Learning the News in Social Networks

Krishnan Rajagopalan[1], Venkatesh Srinivasan[2], and Alex Thomo[3]

[1] Digital Media Technologies, Motion Picture Association of America, Los Angeles, CA, USA, krishnan_rajagopalan@yahoo.com
[2] Department of Computer Science, University of Victoria, Victoria, BC, Canada, venkat@cs.uvic.ca
[3] Department of Computer Science, University of Victoria, Victoria, BC, Canada, thomo@cs.uvic.ca

Abstract. In social media such as facebook, the most popular desire is to learn the news about other people. In this paper, we study the following problem related to information propagation: Suppose that there is a set U of N users in a social network. They meet online from time to time and share information they know about themselves and the other users in the network. Whenever a group $g \subset U$ of users meet, they want to know who has the latest information about every user in U. A naive solution to this problem is to use timestamps. However, there are drawbacks to this scheme including the burden on the users to maintain reliable timestamps and the fact that the timestamps grow unbounded over time. It is natural to ask if it is possible to learn the latest information without using timestamps. We present an efficient method which removes the need to timestamp user information (news). Instead, only the meetings of the groups have to be indexed. Furthermore, we show that this indexing can be performed using a finite set of labels so that each user stores at most $O(N^2 \log N)$ bits of information. We also show that this bound can be improved in some cases if we have further information on the topology of the network.

1 Introduction

Today, it is hard to find people, at least from certain demographics, who are not part of a social network, such as Facebook, Twitter, LinkedIn, and so on. One of the main reasons for joining a social network is to satisfy the inherent human need of learning "the news" about other people. In virtually all the social networks, there are implicit or explicit groups formed for the sole purpose of disseminating information. However, this bears the question of how to determine the latest information about users in a network.

The setting we study in this paper is as follows. Suppose that there is a set U of N users in a social network. They meet online in groups from time to time and share information they know about themselves and other users in the network. Whenever a group $g \subset U$ of users meet, they want to know who has the latest information about every user in U. We assume honest users who are willing to share all they know with the other members of the group. A user can be a member of more than one group, and thus serve as a "bridge" for news propagation. Also we assume that during a group meeting each member of the group gets up to date with respect to all the users that the other group members have information (news) for. Therefore, for a given user $u \in U$, the members of a group g, need to determine who (in the group) has the latest information about u.

A naive solution to this problem is to use timestamps. However, there are two problems with letting users timestamp their personal information. First, there is the burden on each user to produce and use reliable timestamps conforming to some standard. Second, there is an additional problem that the label set for such timestamps grows unbounded over time. Can we avoid this?

In this paper, we present an efficient method to infer the latest information without using timestamps for the user information. We present an algorithm which is able to determine who has the latest information for a given user by reasoning about a "latest information graph" structure we introduce. In this algorithm, only the meetings of the groups have to be indexed (numbered). These indexes can be considered as a form of rudimentary timestamps, but they are on the group meetings level, not on the personal user information level.

Our Contributions. More concretely, we make the following contributions in this paper.

- We introduce the notion of the "latest information graph" that allows users present in a meeting to compare and choose the latest information they have about any other user in the network.
- For a network of N users with no restriction on the network topology, we show that it is sufficient for each user to maintain $O(N^2 \log N)$ bits of information improving a previous bound of $O(N^3 \log N)$[9].
- For the special case of mesh topology, the n-dimensional hypercube, we show that it is sufficient to maintain $O(\log^3 N)$ bits of information for $N = 2^n$ users. This is a significant improvement over the general case.

Related Work. "Gossiping" is a well studied problem in distributed systems [6]. In the traditional setting of this problem, a user has a piece of information that needs to be communicated to all the other users and the goal is to design a protocol that minimizes the total communication needed for a fixed network topology.

Bounded time-stamps have been studied for a long time in various models of distributed systems. Israeli and Li introduced this notion [7] for shared memory model. Their results were further improved in [3] and [4]. However, we study this problem for social networks that can be viewed as an asynchronous communication model and their results are not comparable to ours. Another direction of research studies this problem in the model of asynchronous cellular automata that starts with a different assumptions for interaction (See [1], [2]).

Very recently, Lind *et al.* [8] use a simple model to understand information propagation in scale-free and small-world networks and compare their results to real empirical network of social acquaintances. Fan *et al.* [5] point out the unreliability of time-stamps in the setting of relational databases and the need to maintain data currency in the absence of timestamps.

Organization. In Section 2, we formally define our setting and give a characterization lemma about latest information. Sections 3, 4 and 5 explain how the users compare and update their latest information during meetings. We outline our algorithm and its analysis is Section 6. In Section 7, we prove our result on the special case of hypercubes. We end with our conclusions in Section 8.

2 Groups, Meetings, Information

We denote by U the set of users in a social network and by G the set of groups the users have created over time. Each group has a name, and for simplicity, we will blur the distinction between a name and the group it names. There are possibly many group meetings, and in each such occasion, the members of the group exchange the latest information about themselves and other users they know about in the network. We assume that a user, who belongs in more than one group, does not participate in more than one meeting at the same time. Also we assume that a group meeting can happen only if all the users are available and willing to participate.

If two groups g, h have some user in common, we say that they are *bridged*. Given a group $g \in G$, we denote its meetings by $g_1, g_2, \ldots$. Consider a sequence σ of meetings of different groups (e.g. $g_1h_1g_2k_1h_2$). We say that a meeting h_j *depends on* a meeting g_i, and write $g_i \prec h_j$, if groups g and h are bridged, and g_i happened before h_j in time. We denote by $\preceq^*$ the reflexive and transitive closure of $\prec$.

Latest Information. Given a sequence σ of meetings, and a user u, let g_i be the last meeting that u participates in. Let σ_u be the subsequence of σ containing only meetings which are "smaller" ($\preceq^*$) than g_i. Given another user t, let e_j be the last meeting in σ_u that t participates in (of course $t \in e_j$). Clearly, the information that u has about t corresponds to this meeting e_j. We denote g_i by $last_u(\sigma)$ and e_j by $last_{u,t}(\sigma)$.

For any three users u, v, and t, we have that $last_{u,t}(\sigma)$ can be put in a $\preceq^*$ relationship with $last_{v,t}(\sigma)$. This is because both $last_{u,t}(\sigma)$ and $last_{v,t}(\sigma)$ are meetings where t participates. In general the meetings where a user t participates are called *t-meetings.* Any two t-meetings can be compared with respect to $\preceq^*$. Therefore, $\preceq^*$ is a total order for t-meetings. Purely for convenience of notation, we assume that there is an initial meeting labeled 0 involving all the users.

Figure 1 shows a communication sequence $\sigma = f_1g_1h_1g_2$ of four meetings. There are four users u, v, w and x. Note that $last_u(\sigma) = g_2$ and $last_{u,w}(\sigma) = h_1$. Furthermore, $\sigma_x = f_1g_1h_1$ while $\sigma_u = f_1g_1h_1g_2$. $g_1 \preceq h_1$ as they have a user v in common. Also note that $f_1 \preceq^* g_2$ because $f_1 \preceq h_1$ and $h_1 \preceq g_2$.

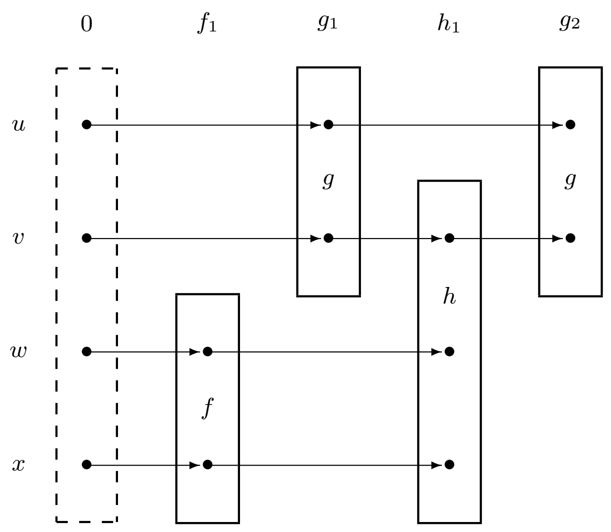

Fig. 1. An example

Lemma 1. *Let u, v, and t be three users in the network, and let σ be a communication sequence. Then one of the following holds.*

(i) $last_{u,t}(\sigma)$, $last_{v,t}(\sigma) \in \sigma_u \cap \sigma_v$ *and* $last_{u,t}(\sigma) = last_{v,t}(\sigma)$.
(ii) $last_{u,t}(\sigma) \in \sigma_u \cap \sigma_v$, $last_{v,t}(\sigma) \in \sigma_v \setminus \sigma_u$ *and* $last_{u,t}(\sigma) \preceq^* last_{v,t}(\sigma)$.
(iii) $last_{v,t}(\sigma) \in \sigma_u \cap \sigma_v$, $last_{u,t}(\sigma) \in \sigma_u \setminus \sigma_v$ *and* $last_{v,t}(\sigma) \preceq^* last_{u,t}(\sigma)$.

Proof. **Case 1.** Since $last_{u,t}(\sigma)$, $last_{v,t}(\sigma)$ are t-meetings, and both in σ_v, we have $last_{u,t}(\sigma) \preceq^* last_{v,t}(\sigma)$. By a symmetric argument, substituting σ_u for σ_v, we get $last_{v,t}(\sigma) \preceq^* last_{u,t}(\sigma)$. Therefore, $last_{u,t}(\sigma) = last_{v,t}(\sigma)$.

Case 2. One of $last_{u,t}(\sigma)$ or $last_{v,t}(\sigma)$ belongs to $\sigma_u \cap \sigma_v$ and the other does not belong to $\sigma_u \cap \sigma_v$. Suppose $last_{u,t}(\sigma) \in \sigma_u \cap \sigma_v$ and $last_{v,t}(\sigma) \in \sigma_v \setminus \sigma_u$. Then, since $last_{u,t}(\sigma) \in \sigma_v$ and we have $last_{u,t}(\sigma) \preceq^* last_{v,t}(\sigma)$. A similar argument holds for the other possibility.

Case 3. Let, if possible, $last_{u,t}(\sigma) \in \sigma_u \setminus \sigma_v$ and $last_{v,t}(\sigma) \in \sigma_v \setminus \sigma_u$. Since, $last_{u,t}(\sigma)$ and $last_{v,t}(\sigma)$ are both t-meetings, they must be totally ordered, i.e, either (a) $last_{u,t}(\sigma) \preceq^* last_{v,t}(\sigma)$ or (b) $last_{v,t}(\sigma) \preceq^* last_{u,t}(\sigma)$. Suppose we have (a). This means that $last_{u,t}(\sigma) \in \sigma_v$ because σ_v is the subsequence of σ containing all the meetings that are $\preceq^*$ than $last_{v,t}(\sigma)$. $last_{u,t}(\sigma) \in \sigma_v$ is a contradiction. A similar argument holds for the other possibility. $\square$

3 Latest Information Graph

The latest information that a user u has at the end of a meetings sequence σ can be organized as a graph $\mathcal{G}_u(\sigma) = (V_u(\sigma), E_u(\sigma))$, where

$$\begin{aligned} V_u(\sigma) &= \{(t, last_{u,t}(\sigma)) : t \in U\} \\ E_u(\sigma) &= \{[(t, last_{u,t}(\sigma)), (t', last_{u,t'}(\sigma))] : \\ &\qquad\qquad last_{u,t}(\sigma) \preceq^* last_{u,t'}(\sigma)\}. \end{aligned}$$

Since $\preceq^*$ is reflexive, a self-loop edge is also contained in $E_u(\sigma)$ for each node in $V_u(\sigma)$.

Whenever two users u and v meet in some group meeting, they want to decide who has the latest information about some other user t. For this we present the following lemma.

Lemma 2. *$last_{u,t}(\sigma) \preceq^* last_{v,t}(\sigma)$, if and only if, there exists $t', t'' \in U$, such that*

$$\begin{gathered} last_{u,t'}(\sigma) = last_{v,t''}(\sigma) \\ [(t, last_{u,t}(\sigma)), (t', last_{u,t'}(\sigma))] \in E_u(\sigma) \end{gathered}$$

Proof. ($\Rightarrow$). $last_{u,t}(\sigma) \preceq^* last_{v,t}(\sigma)$. We have the following possibilities.
Case 1. $last_{u,t}(\sigma) = last_{v,t}(\sigma)$. From Lemma 1, we have, $last_{u,t}(\sigma), last_{v,t}(\sigma) \in \sigma_u \cap \sigma_v$, which implies

$$(t, last_{u,t}(\sigma)) \in V_u(\sigma) \cap V_v(\sigma).$$

Since we always have self-loops in information graphs, we have
$[(t, last_{u,t}(\sigma)), (t, last_{u,t}(\sigma))] \in E_u(\sigma)$. We set now $t' = t$ and the claim follows.
Case 2. $last_{u,t}(\sigma) \neq last_{v,t}(\sigma)$. From Lemma 1, we have $last_{u,t}(\sigma) \in \sigma_u \cap \sigma_v$ and $last_{v,t}(\sigma) \in \sigma_v \setminus \sigma_u$. Let t' be a user who participates in the $\preceq^*$-greatest meeting in $\sigma_u \cap \sigma_v$. Clearly, $last_{v,t}(\sigma) \preceq^* last_{u,t'}(\sigma)$. Thus, we have

$$[(t, last_{u,t}(\sigma)), (t', last_{u,t'}(\sigma))] \in E_u(\sigma).$$

Also, it is clear that $(t', last_{v,t'}(\sigma)) \in V_v(\sigma)$.

($\Leftarrow$). To prove this direction, we observe that $last_{u,t}(\sigma) \preceq^* last_{u,t'}(\sigma)$ and $last_{u,t'}(\sigma) \in \sigma_u \cap \sigma_v$. Therefore, $last_{u,t}(\sigma)$ also belongs to $\sigma_u \cap \sigma_v$. Now it follows from Lemma 1 that $last_{u,t}(\sigma) \preceq^* last_{v,t}(\sigma)$. From these facts the claim follows. □

4 Updating Latest Information

Let $\mathcal{G}_u(\sigma') = (V_u(\sigma'), E_u(\sigma'))$ be the latest information graph for u and $\mathcal{G}_v(\sigma') = (V_v(\sigma'), E_v(\sigma'))$ be the latest information graph for v at the end of communication sequence σ'. After a meeting h involving two users u and v, each of them update their latest information for the communication sequence $\sigma = \sigma' h$ by constructing a common updated graph $\mathcal{G}_{uv}(\sigma) = (V_{uv}(\sigma), E_{uv}(\sigma))$ as follows :

(i) Both $V_{uv}(\sigma)$ and $E_{uv}(\sigma)$ are set to $\emptyset$.
(ii) Let the new meeting under which u and v have synchronized be labeled by l. The process by which a new label l is assigned to the meeting is explained later.
(iii) $(u, l), (v, l)$ are added to $V_{uv}(\sigma)$.
(iv) For each $w \in U - \{u, v\}$, the pair (w, e) is added to $V_{uv}(\sigma)$, where e is the latest of what u and v know about w(as given in lemma 2).
(v) For $s \in \{u, v\}$ and for each $w \in U - \{u, v\}$, $((w, e), (s, l))$ is added to $E_{uv}(\sigma)$.
(vi) For all $s,\ t$ belonging to $U - \{u, v\}$, $((s, e), (t, f))$ is added to $E_{uv}(\sigma)$ if there exists $r, r' \in U$ s.t $((r, e), (r', f)) \in E_u(\sigma') \cup E_v(\sigma')$.

It can be easily seen that the update procedure ensures that each user maintains not only the latest information about all other users, but also maintains any ordering that exists between these meetings.

From Lemma 2, we know that, to compare latest information, we only need to look at the meetings which are currently in the latest information graph of each user. Recall that the vertices of $\mathcal{G}_u$ and $\mathcal{G}_v$ are identified by *labels*. If a meeting e occurs in both $\mathcal{G}_u$ and $\mathcal{G}_v$, then it must be given the same label in both the graphs. Since the update procedure labels only the current meeting, keeping the other meeting labels unchanged, this is ensured. So far, we have achieved our goal of indexing meetings rather than timestamping user information.

5 Achieving Boundedness of Label Sets

Now, a new label (index) has to be chosen for the current meeting. It must be ensured that the label assigned to this meeting is consistent across the system—i.e., if the same label appears in the current latest information graph of different users, the corresponding meeting is actually the same.

Unfortunately, the users in a meeting g_i cannot directly see all the g-meetings which belong to the latest information graphs of the entire system. A g-meeting e may be part of the latest information of user *outside* g.

To enable the users in g to know about all g-meetings in $\{\mathcal{G}_u(\sigma)\}_{u \in U}$, we will maintain auxiliary information.

Auxiliary information. The *auxiliary information* of user u after σ, $auxiliary_u(\sigma)$, is a set of meetings of the form $last_{v,w}(\sigma_{last_{u,v}(\sigma)})$ for some $v, w \in U$. This is the latest w-meeting which v knows about up to the meeting $last_{u,v}(\sigma)$. We abbreviate $last_{v,w}(\sigma_{last_{u,v}(\sigma)})$ by $last_{u,v,w}(\sigma)$.

We represent each auxiliary meeting $e = last_{u,v,w}(\sigma)$ in $auxiliary_u(\sigma)$ as a quadruple (u, v, w, e). However, we can think of the auxiliary information as a set of meetings. Then, $e \in auxiliary_u(\sigma)$ will indicate that for some $v, w \in U$, $(u, v, w, e) \in auxiliary_u(\sigma)$.

Lemma 3 ([9]). *Let σ be a communication sequence, $u \in U$ and e a u-meeting in σ. If $e \notin auxiliary_u(\sigma)$, then e is not a meeting in $\mathcal{G}_w(\sigma)$ for any user $w \in U$.*

So, a user u can keep track of which of its labels are "in use" in the system by maintaining auxiliary information. Each u-meeting e initially belongs to $\mathcal{G}_u(\sigma)$, and hence to $\mathit{auxiliary}_u(\sigma)$ as well. As the computation progresses, e gradually "recedes" into the background and disappears from the latest information graphs of the system. Eventually, when e disappears from $\mathit{auxiliary}_u(\sigma)$, u can be sure that e no longer belongs to $\mathcal{G}_w(\sigma)$ for any user $w \in U$.

Since $\mathit{auxiliary}_u(\sigma)$ is a bounded set, u knows that only finitely many of its labels are in use at any given time. So, by using a sufficiently large finite set of labels, each new meeting can always be assigned an unambiguous label by the users which take part in the meeting.

6 Our Algorithm

For convenience, the algorithm starts with an initial meeting involving all the users. This meeting is denoted by $\langle \mathcal{P}, \ell_0 \rangle$ for an arbitrary but fixed label $\ell_0 \in \mathcal{L}$.

Subsequently, for any meeting of a group g, the users in g do the following:

(i) When a new g-labelled meeting e occurs, the users in g assign a label $\langle g, \ell \rangle$ to e which does not appear in $\mathit{auxiliary}_u(\sigma)$ for any user u in g. Lemma 3 guarantees that this new label does not appear in $\mathcal{G}_w(\sigma)$ for any user $w \in U$. Let $N = |U|$. Since each user keeps track of N^2 auxiliary meetings and at most N users can be at a meeting, there need be only N^3 labels in $\mathcal{L}$.

(ii) The users participating in e now compare and update their latest information about each user $w \notin e$ by checking labels of meetings across their latest information graphs as described in Lemma 2 and the update procedure outlined in the previous section.

(iii) Auxiliary information can be locally updated once the users have decided who has the best latest information—if $u, v \in g$ and $\mathit{last}_{u,w}(\sigma)$ is better than $\mathit{last}_{v,w}(\sigma)$ for $w \in U$, then any auxiliary information of the form $\mathit{last}_{u,w,w'}(\sigma)$ must necessarily be better than the corresponding information $\mathit{last}_{v,w,w'}(\sigma)$, for $w' \in U$.

The Amount of Local Information.

Theorem 1. *Each user $u \in U$ needs to maintain at most $O(N^2 \log N)$ bits of information, where $N = |U|$.*

Proof. Each new meeting e is assigned a label of the form $\langle g, \ell \rangle$, where g was the group of users that participated in e and $\ell \in \mathcal{L}$.

To write down $g \subseteq U$, we need, in general, N bits. This component of the label is required to guarantee that all auxiliary meetings in the system have distinct labels, since the set $\mathcal{L}$ is common across all users. However, we do not really need to use all of g in the label for e to ensure this property. If we order U as $\{u_1, u_2, \ldots, u_N\}$, it suffices to label e by $\langle u_i, \ell \rangle$ where, among the users in g, u_i has the least index with respect to our ordering of U.

Thus, we can modify our automaton so that the users label each meeting by a pair $\langle u, \ell \rangle$, where $u \in U$ and $\ell \in \mathcal{L}$. This pair can be written down using $O(\log N)$ bits.

The *latest information* of each user is stored in two parts:

- An array which stores for each user $u \in U$, the latest it knows about u. This requires $N \log N$ bits.
- The edges of the latest information graph are stored in the form of an adjacency matrix which requires N^2 bits.

The *auxiliary information* of each user clearly requires $O(N^2 \log N)$ bits. Hence, the total information can be described in $O(N^2 \log N)$ bits.

In the following section, we show that when we consider particular case of mesh network topology, the hypercube, we can improve the bound even further. We believe that the hypercube is a good representative of a network with "large" groups. A good understanding of this problem for hypercube is relevant to study of social networks with large community sizes.

7 Hypercube

An *n-dimensional hypercube*, $\mathcal{H}_n$, is a network of 2^n users. The users are numbered 0 through $2^n - 1$ and each user is identified by its binary representation. Each face of the hypercube represents a group of the users that meet from time to time. That is, exactly those users which agree on the k^{th} bit, $1 \leq k \leq n$, can form a group. Thus each group involves 2^{n-1} users and there are $2n$ possible groups. We will call the groups $g_1, g_2, \ldots, g_n, g_{n+1}, \ldots, g_{2n}$, with $g_i,\ i \in \{1, \ldots, n\}$, denoting a group of all the users that have a 1 in their i^{th} bit and $g_j,\ j \in \{n+1, \ldots, 2n\}$, denoting a group of all the users that have a 0 in their $(j-n)^{th}$ bit.

We call two groups g_i and g_j *complementary* if either $i = j - n$ or $j = i - n$. This means that the set of the users that take part in g_i and g_j are disjoint and together make up the set of all users in the system.

Note that two different g-meetings cannot be at the same time in the latest information graph of any user during the computation. The following lemma proves this.

Lemma 4. *Let σ be a communication sequence. For any $u, v, w \in U$, let e be a g-meeting and $e = last_{u,v}(\sigma)$. Let $f = last_{u,w}(\sigma)$. Then, f cannot be a g-meeting different from e.*

Proof. Suppose, f is a g-meeting and f is different from e. Then either $e \preceq^+ f$ or $f \preceq^+ e$, since all g-meetings are totally ordered.

Without loss of generality, let $e \preceq^+ f$. Then, v is a user that participated in f and $e \preceq^+ f$. Hence $e \neq last_{u,v}(\sigma)$ which is a contradiction.

A similar argument hold for the other possibility.

In the case of the hypercube, the number of possible groups is just $2n$ and there are 2^n users. It follows that the latest information graph in general is likely to have many copies of the same g-meeting for many groups g. So, can we "collapse" all the occurrences of this meeting and maintain just one vertex corresponding to that meeting in the graph? This will reduce the number of vertices in the latest information graph from $O(2^n)$ to $O(n)$. This motivates the following definition for the latest information graph of a hypercube.

Latest Information Graph. Let σ be a communication sequence and $u, v \in U$. The *latest information graph* of u after σ is a directed graph $\mathcal{G}_u(\sigma) = (V_u(\sigma), E_u(\sigma))$ where

$$V_u(\sigma) = \{e \mid \exists v \in \mathcal{P} \ s.t \ e \ = \ last_{u,v}(\sigma)\}$$

$$E_u(\sigma) = \{(e, f) \mid e \preceq^* f\}$$

That is, the latest information graph is a directed graph in which each vertex has a label of a meeting that corresponds to the latest information that u knows about some other user in U. There is an edge from vertex e to f if e is below f. Note that there are at most $2n$ vertices in this graph.

The latest information that any user u knows about any other user u' in the system can be recovered from the latest information graph of u by finding all the meetings that u' participates in and getting the $\preceq^*$ − maximal meeting from them. We only have to ensure that there exists a proper update procedure to keep updating the graph after each meeting. The following section shows how to maintain and update the latest information graph.

Updating Latest Information. As in the general case, we outline the procedure by which two users can update their latest information when they meet. When a meeting takes place, the users update their information pairwise, and finally propagate the common updated graph to all the users that participated in the current meeting. The peculiarity of the hypercube architecture is that either two meetings are ordered or are complementary to each other. This fact results in a simple update procedure which is outlined below.

Let $\mathcal{G}_u(\sigma') = (V_u(\sigma'), E_u(\sigma'))$ be the old graph for u, $\mathcal{G}_v(\sigma') = (V_v(\sigma'), E_v(\sigma'))$ be the old graph for v, and let $\mathcal{G}_{uv}(\sigma) = (V_{uv}(\sigma), E_{uv}(\sigma))$ denote the common updated graph for u and v where $\sigma = \sigma' g$. When two users u and v meet during a g-meeting, each of them update their latest information by constructing the new graph $\mathcal{G}_{uv}(\sigma)$ as follows :

(i) A new label l is assigned to the current meeting g.
(ii) If either $last_u(\sigma')$ or $last_v(\sigma')$ belongs to the set $S = V_u(\sigma') \cap V_v(\sigma')$:
If $last_u(\sigma') \in$ S:
Let $\mathcal{G}'_{uv}(\sigma) = (V'_{uv}(\sigma), E'_{uv}(\sigma))$ be the initial updated graph. Then,

$$V'_{uv}(\sigma) \ = \ V_q(\sigma') \cup \ l.$$

$$E'_{uv}(\sigma) \ = \ E_q(\sigma') \ \cup \ \{(e, l) \mid e \in {\scriptstyle V}_q(\sigma')\}.$$

We then remove the redundant meetings in $\mathcal{G}'_{uv}(\sigma)$ to get the final updated graph. We say a meeting e is *redundant* in $\mathcal{G}'_{uv}(\sigma)$ if for all u' taking part in e, $e \preceq^+ last_{v,u'}(\sigma)$. Let the set of redundant meetings be denoted by $R_{uv}(\sigma)$. Then,

$$V_{uv}(\sigma) \;=\; V'_{uv}(\sigma) \;-\; R_{uv}(\sigma).$$

If $last_u(\sigma) \in$ S: Do the same with the roles of u and v reversed.

(iii) If neither $last_u(\sigma)$ nor $last_v(\sigma)$ belong to the set S:
Then,

$$V_{uv}(\sigma) \;= \{\; last_u(\sigma) \;\cup\; last_v(\sigma) \;\cup\; l\}.$$

$$E_{uv}(\sigma) \;=\; \{(last_u(\sigma), l), (last_v(\sigma), l)\}.$$

The following lemmas show that the update procedure works correctly.

Lemma 5. *Let $u, v \in U$ and σ be a communication sequence. Consider the set $S = V_u(\sigma) \cap V_v(\sigma)$. Suppose $last_u(\sigma) \in S$. Then, for any $u' \in U$,*

$$e \;=\; last_{u,u'}(\sigma) \quad \preceq^* \quad f \;=\; last_{v,u'}(\sigma).$$

Proof. In $\mathcal{G}_u(\sigma)$, $e \preceq^* last_u(\sigma)$. Also, $last_u(\sigma) \in \mathcal{G}_v(\sigma)$ and $f \;=\; last_{v,u'}(\sigma)$. Hence, $e \preceq^* f$ since all u'-meetings are totally ordered.

Lemma 6. *Let $u, v \in U$ and σ be a communication sequence. Consider the set $S = V_u(\sigma) \cap V_v(\sigma)$. Suppose both $last_u(\sigma)$ and $last_v(\sigma)$ do not belong to S. Then,*

- *For all u' that participated in $last_u(\sigma)$, $last_{v,u'}(\sigma) \preceq^* last_{u,u'}(\sigma)$.*
- *For all u'' that participated in $last_v(\sigma)$, $last_{u,u''}(\sigma) \preceq^* last_{v,u''}(\sigma)$.*

Proof. Suppose that there exists a user u' that participated in $last_u(\sigma)$ such that $last_{u,u'}(\sigma) \preceq^* last_{v,u'}(\sigma)$. Then, $last_{u,u'}(\sigma) \in \sigma_u \cap \sigma_v$. Hence, $last_u(\sigma) \in S$, which is a contradiction. A symmetric argument holds for the other part.

So, whenever two users meet, they can easily compare and update their latest information provided a new label corresponding to the current meeting can be chosen. As in the general case, this is done by looking at the auxiliary information of users.

Eliminating Redundant Meetings Efficiently. We now show how to remove redundant meetings much more efficiently than exhaustive search.

Lemma 7. *Let σ be a communication sequence. Let $u, v \in U$ and $e \in V'_{uv}(\sigma)$. e is* redundant *iff $\exists f, f' \in V'_{uv}$ such that f and f' are complementary meetings, $e \preceq^* f$ and $e \preceq^* f'$.*

Proof. ($\Longleftarrow$) Since f and f' are complementary meetings, and each involves 2^{n-1} users, they together cover all the users in U. Therefore, since $e \preceq^* f$ and $e \preceq^* f'$, there does not exist any user $u' \in U$ such that e is the latest that u (or v) know about u'. Hence e is redundant.

($\implies$) Let e be a redundant meeting. Suppose e is a g_k-meeting, $k \leq n$, i.e, e is a meeting of all the users that have a 1 in their k^{th} bit.

Suppose there do not exist meetings f and f' such that they are complementary and $e \preceq^* f$ and $e \preceq^* f'$. Then, there can be at most $n-1$ meetings above e in V'_{uv} and each of them is of the form $g_i,\ i \neq k, k+n$. Let T be the set of these meetings. Now consider the user $w \in U$ which has a bit representation as follows :

- A 1 in the k^{th} bit.
- For $i \leq n,\ i \neq k$, a 0 in the i^{th} bit if T contains a g_i-meeting, 1 if T contains a g_{i+n}-meeting and either 0 or 1 in the other bits not covered by meetings in T.

It follows that for the user w, e is the latest meeting that u or v know about w, which is a contradiction.

A similar argument holds if e is a g_k-meeting, $k > n$.

So, in order to check if a meeting is redundant, it is sufficient to check if this meeting is below two complementary meetings. This is much more efficient than exhaustively checking if e is not the latest meeting for every user in e.

Auxiliary Information. Let σ be a communication sequence and $u \in U$. The *auxiliary information* of u after σ, $auxiliary_u(\sigma)$, is a directed graph $\mathcal{G}^s_u(\sigma) = (V^s_u(\sigma), E^s_u(\sigma))$ where

$$V^s_u(\sigma) = \{e \mid \exists u', u'' \in U\ s.t\ e\ =\ last_{u,u',u''}(\sigma)\}$$

$E^s_u(\sigma) = \{(e, f) \mid e \preceq^* f\ and\ \exists\ u', u'', u''' \in U\ s.t\ e = last_{u,u',u''}(\sigma)\ and\ f = last_{u,u',u'''}(\sigma)\}$

That is, the auxiliary information maintains, for each of the meetings e in the latest information, the set of meetings which gives the latest information that users in e know about every other user in the system. This is maintained as a directed graph which has at most $2n$ maximal components, each of which has at most $2n$ vertices.

Updating Auxiliary Information. Auxiliary information can be updated as and when the latest information is being updated. Recall that when two users u and v meet, there are the two possibilities corresponding to which the auxiliary information is updated as given below :

Let σ be a communication sequence and let $\mathcal{G}_u(\sigma')$ and $\mathcal{G}_v(\sigma')$ denote the initial latest information graphs of u and v respectively, i.e, before the occurrence of the current meeting g. Let $\sigma = \sigma' g$. We denote by $\mathcal{G}_{uv}(\sigma)$, the common updated latest information graph of u and v after the meeting g. Let $\mathcal{G}^s_u(\sigma')$ be the auxiliary information of u and $\mathcal{G}^s_v(\sigma')$ the auxiliary information of v. We denote by $\mathcal{G}^s_{uv}(\sigma)$, the common updated auxiliary graph of u and v. Now, when the u and v meet, the possibilities are :

- **Case 1 :** One of $last_u(\sigma')$ or $last_v(\sigma')$ belongs to the set S.
  Suppose $last_u(\sigma') \in$ S.
  Let $\mathcal{G}'$ be the union of all those *maximal components* of $\mathcal{G}^s_q(\sigma')$ such that the maximal meeting in that component belongs to R_{uv}. Then, we get the common updated auxiliary graph as follows:

  $$\mathcal{G}^s_{uv}(\sigma) \;=\; (\; \mathcal{G}^s_q(\sigma') \;-\; \mathcal{G}' \;) \;\cup\; \mathcal{G}_{uv}(\sigma).$$

  A symmetric argument holds for the other possibility.
- **Case 2 :** Neither $last_u(\sigma')$ nor $last_v(\sigma')$ belong to the set S.
  Then, $\mathcal{G}^s_{uv}(\sigma) \;=\; \mathcal{G}_u(\sigma') \;\cup\; \mathcal{G}_v(\sigma') \;\cup\; (\mathcal{G} = (V, E))$
  where $V = \{last_u(\sigma'), last_v(\sigma'), l\}$ and
  $E = \{(last_v(\sigma'), l), (last_v(\sigma'), l)\}$. Remove redundant meetings from $\mathcal{G}^s_{uv}(\sigma)$, if any.

In the first case, the above procedure works since if $last_u(\sigma')$ belongs to S, then the auxiliary information of v is at least as good as the auxiliary information of u.

In the second case, we have observed that the common updated latest information graph has just three vertices : $last_u(\sigma')$, $last_v(\sigma')$ and l. Since $last_u(\sigma')$ and $last_v(\sigma')$ are complementary meetings, for all the users u' that participated in g,

$$\{last_{v,u',u''}(\sigma) \mid r \;= u \textit{ or } v \textit{ and } u'' \;\in U\} \subseteq \{last_u(\sigma'), last_v(\sigma')\}.$$

For any user u' that did not participate in g (there are 2^{n-1} such users), since the latest that u or v know about u' is either $last_u(\sigma')$ or $last_v(\sigma')$, we have,

$$\{last_{r,u',u''}(\sigma) \mid r \;= u \textit{ or } v \textit{ and } u'' \;\in U\} \subseteq \{e \mid e \in V_u \textit{ or } e \in V_v\}.$$

So, we have shown that the update procedure for auxiliary information outlined earlier works.

The Amount of Local Information.

Theorem 2. *Each user $u \in \mathcal{H}_n$ needs to maintain at most $O(n^3)$ bits of information.*

Proof. The local information for user u consists of its latest and auxiliary information. We estimate the number of bits required to store this.
The *latest information* of each user is stored in two parts:

- An $2n \times 1$ array which stores the labels of at most $2n$ meeting containing latest information. Since we have shown, for the general case, that it suffices to have N^3 labels in the system, each label requires $O(logN) \;=\; O(n)$ bits. This component of the latest information therefore requires $O(n^2)$ bits.
- A $2n \times 2n$ adjacency matrix which stores the edges of the latest information graph. This requires $O(n^2)$ bits.

Therefore, storing the latest information requires $O(n^2)$ bits in all.

The *auxiliary information* of each user is stored in two parts:

– An $2n \times 2n$ array which stores the labels of at most $4n^2$ auxiliary meetings. This component of the auxiliary information therefore requires $O(n^3)$ bits.
– $2n$ adjacency matrices of dimension $2n \times 2n$ which store the edges of the auxiliary graph. This clearly requires $O(n^3)$ bits.

Therefore, the auxiliary information requires $O(n^3)$ bits in all. Hence, local information of each users requires $O(n^3)$ bits.

8 Conclusions

In this paper, we have studied a natural problem related to information dissemination in social networks in which users meet in groups over time and are interested in learning the latest news known to the users present in the meeting about all the other users in the network.

Identifying the user in the meeting with the latest information becomes tricky if the users are only allowed to label each meeting with a string from a finite, bounded set of labels. The main difficulty is to ensure that it is still possible to compare and choose among two labels while the labels are being recycled over time.

We have shown two interesting results:

– For a network of N users with no restriction on the network topology, we have shown that it is sufficient for each user to maintain $O(N^2 \log N)$ bits of information improving the previous bound of $O(N^3 \log N)$[9].
– For the special case of mesh topology, the n-dimensional hypercube, we have shown that is sufficient to maintain $O(n^3)$ bits of information for $N = 2^n$ users. This is a significant improvement over the general case.

For future research, we would like to try and extend our results to other topologies of interest. In particular, we would like to understand the structure and connectivity proerties of communities in social networks to see what further assumptions can be made in the case of a scale-free network.

Acknowledgements. We would like to thank Madhavan Mukund and Milind Sohoni for introducing us to this problem and for many useful discussions.

References

1. Cori, R., Metivier, Y. : Asynchronous Mappings and Asynchronous Cellular Automata. *Information and Computation* **106** (1993) 159–202
2. Cori, R., Sopena, E. : Some Combinatorial Aspects of Time-stamp Systems. European Journal of Combinatorics **14** (1993) 95–102

3. Dolev, D., Shavit, N. : Bounded Concurrent Time-stamps are Constructible. In Proc. of 21th Annual ACM Symposium on Theory of Computing (1989) 454–466
4. Dwork, C., Waarts, O. : Simple and Efficient Bounded Concurrent Timestamping or Bounded Concurrent Time-stamps are Comprehensible. In Proc. of 24th Annual ACM Symposium on Theory of Computing (1992) 655–666
5. Fan, W., Geerts, F., Wijsen, J. : Determining the Currency of Data. *In Proc. of the 30th ACM Symposium on Principles of Database Systems* (2011) 71–82
6. Hedetniemi, S. M., Hedetniemi, S. T., Liestman, A. L. : A Survey of Gossiping and Broadcasting in Communication Networks. *Networks* **18** (1988) 319–349
7. Israeli, A., Li, M. : Bounded time-stamps. *In Proc. of 28th IEEE Conference on Foundations of Computer Science* (1987) 371–382
8. Lind, P. G., da Silva, L. R., Andrade Jr, J. S., Herrmann, H. J. : Spreading Gossip in Social Networks. *Phys. Rev. E* **76:3** (2007)
9. Mukund, M., Sohoni, M. : Keeping Track of the Latest Gossip in a Distributed System. *Distributed Computing* **10:3** (1997) 137–148