Network Measurement based Redundancy Model and Maintenance in Dynamic P2P Storage Systems

Guangping Xu, Hua Zhang
The Key Laboratory of Intelligence Computing & New Software Technology, Tianjin
Tianjin, China
xugp2008@yahoo.com.cn

Jing Liu, Gang Wang, Xiaoguang Liu
Information Technology Science College
Nankai University
Tianjin, China
jingliu@nankai.edu.cn

***Abstract*—Peer-to-peer distributed storage systems aggregate the storage space of many peers spread over the Internet. Due to the dynamic and scalable nature of these systems, it is a challengeable issue to access data in an available and reliable way through redundancy. Following the modeling methodology presented in [1], we present the stochastic model to analyze redundancy evolution of these systems under churn. Different from the previous work based on the average peer availability, the stochastic model can be applied into the practice based on both conditional probabilities (α, θ) which can be obtained from network probing easily. First, we apply the model to characterize the redundancy evolution of a fragment system with temporary churn. And we use an empirical trace and a synthetic trace to validate the model. Second, based on the characteristics of different churn from the both probabilities, we propose the redundancy maintenance strategy assisted by network sampling. Our simulations evaluate the performance of the strategy driven by empirical and synthetic traces.**

***Keywords*—Peer-to-peer Storage Systems; Availability; Model; Redundancy; Maintenance**

I. INTRODUCTION

Peer-to-Peer (P2P) systems are widely used to share resources on Internet including publishing files and sharing disk space applications. In these applications, data availability is the primary concern because peer dynamics (i.e., churn) is a prevalent phenomenon. Some P2P storage prototype systems have been designed and implemented, such as CFS[9], PAST[12], OceanStore[11], TotalRecall[8] and so on. Data redundancy is an indispensable part to ensure data available and persistent against individual failures for these applications. All the peers responsible for the redundancy of an object are called a redundancy system. It is a challenging issue to design and maintain the redundancy system in P2P dynamic networks.

Much research work has been presented on the availability of these systems. Most of them are based on the average peer availability and it is hard to accurately capture the time evolution of the storage systems [13, 10]. And there was much work that analyzed the performance the systems under churn [14, 15, 17]. Moreover, network measurement work motivates us to analyze the dynamics of P2P storage systems to gain deeper understanding of the effects of various churn on redundancy parameters choice and determination. Peer availability is estimated based on peer-online status sampling every a fixed time interval in network measurement. For example, Overnet [16] trace dataset is obtained every twenty minutes. Therefore, it is reasonable to use discrete-time stochastic process to model the behavior of peers in these systems.

As one contribution of this paper, we utilize a stochastic model to analyze the evolution of a redundancy system under churn from a novelty perspective. Different from the previous work based on the average peer availability, this model let online peers remain online with a probability α and offline peers rejoin the system with a probability θ from one time to next. These probabilities of the model can be get based on the sampled on/off states of peers. Following the modeling methodology presented in [1], we analyze the time-evolution characteristics of the peers in every redundancy system with the model. We use an empirical trace dataset [2] and a synthetic trace to validate the model.

As the other contribution, our maintenance strategy is proposed to enhance data availability. The strategy replaces some selected peers based on their churn characteristics reflected by both the probabilities. Our simulations evaluate the performance of the strategy driven by empirical and synthetic traces. The results show that our strategy can at least reach the expected availability of the model.

The rest of the paper is organized as follows. In Section II, we introduce some related work. In Section III, we provide the preliminary background and some notions. In Section IV, we proceed to describe the system model and validate it based on the synthetic and empirical traces. In section V, we propose our maintenance strategy and evaluate it. Finally, we conclude the paper in section VI.

II. RELATED WORK

A. Churn and Redundancy

In past several years, through some research efforts for data lookup and location mechanism in wide area networks, such as various DHTs, the networks tend to provide a stable, reliable service for data routing and locating. A number of P2P storage

The paper in part is sponsored by NSF of China (No.90612001), 863 program (No. 2008AA01Z401) and Tianjin NSF(No. 08JCYBJC13000).

systems [8,9,11,12] are designed as large-scale systems built with dynamic peers in these networks.

Recently more studies focus on enhancing data availability impacted by the dynamic behaviors of participated peer, i.e. churn. Churn as an inherent property of these systems is classified into temporary churn and permanent churn [18, 14]. In temporary churn, a peer fails over a period of time and then returns without losing its stored data. But in permanent churn, a peer is failed permanently resulting in its stored data lost. So these P2P storage systems need to employ some form of redundancy to mask or hide the impact from temporary churn and some maintenance scheme to repair lost data from permanent churn. Existing systems [8,9,11,12,22] usually utilize replication and erasure coding as redundancy strategies.

This paper assumes that a (*m,n*) erasure code is adopted. That is, originally *n* encoded distinct fragments for an object are stored on *n* independent peers, and then if any *m(<n)* fragments available, the object can be reconstructed. Replication can be regarded as a special case *(n, 1)* of erasure code. We call the redundancy system for the encoded fragments a fragment system in the later discussion.

B. Redundancy Placement and Maintenance

For placing the redundancy in these systems, two common replica placement strategies [14], random placement and selective placement, are employed. In random placement, the responsible peers of redundancy are determined by a set of known re-hash functions. Implementing this random placement is straightforward, scales well as the number of hosts increase, and also fits well with the way current peer-to-peer systems like Chord and Pastry operate. In selective placement, the responsible peers are chosen based on some prediction algorithms on the peer lifetime, capacity etc.

In both the strategies, it can be regarded that the peers are uniformly random selected in the all peers set or part peers set. Therefore, the peers responsible for the redundancy can be considered independent with each other.

Some redundancy maintenance strategies have been proposed to repair the lost data due to permanent failure. These strategies are classified into proactive maintenance (such as in [4]) and reactive maintenance (such as in [8,15]) according to the different repair occasion. The maintenance strategy should make a decision to choose a proper peer to store the recovered fragment. Among these maintenance strategies, the common approach is to randomly place the fragment.

In the paper, the proposed strategy samples online/offline states of the peers responsible for the fragments of an object periodically. And then it replenishes periodically the redundancy by recovering new fragments at some other online peers based on the behavior characteristics of each peer reflected by both state-transition probabilities.

C. Redundacny Models

Previous work usually modeled the redundancy system from a static perspective. For example, in [10,13,23], they provided similar models to compare the efficiency between replication and erasure encoding.

Recent work modeled redundancy under churn from the process perspective. In [14], D.Wu et al modeled the evolution of a fragment system based on stochastic differential equations and derived closed-form terms for different maintenance strategies under different churn. In [6] T.Li et al developed queuing models for P2P storage systems.

In [19, 20], they used a simple Markov chain model with continuous-time to derive the lifetime of the replicated state and to optimally choose system parameters. In [21], replication model was introduced to analyze data duration for various heavy-tailed distributions of peer lifetime.

The closest analog to our work is that of Datta et al [15]. They used a Markov model with discrete-time parameters to derive the time evolution to facilitate analyzing their randomized lazy maintenance.

Our modeling work is inspired by work in system reliability theory [1], which uses a property of Vandermonde matrices to examine the reliability of an *n-component* system of service. Following the modeling methodology [1], we propose a stochastic model to analyze the evolution of a fragment system under churn. Based on the model, then we propose a redundancy maintenance strategy assisted by network sampling. Moreover, both empirical trace and synthetic trace are used to validate the model.

III. PRELIMINARY

This section describes the preliminary including peer availability measurement and some related notions in our model. Based on the observation that, in real P2P storage systems, the peer availability measurement is done at a certain time interval, it is convinced that the discrete-time assumption in our model and in the simulation is reasonable.

A. Network Measurement

To ensure required availability, P2P storage systems need to monitor the connectivity of the overlay and maintain the availability of stored objects. The premise of the operations is the peer availability measurement. There are many empirical availability measurements in distributed systems.

Most of the traces used active network probing to determine whether they are available in the systems or not every a certain time interval. TABLE.I shows the traces of some real distributed systems. Bolosky et al described the uptimes of over 50,000 PCs in Microsoft Corporation [2]. Guha et al studied a set 4000 peers in the Skype network by sending an application-level ping every 30 minutes for one month [3]. Stribling measured pings every 15 minutes between all pairs of 200-400 PlanetLab peers [5]. Bhagwan et al studied peers availability in Overnet [16]. As listed in TABLE.I, it shows some characteristics of these traces including system scales, observed durations and probing time-intervals.

TABLE I
TRACE DATASETS OF SOME REAL DISTRIBUTED SYSTEMS

Traces	System Scale	Observed Duration	Probing Interval
Microsoft PCs	51,662	35 days	1 hour
Skype	4000	1 month	30 minutes
PlanetLab	3000	6 months	15 minutes
Overnet	200-400	7 days	20 minutes

Therefore, it is reasonable to treat the time *t* in our model and the trace-driven experiments as *discrete time* that is a natural and reasonable assumption. The states of peers in an overlay network dynamically change over time. As recorded in these empirical traces, the participating peers have the following the states: *online state (1)* and *offline state (0)*. It can be observed that every peer changes its state between online and offline during its evolution.

Formally, for a peer *i*, let $X_i(t)$ denote the peer state at time *t*. At each sampling time instance *t*, each peer is denoted as *0* if the peer is offline and *1* if it is online.

$$X_i(t) = \begin{cases} 1 & \text{peer } i \text{ is online at time } t \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

B. Fragment System

To support the object availability, a *(n, m)* erasure code is adopted to provide data redundancy for an object. Originally *n* distinct fragments are stored in the network, and at any time moment *t*, any at least *m* distinct fragments ensure object reconstruction (i.e., $N(t)>=m$). Each peer is responsible for only one fragment of an object. As defined the term in [14], for a specific object *o*, the set of peers that hold a fragment of an object is called *the fragment system of object o*, which is denoted by *S* (i.e., the size of a fragment system is *n*, $|S|=n$). The set of online peers of a fragment system is defined as $S_{on}(t)$ which size is denoted as $N(t)$, $|S_{on}(t)| = N(t)$.

$$N(t) = \sum_{i=1}^{n} X_i(t) \quad (2)$$

As stated in Section II.A, we can make an assumption that the peers in a fragment system are independent. The churn pattern of each peer can be considered as an online/offline alternating process. In online state, the peer is present in a fragment system and the fragment stored on it is available. In offline state, the peer is not present due to temporary failures and we assume that the fragment still exists in the peer's storage space. The parameters *n* and *m* should be properly configured to mask temporary churn and ensure a required level of availability, say 99.9%, for the fragment system. If a peer is permanent failed in a fragment system, the fragment stored on it will be lost and $N(t)$ will be degraded. Our model concerns some characteristics of the evolution of $N(t)$, which is analyzed in Section IV. And our proposed maintenance strategy can recover the lost fragment to some selective peer based on the parameters in the model in Section V.

IV. MODEL

Following the modeling methodology presented in [1], this section models the convergence state evolution of peers in *S* with temporary churn. We validate the results of the model by the simulations driven by both empirical trace and synthetic trace.

A. Assumptions

As stated in Section III.A, the time interval between consecutive state probes in each peer evolution is fixed, and the peer is sampled as discrete state, online (*1*) or offline (*0*). Also, for any redundancy placement, as stated in Section II. B, the transition of any peer between the states of being online and offline is independent of the other peers. Therefore, the following more realistic assumptions are given to model the peer behavior at time *t*.

- Time, measured in discrete units, is identified with the set of positive integers. At initial time *t=0*, all peers in *S* are online. That is, $X_i(t=0)=1$ where $i = 1…n$.
- The probability of an online peer remaining online from one time to the next is fixed and denoted by α. $P\{X_i(t+1)=1 \mid X_i(t)=1\} = \alpha$. For simplicity, the complementary probability is abbreviated as $\underline{\alpha} = 1-\alpha$
- The probability of an offline peer remaining offline from one moment to the next is fixed and denoted by θ. $P\{X_i(t+1)=0 \mid X_i(t)=0\} = \theta$. Let $\underline{\theta} = 1-\theta$.

The probabilities (α, θ and their complements) can be obtained from trace data in which peer state is periodically sampled by network probing. Consider a probing state-path of a peer that alternates between online and offline state. According to the probing state-path for each peer, all state transitions are counted and the average transition probabilities are obtained. As a simple example, the peer probing state-path is 1111001110001111 for peer *i*. The transition count from $X_i(t) =1$ to $X_i(t+1) =0$ is 2, denoted as $tc_{0,1}=2$. Other state transition counts are $tc_{1,1}=8$, $tc_{0,0}=3$ and $tc_{1,0}=2$. Thus $\alpha_i = 0.8$ and $\theta_i = 0.6$.

From the intuition, the transition counts reflect different characteristics of different types of churn. If temporary churn is dominant with peer *i*, then the state transition counts of $0 \rightarrow 1$ and $1 \rightarrow 0$ will be large proportions of all state transition counts. It means that α_i and θ_i both are relatively small for the peer. If peer *i* tends to successively stay online, then α_i increases for an online peer *i*. Reversely, if θ_i increases for an offline peer *i*, then the peer tends to successively stay offline and may be permanent failed if θ_i exceeds a threshold. Based on the different characteristics of different types of churn reflected by the probabilities, our redundancy maintenance strategy is proposed in Section V.

For the *n* peers in *S*, the average probabilities obtained by approximately equal to α and θ. We randomly select different numbers of peers (i.e., the size of a fragment system, $|S|=n=10$,

20, 50 and 100, respectively) responsible for the fragments of an object from the trace of Microsoft Corporation [2] and obtain α and θ of these selected peers for *1000* time units in the trace. The process is repeated 1000 times (i.e., 1000 distinct fragment systems); then α and θ are obtained. Fig.1 plots the median, the 5th and the 95th percentiles of the probabilities (α and θ) for different sizes of a fragment system. As expected, it can be seen that the parameters tends to fluctuate in a narrow range with the increasing size of fragment system. The medians of α (range from *0.9954 to 0.9959*) and θ (range from *0.9609 to 0.9640*) are almost constant values for these fragment systems. The 5th and the 95th percentile values are closer to the medians as the size of fragment system increases. The results validate our assumptions that the probability of an online peer remaining online and the probability of an offline remaining offline are constant values. Moreover, the system scale (i.e., the number of peers) in [2] is stable and temporary churn is dominant in the observed duration. Therefore, we use the empirical trace and a synthetic trace generated with $\alpha = 0.995$ and $\theta = 0.96$ to validate our model.

B. Model

The assumptions described in Section IV.A allow the theory of Markov chains to be held. Moreover, according to the trace of Microsoft [2], the assumptions of the probability pair (α , θ) are grounded in reality. For a (*m*, *n*) fragment system, it takes the number of online peers, *N(t)*, i.e., the number of available fragments, to be the state of the fragment system at each discrete-time *t*. At any time point *t*, if the number of online peers *N(t)>=m*, object *o* is available; otherwise, the object is unavailable.

The transition probability $p_{i,j}$ of transiting from state *j* to state *i* in one time unit is given by

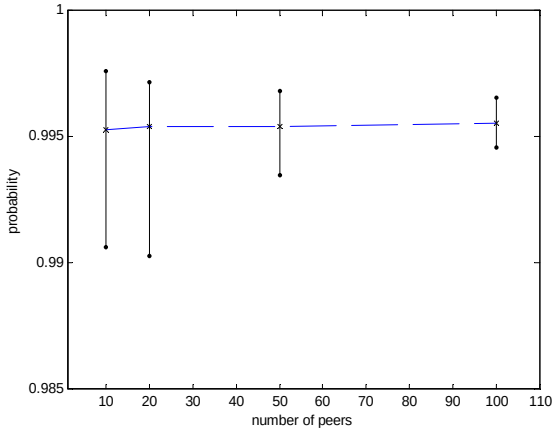
$$p_{i,j} = \sum_{l=0}^{i} \binom{j}{l} \binom{n-j}{i-l} \alpha^{l} \underline{\alpha}^{j-l} \theta^{n+l-i-j} \underline{\theta}^{i-l} . \quad (3)$$

The transition from state *j* to state *i* happens when *l* of the *j* online peers remaining online and *(i-l)* of the *(n-j)* offline peers back online from current time instance to next one, and $l \in [0, i]$ and $i \in [0, n]$. In the formula (3), state *i* and *j* of a fragment system are symmetrical, and *i* may be larger than *j*. Thus the combination number (*j*, *l*) is defined 0 if *l>j*.

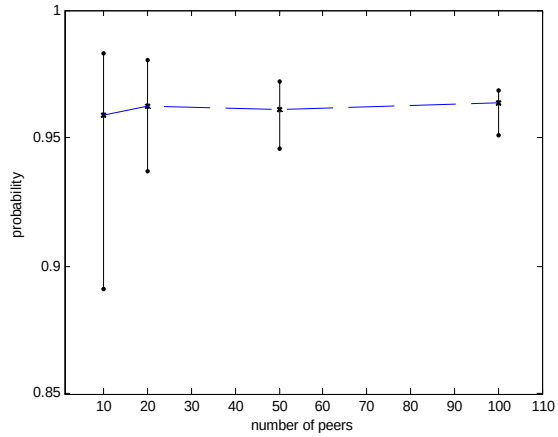
According to the Markov chain with $p_{i,j}$, we present the following results aimed at measuring the object availability in a fragment system.

1) Expectation of N(t)

At initial time *t=0*, the fragments of a specific object are stored on *n* independent peers. After *t* time units, the number of available fragments determines the availability of an object. We use the expectation of *N(t)*, denoted as $\overline{N}(t)$ to describe the evolution of the fragment system. The multi-step transition probability from state *j* to state *i* in *t* time units is denoted as $p_{i,j}^{t}$ which is important in deducing $\overline{N}(t)$. The transition



(a) α



(b) θ

Fig.1 the 5th and the 95th percentile and the median values for α and θ .

matrix *P* can be represented

$$P = (p_{i,j})_{0 \le i,j \le n} = \begin{bmatrix} \theta & \underline{\alpha} \\ \underline{\theta} & \alpha \end{bmatrix}_{n} \quad (4)$$

which is a specific class of matrices, Vandermonde matrices. The property that the product of two Vandermonde matrices is Vandermonde facilitates to multiply, invert and diagonalize our transition matrix *P*. Thus, the multiple-step transition matrix can be obtained and further the proposition can be obtained based on the generating function at time *t*.

Proposition 1: *For a fragment system (|S|=n), the expected number of available fragments at time t is given by*

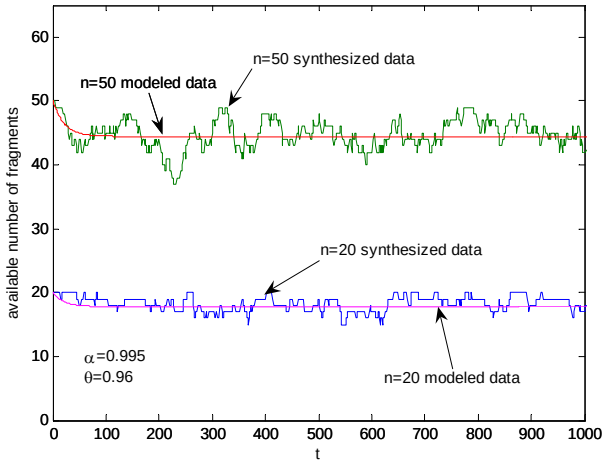
$$\overline{N}(t) = n(\underline{\theta} + \underline{\alpha} q^{t}) / (1 - q) \text{ , where } q = \theta + \alpha - 1 . \quad (5)$$

Due to $1 - q = \underline{\theta} + \underline{\alpha}$, the result of $\overline{N}(t)$ can also be $n(\underline{\theta} + \underline{\alpha} q^{t}) / (\underline{\theta} + \underline{\alpha})$. The restriction of $0 < \theta + \alpha < 2$ follows that $|q| < 1$. The corollary of this proposition is given as follows.

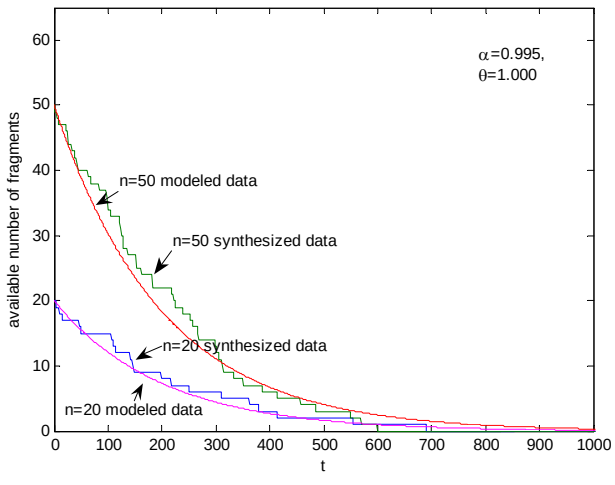
Corollary 1: *As a fragment system (|S|=n) tends to stable (i.e., time t tends to infinite), the expected number of available fragments tends to* $n\underline{\theta} / (1 - q) = n\underline{\theta} / (\underline{\alpha} + \underline{\theta})$. *That is, for a long run, the expected fraction of the available fragments in a fragment system is* $\underline{\theta} / (\underline{\alpha} + \underline{\theta})$.

Proposition 1 and its corollary reflect the quality of convergence of available fragments. It is easily seen to be correct at *t=0* that the fragment system begins in state *n*. The speed of convergence to the expected available fragments is only determined by |*q*|. If the peers in the fragment system have more frequent temporary churn (i.e., $\alpha, \theta \to 0$), the state evolution of a fragment system shows the oscillating convergence to the expected number of available fragments. If the peers tend to stable (i.e., $\alpha, \theta \to 1$), the state evolution shows a monotonous decrease towards the expected number of available fragments. The evolution takes a longer time in the two extreme cases than other cases.

In Fig.2, the modeled curves show the state evolutions of different fragments systems with various values of *n* and (α, θ). In Fig. 2(a), the smooth curves depict the modeled results with $\alpha = 0.995$, $\theta = 0.96$ for the fragment systems (*n=20*, *50*) according to Proposition 1. The fragment system tends towards stable fast due to the effect of temporary churn and



(a) $\alpha = 0.995$, $\theta = 0.96$



(b) $\alpha = 0.995$, $\theta = 1.000$

Fig. 2 the expected number of available fragments with different parameters

the expected number of online fragments tends to a constant. In Fig. 2(b), the smooth curves depict the modeled results with $\alpha = 0.995$ and $\theta = 1$ for the fragment systems (*n=20* and *50*) according to Proposition 1. It is an extreme case that the failed peer doesn't return the fragment system due to $\theta = 1$, i.e., the failure is permanent. The redundancy level of the fragment system gradually decreases towards all fragments lost over time.

Moreover, the synthetic traces of the fragment systems (*n=20, 50*) with $\alpha = 0.995, \theta = 0.96$ and $\alpha = 0.995, \theta = 1$, are generated respectively. For both synthetic traces, at each time *t*, let online peers remain online with a probability α and offline peers remain offline with a probability θ. In Fig.2(a) and Fig.2(b), every fluctuating curve is a synthetic state-path instance. That is, the state evolutions of fragment systems are plotted over time. It can be seen that every synthetic state-path fluctuates along with the corresponding modeled curve.

2) Deviation of N(t)

Proposition 1 only presents the expected available number of fragments in a fragment system. It is also important to determine the extent of deviation from the expected curve. That is, for a long run, it need to be concerned that how often a given state is visited for the fragment system and the expected availability for the fragment system.

Proposition 2: *As time t tends to infinite, the expected fraction of time for which i-out-of-n fragments available approaches*

$$f(i) = \binom{n}{i} \frac{\underline{\alpha}^{(n-i)} \underline{\theta}^{i}}{(\underline{\alpha} + \underline{\theta})^{n}}. \qquad (6)$$

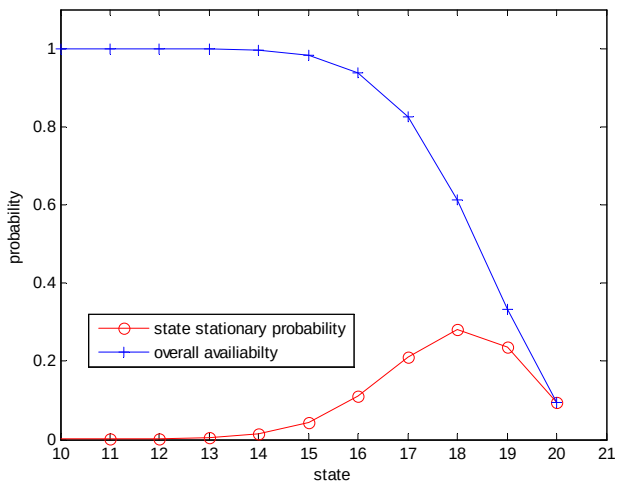
Proposition 2 provides the expected fraction of time for each state of a fragment system. For a long run, we can know the expected time to stay at each state during a period. From another point of view, Proposition 2 means the deviation from the expected number of available fragments, $\overline{N}(t)$, for each state of a fragment system. The expected fraction of time of *i-out-of-n* fragments available can be seen as the stationary probability distribution of the states in the fragment system.

Corollary 2: *As time t tends to infinite, the probability distribution of the states in a fragment system is f(i) and the expectation of the states in the fragment system is given by*

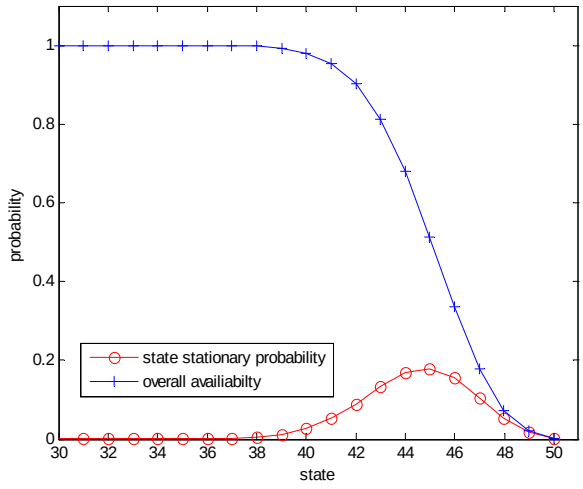
$$\sum_{i=0}^{n} i f(i) = n\underline{\theta} / (\underline{\alpha} + \underline{\theta}) \ , \ i \in \{0,1,...,n\}. \quad (7)$$

The expectation of the states in a fragment system in Corollary 2 is equivalent to the result in Corollary 1. Fig.3 shows the expected fraction of time for the states of the fragment systems with *n=20, 50* and $\alpha = 0.995$, $\theta = 0.96$, respectively.

In our defined fragment system, if an object is available, at least *m* fragments are available. We need to determine the expected fraction of time for which at least *m* fragments out of *n* ones are available. As a straightforward result of Proposition 2, we obtain the following corollary.

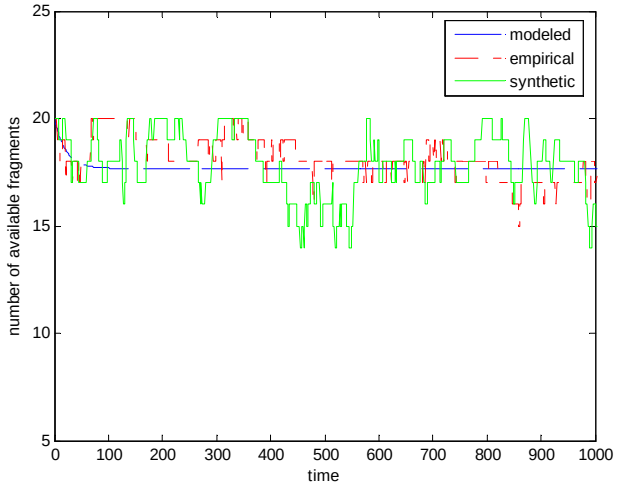


(a) n=20

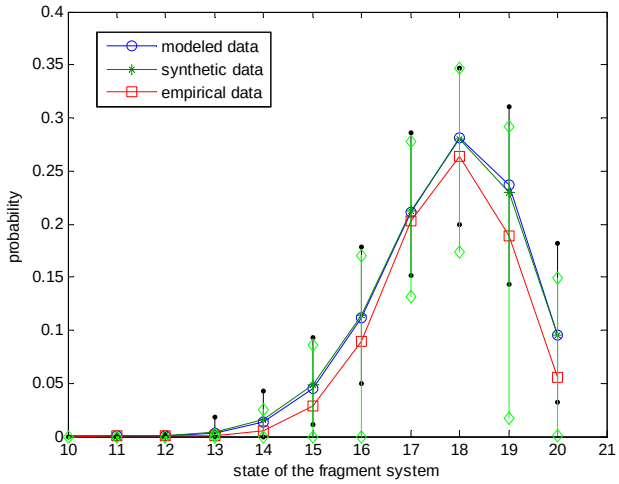


(b) n=50

Fig.3 the expected fraction of time for the states of the fragment systems and overall availabilities with probability pair ($\alpha = 0.995$, $\theta = 0.96$).



(a)



(b)

Fig. 4. (a) the evolutions of a fragment system. (b) the expected probabilities of the states of 1000 fragment systems.

Corollary 3: *For the scenario that the fragment system (n, m) has tended to stable, its expected fraction of time available, i.e., the overall availability of an object, is given by*

$$A = \sum_{i=m}^{n} f(i) = \sum_{i=m}^{n} \binom{n}{i} \frac{\underline{\alpha}^{(n-i)} \underline{\theta}^{i}}{(\underline{\alpha} + \underline{\theta})^{n}}. \qquad (8)$$

Fig. 3 also plots the overall availabilities with the different the erasure coding parameter *m* which is represented as *x*-axis values. This corollary is helpful how to configure the parameter *m* for the required overall availability of the fragment system. For example, if the required availability is at least *99%*, then *m* should not be more than *14* for *n=20*, shown in Fig. 3(a) and *m* should not be more *38* for *n=50*, shown in Fig. 3(b).

To validate the results, we again follow the parameters of a fragment system (i.e., *n=20* $\alpha = 0.995$ and $\theta = 0.96$) to generate the synthetic trace of 1000 distinct objects in 1000 time units period. Moreover, we randomly select a set of peers to store the fragments of 1000 objects from the empirical trace [2]. Fig. 4(a) shows the evolutions of a synthetic fragment system instance and an empirical instance. Observe that the two instance curves accord with the modeled curve well. Fig. 4(b) plots the expected probabilities of the states from the model and the synthetic traces of 1000 fragment systems. It also plots the 5th, the mean and the 95th percentile of time of the states for these fragment systems that show the extents of deviation from the model. We observe that the means are close to the modeled data according to Proposition 2. But the percentile values deviating from their expectations are different for both synthetic data and empirical data. The deviations of the empirical traces are much larger than the deviations of the synthetic traces. The reason is that the online/offline durations of the peers are much more various in the empirical trace than in the synthetic traces. The synthetic traces are generated according to the average probabilities that results in the behaviors of these peers are identical mostly. Even so, if the overall level of availability is required at least *99%* for a fragment system, then *m* could be set so less than 14.

V. MAINTENANCE

A. Maintenance strategy

Our proposed maintenance strategy is based on peer probing. In the strategy, the peers in a fragment system are probed not only for determining the probabilities of the state transitions, but also for determining the replaced peers in a fragment system. The maintenance strategy works as follows:

STEP 1: the average probabilities (α, θ) of the network are determined for any S(m, n) by multiple samplings every T time units interval.
STEP 2: for peer i in S(m, n), the state transitions are counted every T time units and then $\alpha(i,T)$ and $\theta(i,T)$ are obtained.
STEP 3: the comparison between ($\alpha(i,T)$, $\theta(i,T)$) and (α, θ), is done, and then the decision is made whether peer i is replaced.
STEP 4: if peer i is replaced, the maintenance operation replaces with other online peers randomly or selectively.
STEP 5: repeat STEP 2 every T time units.

As stated in Section IV.A, the increased θ of an offline peer may lead to permanent failure. In our maintenance, if $\theta(i,T)$ of an peer *i* is not less than θ and $\alpha(i,T)$ is less than α, peer *i* would be replaced with an online peer randomly. It means that the replaced peers are failed a large proportion of the period that are suspected as vulnerable peers including permanently failed peers to replace. Notably, if a peer failed permanently has the state-path: 111…1000…, the peer may not be replaced due its larger $\alpha(i,T)$ in one period; but it is certain to be replaced in the next period.

Moreover, if $\alpha(i,T)$ and $\theta(i,T)$ are relatively smaller than given thresholds, it means that peer *i* frequently fails and rejoins in the period. Our maintenance can select the peers to be replaced if their evolutions are dominant with temporary churn, which can be an optional part as the strategy. Due to the space limitation, we do not discuss it in the experiments.

B. Evaluation

To evaluate our maintenance, the experiments use Overnet trace [16] because it has more permanent churn than Microsoft trace [2].

1) Experiments

With the simulation driven by Overnet trace, we sample *1000* distinct fragment systems (*n=32)* to acquire the average probability pairs (α,θ) within different *T*s. The results are (*.87541, .65403*), (*.86639, .78835*), (*.86691, .80515*) and (*.86716, .82224*) for *T*= 12, 48, 84 and 120 respectively. It can be seen that θ increases gradually while α almost keeps invariable with the increase of sampled time-lengths resulting in the deceased $\bar{N}(t)s$ due to the effects of accumulated permanent failures with increasing *T*. Our maintenance can replenish the lost fragments and pulls up the downwards curves periodically to the modeled level. As an example, the evolutions of a fragment system with maintenance and without maintenance are plotted in Fig.5, for *T=12*.

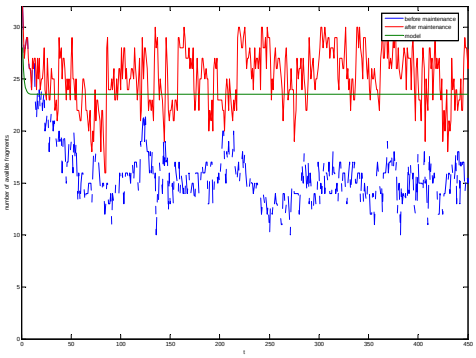


Fig.5. the evolution instance with *T*=12 evolution (n=32)

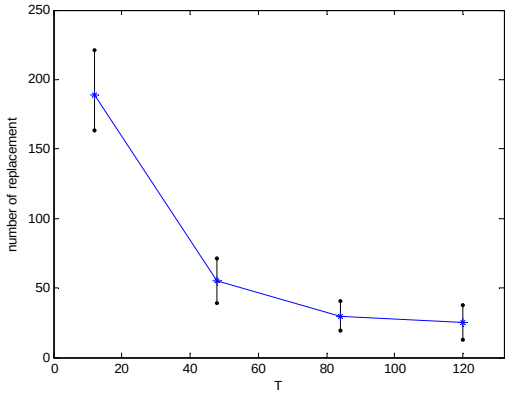


Fig.6.the mean and the 1st and 99st percentiles of the number of replaced peers for a fragment system.

2) Result Analysis

The experiments maintain 1000 fragment systems to evaluate the metrics including bandwidth usage and available fragment distribution of a fragment system with different *T*s.

First, we evaluate the bandwidth usage. The bandwidth usage is determined by the number of replaced peers. Fig.6 plots the mean and the 1st and 99th percentiles of the number of replaced peers for a fragment system in the whole trace duration for different *T*s. Observe that the expected numbers of replacements and the variations of the number of replacements are decreased with linearly increased *T*. The reason for the observation is that the failed peers are lazily replaced for a longer period *T*; but the fragment system may become more vulnerable to temporary failures subsequently. Otherwise, some temporary failed peers may be treated permanent failure wrongly and eagerly replaced if *T* is small that results in much amount of bandwidth consumed.

Next, we evaluate the availability of a fragment system which has relation with the probability distribution of states in a fragment system. The probability distribution of the number of available fragments in a fragment system (*n=32*), as shown in Fig.7 (a) with different *T*s. From the plots, the observed trend is the strategy maintains much larger number of fragments available with smaller *T* for a fragment system. The

largest number of fragments is maintained for *T=12* that *m<=8* to achieve 99% availability while the least redundancy is maintained for *T=120* that *m<=16* to achieve 99% availability.

Finally, to validate the model stated in Section IV, Fig. 7(b) plots there curves of the distributions of states for a fragment system which are obtained from our simulation driven by the synthetic trace (i.e., it accords with the model), the empirical trace with and without our maintenance, respectively. For the concision of the figure, it only plots the curves for *T=12*. Others have the similar results. The more the curve shifts leftwards, the more vulnerable the fragment system is. Therefore, from the positions of the curves, it is clear that our maintenance can be achieved at least the modeled availability according to Corollary 1.

VI. CONCLUSION

Based on the observation that, in real P2P storage systems, the peer availability measurement is done at a certain time interval, this paper has presented a discrete time Markov chain model for analyzing the redundancy evolution of the P2P storage system. The stochastic model can be used to analyze redundancy evolution of dynamic P2P storage systems. And this paper proposes a redundancy maintenance strategy assisted by network sampling. Both empirical trace and synthetic trace are used to validate the model and evaluate the maintenance strategy. As future work, we focus on comparing the maintenance strategy with existing maintenance strategies.

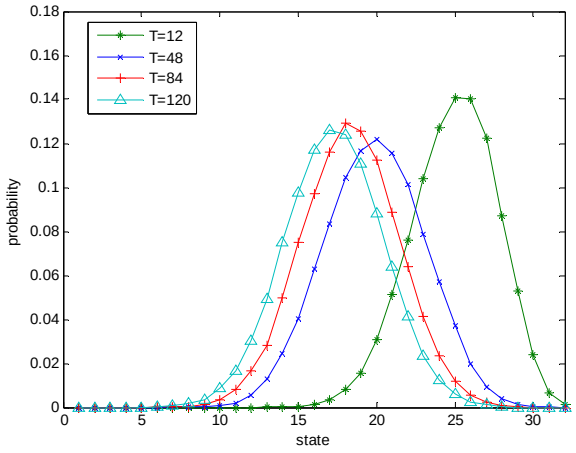


Fig.7(a). the probability distribution of the number of available fragments in a fragment system (*n=32*).

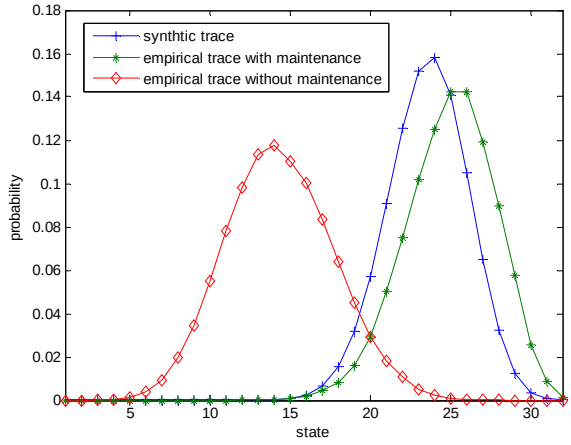


Fig.7(b). The availability comparison for the synthetic trace, the empirical trace with and without our maintenance. (*n=32,T=12*)

REFERENCES

[1] D. Rawlings and L. Sze. On the Reliability of an n-Component System. In *Stochastic Models*, 22(2), 2006, pp.333-339.

[2] W. Bolosky, J. Douceur, D. Ely, and M. Theimer. Feasibility of a serverless distributed file system deployed on an existing set of desktop PCs. In Proceedings of *SIGMETRICS*, 2000.

[3] S. Guha, N. Daswani, and R. Jain. An Experimental Study of the Skype Peer-to-Peer VoIP System. In Proceedings of *IPTPS* , 2006.

[4] E. Sit, A. Haeberlen, F. Dabek, B. Chun, H. Weatherspoon, R. Morris, M.F. Kaashoek, and J. Kubiatowicz. Proactive replication for data durability. In Proceedings of *IPTPS*, 2006.

[5] J. Stribling. All-pairs PlanetLab Ping Data. [online] http://pdos.csail.mit.edu/~strib/pl_app/.

[6] Taoyu Li, Minghua Chen, Dah-Ming Chiu, Maoke Chen. Queuing Models for Peer-to-peer Systems. In Proceedings of *IPTPS*, 2009.

[7] B. Godfrey. Repository of Availability Traces. [online] http://www.eecs.berkeley.edu/~pbg/availability/.

[8] R. Bhagwan, K. Tati, Y-C. Cheng, S. Savage, and G. M. Voelker. TotalRecall: System support automated availability management. In Proceedings of *NSDI*, 2004.

[9] F. Dabek, M.F. Kaashoek, D. Karger, R. Morris, and I. Stoica. Wide-area cooperative storage with CFS. In Proceedings of *SOSP*, 2001.

[10] W. Lin, D. Chiu, and J. Lee. Erasure code replication revisited. In Proceedings of *P2P*, 2004.

[11] J. Kubiatowicz, D. Bindel, Y. Chen, P. Eaton, D. Geels, R. Gummadi, S. Rhea, H. Weatherspoon, W. Weimer, C. Wells, and B. Zhao. Oceanstore: An architecture for global-scale persistent storage. In Proceedings of *ASPLOS*, 2000.

[12] A. Rowstron and P. Druschel. PAST: A large-scale, persistent peer-to-peer storage utility. In Proceedings of *HOTOS*, 2001.

[13] R. Rodrigues, B. Liskov. High availability in DHTs: Erasure coding vs. replication. In Proceedings of *IPTPS*, 2005.

[14] D. Wu, Y. Tian, K.W. Ng, and A. Datta. Stochastic analysis of the interplay between object maintenance and churn. In *Computer Communications*, 31, 2008, pp.220-239.

[15] A. Datta and K. Aberer. Internet-scale storage systems under churn -A steady state using Markov models. In Proceedings of *P2P*, 2006.

[16] R. Bhagwan, S. Savage, and G. Voelker. Understanding availability. In Proceedings of *IPTPS*, 2003.

[17] B. Godfrey, S. Shenker and I. Stoica. Minimizing Churn in Distributed Systems. In Proceedings of *SIGCOMM*, 2006.

[18] K. Tati and G.M. Voelker. On object maintenance in peer-to-peer systems. In Proceedings. of *P2P*, 2006.

[19] S. Ramabhadran, J.Pasquale. Analysis of Long-running Replicated systems. In Proceedings of *INFOCOM*, 2006.

[20] B. Chun, F. Dabek, A. Haeberlen, E. Sit, H. Weatherspoon, M.F. Kaashoek, J. Kubiatowicz and R. Morris. Efficient replica maintenance for distributed storage systems. In Proceedings of *NSDI*, 2006.

[21] G. Xu, W. Ma, G. Wang and J. Liu. Churn Impact on Replicated Data Duration in Structured P2P Networks. In Proceedings of *WAIM*, 2008.

[22] A.Duminuco, E.Biersack. Hierarchical Codes: How to Make Erasure Codes Attractive for Peer-to-Peer Storage Systems. In Proceedings of P2P,2008.

[23] S.Ramabhadran, J.Pasquale.Durability of Replicated Distributed Storage Systems. In Proceedings of SIGMETRICS, 2008.