Asynchronous Binary Byzantine Consensus over Graphs with Power-Law Degree Sequence

Goitom Weldehawaryat, Stephen Wolthusen

Chapter 17

ASYNCHRONOUS BINARY BYZANTINE CONSENSUS OVER GRAPHS WITH POWER-LAW DEGREE SEQUENCE

Goitom Weldehawaryat and Stephen Wolthusen

Abstract Consensus problems are of great interest in distributed systems research, especially in the presence of Byzantine faults. While asynchronous message passing is an interesting network model, Fischer, *et al.* [17] have shown that deterministic algorithms do not exist even for single faults, requiring the use of randomization as proposed by Ben-Or [6].

While most approaches implicitly assume full connectivity, the case of non-complete graphs is particularly interesting when studying the feasibility and efficiency of consensus problems. This topic has received limited scrutiny despite the fact that non-complete graph structures are ubiquitous in many networks that require low overall latency and reliable signaling (e.g., electrical power networks). One of the core benefits of such an approach is the ability to rely on redundant sensors in large networks for detecting faults and adversarial actions without impacting real-time behavior. It is, therefore, critical to minimize the message complexity in consensus algorithms.

This paper studies the existence and efficiency of randomized asynchronous binary Byzantine consensus for graphs in the $G(n, \vec{d})$ configuration model with a power-law degree sequence. The main contribution is an algorithm that explicitly utilizes the network structure to gain efficiency over a simple randomized algorithm while allowing the identification of possible additional edges in the graph to satisfy redundancy requirements.

Keywords: Critical infrastructures, Byzantine consensus, power-law networks

1. Introduction

The consensus problem is a fundamental problem in the domain of fault-tolerant distributed systems. It requires the system processes to agree on a common value despite the presence of some faulty processes. Fischer, *et al.* [17]

have shown that it is impossible to achieve consensus in an asynchronous distributed system that is subject to even a single crash fault. However, Ben-Or [6] has shown that a randomization approach can achieve binary consensus in an asynchronous distributed system that is subject to crash faults.

In applications involving the monitoring and control of electrical power networks, it is critical to detect faults and potential attacks such as sensor manipulations, as well as to ensure that network operations satisfy the mandatory real-time constraints. Existing results have explicitly or, more often, implicitly assumed full or random connectivity, but many large real networks (e.g., electrical power and telecommunications networks) display a scale-free nature, and are sparse and follow a power-law degree sequence. While it is possible to add edges to the graphs (e.g., adding communications links to control networks that partially coincide with electrical power networks), the cost of the additional links must be minimized.

Most critical infrastructure systems require correct interactions among large numbers of geographically-dispersed nodes (e.g., sensors and actuators) as well as at higher levels (e.g., SCADA systems). These systems cannot normally employ fail-stop semantics and must be fault-tolerant; however, robustness to targeted attacks requires a stronger model, namely Byzantine fault tolerance. Byzantine fault detection and tolerance is a known hard problem, as is the consensus problem in the presence of Byzantine faults. The ability to rapidly reach consensus is critical; in most cases, the dominant problem is message complexity.

Castro and Liskov [8], Chun, *et al.* [9] and Veronese, *et al.* [27] have proposed replication algorithms to implement highly-resilient services; some of these algorithms can be used to control services such as water, power and gas [18, 26]. Critical infrastructures require highly-resilient services that function correctly even under Byzantine faults that may corrupt some of the computers involved. Asynchronous Byzantine consensus algorithms address this problem by allowing critical services to continue to operate correctly even when system components exhibit arbitrary behavior (e.g., crashes or intrusions by attackers). Recently, the problem of solving asynchronous Byzantine consensus with $2f+1$ processes has attracted attention [9, 13]. This is possible with a hybrid system model, which extends the traditional model by incorporating a trusted/trustworthy component that constrains the power of faulty processes to exhibit certain behaviors.

One of the core benefits of such an approach is the ability to rely on redundant sensors in large networks for detecting faults and adversarial actions without degrading real-time operations. It is, therefore, critical to minimize the message complexity of a consensus algorithm that reduces latency (e.g., for management of telecommunications networks and state estimation and control in electrical power networks). This has led to the consensus problem being studied in scale-free networks by Wang, *et al.* [28] using the Barabasi-Albert model [5], which relies on a generative model. However, the preferential attachment model only produces networks with a power-law exponent of three, and

some important properties observed in large real-world networks are still missing in graphs that exhibit different exponents while still showing a power-law degree sequence. Therefore, we argue that non-complete graphs are particularly interesting when studying the feasibility and efficiency of consensus problems.

Building on our earlier work on Erdos-Renyi random graphs [29], this paper focuses on randomized asynchronous binary Byzantine consensus for graphs in the $G(n, \vec{d})$ configuration model with power-law degree sequence and presents an algorithm that achieves the desired primary result with reduced message complexity for non-complete graphs. To reach this objective, a refinement of Ben-Or's algorithm recently proposed by Correia, *et al.* [13] is considered. Their approach differs from this work in that it considers fully-connected communications networks. This paper shows that, when choosing a non-complete graph as a communications system, no additional asynchronous messaging assumptions are needed. Moreover, it is possible to increase message complexity efficiency by considering higher degree nodes that forward received messages with high probability P_{high} and lower degree nodes that forward messages with low probability P_{low}.

2. Related Work

The consensus problem is a fundamental problem in the domain of distributed systems. Fischer, *et al.* [17] proved that a deterministic algorithm cannot solve the consensus problem in an asynchronous model even in the presence of one faulty process. In the asynchronous model, each communication can take an arbitrary and unknown amount of time, and there is no assumption of a joint clock as in the synchronous model. However, Ben-Or [6] showed that a randomized algorithm can solve the consensus problem even when a constant fraction of processes are faulty. Interested readers are referred to [1, 4] for a complete proof of correctness of Ben-Or's algorithm and a detailed survey of randomized consensus protocols. Consensus in the asynchronous Byzantine message-passing model has been shown to require $n \geq 3f + 1$ processes in several variations of the basic model. Recently, Correia, *et al.* [13] showed that it is possible to solve Ben-Or's asynchronous Byzantine binary random consensus problem with $2f + 1$ processes. Consensus protocols play an important role in replication algorithms that can be utilized to protect critical infrastructures [14]. Castro and Liskov [8], Chun, *et al.* [9] and Veronese, *et al.* [27] proposed replication algorithms to implement highly-resilient services; some of these algorithms can be used to control services such as water, power and gas [18, 26].

Traditionally, the consensus problem was formulated in the context of random and fully-connected networks, although this assumption is typically not stated. Unfortunately, many large complex networks are poorly approximated by complete graphs or even simple random graphs. Many of these networks also exhibit scale-free properties. This has led to the consensus problem being studied in scale-free networks by Wang, *et al.* [28] using the Barabasi-Albert model [5], which relies on a generative model. However, the preferential attach-

ment model only produces networks with a power-law exponent of three and some important properties observed in large real-world networks are still missing in these graphs, which exhibit different exponents while showing a power-law degree sequence. This paper focuses on the randomized asynchronous binary Byzantine consensus problem for graphs in the $G(n, \vec{d})$ configuration model with power-law degree sequence.

3. Asynchronous Byzantine Consensus Problem

This section describes the Byzantine consensus problem and its assumptions.

3.1 Asynchronous Byzantine Consensus

Asynchronous Byzantine consensus algorithms are important when constructing Byzantine fault-tolerant systems. The consensus problem seeks to get a set of processes to agree on a common value. Many variants of the consensus problem have been proposed. However, this paper considers binary consensus in an asynchronous environment, where faulty processes can behave in an arbitrary manner and no assumptions are made about the relative speed of processes and the timely delivery of messages. A consensus protocol enables a system of n asynchronous processes, some of which are faulty, to agree on a value.

The consensus problem is solved when the following requirements are satisfied [14]:

- **Agreement:** All the processes choose the same value.
- **Validity:** The common output value is an input value of some process.
- **P-Termination:** Every correct process eventually decides with probability one.

Fischer, *et al.* [17] have shown that a deterministic protocol cannot guarantee agreement even against benign failures in asynchronous systems. Over the years, several techniques have been proposed to circumvent this impossibility result. One of the first approaches to solving the consensus problem was to use randomization. Existing results allow processes to reach an agreement in fully-connected networks. However, the case of non-complete graphs is particularly interesting when studying the feasibility and efficiency of consensus problems in real-world networks such as the Internet, World-Wide Web, metabolic networks and power networks with approximate structures [22], all of which have the power-law form $P(k) \sim k^{-\gamma}$.

Several models have been introduced to generate graphs with power-law distributions. This paper considers a simple generalization of the traditional random graph model called the configuration model [2, 20]. Chung and Lu [10] introduced a modified version of the configuration model, where, given a sequence $(d_1, \ldots, d_n)$, nodes v_i, v_j are connected with probability proportional to $d_i d_j$. Bollobas, *et al.* [7] also showed analytically that graphs constructed

according to the preferential attachment rule obey a power-law degree distribution with an exponent of three. The consensus problem has been studied in scale-free networks proposed by Barabasi and Albert [5]. However, the preferential attachment model only produces networks with a power-law exponent of three and some important properties observed in large real-world networks are still missing in these graphs.

The structure of electrical power transmission networks has been studied extensively in a number of countries [22]; despite differences in structure, the efforts have retained an overall power-law degree sequence. This clearly motivates our work on arbitrary power-law degree sequences because it allows the fine-tuning of message complexity and the identification of minimum additional edge requirements.

3.2 System and Network Models

This section describes the system and network models.

System Model. We consider a distributed system consisting of n processes where $n \geq 2$. The processes may be correct or faulty. Correct processes always behave according to their specifications while faulty processes exhibit arbitrary (Byzantine) behavior.

We consider a Byzantine failure model that does not impose any constraints on how processes fail for a certain fraction of network nodes in a distributed system. This (non)assumption about how processes fail is essential for systems that are exposed to malicious attacks and intrusions. However, only f out of n processes can be faulty with $n \geq 2f + 1$, where n is the number of processes in the system and f is the maximum number of faulty processes [13]. The communications network consists of communications channels used by processes to communicate via messages sending and message receiving primitives. The communication channels are reliable in that messages that sent to and received by the correct processes and are not modified by the communications medium. However, messages may be delayed and may be delivered in a different order than they were sent.

We also consider an asynchronous system in which there are no bounds on the message delays and relative speeds of processes. In such a system, it is impossible to detect missing messages; there is also no way to distinguish between a delayed message and a message that is not sent. This (non)assumption is important because attackers often violate some timing properties by launching denial-of-service attacks against processes and communications. However, we assume the existence of failure detector modules that provide hints about faulty processes. In particular, we employ muteness failure detectors, which suspect that a process is mute either because it crashed or is Byzantine and stopped sending messages according to the algorithm [16].

Network Model. Power-law (scale-free) networks are characterized by a specific structural feature of power-law degree distributions. Examples of scale-

free real-world networks include power networks, the World-Wide Web, email networks, social networks and networks of Internet routers [15]. Scale-free networks usually have nonhomogeneous topologies where the majority of the nodes have few links, but a small number of nodes have a large number of links and $P(k)$ decays according to the power-law $P(k) \sim k^{-\gamma}$ where γ is the power-law exponent [21]. Most real-world networks have the scale-free property with γ satisfying the constraint $2 < \gamma < 3$ [3]. For $2 \leq \gamma < 3$, a network with N nodes has constant or at most $O(logN)$ average degree, but the variance of the degree distribution is unbounded. It is in the regime of γ that power-law networks display many of the advantageous properties, such as small diameter, tolerance to random node deletions and a natural hierarchy where there are sufficiently many nodes of high degree.

Many models have been proposed for representing real networks. One of them is the configuration model, which creates random graphs that can have any generic degree distribution, and can, therefore, capture the degree characteristics of real-world networks. This paper uses the configuration model with a predefined degree distribution to generate static power-law networks.

We consider an undirected simple graph $G(n, d)$ consisting of N vertices with a degree sequence $\vec{d} = (d(1), d(2), \ldots d(n))$. The neighborhood of n_i is denoted by Λ_i and the degree distribution for the graph denoted by $P(k)$ is defined to be the fraction of nodes in the graph with degree k. The degree distribution can be calculated as follows [25]:

$$P(k) = \frac{|\{v|d(v) = k\}|}{N}$$

where $d(v)$ is the degree of node v and N is the number of nodes in the graph. The average degree in the graph is denoted by $\langle k \rangle \equiv \sum_k kP(k)$. The number of edges in the graph is given by $m = \langle k \rangle N/2$.

4. Reliable Broadcast in Power-Law Networks

This section investigates the performance of an efficient reliable broadcasting algorithm in the configuration model with power-law degree sequence. The performance of reliable broadcasting in power-law networks can be improved by separating nodes into two sets, each set using a different probability when selecting the neighbors to which received messages are forwarded [19]. This allows high-connectivity nodes to forward messages to their neighbors with a high probability and low-connectivity nodes to forward messages with low probability. The idea behind the algorithm is to reduce the number of redundant messages sent to high-degree nodes and, thus, reduce the overall message complexity. The algorithm has two phases. The first phase searches for high-degree node(s) while the second phase disseminates messages. During the first phase, the network is searched using short biased random walks and bond percolation, where a query message is forwarded on each edge with probability higher than the bond percolation threshold of the network.

4.1 Percolation Search for High-Degree Nodes

The percolation search technique is an efficient way of searching for high-degree node(s) in power-law networks. Power-law networks have few nodes with very high degrees. A node is considered to be highly connected if its degree is greater than or equal to half the maximum degree in the network. The percolation search algorithm leverages the power-law property by making queries reach high-degree nodes.

Cohen, *et al.* [11] have shown that, if the power-law degree distribution γ is greater than three, then the critical probability threshold for the integrity of a power-law network system being compromised is one. In other words, they showed that, for a network with γ less than three, the critical value q_c of q where the transition takes place at which a giant component forms is zero or negative (this indicates that the network always has a giant component or the network always percolates). The results ensure the connectivity of an undirected power-law network when γ is less than three.

The percolation search algorithm has three phases [23]:

- **Content Implantation:** During this phase, a node caches or implants its content in some other nodes in the network. The node does this by taking a short random walk through its nodes, starting from itself and duplicating its content at each step. The random walk has size $O(logN)$ where N is the number of nodes in the network.

- **Query Implantation:** When a node issues a query, it first executes a short random walk of size $O(logN)$ and implants its query request in the nodes visited. For a power-law graph, the random walk quickly converges towards high-degree nodes. However, choosing high-degree nodes to traverse first, improves the search. This way, the requester and all the nodes that have the query implanted in them take part in the search. If a query reaches a node that has already received the same query from another neighbor, the query is not implanted; this avoids loops in the query path.

- **Bond Percolation:** All the implanted query requests are propagated independently and in parallel using a probabilistic broadcast scheme. In this scheme, a node receiving a query message for the first time, relays the message on each of its edges with a probability q, which is vanishingly greater than the percolation threshold q_c ($q = q_c\gamma$) of the underlying power-law network [24]. The percolation probability q corresponds to the probability with which network nodes communicate a message to any of their neighbors.

4.2 Message Complexity

In a straightforward parallel search technique, each node, upon receiving a query message, forwards it to all its neighbors, unless a node has already received the query message. This leads to $O(ln(ln\,N))$ total messages for every

Algorithm 1 : Reliable Broadcast Algorithm (Node n_i).

P_{high}: forwarding probability for high-degree nodes
P_{low}: forwarding probability for low-degree nodes
d: degree threshold

Function RELIABLE BROADCAST(*id,msg*)
Task T1:
$\sigma \leftarrow sign_j(id, msg)$
$\forall_j \neq i$: SEND INITIAL(*i,id,msg*)$_\sigma$ to n_j
DELIVER(*i,id,msg*)
Task T2: {execute only once per message broadcast}
while (message INITIAL(*j,id,msg*$_\sigma$ is received) and (verify(*id,msg*$_\sigma$,K_{uj}))) **do**
  if $V_i > d$ **then**
    if $random() \leq P_{high}$ **then**
      for $n_j \in \Lambda_i$ **do**
        SEND (*j,id,msg*,σ) to n_j
        DELIVER(*j,id,msg*)
      end for
    end if
  else
    if $random() \leq P_{low}$ **then**
      for $n_j \in \Lambda_i$ **do**
        SEND (*j,id,msg*,σ) to n_j
        DELIVER(*j,id,msg*)
      end for
    end if
  end if
end while

query. It has been proven that, when $2 < \gamma < 3$, the diameter of the network $d \sim ln(ln\, N)$ is smaller than small real-world networks ($O(ln\, N)$) and remains almost constant while the network is growing [12]. The bond percolation step guarantees that a query message is received by nodes in a high-connected component of diameter $O(log\, N)$ and consisting of high-degree nodes. The content and query implantation steps ensure that the content/message of a node are cached in at least one of the nodes in this high-degree connected component with probability approaching one, and that one of the nodes in the connected component receives a query implantation with probability approaching one.

When a node issues a query message, each edge passes it with probability q. Thus, with $qE = q_c\langle k\rangle N/\gamma$ total number of messages, any content/high-degree node can be located with probability approaching one in time $O(log\, N)$. After the first phase, the second phase of Algorithm 1 starts message dissemination using the hub node(s). Upon receiving this message and comparing the degree of a node with the degree threshold (d), a high-degree node forwards the received message msg with a high probability P_{high} or a low-degree node forwards it with a low probability P_{low} where $P_{high} > P_{low}$. At each step, the

Algorithm 2 : Byzantine Consensus Algorithm (Process p_i).

$est_i \rightarrow v_i$ {current estimate of the value to be decided}
Step 0: $r_i \rightarrow 1$ {round number}
Step 1: RELIABLE_BROADCAST PHASE1(r_i, est_i)
Step 2: wait until (valid messages PHASE1(r_i, -) are received at least $n - f$ processes) and ($\forall_j$: valid message PHASE1(r_i, -) is received from p_j or p_j is suspected by p_i's FD module)
if (more than $n/2$ messages have the same value) **then**
    RELIABLE_BROADCAST PHASE2(r_i, v,*decision*)
else
    RELIABLE_BROADCAST PHASE2(r_i, $\perp$)
end if
Step 3: wait until (valid messages PHASE1(r_i, -) are received at least $n - f$ processes) and ($\forall_j$: valid message PHASE2(r_i, -) is received from p_j or p_j is suspected by p_i's FD module)
if (there are $n - f$ decision messages PHASE2(r_i, v, *decision*)) **then**
    DECIDE(v)
else if (there is one decision message PHASE2(r_i, v,*decision*)) **then**
    $est_i \rightarrow v$
else
    $est_i \rightarrow 1$ *or* 0 with probability 1/2
end if
    Step 4: $r_i \rightarrow r_i + 1$ **go to Step 1**

message is forwarded from n_j to $P_{high}/P_{low}|\Lambda_i|$ other nodes. In a scale-free network, a given node on the average propagates a message to $P_{high}/P_{low}\langle k\rangle$ nodes. The decision about the degree of a node (high or low) depends on a threshold degree d. By changing the probability values, it is possible to control the effective connectivity of the network while information is forwarded. The message complexity of the algorithm can be considerably reduced because only high-degree nodes (e.g., hubs that are few in number in a power-law network) are responsible for message forwarding with high probability. This differs from normal broadcasting algorithms in which all the periphery nodes also forward messages.

5. Consensus Algorithm

This section describes the consensus algorithm and discusses its key features.

5.1 Consensus Protocol

In the consensus problem, each process begins with an initial value $v_i \in \{0, 1\}$ and all the correct processes must decide on one of the proposed values v. This section presents a binary consensus protocol (Algorithm 2) adapted from the Correia, *et al.* variant of Ben-Or's algorithm. The protocol uses the under-

lying reliable broadcast over power-law networks as the basic communication primitive (Section 4).

The protocol operates in rounds, where each round has two stages. In the first stage of a round, each process p_i reliably broadcasts its current estimate v_i using the high-degree nodes and waits to receive $n - f$ valid messages. If a process receives a strict majority of reports for the same value v, then it proposes v to all the processes; otherwise, it proposes $\perp$.

In the second stage of a round, p_i broadcasts v using high-degree nodes to each destination process, waits for $n - f$ valid messages PHASE2, and then decides on v if there are $n - f$ decision messages PHASE2. If there is one decision message PHASE2 with value v different from $\perp$, then p_i adopts v as its new estimate and a new round is initiated. Otherwise, p_i adopts a random bit (0 or 1) with probability 1/2 for the estimate and a new round is initiated.

5.2 Proof of Correctness

The following is a brief proof of correctness of the consensus algorithm:

- **Agreement:** If a correct process decides v in round r, then no correct process decides $v' \neq v$ in round $r' \geq r$. A correct process p decides the value v in round r if and only if it receives $(n-f)p$ valid decision messages PHASE2 $(r_i, v, decision)$.
- **Validity:** If some process p decides v, then v is the initial value of some process.
- **Termination:** Every correct process eventually decides. It is necessary to prove that the algorithm does not block indefinitely at some point. The only points where a process blocks are where it waits for messages, so it is only necessary to prove that the process does progress and eventually terminates.

Interested readers are referred to [29] for a detailed proof of correctness of an algorithm that has the same basic structure.

5.3 Message Complexity Analysis

This section compares the message complexity of the Correia, *et al.* variant of Ben-Or's algorithm over non-complete graphs using the $G(n, \vec{d})$ configuration model with the original Correia, *et al.* variant of Ben-Or's algorithm, which assumes a flooding algorithm in which every node of a power-law network forwards its first-time received message once to its one-hop neighborhood.

First, we assume that correct processes do not suspect correct processes, i.e., all correct processes receive all the messages from each other. In both algorithms, messages are broadcast in both phases of each round. However, in the original Correia, *et al.* variant of Ben-Or's algorithm, messages are sent to every other process in the system, resulting in $\langle k \rangle N/2$ messages during each phase. On the other hand, in the modified Correia, *et al.* variant of Ben-Or's

algorithm, a high-degree node forwards a received message msg with a high probability P_{high} and a low-degree node forwards it with a low probability P_{low}, resulting in a total of $P_{high}\langle k\rangle + P_{low}\langle k\rangle$ messages during each phase.

In both algorithms, there is eventually a round r in which all the correct processes set est_i to the same value v either in Line 13 or in Lines 11 and 13 of the algorithms. When this occurs, in round $r+1$, all the correct processes broadcast v in Line 3, all the processes receive at least $n-f$ PHASE1 messages with the value (since there are at least that many correct processes), and all the correct processes broadcast PHASE2(r+1, v, $decision$) messages, and all receive each other's PHASE2 messages in Line 10. Since in round $r+1$ all the correct processes broadcast v in Line 3 and PHASE2($r+1, v, decision$), the original Correia, *et al.* variant of Ben-Or's algorithm sends $\langle k\rangle N^2/2$ messages over the entire graph. In contrast, the modified Correia, *et al.* variant of Ben-Or's algorithm sends only $P_{high}N\langle k\rangle + P_{low}N\langle k\rangle$ messages over the non-complete graph.

It is important to also consider the message complexity of the percolation search for high-degree nodes during network setup in the modified Correia, *et al.* variant of Ben-Or's algorithm. Thus, with $qE = q_c\langle k\rangle N/\gamma$ total messages, any content/high-degree node can be located with probability approaching one in time $O(logN)$. Putting everything together, if no process is suspected by the eventually-perfect muteness failure detector, then the original Correia, *et al.* variant of Ben-Or's algorithm requires $\langle k\rangle N/2$ messages and the modified Correia, *et al.* variant of Ben-Or's algorithm requires $P_{high}\langle k\rangle + P_{low}\langle k\rangle + q_c\langle k\rangle N/\gamma$ messages. In the latter case, the original Correia, *et al.* variant of Ben-Or's algorithm requires $\langle k\rangle N^2/2$ messages and the modified Correia, *et al.* variant of Ben-Or's algorithm requires $P_{high}N\langle k\rangle + P_{low}N\langle k\rangle$ messages. Since $O(P_{high}\langle k\rangle + P_{low}\langle k\rangle + q_c\langle k\rangle N/\gamma) \ll \langle k\rangle N/2$ or $O(P_{high}N\langle k\rangle + P_{low}N\langle k\rangle + q_c\langle k\rangle N/\gamma) \ll O(\langle k\rangle N^2/2)$, there is a significant reduction in message complexity in the modified Correia, *et al.* variant of Ben-Or's algorithm.

6. Conclusions

This paper has studied the existence and efficiency of randomized asynchronous binary Byzantine consensus for graphs in the $G(n, \vec{d})$ configuration model with power-law degree sequence. A key result is that it is possible to reduce the message complexity in non-complete random graphs using high-degree nodes to forward messages with high probability and low-degree nodes to forward messages with low probability. Additionally, the modified Correia, *et al.* variant of Ben-Or's algorithm over non-complete graphs using the $G(n, \vec{d})$ configuration model with power-law degree sequence yields the desired primary result. Specifically, it is possible to solve the asynchronous Byzantine binary consensus problem with $2f+1$ processes over non-complete graphs using the $G(n, \vec{d})$ configuration model with power-law degree sequence by employing a reliable broadcast algorithm (that requires a wormhole component, although this has a considerably lower cost than increasing the density of the graph) and an

eventually-perfect muteness failure detector without any additional asynchrony assumptions.

The message complexity analysis shows a significant reduction in message complexity in the modified Correia, *et al.* variant of Ben-Or's algorithm over non-complete graphs using the $G(n, \vec{d})$ configuration model with power-law degree sequence. This occurs when high-degree nodes forward messages with a high probability and low-degree nodes forward messages with a low probability. Because the number of the low-degree nodes is much higher than the number of high-degree nodes in a scale-free network, the message complexity is considerably less than that for a flooding algorithm. The significantly lower message complexity for the consensus algorithm reduces latency during network management in telecommunications networks and state estimation and control in electrical power networks.

Our future research will investigate the properties of multiple power-law graphs and efficient consensus algorithms over these graphs. These graphs are commonly encountered in telecommunications networks and electrical power networks, which are nominally distinct, but frequently interconnected.

References

[1] M. Aguilera and S. Toueg, The correctness proof of Ben-Or's randomized consensus algorithm, *Distributed Computing*, vol. 25(5), pp. 371–381, 2012.

[2] W. Aiello, F. Chung and L. Lu, A random graph model for power-law graphs, *Experimental Mathematics*, vol. 10(1), pp. 53–66, 2001.

[3] R. Albert and A. Barabasi, Statistical mechanics of complex networks, *Reviews of Modern Physics*, vol. 74(1), pp. 47–97, 2002.

[4] J. Aspnes, Randomized protocols for asynchronous consensus, *Distributed Computing*, vol. 16(2/3), pp. 165–175, 2003.

[5] A. Barabasi and R. Albert, Emergence of scaling in random networks, *Science*, vol. 286(5439), pp. 509–512, 1999.

[6] M. Ben-Or, Another advantage of free choice: Completely asynchronous agreement protocols, *Proceedings of the Second Annual ACM Symposium on Principles of Distributed Computing*, pp. 27–30, 1983.

[7] B. Bollobas, O. Riordan, J. Spencer and G. Tusnady, The degree sequence of a scale-free random graph process, *Random Structures and Algorithms*, vol. 18(3), pp. 279–290, 2001.

[8] M. Castro and B. Liskov, Practical Byzantine fault tolerance and proactive recovery, *ACM Transactions on Computer Systems*, vol. 20(4), pp. 398–461, 2002.

[9] B. Chun, P. Maniatis, S. Shenker and J. Kubiatowicz, Attested append-only memory: Making adversaries stick to their word, *ACM SIGOPS Operating Systems Review*, vol. 41(6), pp. 189–207, 2007.

[10] F. Chung and L. Lu, *Complex Graphs and Networks*, American Mathematical Society, Providence, Rhode Island, 2006.

[11] R. Cohen, K. Erez, D. ben-Avraham and S. Havlin, Resilience of the Internet to random breakdowns, *Physical Review Letters*, vol. 85(21), pp. 4626–4628, 2000.

[12] R. Cohen and S. Havlin, Scale-free networks are ultrasmall, *Physical Review Letters*, vol. 90(5), pp. 058701-1–4, 2003.

[13] M. Correia, G. Veronese and C. Lung, Asynchronous Byzantine consensus with $2f + 1$ processes, *Proceedings of the ACM Symposium on Applied Computing*, pp. 475–480, 2010.

[14] M. Correia, G. Veronese, N. Neves and P. Verissimo, Byzantine consensus in asynchronous message passing systems: A survey, *International Journal of Critical Computer-Based Systems*, vol. 2(2), pp. 141–161, 2011.

[15] P. Crucitti, V. Latora, M. Marchiori and A. Rapisarda, Efficiency of scale-free networks: Error and attack tolerance, *Physica A: Statistical Mechanics and its Applications*, vol. 320, pp. 622–642, 2003.

[16] A. Doudou, B. Garbinato and R. Guerraoui, Tolerating arbitrary failures with state machine replication, in *Dependable Computing Systems: Paradigms, Performance Issues and Applications*, H. Diab and A. Zomaya (Eds.), Wiley-Interscience, Hoboken, New Jersey, pp. 27–56. 2005.

[17] M. Fischer, N. Lynch and M. Paterson, Impossibility of distributed consensus with one faulty process, *Journal of the ACM*, vol. 32(2), pp. 374–382, 1985.

[18] C. Hauser, D. Bakken, I. Dionysiou, K. Gjermundrod, V. Irava, J. Helkey and A. Bose, Security, trust and QoS in next-generation control and communication for large power systems, *International Journal of Critical Infrastructures*, vol. 4(1/2), pp. 3–16, 2008.

[19] R. Hu, J. Sopena, L. Arantes, P. Sens and I. Demeure, Efficient dissemination algorithm for scale-free topologies, *Proceedings of the Forty-Second International Conference on Parallel Processing*, pp. 310–319, 2013.

[20] M. Molloy and B. Reed, A critical point for random graphs with a given degree sequence, *Random Structures and Algorithms*, vol. 6(2/3), pp. 161–179, 1995.

[21] M. Newman, Power laws, Pareto distributions and Zipf's law, *Contemporary Physics*, vol. 46, pp. 323–351, 2005.

[22] G. Pagani and M. Aiello, The power grid as a complex network: A survey, *Physica A: Statistical Mechanics and its Applications*, vol. 392(11), pp. 2688–2700, 2013.

[23] N. Sarshar, P. Boykin and V. Roychowdhury, Percolation search in power-law networks: Making unstructured peer-to-peer networks scalable, *Proceedings of the Fourth International Conference on Peer-to-Peer Computing*, pp. 2–9, 2004.

[24] N. Sarshar, P. Boykin and V. Roychowdhury, Finite Percolation at a Multiple of the Threshold (`arxiv.org/pdf/cond-mat/0601211v2.pdf`), 2006.

[25] R. Sharan, Analysis of Biological Networks: Random Models, School of Computer Science, Tel-Aviv University, Tel-Aviv, Israel, 2009.

[26] P. Verissimo, N. Neves, M. Correia, Y. Deswarte, A. Abou Kalam, A. Bondavalli and A. Daidone, The CRUTIAL architecture for critical information infrastructures, in *Architecting Dependable Systems V*, R. Lemos, F. Giandomenico, C. Gacek, H. Muccini and M. Viera (Eds.), Springer-Verlag, Berlin Heidelberg, Germany, pp. 1–27, 2008.

[27] G. Veronese, M. Correia, A. Bessani and L. Lung, Highly-resilient services for critical infrastructures, *Proceedings of the Workshop on Embedded Systems and Communications Security*, 2009.

[28] S. Wang, K. Yan and M. Chiang, Optimal agreement in a scale-free network environment, *Informatica*, vol. 17(1), pp. 137–150, 2006.

[29] G. Weldehawaryat and S. Wolthusen, Asynchronous message-passing binary consensus over non-complete graphs, *Proceedings of the Second IEEE Workshop on Network Science*, pp. 9–15, 2013.