Analysing the efficiency and robustness of gossip in different propagation processes with simulations

Ruiqi Yang
School of Ocean and Earth Science, Tongji University, Shanghai, SH021, China
angela@cas-harbour.org

Abstract. This paper uses the probability theory to analyse the ultimate consistency of the gossip protocol propagation process by dividing it into two phases. The time complexity and information complexity of the flooding, tree and gossip methods are compared by simulation with different variants. Also, three ways of communication between nodes in the network are introduced: only-push, only-pull, and push-and-pull. The paper analysed the simulated results of variable nodes using those three ways and discussed their corresponding time and information complexity. The simulations and analysis compared performances of information transmission under different transmission loss rates and concluded that the push-and-pull has an effect on transmission structure robustness. Finally, an effective method of information transmission was proposed to deal with transmission loss.

1. Introduction
Gossip is an efficient way to synchronize node state in a distributed network. In gossip, a node that already got information would periodically select neighbours randomly and initiates the updating of information in each round. Gossip was first proposed by Alan J Demers etc. [1] in the paper Epidemic Algorithms for Replicated Database Maintenance published in 1987. Its decentralization features easy, fast, robust and low-load operation. Its extensibility allows it to add and subtract nodes at will, with the newly added nodes perfectly matching other nodes. The increase of nodes has limited impact on the load, ensuring consistent performances. This allows us to scale nodes horizontally in a wider range and disseminate information efficiently. It’s worth noting that restarted nodes due to downtime and newly added nodes would perfectly match the existing nodes over time. That is to say, gossip is fault-tolerating. With this advantage, gossip is widely used in Bitcoin, database maintenance, resource exploration and membership services [2,3].

This paper uses probability theory to analyse the ultimate consistency of the gossip protocol propagation process by dividing it into two phases. The time complexity and information complexity of the flooding, tree and gossip methods are compared by simulation with different variants. Also, three ways of communication between two nodes in the network are introduced: only-push, only-pull, and push-and-pull. The paper analysed the simulated results of variable nodes using those three ways and discussed their corresponding time and information complexity.

Finally, simulation experiments on information transmission under different transmission loss rates were conducted. Moreover, by comparing the effects of push and pull on the robustness of transmission structure, an effective method of information transmission was proposed to deal with transmission loss.
2. Mechanism and characteristics of gossip

Gossip protocol, also known as epidemic protocol [4], is a process of disseminating information through flooding them to their neighbouring nodes randomly. The way gossip spreads information mimics the way people spread rumours: In the beginning, only a few people in a group knew the secret. Then, these people revealed the secrets to a few close friends at random. Finally, just like what’s happening in our life, everyone knew the secret. In gossip algorithm, we view gossip carrier as a node and the secret as information. The process of disseminating is a simple demonstration of the principle of gossip protocol. Here are some of the advantages of the gossip protocol.

2.1 Final consistency

The Gossip process is initiated by a seed node. When a seed node needs to be updated to match other nodes in the network, it will randomly select several nodes around to spread the message, and the node receiving the message will repeat the process until the message is delivered all nodes in the network. This process may take some time, because there is no guarantee that all nodes will receive the message before a certain deadline. However, all nodes will theoretically receive the message eventually, so it is a protocol of final consistency.

2.2 Extensibility

In the propagation process of the gossip protocol, the node that has received the message randomly chooses several nodes around to spread the message, until all the nodes in the network receive the message. Therefore, unlike the direct mail algorithm, during the propagation of gossip algorithm, the network allows increase or decrease of information of the nodes, and the newly added nodes will be the same as the other nodes in the end.

2.3 Fault-tolerant

The decentralization of the gossip protocol means that the information is not stored in a centralized memory. Instead, it is distributed by various nodes. The outage and restart of any node in the network will not affect the spread of gossip messages, making it fault-tolerant as a distributed system.

2.4 Decentralization

The Gossip protocol does not require any central node, which means all nodes are peers. There is no need to know the status of the entire network for individual node. This is why gossip protocol is widely used in unstructured and decentralized network communication, with peer-to-peer network as the core technology.

2.5 Low communication load

The decentralized distribution of information in the gossip protocol means that each node that has received information only needs to transmit the message to its neighbouring nodes. Compared with the centralized transmission mode, the gossip protocol has a lower likelihood of information blockage in the transmission process, thus reducing network overhead traffic.

3. Mathematical proof of communication process

First of all, the probability theory has proved that the gossip protocol theoretically enables all nodes to receive messages at last.

To clarify the procedures, preconditions are set: the author supposes that there are N nodes in the propagation process, and messages are spread periodically. Each round of nodes that have received information randomly select an adjacent node to transmit information. Each time a message is propagated, a node that has not received the message is selected for propagation. The node receiving the message no longer propagates to the message sender. For example, if A sends information to B, then B will not send it back to A when it spreads.
The communication process of gossip is divided into two phases. In the first phase of propagation, less than $\frac{n}{2}$ nodes get the information, that is, most nodes do not know what the message is. In the second phase of propagation, more than $\frac{n}{2}$ nodes receive the message. The time complexity and effectiveness of information transmission in these two stages will be considered respectively.

In the first phase (where less than $\frac{n}{2}$ nodes get the information), each node would pick one neighbour at random in each round. If an informed node chooses an uninformed neighbouring node to spread information, it is considered a “good” choice. Suppose the probability for an informed node to make a “good” choice in a round is $\text{Prob}_{\text{good}}$. Clearly,

$$\text{Prob}_{\text{good}} = 1 - \frac{\text{number of uninformed nodes}}{n-1} \geq 1 - \frac{n}{n-1} \geq \frac{1}{2}$$

Therefore, at most, after $\log N$ rounds, all $\frac{n}{2}$ nodes will receive the message. The time complexity is $O(\log N)$.

In the second phase (more than $\frac{n}{2}$ nodes get the information), suppose the probability for an uninformed node to be informed in the next round is $\text{Prob}_{\text{uninformed node to be informed}}$. Clearly,

$$\text{Prob}_{\text{uninformed node to be informed}} = 1 - \left(1 - \frac{1}{n-1}\right)^{\text{number of informed nodes}} > \frac{1}{2}$$

If we assume that after the first round the number of uninformed nodes is $\frac{n}{4}$ (actually it would be less than $\frac{n}{4}$), then in N round, the number of uninformed nodes is at most $0.75^N$, which means at most after $\log_{0.75} \frac{n}{2}$ rounds, every node receives the message. Clearly, the time complexity is $O(\log n)$.

From the probability analysis we can see that, no matter how large the scale (namely $N$) is, by randomly selecting neighboring nodes in the gossip protocol, eventually all nodes will be informed, thus proving the ultimate consistency of the gossip algorithm.

4. Spreading simulations

4.1 Compare between flooding, tree and gossip

4.1.1 Time complexity

Flooding, tree and gossip are three ways to disseminate information. They are applied in different fields and play different roles in network data transmission [5,6].

Flooding is a method of rapidly spreading routing updates to all nodes of the entire large network. It is also sometimes used for multi-point transmission packets. Extremely fast in information propagation (time complexity of $O(1)$), flooding is used in OSPF protocol. However, flooding needs to transmit and receive loads of information at the same time, leading to a heavy burden for individual node.

Tree is a kind of structured transmission process. Before information can be propagated, a tree structure needs to be established between nodes. Information travels along branches, and only a small amount of information needs to be transmitted. The propagation speed is fast (the time complexity of binary tree is $\log N$), but very poor in terms of robustness. Once a path is blocked, the disconnected node and its branches will not receive information.

If we use gossip protocol to disseminate information in a network with $N$ nodes, assuming that each newly infected node can infect at least one new node in each Gossip cycle, then the gossip protocol degrades into a binary tree, and it is easy to see from the above derivation in Part 3 that, after $\log N$ rounds, the entire network will be infected, with a time cost of $O(\log N)$. 

4.1.2 Simulated results
In order to compare the time complexity and message complexity of flooding, tree and pull, a simulation experiment was designed. Assume that there is a fully connected network with branches of equal length, and set following variables:

- The total number of nodes (N)
- Informed the node (I)
- Time when all nodes are informed (ROUNDS)
- Messages been sent during the whole transmission (M)

(The Time and Message of gossip are averaged over 100 repeated experiments)

| Total nodes | Time | Message |
|-------------|------|---------|
| 65          | 1    | 64      |
| 129         | 1    | 128     |
| 257         | 1    | 256     |
| 513         | 1    | 512     |

Table 2. Performance of variable total number of nodes using binary tree.

| Total nodes | Time | Message |
|-------------|------|---------|
| 65          | 6    | 64      |
| 129         | 7    | 128     |
| 257         | 8    | 256     |
| 513         | 9    | 512     |

Table 3. Performance of variable total number of nodes using gossip, through which each time a informed nodes would choose 1 neighbouring nodes.

| Total nodes | Time | Message |
|-------------|------|---------|
| 65          | 11   | 367     |
| 129         | 13   | 833     |
| 257         | 15   | 1826    |
| 513         | 16   | 3940    |

Table 4. Performance of variable total number of nodes using gossip, through which each time a informed nodes would choose 2 neighbouring nodes.

| Total nodes | Time | Message |
|-------------|------|---------|
| 65          | 7    | 421     |
| 129         | 8    | 947     |
| 257         | 9    | 2097    |
| 513         | 10   | 4604    |

Table 5. Performance of variable total number of nodes using gossip, through which each time a informed nodes would choose 3 neighbouring nodes.

| Total nodes | Time | Message |
|-------------|------|---------|
| 65          | 5    | 518     |
| 129         | 6    | 1165    |
4.2 Communication patterns in Gossip: push and pull

4.2.1 Communication process
Under the Gossip protocol, there are three ways of communication between two nodes in the network [7]:
- Push: node A pushes data (key, value, version) and corresponding version number to B node, and B node updates the new data in A.
- Pull: A will only push data key, version to B, and B will push local new data (key, value, version) to A, and A will update the local data.
- Push/Pull: similar to Pull, with one more step. A pushes the new local data to B, and B updates the local data.

If the data synchronization of two nodes is defined as one cycle, then within one cycle, Push needs to communicate once, Pull needs to communicate twice, and Push/Pull needs to communicate three times. Although the number of messages has increased, Push/Pull is the best in effect, theoretically making the two nodes completely consistent in one cycle. Intuitively, Push/Pull also has the fastest convergence rate.

4.2.2 The probability of a node to be informed
Suppose that the probability of a node being informed at period i is $P_i$, and the probability of being infected at period $i+1$ is $P_{i+1}$, it is easy to derive:
the informed probability of Pull is:

$$P_{i+1} = P_i^2$$

the informed probability of Push is:

$$P_{i+1} = P_i (1 - \frac{1}{n})^n (1 - P_i)$$

Obviously, the convergence rate of Pull is greater than that of Push, and the probability of each node being infected in each cycle is fixed at $P$.

4.3 Simulated results

4.3.1 Only-push

Table 6. Performance of variable total number of nodes using only push, through which each time one informed nodes would choose 1 neighbouring nodes.

| Total nodes | Time | Message |
|-------------|------|---------|
| N           | ROUNDS | M       |
| 65          | 11    | 367     |
| 129         | 13    | 833     |
| 257         | 15    | 1826    |
| 513         | 16    | 3940    |

4.3.2 Only-pull

Table 7. Performance of variable total number of nodes using only pull, through which each time one informed nodes would choose 1 neighbouring nodes.

| Total nodes | Time | Message |
|-------------|------|---------|
| N           | ROUNDS | M       |
| 65          | 9     | 196     |
4.3.3 Push and pull

Table 8. Performance of variable total number of nodes using push in phase I and pull in phase II, through which each time a informed nodes would choose 1 neighbouring nodes.

| Total nodes | Time | Message |
|-------------|------|---------|
| N           | ROUNDS | M     |
| 65          | 9     | 226    |
| 129         | 10    | 499    |
| 257         | 11    | 1032   |
| 513         | 12    | 2231   |

4.4 Transmission loss

4.4.1 Only push

Table9. Performance of variable total number of nodes using only push with transmission loss rate of 3%, through which each time a informed nodes would choose 1 neighbouring nodes.

| Total nodes | Time | Message |
|-------------|------|---------|
| N           | ROUNDS | M |
| 65          | 15    | 589    |
| 129         | 25    | 2242   |
| 257         | 82    | 31619  |
| 513         | N/A   | N/A    |

Table10. Performance of variable total number of nodes using only push with transmission loss rate of 5%, through which each time one informed nodes would choose 1 neighbouring nodes.

| Total nodes | Time | Message |
|-------------|------|---------|
| N           | ROUNDS | M |
| 65          | 19    | 833    |
| 129         | 43    | 4612   |
| 257         | 97    | 147533 |
| 513         | N/A   | N/A    |

4.4.2 Only pull

Table11. Performance of variable total number of nodes using only pull with transmission loss rate of 3%, through which each time one informed nodes would choose 1 neighbouring nodes.

| Total nodes | Time | Message |
|-------------|------|---------|
| N           | ROUNDS | M |
| 65          | 18    | 788    |
| 129         | 40    | 4413   |
| 257         | 94    | 124563 |
| 513         | N/A   | N/A    |
Table 12. Performance of variable total number of nodes using only pull with transmission loss rate of 5%, through which each time one informed nodes would choose 1 neighbouring nodes.

| Total nodes $N$ | Time ROUNDS | Message $M$ |
|-----------------|-------------|-------------|
| 65              | 20          | 1005        |
| 129             | 53          | 5706        |
| 257             | N/A         | N/A         |
| 513             | N/A         | N/A         |

4.4.3 push and pull

Table 13. Performance of variable total number of nodes using push in phase I and pull in phase II with transmission loss rate of 3%, through which each time a informed nodes would choose 1 neighbouring nodes.

| Total nodes $N$ | Time ROUNDS | Message $M$ |
|-----------------|-------------|-------------|
| 65              | 19          | 895         |
| 129             | 45          | 4731        |
| 257             | 202         | 48076       |
| 513             | 255         | 61111       |

Table 14. Performance of variable total number of nodes using push in phase I and pull in phase II with transmission loss rate of 5%, through which each time a informed nodes would choose 1 neighbouring nodes.

| Total nodes $N$ | Time ROUNDS | Message $M$ |
|-----------------|-------------|-------------|
| 65              | 27          | 1341        |
| 129             | 76          | 8286        |
| 257             | 266         | 139302      |
| 513             | 543         | 203238      |

5. Conclusion

From simulation results above we can clearly see that the one-to-many information transfer mode of flooding has the best time complexity and message complexity (From Table 1 to Table 5). As for binary tree, the time complexity is the same as the depth of the tree, which is always $O(\log N)$. Compared with the former two ways, the gossip protocol has a larger time complexity and message complexity. However, with the increasing of informed nodes in each round, the delivery time would be reduced. When the number of informed nodes reaches 3, the gossip protocol is already faster than binary tree.

As indicated by our previous probability calculations, Pull has a faster convergence rate (From Tables 6 to Table 8). With the same number of total nodes and informed neighbouring nodes per round, only-pull requires less propagation time and less information to be sent than only-push. When push and pull are combined, the propagation time is basically the same as only-pull, and the amount of information needed to be propagated is slightly improved compared with only pull, which means without considering the propagation loss, only-pull is more effective than only-push and push-and-pull.

With the increase of propagation loss rate from 3% to 5%, the propagation time and information load will increase for all three ways. In terms of propagation loss, only-push has a faster convergence
rate, shorter transmission time and lower information load compared with only-pull. However, both only-push and only-pull are inefficient in information transmission when the number of total nodes reaches a certain scale (From Table 9 to Table 12). However, From Tables 13 and 14, it can be seen that a combination of push in Phase I and pull in Phase II can avoid the occurrence of N/A and enhance the robustness of the propagation.

References
[1] Alan Demers, Dan Greene, Carl Hauser, Wes Irish, John Larson, Scott Shenker, Howard Sturgis, Dan Swinehart, Doug Terry, “Epidemic Algorithms for Replicated Database Maintenance” Proceedings of the sixth annual ACM Symposium on Principles of distributed computing, pp.1-12, August 10-12, 1987
[2] S. M. Hedetniemi, S. T. Hedetniemi, and A. L. Liestman, “A survey of gossiping and broadcasting in communication networks,” Networks, vol. 18, pp. 319–349, 1988.
[3] S. Deb, M. Medard, and C. Choute, “Algebraic gossip: a network coding approach to optimal multiple rumor mongering,” IEEE Transactions on Information Theory, vol. 52, no. 6, pp. 2486-2507, 2006.
[4] A. Demers, D. Greene, C. Hauser, W. Irish, J. Larson, S. Shenker, H. Sturgis, D. Swinehart, and D. Terry, “Epidemic algorithms for replicated database maintenance,” in Proceedings of the 6th Symposium on Principles of Distributed Computing (PODC), 1987, pp. 1-12.
[5] D. Kempe, A. Dobra, and J. Gehrke, “Gossip-based computation of aggregate information,” in Proceedings of 44th Symposium on Foundations of Computer Science (FOCS), 2003, pp. 482-491.
[6] Hyocheol Jeong, Hyeonjun Jeong, Younghwan Yoo, “Dynamic Probabilistic Flooding Algorithm Based on Neighbor Information in Wireless Sensor Networks”, Information Networking (ICOIN), 2012.
[7] Flavio Chierichetti, Silvio Lattanzi, Alessandro Panconesi,” Almost Tight Bounds for Rumour Spreading with Conductance”, STOC’10—Proceedings of the 2010 ACM International Symposium on Theory of Computing, pp.399–407, ACM, New York, 2010.