associated with each case, each having different min-cut values. In this work, it is assumed DC is restricted to connect to any k consecutive active nodes, no matter if they are of different types. Accordingly, assuming $\mathcal{G}^*$ is the one with minimum min-cut value, one can verify that this graph is the one which has k consecutive nodes of the second-type.

Figure 1 illustrates the graph $\mathcal{G}^*$ associated with a $(n, k) = (6, 3)$ regenerating code introduced in [1]. Recall that regenerating codes do not have the ability of selecting d out of $n-1$ surviving nodes, thereby any possible choices of selecting d out of $n-1$ nodes may happen. Thus, one need to rely on the worst case scenario to define the minimum min-cut value. Figure 1 is actually the corresponding worst case graph for a $(n, k) = (6, 3)$ code after three failure/repair stages indexed by i=0,1,2. In each stage, a newcomer is added to the system and connects to $d = 4$ active nodes. For instance, referring to Fig. 1, it is assumed node 6 leaves the network during the first stage, and thus node 7 is added to the network. Similarly, nodes 5 and 4, respectively, leave the network, and in turn, nodes 8 and 9 are initiated. Note that in the current work, we investigate that case in which the newcomer can wisely select d surviving nodes, thereby improving the min-cut value associated with worst case graph. For instance, Fig. 2 depicts the worst case graph $\mathcal{G}^*$ associated with the code $(n, k) = (6, 3)$ in which the newcomer prefers connecting to the active nodes of the first type. The following section aims at investigating the tradeoff curve for the storage capacity per node versus the repair bandwidth for two different scenarios.

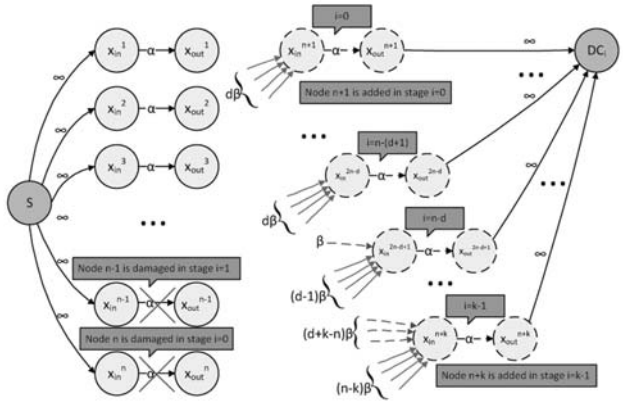

Fig. 3. The graph $\mathcal{G}^*$ for (n,k) selective regenerating code when $\max\{(n-k+1), k\} \leq d \leq n-1$.

III. Selective Regenerating Codes when $\max\{(n-k+1), k\} \leq d \leq n-1$

Note that the newcomer prefers to connect to surviving nodes of the first type. However, there are some cases in which the newcomer can not find d surviving nodes of the first type to connect to. Figure 3 depicts an example in which up to the stage $i = n - (d + 1)$, newcomers can select d surviving nodes of the first type, however, for the next stages newcomers should connect to some of nodes of the second type. This happens when $\max\{(n-k+1), k\} \leq d \leq n-1$, and referring to graph $\mathcal{G}^*$ as is illustrated in Fig. 3, the following condition is necessary to reconstruct the original data file (Min-Cut theorem),

$$\sum_{i=0}^{n-d-1} \min\{d\beta, \alpha\} + \sum_{i=n-d}^{k-1} \min\{(n-i-1)\beta, \alpha\} \geq M \ . \quad (1)$$

Accordingly, noting (1) and after some manipulations, the following tradeoff curve between the minimum storage capac-

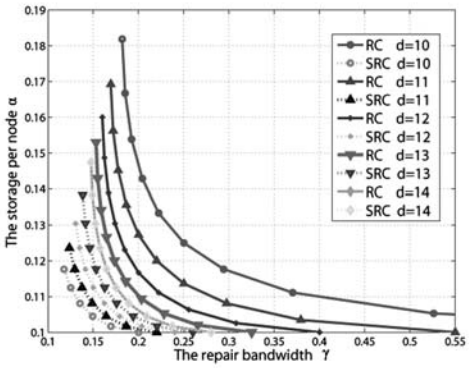

Fig. 4. The tradeoff curves between storage per node and repair bandwidth for (15,10) RC and (15,10) SRC when $10 \leq d \leq 14$.

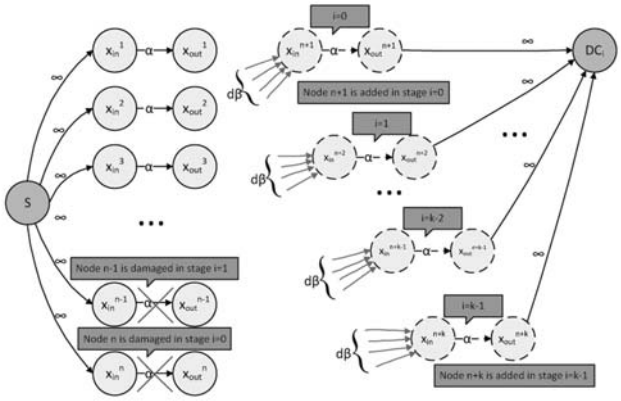

Fig. 5. The graph $\mathcal{G}^*$ for (n,k) selective regenerating code when $k \leq d \leq n-k$.

ity per node (α_{min}) and repair bandwidth (γ) is identified [1],

$$\alpha_{min}(n,k,d,\gamma) = \begin{cases} \frac{M}{k} & \gamma \in \big[f(-1),\infty\big) \\ \frac{2M-g(i)\beta}{2(k-i-1)} & \gamma \in \big[f(i),f(i-1)\big) \end{cases}, \quad (2)$$

where $f(i) \triangleq \frac{2Md}{2k(n-k)+(i+1)(2k-i-2)}$ and $g(i) \triangleq (i+1)(2n-2k+i)$ for $-1 \leq i \leq k+d-n-1$. One can readily verify that $f(.)$ is a decreasing function, hence, γ_{min} (the minimum repair bandwidth) can be computed as $\gamma_{min} = f(k+d-n-1)$.

This tradeoff curve has two extremal points; One end of this curve corresponds to minimum storage point and the other end corresponds to minimum repair bandwidth point. These points can be achieved through the use of linear network codes that we call them Minimum Storage Selective Regenerating (MSSR) codes and Minimum Bandwidth Selective Regenerating (MBSR) codes, respectively. It is worth mentioning that we have $(\alpha_{\text{MSSR}},\gamma_{\text{MSSR}}) = (\frac{M}{k},\frac{Md}{k(n-k)})$ and $(\alpha_{\text{MBSR}},\gamma_{\text{MBSR}}) = (\frac{2Md}{n(2k+2d-n+1)-k(k+1)-d(d+1)},\frac{2Md}{n(2k+2d-n+1)-k(k+1)-d(d+1)})$.

An interesting observation is that the minimum repair bandwidth γ is an increasing function with respect to d and, hence, the minimum repair bandwidth takes its minimum value for $d = \max\{(n-k+1),k\}$, while for common regenerating codes the minimum repair bandwidth is achieved when d takes its maximum value, i.e, $d = n-1$. One can readily observe that both RC and SRC yield the same tradeoff curve for $d = n-1$, i.e., they have the same α_{min} and γ_{min} for $d = n-1$. As is argued earlier, one can readily verify that the tradeoff curve of SRC outperforms that of RC when $d < n-1$. This is more conspicuous when d takes its minimum value, i.e., $\max\{(n-k+1),k\}$. For instance, Fig. 4 compares the tradeoff curves of SRC$(15,10)$ with that of RC$(15,10)$ and for different values of d ranging from $\max\{(n-k+1),k\} = 10$ to $n-1 = 14$, showing the best tradeoff curve for SRC is achieved when $d = 10$.

[1]To derive (2) from (1), we investigate every possible choice of β (or equivalently $\gamma = \beta d$) for which $\min(\beta d,\alpha)$ and $\min((n-i-1)\beta,\alpha)$ take either their first or second arguments, then the corresponding values of α are derived.

IV. Selective Regenerating Codes when $k \leq d \leq n-k$

In this case, the newcomer can always select d surviving nodes of the first type to connect to. Figure 5 depicts the corresponding $\mathcal{G}^*$. Note that $k \leq d \leq n-k$ implies $k \leq \frac{n}{2}$, and referring to $\mathcal{G}^*$, the necessary condition to reconstruct the original data file becomes, $\sum_{i=0}^{k-1} \min\{d\beta,\alpha\} \geq M$, thus after some manipulations, it follows $\alpha_{min}(n,k,d,\gamma) = \frac{M}{k}$, for $\gamma \in \big[\frac{M}{k},\infty\big]$, showing the tradeoff curve merges to the single point $(\alpha^{min},\gamma^{min}) = (\frac{M}{k},\frac{M}{k})$, which can be achieved through the use of network coding.

V. Conclusion

A new variation of regenerating codes, called selective regenerating codes are introduced which outperform existing codes. Accordingly, the corresponding tradeoff curve between the storage per node and repair bandwidth is identified.

Acknowledgment

The authors would like to greatly acknowledge the financial support of Research Institute for ICT, Tehran, Iran.

References

[1] A. G. Dimakis, P. G. Godfrey, Y. Wu, M. J. Wainwright, and K. Ramchandran, "Network coding for distributed storage systems," *IEEE Trans. Inf. Theory*, vol. 56, no. 9, pp. 4539–4551, Sep. 2010.

[2] S. Akhlaghi, A. Kiani, and M. R. Ghanavati, "A fundamental trade-off between the download cost and repair bandwidth in distributed storage systems," in *Proc. IEEE International Symposium on Network Coding*, June 2010.

[3] S. Akhlaghi, A. Kiani, and M. R. Ghanavati, "Cost-bandwidth tradeoff in distributed storage systems," *Computer Commun.*, vol. 33, no. 17, pp. 2105–2115, Nov. 2010.

[4] R. Ahlswede, N. Cai, S. Y. R. Li, and R. W. Yeung, "Network information flow," *IEEE Trans. Inf. Theory*, vol. 46, no. 4, pp. 1204–1216, July 2000.