Ensuring β-Availability in P2P Social Networks

Nashid Shahriar++, Shihabur R. Chowdhury*, Mahfuza Sharmin**, Reaz Ahmed*, Raouf Boutaba*, and Bertrand Mathieu+

++Dept. of CSE, Bangladesh University of Engineering & Technology
*David R. Cheriton School of Computer Science, University of Waterloo
**Department of Computer Science, University of Maryland, College Park
+Orange Labs, France

Presented By: Shihabur R. Chowdhury
People use Online Social Networks (OSNs), e.g., Facebook, Flickr, Google+ *etc.* to share contents with their friends
Background

- People use Online Social Networks (OSNs), e.g., Facebook, Flickr, Google+ *etc.* to share contents with their friends

- Existing OSNs have a centralized view from outside
  - Creates content silos, not interoperable with each other
  - Uses user data for their profit, *e.g.*, in advertisement
  - Users have to agree to future changes in terms of service
Background

- People use Online Social Networks (OSNs), e.g., Facebook, Flickr, Google+ etc. to share contents with their friends.

- Existing OSNs have a centralized view from outside.
  - Creates content silos, not interoperable with each other.
  - Uses user data for their profit, e.g., in advertisement.
  - Users have to agree to future changes in terms of service.

- How to overcome these shortcomings?
  - Decentralize the OSN infrastructure. Do social networking in a more P2P way.
  - Diaspora, PeerSon, SafeBook, SuperNova, Cachet, PrPl are a few approaches to decentralize OSN.
The Problem

- One important question still remains to be answered
  - *How to ensure 24 x 7 content availability with minimal replication overhead?*

- Existing Solutions
  - *The DOSNs are still in early stage and does not provide enough discussion about ensuring availability*
Our Contribution

- **We propose**
  - The notion of $\beta$-availability
    - At least beta members of a replication group will be online
  - **S-DATA protocol**
    - A time based replication group formation protocol to ensure $\beta$-availability
    - Uses structured overlay, *i.e.*, Distributed Hash Table (DHT) to maintain replication groups, advertise availabilities, and resolve queries
Availability Representation

\[ a_{ix} = \text{the probability of user } x \text{ being online during time slot } x, \ 1 \leq x \leq 24 \]

Availability vector (A)

\[
\begin{array}{cccc}
0.2 & 0.1 & \ldots & 0.9 \\
\end{array}
\]

\[
\begin{array}{cccc}
0.9 & 0.9 & \ldots & 0.1 \\
0.0 & & & \\
\end{array}
\]
Availability Representation

\[ a_{ix} = \text{the probability of user } x \text{ being online during time slot } x, \quad 1 \leq x \leq 24 \]

Encoded \( A \) into Linear Binary Code
- Take pairwise average in \( A \)
- Encode each element to 2-bit binary

Availability vector (\( A \))

\[
\begin{array}{cccccccc}
0.2 & 0.1 & \cdots & 0.9 & 0.9 & \cdots & 0.1 & 0.0 \\
\end{array}
\]

Availability pattern

\[
\begin{array}{cccccccc}
0 & 0 & \cdots & 1 & 1 & \cdots & 0 & 0 \\
\end{array}
\]
Availability Representation

\( a_{ix} \) = the probability of user \( x \) being online during time slot \( x \), \( 1 \leq x \leq 24 \)

Encoded \( A \) into Linear Binary Code
- Take pair wise average in \( A \)
- Encode each element to 2-bit binary

Availability vector (\( A \))

\[
\begin{array}{cccccccc}
0.2 & 0.1 & \ldots & 0.9 & 0.9 & \ldots & 0.1 & 0.0 \\
\end{array}
\]

Availability pattern

\[
\begin{array}{cccccccc}
0 & 0 & \ldots & 1 & 1 & \ldots & 0 & 0 \\
\end{array}
\]

Complement

\[
\begin{array}{cccccccc}
1 & 1 & \ldots & 0 & 0 & \ldots & 1 & 1 \\
\end{array}
\]

Advertise

Search

DHT

Result

\[
\begin{array}{cccccccc}
1 & 1 & \ldots & 0 & 0 & \ldots & 1 & 1 \\
\end{array}
\]

encode \( a_{ix} \) into Linear Binary Code
- Take pair wise average in \( A \)
- Encode each element to 2-bit binary

\( a_{ix} \) = the probability of user \( x \) being online during time slot \( x \), \( 1 \leq x \leq 24 \)
System Architecture

- Three major conceptual components
  - Group Index Overlay (GIO)
  - Content Index Overlay (CIO)
  - Replication Groups
System Architecture: GIO

- Stores mapping for group ID to its member peers
- Acts as distributed matchmaking agent
  - Given a user’s availability pattern, find other users with complementary availability patterns
- Given a user’s availability bit pattern, we need to perform partial matching in the GIO DHT
  - Till date, only Plexus (Ahmed et al. TON 2009) is known to have this capability
  - Therefore, we use Plexus as GIO
System Description: CIO and Replication Groups

- CIO
  - Maps content names to group IDs
  - Out of the paper’s scope

- Replication Groups
  - Users are clustered based on their diurnal availability patterns
  - All members of the group replicate each other's contents
Protocol Description

GIO

User A

Searches for users with availability pattern similar to User A's complementary availability pattern

User B
Performs partial search in Plexus DHT to find users with availability pattern similar to User A’s complementary availability pattern
Protocol Description

GIO

User A

Returns a list of users matching the pattern (User B, User C, User D)

User B
Selects User B, since User B’s availability pattern has minimum hamming distance from the desired pattern.
Protocol Description

Requests to send a group formation invitation to User B when it comes online

User A

GIO

User B
Protocol Description

User A

GIO

User B becomes online and retrieves all group formation invitation from GIO

User B
Protocol Description

GIO

User A

User B

User B selects the best invitation and discards the rest
Protocol Description

GIO

Notifies GIO about acceptance of User A’s invitation. All the indices in GIO are updated.

User A

User B
Evaluation

- **Setup**
  - We used **PeerSim** to simulate the protocol
  - **Pareto distribution** was used to generate availability vectors
  - **Extended Golay Code** used for encoding

- **We measured**
  - **Normalized Messaging Overhead**
    - Number of invitations required for forming a single group
    - Compared it with Random, Central and Unstructured grouping approaches
  - **System Availability**
    - Probability of having at least one online user from a group at any given time
  - **Effect of Failure**
    - Probability of having at least one member of a group online when certain percentage of users do not become online in their expected online slot
Evaluation: Results

- Normalized Messaging Overhead
  - Network size increased from 5000 to 30000 in steps of 5000
  - Central approach is baseline
  - Our approach has overhead very close to the central approach
  - Very little effect of the network size

![Normalized Message Overhead](image)
System Availability

- A significant improvement in system availability when $\beta$ increases from 1 to 2
- Improvements for higher beta are very less

![Diagram showing system availability over network size for different values of beta.](image)

Fig. 4. System Availability
Effect of Failure

For $\beta \geq 2$, more than 93% groups are available even after 50% users failing to be online in their expected period.
Evaluation: Take Away

- $\beta = 2$ is a good operating point
  - Can achieve high system availability
  - Lower overhead
  - 93% groups are online even after 50% nodes failing
Ensuring availability in a decentralized social network with not so stable users and taking the social relationship of the peers is challenging.

We take a **first step** towards solving the problem and **solve it without considering social relationships.**

We also **introduce** the notion of **beta-availability.**

In the **next step** we are considering **social relationships.**

Simulation results show $\beta = 2$ is a good operating point.
Questions?