Ensuring β-Availability in P2P Social Networks

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

* Dept. of Comp. Sc. and Engg., Bangladesh University of Engg. and Tech.
nshahriar@cse.buet.ac.bd

†David R. Cheriton School of Computer Science, University of Waterloo
{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 content replication protocol that exploits the cyclic diurnal pattern in user uptime behaviour to ensure content persistence with minimal replication overhead. We also introduce the concept of β-availability, and propose a mechanism for ensuring the availability of at least β members within a replication group at any given time. Simulation results show that a 2-availability grouping policy delivers high content persistence without incurring significant network and storage overheads.**

I. Introduction

Online Social Networks (OSNs) attract increasing numbers of Internet users. 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. It 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 their 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.

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 [1], [2], [3]. However, as addressed by these research efforts, a decentralized OSN solution inherits a very challenging problem from P2P storage systems: *how to ensure 24X7 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 ([4], [5], [6]) 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 grouping and content replication protocol that exploits the cyclic diurnal availability pattern in user uptime behavior as observed in [7]. We store and replicate content within small user groups where users have complementary online patterns and assign the task of indexing group information and content meta information at more stable computing nodes. In order to ensure high availability and low latency access across a geographically distributed setting, we introduce the concept of β-availability, *i.e.*, at least β members of a replication group will be online at any given time. To the best of our knowledge, this is the first attempt to introduce the notion of β-availability. The aforementioned group formation problem is challenging due to three constraints. First, group size should be as small as possible. Second, β-availability should be ensured. 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 utilizes a structured Distributed Hash Table (DHT) for ensuring globally optimal availability groups. This ensures higher system availability without generating high network traffic for group formation. Second, S-DATA can ensure β-availability, as opposed to the 1-availability provided by the existing approaches.

We organize the rest of this paper as follows. We start with a discussion of the related works on decentralized social networks and P2P availability in Section II. Then we provide a conceptual overview of S-DATA (Section III) followed by the protocol details (Section IV). We also simulate the availability architecture to show its performance (Section V). Finally we conclude with some future directions of our work (Section VII).

II. Related Works

A. Decentralized Online Social Networks

A number of recent research efforts strive to decentralize storage and control in OSNs. These efforts have been motivated by the limitations posed by the centralized architecture of current OSNs [8]. However, one of the main challenges that need to be addressed before these systems can be deployed is ensuring high availability of the shared content. PeerSon [1], SafeBook [9], Persona [10], and SuperNova [3] are some of the prominent proposals of decentralized OSNs. They recognize the problem of ensuring 24/7 content availability over a geographically distributed user base as one of the key challenges in deploying decentralized OSNs. However, PeerSon, SafeBook and Persona have not particularly focused on the replication schemes fundamental for increasing content availability. SuperNova allows users to 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.

B. Time Based Replication in P2P Network

A number of approaches to improve availability in P2P storage systems can be found in the literature [11]. However, only a few approaches focus on increasing content availability using a time-based replication strategy. In [4], we proposed the *DATA* protocol that constructs replication groups using complementary availability patterns of peers through a gossip based routing technique. In S-DATA, we improve over the DATA protocol in terms of the availability of the replication groups and network overhead. Moreover, we introduce of the concept of β-availability, whereas DATA targets to ensure 1-availability.The redundancy group based approach proposed by Schwarz et al. [12] tries to improve availability by utilizing the cyclic behaviour of geographically distributed peers. Song et al. [13] proposed a probabilistic model for time-based replica placement in P2P networks using a DHT. They use one hop flooding to find the peers with the most dissimilar availability pattern, while we use a DHT for this purpose and improve significantly on network bandwidth consumption. Rzadca *et al.* [6] proposed to represent peer availability as a function of discrete time to minimize the number of replicas. They represent availability 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. Moreover, their model only targets to ensure 1-availability across time slots, whereas we formulate the concept of β-Availability to provide better reliability.

III. Conceptual Overview

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. In the following we explain each of these three components. In this work, we use the terms peer and user interchangeably.

Replication group : In S-DATA, users are clustered into small groups based on their diurnal availability pattern. Within a replication group, users have mutually exclusive uptime with little overlap. In a replication group with β-availability, it is ensured that at least β members from that group will be online at any given time. All members within a group replicate each others content and serve as proxies for off-line members of that group.

Group index overlay : It has two functions. First, during group formation, it works as a distributed agent for matchmaking users with partial complementary uptime behavior. Second, it acts as an indirection structure during content lookup. Initially each user advertises his availability pattern as a bit-vector to this overlay. During group formation, users willing to form a group search for other users (or groups) having complementary uptime behavior. We use Plexus [14] as the indexing and routing protocol for GIO. To the best of our knowledge, Plexus is the only Distributed Hash Table (DHT) technique that supports approximate bit-vector matching in an efficient manner. At any given time, Plexus maps a group ID to one (or β) online user from that group.

Content index overlay : This overlay can be implemented using any DHT-technique depending on the application-specific requirements. It maps a content name to a group ID. In order to search and download a content, a user will first search the CIO and discover a group ID. Then it will lookup the group ID in the GIO and find the location (IP:port) of an alive user currently hosting that content and download it. Mapping a content name to a group ID, instead of directly mapping to a user ID incurs an additional lookup. But, this lookup is necessary for dynamically associating a content name to the currently online user hosting that content.

B. Availability Vector

The traditional definition of availability is simply measured by the fraction of time a user is online [15] 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, $\frac{\sum_{k=1}^{m} t_k}{T}$. According to Yang *et al.* [16] this formula does not take into account the diurnal availability pattern in user uptime behaviour.

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.

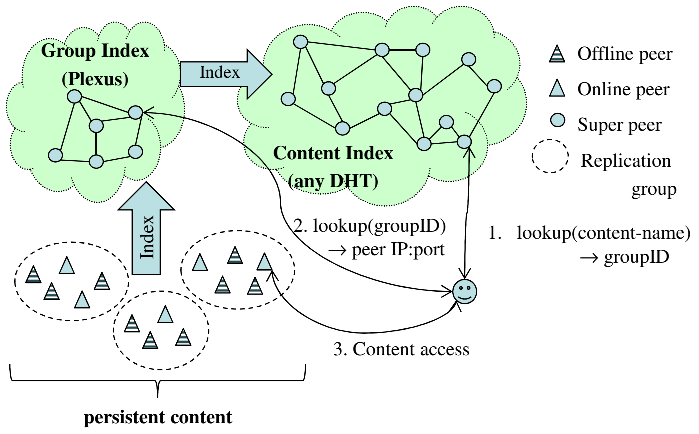

Fig. 1. Conceptual Architecture of S-DATA

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.

IV. S-DATA Protocol Details

A. 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 GIO 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-B1), availability vector ($\mathcal{A}_e$), ID (ID_e) and index location ($\mathcal{I}_e$) for e. $\mathcal{V}_e$ is stored in all nodes $\mathcal{L}_{S_e}$ 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/$\mathcal{I}_G$	$< \mathcal{A}_G, ID_G, \{< ID_x, \mathcal{I}_x, Loc_x > \| x \in G\} >$
$\mathcal{V}_e$	GIO/$\mathcal{L}_{S_e}$	$< S_e, \mathcal{A}_e, ID_e, \mathcal{I}_e >$
$\mathcal{K}_w$	CIO/$\mathcal{J}_w$	$< ID_G, \mathcal{I}_G, ID_{doc}, \{w_i \| w_i \in doc\} >$

B. 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 steps are explained in the following.

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 $< S_x, \mathcal{A}_x, ID_x, Loc_x, ID_{G_x}, \mathcal{I}_{G_x} >$, 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.

C. Group Formation

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:

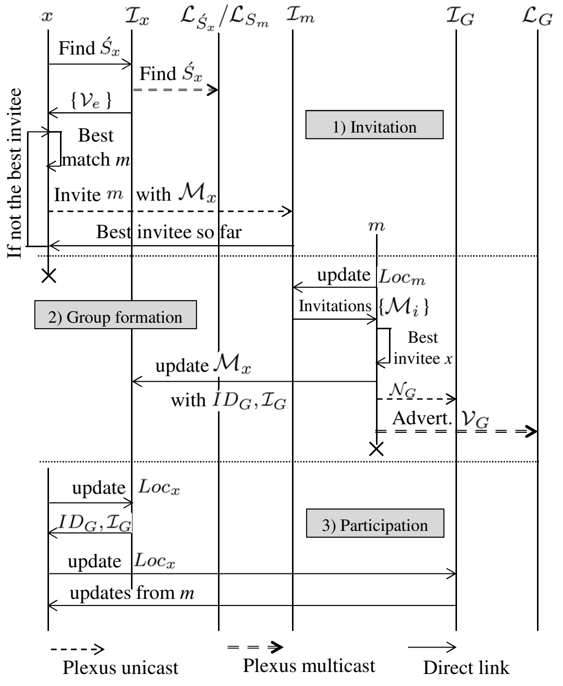

Fig. 2. Sequence diagram shows group formation of x with m

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 $t + L$ to $t + 2L - 1$ set to 1, assuming that the availability pattern S_x of user x has bits t to $t + L - 1$ 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. 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, then 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) finally, m stores 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.

D. 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-C. 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.

E. 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 $\mathcal{J}_w$. On the other hand, 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. Performance Evaluation

We used the Peersim [17] 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 network overhead of our grouping protocol and compare it with other grouping approaches, i.e., random, unstructured, and centralized grouping approaches (Section V-A). Second, we show the advantages of β-availability in terms of fault resilience along with the associated network and storage overheads (Section V-B). We use Pareto distribution for generating the availability vectors based on the observations in [18]. 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.

A. Grouping Efficiency

We perform the simulations in this section for an expected uptime distribution $L = 8$ hours, and vary the network size from 5000 to 30000 in steps of 5000. We compare S-DATA with the following approaches:

- *Unstructured*: We use the gossip protocol as proposed in [4] 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.
- *Centralized*: 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. The Oracle chooses the best invitee, from the invitations for each user and forms a group.

1) Network Overhead: Normalized Message Overhead (NMO) is computed as the ratio of invitation count to the number of successful replies. NMO represents the number of requests required to successfully join a group. Fig. 3 shows a lower NMO of S-DATA compared to that of the random and unstructured approaches. The random and unstructured methods flood the network with messages to find and invite suitable users. On the other hand, the centralized approach does not flood the network and only involves communication between the users and the Oracle only. Therefore, it is used as a baseline for comparing message overhead. Instead of flooding the network, S-DATA uses Plexus to match suitable users. This results in an NMO very close to that of the centralized case and significantly lesser than that of the unstructured and random cases.

B. Performance of S-DATA

We perform the simulations in this experiment for an expected uptime distribution, $L = 8$ hours. We vary the network size from 5000 to 30000 in steps of 5000, and vary β from 1 to 5.

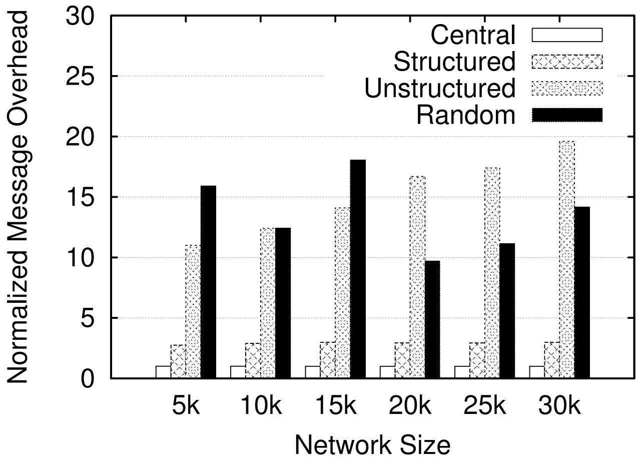

Fig. 3. Normalized Message Overhead

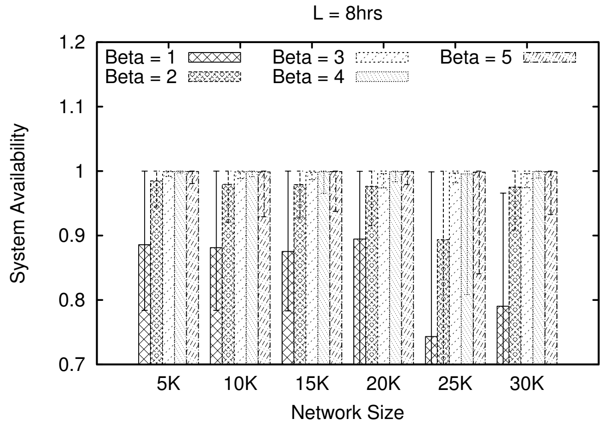

Fig. 4. System Availability

1) System Availability: Fig. 4 presents the expected system availability for different replication levels (β) and network size. 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.

2) Mean Group Size: Fig. 5 shows the effect of changing β on the group sizes ($|G|$) while the network size was kept fixed at 30000. Theoretically, group size ($|G|$) is proportional to β-availability and inversely proportional to L. In other words, a larger group is required to ensure higher β-availability for a given L. This relation is well reflected in the simulation.

3) Failure Resilience: We measure the failure resilience of the groups by taking the percentage of groups having at least one peer online during a time slot under different levels of failure. We have conducted this simulation in a network of 30000 users. We simulate failure by making a user offline during its expected online slot. Fig. 6 shows the effect of

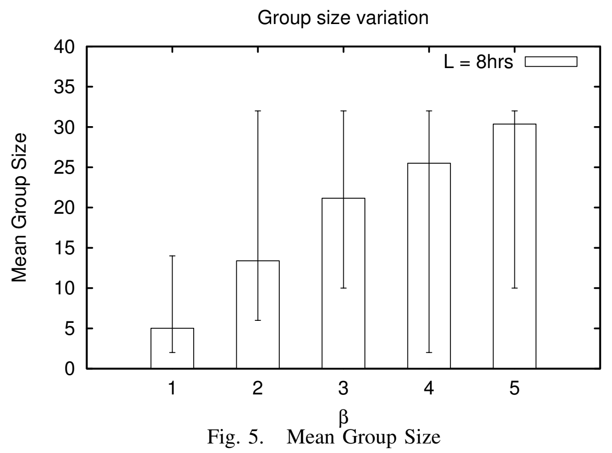

Fig. 5. Mean Group Size

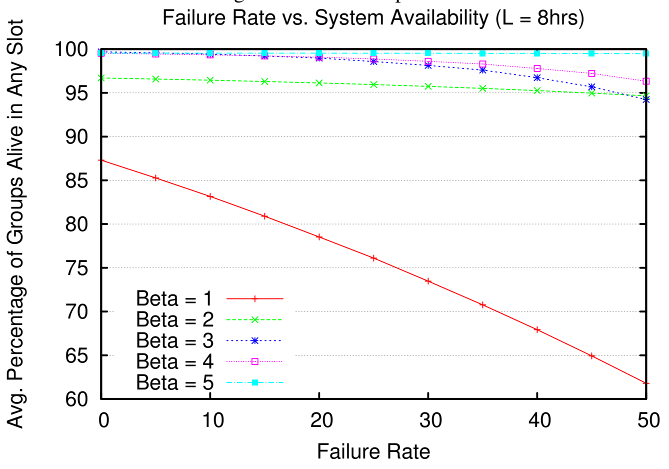

Fig. 6. Effect of Failure on System for Different β

failure on system availability. We can see that with higher values of β the curves are getting more flat. This confirms the improvement in system availability with higher β values. For a mean peer uptime of 8 hours it is possible to have more than 93% of the groups online even under 50% failure rate, with a $\beta >= 3$. As such an increase of β from 1 to 3 brings a significant improvement in terms of the system's availability. However, for further increases in β the improvements are not as significant.

The simulation results suggest that it is possible to achieve high system availability without incurring significant overhead in terms of messaging and convergence time for $\beta = 2$. Therefore, $\beta = 2$ can be a good operating point for a user to ensure high availability. Availability of popular contents can be further increased by caching the contents at other users out of the replication group.

VI. Acknowledgement

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

VII. Conclusion

In this paper, we have introduced the β-availability concept and described an efficient grouping protocol (S-DATA), which ensures data availability around the clock. Simulation results show that the proposed S-DATA protocol ensures very high availability of contents comparable to a centralized group formation protocol. The simulation results also revealed that ensuring 2-availability in replication groups can provide high availability of contents and resilience to peer failure while incurring low network overhead and convergence time. In the future, we intend to deploy S-DATA on a real world system and further investigate its performance for application specific availability requirements. The success of S-DATA also depends on the willingness and truthfulness of the peers. Tackling the potentially malicious behaviour of peers and security issues of group formation is another prospective research issue we plan to investigate.

References

[1] S. Buchegger, D. Schiöberg, L.-H. Vu, and A. Datta, "Peerson: P2P social networking: early experiences and insights," in *Proceedings of SNS*, 2009, pp. 46–52.

[2] 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 MCS*, 2010, pp. 8:1–8:8.

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

[4] 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.

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

[6] 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.

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

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

[9] 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.

[10] R. Baden, A. Bender, N. Spring, B. Bhattacharjee, and D. Starin, "Persona: an online social network with user-defined privacy," in *Proceedings of the ACM SIGCOMM 2009 conference on Data communication*, 2009, pp. 135–146.

[11] R. Hasan, Z. Anwar, W. Yurcik, L. Brumbaugh, and R. H. Campbell, "A survey of peer-to-peer storage techniques for distributed file systems," in *ITCC (2)*, 2005, pp. 205–213.

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

[13] G. Song, S. Kim, and D. Seo, "Replica placement algorithm for highly available peer-to-peer storage systems," in *AP2PS*, 2009, pp. 160–167.

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

[16] 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.

[17] A. Montresor and M. Jelasity, "PeerSim: A scalable P2P simulator," in *Proc. of IEEE P2P*, 2009, pp. 99–100.

[18] F. Bustamante and Y. Qiao, "Friendships that last: Peer lifespan and its role in p2p protocols," in *Web Content Caching and Distribution*, F. Douglis and B. Davison, Eds. Springer Netherlands, Sept. 2004, pp. 233–246.