Sky: an Opinion Dynamics Framework and Model for Consensus over P2P Network

Houwu Chen and Jiwu Shu

Tsinghua University
East Main Building #8-201, Tsinghua Univ., Beijing, China, 100084

Abstract. The decentralized nature of P2P network increases robustness because it removes single points of failure, however, traditional Byzantine consensus does not work in P2P network due to Sybil attack while existing Sybil-proof consensus based on compute power can't resist adversary with dominant compute power. We proposed the sky framework to apply opinion dynamics in P2P network for consensus, as well as the sky model to maximize performance. The sky framework is Sybil-proof through trust relationships and with it consensus may emerge from local interactions of each node with its direct contacts without topology, global information or even sample of the network involved. The sky model has better performance of convergence than existing models in literatures, and its lower bound of fault tolerance performance is also analyzed and proved. Comparing to compute power based consensus, our approach enables disarming faulty or potentially malicious nodes by unfollowing them. To the best of our knowledge, it's the first work to bring opinion dynamics to P2P network for consensus.

Keywords: opinion dynamics, P2P, Byzantine consensus, Sybil attack

1 Introduction

P2P network is well known on its decentralized nature that increases robustness because it removes single points of failure. Emerging cryptocurrencies(e.g., Bitcoin) demonstrate the demand of consensus over P2P network with decentralization still retained [1]. However, to keep decentralization, no logically central and trusted authority vouches for a one-to-one correspondence between entity and identity, thus makes it difficult to resist Sybil attack [2], wherein the adversary creates a large number of pseudonymous identities to gain a disproportionately large influence. Nodes in a P2P network may present Byzantine failure [3], which encompasses both omission failures (e.g., crash failures, failing to receive a request, or failing to send a response) and commission failures (e.g., processing a request incorrectly, corrupting local state, and/or sending an incorrect or inconsistent response to a request). Traditional Byzantine consensus with either signed or unsigned messages generally needs a node to determine value according to the values from majority or at least a sample of all the nodes [3–12], however, they will fail with the presence of Sybil attack. Existing consensus based on

compute power is Sybil-proof but can't resist adversary with dominant compute power [13, 14].

We proposed the *sky* framework to apply opinion dynamics in P2P network for consensus, as well as the *sky* model to maximize performance. In the sky framework, each node is identified by its public key, other nodes follow the node if they trust it, during the process of consensus, each node broadcasts opinion to its followers, which then decide new opinions according to their own followees. In this way, Sybil nodes can freely join the network but they take no effect in consensus among correct nodes, and the network may asymptotically reach almost-everywhere consensus from local interactions of each node with its direct contacts without topology, global information or even sample of the network involved. The sky model has better performance of convergence than existing models in literatures, and its lower bound of fault tolerance performance is also analyzed and proved. Comparing to existing compute power based Sybil-proof consensus, our approach enables disarming faulty or potentially malicious nodes by unfollowing them. Theoretic analysis and simulations both show that it can tolerant failures by at least 13% random nodes while over 96% correct nodes still make correct decision for initial configuration with convergence $\geq 50\%$. Simulations also show that on the SNAP dataset of the Wikipedia who-votes-on-whom network [15] with reasonable latencies, it can reach almost-everywhere consensus within 70 seconds and tolerant failures committed by 2% top influential nodes. To the best of our knowledge, it's the first work to bring opinion dynamics to P2P network for consensus.

2 Related Work

Sybil Attack Resistance Existing approaches to resisting Sybil attack can be classified into several categories. One is relying on a certifying authority to perform admission control [16], however, decentralization is broken by the certifying authority which may also introduce failures. Another one is remotely issuing anonymous certification of identity by identifying distinct property of a node, e.g, utilizing geometric techniques to establish location information [17], but it can't tolerance changes of the network environment which is general in P2P networks. Puzzle computing is also introduced to increase the cost of Sybil attack, such puzzles involve posing a challenge that requires a large amount of computation to solve but is easy to verify [18], however, it forces honest nodes to continually spend computing resources on solving puzzles, and also there's no way to resist Sybil attack if the adversary has dominant computing resources. Sybil prevention techniques based on the connectivity characteristics of social graphs is another direction [19, 20]. Because of the difficulty to engineer social connections between attack nodes and honest nodes, this approach is considered to be more robust over other ones [21–23], and our model is also based on this approach to resist Sybil attack. Those approaches don't target at the consensus problem directly but provide an valuable basis for Sybil-proof Byzantine consensus.

Unbound Participants In a P2P network, peers can join and left freely, and to keep decentralization, there is no central coordinator and it's even impossible to know the exact number of participants taken part in the consensus, thus Byzantine failure is even more difficult to tolerant [24]. Consensus for infinite many processes or unbound concurrency deals with the problem where the exact number is unknown or unbound, but they only handle stop failure instead of Byzantine failure or assume existence of an atomic register [25–30]. Graph theory based Byzantine consensus algorithms can deal with the problem of unknown participants [24], but they are sensitive to the change of topology which is common in P2P network. Random walk based Byzantine consensus can tolerant topology change as well as the case where nodes can join and leave the network continuously over time and achieve almost-everywhere Byzantine agreement with high probability [11]. But all of the work referred here don't take account of Sybil attack.

Cryptocurrency Cryptocurrency(e.g, Bitcoin) is a form of money that use cryptography to control its creation and management, rather than relying on central authorities [31]. Due to its decentralized nature, a cryptocurrency must provide Sybil-proof Byzantine consensus. Bitcoin provides such a mechanism through an ongoing chain of hash-based proof-of-work(PoW) [1], which is actually a puzzle computing based approach. The majority decision of Bitcoin is represented by the longest chain, which has the greatest proof-of-work effort invested in it. However, one has dominant compute power can control the network while the rest of the network has no means to resist it, and the proliferation of ASIC miner and mining pools already leads to the monopoly of compute power [13, 14]. Ripple/Stellar [32] also use a relationship based solution to resist Sybil attack similar to ours, however, their algorithm has a major defect that it relies on the assumption that for a node, if 80% of its followees agree on a opinion, then 80% of all nodes agrees on the same opinion, but the assumption only stands when a node follows an overwhelming majority of all nodes. As reported, Ripple/Stellar and other existing solutions like PoS have problem even bigger than PoW [33, 34].

Opinion Dynamics Opinion dynamics is a field where mathematical-and-physical models and computational tools are utilized to explore the dynamical processes of the diffusion and evolution of opinions in human population [35–37]. Researches on the field shows that opinion might converge when nodes only take local interactions without centralized coordination or global information involved [38, 39] . Various models are studied including voter, majority rule, Sznajd, social impact, and bounded confidence etc [35, 37], they together with their derivatives model various types of phenomena, however they aren't designed to maximize performance of consensus. Some of the models like voter, majority rule and Sznajd can be adapted to P2P network, but others can't, e.g., anyone who set the persuasiveness and supportiveness for each node in social impact model will break the decentralization, and bounded confidence model is for continuous opinions instead of binary ones etc. Committed minority(a.k.a stubborn agent

or zealot) plays a great role in opinion dynamics [40–44]. Communities impact the speed of convergence too [45–48].

3 The Problem and Datasets for Evaluation

In traditional definition of consensus, specifically binary consensus, each node has a initial value $v_i \in \{0, 1\}$, the consensus problem is to decide upon a common value among all nodes. A node is *correct* if it behaves honestly and without error. Conversely, a node is *faulty*. A faulty node may have Byzantine failure exhibiting arbitrary, erratic and unexpected behavior which may even be malicious and disruptive. However, in a P2P network under an eclipse attack [49], an adversary can always isolate some number of correct nodes hence *almost-everywhere consensus* is the best one can hope for in such networks [50]. Similar to existing definition [11, 51, 52], *almost-everywhere consensus* is defined that up to εn correct nodes in a P2P network agreed on the *wrong* value, where n is the network size, and $\varepsilon > 0$ is sufficiently small, the wrong value is 1 if initially 0 is the majority among all correct nodes, and vise versa. Since our approach to consensus is based on opinion dynamics, we use the term *opinion* instead of *value* in later sections to conform the convention of the opinion dynamics field.

We evaluate our approach on the SNAP dataset of Wikipedia who-votes-on-whom [15] called as the *wiki* dataset in later sections, because it presents trust relationships in the form of votes for administration instead of interest or mutual friendship relationships. We also impose a constraint which can be enforced in P2P client of each correct node that $indegree >= 10$, thus all nodes with followees less than 10 are removed. Parameters of the result network is shown in Table 1, and the cumulative distributions of indegrees and outdegrees are shown in Fig. 1.

Table 1. Datasets parameters

Name	Wiki
Nodes Counts	998
Average Degree	33.33
Diameter	5
Average Path Length	2.34
Density	0.033
Average Clustering Coefficient	0.183
Eigenvector Centrality Sum Change	0.029

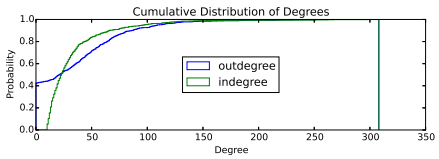


Fig. 1. Degree distribution of the wiki dataset

To facilitate comparing the impact of network size, we also run simulations upon three uniform networks with size of 100, 1000, 5000 nodes, where each node has the same degree and connect to each other randomly. Those dataset are named as *uniform-less*, *uniform* and *uniform-more* respectively.

4 The Framework

We proposed an opinion dynamics framework called the *sky* framework for consensus over P2P network as described in this section.

4.1 Network Constructing

In our framework, each node in the P2P network is owned by somebody and identified by a public key. When the owner of node A trusts the owner of node B, owner of A can set A to follow B in the P2P client, and B is called as *followee* while A is called as *follower*. The network can be abstracted to a directed graph where each peer is a node, and each trust relationship is a directed edge. To ensure connectivity and safety, each correct node is constrained by the P2P client to have at least a minimum number of followees.

4.2 Consensus Process

Nodes in our framework are equally privileged and equipotent participants in the consensus process in any time as ordinary opinion dynamics. However, we introduced the concept of *round* into the consensus process which is commonly used in existing Byzantine consensus but not in opinion dynamics. Starting from a initial state as the first round, each correct node determines when to finish its current round and decides its new value following a common rule according to its current value and the values of its followees, and then enters the next round. The common rule used here shapes *the opinion dynamics model* which will be introduced in section 5. Note here to avoid centralization no global clock or coordinator is used, each node decides how and when to enter next ground separately, thus each node may enter the same round in different time.

A node makes its final decision when enough rounds(e.g., 40) passed. A node is *deciding* before making final decision. If a node finally agrees at 0 or 1, then it's *decided*. A node is *confused* if it's considered to be safe at neither 0 nor 1. For each node, by denoting the count of 0 and 1 in its current value and the values of its followees respectively, the final decision follows the following rules:

1. If $n0 > (n0+n1)*T$ then agree at 0 and the criteria to agree 1 is similar. The T constant controls the strategy to be aggressive or conservative. Greater T results that less nodes to agree at wrong opinion but more nodes to be confused. We use $T = 2/3$ in experiments.
2. If can't agree at 0 or 1, then it's *confused*.

4.3 Message Passing

A followee unidirectional broadcasts signed messages to all its followers. We allow a faulty node's signature to be forged by an adversary, thereby permitting collusion among the faulty nodes. Broadcast is implemented by DHT and asymmetric cryptography. For a node as followee, all its followers and itself form a sharing group(known as a "swarm") identified by the followee's public key. Each broadcasted message is signed with the private key of the followee, and the follwers can check the identity and integrity against the followee's public key.

Each message broadcasted by $node_i$ is a tuple of ($nodeid$, $round$, $opinion$, $state$), where $nodeid$ is the id of $node_i$, $round$ and $opinion$ is its current round and opinion, and $state \in \{deciding, decided, confused\}$.

4.4 Message Handling

According to the well known FLP impossibility [53], consensus cannot be solved deterministically with even a single crash failure in an asynchronous system which may fail to deliver messages, delay them, duplicate them, or deliver them out of order [54], because of the inherent difficulty of determining whether a process has actually crashed or is only "very slow" [55]. We use a *message filter* and a *failure detector* which can make mistakes by erroneously adding nodes to its list of suspects [55].

For a *node*, the message filter will refuse to accept any new messages if it has already made its final decision, and it will always keep at most one message from a followee with the largest round denoted as $round_{max}$ while $round_{max} \geq node.round$. The filter is applied when a node receiving a new message as well as when a node finish a round after broadcasting opinion to its followees.

A failure detector is designed to deal with issues related with asynchronism, and note it does nothing related with Byzantine failure. The key idea of the failure detector is that each node maintains a followee nodes list as well as a suspect nodes list. A message is a *valid message* for a node marked as *node* if $msg.round \geq node.round$ or $msg.state \in \{decided, confused\}$. For each node, initially all followees are in the followee nodes list, in each round, a followee is moved to the suspect nodes list for the followee nodes list if no valid message from it in message buffer for a long time(failure detector time out), while a node is moved from the suspect nodes list to the followee nodes list when a new valid message from it is received.

With the help of message filter and failure detector, a node can apply the common rule which shapes the opinion dynamics model in the following way:

1. If a node received a message passed through the message filter, then it should check whether to apply the common rule or not.
2. On failure detector timeout event for each round, it should check whether to apply the common rule or not.
3. A node apply the common rule only when its message buffer has messages from all nodes in its followee nodes list.

5 The Model

At time t, a node receives all the messages broadcasted by its followees at $t - dt$, then finishes processing the received messages and broadcast its new opinion at t. By designating the opinion of $node_i$ at time t to be $v_i(t)$, the model can be expressed as a function $\mathcal{F}$:

$$v_i(t + dt) = \mathcal{F}(v_i(t), V_i(t)) \tag{1}$$

where $V_i(t) = [v_{f_1}(t), v_{f_2}(t), \ldots v_{f_n}(t)]$ and $f_1, f_2, \ldots j_n$ are followees of $node_i$. In later sections we designate the count of 0 and 1 in $\{V_i(t), v_i(t)\}$ to be $n0_i(t)$, $n1_i(t)$ respectively.

However, due to the difficulty to directly analyze the stochastic process of the interactions between every nodes described in Eq. (1), we build our opinion dynamics model using *mean field theory(MFT)*. MFT studies the behavior of large and complex stochastic models by studying a simpler model. Such models consider a large number of small individual components which interact with each other. The effect of all the other individuals on any given individual is approximated by a single averaged effect, thus reducing a many-body problem to a one-body problem [56]. MFT is widely used in opinion dynamics as an effective modeling method [37, 38, 42, 45]. By MFT, the opinion dynamics model shaped by the common rule can be expressed by a continuous differential equation, and the *round* can be regarded as $dt = 1$ in the corresponding equation shown in Eq. (2).

We denote the densities of correct nodes to be $c = c_0 + c_1$ where c_0 and c_1 are the densities of correct nodes with opinion of 0 and 1, and densities of faulty nodes to be $f = f_0 + f_1 + f_s$ where f_0 and f_1 are the density of faulty nodes with opinion of 0 and 1 and f_s are the density of faulty nodes without opinions broadcasted. So we have $c + f = 1$. We also denote densities of all nodes(including correct and faulty nodes) with opinion 0 and 1 to be a_0 and a_1 respectively, thus we have $a_0 = (c_0 + f_0)/(1 - f_s)$ and $a_1 = (c_1 + f_1)/(1 - f_s)$.

By designating the derivative of c_0 on t to be dc_0/dt which is actually the change speed of c_0, we can have Eq. (2) where s_1 is the probability that a node flips from opinion 1 to opinion 0, and s_0 is the contrary.

$$\frac{dc_0}{dt} = -\frac{dc_1}{dt} = c_1 s_1 - c_0 s_0 \tag{2}$$

We adapt the paradigmatic majority rule(MR) model, and then proposed the *sky* model by incorporating the MR model with a simulated annealing(SA) model we proposed.

5.1 Majority Rule Model

Traditional *majority rule(MR)* model needs to select a group each time and then make all of the nodes in the group conform the majority opinion of the group, however, there's no such group in the sky framework. We adapt the MR model by regarding each node and all of its followees as a group, but instead of making all of the nodes inside the group to have the majority opinion, we just let the node itself to have that opinion without its followees changed. The rule is shown as following:

1. If $n0_i(t) > n1_i(t)$, then set new opinion to 0, and vise versa.
2. If $n0_i(t) = n1_i(t)$, then select from {0, 1} randomly.

We specify the mean indegree and outdegree of a node to be D. According to the first rule, a node flips from opinion 1 to opinion 0 only when the count of its followees with opinion of 1 is less than $D/2$, and vice versa, and according to the second rule, when the count of its followees with opinion of 1 equals to $D/2$, it has probability of 1/2 to flip, thus for Eq. (2), we can have the following

equation:

$$\begin{cases} s_1 = F(\frac{D}{2} - 1; D, a_1) + \frac{1}{2}d(\frac{D}{2}; D, a_1) \\ s_0 = F(\frac{D}{2} - 1; D, a_0) + \frac{1}{2}d(\frac{D}{2}; D, a_0) \end{cases} \tag{3}$$

where $F(k; n, p)$ is the *cumulative distribution function* and $d(k; n, p)$ is the *probability mass function* for k successes in binomial distribution of n trials with probability p.

5.2 Simulated Annealing Model

The *simulated annealing(SA)* model we proposed provides nodes the ability to escape from their current opinion at some probability while keep stable if $n1_i(t)/n0_i(t)$ or $n0_i(t)/n1_i(t)$ is big enough for a node, as shown in the following:

1. If $n0_i(t) > 4 * n1_i(t)$ then set new opinion to 0, while if $n1_i(t) > 4 * n0_i(t)$ then set new opinion to 1.
2. Otherwise set new opinion to 0 with probability of $n0_i(t)/(n0_i(t) + n1_i(t))$ and set new opinion to 1 with probability of $n1_i(t)/(n0_i(t) + n1_i(t))$.

With the notations same as the previous section, for Eq. (2), we can have the following equation:

$$\begin{cases} s_1 = F(0.2D; D, a_1) + \sum_{i=0.2D}^{0.8D} d(i; D, a_1)(\frac{D-i}{D} + \frac{1}{2D}) \\ s_0 = F(0.2D; D, a_0) + \sum_{i=0.2D}^{0.8D} d(i; D, a_0)(\frac{D-i}{D} + \frac{1}{2D}) \end{cases} \tag{4}$$

5.3 Sky Model

For each node, the *sky* model we proposed randomly selects the rule corresponding to the *MR* model probability of *ratio* otherwise selecting the rule corresponding to the *SA* model. In the following sections, we use $ratio = 0.5$ where the *MR* rule and the *SA* rule has the same probability to be chosen. For the *sky* model, dc_0/dt is a linear combination of that of the MR and the SA model as the the following equation, where $d_g c_0/dt$ is Eq. (2) with Eq. (3) and $d_s c_0/dt$ is Eq. (2) with Eq. (4):

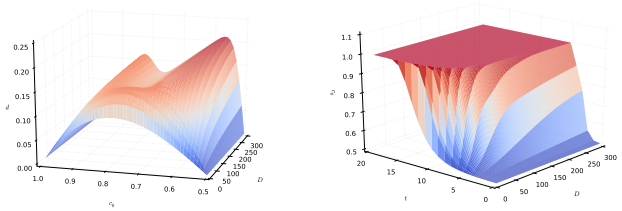
$$\frac{dc_0}{dt} = \frac{d_g c_0}{dt} * ratio + \frac{d_s c_0}{dt} * (1 - ratio) = \frac{1}{2}(\frac{d_g c_0}{dt} + \frac{d_s c_0}{dt}) \tag{5}$$

6 Convergence

Under the assumption that all nodes are correct, we can have $a_0 = c_0$ and $a_1 = c_1$. Because the model is symmetric on binary opinion 0 and 1, and $c_0 + c_1 = 1$, **it's sufficient to only track c_0 and consider $c_0 \geq 0.5$.**

6.1 Numeric Analysis

According to the mean field equations, dc_0/dt(a.k.a. the change speed of c_0) and $\int \frac{dc_0}{dt} dt$ (a.k.a c_0) are demonstrated in Fig. 2a and Fig. 2b respectively.



(a) Derivative of c_0 on t (b) Evolution of c_0

Fig. 2. Numeric analysis of the sky model

From Fig. 2a we can see that $\forall c_0 \in (0.5, 1)$ and $\forall D > 0$, change speed of c_0 is always positive, i.e.,sky c_0 strictly increases with time t. This conclusion can also be proved mathematically, but it won't be presented here due to lack of space in this paper.

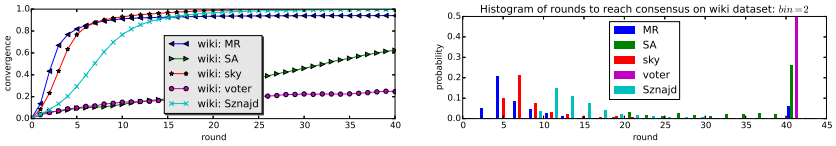
From Fig. 2b we can see that network with greater degree D will converge more quickly. We can also see that with a tiny deviation of c_0 from 0.5, even when $D = 5$, c_0 can still converge to 1 in about 10 rounds.

6.2 Simulations

We simulate the sky model on the wiki dataset for 1000 runs starting with $c_0 = c_1 = 0.5$, where *convergence* is defined as $cvg = |c_0 - c_1|/(c_0 + c_1)$, note here the network may also agree at 1 instead of 0. We also comparing the model to the paradigmatic *voter* model and the *Sznajd* model adapted to our framework as following:

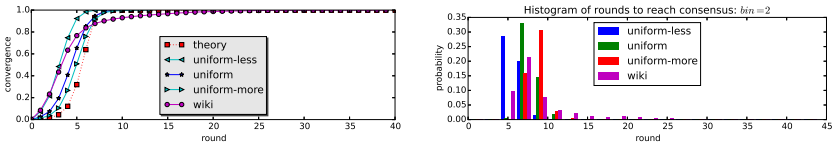
1. Traditional voter model selects a node one time, and chooses the opinion of a randomly select node among the nodes it interacts. We adapt it to the sky framework that for each node the opinion of a random selected node from all of its followees is chosen.
2. Traditional Sznajd model for networks(not the original version for one dimension linear chain) selects a pair of randomly selected nodes who interacts with a random taken third node, if nodes of the pair have the same opinion, then the third node is also set to have the same opinion otherwise nothing happens. We adapt it it to the sky framework that for each node if two randomly chosen followees have the same opinion, then the node set its opinion to that opinion otherwise nothing happens.

The convergence and rounds to converge for all the models on the wiki dataset are show in Fig. 3 , and for the sky model on all the datasets are show in Fig. 4 . Note **round 41 means the network failed to reach consensus within 40 rounds in that run**, and also each bin of the histogram is 2.



(a) Convergence (b) Rounds to reach consensus

Fig. 3. Simulation of all the models on the wiki dataset



(a) Convergence (b) Rounds to reach consensus

Fig. 4. Simulation of the sky model on different datasets

From Fig. 3b , we can see that for the sky model, probability of rounds needed to reach consensus decrease asymptotically when greater than 10. In contract, some of runs of the MR model can never reach consensus, and simulation shows the network may stuck in a stable state where both the nodes with opinion of 0 and 1 exist, but they never change in later rounds. Majority of the runs of the SA model will not reach consensus in 40 rounds, simulations shows that the network may vastly change in each round without steady change direction of convergence, and escape in a tiny probability from the state to the track with convergence steadily increased in each round. Fig. 3a also shows those observations from the perspective of convergence. The voter model has the worst performance so it need no more discussion. For the Sznajd model, its convergence speed is worse than MR and sky models but it has smaller ratio of runs which can't converge within 40 rounds than the MR model. Comparing to sky model, both convergence speed and rounds to converge of the Sznajd model is worse than the sky model. To conclude, the sky model we proposed has the best performance on the wiki dataset than existing paradigmatic models including MR, vote and Sznajd.

From Fig. 4a , we can see that for the sky model on each dataset, simulation result of the sky model approximately fits theoretical analysis. Rounds to converge grows with nodes count(denoted as N), and approaches more closely to theoretical result when N is larger, that's because mean field equation works best when $N \to \infty$, thus $\forall N$ the theoretical result is in fact the theoretical lower bound.

From Fig. 4b , we can see that for the sky model, all the runs on all datasets can reach consensus within 40 rounds. Most of the runs can reach consensus quickly in about 10 rounds. However, average rounds to reach consensus are slightly greater than that of the MR model.

Note that even starts with densities of 0 and 1 to be same and both of them is 0.5, agreement still emerges from the network while in mean field equations

$dc_0/dt = 0$ when $dc_0 = 0.5$, that's because the state in $dc_0 = 0.5$ is unstable and any nonuniform distribution of nodes with opinion 0 or 1 or fluctuation provided by randomization may drive the network away from the unstable state.

7 Fault Tolerance

7.1 Sybil Attack

Sybil attack resistance analysis is straightforward. See Fig. 5, where node marked by A is the current node deciding its new opinion, and A decide its opinions according to opinions broadcasted by its followees including correct nodes C and faulty nodes F while nodes S are Sybil nodes. Because of the difficulty for S to make A trust S which is actually **controlled by A rather than S**, there are no directed links from S to A, so Sybil nodes take no effect when A is deciding its new opinion. Collusion among F and S does not help the attack, because the contribution to the decision of A is still the same with F without S.

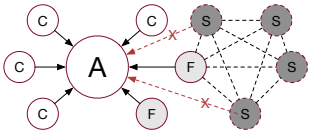


Fig. 5. Sybil Attack

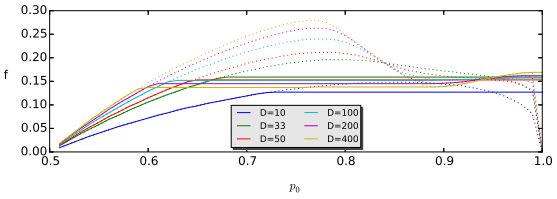


Fig. 6. Critical points

To compromise the network, creating new Sybil nodes or relationships between them are useless, instead, the adversary should attract more correct nodes to follow the nodes controlled by him. Experiments presented in later sections even show that for the same number of trust relationships from correct nodes to faulty nodes, the smaller the faulty nodes number is, the stronger the attack is.

7.2 Lower Bound

Because the model is symmetric on binary opinion 0 and 1, **it's sufficient to only track the case that $c_0 \geq c_1$**.

According to our definition of *almost-everywhere consensus*, a successful consensus process should fulfill the following requirements:

1. If c_0 is far greater than c_1(e.g., $c_0 \geq 2c_1$), then at least $(1-\epsilon)$ proportion of correct nodes should agree at 0.
2. Else at least $(1-\epsilon)$ proportion of correct nodes should agree at the same opinion which is either 0 or 1.

Under Byzantine failures, a faulty node can behave arbitrarily or even collude with other nodes. Different behavior of faulty nodes contributes differently to the evolution of c_0 in the mean field equations. Here are some scenarios:

1. All faulty nodes left the network at $t-dt$, then at t we have $a_0 = c_0/(1-f)$ and $a_1 = c_1/(1-f)$.
2. This scenario is not about failure, but about the dynamic characteristic of P2P network. Same number of correct nodes with opinion 0 joins the network at $t-dt$, then at then at t we have new c_0 denoted as c_0' with $c_0' = 2c_0/(1+c_0)$ together with the corresponding $c_1' = c_1/(1+c_0)$, $a_0' = 2c_0/(1+c_0)$ and $a_1' = c_1/(1+c_0)$.
3. All faulty nodes always broadcast 1 at $t-dt$, then at t we have $a_0 = c_0$ and $a_1 = c_1 + f$.
4. All faulty nodes broadcast 1 to half of their followees and 0 to the other half at $t-dt$, then at t we have $a_0 = c_0 + f/2$ and $a_1 = c_1 + f/2$
5. Faulty nodes broadcast opinion randomly chosen from 0 and 1 at $t-dt$, then at t we also have $a_0 = c_0 + f/2$ and $a_1 = c_1 + f/2$
6. Faulty nodes broadcast 1 when they should broadcast 0 and vise versa at $t-dt$, then at t we have $a_0 = c_0 + f_0'$ and $a_1 = c_1 + f_1'$, where f_0' and f_1' can be calculated according to the mean field equations similar to Eq. (2).

Note that the first two examples show how the agreement evolves in dynamic network, also topology or global view of the network are not involved in our model, and consensus emerges from local interactions of each node with its direct contacts. Failures can't be enumerated exhaustively and they can mix in a network, but since $\max(\frac{c_0+f_0}{1-f_s}) = \max(\frac{c_0+f_0}{c_0+c_1+f_0+f_1}) = c_0 + f$ when $f_0 = f$ and $f_1 = f_s = 0$, and $\min \frac{c_0+f_0}{1-f_s} = c_0$ when $f_0 = f_s = 0$ and $f_1 = f$, the following constraint always stands:

$$\begin{cases} a_0 \in [c_0, c_0 + f] \\ a_0 + a_1 = 1 \end{cases} \tag{6}$$

Lemma 1 (If the network can tolerant any failures committed by given faulty nodes, it must agree at 0).

*For a network with c_0, c_1 and f given at time t to be $c_0(t)$, $c_1(t)$ and $f(t)$, if it can tolerant **any** failures committed by faulty nodes, then it must agree at 0.*

Proof. For the case that $c_0(t)$ that is far greater than $c_1(t)$, it stands according to the almost-everywhere consensus requirements stated above. For the else case, if some failures can stop it to agree at 0, then according to the requirements it must agree at 1, s.t. $\exists$ time $t' > t$ and $c_0(t') \leq \varepsilon(1-f)$. Because of the continuity of c_0 on t, must $\exists t''$, s.t $t' > t'' > t$, $c_0'' \in [c_0(t'), c_0(t)]$, $c_0(t'') = c_1(t) < c_0(t)$ and $c_1(t'') = 1 - f - c_0(t'') = c_0(t'')$. But according to the symmetric property of the model, the failures must also be able to stop it to agree at 1 from time t'', thus leads to contradiction.

Lemma 2 (For given f, greater c_0 tolerant failures equally or better).

For a network with given f, if at two times t' and t'' (no relationship between t' and t'' assumed), s.t $c_0(t') < c_0(t'')$, and for $t > t'$, network can tolerant any failures, then it can also tolerant any failures for $t > t''$.

Proof. If for $t > t'$ and the network can tolerant any failures, then according to Lemma 1, it must agree at 0. We then discuss in two cases. For $c_0(t'') \leq \varepsilon(1-f)$, because of the continuity of c_0 on t, $\exists t'''$ s.t. $c_0(t''') = c_0(t'') \in [c_0(t'), \varepsilon(1-f)]$ and $c_1(t''') = 1 - f - c_0(t''') = c_1(t'')$, thus the network can tolerant any failures for $t > t'''$, we can then conclude the network can also tolerant any failures for $t > t''$. For $c_0(t'') > \varepsilon(1-f)$, if it can't reach consensus successfully, then must $\exists t''' > t''$ s.t $c_0(t''') \in [c_0(t'), \varepsilon(1-f)]$, but it's already known that for $t > t'$ s.t $c_0(t) \in [c_0(t'), \varepsilon(1-f)]$ it can tolerant any failures, thus leads to contradiction.

Lemma 3 (If tolerant smaller a_0, then tolerant greater a_0).

For a network with given f, c_0 and c_1, if at two times t' and t'' (no relationship between t' and t'' assumed), s.t $a_0(t') < a_0(t'')$, and for $t > t'$, network can tolerant any failures, then it can also tolerant any failures for $t > t''$.

Proof. From Eq. (5) we can see that given other parameters, dc_0/dt strictly increases with a_0(note that $a_1 = 1 - a_0$), then $c_0(t' + dt) < c_0(t'' + dt)$. According to Lemma 2, it can also tolerant any failures for $t > t''$.

Theorem 1 (Lower bound of fault tolerance).

For any network with known faulty nodes and initial states of correct nodes, thus c_0, c_1 and f are given, if the network can tolerant the failure that all the faulty nodes always output 1, it can tolerant any other failures.

Proof. According to Lemma 3, and Eq. (6), if a network can tolerant failure with $a_0 = c_0$ together with $a_1 = c_1 + f$, then it can tolerant any other failures. And $a_0 = c_0$ together with $a_1 = c_1 + f$ is exactly the case all the faulty nodes always output 1, thus the theorem stands.

7.3 Fault Tolerance Performance

Because of the constraint that $c_0 + c_1 + f = 1$, it's not convenience to study the performance threshold of fault tolerance on C_0 directly. However, we can translated the threshold question to a new one: if at time t a network with $c_0 = p$ has no faulty nodes, then uniformly choose f proportion of all the nodes(including opinion with 0 and 1) to be faulty, what's the max value of f the network can tolerant?

$f_{critical}$ is the *critical point* for p if $f_{critical}$ fulfill the following two requirements:

1. $\forall f < f_{critical}$, when $t \to \infty$ and under any failures, $c_0/(1-f) \geq 1 - \varepsilon$.
2. $\nexists f'$ such that f' fulfill the previous requirement while $f' > f_{critical}$.

Following the definition of *critical point*, according to Theorem1 for $\varepsilon = 0.05$, by iterating on the mean field equation, critical points can be plotted in Fig. 6, where solid lines are critical points. There are also dotted lines where at each point $dc_0/d_t = 0$. From the figure we can see that $\forall D \in [10, 400]$, as long as $p \geq 0.75$, the network can tolerant any failure with $f \leq 0.13$.

8 Experiment

According to existing studies, latency between peers in DHT is mostly between 50 to 1000 ms [57, 58]. In our experiment, we employ a simply latency model that the time for each message to be delievered conforms gauss distribution of ($\mu = 500$, $\sigma = 500$) with lower cutoff of 50 and no upper cutoff which means a message may never be lost in a small probability even if the node broadcasts it is correct, we also set $timeout = 2000$ for the failure detector.

Since for a network with c_0 far greater than c_1(e.g., $c_0 \geq 2c_1$), reaching consensus at 0 is successful, but that at 1 is failed, we define *signed convergence* as the Eq. (7), thus only if a network survive from failures, *signed convergence* will equal to *convergence* defined earlier.

$$cvg = (c_0 - c_1)/(c_0 + c_1) \tag{7}$$

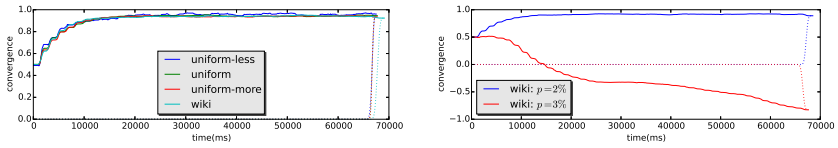
To measure final decision of correct nodes, we also define *decision* as the following equation:

$$decision = |d_0 - d_1|/(d_0 + d_1) \tag{8}$$

where d_0 and d_1 is the count of correct nodes which have final decision on opinion 0 and 1 respectively.

We experiment on network started with $cvg = 0.5$ and $f = 13\%$ while faulty nodes always output 1, then in all decided correct nodes(agree at 0 or 1), for all dataset correct deciding(agree at 0) is about 96%, and uniform datasets have almost the same performance regardless their network scale, as shown in Fig. 7a.

But for the wiki datasets, we also concern the tolerance of failures by collusion of *top* $n\%$ *influential nodes*, defined as the first $n\%$ nodes by sorting all nodes in descendant order on the count of a node's followees. Simulation shows that for the target $\varepsilon < 5\%$, the algorithm can tolerant failures by 2% top nodes on the wiki dataset, as shown in Fig. 7b, where the red dotted lines are the case of failed to reach the goal under failures commited by 3% top nodes. In all decided correct nodes(agree at 0 or 1), correct deciding(agree at 0) is about 96.8%.



(a) By random nodes (b) By top influential nodes

Fig. 7. Convergence under failures

Comparing failures committed by random nodes and top influential nodes, we also find that **more centralized trust relationships leads to more powerful ability to compromise the network** even when the total numbers of trust relationships participated in the collusion are the same, and in theory analysis we have already known that the effect of a specific failure depends on the density of trust relationships for correct nodes to faulty nodes. For the wiki dataset, the

total trust relationships is 33256, and for 13% random nodes, the trust relationships involved is about 4323, while for top 3% nodes, the trust relationships involved is only 2155, in contrast that the network can survive in the former but no in the later. Even excluding the factor that lots of trust relationships are among the faulty nodes which has no effect for correct nodes in the 13% random node case, the result also supports the finding.

9 Discussion and Limitations

Detect and Unfollow If a node is faulty and identified timely, then correct nodes can unfollow it thus it has no impact in later consensus processes. A simple strategy is similar to the failure detector that if a followee has a final decision different to the majority of other followees, it's moved from followee list to a suspect list until it behaves good again. Trust relationships can't be abused many times, and that's also a big advantage over computing power based solutions.

Suppress Top Influential Nodes Top influential nodes generally are most trustworthy nodes, so the chance of collusion between a number of them are much smaller than ordinary nodes with same number. However, to decrease monopoly, a node might unfollow one of its followees if the followees already has too many followers, or assign a weight w with $w < 1$ for that followee.

Distributed Oracles Because in almost-everywhere consensus, there might still be tiny proportion of correct nodes deciding on the wrong opinion even the consensus is considered to be successful. Distributed oracles can be employed to help those node to ensure safety. An oracle behaves exactly the same with a regular correct node, except it does not broadcast messages during consensus process, and it only broadcast its final decision for each consensus process. Thus an oracle is actually a *sink* in the directed graph of trust relationships, and **has no impact on the consensus process**. An oracle must follow many carefully selected nodes in a variety of communities to avoid deciding on the wrong opinion, and it should also publish its followees for public audit. In this way, an oracle acts as a non-intrusive sample of the whole network. Thus a node can subscribe some oracles and compare its own decision against them to ensure safety.

Community Strength Although our approach can successfully runs over the wiki dataset, it also shows the consensus speed degrades comparing to the uniform dataset, as existing studies show that community strength impacts the performance [45–48]. The relationships between our model and community strength need to be studied further.

Fault Tolerance Performance Although our approach can tolerant failures committed by at least 13% nodes when convergence defined in Eq. (7) $\geq$ 50%, otherwise the fault tolerance performance will degenerate as we can see from Fig. 6. However, on the basis of this work, we already developed a consensus mechanism for hash values which can tolerant failures well even when convergence $\leq$ 50% under the premise that hash collision is impossible in reality when hash size is big enough.

10 Conclusion

The sky framework we proposed is Sybil-proof and also applicable in dynamic network. The sky model has better performance of convergence than existing models in literatures, and its lower bound of fault tolerance performance is also analyzed and proved. Comparing to compute power based consensus, our approach enables disarming faulty or potentially malicious nodes by unfollowing them. To the best of our knowledge, it's the first work to bring opinion dynamics to P2P network for consensus.

References

[1] Nakamoto, S.: Bitcoin: A peer-to-peer electronic cash system. `http://www.bitcoin.org/bitcoin.pdf` (2009)

[2] Douceur, J.R.: The sybil attack. In: Revised Papers from the First International Workshop on Peer-to-Peer Systems. pp. 251–260. IPTPS '01, Springer-Verlag, London, UK, UK (2002)

[3] Lamport, L., Shostak, R., Pease, M.: The byzantine generals problem. ACM Trans. Program. Lang. Syst. 4(3), 382–401 (1982)

[4] Attiya, C., Dolev, D., Gil, J.: Asynchronous byzantine consensus. In: Proceedings of the Third Annual ACM Symposium on Principles of Distributed Computing. pp. 119–133. PODC '84, ACM, New York, NY, USA (1984)

[5] Castro, M., Liskov, B.: Practical byzantine fault tolerance. In: Proceedings of the Third Symposium on Operating Systems Design and Implementation. pp. 173–186. OSDI '99, USENIX Association, Berkeley, CA, USA (1999)

[6] Abd-El-Malek, M., Ganger, G.R., Goodson, G.R., Reiter, M.K., Wylie, J.J.: Fault-scalable Byzantine Fault-tolerant Services. In: Proceedings of the Twentieth ACM Symposium on Operating Systems Principles. pp. 59–74. SOSP '05, ACM, New York, NY, USA (2005), 00249

[7] Cowling, J., Myers, D., Liskov, B., Rodrigues, R., Shrira, L.: HQ Replication: A Hybrid Quorum Protocol for Byzantine Fault Tolerance. In: Proceedings of the 7th Symposium on Operating Systems Design and Implementation. pp. 177–190. OSDI '06, USENIX Association, Berkeley, CA, USA (2006), 00204

[8] Kotla, R., Alvisi, L., Dahlin, M., Clement, A., Wong, E.: Zyzzyva: Speculative Byzantine Fault Tolerance. In: Proceedings of Twenty-first ACM SIGOPS Symposium on Operating Systems Principles. pp. 45–58. SOSP '07, ACM, New York, NY, USA (2007), 00288

[9] Clement, A., Wong, E., Alvisi, L., Dahlin, M., Marchetti, M.: Making Byzantine Fault Tolerant Systems Tolerate Byzantine Faults. In: Proceedings of the 6th USENIX Symposium on Networked Systems Design and Implementation. pp. 153–168. NSDI'09, USENIX Association, Berkeley, CA, USA (2009), 00144

[10] Aublin, P.L., Mokhtar, S.B., Quéma, V.: RBFT: Redundant Byzantine Fault Tolerance. In: Proceedings of the 2013 IEEE 33rd International Conference on Distributed Computing Systems. pp. 297–306. ICDCS '13, IEEE Computer Society, Washington, DC, USA (2013), 00007

[11] Augustine, J., Pandurangan, G., Robinson, P.: Fast Byzantine Agreement in Dynamic Networks. In: Proceedings of the 2013 ACM Symposium on Principles of Distributed Computing. pp. 74–83. PODC '13, ACM, New York, NY, USA (2013), 00009
[12] Aublin, P.L., Guerraoui, R., Knežević, N., Quéma, V., Vukolić, M.: The Next 700 BFT Protocols. ACM Trans. Comput. Syst. 32(4), 12:1–12:45 (2015), 00002
[13] Cawrey, D.: Are 51% attacks a real threat to bitcoin? `http://www.coindesk.com/51-attacks-real-threat-bitcoin/` (June 20 2014)
[14] Courtois, N.T., Bahack, L.: On subversive miner strategies and block withholding attack in bitcoin digital currency. CoRR abs/1402.1718 (2014)
[15] Leskovec, J., Krevl, A.: SNAP Datasets: Stanford large network dataset collection. `http://snap.stanford.edu/data` (Jun 2014)
[16] Castro, M., Druschel, P., Ganesh, A., Rowstron, A., Wallach, D.S.: Secure routing for structured peer-to-peer overlay networks. SIGOPS Oper. Syst. Rev. 36(SI), 299–314 (2002)
[17] Bazzi, R.A., Konjevod, G.: On the establishment of distinct identities in overlay networks. In: Proceedings of the Twenty-fourth Annual ACM Symposium on Principles of Distributed Computing. pp. 312–320. PODC '05, ACM, New York, NY, USA (2005)
[18] Borisov, N.: Computational puzzles as sybil defenses. In: Proceedings of the Sixth IEEE International Conference on Peer-to-Peer Computing. pp. 171–176. P2P '06, IEEE Computer Society, Washington, DC, USA (2006)
[19] Yu, H., Kaminsky, M., Gibbons, P.B., Flaxman, A.: SybilGuard: Defending against sybil attacks via social networks. In: Proceedings of the 2006 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications. pp. 267–278. SIGCOMM '06, ACM, New York, NY, USA (2006)
[20] Lesniewski-Laas, C., Kaashoek, M.F.: Whanau: A sybil-proof distributed hash table. In: Proceedings of the 7th USENIX Conference on Networked Systems Design and Implementation. pp. 8–8. NSDI'10, USENIX Association, Berkeley, CA, USA (2010)
[21] Al-Ameen, M.N., Wright, M.: Persea: A sybil-resistant social DHT. In: Proceedings of the Third ACM Conference on Data and Application Security and Privacy. pp. 169–172. CODASPY '13, ACM, New York, NY, USA (2013)
[22] Alvisi, L., Clement, A., Epasto, A., Lattanzi, S., Panconesi, A.: SoK: The evolution of sybil defense via social networks. In: Proceedings of the 2013 IEEE Symposium on Security and Privacy. pp. 382–396. SP '13, IEEE Computer Society, Washington, DC, USA (2013)
[23] Wei, W., Xu, F., Tan, C.C., Li, Q.: SybilDefender: A defense mechanism for sybil attacks in large social networks. IEEE Trans. Parallel Distrib. Syst. 24(12), 2492–2502 (2013)
[24] Alchieri, E.A., Bessani, A.N., Silva Fraga, J., Greve, F.: Byzantine consensus with unknown participants. In: Proceedings of the 12th International

Conference on Principles of Distributed Systems. pp. 22–40. OPODIS '08, Springer-Verlag, Berlin, Heidelberg (2008)
[25] Gafni, E., Merritt, M., Taubenfeld, G.: The Concurrency Hierarchy, and Algorithms for Unbounded Concurrency. In: Proceedings of the Twentieth Annual ACM Symposium on Principles of Distributed Computing. pp. 161–169. PODC '01, ACM, New York, NY, USA (2001), 00051
[26] Aspnes, J., Shah, G., Shah, J.: Wait-free Consensus with Infinite Arrivals. In: Proceedings of the Thiry-fourth Annual ACM Symposium on Theory of Computing. pp. 524–533. STOC '02, ACM, New York, NY, USA (2002), 00022
[27] Merritt, M., Taubenfeld, G.: Resilient Consensus for Infinitely Many Processes. In: Fich, F.E. (ed.) Distributed Computing, pp. 1–15. No. 2848 in Lecture Notes in Computer Science, Springer Berlin Heidelberg (2003), 00013
[28] Chockler, G., Malkhi, D.: Active Disk Paxos with Infinitely Many Processes. Distrib. Comput. 18(1), 73–84 (2005), 00093
[29] Baldoni, R., Bonomi, S., Kermarrec, A.M., Raynal, M.: Implementing a Register in a Dynamic Distributed System. In: 29th IEEE International Conference on Distributed Computing Systems, 2009. ICDCS '09. pp. 639–647 (Jun 2009), 00031
[30] Afek, Y., Morrison, A., Wertheim, G.: From Bounded to Unbounded Concurrency Objects and Back. In: Proceedings of the 30th Annual ACM SIGACT-SIGOPS Symposium on Principles of Distributed Computing. pp. 119–128. PODC '11, ACM, New York, NY, USA (2011), 00003
[31] Brito, J., Castillo, A.: Bitcoin: A Primer for Policymakers. Mercatus Center at George Mason University, 1 edn. (8 2013)
[32] Schwartz, D., Youngs, N., Britto, A.: The Ripple protocol consensus algorithm. `https://ripple.com/files/ripple_consensus_whitepaper.pdf` (2014)
[33] Kim, J.: Safety, liveness and fault tolerance—the consensus choices stellar. `https://www.stellar.org/blog/safety_liveness_and_fault_tolerance_consensus_choice/` (2014)
[34] Poelstra, A.: A treatise on altcoins. `https://download.wpsoftware.net/bitcoin/alts.pdf` (10 2014)
[35] Castellano, C., Fortunato, S., Loreto, V.: Statistical physics of social dynamics. Rev. Mod. Phys. 81(2), 591–646 (2009)
[36] Castelló, X., Baronchelli, A., Loreto, V.: Consensus and ordering in language dynamics. The European Physical Journal B 71(4), 557–564 (2009)
[37] Xia, H., Wang, H., Xuan, Z.: Opinion Dynamics: A Multidisciplinary Review and Perspective on Future Research. Int. J. Knowl. Syst. Sci. 2(4), 72–91 (2011), 00000
[38] Hegselmann, R., Krause, U.: Opinion dynamics and bounded confidence:models, analysis and simulation. Journal of Artificial Societies and Social Simulation 5(3) (2002)
[39] Lyst, J., Kacperski, K., Schweitzer, F.: Social impact models of opinion dynamics. Annual reviews of computational physics 9, 253–273 (2002), 00092 bibtex: lyst2002social

[40] Xie, J., Sreenivasan, S., Korniss, G., Zhang, W., Lim, C., Szymanski, B.: Social consensus through the influence of committed minorities. Physical Review E 84(1), 011130 (2011)
[41] Singh, P., Sreenivasan, S., Szymanski, B.K., Korniss, G.: Accelerating consensus on coevolving networks: The effect of committed individuals. Phys. Rev. E 85(4), 046104 (2012)
[42] Xie, Jierui AND Emenheiser, J.A.K.M.A.S.S.A.S.B.K.A.K.G.: Evolution of opinions on social networks in the presence of competing committed groups. PLoS ONE 7(3), e33215 (2012)
[43] Liu, X.T., Wu, Z.X., Zhang, L.: Impact of committed individuals on vaccination behavior. Phys. Rev. E 86(5), 051132 (2012)
[44] Turalska, M., West, B.J., Grigolini, P.: Role of committed minorities in times of crisis. Sci. Rep. 3 (2013)
[45] Lambiotte, R., Ausloos, M.: Coexistence of opposite opinions in a network with communities. Journal of Statistical Mechanics: Theory and Experiment 2007(08), P08026–P08026 (Aug 2007), 00042
[46] Ru, W., Li-Ping, C.: Opinion Dynamics on Complex Networks with Communities. Chinese Physics Letters 25(4), 1502 (Apr 2008), 00029
[47] Guo, L., Cai, X.: Bifurcation Phenomena of Opinion Dynamics in Complex Networks. In: Zhou, J. (ed.) Complex Sciences, pp. 1146–1153. No. 4 in Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, Springer Berlin Heidelberg (2009)
[48] Gargiulo, F., Huet, S.: Opinion dynamics in a group-based society. EPL (Europhysics Letters) 91(5), 58004 (Sep 2010), 00012
[49] Singh, A., Castro, M., Druschel, P., Rowstron, A.: Defending Against Eclipse Attacks on Overlay Networks. In: Proceedings of the 11th Workshop on ACM SIGOPS European Workshop. EW 11, ACM, New York, NY, USA (2004), 00171
[50] Dwork, C., Peleg, D., Pippenger, N., Upfal, E.: Fault Tolerance in Networks of Bounded Degree. In: Proceedings of the Eighteenth Annual ACM Symposium on Theory of Computing. pp. 370–379. STOC '86, ACM, New York, NY, USA (1986), 00122
[51] Upfal, E.: Tolerating Linear Number of Faults in Networks of Bounded Degree. In: Proceedings of the Eleventh Annual ACM Symposium on Principles of Distributed Computing. pp. 83–89. PODC '92, ACM, New York, NY, USA (1992)
[52] King, V., Saia, J., Sanwalani, V., Vee, E.: Towards Secure and Scalable Computation in Peer-to-Peer Networks. In: Proceedings of the 47th Annual IEEE Symposium on Foundations of Computer Science. pp. 87–98. FOCS '06, IEEE Computer Society, Washington, DC, USA (2006), 00055
[53] Fischer, M.J., Lynch, N.A., Paterson, M.S.: Impossibility of distributed consensus with one faulty process. J. ACM 32(2), 374–382 (1985)
[54] Correia, M., Veronese, G.S., Neves, N.F., Verissimo, P.: Byzantine consensus in asynchronous message-passing systems: a survey. Int. J. Crit. Comput.-Based Syst. 2(2), 141–161 (2011), http://dx.doi.org/10.1504/IJCCBS.2011.041257, 00013

[55] Chandra, T.D., Toueg, S.: Unreliable failure detectors for reliable distributed systems. J. ACM 43(2), 225–267 (1996)
[56] Mean field theory. http://en.wikipedia.org/wiki/Mean_field_theory
[57] Dabek, F., Li, J., Sit, E., Robertson, J., Kaashoek, M.F., Morris, R.: Designing a DHT for low latency and high throughput. In: Proceedings of the 1st Conference on Symposium on Networked Systems Design and Implementation - Volume 1. pp. 7–7. NSDI'04, USENIX Association, Berkeley, CA, USA (2004)
[58] Li, J., Stribling, J., Morris, R., Kaashoek, M., Gil, T.: A performance vs. cost framework for evaluating DHT design tradeoffs under churn. In: Proceedings IEEE INFOCOM 2005. 24th Annual Joint Conference of the IEEE Computer and Communications Societies. vol. 1, pp. 225–236 vol. 1 (Mar 2005)