Repair Topology Design for Distributed Storage Systems

Quan Yu
Department of Electronic Engineering
City University of Hong Kong
Email: quanyu2@student.cityu.edu.hk

Chi Wan Sung
Department of Electronic Engineering
City University of Hong Kong
Email: albert.sung@cityu.edu.hk

Terence H. Chan
Institute for Telecommunications Research
University of South Australia
Email: terence.chan@unisa.edu.au

***Abstract*—In a heterogenous networking environment, a new practical distributed storage model is defined by introducing the concepts of repair topology and retrieval sets. How to repair a failed storage node so as to minimize the system repair cost is investigated. It is shown that the repair cost minimization problem can be decomposed into a combinatorial problem and an integer linear programming problem. Moreover, a heuristic algorithm to find suboptimal repair topologies is given.**

I. Introduction

Distributed storage system provides an elegant solution to the ever-increasing demand of large-scale and reliable data storage. In such an architecture, when a storage node fails, a new node, called newcomer, needs to repair the lost data of the failed node by downloading data from surviving storage nodes. To reduce the traffic required in repairing a failed storage node, which is commonly called the repair bandwidth in the literature, Dimakis et al. propose the regenerating codes by combining traditional erasure coding techniques with network coding [1]. In their formulation, a newcomer is able to recover the lost data by randomly connecting to d surviving storage nodes, and a data collector (DC) can reconstruct the data object by downloading data from any k out of the n storage nodes, which is called the maximum distance separable (MDS) property of the distributed storage system. A tradeoff between the storage capacity of each storage node and repair bandwidth per failed node is also identified.

In most of the existing works, it is simply assumed that all storage nodes are identical and fully connected with one another. This assumption may not be valid in practical distributed storage systems. For example, the storage capacity and cost associated with different storage nodes may not be the same. Furthermore, the communication links between each pair of storage nodes may have different characteristics such as bandwidth, communication cost, and transmission rate. It is also possible that some nodes are not directly connected. In such a heterogeneous environment, it is unclear how a distributed storage system should be appropriately designed. As far as we know, there is no general result for this problem setting. Only disparate results can be found in the literature. The storage allocation problem, which focuses on how to allocate a given storage budget over the storage nodes such that the probability of successful recovery is maximized, is studied in [2]. A distributed storage system in which the storage nodes have different download costs is considered in [3]. In a distributed storage system with storage cost, how to allocate storage capacities among the storage nodes so as to minimize the total storage cost is investigated in [4]. In [5], the authors take the bandwidth heterogeneity into account and demonstrate that the tree-structured regeneration topology is an efficient topology to reduce the regeneration time. Under functional repair model, the link costs and the impact of network topology are jointly considered in [6].

In this paper, we consider a more practical distributed storage model, in which the underlying network topology can be arbitrary, the storage capacity and cost of different storage nodes are allowed to be different, and the bandwidth and cost of communication links are heterogenous. We relax the constraints of repair and data reconstruction in the conventional model by introducing the concepts of repair topology and retrieval sets. While our concept of repair topology is essentially the same as the concept of repair table in [7], we distinguish the use of these two terms. We use the term *repair topology* to refer to the structure of an *overlay* network for data repairing, and *repair table* to a data structure for storing the topology information. In the work of [7], the repair topology is restricted to be a regular graph with each vertex having degree d, and the graph is randomly generated. In this paper, we do not restrict the repair topology to be a regular graph. Moreover, we consider a heterogenous underlying network and frame the problem into an optimization framework, with the aim of minimizing system repair cost by properly designing the repair topology. Since the whole problem is hard to solve optimally, we decompose it into a combinatorial problem and an integer linear programming problem, and propose a heuristic algorithm to find suboptimal solutions.

II. The Concepts of Repair Topology and Retrieval Sets

The conventional model assumes that a newcomer is able to replace a failed node by contacting any d surviving storage nodes and a data collector can retrieve the stored data object by downloading data from any k out of the n nodes. In reality, however, the communication costs between the newcomer and

This work was partially supported by a grant from the University Grants Committee of the Hong Kong Special Administrative Region, China (Project No. AoE/E-02/08).

each of the surviving storage nodes are different. Furthermore, the distances and transmission rates between the data collector and each of the n storage nodes varies with the location of the data collector. The d surviving nodes to be contacted by the newcomer and the k storage nodes to be contacted by the data collector need not be arbitrary. It is more reasonable to determine a set of helper nodes, called *helper set*, for each storage node and some subsets of the n storage nodes, called retrieval sets, for the data collector. The collection of helper sets defines the repair topology. Thus, we modify the repair and reconstruction mechanisms based on the concepts of repair topology and retrieval sets. We only require that a newcomer can rebuild a failed node by contacting any d nodes in its helper set and a data collector can reconstruct the data object by contacting any k nodes in any one of the retrieval sets.

There are two modes for storage-node repair. The first one is called functional repair, in which the data stored in the newcomer is not necessarily the same as that in the failed node as long as the MDS property is still maintained after the repair process. The second one is exact repair, in which the data stored in the newcomer is exactly the same as the data in the failed node. In [7], the authors construct a class of exact minimum-bandwidth regenerating (MBR) codes, which is a concatenation of an outer MDS code and an inner Fractional Repetition (FR) code. We call this construction the MDS-FR code. In the process of repairing a failed storage node, the corresponding newcomer directly read the exact data it needs from surviving storage nodes without any processing, which is referred to as the uncoded repair property. The exact and uncoded repair property makes the MDS-FR code an excellent candidate for practical distributed storage systems. We therefore adopt the MDS-FR code in our system design.

Consider a distributed storage system with the following parameters: $n = 6$, and $d = k = 2$. For the conventional model, the corresponding tradeoff curve between storage capacity and repair bandwidth under functional repair is shown in Fig. 1. The points below the tradeoff curve are impossible to achieve by both functional repair and exact repair. Now we consider a new model, which includes the concepts of repair topology and retrieval sets. Assume that the chosen repair topology is a ring with 6 nodes, as shown in Fig. 2(a) (solid lines). The data object is divided into four equal-sized packets, B_1, B_2, B_3, and B_4, which are encoded into six coded packets, F_1, F_2, $\ldots$, F_6 by a $(6, 4)$-MDS code. Each edge in the ring is associated with a coded packet. Each node stores the two packets that are associated with its incident edges, as shown in Fig. 2(a). In this example, a newcomer can rebuild the lost data by connecting to $d = 2$ nodes in its helper set, i.e., its two neighboring nodes in the ring, rather than any d surviving nodes. Suppose node 3 fails. A newcomer can replace it by downloading coded packets F_2 and F_3 from its two helper nodes 2 and 4, respectively, as shown in Fig. 2(b). In this example, there are two retrieval sets, $R_1 = \{1, 3, 5\}$ and $R_2 = \{2, 4, 6\}$. A data collector can reconstruct the data object by connecting to any $k = 2$ storage nodes in any one of the two retrieval sets. After normalizing by the number of

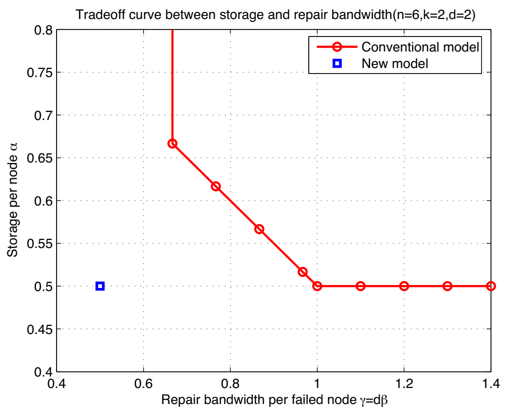

Fig. 1. Tradeoff between storage capacity and repair bandwidth (n=6, d=2, and k=2).

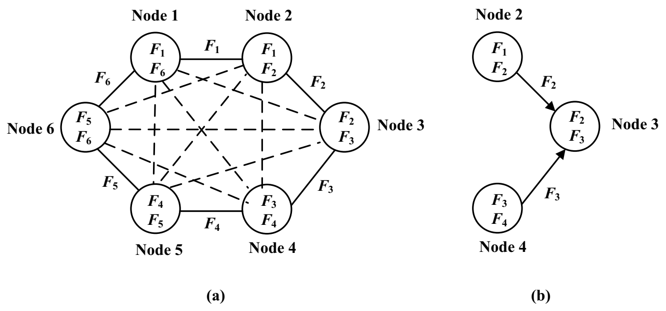

Fig. 2. An example of MDS-FR code construction.

packets of the original data object, the storage amount of each storage node is 0.5 and the repair bandwidth per failed node is also 0.5. This point is also plotted in Fig. 1, which is below the tradeoff curve of the conventional model. From Fig. 1, we can see that with the same storage amount per node, the repair bandwidth is reduced by 50%, which shows that the potential gain can be enormous.

III. A New System Model

In this section, we formally define our new model. Note that it includes the conventional model as a special case.

Consider a distributed storage network, in which n storage nodes (labeled from 1 to n) are distributed across a wide geographical area and connected with a specific topology. A data object with size B is encoded and distributed among the n storage nodes. We model the underlying storage network as a weighted undirected graph $G = (\mathcal{V}, \mathcal{E})$, where vertices represent the storage nodes and edges represent the communication links. Each vertex $i \in \mathcal{V}$ has an associated storage cost s_i indicating the cost of storing one unit of data in node i. We define the storage cost vector $\mathrm{s} \triangleq [s_1, s_2, ..., s_n]$. Besides, each edge $e = (i, j) \in \mathcal{E}$ connecting vertices i and j $(i \neq j)$ has an associated weight $\tilde{c}_{ij}$, called single-hop cost, which represents the cost of transmitting one unit of data along this edge. If there is no direct communication link between two nodes, we let the corresponding single-hop cost be infinite. The cost to transmit one unit of data from node i to j is called the communication cost and is denoted by c_{ij}. The values of c_{ij}'s can be obtained from $\tilde{c}_{ij}$, depending on the underlying communication assumptions. For example, if multi-hop transmissions are allowed, then c_{ij} can be defined as the total cost of the minimum-cost path from i to j. If only single-hop transmissions are allowed, then c_{ij} equals $\tilde{c}_{ij}$ for all i and j. The matrix $\mathbf{C} = [c_{ij}]$ is called the communication cost matrix, or simply the cost matrix. Note that under our assumptions, it is symmetric.

(i) *Repair Mechanism*: We assume that once a storage node fails, a newcomer would immediately join the storage network to replace the failed one. When a newcomer chooses surviving storage nodes from which it would download data to rebuild the lost data, the range of choice is restricted to its helper set. The collection of helper sets of all the n storage nodes defines the repair topology, denoted by τ. Precisely, a repair topology τ corresponds to a directed graph. Its vertex set is the same as that of G. Its edge set is denoted by $\mathcal{E}^\tau$. Node j is in the helper set of node i if and only if $(j,i) \in \mathcal{E}^\tau$. Let the helper set of node i ($i \in \mathcal{V}$) under τ be denoted by H_i^τ ($H_i^\tau \subseteq \mathcal{V}$). When node i fails, we require that the newcomer is able to replace the failed node by downloading data from any d nodes in H_i^τ. If $|H_i^\tau| \leq d$, then it means that the newcomer can contact all nodes in its helper set for repair purpose. Besides, we also require that each newcomer is identical to the corresponding failed node, which indicates that both the connections and stored contents of the newcomer are exactly the same as the failed node. Note that if $H_i^\tau = \mathcal{V}\backslash\{i\}$ for all i, our requirement of the repair process is the same as that of the conventional model. We denote the entire possible repair topologies by $\mathcal{T}$.

(ii) *Reconstruction Mechanism*: We define a collection of retrieval sets, $\Psi \triangleq \{R_1, R_2, \ldots\}$, where $R_j \subseteq \mathcal{V}$. Furthermore, we require that

$$|\Psi| = w \tag{1}$$

and

$$k \leq t_0 \leq |R_j|, \text{ for } j = 1, 2, \ldots, w, \tag{2}$$

where w and t_0 are given constants. A data collector is required to be able to reconstruct the data object by downloading data from any k storage nodes in any one of the retrieval sets. For the case where $w = 1$ and $t_0 = n$, there is one and only one retrieval set, which can only be equal to $\mathcal{V}$. Note that this special case corresponds to the original reconstruction model.

IV. Repair Cost Minimization

To simplify system design, we impose two conditions on the repair topology: (i) $|H_i^\tau| \leq d$ for all i; (ii) $j \in H_i^\tau$ if and only if $i \in H_j^\tau$. The first condition means that there are at most d nodes in the helper set of each node. When a node fails, a newcomer is allowed to contact all nodes in its helper set for regenerating the lost data. The second condition means that the helper relation between two nodes is symmetric, implying that the repair topology can be represented by an undirected graph.

Our code construction can be described as follows. The source node first encodes the original data object by using an (F, B)-MDS code, and then distributes the F encoded equal-sized packets to the storage network. The set of these F coded packets is denoted by $\mathcal{F}$. To illustrate how the F coded packets would be distributed among the n storage nodes, we assume that these coded packets are first assigned to edges contained in a repair topology τ, say $\mathcal{E}^\tau$. For edges in graph G but not contained in repair topology τ, no coded packets would be assigned to them. A coded block $\mathcal{B}_{ij} \subseteq \mathcal{F}$ would be assigned to edge $(i,j) \in \mathcal{E}^\tau$. Therefore, we have

$$|\mathcal{B}_{ij}| \triangleq \beta_{ij} \begin{cases} \geq 0 & \text{if } (i,j) \in \mathcal{E}^\tau, \\ = 0 & \text{otherwise .} \end{cases} \tag{3}$$

We further assume that the coded blocks assigned to two different edges (i,j) and (u,v) in the repair topology do not share any packets in common, i.e., $\mathcal{B}_{ij} \bigcap \mathcal{B}_{uv} = \phi$. We use $\mathbf{B}$ to denote the symmetric matrix $[\beta_{ij}]$, and we call it the block assignment matrix. Note that β_{ij}'s are all integers.

As regard to each storage node, it stores all the coded packets associated with its incident edges. Thus the storage amount of node i under repair topology τ, denoted by α_i^τ, can be obtained as

$$\alpha_i^\tau = \sum_{j \in H_i^\tau} \beta_{ij}, \ \forall i. \tag{4}$$

Moreover, F can be calculated as $F = \sum_{(i,j) \in \mathcal{E}^\tau : i<j} \beta_{ij}$. In summary, if nodes i and j are connected by an edge $(i,j) \in \mathcal{E}^\tau$, they store a common coded block $\mathcal{B}_{ij}$ that associates with the edge (i,j). When node i fails, the newcomer of node i will download the coded block $\mathcal{B}_{ij}$ from node j, for all $j \in H_i^\tau$.

For each retrieval set $R_t \in \Psi$, there are $\binom{|R_t|}{k}$ subsets of cardinality k. Denote the collection of these subsets by $\mathcal{P}_t$. For reconstruction of the data object, it is required that the k nodes in any $P \in \mathcal{P}_t$ jointly store at least B distinct encoded packets in $\mathcal{F}$, for all t. In other words, we have

$$\sum_{i \in P} \alpha_i^\tau - \sum_{i,j \in P} \beta_{ij} \geq B, \ \forall P \in \mathcal{P}_t, \ t = 1, 2, \ldots, w. \tag{5}$$

We further assume that there is a constraint on the system storage cost, denoted by c_s. Let C_s be the maximum system storage cost per unit data object. Then we have

$$c_s(\mathbf{B}, \tau, \Psi) \triangleq \frac{1}{B} \sum_{i=1}^{n} s_i \alpha_i^\tau \leq C_s. \tag{6}$$

Note that C_s is a given constant.

Our objective function is the system repair cost, denoted by c_r. Formally, the repair cost minimization problem can be stated as follows.

$$\text{Minimize} \ \ c_r(\mathbf{B}, \tau, \Psi) \triangleq \frac{1}{B} \sum_{(i,j) \in \mathcal{E}^\tau} c_{ij} \beta_{ij}, \tag{7}$$

subject to (1)–(6), $\tau \in \mathcal{T}$, and β_{ij}'s being integers. In (6) and (7), both the system storage cost, c_s, and system repair cost, c_r, are normalized by the data object size, B. Note that β_{ij}, α_i^τ, τ and Ψ are optimization variables.

In our formulation, it is clear that there is a tradeoff between the storage cost and the repair cost. To make this relationship more explicit, we introduce the following two notions:

Definition 1: A cost pair (c_r^*, c_s^*) is B-achievable by the MDS-FR code if given any data object of size B, there exists a repair topology τ, a collection of retrieval sets Ψ, and a block assignment matrix $\mathbf{B}$ such that $c_r(\mathbf{B}, \tau, \Psi) \le c_r^*$ and $c_s(\mathbf{B}, \tau, \Psi) \le c_s^*$.

Definition 2: A cost pair (c_r^*, c_s^*) is asymptotically achievable by the MDS-FR code if for any $\epsilon > 0$, there exists for sufficiently large B, a repair topology τ, a collection of retrieval sets Ψ, and a block assignment matrix $\mathbf{B}$ such that $c_r(\mathbf{B}, \tau, \Psi) < c_r^* + \epsilon$ and $c_s(\mathbf{B}, \tau, \Psi) < c_s^* + \epsilon$.

The following result shows that the asymptotically achievable cost can be obtained by relaxing the integer constraint on $\mathbf{B}$:

Theorem 1: Given any B and C_s, let $\mathbf{B}^* = [\beta_{ij}^*]$, τ^*, and Ψ^* be the solution to the repair cost minimization after relaxing the integer constraint on $\mathbf{B}$, and $c_r^* \triangleq c_r(\mathbf{B}^*, \tau^*, \Psi^*)$ be the corresponding repair cost. The cost pair (c_r^*, C_s) is asymptotically achievable by the MDS-FR code.

Proof: First of all, note that c_r and c_s are invariant to scaling B and all β_{ij}'s by the same amount, no matter whether β_{ij}'s are integers or not. Suppose we scale up B and β_{ij}'s all by $\gamma > 1$. Then we round *up* all β_{ij}'s to the nearest integers. The new repair cost is then given by

$$\tilde{c}_r = \frac{1}{\gamma B} \sum_{(i,j) \in \mathcal{E}^\tau} c_{ij}(\gamma \beta_{ij} + e_{ij}) \tag{8}$$

$$= \frac{1}{B} \sum_{(i,j) \in \mathcal{E}^\tau} c_{ij}\beta_{ij} + \frac{1}{\gamma B} \sum_{(i,j) \in \mathcal{E}^\tau} c_{ij} e_{ij}, \tag{9}$$

where $0 < e_{ij} < 1$. Since γ can be arbitrarily large, the second term can always be made smaller than ϵ. Similarly, the new storage cost can be proven to be smaller than $C_s + \epsilon$ for sufficiently large γ. Therefore, (c_r^*, C_s) is asymptotically achievable. ■

V. A HEURISTIC SOLUTION

Given a repair topology τ and a collection of retrieval sets Ψ, let $\mathbf{B}^*(\tau, \Psi)$ be the solution obtained by solving the ILP problem. (Alternatively, we can solve the LP problem by relaxing the integer constraint if we want to minimize the asymptotically achievable cost.) Now we have to determine τ and Ψ that minimizes $c_r(\mathbf{B}^*(\tau, \Psi), \tau, \Psi)$, subject to the constraints described in the previous section.

Enumerating all feasible solutions to find the best one is clearly inefficient, since the number of feasible solutions grows exponentially with the network size. To overcome this difficulty, we present a heuristic algorithm. Our idea is to determine the repair topology first. We choose an edge, not previously chosen, with the smallest communication cost at each step while obeying the degree constraint d. This greedy approach can be formally stated as follows:

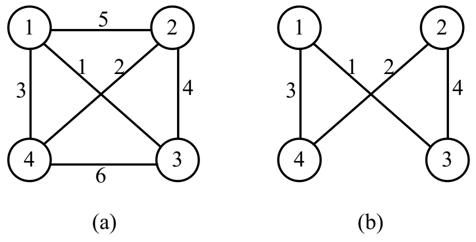

Fig. 3. An example of finding a repair topology τ in given graph G.

Algorithm 1: Let N_i^τ be the degree of node i in the repair topology $\tau = (\mathcal{V}^\tau, \mathcal{E}^\tau)$.

1) Initialize τ with $\mathcal{V}^\tau = \mathcal{V}$, $\mathcal{E}^\tau = \emptyset$, and $N_i^\tau = 0$, for $i = 1, 2, \ldots, n$.
2) Sort the edges of $\mathcal{E}$ in ascending order of communication cost and get the sequence $e_1, e_2, \ldots, e_m$, where $c_{e_1} \le c_{e_2} \le \cdots \le c_{e_m}$. Denote the two end points of e_i by u_i and v_i, for $i = 1, 2, \ldots, m$.
3) For $i = 1$ to m, if both $N_{u_i}^\tau$ and $N_{v_i}^\tau$ are strictly less than d, then add e_i to $\mathcal{E}^\tau$, and increase both $N_{u_i}^\tau$ and $N_{v_i}^\tau$ by 1.
4) Output $\tau = (\mathcal{V}, \mathcal{E}^\tau)$.

The complexity of Algorithm 1 is $O(m \log m)$, since the sorting in the second step has complexity of $O(m \log m)$ and the third step has complexity of $O(m)$.

After determining τ, we need to find Ψ. According to the structure of the MDS-FR code, we know that the coded blocks stored in two vertices that are not neighbors are all distinct. For this reason, we require the retrieval sets to be independent sets of τ, as far as possible. (Recall that an independent set of a graph refers to a set of vertices, no two of which are adjacent.) In general, there are many possible independent sets. We may simply randomly picked w of them, each of which has cardinality t_0. If the cardinality constraint cannot be satisfied, we may arbitrarily add some vertices to those independent sets. (In that case, the resulting retrieval sets are no longer independent sets.)

Example: Consider a 4-node fully connected graph G shown in Fig. 3(a). The numbers associated with the edges denote the communication costs. The vertex set $\mathcal{V} = \{1, 2, 3, 4\}$ and edge set $\mathcal{E} = \{e_1 = (1,3), e_2 = (2,4), e_3 = (1,4), e_4 = (2,3), e_5 = (1,2), e_6 = (3,4)\}$. Suppose the degree constraint is $d = 2$. According to Algorithm 1, the edges e_1, e_2, e_3, and e_4 are successively added into $\mathcal{E}^\tau$. The resulting subgraph τ of G is shown in Fig. 3(b). Furthermore, suppose that $w = t_0 = 2$. We may choose $R_1 = \{1, 2\}$ and $R_2 = \{3, 4\}$, both of which are independent sets of τ.

VI. SIMULATION RESULTS

In this section, we compare the minimum system repair cost of our model with that of the conventional model for different network size. In our simulations, both $\mathbf{s} = [s_i]$ and $\mathbf{C} = [c_{ij}]$ are randomly generated, in which each entry is selected from the uniform distribution on the interval $(0, 50)$. The simulation for each value of n is averaged over 100 runs.

Note that all the simulation results are normalized by the data object size, B. Given a distributed storage network, a repair topology τ is selected by Algorithm 1. Since we want to compare the performance of our model with that of the conventional model, the underlying network is assumed to be fully connected. According to the input values of the number of retrieval sets, w, and the cardinality of each retrieval set, t_0, the retrieval sets are randomly picked from the independent sets of τ. Note that if $w = 1$ and $t_0 = n$, there is only one retrieval set that contains all the n storage nodes. This special case corresponds to the original reconstruction model.

In our model, to illustrate the integrality gap, we consider a distributed storage system with parameters: $n = 10, d = 6, k = 4$, $w = 1$, and $t_0 = n$. We increase the data object size B from 10 to 30, with step size 10. For each value of B, we increase the maximum system storage cost per unit data object, C_s, from 80 to 100 and solve the corresponding ILP. The tradeoff curves between the system storage cost, c_s, and the minimum system repair cost, c_r^*, are plotted in Fig. 4. We can observe that the gap between the solution of the ILP and of its relaxation is tiny and decreases with the growing of the data object size B. Thus, to improve the efficiency of simulation, we solve the LP problem by relaxing the integer constraints on $\mathbf{B}$ to minimize the asymptotically achievable repair cost in our simulation.

We next compare the minimum system repair cost of our model with that of the conventional model for different network size. For the conventional model, if each newcomer downloads the minimum amount of data $\beta^* = \frac{2B}{2kd-k^2+k}$ each via the d links with the smallest communication costs, the minimum system repair cost can be achieved. The corresponding system storage cost per unit data object is at least $\frac{2d}{2kd-k^2+k}\sum_{i=1}^{n} s_i$. For our model, C_s is set to $\frac{2d}{2kd-k^2+k}\sum_{i=1}^{n} s_i$, which is the minimum storage cost of the conventional model when the minimum system repair cost is achieved. From Fig. 5, it can be seen that with the same system parameters, the asymptotically achievable minimum repair cost of our model is roughly reduced at least by 50%. Since the constraints of repair and reconstruction are relaxed in our model, it is not surprising that there exists a performance gain. Nevertheless, it demonstrates that there is a large room for improvement if the conventional model is refined. This is particularly relevant when the networking environment is heterogenous.

VII. Conclusion

In this paper, based on the practical heterogenous networking environment, we propose a new distributed storage model by introducing the concepts of repair topology and retrieval sets. In particular, we focus on the case where immediate repair is performed when a node fails and formulate it as a system repair cost minimization problem. It shows that the repair cost minimization problem can be decomposed into a combinatorial problem and an integer linear programming problem and can be solved in tandem. Moreover, the simulation results demonstrate that there exists an enormous system performance gain if the conventional model is refined.

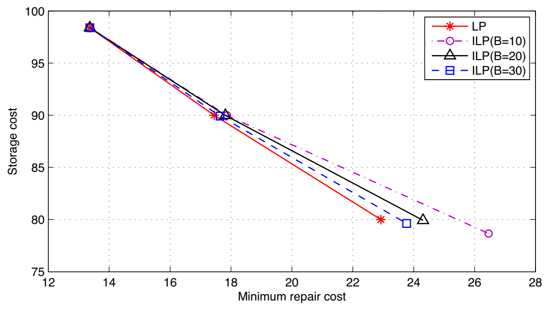

Fig. 4. Tradeoff curves between storage cost and minimum system repair cost ($n = 10, d = 6, k = 4, w = 1$, and $t_0 = n$).

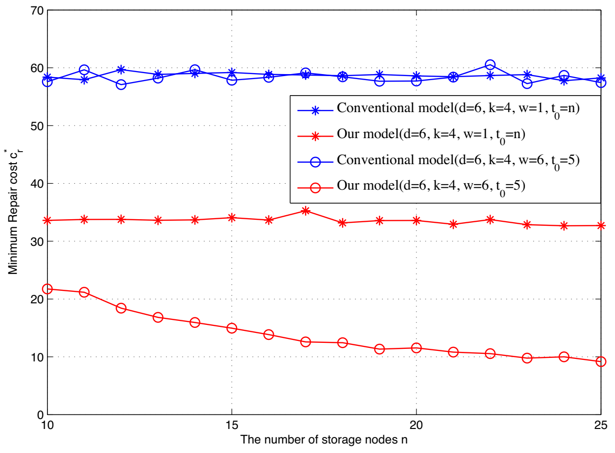

Fig. 5. The minimum system repair cost for different network size.

References

[1] A. G. Dimakis, P. B. Godfrey, Y. Wu, M. J. Wainwright, and K. Ramchandran, “Network coding for distributed storage system,” in *Proc. IEEE Int. Conf. on Computer Commun. (INFOCOM ’07)*, Anchorage, Alaska, May 2007.

[2] D. Leong, A. G. Dimakis, and T. Ho, “Distributed storage allocations,” Nov. 2010, arXiv:1011.5287 [cs.IT].

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

[4] Q. Yu, K. W. Shum, and C. W. Sung, “Minimization of storage cost in distributed storage systems with repair consideration,” arXiv:1107.5645v1 [cs.IT], to appear in *Proc. IEEE GLOBECOM*, Houston, Texas, Dec. 2011.

[5] J. Li, S. Yang, X. Wang, and B. Li, “Tree-structured data regeneration in distributed storage systems with regenerating codes,” in *Proc. IEEE INFOCOM*, San Diego, Mar. 2010, pp. 1–9.

[6] M. Gerami, M. Xiao, and M. Skoglund, “Optimal-cost repair in multi-hop distributed storage systems,” in *Proc. IEEE Int. Symp. Inf. Theory*, St. Pertersburg, Jul. 2011.

[7] S. El Rouayheb and K. Ramchandran, “Fractional repetition codes for repair in distributed storage systems,” in *Allerton conference on commun. control and computing*, Monticello, Sep. 2010.