Gossip Based Information Dissemination in a Distributed System

[1] Navaraj Chettri, [2] Dhruba Ningombam

[1, 2] Computer Science & Engineering, Sikkim Manipal Institute of Technology
Gantok, India

Abstract- Gossip protocol is one of the most scalable and reliable protocol for message dissemination in a large network where nodes joins and leaves the network frequently. In this paper we have implemented gossip protocol for message/ information dissemination using two style of epidemic information dissemination i.e. rumor mongering and anti-entropy. The membership is managed by using CYCLON membership protocol. Here we analyze the change in behavior of four performance metrics of gossip protocol which are: Average Path Length, Average Clustering Coefficient; and In-degree and average message exchange with respect to different number of nodes.

Keywords- *Membership, Gossip Protocol, Average cluster coefficient, Average path length, Rumor Mongering, Anti Entropy.*

1. Introduction

A gossip protocol also called "Epidemic Protocol" was first introduced by A. Demers et al [1] in 1987 for database replication, since then gossip protocol has been used for message dissemination in P2P system [2, 3, 4]. The gossip approach has also been applied to other problems, especially for its inherent scalability. For example, gossip algorithms are used to solve the distributed averaging problem [6, 7, 8,9] or distributed computation of separable functions [10]. On the other side, some works analyzed gossip as a suitable protocol for failure detection [11], ad-hoc routing [12], network size estimation [13], load balancing [14, 15], and communication optimization in wireless [16] and sensor networks [17, 18].

Gossip protocol in recent time has been developed as an efficient protocol for disseminating the information in large scale dynamic system, especially in P2P system deployed on Internet or ad-hoc Network. Through gossip protocol message are spread or propagated in the same way of contagious diseases in human population, nodes passes data to randomly chosen neighbor in the same way the infected individuals spread the virus.The rest of the paper is organized as follows. In Section 2, we have given the background of two gossip protocol i.e. rumor mongering and anti-entropy for information dissemination and CYCLON protocol for membership management. In Section 3, we have described the methods of gossip protocol; section 4 describes the theoretical notations and the basic scheme of gossip protocol. Section 5 describes the architecture of the gossip protocol for message dissemination; section 6 shows the evaluation results that we have obtained during our simulation. We conclude this paper in section 6.

2. Background

In this section we describe the two gossip protocol that we have used in our simulation: one for topology bootstrapping and membership management and one for information dissemination.

2.1 Information Dissemination

Information / Data dissemination is a process by which queries or data are routed in the sensor network. The data collected by sensor nodes has to communicate to the base station or to any other node interested in the data.The epidemic protocols for information dissemination have been introduced more than 20 years ago which were inspired from the spreading of epidemics and gossip rumors [1]. Two styles of epidemic protocols have been proposed: rumor mongering and anti-entropy.

In rumor mongering, peers are initially ignorant; when an update is learned by a peer, it becomes a hot rumor. While a peer holds a hot rumor, it periodically (every δ_{RUMOR} time units) chooses a random peer from the current population and pushes (sends) the rumor to it. When a peer p has tried to push a rumor m too many times, m stops being hot and it is retained by p without further pushing, to prevent unbounded dissemination. Among the variants defined by Demers et al. [5], we selected coin and blind, meaning that a rumor stops being hot with a probability P_{RUMOR} after each push operation, independently from the fact that the peers to which the update is sent is ignorant or not.

In anti-entropy, each peer p periodically (every $\delta_{ENTROPY}$ time units) contacts a random partner q selected from the current population; p and q engage in an information exchange protocol where updates known to p but not to q are transferred from p to q (push), and/or vice versa (pull). Among the variants defined by Demers et al. [5], we have selected push-pull, meaning the information flows in both directions.

2.2 Membership Management

Maintaining the list of membership of the entire network and diffusing the information to them in highly dynamic network may incur high network load and might adversely affect the performance. So in order to overcome this problem the membership protocol was introduced that maintains Partial View instead of global view of the membership information of the participating nodes. In this paper we have used CYCLON Protocol for maintaining the membership information.

CYCLON is a cyclic inexpensive membership protocol [5] that keeps the network connected even in a disaster situation without maintaining any global information (whole knowledge of the network) or requiring any sort of administration. It is an enhanced version of shuffling protocol. Each peer maintains a fixed size of neighbour list in its Partial View (the list contains the (ip: Port) of another peer) and peers repeatedly initiates a neighbour exchange operation, known as shuffle that is executed every ΔT time unit. In addition to the network address, cache entries contain an extra field called age, which denotes roughly the age of the entry expressed in ΔT intervals since the moment it was created by the node it point sat.

The Shuffle operation is initiated by a node selecting the neighbour whose information was injected first in its partial view and exchanges the information with that node. The receiving node replies by sending back a subset of nodes from its partial view and updates its own cache to accommodate all received entries. If the shuffle initiator does not get reply from the node within the predefined timeout, it simply assumes that neighbour to be disconnected and removes the node from its Partial View.
Whenever a new node 'A' wants to join the over lay the node simply needs to know any single node that is already part of the overlay, called its introducer. The joint operation is based on fixed length random walks on the overlay. The join process ensures that, if there are no message losses or node failures, the in-degree of all nodes will remain unchanged. Additionally, the partial view of the new node will exhibit the same properties of the partial views of all other nodes in the overlay.

3. Methods of Gossip Protocol

The methods that are being used in gossip protocol are:

3.1 Push Method

In this method a node receiving a message actively passes it on to a few random other nodes, which recursively do the same until some termination condition is met. The termination condition ensures that the recursion does not go on forever.

3.2 Pull Method

In a pull method, each node periodically probes random peers in the network in hope to reach an already informed peer, and retrieves new messages when available. Typically, during a pull round, random pairs of peers exchange information about the messages they have recently received and request missing messages from each other.

3.3 Eager Push Method

Nodes send messages to random selected peers as soon as they receive them for the first time.

3.4 Lazy Push Method

When a node receives a message for the first time, it gossips only the message identifier (i.e. for instance, the hash of the message) and not the full payload. If peers receive an identifier of a message they have not received, they make an explicit pull request.

3.5 Hybrid Method

Gossip is executed in two distinct phases. A first phase uses push gossip to disseminate a message in a best-effort manner. A second phase of pull gossip is used in order to recover from omissions produced in the first phase.

4. Theoretical Notions and Basic Schemes

There are different gossip methods, but all fall into three main ones: push, pull and hybrid (push-pull). In order to better understand the behavior of each strategy, it's worth introducing the algorithmic approach of gossip by means of epidemiology. According to this terminology, when a single message is created, each node can be in one of these three states:

Susceptible (S): the node doesn't know about the message.

Infected (I): the node knows the message and is actively spreading it.
Removed (R): the node has seen the message, but is not participating in the spreading process (in epidemiology, this corresponds to death or immunity).

These states are related to the behavior of a node with regard to a single message. In the presence of multiple concurrent messages, each node is in a different state for each message. Two known models are SI and SIR, which differ in the number of possible states.

4.1 The SI Model

In the SI model, each node can be susceptible or infected. Once infected, a node cannot change its state anymore.

Algorithm 1: SI gossip

```
1: loop
2:          wait (Δ)
3:          p ← random peer
4:          if push and in state I then
5:                    send message to p
6:          end if
7:           if pull then
8:                    send update-request to p
9:          end if
10: end loop

11: procedure onUpdate (m)
12:         store m: update
13: end procedure
14:
15: procedure onUpdateRequest (m)
16:          if in state I then
17:                    send message to m: sender
18:          end if
19: end procedure
```

Algorithm 1 represents the generic SI procedure. The active thread (lines 1 - 10) is executed every Δtime units. The parameter p gets the value retrieved by a random peer sampling procedure. Statements at line 5 and 8 trigger two other procedures: in the first case, the message sent to node p is stored in its cache (line 12), thus making p switch to state I; in the second case, an update-request executed by a node q makes p, if in state I, send updates to q (line 17). The two boolean parameters push and pull describe the dynamics of the algorithm. Depending on these parameters, we can talk about push, pull, and push-pull gossip. In push gossip, susceptible nodes are passive and infective nodes actively infect the population. In pull and push-pull gossip each node is active.

4.2 The SIR Model

According to the SI model, nodes continue forwarding useless updates endlessly, because they don't have memory of already forwarded messages.

Algorithm 2: SIR gossip

```
1: loop
2:          wait (Δ)
3:           p← random peer
4:          if push and in state I then
5:                    send message to p
6:          end if
7:           if pull then
8:                    send update-request to p
9:           end if
10: end loop
11:
12: procedure onFeedback (m)
13:         switch to state R with prob. 1=k
14: end procedure

15: procedure onUpdate (m)
16:         if in state I or R then
17:                    send feedback to m: sender
18:         else
19:                    store m: update
20: end if
21: end procedure
22:
23: procedure onUpdateRequest (m)
24:         if in state I then
25:                    send message to m: sender
26:         end if
27: end procedure
```

The SIR model faces the termination problem considering the age of each message and basically discarding too `old' messages, thus stopping their propagation. In other words, each node switches to the removed state in relation to a message when its age crosses a threshold value. Lifespan of messages represents a trade-off between complete coverage and redundancy. In fact, the expected result of a gossip protocol is 100% coverage with as little network overhead as possible that is absence of redundant messages. Algorithm 2 shows the SIR gossip variant. It's the same as SI gossip, but it shows a different behaviour in procedure onUpdate (line 15). The first reception of a message makes the node switch to state I. A new reception of the same message has a probability of 1=k to make the node switch from state I to state R, which means immunity. If this happens, the node is not able to forward that message anymore, neither by pushing (line 4) nor

answering to an update request (line 24). Thus, a correct tuning of the value k is crucial for good performances.
SI and SIR are the basic propagation models and they can be implemented using push, pull or push-pull approach.

5. Architecture

The Gossip or "Epidemic" Protocol we have simulated is based on two basic protocols: Membership Protocol and epidemic broadcast protocol described in section 2. In traditional P2P system, the Membership protocol is used to deal with the peers membership problems, and epidemic broadcast protocol (Rumor Mongering and Anti-Entropy) for information dissemination. Normal peer maintains the information (information to be diffused and partial views of the systems membership), and executes active protocol (Membership Protocol and epidemic broadcast with two sub protocols rumor mongering and anti-entropy)

The architecture of Gossip or "Epidemic" Protocol is shown in figure1;the rumor mongering is used to push updates as quickly as possible towards the peers. As rumor mongering requires lesser resources then anti-entropy so, a cycle length smaller than the one used for anti-entropy can be used for rumor mongering. On the other hand, there is chance that every peer in the network may not receive the information or updates because the rumor prematurely stops being hot so to ensure that each and every peer receives the updates the rumor mongering is backed up by anti-entropy protocol. Anti-entropy also enables peers joining the network to receive updates created during their absence.

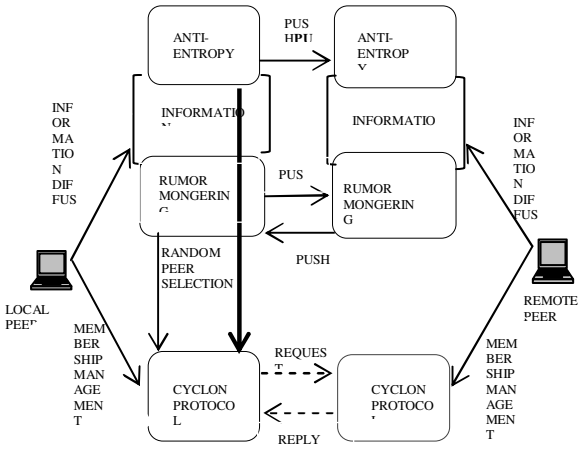

Figure 1: The Architecture of the gossip protocol for information diffusion and membership management.

The membership protocol (CYCLON Protocol) is used to maintain the membership information and keep the network connected even in heavy node failure. It helps in selecting up-to-date subset of random peers from the entire network for information diffusion and sending request and replies among them. In CYCLON, each peer maintains a partial view of the system, i.e. a collection of entries(the entries contains the IP and Port Number of other peers) representing a subset of the entire population of peers. Each peer periodically selects a random peer from its partial view to perform an epidemic information exchange, during this operation the oldest entry are discarded, existing entries are shuffled between the nodes, and new entries are created (to advertise that the peer involved in the exchange are still active). The CYCLON Protocol is modeled by means of two distinct threads (Passive and Active thread) executed at each peer: the active thread takes the initiative to communicate, while the passive thread accepts incoming exchange messages.
Parameters of the CYCLON protocol are the maximum size of the partial view (called c) and the size of the messages sent around (called s); they are both measured as number of entries included in them.

Finally the two threads are combined and perform a continuous random shuffling of the views of peers participating in the protocol, with requests and replies exchanged among them. The CYCLON protocol provides sufficient random samples and maintains a random overlay topology which is robustly connected network in normal operation (with relatively low churn), as well as in massive churn and even after catastrophic failures, by quickly removing failed peers from the local views of correct peers.

6. Evaluation

In this paper we have implemented Gossip protocol for information dissemination in large dynamic distributed system. We have simulated the protocol in PeerSim Simulator [19]. We have implemented two epidemic style protocol anti-entropy and Rumor mongering for information dissemination and CYCLON protocol for managing the member as Membership Protocol. In this paper we have evaluated the protocol using performance metrics such as Average path length, clustering co-efficient, In-degree distribution and average message exchange. The literature states that the performance metrics like average path length, clustering co-efficient must be low in order to have an efficient dissemination of information in the network.

6.1 PeerSim (A Peer-to-Peer Simulator)

PeerSim is a Peer-to-peer simulator [19] that supports simulation of millions of nodes that joins and leaves the network at any point of time. It was started under EU projects BISON and DELIS and the main developer were Mark Jelasity, Alberto Montresor, Gian Paolo Jesi, and

Spyros Voulgaris. The peerSim developed in Trento is now partially supported by the Napa-Wine project.
PeerSim has been developed with extreme scalability and support for dynamicity .It is written in java and it is composed of two simulation engines, a simplified (cycle-based) one and event driven one. The engines are supported by many simple, extendable, and pluggable components, with a flexible configuration mechanism.

The cycle-based engine, to allow for scalability, uses some simplifying assumptions, such as ignoring the details of the transport layer in the communication protocol stack. The event-based engine is less efficient but more realistic. Among other things, it supports transport layer simulation as well. In addition, cycle-based protocols can be run by the event based engine too.

6.2 Average Path Length

Table 1: Average Path Length

No. of Nodes	*Average Path Length*
100	1.7575
500	2.1707
1000	2.4212
5000	2.8850
10000	3.06608
20000	3.33736

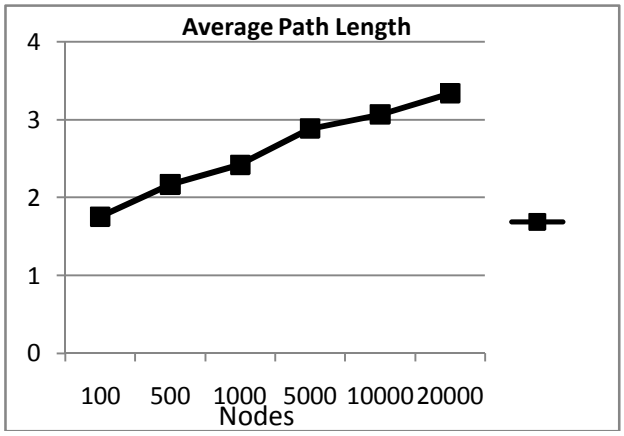

Figure 2: Average path length

The shortest path length between nodes *A* and *B* is the minimum number of edges needed to traverse to reach *B* from *A*. The average path length is the average of the shortest path lengths between any two nodes. The average path length is a metric of the number of hops (and hence, communication costs and time) to reach nodes from a given source. A small average path length is therefore essential for broadcasting or, generally, information dissemination applications. In this paper we have evaluated the protocol for different numbers of nodes as mentioned below in the figure 1. This result has been obtained by conducting series of simulation involving network size of 100 up to 20000. The results shows that as the number of nodes increases the average path length also increases which obviously does but the values we have obtained is low which helps in efficiently disseminating the information in a large dynamic network.

6.3 Average Clustering Coefficient

Table 2: Average Clustering Coefficient:

No. of Nodes	*Average Clustering Coefficient*
100	0.2824
500	0.5707
1000	0.29077
5000	0.00734
10000	0.0049
20000	0.0026

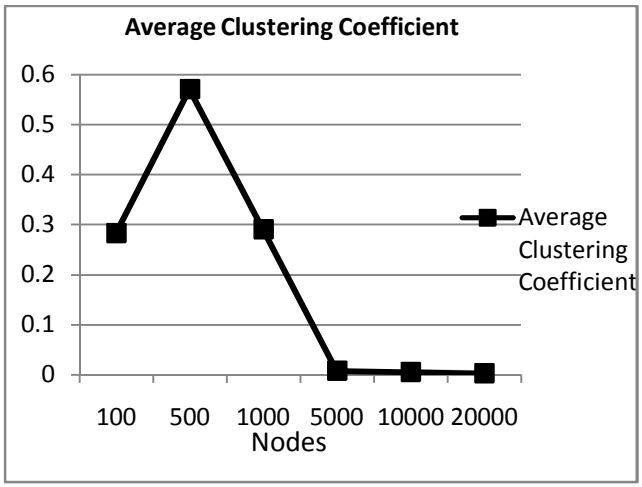

Figure 3: Average Clustering Coefficient

Average Clustering Coefficientis ratio of the existing links among the node's neighbours over the total number of possible links among them. It basically shows to what percentage the neighbours of a node are also neighbours among themselves. A high average clustering coefficient indicates higher chances of network partitioning and high number of redundant message deliveries. The result that has been shown above is obtained by numbers of simulation involving network size up to 20000. The results show that the average clustering coefficient of our

membership protocol decreases as the number of nodes increases. So by this result we can say that the performance of the protocol is extremely good in large scale distributed systems. The literature states that in order to have efficient information dissemination the average clustering coefficient must be low.

6.4 In Degree Distribution

Table 3: In degree Distribution

Nodes	*In-degree Distribution*
100	1389
500	6739
1000	13576
5000	67745
10000	135377
20000	270163

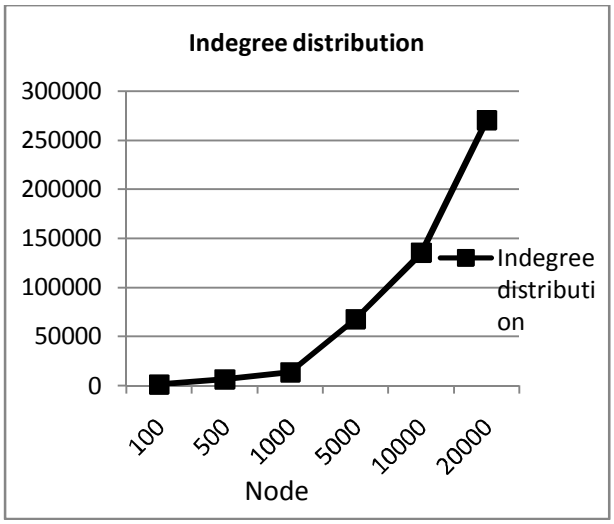

Figure 4 : In degree Distribution

The degree of a node is the number of links it has to other nodes, in the undirected connection graph. The in-degree of a node is the number of edges ending at this node in a directed graph. The degree distribution is related to the robustness of the overlay in presence of failures and indicates how the resources are distributed across the nodes. We consider only the in-degree because the out-degree will be a fixed value equals to the cache size.

6.5 Average Message Exchange

Table 4 Average number of times Message exchanged per peer

Nodes	*Average message exchange*
100	14.65
500	13.46
1000	16.18
5000	15.72
10000	16.02
20000	11.12

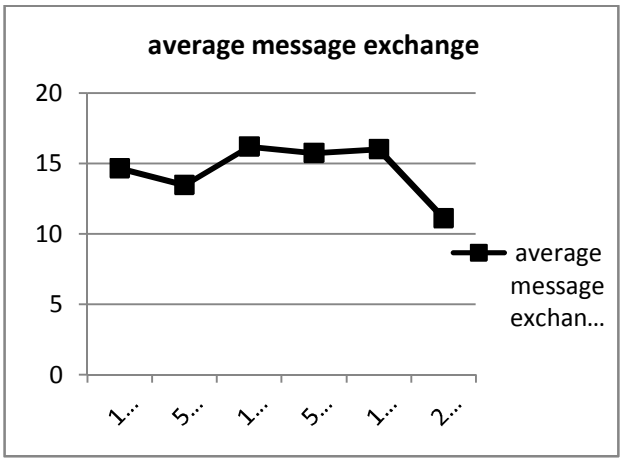

Figure 5: Average number of times Message exchanged per peer

This result shows the average number of time an infected message is exchanged between nodes, measured as the average number of times single information is exchanged among peers. The values of the results are dependent on the rumor mongering strategies, the values of configuration parameters like P_{RUMOR}, the ratio between δ_{RUMOR} and $\delta_{ENTROPY}$and the maximum delay that can be tolerated. We have obtained an acceptable communicational overhead values from this simulation for message dissemination in a large distributed dynamic network.

7. Conclusion and Future Works

In this paper we have implemented gossip protocol for message/ information dissemination using two style of epidemic information dissemination i.e. rumor mongering and anti-entropy. The membership is managed by using CYCLON membership protocol. The results of our implementation mentioned above, states that the performance of the protocol increases as the number of network size increases as expected. The results of the performance metrics are relatively low as mentioned in literature for better performance of information dissemination. Hence, we can conclude from our simulation that gossip protocol is one of the best tools for information dissemination in large and dynamic distributed system.

The future direction for this work is that the delay factor in the message exchange could be reduced by the use of other

epidemics information dissemination strategies (like rumor feedback, counter). The membership maintenances strategies could be changed like instead of using CYCLON Membership protocol HyPar View Membership Protocol could be used to check the performance of the protocol. The continue pursuit of this paper will include this entire factor.

References

[1] A. Demers et al., "Epidemic algorithms for replicated database maintenance," in Proc. of the 6th ACM Symp. On Principles of Distributed Computing (PODC'87). Vancouver, BC, Canada: ACM Press, Aug. 1987, pp. 1–12

[2] Birman, K. P., Hayden, M., Ozkasap, O., Xiao, Z., Budiu, M., and Minsky, Y Bimodal multicast. *ACM Trans. Comp. Syst.*, 17(2):41–88,1999

[3] Kermarrec, A.-M., Massouli´e, L., and Ganesh, A. J. (2003). Probabilistic Reliable Dissemination in Large-Scale Systems. *IEEE Trans. Par. Distr. Syst.*, 14(2):248–258.

[4] Eugster, P. T., Guerraoui, R., Kermarrec, A.-M., and Massouli´e, L. (2004). Epidemic information dissemination in distributed systems. *IEEE Computer*, 37(5):60–67.

[5] Spyros Voulgaris, Daniela Gavidia, and Maarten van Steen, "CYCLON: Inexpensive Membership Management for Unstructured P2P Overlays", June 2005, Journal of Network and Systems Management, Vol. 1

[6] Ming Cao, Daniel A. Spielman, Edmund M. Yeh. "Accelerated gossip algorithms for distributed computation". In Proc. 44th Annu. Allerton Conf. Commun. Control Comput., Monticello, IL, Sep. 2006.

[7] Stephen Boyd, Arpita Ghosh, Balaji Prabhakar, Devavrat Shah. "Analysis and optimization of randomized gossip algorithms". Decision and Control, 2004. CDC. 43rd IEEE Conference on, volume 5, pages 5310 - 5315.

[8] Stephen Boyd, Arpita Ghosh, Balaji Prabhakar, Devavrat Shah. "Gossip algorithms: design, analysis and applications". INFOCOM 2005. 24th Annual Joint Conference of the IEEE Computer and Communications Societies, Proceedings IEEE. Volume 3, pages 1653 - 1664.

[9] Paolo Frasca, Fabio Fagnani. "Broadcast gossip averaging algorithms: interference and asymptotical error in large networks". arXiv preprint arXiv:1005.1292, 2010.

[10] Damon Mosk-Aoyama, Devavrat Shah. Fast distributed algorithms for computing separable functions. Information Theory, IEEE Transactions on, july 2008. Volume 54, issue 7, pages 2997 - 3007.

[11] Robbert van Renesse, Yawn Minsky, Mark Hayden. " A gossip-style failure detection service". Middleware'98, 1998, pages 55 - 70.

[12] Zygmunt J. Haas, Joseph Y. Halpern, Li Li, " Gossip-Based ad hoc routing". INFOCOM 2002. Twenty-First Annual Joint Conferences of the Bibliography 85 IEEE Computer and Communications Societies. Proceedings. IEEE. Volume 3, pages 1707 - 1716.

[13] Ali Ghodsi, Sameh El-Ansary, Supriya Krishnamurthy, Seif Haridi, " A self-stabilizing network size estimation gossip algorithm for peer-to-peer systems". Technical Report T2005:16, SICS (2005).

[14] Mauro Franceschelli, Alessandro Giua, Carla Seatzu, " Load balancing on networks with gossip-based distributed algorithms". Decision and Control, 2007 46th IEEE Conference on, pages 500 - 505.

[15] Mauro Franceschelli, Alessandro Giua, Carla Seatzu. Load balancing over heterogeneous networks with gossip-based algorithms. American Control Conference, 2009. ACC '09, pages 1987 - 1993.

[16] Eytan Modiano, Devavrat Shah, Gil Zussman, " Maximizing throughput in wireless networks via gossiping". SIGMETRICS '06/Performance '06 Proceedings of the joint international conference on Measurement and modeling of computer systems, pages 27 - 38.

[17] Alexandros G. Dimakis, Soummya Kar, Jos_e M. F. Moura, Michael G. Rabbat, Anna Scaglione. "Gossip algorithms for distributed signal processing". Proceedings of the IEEE, november 2010, volume 98, issue 11, pages 1847 - 1864.

[18] Konrad Iwanicki, Maarten van Steen. "The PL-Gossip algorithm". Technical Report IR-CS-034.

[19] http://peersim.sourceforge.net

First Author

Mr. Navaraj Chettri is a student. He has achieved his Master of Computer Application (MCA) from Sikkim Manipal Institute of Technology, Sikkim India and currently pursuing M.Tech degree in CSE from Sikkim Manipal Institute of Technology, Sikkim India. He has recently published one paper", in the proceeding of IRF International Conference, ISBN: 978-93-82702-58-0, pp 51-56.
and his area of interest is networking and his present interest is on wireless Sensor Network routing protocol i.e. "gossip protocol"

Second Author

Mr. Dhruba Ningombam is a research scholar. He has completed his M.Tech from Sikkim Manipal Institute of Technology, Sikkim India, and B.Tech from Government College of Technology, Coimbatore, India. He has published several papers in international conferences and journals. He is also a member of IAENG International Association of Engineers. His area of interest includes Artificial Intelligence, Cloud Robotics Communication and platform.