Practical Single Node Failure Recovery Using Fractional Repetition Codes in Data Centers

May Itani*, Sanaa Sharafeddine† and Islam ElKabbani*‡
*Department of Computer Science and Mathematics
School of Sciences, Beirut Arab University
Beirut, Lebanon
Email:may.itani@lau.edu.lb; Email:islam.kabani@bau.edu.lb
†Department of Computer Science and Mathematics
School of Arts and Sciences, Lebanese American University
Beirut, Lebanon
Email: sanaa.sharafeddine@lau.edu.lb

***Abstract*—Node failures in distributed storage systems are becoming a critical issue, and many erasure codes are designed to handle such failures. The purpose of this paper is to evaluate fractional repetition (FR) codes, a class of regenerating codes for distributed storage systems, as a practical solution. FR codes consist of a concatenation of an outer maximum distance separable (MDS) code and an inner fractional repetition code that splits the data into several blocks and stores multiple replicas of each on different nodes in the system. We model the problem as an integer linear programming problem that uses modified versions of the fractional repetition code by allowing different block sizes, and minimizes the recovery cost of all single node failure scenarios. The contribution of this work is three fold: We generate an optimized block distribution schema that minimizes the total system repair cost in a data center and we present a full recovery plan for the system. In addition, we account for new-comer blocks and allocate them to nodes with minimal computations and without changing the original optimal schema. This makes our work practical to apply. Hence, a practical solution for node failures is presented by using a self-designed genetic algorithm that searches within the feasible solution space. We show that our results are close to optimal.**

***Keywords*-distributed storage systems; FR codes; failure recovery; genetic algorithms**

I. Introduction

World data is increasing by more than two-fold every two years and Internet traffic is dominating our network highway. Photo storage in Facebook reached over 20 PB in 2011 and is increasing by 60 TB every week [1]. In fact, data is growing exponentially and data centers are becoming a top priority for businesses. Data centers have become critical for the very functioning of a big business enterprise. Any interruptions in the data center operations might cause huge losses for businesses if measures for interruptions or failures were not considered [2], [3]. Data centers use inexpensive individual hardware components that are prone to failure. "Stuff fails in data centers, and always has" [4]. For example, a study examined a large Facebook cluster of 3000 nodes with about 45 PB of raw capacity [5]. Fig. 1 shows that the average failure rate was 22 nodes a day, but failures could spike to more than 100 in a single day. Thus there is a great need for data protection from device failures, and for mechanisms to quickly recover or at least mask the effects of node failures from users and connected devices with minimum performance cost. The easiest way for storage system to tolerate failures and prevent data loss is to store replicas. This, however, results in decreased storage efficiency. Another alternative is to store encoded data using erasure coding [6] . Classical erasure codes transform a message of k symbols into a longer message (codeword) with n symbols such that the original message can be recovered from a subset of the n symbols. Although traditional erasure codes can reduce the storage overhead as compared to replication, extensive network resources are needed to repair a lost node. This is due to the fact that a surviving node should read all its data, process them, then send a linear combination of them to the replacement node. To minimize the bandwidth consumed during the repair process, regenerating codes were introduced in literature where a failed node can be recovered by connecting to a subset of d surviving nodes and downloading one block of data from each.

In this work, we consider a family of regenerating erasure codes that provide exact and uncoded repair where a surviving node reads the exact amount of data it needs to send to a replacement node without any processing. This allows for a low complexity repair process. These are the Fractional Repetition (FR) codes that were first introduced in [7].

FR codes consist of a concatenation of an outer maximum distance separable (MDS) code and an inner fractional repetition code. In an (n,k,d) system where n is the total number of storage nodes, k<n, is the total number of nodes

‡On leave from Alexandria University, Egypt

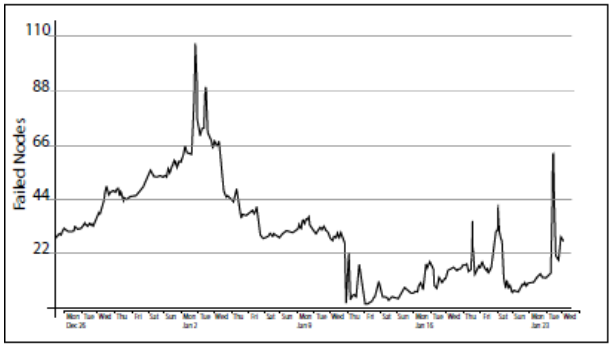

Figure 1. Number of failed nodes over a single month period in a 3000 node production cluster of Facebook [5]

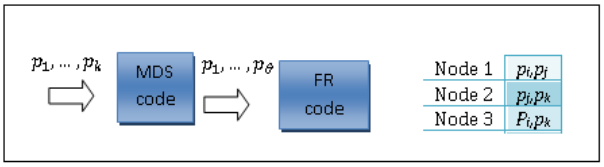

Figure 2. An FR code with repetition degree ρ =2. A file consisting of k=2 packets is encoded using (3,2) MDS code. θ =3 distinct encoded packets are replicated and distributed on n=3 nodes.

contacted to retrieve a file and d≥k, is the number of nodes contacted by a replacement node during node repair [7]. First, a file of size k packets is encoded using a (θ;k) MDS code as indicated in Fig. 2 such that any k out of θ packets can recover the file (MDS property). Then θ distinct packets are replicated ρ times and distributed on n storage nodes where each coded packet is replicated on distinct nodes. A user contacting k nodes can always decode the stored file and this is achieved by the MDS property of the outer code. A node failure can be repaired by constructing a new node that contacts a specific set of d nodes for repair depending on which node has failed. d represents also the minimum node storage capacity expressed in packets [7].

In this work we present a practical solution for node failures by implementing FR codes in a genetic algorithm that is based on natural evolution. Since the problem is huge and cannot be solved in polynomial time, the genetic algorithm provides acceptable solutions in acceptable time. The algorithm generates a close-to-optimal and a post-optimal block distribution schema after arrival of a new set of packets. Moreover, a full system recovery plan that minimizes the total system repair cost is generated. We demonstrate the optimality of the suggested algorithm through simulation results on a variety of nodes and blocks parameters.

We first provide a summary of the literature in section II. The system model and our formulated problem are discussed afterwards in sections III and IV, respectively. This is followed by algorithms description and implementation in section V, running examples and simulation results in section VI, and finally the conclusion in section VII.

II. Related Work

Several erasure codes [8]–[10] are being designed for the purpose of optimizing performance in distributed storage systems. Performance metrics include storage space, reliability and recovery cost. Of these codes, regenerating codes were first proposed as a new paradigm in coding that achieves minimum bandwidth requirements by Dimakis [11]. However, in case of node failure, regenerating codes require a helper node to read all its data and generate a linear combination of them to send it to the newcomer node. In this way the recovery approach does not satisfy the uncoded repair property [7] explained earlier in the introduction.

Later in 2010 constructions of FR codes were introduced by Rouayheb et al [7]. Of these, deterministic constructions based on regular graphs and Steiner systems were presented as a first step in the study of fractional repetition codes, where a helper node reads only one of its stored packets and sends it to the newcomer with no processing. Pawar et al proposed a randomized construction of FR codes based on the balls and bins probabilistic model [12]. These randomized constructions provided more flexibility in terms of possible system parameters compared to those constructions of El-Rouayheb et al [7]. Both constructions take different design considerations into account and are based on mathematical models that do not provide an optimal block storage scheme to achieve the goal of minimal recovery cost.

All of the above constructions were theoretical. Studying the feasibility of implementing these new codes in practical storage systems remains open.

As to recovery approaches, several studies were conducted on the failure recovery problem in distributed storage systems. We state that of Zhu, 2012 who proposed a recovery solution for distributed systems that use RDP and EvenOdd RAID6 erasure codes. Single node failures can be recovered by reconstructing the data of a failed node using decoding techniques [13]. Another work was conducted on Facebook warehouse clusters, where a framework was designed for a recovery approach based on codes that are efficient in the amount of data read and downloaded during node-repair. The basic idea behind this framework was to take multiple stripes of existing data on nodes and carefully add functions of the data of one stripe to other stripes. This technique required extra parities and computation of specific functions, and thus used coded repair [14]. As we observe, the above two approaches minimize the amount of data read during repair but require extra computations and decoding. The most recent and relevant work is that of Yu [15]. The author proposed a new version of FR codes, irregular FR codes, where different storage capacities of nodes, different communication costs, and different number of packets per

coded block are assumed. The design considerations require selection of helper nodes from a limited set of nodes determined using hyper graphs. In this work, we provide a new practical problem that was not tackled before where a new set of packets is to be allocated on the current system nodes following the optimal allocation of current packets. The problem is approached using the concept of incidence matrices such that all nodes in the network are candidates for being helper nodes. Our goal is to implement a function that generates a feasible and optimal (n$\times\theta$) incidence matrix that satisfies the FR code constraints, given specific system parameters n, d, ρ, and θ. Then the incidence matrix will be augmented with additional columns that account for newcomer blocks in a practical minimal computation way that can be easily implemented in reality without the need to redistribute original on-node blocks. A storage allocation matrix is computed based on an integer linear program that selects the optimum recovery cost distribution scheme and the corresponding table-based full-recovery plan for all possible scenarios of single node failures.

III. System Model

Data center infrastructure design is based on a layered approach where issues regarding improving scalability, performance, flexibility, resiliency, and maintenance are considered. The three functional layers of the data center are the core layer, aggregation layer and access layer [16] and they are described below:

1) The core layer is central to the data center network and provides interconnection between the aggregation layers. It uses high performance low latency switches providing high densities that operate only on layer 3 devices.
2) The aggregation layer acts as a services layer for the data center. Services such as load balancing, SSL(Secure Socket Layer Optimization), Firewalling, etc are typically found at this layer. It is used also as an interconnection point for multiple access layer switches.
3) The access layer is where the servers are physically attached to the network. This layer contains modular switches that afford layers 2 and 3 topologies and allow future proofing the network [16].

Consider an (n,k,d) distributed storage system where the FR code of repetition factor ρ is to be used as a redundancy scheme. The underlying networked environment connecting the nodes may have different transmission bandwidths and topologies. Thus each storage node will have different communication costs. Our system model is depicted in Fig. 3 where $c_{\alpha j}$ represents the retrieval cost of a block j from a helper node α.

To explain our work, we provide a detailed example where we consider a distributed system with 6 nodes such that an encoded data object with 4 distinct blocks is to

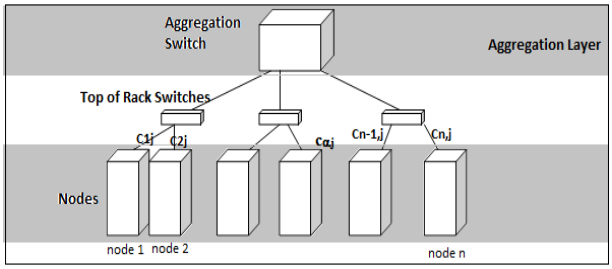

Figure 3. Distributed Storage System Diagram

be stored. Assume that the system can tolerate 2 node failures for a replication factor of 3. Each block will be replicated on three different nodes and each node will store two distinct blocks. The bandwidth and communications costs of different nodes need not be the same. An initial random block assignment matrix B where rows correspond to nodes and blocks correspond to columns would be as follows together with a given communication cost matrix C, is given below. The cost matrix associates with every block in a node a generic retrieval cost.

$$B = \begin{bmatrix} 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix} \quad C = \begin{bmatrix} 2 & 2 & 2 & 8 \\ 7 & 2 & 10 & 2 \\ 5 & 7 & 6 & 6 \\ 3 & 9 & 7 & 4 \\ 1 & 6 & 1 & 6 \\ 9 & 8 & 9 & 4 \end{bmatrix}$$

For the case of one node failure, assuming node 1 fails the optimal recovery schema is to recover block 3 from node 3 and block 4 from node 4. If node 2 fails, it is optimal to recover block 1 from node 4 and block 2 from node 5, and so on as specified in retrieval plan matrix R.

$$R = \begin{bmatrix} 0 & 0 & 3 & 4 \\ 4 & 5 & 0 & 0 \\ 4 & 0 & 1 & 0 \\ 3 & 0 & 0 & 5 \\ 0 & 2 & 0 & 4 \\ 0 & 2 & 1 & 0 \end{bmatrix}$$

The total recovery cost would be the cost of recovering all nodes individually taking into consideration all single node failure scenarios. The total optimal recovery cost for this distribution schema in this case will be RC = $c_{33} + c_{44} + c_{41} + c_{52} + c_{41} + c_{13} + c_{31} + c_{54} + c_{22} + c_{44} + c_{22} + c_{33} + c_{44}$ = 45.

However, if the block distribution matrix B was given as:

$$B = \begin{bmatrix} 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \end{bmatrix}$$

then the total recovery cost for all nodes would be 23. Hence, there should be an optimal distribution of blocks and replicated blocks among nodes so that the system recovery cost would be minimal. The optimal recovery schema for the second block assignment matrix B is given below in modified matrix R, where the new-comer node will replace the failed node by downloading respective blocks from the nodes specified in R.

$$R = \begin{bmatrix} 0 & 2 & 5 & 0 \\ 0 & 1 & 0 & 6 \\ 0 & 0 & 5 & 2 \\ 5 & 2 & 0 & 0 \\ 4 & 0 & 1 & 0 \\ 5 & 0 & 0 & 2 \end{bmatrix}$$

The interpretation of R would be to recover node 1 from nodes 2 and 5, recover node 2 from nodes 1 and 6, and so on..., based on optimal solution of block assignment matrix for all nodes.

For the case of new-comer blocks, once a block arrives, it is replicated ρ times and split among the nodes in an optimal way such that the original optimal block distribution matrix and the recovery plan are not changed. The new block locations are added on top of the optimized problem plan as described in the next example.

Given the optimized block distribution schema B, suppose a set of packets grouped in a single new block is to be stored in the system. Using the same concept of minimizing total system repair cost on top of the given optimal arrangement, the post-optimal block assignment matrix would be

$$B = \begin{bmatrix} 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 \end{bmatrix} \text{ given new retrieval costs } \mathrm{c}_{aj} \begin{bmatrix} 3 \\ 1 \\ 4 \\ 8 \\ 9 \\ 8 \end{bmatrix}$$

The corresponding post-optimal recovery plan is evaluated as:

$$R = \begin{bmatrix} 0 & 2 & 5 & 0 & 2 \\ 0 & 1 & 0 & 6 & 1 \\ 0 & 0 & 5 & 2 & 0 \\ 5 & 2 & 0 & 0 & 0 \\ 4 & 0 & 1 & 0 & 2 \\ 5 & 0 & 0 & 2 & 0 \end{bmatrix}$$

The interpretation of the above R would be the same as that in subsection A except for the new-comer block which should be retrieved from node 2 if nodes 1 or 5 fail and from node 1 if node 2 fails. This will be the best practical solution that will take minimal computations which allows it to be implemented on the spot.

IV. Problem Formulation

Given a distributed system with n nodes. θ distinct blocks are to be stored on the nodes with a given replication factor ρ. Every node can store a minimum of d blocks. Given different communication costs and link bandwidths between nodes, the problem of failure recovery with different failure scenarios is modeled and solved using incidence matrices.

Let the communication cost matrix represent the cost of new-comer node β to retrieve block j from a helper node α. Then we need to find a block assignment matrix that minimizes the following value:

Retrieval Cost

$$RC = min \sum_{\substack{i=1 \\ \alpha \neq i}}^{n} \sum_{j=1}^{\theta} \underbrace{min}_{\alpha \neq i} c_{\alpha j} x_{ij} . x_{\alpha j} \tag{1}$$

$\forall$ node α that holds a retrieval block and satisfies $\mathrm{c}_{\alpha j}$ minimal such that $\mathrm{x}_{\alpha j}$ & $x_{ij} \neq 0$.

$\mathrm{x}_{ij} \in \{0, 1\}$ is a Boolean variable indicating that node i holds block j. The value RC is to be minimized under the condition that the block assignment matrix satisfies the following constraints adopted from the FR code requirements, and other additional constraints explained next.

$$\sum_{i=1}^{n} x_{ij} = \rho \tag{2}$$

The above constraint is a replication factor constraint where the total number of replicas of a specific block j in all nodes is ρ.

$$\sum_{j=1}^{\theta} x_{ij} = d \tag{3}$$

Constraint (3) specifies the total number of blocks per node.

Assuming different size blocks and a specific storage capacity per node, we should add the following storage constraint, where s_j is the size associated with every block j and SC is the total storage capacity of each node in the system.

$$\sum_{j=1}^{\theta} s_j \leq SC. \quad (4)$$

The failure recovery problem is an integer linear programming problem and cannot be solved optimally in polynomial time for large network sizes. The next section will present the heuristic approach used to tackle the problem.

V. Genetic Algorithm Description and Implementation

Genetic algorithms are stochastic search algorithms based on natural evolution. Once a problem is clearly defined, and candidate solutions are represented in a discrete way (i.e binary string, distinct numbers string), then it can be solved using the GA major steps described in the context of our algorithm. Every iteration in a GA represents a generation. The entire set of generations constitutes a run. One or more highly fit chromosomes are expected to be generated at the end of a run.

The implemented code for the optimization methodology was designed as a self-cross-over genetic algorithm that starts with a random distribution of blocks on nodes and then searches within the feasible space by redistributing the blocks and generating a close-to-optimal solution [17], [18]. A description of the algorithm phases is stated next.

A. Chromosome Representation

The modeling of a chromosome was that each one represents a feasible solution that constitutes a binary block assignment matrix generated using different permutations and satisfying the constraints (2), (3) and (4). An example of a chromosome representation for the values (n=3, d=2, θ=3,ρ=2) is [1 0 1 1 1 0 0 1 1] where the n$\times\theta$ block assignment matrix is transformed to a single (n$\times\theta$,1) row matrix.

B. Generation of Initial Population

The phase that follows is generating an initial population for reproduction after carrying out the chromosome encoding phase. Given a pre-specified population size *p*, the initial population will include *p* chromosomes that are generated at random using a self-designed constructive method implemented as a function that generates feasible allocation schema.

C. Self Cross-over and Mutation Operations

The conventional cross-over technique cannot be applied in our model since it will result in an in-feasible new chromosome. This paper adopts the self-cross-over method inspired by the fact that the feasible allocation matrices can be generated from one another by exchanging rows within a single matrix. The implemented cross-over procedure was done as specified in Algorithm 1.

Algorithm 1. Self Cross-over Implementation

1. Select single parent chromosome from population based on fitness function
2. Generate one, or two cross-over points (either random or specified by user)
3. Genes between cross-over points move to the offspring swapped.
4. Repeat steps 1 to 3 till we generate a certain number of off-springs.

Note that elitism was used to help keep up the optimal chromosome in every generation. Mutation is also used to maintain diversity in the population and to make sure that the achieved solution is not a local optima. In mutation, the whole chromosome bits are swapped as if the chromosome is read from right to left as follows:

original chromosome: [1 0 1 1 1 0 0 1 1]
mutated chromosome: [1 1 0 0 1 1 1 0 1]

D. Fitness Function

Every chromosome i.e. individual is associated with a fitness function calculated using Algorithm 2.

Algorithm 2. Fitness Function Calculation

Given a chromosome that represents a binary block assignment matrix, perform the following:
$\forall$ node i $\in$ the system

1. Repeat
2. Let i =1 be the failed node
3. $\forall$ block j belongs to node i
4. Repeat
5. Check for all nodes α such that node α has a replica of block j
6. Select the node α with minimum retrieval cost $c_{\alpha j}$
7. Assign α as one of the helper nodes for failed node i and save it in recovery plan matrix
8. Update node i retrieval cost value to be $\sum_{j=1}^{\theta} \underbrace{min}_{\alpha \neq i} c_{\alpha j} x_{ij} . x_{\alpha j}$
9. Update recovery plan to include helper nodes for recovering all blocks j of node i
10. End of inner loop
11. Update total retrieval cost value for all nodes to be $\sum_{\substack{i=1 \\ \alpha \neq i}}^{n} \sum_{j=1}^{\theta} \underbrace{min}_{\alpha \neq i} c_{\alpha j} x_{ij} . x_{\alpha j}$
12. Update system recovery plan for next failure scenario
13. i=i+1
14. End of outer loop when i exceeds n

Algorithm 3 accounts for new-comer blocks and generates a post-optimal recovery plan with the best possible distribution of new-comer blocks on the system nodes.

Algorithm 3. New-comer Blocks Allocation

Given a chromosome that represents an optimal binary block assignment matrix, perform the following:
$\forall$ newcomer block j

1. Replicate block j , ρ times
2. Distribute replicas of j by augmenting the optimal block assignment matrix that was generated based on best fitness _ using Algorithm 2
3. Calculate the fitness of the newly augmented column
4. Update the value of the optimal fitness; i.e. cost generated by Algorithm 2
5. Select chromosomes with best fitness
6. Apply cross-over and mutation operations only on augmented columns of chromosomes with best fitness
7. After a specific number of generations, select the optimal chromosome and update the original optimal block allocation schema together with the optimal recovery plan to account for the new-comers
8. Repeat steps 1-7 for all new-comer blocks

Our heuristic can get an estimation of the minimum system repair cost within few minutes when we have large network sizes together with a table-based full recovery plan.

VI. Simulation Results

In this section, we present the results of our implementation for different cases of single node failures. The machine employed for simulation is a Lenovo laptop with an Intel (R) Core i7 CPU running at 2 GHz with 8 GB RAM. The operating system is Windows 7, and the computer is a 64-bit machine. The elapsed time was 1.845 seconds for Figure 4 results and 34 seconds for Figure 5 results. The simulation programs were written in MATLAB.

To gain more understanding about the performance, we vary the parameters n, θ and ρ. We then increase the network size and measure its running time. Figure 4 shows the convergence curve of the optimal repair cost together with the average cost for each generation given n=10 nodes, θ = 50 blocks and replication factor ρ = 4. Figure 5 shows the curve for values of n= 50 nodes, θ = 125 blocks and replication factor ρ = 2.

As observed from Fig. 4 and Fig. 5, the average fitness of individuals decreases in each generation. This illustrates the fact that better solutions are being generated in newer populations. It is also clear that the optimal fitness, i.e., optimal system repair cost decreases till it converges to a minimum optimal value. We can make sure that the final

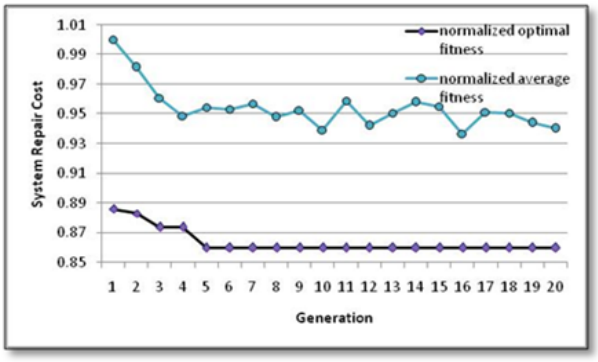

Figure 4. Average and minimum system repair cost for n=10, θ = 50, ρ = 4

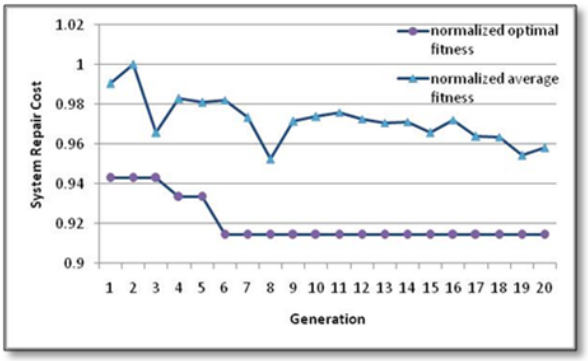

Figure 5. Average and minimum system repair cost for n=50, θ = 125, ρ = 2

minimum cost value is not a local optima by increasing the mutation rate and re-running the algorithm.

Moreover, it is worth to state that the optimal solution is achieved at the fifth generation for n = 10 nodes (Fig. 4) and at the sixth generation when the number of nodes is increased to 50 nodes(Fig. 5). That shows the quick convergence of the algorithm even when we have large number of nodes.

We next compare the minimum system repair cost of our heuristic implementation for the case of new-comer blocks to that of the optimal brute force implementation for different network sizes and this is shown in Fig. 6. Our results are proven to be near optimal since the difference between the heuristic solution and optimal solution is calculated at most 1.2%. The simulation for each value of n is averaged over 20 runs and the post-optimal cost after distribution of new-comer blocks is normalized by the maximum average cost for each generation for fair comparison.

VII. Conclusion

With the emergence of many erasure coding techniques that help provide reliability in practical distributed systems composed of unreliable components, we selected the frac-

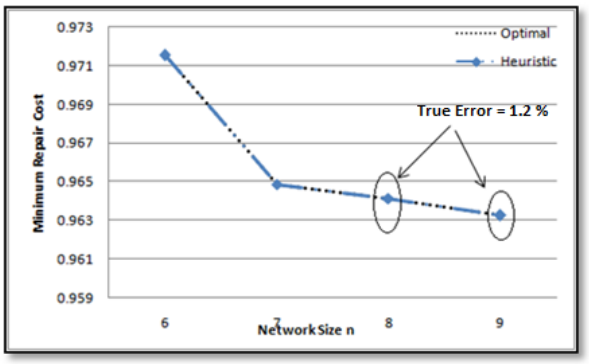

Figure 6. Minimum post-optimal system repair cost for different network sizes for the case of newcomer blocks

tional repetition coding scheme to implement as a code in designing an optimum block allocation method that minimizes system repair cost. A key property of the FR code is that it has a simple repair mechanism that minimizes the repair and disk access bandwidth together with the property of un-coded repair process. To minimize the system repair cost we formulated a problem using incidence matrices and solved it heuristically using a genetic algorithm for all possible scenarios of single node failures. We then presented and solved a practical variation of the main problem that accounts for new-comer blocks. A storage allocation matrix for the new-comers is computed in a minimal computation way that can be easily implemented in reality without the need to redistribute original on-node blocks. The proposed heuristic is shown to provide very close performance to the optimal solution.

REFERENCES

[1] D. Beaver, S. Kumar, H. Li, J. Sobel, and P. Vajgel, "Finding a Needle in Haystack: Facebook's Photo Storage," *Proc. of the 9th USENIX Conf. on Operating Systems Design and Implementation (OSDI'10),* USENIX Association, Berkeley, CA, USA, vol. 10, pp. 1-8, 2010.

[2] A. Carroll, "Why Data Centers are Necessary for Enterprise Businesses," 2013, doi: http://www.lifelinedatacenters.com/data-center/why-data-centers-are-necessary-for-enterprise-businesses/

[3] B. Lavalle, "Proliferation of Remote Data Centers Creates New Networking Challenges," 2015, doi: http://www.datacenterknowledge.com/archives/2015/05/06/proliferation-remote-data-centers-creates-new-networking-challenges/

[4] R. Miller, "Failure Rates in Google Data Centers," 2008, doi: http://www.datacenterknowledge.com/archives/2008/05/30/failure-rates-in-google-data-centers/

[5] R. Harris, "Facebook's Advanced Erasure Codes," 2013, doi: http://storagemojo.com/2013/06/21/facebooks-advanced-erasure-codes/

[6] J.S. Plank and M. Blaum, "Sector-Disk (SD) Erasure Codes for Mixed Failure Modes in RAID Systems," *ACM Trans. on Storage (TOS),* vol. 10, no. 1 , pp. 4, Jan. 2014.

[7] S. ElRouayheb and K. Ramchandran, "Fractional Repetition Codes for Repair in Distributed Storage Systems," *48th IEEE Annual Allerton Conf. on Communication, Control and Computing,* 2010.

[8] O. Khan, R.C. Burns, J.S. Plank, W. Pierce and C. Huang, "Rethinking Erasure Codes for Cloud File Systems: Minimizing I/O for Recovery and Degraded Reads," *FAST Proc.,* pp. 20, Feb. 2012.

[9] M. Li and P.P. Lee, "STAIR Codes: A General Family of Erasure Codes for Tolerating Device and Sector Failures in Practical Storage Systems," *ACM Trans. on Storage (TOS),* vol. 10, no. 4, pp.14, Oct. 2014.

[10] D.S. Papailiopoulos, J. Lou, A.G. Dimakis, C. Huang and J. Li, "Simple Regenerating Codes: Network Coding for Cloud Storage," *IEEE INFOCOM Proc.,* pp. 2801-2805, March 2012.

[11] A. Dimakis, P. Godfrey, Y. Wu, M. Wainwright, and K. Ramchandran, "Network Coding for Distributed Storage Systems," *IEEE Trans. on Information Theory,* vol. 56, no.9, pp. 4539-4551, 2010.

[12] S. Pawar, N. Noorshams, N. ElRouyaheb and K. Ramchandran,"DRESS Codes for the Storage Cloud: Simple Randomized Constructions," *Proc. of IEEE on Information Theory (ISIT),* pp. 2338-2342, July 2011.

[13] Y. Zhu, P.P. Lee, L. Xiang, Y. Xu, and L. Gao,"A Cost Based Heterogeneous Recovery Scheme for Distributed Storage Systems with RAID-6 Codes," *42nd IEEE/IFIP Intl. Conf. on Dependable Systems and Networks,* pp. 1-12, June 2012.

[14] K.V. Rashmi, N.B. Shah, D. Gu, H. Kuang, D. Borthakur and K. Ramchandran, "A Solution to the Network Challenges of Data Recovery in Erasure Coded Storage Systems: A Study on the Facebook Warehouse Cluster," *UNISEX Hotstorage,* 2013.

[15] Q. Yu,C.W. Sung, and T.H. Chan, "Irregular Fractional Repetition Code Optimization for Heterogeneous Cloud Storage," *IEEE Journal on Selected Areas in Communications,* vol. 32, no. 5, pp. 1048-1060, 2014.

[16] CISCO, "Cisco Data Center Infrastructure," 2011, doi: http://www.cisco.com.

[17] S. Hou, Y. Liu, H. Wen and Y. Chen, "A Self-crossover Genetic Algorithm for Job Shop Scheduling Problem," *IEEE Intl. Conf. on Industrial Engineering and Engineering Management,* pp. 549-554, Dec. 2011.

[18] M. Negnevitsky, "Artificial Intelligence: A Guide to Intelligent Systems," Pearson Education, 2005.