Effect of Similarity Distribution on the Convergence of Decentralized Similarity Overlays

Irum F. Bukhari **Aaron Harwood** **Shanika Karunasekera**

Department of Computing and Information Systems
University of Melbourne,
PO Box 3010, Victoria, Australia
Email: ibukhari@student.unimelb.edu.au
aharwood@unimelb.edu.au
karus@unimelb.edu.au

Abstract

Decentralized systems are used by a significant number of Internet users due to their specific features such as autonomy and content privacy. With the evolution of big data, the problem of filtering information based on user interests has become prevalent. Decentralized strategies for information filtering, such as similarity clustering, seem appealing due to its simplicity over model-based approaches (Matrix factorization). However, decentralized similarity clustering is not as well studied as centralized ones. In this work, we studied gossip-based similarity clustering with different real world similarity distributions. We analyzed convergence time and found the trade-off between convergence time and bandwidth utilization based on optimal protocol parameters (message size, neighbor-list size). The optimal settings of the protocol parameters not only give the minimum convergence time (approximately), from a worst case random structure, but also avoid wastage in bandwidth.

Keywords: Peer-to-Peer Networks, Gossip Protocols, Vicinity*, Semantic Overlays, Similarity Distribution, Convergence, Information Filtering*

1 Introduction

There are a number of existing, real world, decentralized applications ranging from file sharing to other exciting and useful applications such as search engines, social networks, and personalized recommender systems. Some of the examples are BitTorrent (Cohen 2008), Tribler (Pouwelse et al. 2008), YaCy (Ltjohann et al. 2011, Community 2012), Gossple (Bertier et al. 2010) and WhatsUp (Boutet et al. 2013). These applications offer users provisions such as autonomy and privacy. Users experience full control of their content and resources. Moreover, unlike centralized systems, these systems are economical due to elimination of power consumption costs of running data centers. The number of BitTorrent users has increased to 150 million as of January 2012, reported by BitTorrent Inc. These applications are responsible for 3.35% of all worldwide bandwidth, more than half of the 6% of total bandwidth dedicated to file sharing until February 2013. The amount of data produced in real time, driven by these applications, have reached between 20% and 40% of global Internet traffic. These facts depict the significant research into decentralized systems.

Many of these decentralized applications require complex big data processing such as anomaly detection, information filtering, document clustering and content recommendation. Similarity-based Clustering (SBC), either of users or items, is a primary technique used by all of these systems. Many decentralized works make use of decentralized overlays for SBC because the overlay directly supports communication between the nodes in the system; i.e. the clusters formed are essentially part of the overlay. Building and using clusters is less complex and more direct, than other model-based approaches for clustering.

Decentralized SBC is not as well understood as it's centralized counterpart. Methods to analyze the correctness of the distributed approach are not clearly defined and its harder to understand the performance of the overall distributed system. Communication complexity is a central factor to the performance of these systems. How many messages need to be communicated for a reasonable performance? How big must be the messages? How quickly the system gets ready to be used? Running at Internet scale (10^8 users) makes it even harder to understand overall performance. We, therefore focus on the fundamental aspect of *performance versus resource consumption trade-off* in a decentralized similarity clustering system.

Of the decentralized systems mentioned earlier, many of them use gossiping as their basis for building an overlay and for processing data (Magureanu et al. 2012, Kermarrec & Taíani 2012, Ormándi et al. 2010, Bertier et al. 2010). Gossiping (Voulgaris & Steen 2013, Voulgaris et al. 2005, Jelasity & Babaoglu 2005, Jelasity et al. 2007) has features including robustness, scalability, simplicity and is usually modeled using epidemic models. Gossip-

based overlays should handle hundreds of millions of users in the system. Due to these advantages over structured approaches, many researchers are moving towards gossip-based systems.

Vicinity (Voulgaris & Steen 2013) and T-Man (Jelasity et al. 2009) are well known gossiping protocols that enable similarity clustering. These protocols have been well studied and have comparable properties. They are based on random sampling which is a cost-optimal approach. It requires time to converge to an accurate similarity clustering, when these systems are implemented with gossiping. Convergence to an accurate similarity clustering is important because the accuracy of the clustering affects the overall accuracy of any outcome based upon it. Furthermore, as data changes by node churns or changing user profiles, similarity clustering is a continuous self-healing operation that needs to maintain accuracy. Therefore, we study the *quality* of the convergence, the time and the bandwidth required for convergence.

A trade-off between convergence time and bandwidth is very useful to be known before building an application, e.g. recommendation systems or clustering documents, on top of similarity clustering. This trade-off provides the designers of such systems knowledge about bandwidth requirements for a favorable convergence speed.

We studied Vicinity as a similarity clustering protocol for convergence analysis and performed experiments with real-world distributions, i.e. user graphs where similarity is based on data extracted from real applications. These include Movielens, Yahoo and Epinion. These distributions make our results indicative of behavior that we expect to see in a real implementation at large scale.

1.1 Our Contribution

Our work has two core contributions:

- Analyzing the convergence of SBC overlays when exposed to real time similarity distributions. The existing work in this domain considers synthetically generated network graphs with regular or symmetric properties such as Mesh, Torus, Binary Tree or sorted Ring to make them understandable. However, most of the real world datasets have irregular and asymmetric structure and properties. Most of these have power law distribution such as Movielens, Epinion, Book Crossings (Ormándi et al. 2010) and the literature does not have the detailed study of such similarity distributions. Therefore, our work investigates Movielens dataset primarily with two other real datasets namely Yahoo and Epinion.
- Finding the trade-off between convergence time and bandwidth to decide optimal settings that lead to fast convergence and affordable bandwidth from SBC system's perspective.

1.2 Organization

The paper is organized as follows. Section 2 explains similarity distributions used for analysis of the protocol. Section 3 explains an SBC system model by describing the protocol, the system itself and the criteria we have used for evaluation of results. Section 4 evaluates the results. Section 5 explains findings for future work. Section 6 describes the review of literature most relevant to our work and Section 7 makes some concluding remarks.

2 Characteristics of Graphs Based-on Similarity Distributions

In an SBC system, ratings given by users of the system are used to calculate similarity between user profiles. The value depicting similarity of interests represents the distance between users. On the basis of this distance, users are grouped together. This grouping leads to different types of in-degree distributions of the network, where some nodes may have very high in-degree or some nodes very low in-degree or all nodes of the network may have uniform in-degree. For experiments, real traces from the benchmark datasets, Movielens, Epinion and Yahoo are used to observe the convergence behavior of the protocol. A synthetically generated Mesh network is used in initial experiments for protocol analysis due to its regular structure, unlike real distributions, and as a baseline comparison to the real datasets.

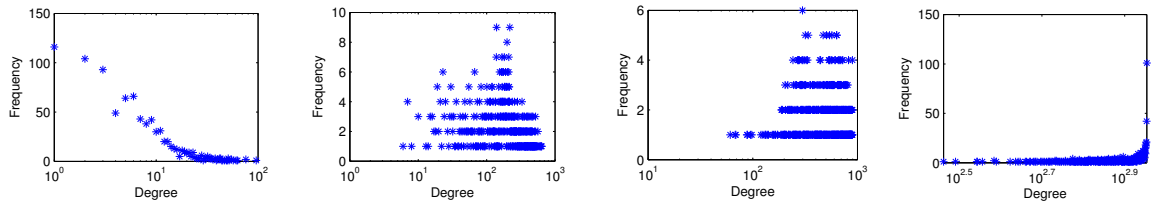
2.1 Datasets

To analyze real world similarity distributions, three benchmark datasets are used including Movielens, Yahoo and Epinion. The Movielens (Labs 2011) dataset, contains one million ratings on 1682 movies by 943 users. The Web-scope dataset from Yahoo labs (Labs 2012) contains approximately 3 million ratings about 1000 songs from exactly 15,400 users. The Epinion (Massa 2003) traces contain 664,824 ratings about 139,738 products by 49,290 users.

For analysis of convergence between distributions, 900 users are randomly selected from respective dataset. Based on user profiles, the top k optimal users for each user are calculated using Cosine similarity. These tables for each value of k are then used to calculate optimal nodes in each nodes view. The similarity distances range from 0 (completely non-similar) to 1 (the most similar). Table 1 shows the clustering co-efficients for the three real networks comprised of 900 nodes, with three different values of k. Movielens has high clustering co-efficients whereas Epinion has the least

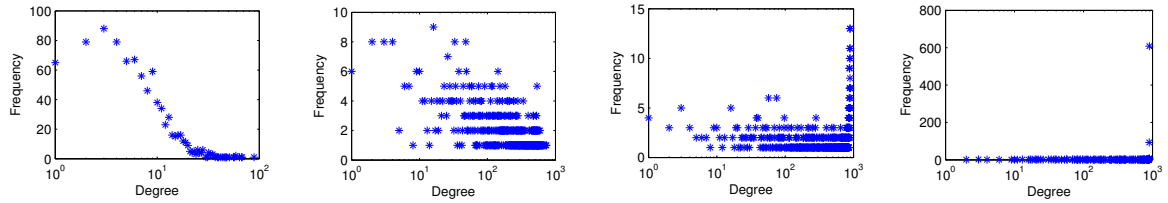
Top k	Movielens	Yahoo	Epinion
10	3.13×10^{-4}	2.13×10^{-4}	1.86×10^{-4}
20	3.79×10^{-4}	2.53×10^{-4}	2.11×10^{-4}
40	4.64×10^{-4}	3.03×10^{-4}	2.49×10^{-4}

Table 1: Clustering Co-efficient of Real Datasets



(a) $k = 10$ (b) $k = 200$ (c) $k = 500$ (d) $k = 800$

Figure 1: Similarity distribution of Movielens with 900 nodes.



(a) $k = 10$ (b) $k = 200$ (c) $k = 500$ (d) $k = 800$

Figure 2: Similarity distribution of Epinion with 900 nodes.

among all which depicts differences in interests of users.

Preliminary experiments are performed on a 2-D Mesh due to its simple and understandable structure. To generate a mesh network, the distance metric used is number of hops. Each node is assigned a distance in terms of number of hops to every other node in the network. The resulting network is a single connected component. All nodes collectively build a mesh topology with in-degree = out-degree which is 4 for all nodes except edge nodes with degree 3 and corner nodes with degree 2.

2.2 Degree Distribution

Determining the degree distribution of a particular dataset based on similarities between nodes, depends upon the value of k. Figure 1 and Figure 2 show the similarity distribution of 900 nodes in the Movielens and Epinion datasets respectively, with four different values of k. With $k = 10$, both datasets have Power-law distribution which no longer remains Power-law as k increases to half of the network size. Differences in degree distribution leads to differences in convergence time as explained in Section 4.3.

3 Decentralized SBC System Model

For the purpose of SBC approach, we consider a system containing a set of users $U = \{1, 2, 3, \ldots, n\}$, where each node of the network represents a user of the system, a set of items $I = \{1, 2, 3, \ldots, k\}$ and a set of ratings about items $R = r_{u,i}$ where $i \in I$ and $u \in U$. Each user of the system has a profile that contains information about the items held and associated ratings. The system shall provide a user, access to the profile information of the top k similar users. For the purpose, the system requires a similarity metric which is Cosine for Movielens dataset. The Cosine similarity between user a and b is calculated using the following equation:

$$Similarity(a, b) = \frac{\sum_{l=1}^{n} r_{a,l} r_{b,l}}{\sqrt{\sum_{l=1}^{n} r_{a,l}^2 \sum_{l=1}^{n} r_{b,l}^2}},$$

where $a, b \in U, l \in I$.

To realize the system, Vicinity is proposed as the overlay protocol, as argued in Section 1.

3.1 Vicinity

Vicinity is a gossip-based protocol which is used to cluster peers that are semantically close. Vicinity is a two layer protocol. The upper layer consists of Vicinity (proper) and the lower layer consists of the Cyclon protocol. Each node maintains a list of neighbors which is a subset from the whole network, called a "view". All nodes converge to the semantically closest neighbors in their views, once the protocol is converged. These optimal views can be used for generating recommendations. To restrain the protocol from saturation, i.e. the point where long distant nodes are no longer accessible thereby preventing some of the views to reach to their optimal, fresh random samples from the network are acquired with the peer sampling service provided by the Cyclon protocol. Cyclon continuously feeds Vicinity with random peer samples from the network, which guarantees convergence given a sufficiently long period of time.

Each node in the network is required to run two protocol instances, one for Vicinity and one for Cyclon. Therefore, every node maintains two

views, one for Cyclon and the other for Vicinity. As Cyclon is a peer sampling service, its view contains fresh random samples of the network. Vicinity, however, has only those nodes in the view which are semantically close. Vicinity uses a selection function, which decides what peers to keep in the Vicinity view on the basis of similarity metric used.

Each node in the network has an associated *descriptor* that contains a *Node Identifier*, an *Age* parameter, and the *Profile* of the node. The *Age* parameter gives some indication of when the node was last contacted. Every node initiates gossip exactly once in each round. During gossiping, nodes exchange neighbors from their views. The number of neighbors exchanged in each gossip, is called *gossip length*.

Vicinity is described in Algorithm 1 and 2. Two threads are required to execute simultaneously, by a node, active and passive. Every node executes the active thread exactly once every T time units to initiate gossip. Each node also needs to execute a passive thread in response to a gossip request. The passive thread is not always running. Only the nodes selected for gossiping are required to run the thread at the time of a gossip request being received. There are three methods used by nodes in the network. To initiate gossip, the node executes `SelectToGossip()`, which selects the oldest node in the view. Functions `SelectToSend()` and `SelectToKeep()` select optimal nodes from all buffers by applying a selection function, and exchange a number of neighbors equal to gossip length. A more detailed explanation of the protocol is described in (Voulgaris et al. 2007).

while *true* **do**
  wait(T time units);
  increase age of all descriptors in the view;
  $Q \leftarrow$ `SelectToGossip()`;
  remove Q from the view;
  bufSend $\leftarrow$ `SelectToSend()`;
  send bufSend to Q;
  receive bufRcv from Q;
  view $\leftarrow$ `SelectToKeep()`;
end

Algorithm 1: Active Thread

receive bufRcv from Q;
bufSend $\leftarrow$ `SelectToSend()`;
send bufSend to Q;
view $\leftarrow$ `SelectToKeep()`;

Algorithm 2: Passive Thread

3.2 Building an SBC System

An SBC system, for instance, can be used to predict rating about an item on the basis of the ratings from the most similar users if the application is a recommendation system. Vicinity converges to the most similar neighbors for each user which makes it suitable for building an SBC system.

To bootstrap Cyclon and Vicinity protocols (for the purpose of simulation, but not necessarily in a real deployment), the two views associated with each node are filled with randomly selected neighbors. Each node of the network is then associated with the list of items and associated ratings from the dataset. The selection function, Cosine, is applied to calculate similarity between nodes. A similarity value of 1 between two nodes shows that the nodes are identical. The similarity value decreases to 0 with a decrease in commonality between profiles.

Due to the use of real datasets instead of a dynamic data source, user profiles remain static throughout the presence of the node in the network. After implementation of Vicinity with different distributions, the differences in convergence behavior of the protocol are observed. We leave the study of changing profile information to future work.

3.3 Convergence Criteria

The protocol is considered to be fully converged when all nodes of the network have optimal nodes in their views. Formally, Let $\Gamma_d(u)$ be the set of nodes at distance d, for a given node u. Consider a view V, of size v. Find the smallest d_0 such that $\mid \Gamma_{d_0}(u) \mid \geq v$. Then the node u is converged if and only if:

$$V \subseteq \Gamma_{d_0}(u), \text{and}$$
$$V \cap \Gamma_d(u) = \Gamma_d(u) \text{ for all } d < d_0. \qquad (1)$$

From Eq. 1, if many equi-distant nodes are found for a given node, then any of those found in the view is considered optimal. This is illustrated in Figure 3 where optimal nodes of node m are shown in a 2-D Mesh.

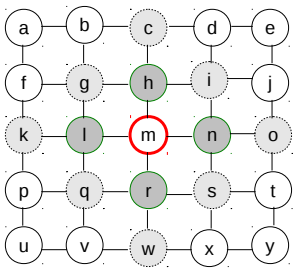


Figure 3: Optimal links for node m in a 2-D Mesh, with view sizes 4 and 6.

With view size 4, optimal neighbors for node m are $\{h, l, n, r\}$ whereas, with view size 6, optimal neighbors for node m are 4 nodes at the distance of 1 hop; $\{h, l, n, r\}$ and any 2 nodes among 8 nodes at the distance of 2 hops; $\{g, i, q, s, c, k, w, o\}$, any 2 of these can fill empty view slots as optimal neighbors. We define percentage convergence at a given cycle as the extent that a node has converged at that cycle. Formally, Let $V^*(u)$ be any set V that satisfies Eq. 1 for node u. Then the percentage convergence $\xi_\tau(u)$, for node u at cycle τ is:

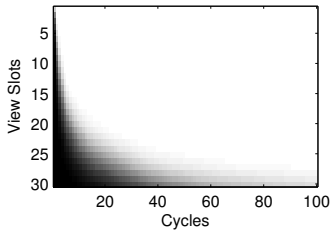


Figure 4: AVQ

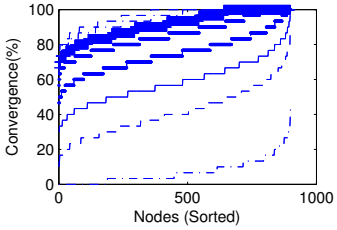


Figure 5: PC

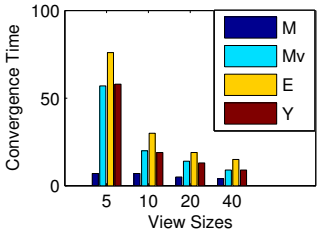


Figure 6: 80% nodes converged

$$\xi_\tau(u) = max_{V^*(u)} \left\{ \frac{\mid V_\tau(u) \cap V^*(u) \mid}{\mid V^*(u) \mid} \right\}, \quad (2)$$

i.e. the percentage convergence is the maximum over all the possible converged sets.

4 Simulations and Experiments

All experiments were performed in PeerNet, which is an extended version of PeerSim (Montresor & Jelasity 2009). The simulations were carried out using both modes of the simulator, cycle-driven and event-driven. The cycle-driven mode executes each experiment in cycles where one cycle is completed when all nodes initiate gossip exactly once. The event-driven mode is activated when events processing or exchange of messages between nodes is required.

4.1 Environment settings

The parameters that affect convergence are shown in Table 2.

4.1.1 View Size

Vicinity uses Cyclon for randomly linking to long distant nodes, which makes it capable of converging quickly towards the optimal links. Due to the random selection, the convergence time varies for each run. Therefore, we run the experiment multiple times to get an acceptable standard error in mean (SEM). For SBC systems, it is crucial to know how long it can take to converge. The longer the convergence time, the longer it takes to generate accurate results based upon this. For each distribution, this time may vary. Keeping view size small makes the convergence slower but helps maintaining fresh links. When using large view sizes, it not only takes longer to contact all the links in the list, but also increases the chances of invalid or stale links in the view. It seems more appealing to use larger view sizes when more choices are required. On the other hand, smaller view sizes give most recent information from a small number of optimal nodes with the compromise on the number of choices of users. For simulations, view sizes are kept equal for both protocols ranging from 4 to 45, in all experiments.

Parameter	Symbol	Values
View Size	$V_c = V_{vic}$	5,10,20,40
Gossip Length	$g_c = g_{vic}$	all values of $g \leq V$
Network Size	N	100 to 6000 Nodes

Table 2: Protocol Parameters

4.1.2 Gossip Length

Using a large gossip length, reduces the convergence time significantly but costs more bandwidth, which is a critical parameter. A trade-off between bandwidth and convergence time is required to determine the optimal gossip length for SBC systems. Gossip length is kept equal for Cyclon and Vicinity, in all experiments.

4.1.3 Network Size

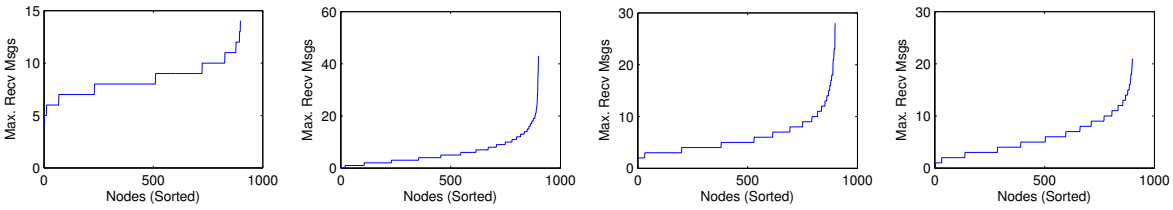
For SBC systems, larger networks can be more favorable because they increase the chances of finding similar nodes. Initial experiments are conducted with networks containing 900 nodes. The reason of keeping the network size this small is to simplify the comparison between different distributions. However, network sizes varies between 100 and 6000 for experiments concerning the effect of network size on convergence time. All these networks are extracted from Movielens containing 6040 users.

4.2 Protocol Analysis

Each experiment in each simulation was run for 100 trials for each distribution due to the randomness of Cyclon and the initial conditions. This leads to different convergence times which can be observed clearly by ripples after $g = v/4$ for each view size in Figure 12 in Section 4.3.

4.2.1 Average View Quality

An interesting behavior of Vicinity is that it fills views of each node with the most similar neighbors first. This behavior is observed using average view quality which is shown in Figure 4 using view slots. By Average View Quality (AVQ), we mean how similar a node is to its neighbors in its view, on average. The average similarity taken over all nodes of the network is represented by shades of gray. Black shows the presence of non-optimal links in the view slots, that are replaced by more similar links turning to white as the number of cycles increases. Black is more dominant



(a) Mesh (b) Movielens (c) Epinion (d) Yahoo

Figure 7: Message Load on each node during 1000 cycles with v=20, g=5, N=900

on the tail of the view, which represents existence of least-optimal links. These links are the least similar, as compared to the others in the view, but more similar than other nodes in the network. The reason for the long tail is that as the protocol converges, most of the nodes acquire their optimal links, hence decreasing the chances for some other nodes still foraging for their optimal links in the neighbors views. Hence, the random peer sampling service must be relied on to obtain those links, which can take some time.

4.2.2 Percentage Convergence

In Figure 5, the Percentage Convergence (PC) for each node in a 30×30 Mesh, is shown at cycles between 1 to 50, with view size 30 and gossip length 4. Each line represents a separate cycle in the order $\{1, 3, 5, 10, 20, 30, 40, 50\}$, moving upwards. Nodes are sorted according to the percentage of convergence to increase readability. At each cycle, some nodes are converging faster than others, due to the path taken to reach to the destination nodes. At cycle 50, there are still some nodes kept waiting for their optimal nodes causing the long tail shown in the Figure 4.

4.3 Convergence Analysis

4.3.1 Convergence Time

Figure 6 demonstrates the convergence time for each distribution when 80% of nodes are fully converged. Here, each distribution is represented by its initial letter(s). It can be observed that the optimal convergence time for each distribution is different for view sizes 5, 10, 20 and 40 when the gossip length is equal to view size. All real world distributions took longer to converge than Mesh due to the distribution itself where the in-degree distribution forms an irregular graph unlike Mesh. Therefore, some nodes acquire link to their destination nodes in initial cycles whereas some nodes keep foraging for their optimal nodes for many cycles. Depending on the clustering co-efficients as described in Table 1, each real distribution converges accordingly. The higher the distribution is clustered, the faster it reaches to convergence. Figure 7 shows the maximum load on each node (sorted) during 1000 cycles, with different distributions.

4.3.2 Network Size

Experiments with network sizes between 100 and 6000 nodes from Movielens dataset, suggest the increase in convergence time when the network size is increased to one million. Above 6000, the results are based on extrapolation for the convergence time when nodes are $10^4, 10^5$ and 10^6. Figure 8 shows the increase in convergence time to 7000 cycles (approximately) when the network size grows to one million nodes.

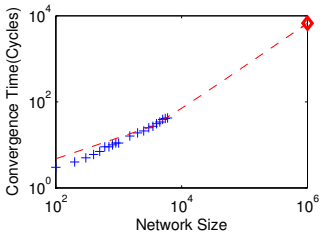


Figure 8: Convergence time for Movielens reaches 7,000 cycles for a 1 million nodes network.

4.3.3 Message Load

When the in-degree distribution of the network is Power-law, which is mostly the case with real datasets, the number of messages received at maximum and on average, are not the same for all nodes present in the network. This leads to higher bandwidth requirement for some nodes. Figure 7 shows the maximum number of received messages by each node during 1000 cycles.

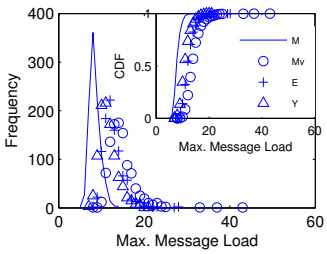
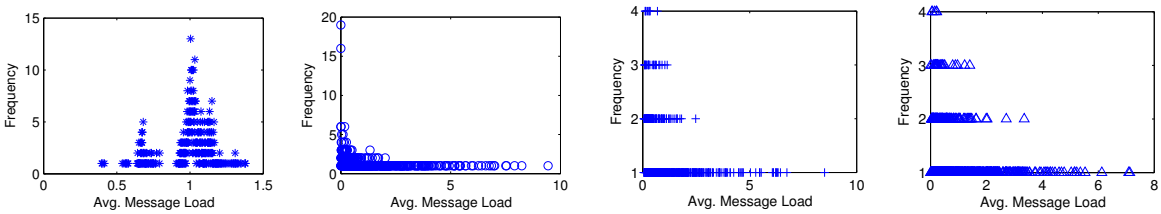


Figure 9: Maximum no. of messages received by a node in 1000 cycles.

Figure 9 and 10 show the maximum and average message load experienced by a node for Movielens and other distributions. Mesh has the least



(a) Mesh (b) Movielens (c) Epinion (d) Yahoo

Figure 10: Average no. of messages received by a node in 1000 cycles.

average message load approximately equal to 1. Whereas, Movielens and other real distributions have average message load less than 5. However, the maximum number of messages received by any node in the network reaches 40 for Movielens where 90% nodes have the maximum message load less than 20 as shown in Figure 9.

It can be seen that only one node received a maximum of 43 messages in a cycle. For nodes experiencing greater than the average message load, the bandwidth can be a critical parameter. Figure 11 shows the maximum and average number of messages received by each individual node, averaged over 1000 cycles. On average, each node receives one message where the actual number of maximum messages can increase to above 40. When the network grows to 100,000 nodes, the maximum messages received by a node reached greater than 400 per cycle, increasing bandwidth requirement up to 1 gigabyte per second.

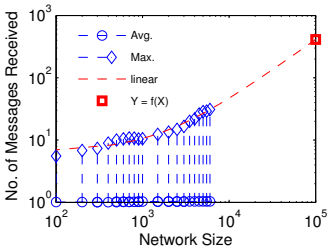


Figure 11: Extrapolation for $N > 6000$ shows the maximum number of messages reach above 400.

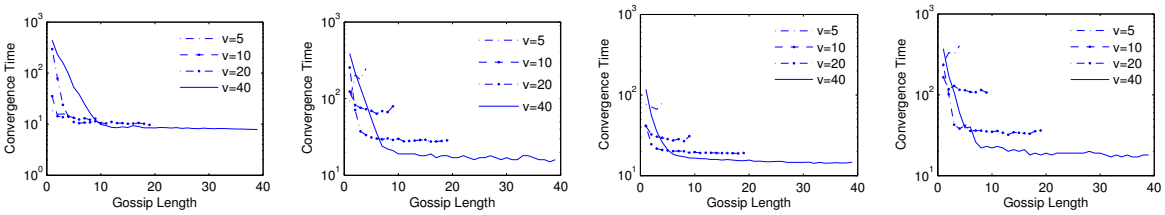
4.3.4 Bandwidth Utilization

The communication overhead in gossip-based SBC overlays is due to the self-healing nature of gossip protocols. The nodes need to exchange messages frequently that requires bandwidth. The bandwidth cost depends directly on factors such as message load, network size and convergence time as explained in section 4.3.

To understand the bandwidth utilization in a gossip-based SBC system, consider a movie recommendation system where a given nodes' table is comprised of v other nodes, each with its associated profile. Let s be the size of a node profile, where in this case the profile could be the m latest movies watched containing the user assigned ratings and identifying information including title, year, producer, director. For a subset of size g, the total bytes required to send/receive one message is $g\,s$, or $g\,s$ bytes per second for a cycle time $T = 1$ second.

If we consider exchanges between random nodes then the total number of messages exchanged in each cycle is on average 4 per protocol, exactly 2 as initiator and on average 2 as responder. Using Vicinity costs 4 messages per cycle. For simple comparison in our work, we assume some reasonable values of $s = 100$ bytes for each profile item and $m = 100$ items in the profile, and a subset size of $g = 4$, the required bandwidth is $4 \times 4 \times 100 \times 100 = 160k$ bytes per cycle. If the convergence time is 1000 cycles for a given distribution and given network size (let's say 1000 nodes), and $T = 1$ hour (very slow cycles), the bandwidth is (a tiny) $160k/3600 = 44$ bytes per second but it takes 1000 hours to converge and consumes a total of 1600 MB per node. For $T = 2$ seconds (unrealistically fast, since Internet delays start to dominate), the bandwidth increases to $160k/2 = 80k$ bytes per second (quite appreciable), taking 2000 seconds to converge with the same amount of total information exchanged. In this paper, we infer this trade-off for large scale networks from simulation data as shown in Section 4.3.2. Smaller messages cause less communication overhead and hence desirable. To find the message size which is suitable for optimal convergence, we consider gossip length g with three different cases: $g > v$, $g = v$, and $g < v$. Keeping $g > v$ gives the same convergence speed as $g = v$ because a node only knows v neighbors. Therefore, $g = v$ and $g < v$ were considered for experiments with view sizes $\{5, 10, 20, 40\}$. It is observed that with $g = v$, the optimal convergence time is achieved because every node is exchanging whatever it has in the view. To analyze the convergence speed for $g < v$, all possible values of g less than v are considered beginning from $g = 2$ to v. A value of $g = 1$ leads to very long convergence time.

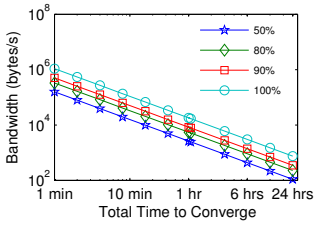
Figure 12 shows the convergence time in cycles against gossip length for four different distributions when all nodes' views are 80% converged. It is observed that using g less than $v/4$ reduces the convergence time, significantly. However, with gossip length larger than $v/4$, only a small decrease in number of cycles (approximately 2 to 3 cycles) is observed. This is due to the reason that each node exchanges a fixed size portion (g) of its view with the other node. The exchanged portion consists of the most similar nodes selected



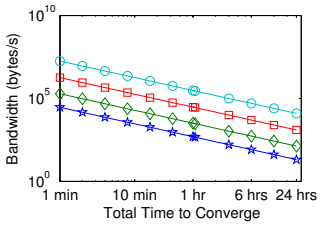
(a) 2-D Mesh (b) Movielens (c) Epinion (d) Yahoo

Figure 12: Convergence time with $g \leq v$, for 80% convergence

from the other nodes perspective. Hypothetically, a larger portion must lead to faster convergence by providing with more neighbor choices. But actually the larger portion provides with only a few more *less similar* neighbor choices having decreased chances to be selected as optimal nodes. This is evident by the difference of 2 to 3 cycles between $g = v/4$ and $g = v$. Therefore, using $g = v$ to get approximately the same convergence speed, at the cost of more bandwidth, is wasting resources. Figure 12(a) shows the convergence



(a) Convergence to a percentage.



(b) Different network sizes $(10^3, 10^4, 10^5, 10^6)$, moving upwards.

Figure 13: Bandwidth-Convergence Trade-off

time for Mesh distribution when all nodes are 80% converged. The same can be observed for all other distributions, i.e. Epinion and Yahoo as shown in Figure 12. The results shown in Figure 12(c) are when 80% nodes are 80% converged, to avoid long simulation runs. Epinion holds the smallest clustering co-efficient among all the three real distributions. This causes the least clustered nodes to look for their optimal nodes for thousands of cycles.

Figure 13(a) demonstrates the bandwidth requirement as the total convergence time varies between one minute to one day, for 50%, 80%, 90% and 100% convergence with Movielens dataset. For all other experiments on network size, only 80% convergence is used for calculating convergence time.

Figure 13(b) plots the bandwidth against the total time required to converge for four different network sizes. The convergence time for network sizes 10^3, 10^4,10^5, and 10^6 are 11, 72, 675, and 6750 cycles, respectively. It can be observed that with the increase in the size of a network, the required bandwidth increases in the same time. The reason is that the larger networks take more cycles to converge. This shows that decreasing number of cycles required to converge, contributes towards decreased bandwidth. Moreover, the bandwidth utilization is also dependent on the time that the application can allow to converge the network. If this time is in seconds rather than hours, more bandwidth will be required to reach to the convergence state. These experiments are performed using the optimal settings of view size and gossip length. Without these settings, the required bandwidth will increase as gossip length increases.

5 Long-tail Behavior

Investigation about convergence of SBC overlays leads to a few questions about the scenario where some nodes keep foraging for very low in-degree nodes but cannot reach them through Vicinity gossip rounds. In that case, the only solution that the protocol offers is the peer sampling service provided by Cyclon. Nodes rely on this service's random input to reach those low in-degree nodes leading to very long convergence time. This situation is depicted in Figure 14 where nodes a and b are low in-degree (equal to 1), fully converged nodes. Node z is foraging for a and b as it's optimal nodes but cannot reach them. No other nodes have links to a or b except t and v, respectively. Nodes y

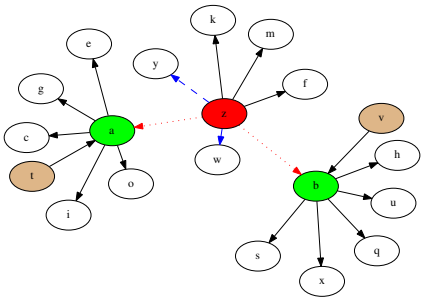


Figure 14: Problem reaching low in-degree nodes.

and w are filling the neighbor list as replacements.

Vicinity gossiping does not serve any benefit except increasing the convergence time. These low in-degree nodes get isolated and need a way to be reached by other nodes. This problem will be further investigated in the future work. The worst case for SBC overlays is when nodes in the overlay have nothing common in their profiles. This situation is common when bootstrapping the overlay.

6 Related Work

6.1 Similarity Clustering

The preliminary work in semantic clustering includes (Voulgaris et al. 2004) and (Handurukande et al. 2004). The idea was to improve file sharing and content searching by developing relationships between users. These semantic relationships between users are used to search for files or contents with significant performance. The authors in (Mekouar et al. 2012) analyze different similarity metrics to generate accurate recommendations using a super peer based hierarchical overlays.

6.2 Gossip-based SBC Protocols

Voulgaris et al. in (Voulgaris et al. 2007, Voulgaris & Steen 2013) introduce a gossip-based semantic clustering protocol named Vicinity. This protocol uses Cyclon (Voulgaris et al. 2005) as a peer sampling service. On the basis of a selection function, Vicinity converges to a set of optimal neighbors for each user in the network. The similarity metric used by Vicinity is further analyzed by (Busnel & Kermarrec 2005) to incorporate peer generosity and file popularity to it. They introduced Vicinity-based decentralized algorithm for similarity measurement using the eDonkey2000 dataset. The results give a good idea about the in-degree distribution of the network and the hit ratio of the similarities considered (i.e. random, overlap and others). Another semantic clustering protocol, which is very close to Vicinity, is T-Man. Although, Vicinity is quite different from the original T-Man, the most recent version of T-Man (Jelasity et al. 2009) is quite similar to the baseline version of Vicinity protocol as explained in (Voulgaris & Steen 2013). The difference lies between the two protocols in terms of garbage collection and bandwidth usage. Vicinity is found better in these two due to its round robin policy for neighbor selection.

6.3 Gossip-based SBC Systems

In (Ormándi et al. 2010), different versions of T-Man (Jelasity et al. 2009) are compared with Buddy-Cast protocol. This is the most relevant work to our research. They proposed T-Man-based algorithm that produces a topology with favorable convergence speed and balanced load in a user-based collaborative filtering system. All datasets used for comparison including Movielens, Jesters and Book Crossing, have a Power-law distribution. Kermarrec et al. (Kermarrec et al. 2010) use random walks to handle sparsity of data in decentralized recommendation systems. They also compare two different similarity measures to find which one is suitable for their T-Man-based random walk recommendation algorithm. In further work by (Kermarrec & Taíani 2012), authors make use of heterogeneous similarities instead of homogeneous one, for generating recommendations for decentralized social networks. The research (Magureanu et al. 2012), focus on designing a T-Man-based recommendation system which is efficient in terms of accuracy of recommendations and item coverage. They introduced a trust metric to calculate trust between users. Two different datasets; Yahoo! web-scope and Epinion, are used to analyze sparsity. The work by (Baraglia et al. 2013) gives the concept of developing public and private communities on the basis of similarity between different user profiles. The authors compare their proposed protocol with Vicinity and Cyclon using Movielens dataset, although, the results are very close to Vicinity protocol. Tribler (Pouwelse et al. 2008), which is a popular peer-to-peer Bit-Torrent client application is developed using Buddy-Cast (Bakker et al. 2009), for the purpose of generating and receiving recommendations between users. This shows the significance and suitability of gossip protocols for the purpose of SBC but it does not give any idea of how the convergence behavior changes when the degree or similarity distribution of the network changes, and how this affects the results based upon them.

7 Conclusion

Decentralized SBC overlays based on gossiping offer a simple and scalable platform to a variety of applications such as document clustering, information filtering and anomaly detection. The suitability of such overlays to a particular application depends upon their convergence level. There are limited studies about the convergence of these overlays based on factors such as similarity distribution, message load and size of the network.Fast convergence of SBC overlays to a percentage, is desirable because reaching convergence costs bandwidth. Using unbounded bandwidth results in the fastest convergence but is unreasonable in practical applications. Studying the trade-off between convergence speed and bandwidth, gives a bandwidth conserving solution with ideal convergence speed. We found that there exists a threshold for gossip length, at which the convergence speed is nearly optimal and after that utilizing more bandwidth for increasing convergence speed is largely wasting resources. These settings are independent of the type of distribution. However, the convergence time is different for each distribution due to different network characteristics such as clustering co-efficients. Experiments with message load show that bandwidth is a critical parameter for the nodes with high message load especially when network size is very large. Therefore, using optimal settings leads to bandwidth-efficient convergence. Future work includes implementation of this work with some real world data where profiles of users

are dynamic and the network size is very large to realize a gossip-based SBC system.

References

Bakker, A., Mol, J. et al. (2009), 'Tribler protocol specification'.
URL: *http://svn.tribler.org/bt2-design/proto-spec-unified/*

Baraglia, R., Dazzi, P. et al. (2013), 'A peer-to-peer recommender system for self-emerging user communities based on gossip overlays.', *Journal of Computer and System Sciences* **79**(2), 291 – 308.

Bertier, M., Frey, D., Guerraoui, R., Kermarrec, A.-M. & Leroy, V. (2010), The gossple anonymous social network, *in* 'Proceedings of the ACM/IFIP/USENIX 11th International Conference on Middleware', Middleware '10, Springer-Verlag, Berlin, Heidelberg, pp. 191–211.

Boutet, A., Frey, D., Guerraoui, R., Jegou, A. & Kermarrec, A.-M. (2013), Whatsup: A decentralized instant news recommender, *in* 'Parallel Distributed Processing (IPDPS), 2013 IEEE 27th International Symposium on', pp. 741–752.

Busnel, Y. & Kermarrec, A.-M. (2005), *Integrating File Popularity and Peer Generosity in Proximity Measure for Semantic-based Overlays*, Publication interne - IRISA, IRISA.

Cohen, B. (2008), 'Bittorrent'.
URL: *http://www.bittorrent.org/*

Community, Y. D. (2012), 'Yacy search engine'.
URL: *http://yacy.net/en/*

Handurukande, S., Kermarrec, A.-M. et al. (2004), Exploiting semantic clustering in the edonkey p2p network, *in* '11th workshop on ACM SIGOPS European workshop', EW 11, New York, NY, USA.

Jelasity, M. & Babaoglu, O. (2005), T-man: Gossip-based overlay topology management, *in* 'Workshop on Engineering Self-Organising Applications', Springer-Verlag, pp. 1–15.

Jelasity, M., Montresor, A. & Babaoglu, O. (2009), 'T-man: Gossip-based fast overlay topology construction', *Computer Networks* **53**(13), 2321 – 2339. Gossiping in Distributed Systems.

Jelasity, M., Voulgaris, S. et al. (2007), 'Gossip-based peer sampling', *ACM Trans. Comput. Syst.* **25**(3).

Kermarrec, A.-M., Leroy, V., Moin, A. & Thraves, C. (2010), Application of random walks to decentralized recommender systems., *in* 'Lecture Notes in Computer Science', number 6490 *in* 'International conference on principles of distributed systems', pp. 48 – 63.

Kermarrec, A.-M. & Taíani, F. (2012), Diverging towards the common good: heterogeneous self-organisation in decentralised recommenders, *in* 'Workshop on Social Network Systems', SNS '12, New York, USA, pp. 1–6.

Labs, G. R. (2011), 'Movielens dataset'.
URL: *http://www.grouplens.org/node/12*

Labs, Y. (2012), 'Webscope dataset'.
URL: *http://webscope.sandbox.yahoo.com/*

Ltjohann, D. S., Shah, A. H. et al. (2011), ''sciencenet' - towards a global search and share engine for all scientific knowledge.', *Bioinformatics* **27**(12), 1734–1735.

Magureanu, S., Dokoohaki, N. et al. (2012), Design and analysis of a gossip-based decentralized trust recommender system, *in* '4th ACM Recommender Systems (RecSys) Workshop on Recommender Systems and the Social Web'. QC 20130220.

Massa, P. (2003), 'Epinions dataset'.
URL: *http://www.trustlet.org/wiki/*

Mekouar, L., Iraqi, Y. & Boutaba, R. (2012), 'An analysis of peer similarity for recommendations in p2p systems.', *Multimedia Tools and Applications* **60**(2), 277 – 303.

Montresor, A. & Jelasity, M. (2009), PeerSim: A scalable P2P simulator, *in* 'International Conference on Peer-to-Peer (P2P'09)', Seattle, WA, pp. 99–100.

Ormándi, R., Hegedűs, I. & Jelasity, M. (2010), Overlay management for fully distributed user-based collaborative filtering, *in* 'International Euro-Par conference on Parallel processing: Part I', EuroPar'10, Springer-Verlag, Berlin, Heidelberg, pp. 446–457.

Pouwelse, J. A., Garbacki, P. et al. (2008), 'Tribler: A social-based peer-to-peer system: Research articles', *Concurr. Comput. : Pract. Exper.* **20**(2), 127–138.

Voulgaris, S., Gavidia, D. & Steen, M. V. (2005), 'Cyclon: Inexpensive membership management for unstructured p2p overlays', *Journal of Network and Systems Management* **13**, 197–217.

Voulgaris, S., Kermarrec, A.-M. et al. (2004), Exploiting semantic proximity in peer-to-peer content searching., *in* 'IEEE Computer Society workshop on future trends of distributed computing systems', Vol. 10 of *IEEE International Workshop on Future Trends of Distributed Computing Systems; FTDCS 2004*, pp. 238 – 243.

Voulgaris, S. & Steen, M. (2013), Vicinity: A pinch of randomness brings out the structure, *in* D. Eyers & K. Schwan, eds, 'Middleware 2013', Vol. 8275 of *Lecture Notes in Computer Science*, Springer Berlin Heidelberg, pp. 21–40.

Voulgaris, S., Steen, M. & Iwanicki, K. (2007), 'Proactive gossip-based management of semantic overlay networks: Research articles', *Concurr. Comput. : Pract. Exper.* **19**(17), 2299–2311.