Availability in P2P based Online Social Networks

Nashid Shahriar*, Shihabur Rahman Chowdhury*, Reaz Ahmed*,
Mahfuza Sharmin†, Raouf Boutaba* and Bertrand Mathieu‡

*David R. Cheriton School of Computer Science, University of Waterloo
{nshahria | sr2chowdhury | r5ahmed | rboutaba}@uwaterloo.ca

† Dept. of Computer Science, University of Maryland, College Park
{msharmin@umiacs.umd.edu

‡Orange Labs, Lannion, France
bertrand2.mathieu@orange.com

***Abstract*—Despite their tremendous success, centrally controlled cloud based solutions for social media networking have inherent issues related to privacy and user control. Alternatively, a decentralized approach can be used, but ensuring content availability will be the major challenge. In this work, we propose a time-based user grouping and replication protocol that ensures content availability for decentralized sharing of online social media. The protocol exploits cyclic diurnal patterns in user uptime behaviors to ensure content persistence with minimal replication overhead. We also introduce the concept of β-availability that represents the probability that at least β members of a replication group will be online at any given time. We present a mathematical model for measuring β-availability as a function of peer-uptime duration and replication group size. Simulation results show that our protocol achieves high content persistence without incurring significant network and storage overheads.**

I. Introduction

Online Social Networks (OSNs) attract majority of the Internet users [1], [2]. According to *Nielsen's Social Media Report 2011* (http://cn.nielsen.com), around 80% of the active Internet users visit one of the OSN sites. Popular OSNs (like Facebook and Google+) provide free online storage for users to upload and share their social content. Facebook is the largest online social network with one billion active users. Facebook maintains more than 100 petabytes of online storage and stores more than 100 billion images.

Despite their tremendous success and apparently free services, OSNs are imbalancing the Internet's ecosystem in many ways. First, an OSN provider stores its users' social contents in a cloud based storage under the control of a central authority. This poses serious threats to user privacy and content ownership. For example, many OSN sites use its users' data to feed the advertisement industry. Second, users have to obey the restrictions imposed by the OSN sites (*e.g.*, resolution and format of uploaded content, storage limits *etc*.). Third, users cannot use their uploaded content across multiple OSN sites. Finally, OSN providers rely on third party content distribution networks (CDNs) for load distribution and low latency access across the globe. This aggravates the privacy concerns, since users' contents are now being cached at third party locations.

978-1-5386-3288-8/17/$31.00

These drawbacks of the current OSN sites have motivated the research community to investigate the possibility of a peer-to-peer (P2P) architecture for a decentralized OSN [3], [4], [5], [6], [7]. However, as addressed by these research efforts, a decentralized OSN solution inherits a very challenging problem from P2P storage systems: *how to ensure 24×7 content availability with minimal replication overhead*. Most of the proposals in P2P networks continuously maintain a fixed number of replica to ensure a content's availability. A few P2P approaches ([8], [9], [10]) use time-based replication strategy, where a peer's daily uptime behavior is utilized to place and reuse content replicas across its online sessions.

In this work, we propose *S-DATA* (Structured approach for Diurnal Availability by Temporal Assemblage), a time based user grouping and content replication protocol that exploits the cyclic diurnal pattern in user uptime behavior as observed in [11], [12], [13]. We store and replicate content within small user groups and assign the task of indexing group information and content meta information at more stable computing nodes. Users with complementary online patterns collaborate in small groups for replicating each others' contents across their online sessions. In order to ensure high availability and low latency access a geographically distributed setting, we introduce the concept of β-availability, *i.e.*, the probability that at least β members of a group will be online at any given time. This group formation problem is challenging due to three constraints. First, group size should be as small as possible. Second, β-availability should be maximized. Third, the group formation process should be globally optimized and should not incur significant network overhead.

In contrast to the existing time-based P2P replication approaches, S-DATA has two advantages. First, it uses a structured Distributed Hash Table (DHT) protocol, namely Plexus protocol [14], to construct globally optimized availability groups. S-DATA leverages the Hamming distance-based approximate matching and and index-replication offered by Plexus protocol to form groups and improve reliability, respectively. These techniques ensure higher system availability without generating high network or storage overhead for the group formation process. Second, S-DATA strive to ensure β-

availability, as opposed to 1-availability provided by the other approaches in the literature [8], [9], [15], [16], [10].

This paper extends our initial work [17] in several aspects. First, we derive a mathematical model for measuring β-availability as a function of peer-uptime duration and replication group size in Section V. Second, we extend Section VI to analyze system availability, convergence time, and communication overhead by varying expected uptime of peers. Finally, we update related work section to include an in-depth discussion on the subtle differences between our work and the state-of-the-art literature.

We organize the rest of this paper as follows. We start with a comparative study of S-DATA against the related works on distributed social networks and P2P availability in Section II. Then, we provide a conceptual overview of S-DATA in Section III followed by the protocol details in Section IV. We also present a mathematical model for measuring β-availability as a function of expected user uptime and group size in Section V. In Section VI, we provide a simulation study of the availability architecture to show its performance and to validate the mathematical model. Finally we conclude with some future directions of our work in Section VII.

II. Related Works

Decentralized Online Social Networks

A number of recent research efforts strive to decentralize storage and control in OSNs [18]. These efforts have been motivated by the limitations posed by the centralized architecture of the current OSNs [19]. Recently, several decentralized OSN architectures have been proposed [1], [2]. However, ensuring that the shared content is highly available is one of the main challenges that need to be addressed before these systems can be deployed. Authors in [20] show that in a P2P network the problem of finding a minimal set of peers to form replication groups achieving maximum availability is an NP-hard problem. Consequently, heuristic based approaches are required to ensure content availability in these systems.

PeerSon [4], SafeBook [5], PrPl [6], Lilliput [3], Cachet [21], Decent [22] are the prominent proposals of decentralized OSNs. These works mostly focus on the system design, communication protocols, consistency, encryption schemes, and dissemination of profiles in decentralized OSNs. However, they have not particularly focused on the replication schemes fundamental for increasing content availability. Certainly *S-DATA* can aid in addressing the availability requirement in these works. SuperNova [7] proposed a super peer based decentralized OSN which focuses on ensuring content availability. In this architecture a user can select a number of other users it trusts (storekeepers) to replicate a part of its contents and serve it in its absence. In contrast to *S-DATA*, this replication scheme does not take into account the uptime distribution of the other users when choosing them as replicas.

Time Based Replication in P2P Network

Several approaches to improve availability in P2P systems can be found in the literature. However, only a few of these approaches focus on increasing content availability using a time-based replication strategy. Blond et al. [9] proposed two availability-aware applications that take into account peers' previous availability history collected through an epidemic protocol. Using a simple predictor, they propose to find the best matching peer to meet the specific goals of the application. A group based Chord model is proposed in [15] to minimize the impact of frequent peer arrivals and departures. The redundancy group based approach proposed in [16] tries to improve availability by utilizing the cyclic behavior of geographically distributed peers. They proposed a hill-climbing strategy to determine redundancy groups for data objects and a counter-based availability score update mechanism. The latter periodically scans the network to find online peers and increases their score while reducing the offline peers' score. However, the counter mechanism cannot consistently capture phase relationships within a peer and between peers.

In a previous work, we proposed a peer grouping protocol that constructs replication groups of peers through a gossip based routing technique [8]. This approach may need a long convergence time to find a suitable group. Rzadca et al.[10] represented peer availability as a function of discrete time to minimize the number of replicas. In their model, availability is represented by a set of time slots in which a peer is available with certainty, *i.e.*, using discrete on-off availability. In contrast, we represent availability by historical probabilities at discrete time slots. Our probabilistic model captures diurnal availability patterns more accurately, since peer connectivity cannot be predicted with absolute certainty in a real world network. Moreover, the group formation approach proposed in [10] uses a single-valued scoring function, which only considers the number of newly covered slots while making group formation decision. On the contrary, our utility function considers relative improvement from both users and the size of the resulting group. Finally, their model only targets to ensure 1-availability across time slots, whereas we formulate the concept of β-Availability to provide better reliability.

III. Background

A. Architecture

As depicted in Fig. 1, S-DATA architecture revolves around three conceptual components: *replication group*, *Group Index Overlay* (GIO) and *Content Index Overlay* (CIO). Replication groups provide a persistent storage by exploiting users' diurnal uptime-behavior, GIO maintains availability information for individual users and user groups, while CIO retains an indirect mapping from content name to content location. We do not discuss the details of these components since they can be found in our initial work [17].

B. Availability Vector

The traditional definition of availability is simply measured by the fraction of time a user is online [23] within a certain time period. If a user joins and leaves m times during a period of T hours, and every time remains up for t_k hours, then its availability can be computed as, $\frac{\sum_{k=1}^{m} t_k}{T}$. This formula does

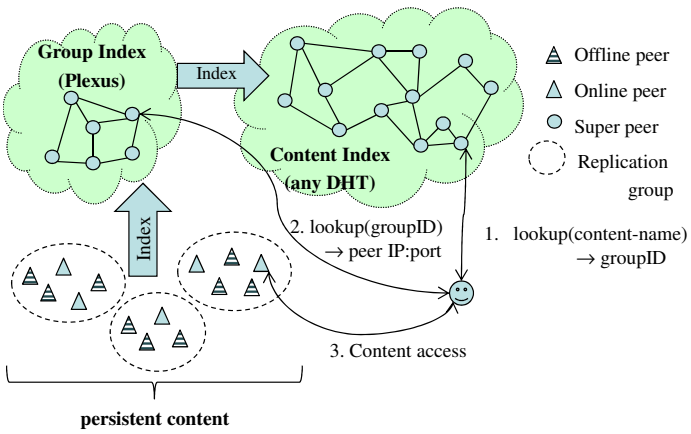


Fig. 1. Conceptual Architecture of S-DATA

not take into account the diurnal availability pattern in user uptime behavior. This fact has been mathematically proven by Yang et al. in [24].

In this work, we divide 24-hours of a day in K equal-length time-slots *w.r.t.* GMT+0, and estimate the probability of a user being online in each time-slot based on its historical behavior. Thus the availability of a user, say x, is defined as $\mathcal{A}_x = \{a_{x1}, a_{x2}, ..., a_{xk}, ..., a_{xK}\}$, where $\mathcal{A}_x$ is the K-dimensional availability vector for user x, and a_{xk} is the probability of user x being online in slot k.

The responsibility of computing and maintaining the availability vectors can be handled by the P2P client software or GIO. Each of these alternatives has its own merits and demerits, and can be left as an implementation specific choice. Computing and maintaining availability vectors at the client software will give more accurate estimates and will generate minimal network traffic. However, a client software can be maliciously modified to report a fake availability vector. Alternatively, the availability vectors can be computed and maintained by the GIO. This approach can generate more reliable availability vectors, though at the expense of increased network traffic and decreased accuracy.

C. Terminology

In S-DATA we use four indexes (see Table I) for group formation and content lookup. $\mathcal{I}_e$ represents an indexing node in CIO which is responsible for storing the ID of e (ID_e), where e can be a user or a group. $\mathcal{I}_e$ works as e's proxy for meta-information exchange. For user, say x, $\mathcal{I}_x$ stores an $\mathcal{M}_x$ record, which contains the availability vector (A_x), ID (ID_x) and network location (Loc_x) for x, as well as the group ID (ID_{G_x}) and index location ($\mathcal{I}_{G_x}$) for x's group G_x. For a group G, $\mathcal{I}_G$ contains index record $\mathcal{N}_G$, which contains group availability vector ($\mathcal{A}_G$), group ID (ID_G), and for each member x of G, its ID (ID_x), index location ($\mathcal{I}_x$) and network location (Loc_x). To enable approximate matching between users' and groups' availability vectors, we maintain $\mathcal{V}_e$ indexes that contain availability pattern (S_e, explained in Section IV-A1), availability vector ($\mathcal{A}_e$), ID (ID_e) and index location ($\mathcal{I}_e$) for e. $\mathcal{V}_e$ is stored in all nodes $\mathcal{L}_e$ within a pre-specified Hamming distance from S_e. Finally, for content lookup another set of indexes ($\mathcal{K}_w$) is maintained in CIO. For each keyword w attached to a content, an index ($\mathcal{K}_w$) is stored in CIO at node $\mathcal{J}_w$, which is responsible for keyword w. $\mathcal{K}_w$ retains the content's ID (ID_{doc}), other keywords describing the content ($\{w_i\}$), group ID (ID_G) and index location ($\mathcal{I}_G$) of the group that hosts the content.

TABLE I
LIST OF INDEXES IN S-DATA

Name	Overlay	Indexed information
$\mathcal{M}_x$	GIO/$\mathcal{I}_x$	$< \mathcal{A}_x, ID_x, Loc_x, ID_{G_x}, \mathcal{I}_{G_x} >$
$\mathcal{N}_G$	GIO/$\mathcal{I}_G$	$< \mathcal{A}_G, ID_G, \{< ID_x, \mathcal{I}_x, Loc_x > \mid x \in G\} >$
$\mathcal{V}_e$	GIO/$\mathcal{L}_{S_e}$	$< S_e, \mathcal{A}_e, ID_e, \mathcal{I}_e >$
$\mathcal{K}_w$	CIO/$\mathcal{J}_w$	$< ID_G, \mathcal{I}_G, ID_{doc}, \{w_i \mid w_i \in doc\} > \mathcal{L}_{\acute{S}_x}$

IV. S-DATA PROTOCOL

We use four processes to describe the proposed S-DATA protocol. In Section IV-A, we present the mechanism for a peer to advertise its (or its group's) availability pattern to the GIO. The group formation process is discussed in detail in Section IV-B. The process of refreshing group indexes in GIO with the most up to date online status of the group members is presented in Section IV-C. Finally, the content lookup and content index update process is presented in Section IV-D.

A. Indexing Availability Information

To cluster users in globally optimized replication groups, we need to index each user's availability information ($\mathcal{V}_e$) to GIO. This indexing process involves two steps: i) encoding availability vector ($\mathcal{A}_e$) to bit-vector (S_e) and ii) advertisement using Plexus protocol. These two steps are explained next.

1) Availability Vector Encoding: It can be easily seen that the availability vector $\mathcal{A}_i$ is a K-dimensional vector of uptime probabilities, whereas the advertisement (or query) patterns in a Plexus network built on an $< n, k, d >$ code are n-bit values. Hence, we need a means to encode a K-dimensional availability vector into an n-bit pattern.

In this work we have used $K = 24$ slots for availability vector. While for Plexus implementation, we have used the $< 24, 12, 8 >$ Extended Golay Code G_{24}. Trivially, we can directly encode each probability value a_{ik} in $\mathcal{A}_i$ to one-bit in the 24-bit advertisement (or query) pattern. We can use a threshold, say θ, and can set the k-th bit of the 24-bit encoded pattern to 1 if $a_{ik} > \theta$. Unfortunately, this encoding will incur significant information loss and will degrade approximate matching performance in Plexus network.

Instead, we use a better encoding scheme based on the observation that consecutive values in the availability vector are usually similar in magnitude. To exploit this observation, we average the probability values in two adjacent slots and obtain a 12-dimensional availability vector $\acute{\mathcal{A}}_i = \{\acute{a}_{i1}, \acute{a}_{i2}, \ldots \acute{a}_{i12}\}$, where $\acute{a}_{ij}$ is computed as $\acute{a}_{ij} = \frac{(a_{i(2j-1)} + a_{i(2j)})}{2}$. Now, we encode each $\acute{a}_{ij}$ into two bits in the 24-bit advertisement pattern as follows. $\acute{a}_{ij}$ is encoded to 00 if $\acute{a}_{ij}$ is less than $\frac{1}{3}$.

If $\acute{a}_{ij}$ is between $\frac{1}{3}$ and $\frac{2}{3}$ then the encoding is 01. Otherwise, $\acute{a}_{ij}$ is greater than $\frac{2}{3}$ and is encoded to 11. This encoding reflects the numeric distance in $\acute{a}_{ij}$ to the Hamming distance in advertisement patterns.

2) Advertisement: An advertising user, say x, first computes the n-bit advertisement pattern, say S_x, as explained above. Then x sends the tuple $< \mathcal{S}_x, \mathcal{A}_x, ID_x, Loc_x, ID_{G_x}, \mathcal{I}_{G_x} >$, to $\mathcal{I}_x$. If x has not formed a group then ID_{G_x} and $\mathcal{I}G_x$ will be empty. Upon receiving the advertisement message $\mathcal{I}_x$ computes the codewords within a pre-specified Hamming distance from S_x and uses Plexus routing to route and index the advertisement ($\mathcal{V}_x$) to the nodes ($\mathcal{L}_{S_x}$) responsible for these codewords.

B. Group Formation

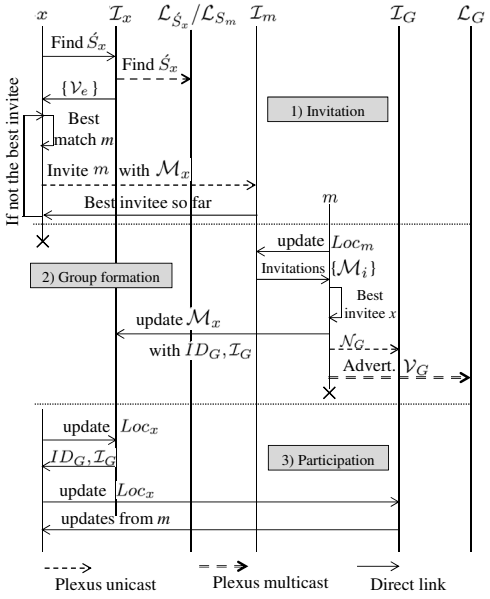


Fig. 2. Sequence diagram shows group formation of x with m

This process lies at the core of S-DATA protocol. Our target is to cluster users into groups in such a way that the group sizes are minimal and at any given time at least $\beta \geq 1$ users from a group are online with the highest possible probability.

The most challenging part of this process is to relay group formation messages between users that may not be simultaneously online. To this end, we use GIO as a message relay. Fig. 2 presents a sequence of message exchanges between indexing nodes in GIO and users x and m while forming a 1-availability group G. It is worth noting that x and m are not online simultaneously and hence they have no direct message exchange. The Group formation process is composed of the following three steps:

1) *Invitation* : We assume that on average a user will be online for L time-slots on a daily basis. It will be the responsibility of a user to maintain β users in its group during the L-slots it is online and the next L-slots. To find a suitable user that can improve group's availability for the next L-slots, user x computes an availability pattern $\acute{S}_x$. $\acute{S}_x$ has bits $t + L + 1$ to $t + 2L$ set to 1, assuming that the availability pattern S_x of user x has bits t to $t + L$ set to 1. Once $\acute{S}_x$ is computed, user x forwards it to $\mathcal{I}_x$. $\mathcal{I}_x$ uses Plexus multi-cast routing to find the users ($\mathcal{L}_{\acute{S}_x}$) in GIO responsible for indexing user/group availability records ($\mathcal{V}_e$) similar to $\acute{S}_x$. From the availability records ($\mathcal{V}_e$) returned by $\mathcal{I}_x$, user x selects the most appropriate user, say m, that has minimum Hamming distance from $\acute{S}_x$ and maximizes its group availability. A mathematical model for selecting the most suitable user will be presented in Section V-C. User x locates the indexing node ($\mathcal{I}_m$) for m using Plexus routing and sends an invitation request to $\mathcal{I}_m$ that includes the $\mathcal{V}_x$ record.
2) *Group formation* : Upon becoming online m updates $\mathcal{I}_m$ with its new network location (Loc_m). In response $\mathcal{I}_m$ sends all the invitations ($\{\mathcal{V}_e\}$) for m that have been accumulated during m's offline period. Among these invitations, m selects the best candidate x. If x is already a member of an existing group then m simply joins the group otherwise it creates a new group G. To create or update the group index in GIO, m may require to transmit three messages: a) if m created a new group, it has to update the $\mathcal{M}_x$ record in $\mathcal{I}_x$ so that x can learn about G upon returning; b) m has to index ($\mathcal{V}_G$) to all nodes ($\mathcal{L}_G$) within a certain Hamming distance from S_G; c) m has to store the group index $\mathcal{N}_G$ to $\mathcal{I}_G$.
3) *Participation* : During its next online session user x will update I_x with its new network location Loc_x. If the previous invitation from x was honored by m then I_x responds with the newly formed group's information (ID_G and $\mathcal{I}_G$). x updates $\mathcal{I}_G$ with its location information Loc_x and $\mathcal{I}_G$ responds with any update from m or other members of G. On the other hand, if the invitation from x was not accepted by m, then x has to restart the group formation process with the next best matching user, other than m.

The above described process of forming 1-availability group can be easily extended to construct β-availability groups. Two modifications in Step 1 of the above process are required. First, x should be the highest ID user among the online members of its group (G_x). Second, x should send invitations to $\beta - f$ users simultaneously, where f is the number of users in x's group who will be online in the L-time slots following the online period of x.

C. Group Maintenance

The diurnal availability pattern of a user may change over time. In such a situation a user, say x, may want to change its group. Group changing involves leaving the current group and joining a new group. The process of joining a group has been described in Section IV-B. To leave its current group G_x, user x has to update two nodes in GIO. First, x has to remove its index information from $\mathcal{N}_{G_x}$ record, which is stored at node $\mathcal{I}_{G_x}$. Second, x has to clear the ID_{G_x} and $\mathcal{I}_{G_x}$ fields in $\mathcal{M}_x$ record, which is stored in $\mathcal{I}_x$. It should be noted that we use soft-state registration for advertising $\mathcal{V}_x$ records to $\mathcal{L}_{S_x}$.

Hence, the $\mathcal{V}_x$ records will be automatically removed from the nodes in $\mathcal{L}_{S_x}$, if x does not re-advertise before the previous advertisement expires.

D. Content Indexing and Lookup

1) Content Indexing: Traditionally a content in a P2P network is tagged with a set of descriptive keywords, ($w \in \{w_i\}$). These keywords are used to locate the node ($\mathcal{J}_w$) in CIO for storing the $\mathcal{K}_w$ record. While advertising a content a user, say x, may or may not be a member of a replication group. If x is a member of a replication group, say G_x then ID_{G_x} and $\mathcal{I}_{G_x}$ are stored in $\mathcal{K}_w$ record, otherwise ID_x and $\mathcal{I}_x$ are used. However, $\mathcal{K}_w$ is not updated when x forms a group. Rather, $\mathcal{K}_w$ is updated in a reactive manner during content lookup. This process is described in the following.

2) Content Lookup: A query for keyword w will be routed to $\mathcal{J}_w$ using the routing protocol in CIO. Based on the information found in $\mathcal{K}_w$, the query will be forwarded to either $\mathcal{I}_{G_x}$ if the content host x has formed a group and $\mathcal{K}_w$ has been updated, or the query will be forwarded to $\mathcal{I}_x$. In a regular scenario, the query will be forwarded to $\mathcal{I}_{G_x}$ and the location Loc_y of the currently alive user y in G_x will be returned to the querying user via $\mathcal{J}_w$. In contrast, if x has formed a group but $\mathcal{K}_w$ has not been updated, then $\mathcal{J}_w$ will contact $\mathcal{I}_x$, which will respond with ID_{G_x} and $\mathcal{I}_{G_x}$. Accordingly, $\mathcal{J}_w$ will update $\mathcal{K}_w$ for future references. Finally, $\mathcal{J}_w$ will contact $\mathcal{I}_{G_x}$ to obtain the location (Loc_y) of the currently active user (y) in G_x.

V. Mathematical Model

In this section we first present a mathematical model for calculating β-availability. Then we present a model for expressing β-availability as a function of the average uptime and group size. Finally, we present a utility function for computing the relative gain in β-availability that can be achieved by combining two peers (or groups) in a group.

A. Defining β-Availability

As presented in Section III-B, we express the availability of peers x as K dimensional vector $\mathcal{A}_x = \{a_{x1}, a_{x2}, \ldots a_{xK}\}$, where a_{xk} represents the probability of peer x being online in time-slot k. When peers collaborate in a replication group, say G, we can model the 1-availability vector of the group as $\mathcal{A}^1_G = \{a^1_{G1}, a^1_{G2}, \ldots, a^1_{GK}\}$. The combined probability of at least one peer being online at any given slot k can be computed as follows:

$$\begin{aligned} a^1_{Gk} &= Pr[at\ least\ 1\ member\ is\ online\ in\ slot\ k] \\ &= 1 - Pr[no\ peer\ is\ online\ in\ slot\ k] \\ &= 1 - \prod_{\forall x \in G} (1 - a_{xk}) \end{aligned} \tag{1}$$

We can extend the equation for a^1_{Gk} to compute 2-availability, *i.e.*, the probability of at least 2 peers being online, at slot k as follows:

$$\begin{aligned} a^2_{Gk} &= Pr[at\ least\ 1\ member\ is\ online\ at\ slot\ k] \\ &\quad - Pr[exactly\ 1\ member\ is\ online\ at\ slot\ k] \\ &= a^1_{Gk} - a^{exactly\ 1}_{Gk} \end{aligned}$$

Here $a^{exactly\ 1}_{Gk}$ is the probability of exactly one member of G being online at slot k. This can be computed as:

$$a^{exactly\ 1}_{Gk} = \sum_{\forall x \in G} a_{xk} \prod_{\forall y \in G, y \neq x} (1 - a_{yk}) \tag{2}$$

In a similar manner, we can generalize the 2-availability equation to compute β-availability at any slot k as follows:

$$\begin{aligned} a^{\beta}_{Gk} &= a^{(\beta-1)}_{Gk} - a^{exactly\ (\beta-1)}_{Gk} \\ &= a^1_{Gk} - \sum_{j=1}^{\beta-1} a^{exactly\ (j)}_{Gk} \end{aligned} \tag{3}$$

Now we can average the slot-wise availability values to obtain an estimated β-availability of the group over time:

$$|\mathcal{A}^{\beta}_G| = \frac{1}{K} \sum_{k=1}^{K} a^{\beta}_{Gk} \tag{4}$$

B. Estimating β-availability

Here we develop a mathematical model to establish the relationship between β-availability, group size and uptime distribution. Without loss of generality we assume that each peer is online for L consecutive slots of a day with high probability, while its probability of being online for the rest of the slots is very low. Following the finding of Bustamante et al. in [25], we model the duration of a peer's online session, *i.e.*, L, using the Pareto distribution with shape parameter α. According to this distribution the expected uptime can be computed as $L = \frac{\alpha}{1-\alpha}$.

Besides modeling uptime distribution, we have to model the positive and negative correlation between the probabilities of a peer being online between consecutive time slots. For this we have used the short tailed Cauchy distribution that can represent the correlation between consecutive probability values. Accordingly, we partition the availability vector into high and low regions based on uptime L. For peer x, a_{xk} in any slot k in the high availability region can be computed as:

$$a_{xh[i]} = s * \frac{\gamma}{\pi[(i * \delta_h)^2 + \gamma^2]} \tag{5}$$

where, $\delta_h = \frac{h}{L}$ and $i = 0$ to $L - 1$.

Similarly, we can compute a_{xk} in any slot k in the low availability region as follows:

$$a_{xl[i]} = s * \frac{\gamma}{\pi[(i * \delta_l + p)^2 + \gamma^2]} \tag{6}$$

where $\delta_l = \frac{l}{K-L}$ and $i = 1$ to $K - L$. Here, h, l, s and p are constants, which can be manipulated to model different uptime behaviors.

TABLE II
1-AVAILABILITY AT DIFFERENT SLOTS FOR PEERS OF SAME GROUP

Peak at	slot 1	slot L	slot $2L$	slot $3L$	slot kL
1^{st} L slots	$f(-\frac{L}{2}.\delta_h)$	$f(\frac{L}{2}.\delta_h)$			
2^{nd} L slots	$f(-\frac{L}{2}.\delta_h - L\delta_l)$	$f(-\frac{L}{2}.\delta_h)$	$f(\frac{L}{2}.\delta_h)$		
3^{rd} L slots	$f(-\frac{L}{2}.\delta_h - 2L\delta_l)$	$f(-\frac{L}{2}.\delta_h - L\delta_l)$	$f(-\frac{L}{2}.\delta_h)$	$f(\frac{L}{2}.\delta_h)$	
$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$
i^{th} L slots	$f(-\frac{L}{2}.\delta_h - (i-1)L\delta_l)$	$\cdots$	$\cdots$	$\cdots$	$f(-\frac{L}{2}.\delta_h - kL\delta_l)$

Suppose group G is composed of peers, $x, y \ldots z$. We pick a peer, say x, having peak slot in the middle of the first L slots. Then we compute its availability vector, $\mathcal{A}_x = \{a_{x1}, a_{x2} \ldots a_{xK}\}$ using Equation (5) and Equation (6). We get K equations similar to Equation (5) and Equation (6) corresponding to each of the K slots. We pick another peer y, whose peak slot is at the middle of the second L slots and get another K equations corresponding to the vector, $\mathcal{A}_y = \{a_{y1}, a_{y2} \ldots a_{yK}\}$ in a similar way. In this way, we pick a peer z having the peak slot at the middle of the $\frac{K}{L}$th L slots and obtain $\mathcal{A}_z = \{a_{z1}, a_{z2} \ldots a_{zK}\}$. Using these availability vectors we can find β-availability at each slot. It should be noted that the β-availability of a group depends on L, while L depends on α. According to the recurrence relation in Equation (3), finding β-availability requires computation of 1-availability, 2-availability and so on. Therefore, as the first step of β-availability computation, we present a method to calculate 1-availability = $1 - (1 - a_{xi})(1 - a_{yi}) \ldots (1 - a_{zi})$ at any slot i in the following.

We can generalize Equation (5) and Equation (6) as

$$f(w) = s * \frac{\gamma}{\pi[(w)^2 + \gamma^2]}$$

where, w represents either a high or a low availability slot. As depicted in Table II, for an ideal scenario, a group will have $\frac{K}{L}$ peers and the peak of the 1^{st} peer will align with the 1^{st} L slots, peak of the 2^{nd} peer will align with the 2^{nd} L slots and so on. Table III, on the other hand, presents 1-availability of peers from the same group at k^{th} slot. From these two tables we can find a^1_{Gk} for $k < L$ as follows:

$$\begin{aligned} a^1_{Gk} =& 1 - (1 - f(-\frac{L}{2}.\delta_h + k\delta_h)) \\ & * \prod_{i=2}^{\frac{K}{L}} (1 - f(-\frac{L}{2}.\delta_h - ((i-1)L - k)\delta_l)) \end{aligned} \quad (7)$$

To obtain a_{Gk} for all $k = 1$ to K, a more generalized formulation is required. Moreover, the best case scenario, as presented in Tables II and III, will not be found in a real situation. Considering these two factors we average slot availability values for high and low regions separately. We use a_h and a_l to denote the average availability at high and low regions, respectively. Within a group these two average values will the be same for all peers. Hence we have not added the peer name in a_h or a_l. Evidently, we can obtain a_h and a_l by integrating the slot-wise availability values presented in Equation (5) and Equation (6), respectively as follows:

$$a_h = \frac{2}{L} \int_0^{L/2} a_{h[i]} di = \frac{2s}{h\pi} \arctan \frac{h}{2\gamma} \quad (8)$$

$$a_l = \frac{2}{K-L} \int_0^{\frac{K-L}{2}} a_{l[i]} di = \frac{2s}{l\pi} \Big(\arctan \frac{l+2p}{2\gamma} - \arctan \frac{p}{\gamma} \Big) \quad (9)$$

Now, using a_h and a_l in Equation Equation (1), we get a simplified form of 1-availability: $a^1_{Gk} = 1 - (1 - a_h)(1 - a_l)^{|G|-1}$. Here $|G|$ is the number of members in group G. Now combining Equations (2), (5), (6), (8), and (9) we can compute $a^{exactly\ 1}_{Gk}$ as,

$$\begin{aligned} a^{exactly\ 1}_{Gk} &= a_h(1 - a_l)^{|G|-1} \\ &\quad + (|G| - 1)a_l(1 - a_h)(1 - a_l)^{|G|-2} \end{aligned} \quad (10)$$

We can extend Equation 10 to get $a^{exactly\ 2}_{Gk}$ as

$$\begin{aligned} a^{exactly\ 2}_{Gk} &= \sum_{\forall P_x, P_y \in G, x \neq y} a_{xk} a_{yk} \prod_{\forall P_z \in G, x \neq y \neq z} (1 - a_{zk}) \\ &= \sum_{m=0}^{1} a_h^{1-m} a_l^m (1 - a_h)^m (1 - a_l)^{|G|-2-m} \\ &+ (|G| - 2) a_l^2 (1 - a_h)^2 (1 - a_l)^{|G|-4} \end{aligned}$$

In a similar manner, the probability of exactly j members of G to be available at k^{th} slot can be computed as:

$$\begin{aligned} a^{exactly\ j}_{gk} &= \sum_{m=0}^{j-1} a_h^{j-m} a_l^m (1 - a_h)^m (1 - a_l)^{|G|-j-m} \\ &+ (|G| - j) a_l^j (1 - a_h)^j (1 - a_l)^{|G|-2j} \end{aligned} \quad (11)$$

Now, combining Equation (3) and (11), we can obtain the simplified form of β-availability as follows:

$$\begin{aligned} a^{\beta}_{gk} &= 1 - (1 - a_h)(1 - a_l)^{|G|-1} \\ &\quad - \sum_{j=1}^{\beta-1} \sum_{m=0}^{j-1} a_h^{j-m} a_l^m (1 - a_h)^m (1 - a_l)^{|G|-j-m} \\ &\quad - \sum_{j=1}^{\beta-1} (|G| - j) a_l^j (1 - a_h)^j (1 - a_l)^{|G|-2j} \end{aligned} \quad (12)$$

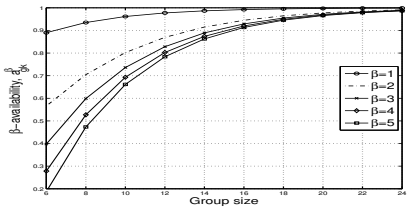


Fig. 3. Estimation of β-availability

Equation (12) establishes a relationship between β-availability, uptime and group size. To solve it analytically,

we first estimate the average availabilities, *i.e.*, a_h and a_l, using the constant values: $\gamma = 1$, $s = 2$, $h = 1$, $\alpha = 1.5$, $p = 0.5$ and $l = 4$. These values give the best fit for the real-trace data that can be found in [26] and [27]. Putting the values of a_h and a_l in Equation 12, we can estimate slot-wise availability for different values of β and group-sizes. The resulting β-availability curves are shown in Fig. 3, which confirms that larger groups are required to attain a given level of β-availability for higher values of β.

TABLE III
1-AVAILABILITY AT kTH SLOT FOR PEERS OF SAME GROUP

Peer at	k^{th} slot
1^{st} L slots	$f(-\frac{L}{2}.\delta_h + k\delta_h)$
2^{nd} L slots	$f(-\frac{L}{2}.\delta_h - (L-k)\delta_l)$
3^{rd} L slots	$f(-\frac{L}{2}.\delta_h - (2L-k)\delta_l)$
$\cdots$	$\cdots$
i^{th} L slots	$f(-\frac{L}{2}.\delta_h - ((i-1)L-k)\delta_l)$

C. Peer Selection Metric

During the group formation process, as presented in Section IV-B, a peer needs to select the most appropriate candidate for group formation from a set of peers or groups with the desired availability pattern. We define $C_{x,y}$ to be the combined gain that can be achieved by placing peer (or group) x and peer (or group) y in a new group, say G. $C_{x,y}$ can be obtained by adding the individual gains $U_{x,G}$ and $U_{y,G}$ for the participants x and y, respectively. Equations for computing combined gain $C_{x,y}$ and individual gain $U_{m,G}$ are given below.

$$C_{x,y} = \frac{(U_{xG} + U_{yG})}{|G_x \cup G_y|} \tag{13}$$

where,

$$U_{m,G} = \sum_{k=1}^{K} (a^{\beta}_{Gk} - a^{\beta}_{mk})$$

To reduce replication overhead, we want groups to be as small as possible. We incorporate this constraint in $C_{x,y}$ computation, by placing the new group size, *i.e.*, $|G| = |G_x \cup G_y|$, as dominator in Equation (13).

VI. Performance Evaluation

We used the Peersim [28] simulator for implementing *S-DATA* protocol on a Plexus network deployed using the Extended Golay Code G_{24} as described in Section III-A. Our simulation is focused on the following aspects: first, we measure the efficiency of our grouping protocol and compare it with other grouping approaches, i.e., random, unstructured, and centralized grouping approaches (Section VI-A). Second, we show the availabilities achieved by *S-DATA* along with the associated network and storage overheads (Section VI-B). We use Pareto distribution for generating the availability vectors based on the observations in [25], [29]. We design the simulations around GIO and replication groups, and deliberately omit to simulate CIO, since it is out of the scope of this work. We use the following performance metrics to evaluate *S-DATA*:

- *Group availability* is measured in units of nine [30] and defined as $-log_{10}(1-T)$, where T is the fraction of the total observed time when groups are available. For instance, a group availability of 2 nines implies that the group is accessible during 99% of the total time.
- *System availability* is computed as the average of the availability ($|\mathcal{A}^{\beta}_{G}|$) across all groups.
- *Convergence time* is measured by the number of slots required for 99% of the peers to join some group.
- *Normalized message overhead* is measured as the ratio of total number of messages exchanged and the number of successful replies sent by users in the system required to form a group.

A. Grouping Efficiency

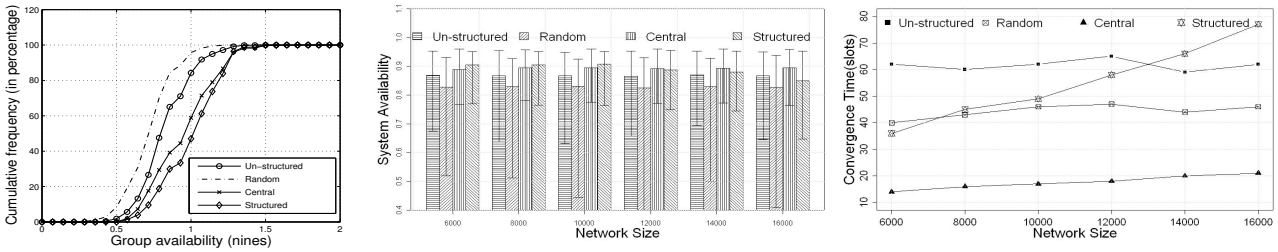
We perform the simulations in this section for an expected uptime distribution $L = 6$, and vary the network size from 6000 to 16000 in steps of 2000. We compare *S-DATA* with the following approaches:

- *Unstructured*: We use the gossip protocol as proposed in [8] in this strategy, where users reply based on their local knowledge for group formation.
- Random: In this strategy, a user randomly invites a peer within two hop neighbourhood without using any selection metric. The invited user then decides to accept or deny the invitation according to a random toss.
- Central: In this scheme, a central entity (Oracle), which can be a single cloud service provider, stores the availability vectors for all users in the system. Alive users communicate with the Oracle to select and invite the best matching user according to Equation (13). The Oracle chooses the best invitee, from the invitations for each user and forms a group.

Fig. 4(a) depicts the cumulative frequency of the percentage of groups as a function of group availability. For the non-random grouping strategies, all the groups remain available for more than 70% (0.5 nines) of the time, whereas all of the groups formed by random grouping strategy remain available for only 55% (0.35 nines) of the time. From Fig. 4(a), we can also see that our structured grouping approach outperforms all other approaches.

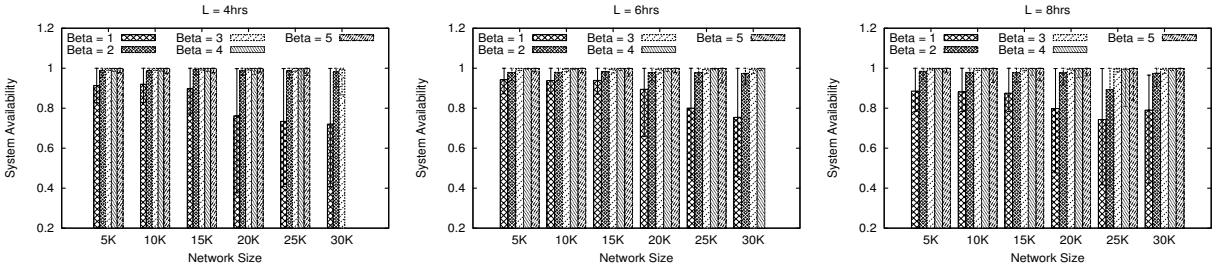
Expected values of system availability after convergence are presented in Fig. 4(b). Besides the average values of system availability, we also present the minimum and maximum group availabilities as error bars. The structured approach exhibits superior system availability than that of random and unstructured approaches. Shorter error bars on *S-DATA* also indicate that the maximum and minimum group sizes are close to the average group size. The random approach has the worst result of the four with larger deviation in the error bar and lowest system availability. The performance of the unstructured method lies in between central and random strategies.

Fig. 4(c) presents system convergence time as a function of network size. The central strategy has the lowest convergence time because the task of index storage and group formation are done in one location in the network. The convergence time is not affected by network size for random and unstructured approaches, whereas for central and *S-DATA*, convergence time



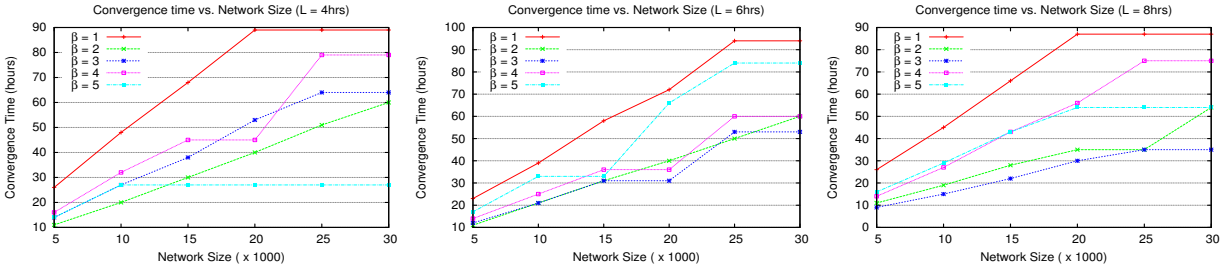
(a) Group Availability (b) System Availability (c) Convergence Time

Fig. 4. Grouping Efficiency



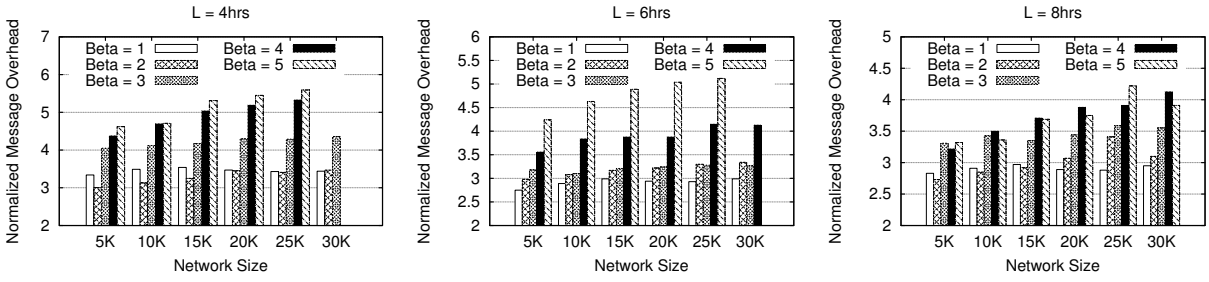
(a) L = 4hrs (b) L = 6hrs (c) L = 8hrs

Fig. 5. System availability



(a) L = 4hrs (b) L = 6hrs (c) L = 8hrs

Fig. 6. Convergence Time (Varying Network Size)



(a) L = 4hrs (b) L = 6hrs (c) L = 8hrs

Fig. 7. Message Overhead

increases with increased network size. For *S-DATA* users with low uptime remain ungrouped until the group formation for highly available users is complete, hence the longer convergence time.

B. *Performance of* S-DATA

We perform the simulations in this experiment for three uptime distributions, $L = 4$, 6, and 8 hours, respectively. For each value of L we vary the network size from 5000 to 30000 in steps of 5000, and vary β from 1 to 5.

System Availability: Fig. 5 presents the expected system availability for different replication levels (β), network size and uptime distribution (L). System availability increases with β as we have more redundancy. The diminishing error bars for higher β implies that more individual groups are achieving availability very close to the system average. It is worth mentioning that a significant improvement in system availability is achieved when β rises from 1 to 2. But for higher β, the increase is not significant.

Convergence Time: Fig. 6 shows convergence time for different network sizes. With an increase in network size, the convergence time also increases almost linearly. For most of the cases, a higher user uptime results in a faster convergence in group formation. It is worth noting that the convergence time is higher for $\beta = 1$ than that for other values of β. For $\beta = 1$, users with lower uptime have lesser option to form groups, and they have to wait for the highly available users to complete their group formation. This results in the higher convergence time. It can be observed that $\beta = 2$ provides a fast converging system.

Normalized Message Overhead (NMO): Fig. 7 shows the messaging overhead for group formation using *NMO*. It can be seen that less messaging overhead is incurred when users have relatively high uptime. As the network size grows, a user receives more group formation invitation from other users, but can accept only one. Therefore, it rejects all but one invitation. The rejected users then need to send more invitations to join a group. Moreover when users want to ensure β-availability for higher β, they need to have more users in the group and even more invitations are sent to other users. This obviously results in higher *NMO*.

VII. Conclusion

In this paper, we have introduced the β-availability and described an efficient grouping protocol (*S-DATA*), which ensures data availability around the clock in a P2P based OSN. We have derived a mathematical relationship between expected β-availability, group size, and uptime distribution of peers. This can assist in selecting the most suitable users, while forming a replication group. Simulation results showed that the proposed *S-DATA* protocol ensures very high availability of contents, comparable to a centralized group formation protocol. The results suggest that $\beta = 2$ can be a good operating point since it achieves high system availability without incurring significant overhead in terms of messaging and convergence time. Availability of popular contents can be further increased by caching the contents at other users not belonging to the replication group of the contents.

In the future, we intend to deploy *S-DATA* on a real world system and further investigate its performance for specific application availability requirements. The success of *S-DATA* also depends on the willingness and truthfulness of the peers. Tackling the potentially malicious behavior of peers and security issues of group formation is another prospective research issue we plan to investigate.

VIII. Acknowledgement

This work was supported by Natural Science and Engineering Council of Canada (NSERC) under its Discovery program.

References

[1] S. R. Chowdhury, A. R. Roy, M. Shaikh, and K. Daudjee, "A taxonomy of decentralized online social networks," *Peer-to-Peer Networking and Applications*, vol. 8, no. 3, pp. 367–383, 2015.

[2] T. Paul, A. Famulari, and T. Strufe, "A survey on decentralized online social networks," *Computer Networks*, vol. 75, pp. 437–452, 2014.

[3] T. Paul, N. Lochschmidt, H. Salah, A. Datta, and T. Strufe, "Lilliput: A storage service for lightweight peer-to-peer online social networks." IEEE (26th International Conference on Computer Communications and Networks (ICCCN)), 2017.

[4] S. Buchegger, D. Schiöberg, L.-H. Vu, and A. Datta, "Peerson: P2P social networking: early experiences and insights," in *Proceedings of the Second ACM EuroSys Workshop on Social Network Systems*, ser. SNS '09, 2009, pp. 46–52.

[5] L. Cutillo, R. Molva, and T. Strufe, "Safebook: A privacy-preserving online social network leveraging on real-life trust," *Communications Magazine, IEEE*, vol. 47, no. 12, pp. 94 –101, Dec. 2009.

[6] S.-W. Seong, J. Seo, M. Nasielski, D. Sengupta, S. Hangal, S. K. Teh, R. Chu, B. Dodson, and M. S. Lam, "Prpl: a decentralized social networking infrastructure," in *Proceedings of the 1st ACM Workshop on Mobile Cloud Computing & Services: Social Networks and Beyond*, ser. MCS '10, 2010, pp. 8:1–8:8.

[7] R. Sharma and A. Datta, "Supernova: Super-peers based architecture for decentralized online social networks," in *Proceedings of COMSNETS*, 2012, pp. 1–10.

[8] N. Shahriar, M. Sharmin, R. Ahmed, M. Rahman, R. Boutaba, and B. Mathieu, "Diurnal availability for peer-to-peer systems," in *Proc. CCNC*, Las Vegas, Nevada, USA, Jan 2012.

[9] S. Blond, F. Fessant, and E. Merrer, "Finding good partners in availability-aware p2p networks," in *Proc. SSS*, 2009.

[10] K. Rzadca, A. Datta, and S. Buchegger, "Replica placement in p2p storage: Complexity and game theoretic analyses," in *Proc. DCS*, June 2010, pp. 599–609.

[11] D. Stutzbach and R. Rejaie, "Understanding churn in peer-to-peer networks," in *Proc. IMC*, 2006, pp. 189–202.

[12] J. Chu, K. Labonte, and B. N. LevineH, "Availability and locality measurements of peer-to-peer file systems," in *Proc. ITCom*, 2002.

[13] S. Saroiu, P. K. Gummadi, and S. Gribble, "A measurement study of peer-to-peer file sharing systems," in *Proc. MMCN*, 2002.

[14] R. Ahmed and R. Boutaba, "Plexus: a scalable peer-to-peer protocol enabling efficient subset search," *IEEE/ACM Trans. on Networking (TON)*, vol. 17, no. 1, pp. 130–143, Feb 2009.

[15] Y. Dan, C. XinMeng, and C. YunLei, "An improved p2p model based on chord," in *Proc. 6th PDCAT*, 2005.

[16] T. Schwarz, Q. Xin, and E. Miller, "Availability in global peer-to-peer storage systems," in *Proc. IPTPS*, 2004.

[17] N. Shahriar, S. R. Chowdhury, M. Sharmin, R. Ahmed, R. Boutaba, and B. Mathieu, "Ensuring beta-availability in p2p social networks," in *2013 IEEE 33rd International Conference on Distributed Computing Systems Workshops*, July 2013, pp. 150–155.

[18] M. Qamar, M. Malik, S. Batool, S. Mehmood, A. W. Malik, and A. Rahman, "Centralized to decentralized social networks: Factors that matter," pp. 37–54, 2016.

[19] M. Marcon, B. Viswanath, M. Cha, and K. P. Gummadi, "Sharing social content from home: a measurement-driven feasibility study," in *Proceedings of NOSSDAV '11*, 2011, pp. 45–50.

[20] R. Sharma, A. Datta, M. Dell'Amico, and P. Michiardi, "An empirical study of availability in friend-to-friend storage systems," in *Proceedings of 2011 IEEE International Conference on Peer-to-Peer Computing*, 2011, pp. 348–351.

[21] S. Nilizadeh, S. Jahid, P. Mittal, N. Borisov, and A. Kapadia, "Cachet: a decentralized architecture for privacy preserving social networking with caching," in *Proceedings of the 8th CoNEXT*. ACM, 2012, pp. 337–348.

[22] S. Jahid, S. Nilizadeh, P. Mittal, N. Borisov, and A. Kapadia, "Decent: A decentralized architecture for enforcing privacy in online social networks," in *Pervasive Computing and Communications Workshops, 2012 IEEE International Conference on*, pp. 326–332.

[23] R. Bhagwan, K. Tati, Y. Cheng, S. Savage, and G. Voelker, "Total recall: system support for automated availability management," in *Proc. NSDI*, 2004.

[24] Z. Yang, J. Tian, and Y. Dai, "Towards a more accurate availability evaluation in peer-to-peer storage systems," *Intl. Journal of High Performance Computing and Networking*, vol. 6, no. 3/4, pp. 233–246, 2010.

[25] F. E. Bustamante and Y. Qiao, "Friendships that last: Peer lifespan and its role in p2p protocols," in *Proc. Web Content Caching and Distribution*, 2004, pp. 233–246.

[26] Repository of availability traces. [Online]. Available: http://www.cs.uiuc.edu/homes/pbg/availability/

[27] The peer-to-peer trace archive. [Online]. Available: http://p2pta.ewi.tudelft.nl/pmwiki/?n=Main.Home.

[28] Peersim: A peer-to-peer simulator. [Online]. Available: http://peersim.sourceforge.net/

[29] S. Saroiu, P. K. Gummadi, and S. D. Gribble, "Measurement study of peer-to-peer file sharing systems," *MMCN*, 2002.

[30] J. R. Douceur and R. P. Wattenhofer, "Large-scale simulation of replica placement algorithms for a serverless distributed file system," in *Proc. MASCOTS*, 2001.