Regenerating Codes Consider with Nodes Performances

Bo Wang
Data Utilization Center Technology and Engineering Center for Space Utilization, Chinese Academy of Sciences
Beijing, China
wangbo@csu.ac.cn

Shengyang Li
Data Utilization Center Technology and Engineering Center for Space Utilization, Chinese Academy of Sciences
Beijing, China
shyli@csu.ac.cn

Kaigui Wu
College of Computer Science Chongqing University, CQU
Chongqing, China
kaiguiwu@cqu.edu.cn

***Abstract*—Regenerating codes is a design of network code with efficient repair capabilities. It requires the lost data regenerated in new system nodes to compensate the data loss incurred by storage nodes failure. In this paper, our general objective is to construct an optimal topology for regeneration process to minimize the volume of actual network traffic caused by nodes repair and take computing and communication performance of nodes into consider. For this purpose, a new topology construction method referred to as NARP (Node-state Awareness Regeneration Process) has been proposed. It combines an evaluation scheme of nodes with a node-state awareness path-finding algorithm to achieve our purpose. With the extensive analysis and experimental evaluations, we show that NARP is able to achieve a both fast and reliably regeneration especially for complex distributed storage system with large number of nodes.**

Keywords- Distributed Storage System; Data Regeneration; Network Topology

I. INTRODUCTION *(HEADING 1)*

In order to provide high data reliability, distributed storage systems disperse data to a number of storage nodes. In storage nodes, redundancy is normally existences in order to deal with node failures. There are two common methods in introducing redundancy, namely replication coding and erasure coding [1]. In the former method, the data block is duplicated and stored in different storage nodes. A coding scheme in which a data file is replicated three times is employed by the Google file system [2]. Although replication coding is easy to implement and manage, it has lower storage efficiency than erasure codes, such as Reed–Solomon (RS) codes. In order to achieve higher storage efficiency, erasure codes such as RS code are recently adopted in several cloud storage systems, including Oceanstore [3], Windows Azure [4], etc.

In a large-scale distributed storage system, it is a common case to recovery data. Once a storage node fails, the whole data block of other surviving nodes should be transferred into a new node to regenerate the data. Therefore, the deployment of erasure codes incurs a significant overhead of network traffic during the repair process. The required traffic for repairing a failed node, which is called repair bandwidth per node, is particularly importance in bandwidth-limited storage networks. *Regenerating codes* (RGC) was introduced by Dimakis et al. for the purpose of reducing the repair bandwidth. In the pioneer work in [5], a data file with M bits over the finite field is divided into k parts data block and encoded into n nodes, with each node storing bits original data such that the original data file can be recovered from any k nodes, as an (n, k, d)-distributed storage system (DSS).

When a node fails, we replace it by a new node namely *newcomer* and regenerate the failure data by downloading data from d number of surviving nodes ($k \le d \le n$). The storage nodes which participate in the repair process are called the *providers*. The total data downloaded from the providers is named *repair bandwidth* in [5]. In the tradeoff between the storage and the repair bandwidth, there are two extreme points of special interest: minimum storage regenerating (MSR) codes and minimum bandwidth regenerating (MBR) codes.

Most of the studies on regenerating codes in literatures focus on the parameter optimization in encoding process. However, in large-scale and complex distributed storage systems, how to construct a network topology for data transmitting to achieve optimal tradeoff between utilization efficiency of bandwidth and reliability of system is an important research hotspot for regeneration process. It has conventionally been assumed that the regeneration process is performed on a simple *star-structured* topology, *i.e.*, the newcomer receives coded segments directly from each of providers. In the overlay mesh connecting storage nodes, however, not all overlay links enjoy the same available bandwidth. Besides, we used to acquiesce that all nodes running in data center of distributed storage system have same physical property. But the difference in update batch and service condition will cause great disparities among nodes in property, cost, reliability and residual life. If we take the heterogeneity of storage nodes and heterogeneity of bandwidth on links into account at the same time, the node-state awareness topology constructed for transmitting data to newcomer naturally ensues over time.

In this paper, we consider the general case of constructing such node-state awareness topologies for regeneration process (referred to as NSA). With a variable number of providers in this topology, as well as the use of regenerating codes to balance the relationship between repair bandwidth and system reliability. Our new design is able to work effectively with the bandwidth heterogeneity and nodes heterogeneity. With the extensive analysis and quantitative evaluations based on statistical data in PlanetLab, we are able to show that the new topology helps to be one step closer towards a practical repair of failed storage nodes in distributed storage systems.

The remainder of the paper is organized as follows. Sec. II shows several conventional topology structures and presents their disadvantages with an illustrative example. In Sec. III we introduce the network model, and present

978-1-5386-7445-1/18/$31.00 ©2018 IEEE
DOI 10.1109/DCABES.2018.00015

our extensive analysis on the node-state awareness regeneration process. Sec. IV analyzes the performance of NSA in comparison with some existing schemes. Sec. V concludes this paper.

II. IMPROVED REGENERATING CODES: A MOTIVATING EXAMPLE

A. Bandwidth heterogeneity

We now introduce an illustrative example of traditional data regeneration in the distributed storage system in Figure1. Figure1 (a) shows the network model. There are five storage nodes, denoted by V_0,V_1,V_2,V_3 and V_4 .The bandwidth capacity of the link between two providers is heterogeneous. We assume that the redundancy is coded by a (5,3,*d*) MSR code, stored in V_0,V_1,V_2,V_3,V_4 . Each storage node stores a coded block of $M/3$ bits, the size of the original data is also *M* bits. Even though MSR codes are not able to reach the minimum regeneration traffic of regenerating codes, they cost the least amount of storage space in storage nodes. Other kinds of regenerating codes can further reduce the regeneration traffic with an increased storage cost, but MSR codes use the storage space most effectively. Therefore, we consider MSR codes in this paper.

When a node failure occurs, the redundancy that is lost due to the failure should be regenerated on a replacement newcomer. We assume V_0 is the failed node, V_5 is selected to be the newcomer. Since (5, 3, *d*) MSR code is used, V_5 needs to receive redundancy from at least three providers.

Figure1 (b)-Figure1 (d) show illustrations of three regeneration schemes. For the conventional star-structured regenerating code (STAR) in Figure1 (b), when $d = 3$, which means V_5 select V_1,V_3 and V_4 as providers, V_5 receives data directly from the three providers, illustrated by the darkened edges. Considering the regeneration time, *i.e.*, the time that the newcomer spends on regenerating a new coded block, STAR costs $(M/3)/25Mbps$ seconds to accomplish the regeneration, because the transmission is bottlenecked by the link between V_5 and V_4 .We ignored the encoding time of MRS code because the processors usually perform encoding operations much faster than the network transmission, and the encoding can be performed simultaneously with the transmission.

On the other hand, if $d = 4$, there are more than three providers, for example V_1,V_2,V_3 and V_4 are used as providers in the regeneration. In Figure1(c), with the employment of MSR codes in STAR, there will be only *half* of $M/3$ bits transferred on each darkened edge. Therefore, the regeneration time can be reduced to $(M/6)/20Mbps$ seconds.

STAR, however, suffer from the bottleneck links of (V_5, V_4) and (V_5, V_2), respectively. If we consider the links between providers, we can utilize these links to bypass the slow bottleneck link in STAR. Jun Li *et al.* [8] proposed a tree-structured regenerating code, which construct a regeneration tree with V_1,V_2,V_3 and V_4 as providers (TREE), and use MSR codes in the system. We show this construction in Figure1 (d). However, the reliability of tree structure is worse than star structure, on account of the data transmission of every provider has strong dependence on its father nodes. In Figure1 (d), V_2 and V_4 directly contact V_3 to relay data to newcomer, we call V_3 *father node* or *next hop* of them. As shown in Figure1 (e), if V_3 fail when it has received data transmitted from V_2 and V_4 , the preceding data transmission is invalid. Similarly, the failure of V_1 in regeneration process will lead to data transmission malfunction of all nodes. Practical distributed storage system has large number of storage nodes. Therefore, there will be a certain amount of nodes on the path from a provider to a newcomer. Failure of any transit node on the path will destroy the data transmission of the whole path. We assume that the failure probability of every node is identical, the failure probability of a path is exponentially advanced with the growth of nodes on path. The number of nodes on the path of a provider relay data to newcomer which is also named *hop counts* should be taken into account during our design process. In Figure1 (f), we show a sample example of a revised TREE just take hop counts in to consideration. In this situation, the hop counts in this regeneration schemes is smaller than in Figure1(d), only the failure of V_1 will influent the success of data transmission during regeneration process.

B. Nodes heterogeneity

The equipment performance of nodes in large-scale data center, especially in cloud storage system have great difference in storage capacity, for instance, CPU utilization, I/O speed and so on. When choose node to relay data transmitting from a provider, we must have a comprehensive consideration on *node state*, which include three parts, the repair bandwidth, performance of node and hop counts of each path. The real-time change in link bandwidth between nodes and *the service character* of distributed storage system lead to strong time dependency of network status. Distributed storage system need to provide real-time services for users who want to access or download file. When node is providing services, available link bandwidth for data recovery is obviously reduced. But the performance of node always has relatively small change over time. It is necessary to update the two factor asynchronously so that we can get the real-time state of nodes and avoid the dissipation of bandwidth consumption

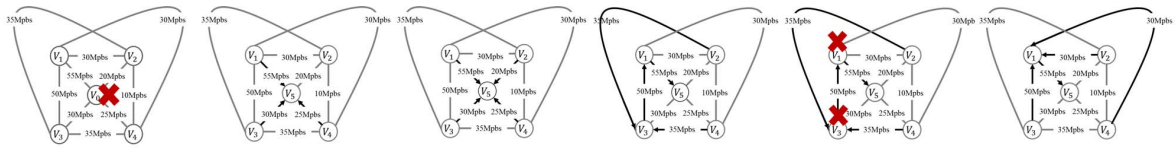


Figure 1. Examples of sex regeneration schemes

caused by frequently updating at the same time.

III. REGENERATION PROCESS BASED ON NODE-STATE AWARENESS

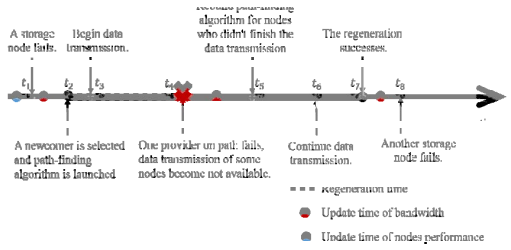


Figure 2. Timeline of the NSA regeneration process

In Figure2, we use a timeline to describe date transmit of a successful regeneration process with NSA algorithm. Once a storage node fails at t_1, the system has to select a replacement node, which is called newcomer, to regenerate the lost code block. If there is no available newcomer, the regeneration process has to be postponed until newcomers join to network at t_2. When a regeneration process beginning, node-state awareness path-finding algorithm is launched to construct an optimal data transmitting topology between providers and newcomer. Firstly, node performance is calculated by evaluate algorithm introduced by Sec. III. A. If we mark the newcomer as N_0, transmission cost from each provider to N_0 can be calculated by algorithm introduced in Sec. III. B. We can select the optimal provider and mark it as N_1. Base on this, we calculate transmission cost from each other provider to N_1 and compare to choose the provider which has minimum cost when transmit data to set $[N_0, N_1]$. By the same way, we can get the optimal data transmission topology with d providers $[N_1, N_2, ..., N_d]$. After that, data transmission to newcomer along the topology begins at t_3. Once there is a failure occurred on a provider or newcomer at t_4, regenerating process is terminated and the system has to rebuild path-finding algorithm for nodes who didn't finish the data transmission at. And continue to transmit data at t_6. If smoothly, regeneration will succeed at t_7. The mechanism keeps idle until another storage node fails at t_8. Fig. 3 describe an intact regeneration process by imaginary line on the timeline. The red dot and blue dot on the timeline show the asynchronous update of bandwidth and performance of nodes respectively.

A. Evaluate the Performance of Node in System

In this section, we discuss how to evaluate nodes performance in regeneration process. Assuming there are t_5 m evaluation indicators for n-1 providers, notation $N_i, i \in [0, n-1]$ is used to represent providers who are waiting for evaluating. Notation B_j is used to represent evaluation indicators. Then, we use x_{ij} $(i = 1,2,\ldots,n-1; j = 1,2,\ldots m)$ to record assessment result of provider N_i under... Finally, we obtain the decision matrix of composite indexes as A:

$$A = \left[x_{ij}\right]_{(n-1)\times m} = \begin{bmatrix} x_{11} & \cdots & x_{1m} \\ \vdots & \ddots & \vdots \\ x_{(n-1)1} & \cdots & x_{(n-1)m} \end{bmatrix} \quad (1)$$

Defining A_i as the comprehensive assessment result of provider N_i, it can be calculated by: $A_i = \sum_{j=1}^{m} \omega_j x_{ij} = \boldsymbol{\omega} \boldsymbol{x_i}$, where $\boldsymbol{x_i} = (x_{i1}, x_{i2}, \ldots, x_{im})^T$ is results vector of c_i which is made up by m assessment results, $\boldsymbol{\omega} = (\omega_1, \omega_2, \ldots, \omega_m)^T$ is weight vector which is made up by weight of evaluation indicators.

Now, we consider how to calculate A_i by information entropy. Learning from information theory, information entropy could be used to measure the degree of importance of each evaluation indicator. The greater the range of indicator, the larger the information entropy will be and the more important the indicator will be. Take an indicator as example, if it plays an important role in nodes performance evaluation process, there will be much difference between evaluation results of this indicator on different providers. On the contrary, if it is not important for performance evaluation, the gap of evaluation results between different providers will be small. Therefore, we calculate the information entropy of each evaluation indicator according to evaluation results of providers on different indicators. Then, regarding the information entropy as weight of each indicator. We can form the weight vector and obtain comprehensive assessment result A_i.

Step 1: Before evaluating equipment performance of nodes, factors which are necessary to evaluate need to be listed at first. In this paper, we choose four main affecting factors and show the necessity as well as assessment method:

- I/O speed b_1: literature statistics [9] shows that I/O speed is one of important influencing factors for data recovery process. Specifically, time of I/O can account for up to 30% in the entire data transfer process. It is bottleneck of data transmission especially for large data block file storage systems. I/O speed which expressed as v is measured by iostat tool of Linux System in this paper. This factor is the property of efficiency so that it could be calculated by:

$$b_1 = \begin{cases} 1 & v \geq v_{\max} \\ \dfrac{v - v_{\min}}{v_{\max} - v_{\min}} & v_{\min} < v < v_{\max} \\ 0 & v \leq v_{\min} \end{cases} \quad (2)$$

  Where v is the best to be superior to v_{max} and the worst to be junior to v_{min}. The best value of b_1 is 1.
- CPU utilization b_2: When regenerating data, we should avoid choosing nodes with high CPU utilization. On the one hand, nodes with high CPU utilization will result in delay of data regeneration because of its frequently read-write or calculation. On the other hand, high CPU utilization may be caused by system file errors or computer viruses. Given the above, it is certainly unwise to choose nodes with high CPU utilization to recovery data. In view of system load balancing and economic

efficiency, data regeneration should occur on node who is close to average cluster CPU utilization. We can calculate this facto follow the example of Hadoop. Defining p_{cpu} as the node CPU utilization, p_{ccpu} as the average load of cluster, $\beta(0<\beta<1)$ is select range around p_{ccpu}, we can obtain the boundary value of p_{cpu} is $p_{ccpu}(1-\beta)$ and $p_{ccpu}(1+\beta)$ .accordingly, b_2 turns out to be

$$b_2=\begin{cases}1-\dfrac{p_{ccpu}-p_{cpu}}{p_{ccpu}-p_{ccpu}(1-b)} & p_{ccpu}(1-b)<p_{cpu}<p_{ccpu}\\ 1-\dfrac{p_{ccpu}-p_{cpu}}{p_{ccpu}(1+b)-p_{ccpu}} & p_{ccpu}<p_{cpu}<p_{ccpu}(1+b)\\ 0 \quad other & \end{cases} \quad (3)$$

- Credence value b_3 : In order to describe safety degree of nodes, we introduced system security model to distributed storage system and assigned each node a value to describe the degree to which it is trusted [10]. In distributed storage system with autonomous network, nodes may not be credible since there are many anonymous nodes in the system. System security model is necessary for such systems to avoid malicious nodes sneaking into and eavesdropping data. Therefore, we assess credence value as an important indicator of node performance and obtain the value from the security model of system. More concretely, interactive unit of security model has a record for every node in system to record its interaction with other system nodes. The record will feed back to trust management unit when it is updated. We can treat the record as credence value b_3 to evaluate performance of system nodes.
- Network transmission delay b_4 : the indicator is defined to measure the stability and congestion level when transmitting data from one node to another. Generally, network transmission delay between nodes varies greatly for distributed system with high concurrent. It is even worse for large distributed systems with distributed data center because of the change of routing and bandwidth. From the above, the shorter the Network transmission delay, the better performance the system node has. This is a cost indicator and can calculated by

$$b_4=\begin{cases}0 & c_w\geq c_{w\max}\\ 1-\dfrac{c_w-c_{w\min}}{c_{w\max}-c_{w\min}} & c_{w\min}<c_w<c_{w\max}\\ 1 & c_w\leq c_{w\min}\end{cases} \quad (4)$$

Step 2: construct the judgment matrix *A* according to assessment results of evaluation factors. If the number of nodes waiting for evaluating is *n*-1, the number of evaluation factors is *m* (4 index we list in this paper), then the judgment matrix is $A=(x_{ij})_{(n-1)\times m}$.

Step 3: calculate the geometric mapping for assessment results:

$$p_{ij}=x_{ij}\Big/\sum_{i=1}^{n-1}x_{ij} \quad (5)$$

When $p_{ij}=0$, because of no sense of $\ln(p_{ij})$, we specify $\ln(p_{ij})=0$.

Step 4: calculate the information entropy of each evaluation factor:

$$E_j=\frac{-k}{\sum_{i=1}^{n-1}p_{ij}\ln(p_{ij})} \quad (6)$$

In which $k=1/\ln(n-1)$.

Step 5: calculate the weight of evaluation factor:

$$\omega_j=(1-E_j)\Big/\sum_{j=1}^{m}(1-E_j) \text{ and } \sum_{j=1}^{m}(\omega_j)=1 \quad (7)$$

Finally, we get the vector of equipment performance evaluation index $\boldsymbol{\omega}=(\omega_1,\omega_2,\ldots,\omega_m)^T$ (*m* is the total number of index) and the assessment result of nodes A_i .

B. Node-state Awareness Path-finding Algorithm

In this section, we discuss how to construct an optimal path from providers to newcomer according to equipment performance of nodes and available repair bandwidth between two nodes, while taking hop counts into account at the same time.

By calculating accurate transmission reliability of nodes according to equipment performance, available repair bandwidth and hops count, the path finding strategy always choose the optimal next hop to transmit data. Therefore, it can be determined that the path is optimal under all factors.

A regeneration based on node-state awareness path-finding algorithm includes three phases, path construction, data transmission and state maintaining. Path construction is the most important step. In this step, master node of system need to statistics the whole relay nodes from each provider to newcomer and calculate transmission reliability of them. Reliability of a provider could be calculated based on communication cost from it to newcomer. In the following description, we use $Cost(N_i)$ denote communication cost of the provider N_i . Since there are variety of paths from a provider to newcomer, the communication cost of a provider need to take a weighted average of all path. The path-finding algorithm we design is constructed as follow:

Step 1: When a storage node does fail and a newcomer plans to fetch data from providers, master node need to broadcast path construction message to providers who can transfer data to newcomer directly first. A cost field is set for these providers to record their transmission reliability. The cost field is made up by performance assessment result of this provider a_i^{β} , cost of transferring data to newcomer $Cost(N_i)$ which is initialized as 0 and a stack κ_i storing relay nodes from it to newcomer.

Step 2: Update the cost of provider referred in step1. If we mark newcomer as N_0 , cost of provider who can transfer data to N_0 directly could be obtained by:

$$Cost(N_i) = \frac{1}{e_{i0}^{\alpha}} + p \qquad (8)$$

e_{i0}^{α} is transmission bandwidth from provider N_i to N_0 ,p is fault probability of N_i . From the formula, if transmitting data with STAR, the greater the link bandwidth, the smaller the cost will be. This value will store in cost field of provider.

Step 3: when $Cost(N_j) > Cost(N_i)$, in other word, bandwidth from N_j to N_0 is smaller than bandwidth from N_i to N_0 , calculate the cost N_j transmitting data to N_0 through relay node N_i :

$$C_{N_j,N_i} = Cost(N_i) + Metric(N_j, N_i) \qquad (9)$$

In this equation, $Metric(N_j, N_i)$ is the cost from N_j to N_i . It is given by

$$Metric(N_j, N_i) = \frac{1}{e_{ji}^{\alpha} \times a_i^{\beta}} + p \ , \qquad (10)$$

α and β are constants which can measure the influence to provider transmission reliability. The measurement already takes relay property, bandwidth and path length into consideration.

Step 4: Compare C_{N_j,N_i} and $Cost(N_j)$, if $C_{N_j,N_i} > Cost(N_j)$, discard the path with high cost. Otherwise, we choose the path through N_i as the transmission path of N_j .update cost $Cost(N_j)$ to C_{N_j,N_i} and record it in cost field of N_j . Finally, push N_i into κ_j .

Step 5: repeat step3 and step4 until the cost of all providers no longer change.

Finally, we choose the providers who have lower costs to complete regeneration process. Nodes in stack κ_j constitute the data transmission path of each provider.

When data transmitting, providers transmit data to newcomer by encoding data and transmitting coding block to relay nodes recorded in stack. As for state maintenance, we can maintain the connectivity of paths and ensure the activeness of nodes by periodically querying based on flooding.

IV. Analysis And Verification

A. Evaluation result of equipment performance

In order to analyze performance of NARP, we need experiments to assess the node performance evaluation algorithm first. We build a small-scale Hadoop cluster with 9 nodes, one master node and 8 storage nodes. We installed Ubuntu10 and Hadoop0.19.0 respectively as operating system and file system. Collecting performance data of storage nodes and obtain performance evaluation result by MATLAB program.

Reference the idea of Eucalyptus Framework, we choose the best machine in frame as master node to collect performance data of storage nodes in a fixed time interval (5min in this paper). Master node gets node assessment values under the four indexes introduced in Sec III.A and combine the data into decision matrix *A*. *A* will be input of MATLAB program which calculate the node performance results for combination index. Since path-finding simulations need large number of experimental data to simulate distributed storage system with lots of nodes, we need large number of node performance evaluation results for the following experiments. We conducted this experiment of 137 hours and 35 minutes then obtained 1651 sets evaluation result. The result which equivalent to evaluation result of 13208 nodes is counted in Table1. Specially, we choose five sets of typical data and showed them in Figure3. Experiments not only show that the node performance evaluation algorithm based on entropy method can measure performance of nodes well but also reflect the regeneration process introduced in this paper is feasible from the side.

TABLE I. Statistical Result For Nodes Assessment

nodes assessment result	number	percentage
0.75-0.85	5893	44.62
0.75-0.65	2026	15.34
0.65-0.55	3911	29.61
0.55-0.45	619	4.69
0.45-0.15	588	4.45
0.15 以下	171	1.29

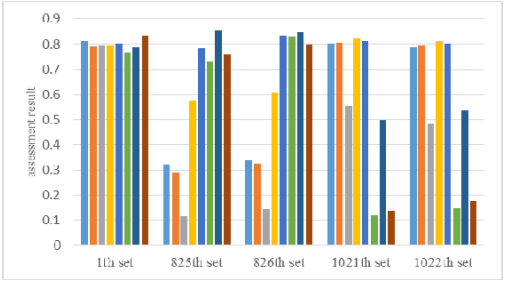
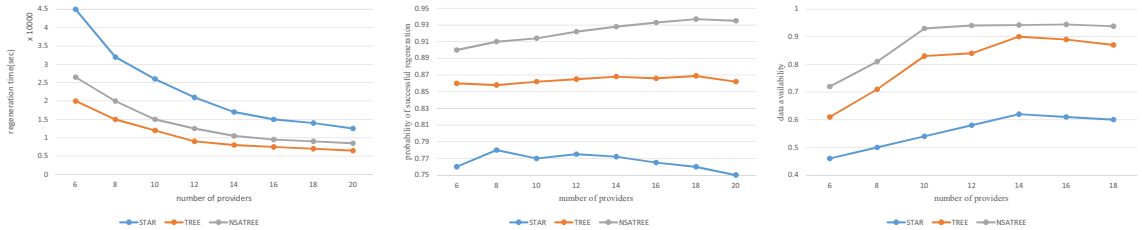


Figure 3. Evaluation results for combination index based on information entropy

B. Empirical studies in dynamic network environment

In this section, we introduce the evaluation result of equipment performance obtained from Sec IV.A into our simulation and compare some metrics which are important to system property. We make experiment to evaluate the effect of the node-state awareness regeneration algorithms based on the real data of available bandwidth [11] and equipment performance measured in PlanetLab [12]. The trace file containing the join/departure behaviors of nodes in PlanetLab is provided by [13]. The repair bandwidth between nodes is detected by the pings repeated every 5 minutes from Jan. 2004 to Jan. 2005. Naturally, the equipment performance of node is set by the result we have obtained from Sec IV.A every 30 minutes. Based the trace file, we run our simulation in an event-driven simulator which simulates a practical distributed storage system in PlanetLab. We compare three algorithms in the simulation.

Figure4 shows the simulation results. In Figure4 (a), we can see that NSA can reduce the regeneration time by 40% compared with STAR. Because the number of links in the network increases with *k*, it is more likely to select the links with higher available bandwidth when *k* is larger. Thus the curves all go down with the increasing of *k*. Compared with TREE, the regeneration time of NSA is a



(a) Regeneration time (b) Probability of successful regeneration (c) Data availability

Figure 4. Simulation results in the dynamic network environment

little longer. On the one hand, the node-state awareness path-finding construction algorithm need more time than building a regeneration tree, on the other hand, the time of nodes performance evaluation we have simulated in Matlab is not counted. However, the curves of NSA and TREE are quite close together with the increase of *k*. They can select the best links to achieve the optimal regeneration time, so the gap between NSA and TREE is not big when *k* is small. When *k* becomes larger, the partial rebuilding probability of NSA will smaller than TREE, so the gap becomes smaller and smaller, as illustrated in Figure1 (e).

We also compare another two important performance metrics of the regeneration in the distributed storage system. One metric is the probability of successful regeneration. The regeneration need to restart when one provider or the newcomer fails during the regeneration. We can see in Figure4(b) that NSA have the probability of successful regeneration more than 90%, while only about 75% of the regeneration processes using STAR and about 85% of the regeneration processes using TREE success. Moreover, when *k* is large, the probability of the successful regeneration process begins to reduce, because node departures are more likely to happen during the regeneration.

We notice in Figure4(c), the file availability of STAR is only about 60%. The file availability is the probability that the file can be recovered. If too many regeneration processes fail, the number of coded blocks cannot be kept always larger than *k*. However, when $k > 14$, the file availability of NSA can be more than 94%. Therefore, we can regard the file as highly available by the NSA regeneration algorithms.

V. CONCLUSION

In this paper, we address challenges in constructing the node-state awareness regeneration in distributed storage systems with regenerating codes. We first analyze the evaluation architecture for equipment in distributed storage system and present the assessment result we have simulate in Hadoop cluster. Based on this analysis, we discuss the node-state awareness regeneration combined with regenerating codes (NSA) and analyze its algorithm implementation scheme. By extensive simulations, we evaluate the performance of the regeneration algorithms we designed using real data measured in PlanetLab. Our assessment results show that NSA is not the fastest scheme, but it has high reliability and stability for regenerating codes especially in large and complex distributed storage system. Therefore, NSA is suitable for distributed storage systems with a substantial degree of bandwidth heterogeneity and nodes heterogeneity.

REFERENCES

[1] Dimakis A G, Ramchandran K, Wu Y, et al. A Survey on Network Codes for Distributed Storage [J]. Proceedings of the IEEE, 2010, 99(3):476-489.

[2] S. Ghemawat, H. Gobioff, and S.-T. Leung, “The Google file system,” in Proc. 19th ACM SIGOPS Symp. Oper. Syst. Principles, Oct. 2003, pp. 29–43.

[3] J. Kubiatowicz et al., “OceanStore: An architecture for global-scale persistent storage,” in Proc. 9th Int. Conf. Architect. Support Programm. Lang. Oper. Syst., Cambridge, MA, Nov. 2000, pp. 190–201.

[4] B. Calder et al., “Windows Azure storage: A highly available cloud storage service with strong consistency,” in Proc. 23rd ACM Symp.Oper. Syst. Principles, 2011, pp. 143–157.

[5] A. G. Dimakis, P. B. Godfrey, M. J. Wainwright, and K. Ramchandran, “Network coding for distributed storage systems,” in Proc. 26th IEEE INFOCOM, Anchorage, AK, USA, May 2007, pp. 2000–2008.

[6] Shum K W, Hu Y. Cooperative Regenerating Codes [J]. IEEE Transactions on Information Theory, 2013, 59(11):7229-7258.

[7] Y. Wu and A. Dimakis, “Reducing Repair Traffic for Erasure Coding based Storage via Interference Alignment,” in Proc. IEEE Int. Symp. Inform. Theory (ISIT), 2009.

[8] Li J, Yang S, Wang X, et al. Tree-structured data regeneration in distributed storage systems with regenerating codes[C]//INFOCOM, 2010 Proceedings IEEE. IEEE, 2010: 1-9.

[9] Vazhkudai S, Schopf J M, Foster I T. Predicting the performance of wide area data transfers[C]// Parallel and Distributed Processing Symposium. Proceedings International, IPDPS 2002, Abstracts and CD-ROM. IEEE, 2002:10 pp.

[10] 王兵. 用于分布式系统的安全模型[J]. 硅谷, 2011(4):34-34.

[11] S. Banerjee, S.-J. Lee, P. Sharma, and P. Yalagandula. S3 (Scalable Sensing Service). [Online]. Available: http://networking.hpl.hp.com/scube/PL/

[12] [Online]. Available: http://www.planet-lab.org/

[13] J. Stribling. Planetlab All Pairs Ping. [Online]. Available:http://infospect.planet-lab.org/pings