Computing and Informatics, Vol. 36, 2017, 1235–1260, doi: 10.4149/cai_2017_5_1235

ON THE COMMUNICATION COST OF MDS ERASURE CODES IN DISTRIBUTED STORAGE SYSTEMS

Elif Haytaoglu

Department of Computer Engineering
Pamukkale University 20160, Denizli, Turkey
e-mail: eacar@pau.edu.tr

Mehmet Emin Dalkilic

International Computer Institute
Ege University
35100, Izmir, Turkey
e-mail: mehmet.emin.dalkilic@ege.edu.tr

Abstract. Distributed storage systems store some redundant data to keep the degree of availability of the stored data constant and also to increase the system's resistance against failures. This type of systems usually use pure replication or methods based on RAID systems as redundancy schemes. In this paper, we study the communication cost of a distributed data storage system using Maximum Distance Separable (MDS) erasure codes. Our focus is reduction of the cost of one-to-many communication used in data reconstruction/repair initialization and update operations. We propose the use of two different communication approaches on the area of distributed storage systems for the above operations; Steiner tree approach and multi-shortest path approach. We also analyse these two communication approaches empirically and theoretically. Our theoretical results indicate that Steiner tree approach has lower message usage, whereas, multi-shortest path approach has lower time usage for data reconstruction/repair initialization operations. On the other hand, Steiner tree approach has better message and time metrics for the data update process. Furthermore, our experimental results support these theoretical results. Thus, users can choose between the two approaches depending on their needs and priorities.

Keywords: Distributed storage systems, MDS codes, minimum Steiner tree, shortest paths, communication cost

Mathematics Subject Classification 2010: 68P20

1 INTRODUCTION

According to IDC, the magnitude of digital data is estimated to grow by a factor of 300 from 2005 to 2020 [1]. Thus, it is crucial to increase the capacity of data storage, especially for organizations heavily interacting with their users.

There are lots of distributed storage systems such as OceanStore, PAST, CFS, TotalRecall, FarSite and Ivy [2, 3, 4, 5, 6, 7]. However, storage servers which constitutes these systems have the risk of breakdown by a variety of reasons. Therefore, distributed storage systems usually store redundant data for better reliability. One way to provide reliability is the use of erasure codes.

Pure replication scheme is generally used for redundancy. An alternative scheme is erasure coding. In this scheme, data is expanded using a generator matrix and then this expanded data is distributed to storage servers. If a failure occurs in any of these servers, the related parts of data must be gathered from the running servers.

The network topology directly and significantly changes the performance of MDS based distributed systems by affecting the three processes: data reconstruction, node repair and data update. The common ground of these three processes is that all three may need to use the multicast communication model. In this paper, we analysed how the communication cost of the distributed storage systems for data reconstruction/repair initialization and update processes is affected by, either the usage of the multicast tree produced by the approximation algorithm of minimum Steiner tree [8] or the multi-shortest path.

This paper is an extended version of a proceedings paper that appeared as [9]. The theoretical analyses of the proposed approaches in terms of the time usage and the message usage are the key additions of this journal version. Furthermore, we have rerun the simulations improved with more realistic parameters. We have also included new simulations for data update communication to show the influences of message size on the approaches.

The remainder of paper is organized as follows. Related work is discussed in Section 2, definitions and the system model are described in Section 3, the proposed methods are explained in Section 4. In Section 5, the communication of data reconstruction and repair initialization due to the two different approaches are analysed theoretically. Furthermore, in Section 6, the theoretical analysis of the data update communication is given according to the two proposed approaches. Finally, the simulation results are given in Section 7.

2 RELATED WORK

There are lots of algorithms introducing approximation algorithms for minimum Steiner tree problem [8, 10, 11, 12]. Amongst these algorithms, Dolagh and Moazzami's algorithm is more practical and comprehensible in comparison to the others with an experimental approximation ratio of 1.5.

Various studies exist in the literature about reducing the node repair cost. Li et al. investigate the node repair time and the probability of node repair using a tree based communication scheme [13]. In that work, (n, k) linear network coding is used as a coding scheme. Helper nodes forward their coded blocks to the newcomer node by encoding other helper nodes' coded blocks through utilizing linear network coding. The communication cost of the data update process or data reconstruction operations are not addressed in that paper. Li et al. also proposed a new fast node repair scheme for distributed storage systems using regenerating codes by taking network topology into account [14]. They propose a new tree structured regeneration topology on a network consisting of all storage nodes. They decreased the cost of the convergecast communication in node repair in terms of time metric by providing encoding of some helper nodes' encoded data by other helper nodes. They mainly focused on reducing the time used in node repair by utilizing links with high bandwidth capacity. However, they did not consider the communication cost of update operation or control messages of data reconstruction or control messages of node repair as well.

In 2011, Gerami et al. studied the cost of node repair process of a multi-hop distributed storage system using regenerating codes by taking the impact of the network topology into consideration [15]. The cooperation of helper nodes through encoding of other helper nodes' packets was also proposed in that work. They showed the results of the proposed methods in tandem, star and grid networks. They formulate the cost of node repair problem based on some constraints which are specific to regenerating codes. Data update or data reconstruction operations are not discussed in that work.

The reduction of the communication cost through host selection in the distributed context data storage is studied in [16].

There are also some studies about data update process in MDS codes. In 2005, Aguilera et al. considered linear systematic MDS codes under concurrent updates and derived tight bounds for the erasure recoverability of an (n, k) MDS code [17].

Anthapadmanaphan et al. considered frequently changing data. In this concept, they propose a class of random codes with logarithmic complexity [18]. But, the issue of the affect of the network topology is not discussed in their study.

Plank issued a report about the explanation of how Reed Solomon codes used to store data safely [19]. He explained how the data is encoded, updated and how the system is recovered to the stable state if any failures occur. However, the issue of the affect of the network topology on data update is not discussed in his work.

Although, some previous works consider the minimum Steiner tree approach for solving multicast communication, they either transform it into a linear programming

problem or advice avoiding the use of it because it is NP-complete [20]. Most of the related work focuses on the communication cost of the node repair process rather than update process. Furthermore, to the best of our knowledge, the cost of initialization of node repair and data reconstruction processes have not been considered.

3 DEFINITIONS AND THE SYSTEM MODEL

In this section, minimum Steiner tree problem, erasure codes and the system model are explained, respectively.

3.1 Minimum Steiner Tree Problem

One of the well-known problems of graph theory is the minimum Steiner tree problem which is also an NP-complete problem [20]. This problem is as follows: Let $G(V, E)$ be a connected graph and S be a set of vertices where $S \subseteq V$ and a cost function $c(v_i, v_j)$ exists over all edges in E where $v_i \in V$ and $v_j \in V$. The cost of a graph is defined as the total weights of all edges in E. The minimum Steiner tree problem on a graph $G(V, E)$ and a given set $S \subseteq V$ is to find a minimum cost tree $T(V^{'}, E^{'})$ that contains all vertices of S and satisfies the conditions $S \subseteq V^{'} \subseteq V$ and $E^{'} \subseteq E$ [21].

3.2 Erasure Codes For Distributed Storage Systems

A typical distributed storage system using erasure codes can be explained as follows. In the classical (n, k) erasure coding based on [22], original data is divided into k fragments. Then, this data is multiplied by an $(n \times k)$ matrix (generator matrix). Thus, the generated data forms an $(n \times 1)$ matrix which means that the original $(k \times 1)$ data is expanded to $(n \times 1)$. Next, fragments, each row of the generated data, are distributed to n different nodes. The distributed system constructed this way can tolerate up to $(n - k)$ failures.

If the original data needs to be reconstructed, a node should gather data fragments from any k of n nodes. After this node gets k fragments, it decodes them with the reverse of the generator matrix which has rows only related to the received fragments. After that, the resulting matrix constitutes the original data. When a failure occurs in this distributed system, the lost data should be repaired to keep system up. A node aware of the failure should gather any k of the fragments from other nodes and decode them. After reconstructing the original data, the node encodes it and sends the lost part of the generated data to a new node.

Update operation is performed in MDS codes as follows: if data needs to be updated, initially the difference between the new value and the old one is calculated (Δ value) by applying subtraction over Galois Field, then the result is multicast to the related nodes ($n-k$ nodes). After receiving Δ, these nodes multiply Δ with the related part of the generator matrix and add this value to the old value over Galois Field.

3.3 The System Model

A computer network which constitutes a distributed storage system can be represented as a graph having vertices that symbolize the computers (nodes), edges that represent the links between computers; link weights indicating the communication costs between computers. In this graph, some specific nodes are defined as storage nodes and the remaining nodes are defined as clients which use (load, request, update data) the system. We assume that the system has n storage nodes storing equal amount of data and any k of these storage nodes are sufficient for data reconstruction/repair. The graph has N nodes in total which means we have $N-n$ client nodes ($k < n \leq N$). User nodes can request or update the data through related storage nodes. In addition, we assume that no node failures or abandons occur during the data reconstruction, node repair or data update operations are employed.

In this system we can repair, reconstruct and update the data using linear MDS code features.

4 PROPOSED METHODS

We propose the usage of two new communication approaches in the area of distributed storage systems for data reconstruction/repair initialization and data update operations. To that end, we investigated the communication phases of these operations and we elaborated the proposed two communication approaches for the initialization of data reconstruction and node repair in the following two subsections.

4.1 Data Reconstruction and Repair Initialization

We examined two approaches for one-to-many communication of the data reconstruction and repair initialization processes. The first one uses a minimum Steiner tree approximation algorithm. The second approach uses Dijkstra's shortest path algorithm. The requirements of the data reconstruction initialization and the repair initialization are the same. So, proposed approaches are equally applicable to both processes. Legends in the figures are described in Table 1.

4.1.1 Steiner Tree Approach (STA)

Since the computers in the system are divided into two different types (client node, storage node), reducing the cost of the multicast communication is somehow aligned with finding the minimum Steiner tree. So, to reduce the communication cost between storage nodes, the client nodes can be included in the multicast session occurring in data reconstruction process.

When a storage node (called initiator) receives a data reconstruction/repair request, it should send data reconstruction/repair request message to the sufficient number of the storage nodes in order to get necessary fragments which would be

Legends	Meaning
	Storage Node
	Data Requesting Client
	Client Node
	Message
	Storage Node not in selected k nodes
	Message between nodes 3 and 17
	Message between nodes 3 and 6
	Message between nodes 3 and 15

Table 1. Legends in the figures

used in the decoding phase. This kind of communication is appropriate for multicasting, since the same message should be transmitted to multiple network nodes. In addition, the multicast tree required in a multicast session can be generated by the approximation algorithm for minimum Steiner tree in [8]. We name this method as Steiner tree approach (STA).

In STA, when a storage node receives data reconstruction/repair request from any client, firstly, the storage node selects k random storage nodes for multicast session, then it constructs a Steiner tree (if there is none) between itself and selected nodes using the algorithm in [8]. At the next step, the initiator sends data reconstruction request message to its neighbors in the constructed Steiner tree. When a node in Steiner tree receives reconstruction/repair request, it forwards this message to its neighbors in the tree. If this node is also a storage node, it sends relevant fragments to the initiator. The Steiner tree construction process should take place only after requesting data for the first time or any node's failure in the already constructed tree. Thus, if the tree is constructed from scratch, relevant tree information should be sent to the tree members. Otherwise, the initiator can use the existing Steiner tree. A sample scenario using this approach is given in Figure 1. In Figure 1 a) client node 5 sends data reconstruction/repair request to node 3, then node 3 generates Steiner tree and multicasts this request to the target storage nodes in the constructed tree b), in c) storage nodes send their relevant fragments to the initiator node 3, finally in d) node 3 decodes data and sends the requested data to node 5 (data reconstruction/repair initialization process includes only a) and b)).

4.1.2 Multi-Shortest Path Approach (MPA)

The second way for data reconstruction/repair initialization process is using multi-unicast model which means sending request messages to targets separately. This model can be implemented optimally using Dijkstra's shortest path algorithm for each path between initiator and targets [23]. We name this approach as multi-shortest path approach (MPA). Unless there exists a specific multicast model over the group of nodes which does not belong to the same subnet, multi-unicast commu-

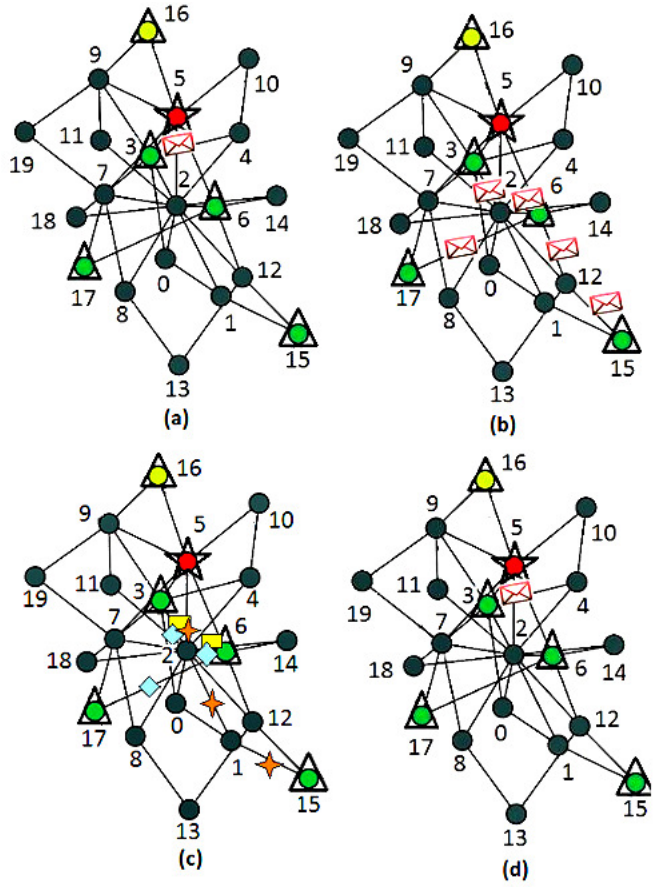

Figure 1. Data reconstruction/repair operation using STA

nication model is used for one-to-many communication. Thus, if no multicast model is specified for multicasting, a storage node (the initiator) should send reconstruction/repair request message one by one. Upon receiving a reconstruction/repair request, the initiator sends this request message to the randomly selected k storage nodes through the shortest paths. If a node on the shortest path receives a request, it simply forwards this message to the next edge in the path. In the case of this data request's first arrival to the initiator, initiator node should generate shortest paths to target storage nodes and should send the path information to nodes in the shortest paths. A sample scenario using this approach is given in Figure 2: a) client node 5 wants data reconstruction/repair from node 3, node 3 generates shortest paths using Dijkstra's algorithm (if paths have not been generated already) after receiving the request and then sends this request to storage nodes one-by-one in b),

storage nodes send their relevant fragments to the initiator node 3 in c), then node 3 decodes data and sends requested data to node 5 in d) (data reconstruction/repair initialization process includes only a) and b)).

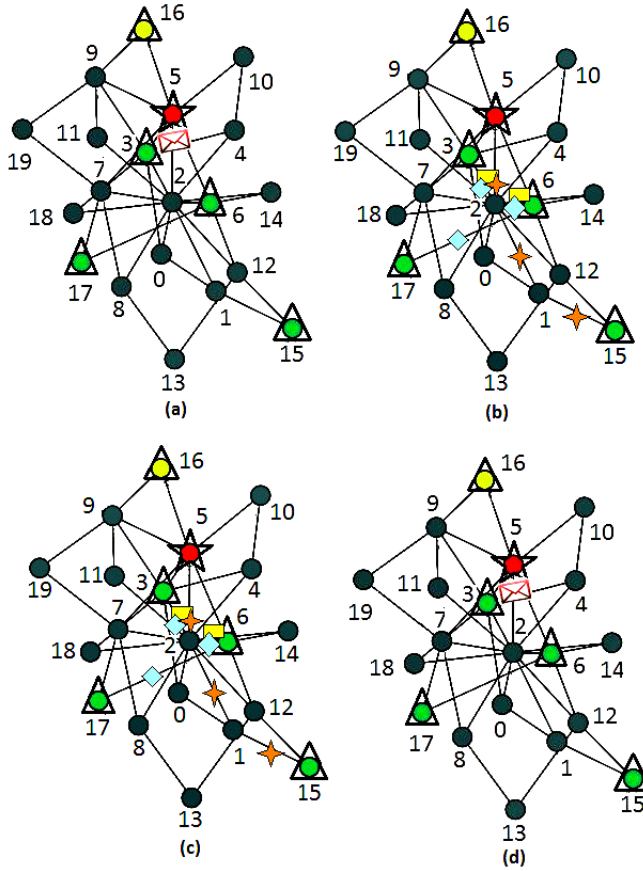

Figure 2. Data reconstruction/repair operation using MPA

4.2 Data Update Process

Let $D = \{D_1, D_2, \ldots, D_k\}$ be the original data vector and $C = \{C_1, C_2, \ldots, C_{n-k}\}$ be the vector containing encoded version of D (each element of D and C is stored at different node). We want to update D_i to $D_i^{'}$. The value of C_j ($j \in \{1, \ldots, n-k\}$) containing encoded version of D_i, is affected by D_i's change ($i \in \{1, \ldots, k\}$). Suppose $\Delta = D_i^{'} - D_i$; the new value for $C_j, C_j^{'}$, is calculated as $C_j^{'} = C_j - G_{ji}\Delta$ where G_{ji} is the element in generator matrix's j^{th} row i^{th} column and all arithmetic

operations performed over Galois Field. In other words, to update, initially Δ value is calculated, then this Δ value is sent to the nodes storing C_j for all j. Then each node j calculates its affected value with $G_{ji}\Delta$. Since Δ value should be sent to the affected storage nodes, this scheme involves one-to-many communication. A detailed explanation of update process in MDS codes can be found in [19]. A client can update data by sending a request to a storage node which is referred as the source node.

4.2.1 Steiner Tree Approach (STA)

We create a Steiner tree on the graph including affected storage nodes ($(n-k)$ nodes) and some other nodes as the pathway nodes using the algorithm proposed in [8]. Suppose that the updated value is in i^{th} element of D. The source node calculates Δ value by subtracting data's current value from desired new value using Galois Field arithmetic. The source node sends Δ value to all target storage nodes by multicasting Δ over the constructed Steiner tree. This means only nodes in the tree receive the value. If Δ value's size is larger than the MTU (Maximum Transmission Unit), this value is fragmented and then each packet is multicast to all target nodes separately. Since the target storage node j receiving Δ also stores the relevant rows of the generator matrix (G_j), node j updates its fragments by subtracting $G_{ji}\Delta$ from its fragments.

4.2.2 Multi-Shortest Path Approach (MPA)

When a storage node receives an update request for D_i, it calculates Δ by $\Delta = D_i^{'} - D_i$. If Δ value is larger than MTU, it should be fragmented in packets. Then the source node sends these packets to all target storage nodes one by one using the shortest paths (if these paths are not generated before, source node generates them using Dijkstra's shortest path algorithm). Since each node j calculates its affected value using G_{ji}, received Δ is same for all target nodes. In MPA, there is no network overlay constructed for multicast, this Δ is sent to all target nodes one by one whereas in STA a tree is constructed for multicast operations. After receiving all packets, a target storage node j calculates the new value as: $C_j^{'} = C_j - G_{ji}\Delta$. After all target nodes perform this process, the update operation is completed.

5 THEORETICAL ANALYSIS OF DATA RECONSTRUCTION AND REPAIR INITIALIZATION

We show the theoretical complexity analyses of Steiner tree and multi-shortest path approaches in data reconstruction and repair initialization processes. In Table 2, we define some legends that we use in the analyses (also for Section 6). The required communication schemes are the same for both of the data reconstruction initialization and repair initialization processes. Therefore, the analyses in this section are for both of them.

Variable	Meaning
d	The propagation delay time of each link
t	The transmission time for one packet (one packet size is smaller than or equal to the MTU size)
N	Total node count in the graph
k	Target storage node count which is sufficient for data reconstruction/repair
n	The number of storage nodes
δ	The average node degree in the graph
L_k	The number of links (edges) in a multicast tree having k target nodes

Table 2. Legends in the theoretical analyses

5.1 Steiner Tree Approach

The message complexity and the time complexity analyses of data reconstruction and repair initialization processes using Steiner tree approach are explained below.

5.1.1 Average Case Message Complexity

The message complexity of the proposed scheme depends on the structure of Steiner tree produced by algorithm in [8].

Lemma 1. In the graph having N nodes, the mean distance between any two nodes in terms of link count (also mean unicast distance between any two nodes), $E(L_u)$, is almost of order $\Theta(\log(N)/\log(\tilde{d}))$. Here $\tilde{d}$ is the weighted average of the sum of squares of the expected degrees [24]. (For simplicity, we assume that this condition is true in the following analyses.)

Lemma 2. The expected link count of a multicast tree having one source node and m target nodes, $E(L_m)$ is upper bounded by $O(E(L_u)m^{0.8})$ according to the Chuang-Sirbu law [25] where $E(L_u)$ denotes the mean number of links in a unicast path.

Theorem 1. The average message count required in Steiner tree approach in the initialization of data reconstruction and node repair processes for one multicast session is upper bounded by $O\left(k^{0.8}\frac{\log(N)}{\log(\tilde{d})}\right)$.

Proof. In this case, the total message count is equal to the number of links of Steiner tree constructed by the algorithm in [8]. The approximation algorithms for solving minimum Steiner tree problem are designed to find a tree having approximately the minimum cost. Therefore, the trees constructed by these algorithms generally have less cost than that of the randomly generated multicast trees. Since, the mean distance between two nodes is order of $\frac{\log(N)}{\log(\tilde{d})}$ by Lemma 1 and the average ratio of $E(L_{m=k})$ over $E(L_u)$ is order of $k^{0.8}$ by Lemma 2, the mean message count required in Steiner tree approach is upper bounded by $O\left(k^{0.8}\frac{\log(N)}{\log(\tilde{d})}\right)$. □

5.1.2 Average Case Time Complexity

The elapsed time for a multicast session is determined by the time interval that begins when the source (initiator) node sends the initialization message and ends at the latest arrival time of the message. The mean value of the multicast tree's height should be found to calculate the expected elapsed time for Steiner tree approach.

Lemma 3. The expected height of Steiner tree constructed by the algorithm in [8] (containing k target nodes on a graph containing N nodes in total) is upper bounded by $O\left(\frac{k\log(N)}{\log(\bar{d})f(k)}\right)$ where $f(k)$ is a monotonically increasing function of k.

See proof of Lemma 3 in the Appendix A.

Theorem 2. The expected elapsed time for Steiner tree approach is upper bounded by $O\left(k\frac{\log(N)}{\log(\bar{d})f(k)}(t+d)\right)$ where $f(k)$ is a monotonically increasing function of k.

Proof. The average height of a Steiner tree is upper bounded by $\rho(k) = O\left(\frac{k\log(N)}{\log(\bar{d})f(k)}\right)$ by Lemma 3. Until the packet reaches the destination, in each level of the tree, time for transmission of the packet (t) and propagation delay (d) is required. Thus, the expected elapsed time in Steiner tree approach in the data reconstruction and repair initialization is bounded by $O(\rho(k)(t+d))$. □

5.2 Multi-Shortest Path Approach

The message complexity and the time complexity analyses of the data reconstruction/repair initialization processes in multi-shortest path approach are described below.

5.2.1 Average Case Message Complexity

The message complexity of the proposed scheme depends on the total link counts of the shortest paths between the source node and the target nodes.

Theorem 3. The average number of messages used in multi-shortest path approach for one multi-unicast session for data reconstruction and repair initialization processes is upper bounded by $O\left(k\frac{\log(N)}{\log(\bar{d})}\right)$.

Proof. The mean link count between two arbitrary nodes is $\Theta\left(\frac{\log(N)}{\log(\bar{d})}\right)$ from Lemma 1. In multi-shortest path approach, there are k target nodes. So, the average message count is upper bounded by $O\left(k\frac{\log(N)}{\log(\bar{d})}\right)$. □

5.2.2 Average Case Time Complexity

The time complexity of the proposed scheme depends on the common edges, which are adjacent to the source node, of the shortest paths. The time complexity in this scheme also depends on the size of the longest shortest path which is between the source node and the furthest target node.

Lemma 4. The diameter of a power law random graph having an exponent $\beta > 3$ and average degree δ which is strictly greater than 1 is $\Theta(\log(N))$ (β is a parameter related with node degrees) [24].

Theorem 4. The mean elapsed time of the multi-unicast session for the data reconstruction/repair initialization process is upper bounded by $O\left(\frac{k}{\delta}t + \log(N)(t+d)\right)$.

Proof. Since the source node sends packets to its outgoing links in parallel, each of the outgoing links of the source node carries $\frac{k}{\delta}$ messages in average. Since the source node has the average degree of δ and each link adjacent to the source node transmits packets to $\frac{k}{\delta}$ storage nodes in average, before sending the last message on one link, $\left(\frac{k}{\delta} - 1\right)t$ time is elapsed. By Lemma 4, the average diameter is bounded by $\Theta(\log(N))$. So, the average elapsed time is upper bounded by $O\left(\frac{k}{\delta}t + \log(N)(t+d)\right)$. □

Steiner tree approach is generally more efficient than multi-shortest path approach in terms of number of messages. Furthermore, multi-shortest path approach is more efficient than Steiner tree approach in terms of time metric. These results can be derived from the theoretical analyses as well as from the experimental results. The summary of the analyses of data reconstruction/repair initialization communication is shown in Table 3.

Approaches	**Average Case Complexities**	
	Msg. Complexity	**Time Complexity**
Steiner Tree	$O\left(k^{0.8}\frac{\log N}{\log(\bar{d})}\right)$	$O(\rho(k)(t+d))$
Multi-Shortest Path	$O\left(k\frac{\log N}{\log(\bar{d})}\right)$	$O(\frac{k}{\delta}t + \log N(t+d))$

Table 3. The complexity analyses of the two approaches in the data reconstruction and repair initialization communication

For the average case, multi-shortest path approach uses $O\left(\frac{k}{\delta}t + \log N(t+d)\right)$ time for one multicast session, whereas Steiner approach uses $O\left(\rho(k)(t+d)\right)$ time for one multicast session. Generally, t value is very small with respect to k and $\log(N)$ values for small packets. As a result, in multi-shortest path approach $\left(\frac{k}{\delta}t\right)$ value is neutralized by t, even for high k values. On the other hand, in Steiner tree approach, average height of a tree is generally greater than link count of the longest path in multi-shortest path. (Note that we omitted the processing time which occurs before transmission and overlay construction time in Steiner tree and multi-shortest path approaches.)

6 THEORETICAL ANALYSIS OF THE DATA UPDATE COMMUNICATION

When the data update operation is required, storage nodes having the related encoded data should also update their encoded data. Therefore, in this section, there are $n-k$ target nodes instead of k.

6.1 Steiner Tree Approach

Steiner tree approach for the data update process is described in Section 4.2.1. Here, we derived theoretical results about the message complexity and the time complexity of this approach.

6.1.1 Average Case Message Complexity

The number of messages used in the data update operation is proportional to the number of the links in the constructed Steiner tree and the size of data to be updated. Due to MTU, "maximum amount of data that a link layer frame can carry [26]", large amounts of data cannot be sent in one go. In the analyses, we denote the amount of data to be updated as $\Delta size$ and the size of MTU as MTU.

Theorem 5. The average number of messages used in Steiner tree approach for the data update operation is upper bounded by $O\left((n-k)^{0.8}\frac{\log(N)}{\log(\tilde{d})}\frac{\Delta size}{MTU}\right)$.

Proof. The number of messages used in the data update process is equal to $\frac{\Delta size}{MTU}$ times of the number of links in the Steiner tree constructed by the algorithm in [8]. Since the mean distance between two nodes is $\Theta\left(\frac{\log N}{\log(\tilde{d})}\right)$ from Lemma 1 and the ratio of $L_{m=n-k}$ over L_u is $O\left((n-k)^{0.8}\right)$ from Lemma 2, the mean message complexity is upper bounded by $O\left((n-k)^{0.8}\frac{\log N}{\log(\tilde{d})}\frac{\Delta size}{MTU}\right)$ for the data update process. □

6.1.2 Average Case Time Complexity

The time complexity of this approach for the data update process is affected by the height of the constructed Steiner tree and the amount of data to be updated.

Theorem 6. The average elapsed time used in this approach for the data update communication is upper bounded by $O\left(t\frac{\Delta size}{MTU}+\rho(k)(t+d)\right)$.

Proof. The order of the mean height of a tree with k target nodes is denoted as $\rho(k)$ by the proof of Lemma 3. The required time for the transmission of entire Δ message and the height of the constructed tree determine the elapsed time. Therefore, this time is upper bounded by $O\left(t\frac{\Delta size}{MTU}+\rho(k)(t+d)\right)$. □

6.2 Multi-Shortest Path Approach

The message complexity and the time complexity analyses of the data update communication using multi-shortest path approach are described below.

6.2.1 Average Case Message Complexity

The message complexity of the proposed approach depends on the size of the data to be updated and the number of links of the shortest paths between the source node and the target nodes.

Theorem 7. The average number of messages used in multi-shortest path approach for data update is upper bounded by $O\left((n-k)\frac{\Delta size}{MTU}\frac{\log(N)}{\log(\bar{d})}\right)$.

Proof. We know that the mean link count between two arbitrary nodes is bounded by $\Theta\left(\frac{\log(N)}{\log(\bar{d})}\right)$ from Lemma 1. The source node has to send Δ message to $(n-k)$ target nodes in $\frac{\Delta size}{MTU}$ rounds. Eventually, the average number of messages used for data update in this approach is upper bounded by $O\left((n-k)\frac{\Delta size}{MTU}\frac{\log(N)}{\log(\bar{d})}\right)$. □

6.2.2 Average Case Time Complexity

The time complexity of the data update communication in this approach depends on the size of the data to be updated and the link count of the longest shortest path (between the source node and the furthest target node). The time complexity in this approach also depends on the the shortest paths' common links which are adjacent to the source node.

Theorem 8. The average elapsed time used for the data update communication using this approach is bounded by $O\left(\frac{n-k}{\delta}\frac{\Delta size}{MTU}t + \log(N)(t+d)\right)$ where $(n-k)$ is the number of target nodes.

Proof. The average degree of the source node is δ and each of these δ links is the first edge of the $\frac{n-k}{\delta}$ different shortest paths in average. The diameter of a graph is bounded by $\Theta(\log(N))$ from Lemma 4. In each link, before sending the last packet, $\left(\frac{n-k}{\delta}-1\right)\frac{\Delta size}{MTU}t$ time is elapsed in average. Consequently, the average elapsed time is bounded by $O\left(\frac{n-k}{\delta}\frac{\Delta size}{MTU}t + \log(N)(t+d)\right)$. □

In Table 4, the overview of the analyses for the data update communication is given. According to the theoretical and the experimental results, Steiner tree approach is advantageous in the most of the cases for the data update communication.

The order of message count in Steiner tree approach is lower than that of multi-shortest path approach for the data update operation. We can ignore the second terms of the average time complexities of both approaches, since these values are substantially smaller than the first terms in the real experiments. Thus, Steiner tree approach requires less time than multi-shortest path approach.

Approaches	**The Average Case Complexities**	
	Msg. Complexity	**Time Complexity**
Steiner Tree	$O\left((n-k)^{0.8\frac{\log(N)}{\log(\bar{d})}\frac{\Delta size}{MTU}}\right)$	$O\left(\frac{\Delta size}{MTU}t+\rho(k)(t+d)\right)$
Multi-Shortest Path	$O\left((n-k)\frac{\Delta size}{MTU}\frac{\log(N)}{\log(\bar{d})}\right)$	$O\left(\frac{n-k}{\delta}\frac{\Delta size}{MTU}t+\log(N)(t+d)\right)$

Table 4. The complexity analyses of the two approaches in the data update communication

7 SIMULATION RESULTS

The data reconstruction/repair initialization and update processes based on STA and MPA are both implemented in ns-3 and tested on topologies containing 500 nodes. BRITE [27] is used for generating network topologies. Available bandwidth values are selected uniformly between 10Mbps-100Mbps. Each link has the same propagation delay of 50 milliseconds. We have used Waxman topology model. Moreover, Reed-Solomon [22] scheme is used as an MDS code. We measured two metrics: the message count and the elapsed time used in the communication of initialization of data reconstruction/repair and data update processes. Message count is the number of packets that are passed through any link along the path. If the message size is higher than MTU, for example update size Δ, this message can be sent in $\Delta size/MTU$ different packets.

The values of n and k in erasure codes change due to on which platform the distributed storage system runs. For instance, in Peer-to-Peer (P2P) systems, which can also be used for storage systems [28], the node failures occur frequently due to the temporary disconnections or abandons, etc. [29]. In such systems, n and k values should be higher than that of dedicated data centers. Therefore, we have simulated different n, k values for collapsing the general view.

7.1 Data Reconstruction/Repair Initialization

Message count used in data reconstruction/repair initialization processes due to different target storage node counts (k), in networks which contain 500 nodes (N), is shown in Figure 3. Accordingly, after target storage node count exceeds some ratio of the total nodes (N), STA uses fewer messages than that of MPA. When the target storage node count increases in a fixed network size, STA's advantage increases.

MPA achieves better efficiency on message count in the network having 500 nodes when the target node count is less than 28. However, STA achieves better performance on message count in all tested target node counts in the networks having 500 nodes where target node count is higher than 28. The link counts of multi-shortest path trees generated with target node count less than 28, are less than the link count of the Steiner Tree generated with the same target node counts by the

approximation algorithm. So, the reason of the results of message counts shown in Figure 3 depends on the communication trees' link counts.

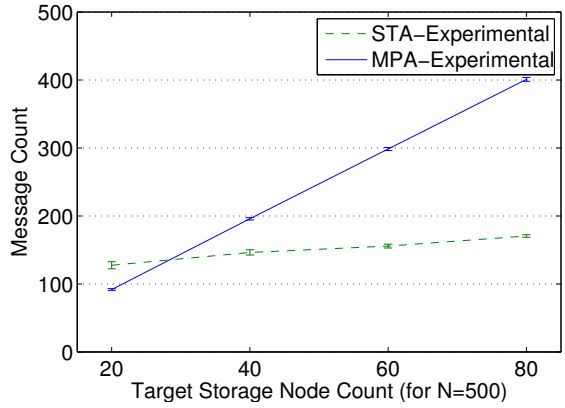

Figure 3. Experimental results of message count used in MPA and STA for data reconstruction/repair initialization

While STA optimizes the message count in most cases, it degrades the time value. In Figure 4, elapsed time in data reconstruction/repair initialization processes due to different target storage node counts, in the networks consisting of 500 nodes, is shown. While in MPA the elapsed time stays almost constant with the target nodes' increase, STA shows a decrease but it still requires more time. Since, the chance to find a shorter path increases when target node count increases in the algorithm in [8], the constructed tree's height tends to have a smaller value while target node count increases. But the tree's height is saturated for higher values of the target node counts. The reason of STA's bad performance in time value is the fact that the constructed Steiner trees' heights (hop count of the path between furthest node and the initiator node) are higher than that of the longest shortest paths constructed in MPA in average. So the required time for reaching the furthest node is higher in STA.

7.2 Data Update

In the simulations, initially, we examine the update of a data file of size 1MB. Every target node stored the encoded data, should receive (Δ) in 1MB size for the update. Message count used in both approaches with varying target node counts on the networks having 500 nodes is shown in Figure 5. After the target storage node count exceeds a specific value, STA uses fewer messages than MPA for data update. When the target storage nodes increase in a fixed network size, STA's advantage in message count increases.

Elapsed time for data update process with varying target node counts on the networks having 500 nodes is shown in Figure 6. MPA uses more time than STA

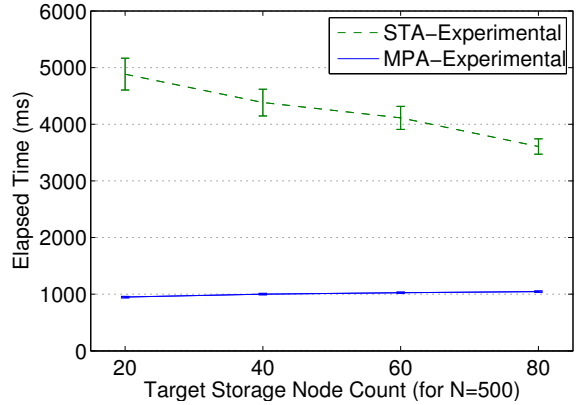

Figure 4. Experimental results of elapsed time in MPA and STA for data reconstruction/repair initialization

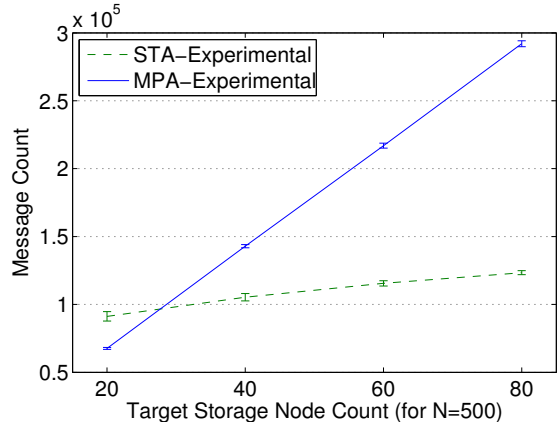

Figure 5. Experimental results of message count used in MPA and STA for data update

on every target node count for data update process. The target nodes on the same path receive the same packet in STA and different packets are sent (transmission time overhead) for each target nodes in the same path in MPA. So, when the target node count increases in a fixed network size, the advantage of STA in elapsed time increases.

Furthermore, message count and the elapsed time used in data update processes with respect to different update sizes are shown in Figure 7 and Figure 8, respectively. Since, Steiner trees have more links than multi-shortest path trees for 20 target nodes, message count used in STA is higher than MPA for every update sizes in that target node count. In Figure 8, it can be seen that the elapsed time in

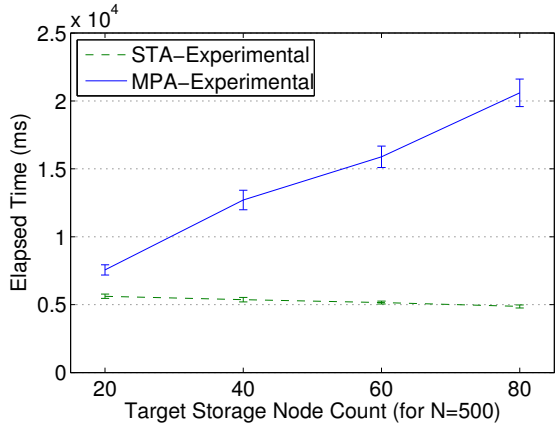

Figure 6. Experimental results of elapsed time in MPA and STA for data update

STA is proportional to the update size. In addition, the elapsed times in STA for 20 and 40 target nodes are almost identical.

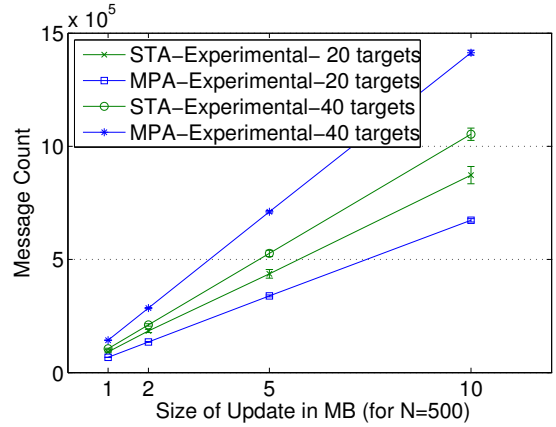

Figure 7. Experimental results for used message count in MPA and STA for data update on different update sizes for 20 and 40 target nodes

We searched for the proper constant values for the orders to check the consistency between the theoretical analyses and the experimental results that we obtained in Section 5 and Section 6. For example, we searched for the proper c_1 and c_2 values for the order in Theorem 3 to find the appropriate $\left(c_1\left(k\frac{\log(N)}{\log(d)}\right)+c_2\right)$.

In Figure 9 a), the experimental results and theoretical analyses with appropriate constants for the number of messages used in data reconstruction/repair initialization process are presented. As it can be seen, the theoretical results and the

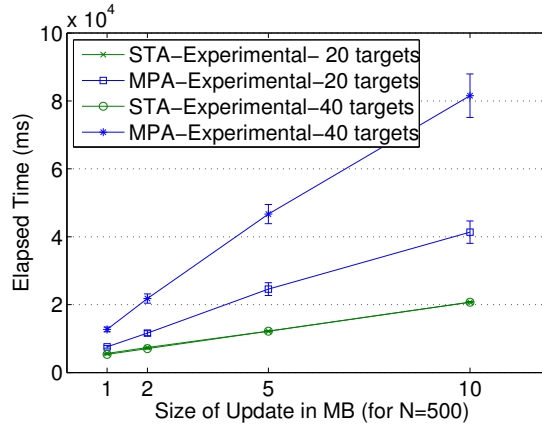

Figure 8. Experimental results for elapsed time in MPA and STA for data update on different update sizes for 20 and target nodes

experimental results overlap. In Figure 9 b) the experimental results and theoretical analyses with appropriate constants for the elapsed time used in data reconstruction/repair initialization process are given.

In Figure 9 c), the experimental results and theoretical analyses with proper constants for the number of messages required in data update are given. These experimental results correspond substantially to the theoretical results with proper constants in data update. In Figure 9 d), the experimental results and theoretical analyses with proper constants for the elapsed time in data update are given. Here, the results overlap substantially.

8 CONCLUSIONS

In this work, analysis of the communication cost of a distributed data storage system using MDS codes has been studied. To that end, two approaches, STA and MPA, were tested for data reconstruction/repair initialization and update operations for different networks. Furthermore, we derived theoretical bounds of the complexity of message and time usage in these approaches. Simulation results indicate that, while MPA brings an advantage in terms of time usage, STA has better efficiency in terms of message count for data reconstruction/repair operation. Also, STA is more efficient in both parameters than MPA for the data update operation (except some target node count values). These experimental results are supported by the theoretical results.

These results can shed light to the designers of distributed storage systems to choose between different communication approaches taking their needs and priorities into account. For instance, for a system where time is abundant but bandwidth is scarce, STA can be chosen for all operations. On the other hand, for an application

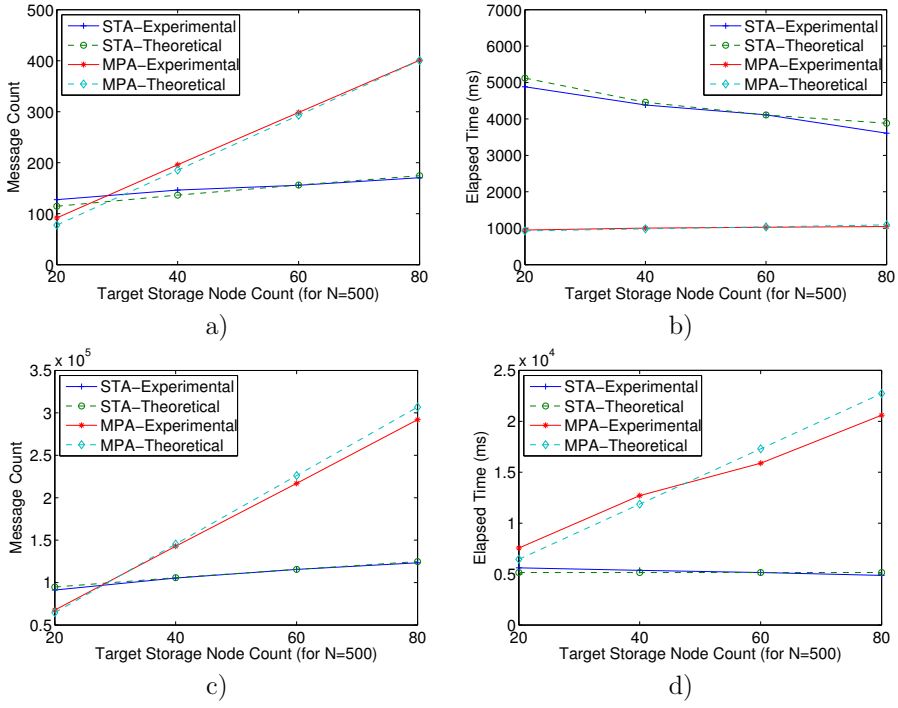

Figure 9. Comparison of the theoretical analysis and experimental results of MPA and STA in data reconstruction/repair initialization (a), b)) and data update (c), d))

where communication costs are relatively low but time is limited, MPA can be chosen for data reconstruction/repair initialization operations. A new approximation algorithm for minimum Steiner tree construction which is capable of keeping the trees height under some lower order can be conceived for the future research. Furthermore, analyzing the approaches in different settings, in which multiple operations occur concurrently, could be considered as a future work.

Acknowledgements

This study was supported by OYP research fund of Turkish Government No. 05-DPT-003/35.

A PROOF OF LEMMA 3

Proof. Steiner tree approximation algorithm in [8] initially eliminates the paths which may increase the cost (link count) of the tree. Afterwards, it divides nodes into two distinct sets. The first set contains only one storage node and the other set

contains remaining $k-1$ storage nodes. Then, in the first iteration, the algorithm searches for a node (from the second set) which is the closest (in terms of link count) to the node in the first set. After finding the closest node, it is removed from the second set and the nodes in the path between the selected node (including) and the node in the first set are added to the first set. Then, in the second iteration, the algorithm searches for a node from the second set which is the closest to the first set. In other words, it searches the closest pair of nodes one from each set. Then, the closest node is removed from the second set and it together with the nodes in the path between the closest pair are added to the first set. The algorithm continues until there is no node left in the second set. After the $(k-1)^{\text{th}}$ iteration is completed, the algorithm is finished. Finally, each path between the closest nodes constitutes Steiner tree's branches.

The average distance between two nodes is $O\left(\log(N)/\log(\tilde{d})\right)$ from Lemma 1. Assume that we consider k nodes which are randomly selected from the graph (the case in the first iteration). The mean value of the path length between the source node and the closest one to the source (initiator) node (one of the $k-1$ nodes) should be smaller than $O\left(\log(N)/\log(\tilde{d})\right)$. In this case, the closest pair is one of the $(k-1)$ pairs, one is target and the other one is any of the $k-1$ nodes, and these pairs have the mean $O\left(\log(N)/\log(\tilde{d})\right)$ for the distance.[1] Since the distances are not uniformly distributed, the closest pair in the set of $k-1$ pairs has the mean distance which is smaller than $O\left(\log(N)/\log(\tilde{d})\right)$. Furthermore, the mean distance of the closest pair of $k-1$ pairs is inversely proportional to $f(k)$ where $f(k)$ is a monotonically increasing function of k.[2] In other words, the mean value of the shortest path in a set of nodes is inversely proportional to a function of the number of pairs. The same holds for the next iterations, the closest path length is inversely proportional to a function of the number of pairs in the iteration i.

The average height of a Steiner tree constructed by algorithm in [8] can be calculated as follows. When the first closest node is selected, the tree has one source node and one target node. In this case, the tree's mean height is $O\left(\frac{\log(N)}{\log(\tilde{d})f(k)}\right)$. In the remaining closest pair selections, the tree's height can be increased at most as the next shortest paths' lengths. The link count of a tree having i target nodes is bounded by $O\left(i^{0.8\frac{\log(N)}{\log(\tilde{d})}}\right)$ by Lemma 1 and Lemma 2. This means that the constructed tree has $O\left((i)^{0.8\frac{\log(N)}{\log(\tilde{d})}}\right)$ nodes when the i^{th} iteration begins where $i>1$. Accordingly, when the i^{th} iteration begins, there are $O\left((i)^{0.8\frac{\log(N)}{\log(\tilde{d})}}\times(k-i)\right)$ pairs, and the algorithm searches for the closest one of the $O\left((i)^{0.8\frac{\log(N)}{\log(\tilde{d})}}\times(k-i)\right)$ pairs.

[1] We know from the sample mean, the mean distance of a subset of a distribution is the same as the mean distance of the all distribution.

[2] Consider the case of $k=2$, then the mean distance is bounded by $O\left(\log N/\log(\tilde{d})\right)$ and if $k=500=N$, then the mean distance of the closest pair is $O(1)$.

Thus, when the i^{th} iteration begins, the order of the number of pairs, in which the closest one is searched, is given below:

$$P(i) = \begin{cases} O\left((i)^{0.8\frac{\log(N)}{\log(\tilde{d})}} \times (k-i)\right), & \text{where } i > 1,\ i < k,\ i \in \mathbb{N}^+, \\ k-1, & \text{where } i = 1. \end{cases} \tag{1}$$

When $i > 1$, $P(i)$ function is a concave function, since its second derivative is negative. Since $P(i)$ is evaluated with the integers in the interval $[2, k-1]$, the minimum of $P(i)$ is observed in $P(2)$ or $P(k-1)$. $P(2)$ is bounded by $O\left(k\frac{\log(N)}{\log(\tilde{d})}\right)$ whereas $P(k-1)$ is bounded by $O\left(k^{0.8}\frac{\log(N)}{\log(\tilde{d})}\right)$. Thus for high values of N, the minimum order is provided in $P(1)$ which is $O(k)$ (when the order of N is greater than $\tilde{d}^{k^{0.2}}$).

The height of a tree constructed in the i^{th} iteration is denoted as $H(i)$. In the $(i+1)^{\text{th}}$ closest pair selection, the node, that a new selected path (between the closest pair) will be connected to, can be at any level in the tree constructed in the i^{th} iteration. If the new selected path is connected to the tree through a node which is furthest to the root, it can increase the height by $O\left(\frac{\log(N)}{\log(\tilde{d})f(P(i+1))}\right)$. But if the next path is connected to the tree through a node whose level is one less than the height's level, it can increase the height by $O\left(\frac{\log(N)}{\log(\tilde{d})f(P(i+1))} - 1\right)$, and so on. Therefore, the order of the expected increment of tree's height in $(i+1)^{\text{th}}$ iteration is given below:

$$O(E(H(i+1) - H(i))) = O\left(\sum_{l=1}^{\frac{\log(N)}{\log(\tilde{d})f(P(i+1))}} \frac{1}{H(i)} l\right), \tag{2}$$

$$O(E(H(i+1) - H(i))) = O\left(\frac{1}{H(i)} \frac{\log(N)}{\log(\tilde{d})f(P(i+1))} \left(\frac{\log(N)}{\log(\tilde{d})f(P(i+1))} + 1\right)/2\right). \tag{3}$$

Consequently, the order of the mean height of a tree, $H(i+1)$, in the $(i+1)^{\text{th}}$ iteration is as follows:

$$O(H(i+1)) = O(H(i)) + O\left(\sum_{l=1}^{\frac{\log(N)}{\log(\tilde{d})f(P(i+1))}} \frac{1}{H(i)} l\right) \tag{4}$$

where $O(H(0)) = 0$, $O(H(1)) = O\left(\frac{\log(N)}{\log(\tilde{d})f(P(1))}\right)$. Since, we consider the upper bound of the expected height, we can use $P(1)$ instead of $P(i)$ when $i > 1$. (Since $P(1)$ has minimum order among all $P(i)s, i > 1, i < k,$) In other words, if we use $P(1)$ for other iterations (i.e. $i > 1$), the upper bound is preserved, since it

increases the order of the lengths of the shortest paths between the closest pairs for the iteration i where $i > 1$ and $i < k$. Thus, the order of the expected height of the tree constructed in the $(i+1)^{\text{th}}$ iteration can be calculated as below:

$$O(H(i+1)) = O(H(i)) + O\left(\sum_{l=1}^{\frac{\log(N)}{\log(d)f(P(1))}} \frac{1}{H(i)} l\right), \tag{5}$$

$$O(H(i+1)) = O(H(i)) + O\left(\sum_{l=1}^{\frac{\log(N)}{\log(\tilde{d})f(k)}} \frac{1}{H(i)} l\right), \tag{6}$$

$$O(H(i+1)) = O(H(i)) + O\left(\frac{1}{H(i)} \left(\frac{\log(N)}{\log(\tilde{d})f(k)}\right) \left(\frac{\log(N)}{\log(\tilde{d})f(k)} + 1\right)/2\right) \tag{7}$$

where $O(H(0)) = 0$, $O(H(1)) = O\left(\frac{\log(N)}{\log(\tilde{d})f(k)}\right)$.

Assume that a is equal to $\frac{\log(N)}{\log(\tilde{d})f(k)}$ and c is equal to $\left(\sum_{l=1}^{\frac{\log(N)}{\log(\tilde{d})f(k)}} l\right)$. Thus, the order of the mean height of a tree after the $(k-1)^{\text{th}}$ iteration can be calculated as below:

$$H(k-1) = a + \frac{c}{a} + \underbrace{\frac{c}{a + \frac{c}{a}} + \frac{c}{a + \frac{c}{a} + \frac{c}{a + \frac{c}{a}}} + \ldots}_{(k-3) \text{ terms}} \tag{8}$$

Note that, i^{th} term is always greater than the $(i+1)^{\text{th}}$ term in Equation (8) where $i > 0$. The order of the height of a tree having k target nodes, $\rho(k)$, is the order of the height of the tree after $(k-1)^{\text{th}}$ iteration is completed. Therefore, the order of $H(k-1)$, $\rho(k)$, can be upper bounded by:

$$\rho(k) = a + \frac{c}{a} + (k-3)\frac{c}{a + \frac{c}{a}}, \tag{9}$$

$$\rho(k) = O\left(\frac{\log(N)}{\log(\tilde{d})f(k)} + \frac{\log(N)}{\log(\tilde{d})f(k)} + \frac{(k-3)\log(N)}{\log(\tilde{d})f(k)}\right), \tag{10}$$

$$\rho(k) = O\left(\frac{\log(N)k}{\log(\tilde{d})f(k)}\right). \tag{11}$$

□

REFERENCES

[1] Gantz, J.—Reinsel, D.: The Digital Universe in 2020: Big Data, Bigger Digital Shadows, and Biggest Growth in the Far East. IDC iView: IDC Analyze, Future, 2012.

[2] Kubiatowicz, J.—Bindel, D.—Chen, Y.—Czerwinski, S.—Eaton, P.—Geels, D.—Gummadi, R.—Rhea, S.—Weatherspoon, H.—Weimer, W.—Wells, C.—Zhao, B.: OceanStore: An Architecture for Global-Scale Persistent Storage. ACM SIGPLAN Notices, Vol. 35, 2000, No. 11, pp. 190–201, doi: 10.1145/378993.379239.

[3] Druschel, P.—Rowstron, A.: PAST: A Large-Scale, Persistent Peer-to-Peer Storage Utility. Proceedings of the 8th Workshop on Hot Topics in Operating Systems, May 2001, pp. 75–80, doi: 10.1109/HOTOS.2001.990064.

[4] Dabek, F.—Kaashoek, M. F.—Karger, D.—Morris, R.—Stoica, I.: Wide-Area Cooperative Storage with CFS. Proceedings of the 18th ACM Symposium on Operating System Principles, 2001, pp. 202–215, doi: 10.1145/502034.502054.

[5] Bhagwan, R.—Tati, K.—Cheng, Y.—Savage, S.—Voelker, G. M.: Total Recall: System Support for Automated Availability Management. Proceedings of the 1st Conference on Symposium on Networked Systems Design and Implementation (NSDI '04), 2004, pp. 25–25.

[6] Adya, A.—Bolosky, W. J.—Castro, M.—Cermak, G.—Chaiken, R.—Douceur, J. R.—Howell, J.—Lorch, J. R.—Theimer, M.—Wattenhofer, R. P.: FARSITE: Federated, Available, and Reliable Storage for an Incompletely Trusted Environment. Proceedings of the 5th Symposium on Operation Systems Design and Implementation, 2002, pp. 1–14, doi: 10.1145/1060289.1060291.

[7] Muthitacharoen, A.—Morris, R.—Gil, T. M.—Chen, B.: Ivy: A Read/Write Peer to Peer File System. Proceedings of the 5th Symposium Operating System Design and Implementation (OSDI 02), 2002, pp. 31–44, doi: 10.1145/1060289.1060293.

[8] Dolagh, S. V.—Moazzami, D.: New Approximation Algorithm for Minimum Steiner Tree Problem. International Mathematical Forum, Vol. 6, 2011, No. 53, pp. 2625–2636.

[9] Haytaoglu, E.—Dalkilic, M. E.: On the Communication Cost of Distributed Storage Systems Using MDS Erasure Codes. Proceedings of the 3rd International Symposium on Computing in Science & Engineering, 2013, pp. 102–107.

[10] Berman, P.—Ramaiyer, V.: Improved Approximations for the Steiner Tree Problem. Journal of Algorithms, Vol. 17, 1994, No. 3, pp. 381–408, doi: 10.1006/jagm.1994.1041.

[11] Karpinski, M.—Zelikovsky, A.: New Approximation Algorithms for the Steiner Tree Problems. Journal of Combinatorial Optimization, Vol. 1, 1995, pp. 47–65, doi: 10.1023/A:1009758919736.

[12] Hougardy, S.—Prömel, H. J.: A 1.598 Approximation Algorithm for the Steiner Problem in Graphs. Proceedings of the Tenth Annual ACM-SIAM SODA, 1999, pp. 448–453.

[13] Li, J.—Yang, S.—Wang, X.—Xue, X.—Li, B.: Tree-Structured Data Regeneration with Network Coding in Distributed Storage Systems. 17th International Workshop on Quality of Service (IWQoS 2009), 2009, pp. 1–9.

[14] Li, J.—Yang, S.—Wang, X.—Li, B.: Tree-Structured Data Regeneration in Distributed Storage Systems with Regenerating Codes. Proceedings of the 29th Conference on Information Communications (INFOCOM '10), 2010, pp. 2892–2900, doi: 10.1109/INFCOM.2010.5462122.

[15] Gerami, M.—Xiao, M.—Skoglund, M.: Optimal-Cost Repair in Multi-Hop Distributed Storage Systems. 2011 IEEE International Symposium on Information Theory Proceedings (ISIT), 2011, pp. 1437–1441, doi: 10.1109/ISIT.2011.6033777.

[16] Qian, Z.—You, I.—Lu, Y.—Lu, S.: A Tree-Based Hierarchy Data Storage Framework in a Pervasive Space. Computing and Informatics, Vol. 30, 2011, No. 6, pp. 1181–1199.

[17] Aguilera, M.—Janakiraman, R.—Xu, L.: On the Erasure Recoverability of MDS Codes under Concurrent Updates. Proceedings of Symposium on IEEE Information Theory, 2005, pp. 1358–1362, doi: 10.1109/ISIT.2005.1523564.

[18] Anthapadmanabhan, N. P.—Soljanin, E.—Vishwanath, S.: Update-Efficient Codes for Erasure Correction. 48th Annual Allerton Conference on Communication, Control, and Computing, 2010, pp. 376–382, doi: 10.1109/ALLERTON.2010.5706931.

[19] Plank, J. S.: A Tutorial on Reed-Solomon Coding for Fault-Tolerance in RAID-Like Systems. Software: Practice and Experience, Vol. 27, 1997, No. 9, pp. 995–1012.

[20] Karp, R. M.: Reducibility Among Combinatorial Problems. Complexity of Computer Computations, 1972, pp. 85–103, doi: 10.1007/978-1-4684-2001-2_9.

[21] Takahashi, H.—Matsuyama, A.: An Approximate Solution for the Steiner Problem in Graphs. Mathematics Japonica, Vol. 24, 1980, pp. 573–577.

[22] Reed, I. S.—Solomon, G.: Polynomial Codes over Certain Finite Fields. Journal of the Society for Industrial and Applied Mathematics, SIAM, Vol. 8, 1960, No. 2, pp. 300–304.

[23] Dijkstra, E. W.: A Note on Two Problems in Connexion with Graphs. Numerische Mathematik, Vol. 1, 1959, No. 1, pp. 269–271, doi: 10.1007/BF01386390.

[24] Chung, F.—Lu, L.: The Average Distance in a Random Graph with Given Expected Degrees. Internet Mathematics, Vol. 1, 2003, No. 1, pp. 91–114.

[25] Chuang, J.—Sirbu, M.: Pricing Multicast Communication: A Cost-Based Approach. Telecommunication Systems, Vol. 17, 2001, No. 3, pp. 281–297.

[26] Kurose, J. F.—Ross, K. W.: Computer Networking: A Top Down Approach. 3rd ed. Addison Wesley, 2005.

[27] Medina, A.—Matta, I.—Byers, J.: BRITE: A Flexible Generator of Internet Topologies. Technical Report, Boston University, 2000.

[28] Gupta, A.—Awasthi, L. K.: Peer-to-Peer Networks and Computation: Current Trends and Future Perspectives. Computing and Informatics, Vol. 30, 2011, No. 3, pp. 559–594.

[29] Duminuco, A.—Biersack, E.: Hierarchical Codes: How to Make Erasure Codes Attractive for Peer-to-Peer Storage Systems. Eighth International Conference on Peer-to-Peer Computing (P2P '08), 2008, pp. 89–98, doi: 10.1109/P2P.2008.9.

Elif Haytaoglu received her Ph.D. degree in information technologies in 2015 from International Computer Institute Ege University, Turkey. Currently, she is Assistant Professor at Department of Computer Engineering, Pamukkale University, Denizli, Turkey. Her research interests include distributed systems, distributed storage systems, regenerating codes, computer networks, linear algebra and parallel programming.

Mehmet Emin Dalkilic received his B.Sc. degree in electrical and electronics engineering in 1985 from Hacettepe University, Turkey. He received his M.Sc. and Ph.D. degrees in Electrical and Computer Engineering Department at Syracuse University, USA, in 1989 and 1994, respectively. Currently he is Professor International Computer Institute, Ege University, Turkey. His research interests include algorithms, computer networks, network security, and parallel and distributed computing.