Hindawi
Complexity
Volume 2017, Article ID 4832740, 11 pages
https://doi.org/10.1155/2017/4832740

WILEY | Hindawi

Research Article

Distributed Sequential Consensus in Networks: Analysis of Partially Connected Blockchains with Uncertainty

Francisco Prieto-Castrillo,[1,2,3] Sergii Kushch,[3] and Juan Manuel Corchado[3]

[1]*Media Laboratory, Massachusetts Institute of Technology, Cambridge, MA 02139-4307, USA*
[2]*Harvard T. H. Chan School of Public Health, Harvard University, Boston, MA 02115, USA*
[3]*BISITE Research Group, University of Salamanca, Edificio Multiusos I+D+i, 37008 Salamanca, Spain*

Correspondence should be addressed to Francisco Prieto-Castrillo; fprieto@mit.edu

Received 9 July 2017; Revised 13 September 2017; Accepted 10 October 2017; Published 1 November 2017

Academic Editor: Dimitri Volchenkov

This work presents a theoretical and numerical analysis of the conditions under which distributed sequential consensus is possible when the state of a portion of nodes in a network is perturbed. Specifically, it examines the consensus level of partially connected blockchains under failure/attack events. To this end, we developed stochastic models for both verification probability once an error is detected and network breakdown when consensus is not possible. Through a mean field approximation for network degree we derive analytical solutions for the average network consensus in the large graph size thermodynamic limit. The resulting expressions allow us to derive connectivity thresholds above which networks can tolerate an attack.

1. Introduction

Trust is usually conceived as the additive aggregation of reliable pieces. However, when it comes to cyber-security or privacy requirements, the challenge is how to collaboratively create trust out of uncertain sources in a networked environment [1–6]. A remarkable success story of this approach is Bitcoin [7]. In Bitcoin, trust is built by a set of agents—*miners*—which collaborates in sequencing *blocks* of transactions in a chain. *Blockchain* (BC) is the underpinning technology of Bitcoin, a protocol in which miners compete to solve a computationally expensive problem, known as *Proof-of-Work* (POW) [8]. The miners' results are then assembled together in a distributed data chain. The outcomes are only embedded in the final version of the chain after consensus, which is only reached if the order relationships are consistent. POW is a proxy of trust and, hence, reliability increases as the chain grows; it is incrementally more difficult to revert—hack—the chain since this requires increasing computing power. Thus, although each agent generates insecure information locally, the resulting aggregate becomes more and more reliable over time.

Recently however, these advantages have also caused concerns about how the BC paradigm can be exported to domains other than cryptocurrency, such as the Internet-of-Things (IoT) or Wireless Sensor Networks (WSN) [9, 10]. This difficulty arises from the limitations of the BC architecture, which hamper the possibility of extending it to small devices (e.g., sensors). Sensors, in particular, lack the computing power to perform POW. An even more challenging fact is that BC requires full connectivity to operate (which is unfeasible for WSNs). Therefore, the question at issue is how to design blockchains without POW and partial connectivity while maintaining robustness to failures and attacks.

Distributed consistency is not a novel concept. In [11] the authors analyse the consistency of distributed databases by using algorithms which are closely related to epidemiological models [12]. Two information diffusion mechanisms, antientropy and rumor mongering, happen to be particularly useful for modelling distributed consistency. Antientropy regularises entries in the databases while rumor mongering updates the last information content from neighbour instances. This trade-off between ordered and random infection allow the authors to find exponential epidemic growth

by using a mean field approach. The concept of diffusion in partially connected networks is treated rigorously in [13] in the context of glassy relaxation. Here, the geometrical aspects of the return probability of a Markovian hypercube walk are also analysed using mean field theory.

The effect of graph topology on information spreading has been extensively discussed in the literature (e.g., [14–16]). However, the model in [16] (a random graph superposed to a ring lattice) is particularly relevant to our discussion, since it ensures a minimum connectivity while maintaining the *small-world* property (i.e., high clustering coefficient and small characteristic path length [17]).

In [18] the general distributed consensus problem is described; n nonfailing sites out of m choices have to decide on a common value v. The authors of that study found that the key components for consensus breakdown are asynchronicity and failure, which both inject uncertainty into the system at different scales. Distributed consensus in networks is also analysed in [19], where the authors address the most important applications of the concept, such as clock synchronisation in WSNs. The authors introduce the average consensus as the limit to which initial states converge, provided this limit is equal to the averaged initial values. Interestingly, a randomised consensus protocol (where only a fraction of sites needs to agree on a value) is shown to be more robust against crash than a deterministic algorithm [20].

When consensus is not reached, systems usually break down. From the point of view of control theory, a number of interesting results have been obtained in studies focused on this issue, for example, [19], aimed at self-healing the system momentously after failure. However, security and resilience are multidimensional objects which can be tackled more consistently through a complex systems approach [21, 22]. For instance, [23] proposes a phone call model where n players broadcast rumors randomly among their partners. The authors study the effect of node failure and concentrate on an interesting result; if failure patterns are random, F crashing nodes result in only $O(F)$ uninformed players with high probability. The work also shows that any randomised rumor spreading algorithm running for $O(\log n)$ rounds requires $O(n)$ transmissions. This is consistent with what we know from network science [24]; random failures do not spread so easily. The model considered in [25] consists of n sites running processes asynchronously where failures are modelled as a Bernoulli process. In [26] the problem is set in terms of a voter model and an invasion process; agreed values are exported from a set of sites but imported errors infect the rest of nodes.

When it comes to blockchain implementations, [27] analyses information propagation in the Bitcoin network. This work highlights the limitations of the synchronisation mechanisms in BC and the system's weaknesses under attack. Here, the communication network is modelled as a random graph with a mean degree of ≈32 and it is found that the block verification process can majorly contribute to delay propagation and inconsistency. In their experiments the authors show that the probability distribution of the rate at which nodes learn about a block has a long tail. This means that there is a nonnegligible portion of nodes which does not receive information timely. The effect is equivalent to considering an incomplete consensus network. A typical example of organised attack in the BC is the so-called *selfish-mine* strategy. This consists of a subset of nodes which diffuse information partially to targets, instead of distributing updates homogeneously [28]. In [29] a Markov chain model is used to analyse the selfish-mine strategy in Bitcoin. This and other block-withholding behaviour can have a devastating effect on the performance if the dishonest community is around half the size of the network.

All these works provide key insights into the problem of network resilience, diffusion, and consensus from different perspectives. However, to the authors' knowledge, a mathematical model of partially connected blockchains is still missing. Therefore, in this paper we make a theoretical and numerical analysis of the conditions under which a distributed sequential consensus is possible. In concrete, we examine the consensus level of partially connected blockchains under failure/attack events. To this end, we develop stochastic models for both verification probability once an error is detected and network breakdown when consensus is not possible. The resulting expressions allow us to derive connectivity thresholds above which networks can tolerate attack.

The paper is organised as follows. In Section 2 we formulate the problem. The results obtained in the study are presented in Section 3. Finally, we present the conclusions obtained from our research and discuss the possibilities for future work in Section 5.

2. Problem Formulation

Blockchains can be conceived as dynamical distributed databases whose constituents (blocks) are collaboratively and incrementally built by a set of agents. There are three key factors in this process: (a) how information spreads, (b) how consensus can be achieved, and (c) how errors affect the overall performance. We elaborate on these elements below.

2.1. Partial Connectivity in Consensus Networks. From a network perspective we consider a *Peer-to-Peer* (P2P) infrastructure with two types of nodes: communication sites and processing sites, miners (Figure 1). Users connected to nodes can launch transactions to other users in the network. If a group of users $\{1, 2, 3, 4\}$ is involved in a transaction arrangement, one or more miners can attempt to verify the intended transactions and if successful, pack them into a block. This problem can be conceived as the interplay of three graphs: communication, transactions, and miners. As stressed, the usual BC protocol takes the full graph for granted, which is not always possible; there may either be failures or intentional attacks on a portion of the network. However, it is unlikely for a network to get disconnected under normal operation. Hence, graph connectedness is a reasonable lower bound assumption (particularly in the case of sensor networks and IoT). This leads us to consider the network proposed in [16]; $\widetilde{G} = \mathscr{R} \cup G$, consisting of a random graph G superposed to a ring lattice $\mathscr{R}$. This model still exhibits the *small-world* property found in [14, 15] but it is closer to the real

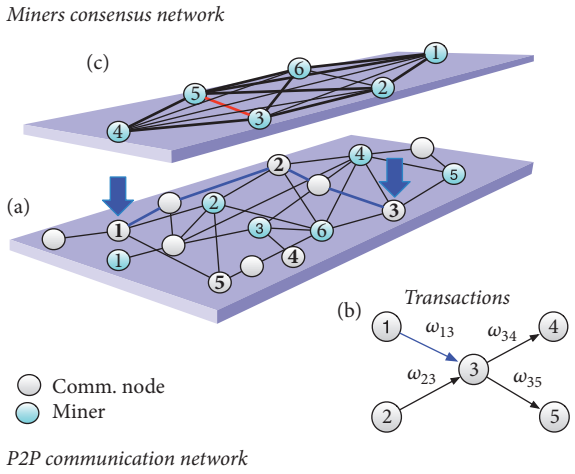


Figure 1: Schematic of nodes, transactions, and layers in the blockchain problem. A subset of the communication P2P nodes (a) are sites responsible for block processing, miners (light blue circles). When a user intends to make a transaction (b) to another user (dark blue lines) with weight ω_{13}, the miner consensus network (c) needs to reach a consensus. However, the full connectivity of the miners' graph is not guaranteed as there can be attacks (red line) or failures (thin black lines).

requirements of minimum connectivity found in WSNs and other networked systems such as computer networks [30].

At this point it seems that information spreading in the BC can be reduced to the well-known problem of diffusion on graphs. This area is vastly covered in the literature (see, e.g., [13]). However, BC diffusion holds some subtleties under the hood as we show below.

2.2. Why Order Matters: Sequential Diffusion. At every transaction arrangement, the ordering of each operation in the set is a key factor. Consider the simple arrangement shown in Figure 1(b), which represents four possible transactions. These can be formalised as the directed links $E = \{e_{13}, e_{34}, e_{23}, e_{35}\}$ shown in the graph. There are $|E|!$ ways to sort this set but not all of them are consistent. The type of consistency we refer to is that which avoids the *double-spending problem*. Take two possible order relationships $\preceq_1$ and $\preceq_2$ implemented by the bijections $T_{1,2} : E \to \mathbb{N}$. They result in

$$\begin{aligned}
(E, \preceq_1): \quad & T_1(e_{13}) = 1, \\
& T_1(e_{34}) = 2, \\
& T_1(e_{23}) = 3, \\
& T_1(e_{35}) = 4, \\
(E, \preceq_2): \quad & T_2(e_{13}) = 1, \\
& T_2(e_{34}) = 2, \\
& T_2(e_{35}) = 3, \\
& T_2(e_{23}) = 4.
\end{aligned} \tag{1}$$

Table 1: Evolution of states in the transaction graph shown in Figure 1(b) obtained by iterating (1) $n = 1, \ldots, 4$ steps for $\preceq_2$ (a) and $\preceq_2$ (b) orderings. The initial state is $S(0) = (1, 1, 0, 0, 0)$ and $\omega(n) = 1$, $\forall n$. The order relationship $\preceq_2$ induces the double spending effect.

(a)

n	s_1	s_2	s_3	s_4	s_5
0	1	1	0	0	0
1	0	1	1	0	0
2	0	1	0	1	0
3	0	0	1	1	0
4	0	0	0	1	1

(b)

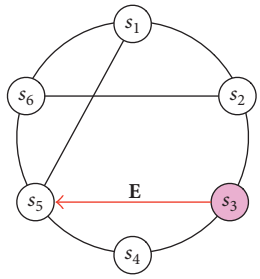
n	s_1	s_2	s_3	s_4	s_5
0	1	1	0	0	0
1	0	1	1	0	0
2	0	1	0	1	0
3	0	1	−1	1	1
4	0	0	0	1	1

The first ordering does not induce any inconsistency but the ordered set $(E, \preceq_2)$ violates the double spend constraint depending on the weights ω_{ij}. If we label by $S(n) = (s_1(n), s_2(n), s_3(n), s_4(n))$ the state vector at step n, a transition, say e_{13}, results in the update equation: $\Delta S_{13} = -\omega_{13} u_1 L_{13}$ where u_1 represents the row-base vector $(1, 0, 0, 0)$ and L_{ij} is the graph Laplacian corresponding to the transaction subgraph $g_{ij} = (\{i, j\}, e_{ij})$. The ordering allows writing compact update equations as

$$S(n+1) = S(n) - \omega(n) X(n) L(n), \tag{2}$$

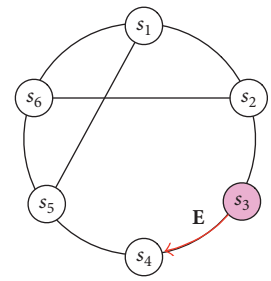
where $\omega(n)$, $X(n)$, and $L(n)$ represent transaction weights, base vectors, and graph Laplacians for each transaction. In Table 1, we show the evolution of states in the case $\omega(n) = 1$, $\forall n$ with initial state $S(0) = (1, 1, 0, 0, 0)$ for $\preceq_1$ and $\preceq_2$. Notice that for $\preceq_2$ node 3 has ran out of values at step 2 but it still intends to perform a transaction to node 5 at step 3. This is like having a balance of \$10 in a bank account and spending it twice by sending \$10 to two different recipients. When it comes to measurements in WSNs (say energy consumption data) avoiding these inconsistencies is imperative [31]. If a miner attempted to pack these transactions along with $\preceq_2$ into a block, he will reach an inconsistency. These order constraints make the BC diffusion different to regular graph diffusion [13]. In fact, BC protocol ensures that double-spending paradoxes cannot occur by imposing constraints such as $s_k(n) \geq 0$, $\forall k, n$. An example of this is the distributed ledger in Bitcoin [27]. The next question is how this ordering couples with failures in the network.

2.3. Attack and Failure in Consensus Dynamics. Blockchain technology copes with the above restrictions efficiently by elevating the transaction order relationships to the block scale. Thus, every block (which can hold one or more transactions) in the resulting blockchain builds on top of



n_i	i						
	0	1	2	3	4	5	6
1	0	1	2	2	2	5	6
2	0	1	2	3	3	3	6
3	0	0	2	3	4	5	5
4	**0**	**0**	**0**	**3**	**4**	**5**	**5**
5	0	1	1	*E*	4	5	6
6	0	1	2	2	2	5	6

(a)



n_i	i							
	0	1	2	3	3^*	4	5	6
1	0	1	2	2	2	2	5	6
2	0	1	2	3	3	3	3	6
3	0	0	2	3	3	4	4	4
4	**0**	**0**	**0**	*E*	1	4	**5**	**5**
5	0	1	1	1	1	4	5	5
6	0	1	2	2	2	2	5	6

(b)

FIGURE 2: Two ways for error propagation in the miners consensus network: to a nonneighboring node (a) and to the next block solver (b). The tables summarise the blockchain dynamics in a cycle. Rows represent sites and columns represent iterations. In the first case, the error (represented as E) cannot be restored and it persists in the blockchain. In the second case, an additional recovery step 3^* can restore the error to the agreed value of 1.

the preceding block to preserve sequential diffusion. This strategy can however be used by dishonest users to create massive damage in the network. Consider the case depicted in Figure 2 where 6 miners collaborate to build a blockchain. Without loss of generality we can label the miners according to the order of their block resolution (it is very unlikely that two miners solve a block at the same time and, if this happens, BC would still sort the resulting blocks in order with high probability [20]). Node 3 is a *failure node*; it sends an error/attack to either a nonneighboring node (a) or to a miner who happens to be the one solving the next block (b). Below each graph, we also show the evolution of the chain. In this schematic, rows represent sites and columns represent iterations within the cycle. A row stands for the local instance of the chain at a given site and a column represents the collective blockchain built up to a given step. The blockchain is constructed as follows. At step 0 all sites own the 0-*genesis block*. At step i if miner n_i finds no error in the last block of his local instance of the chain he solves the next block and broadcasts the solution to neighbours. The nonreached sites simply replicate their state. However, if the sending site is a failure node, it will broadcast a failure to one of his neighbours. In this case, if the affected block finds the error in his solving step, he still has a chance to restore the block upon consensus from his acquaintances. In case this consensus is not possible the blockchain breaks down. This flow is depicted in Figure 3.

Both situations shown in Figure 2 trigger different phenomena and have different effects in the overall network performance. In the first case, the error (represented as E in the table) has no chance of being restored and it persists in the blockchain. However, in the second case an additional recovery step 3^* can restore the error to the agreed value of 1. Notice also that since the network is not fully connected there are sites that lack state updates and their local instances of the chain are not synchronised. This limits the information spreading in the network as we show in the next section.

We highlight the fact that, in the Bitcoin implementation, miners asynchronously relay blocks and transactions as soon as they either receive or mine them [32]. In our case agents hold received blocks and transmit their knowledge to neighbours sequentially. In Figure 4 we compare the sequence diagrams for both models in the case of three miners (for the sake of simplicity we have only considered one thread per miner. Since mining times are much larger than relay times, splitting mining and relay processes in two threads would not affect the conclusions of this comparison). Without loss of generality miners S_{1-3} will solve blocks b_{1-3} in first, second, and third order. In the Bitcoin blockchain implementation (a) the processes of mining and the relay of blocks have different timescales; ≈10 minutes for mining and a few seconds for block forwarding. However in a context where POW is absent (b), the mining lags tend to zero and the processes of mining, verification, and relay converge. In (a) if site S_1 at time t_1 sends a block b_1 to S_2, this miner will forward it to S_3 after a short verification lag t_2. Then, S_2 will release b_2 after a big mining delay. However, in (b), since there is no POW, S_2 will broadcast b_2 to neighbours pretty soon at epoch t_2. This enables saving time and reducing the network traffic considerably.

2.4. Mathematical Model. By putting all these facts together, we obtain a minimum blockchain model that captures the dynamics described above: (a) partial connectivity, (b) sequential diffusion, and (c) failure spreading. Below we develop a stochastic process analysis to examine the averaged network performance under different conditions.

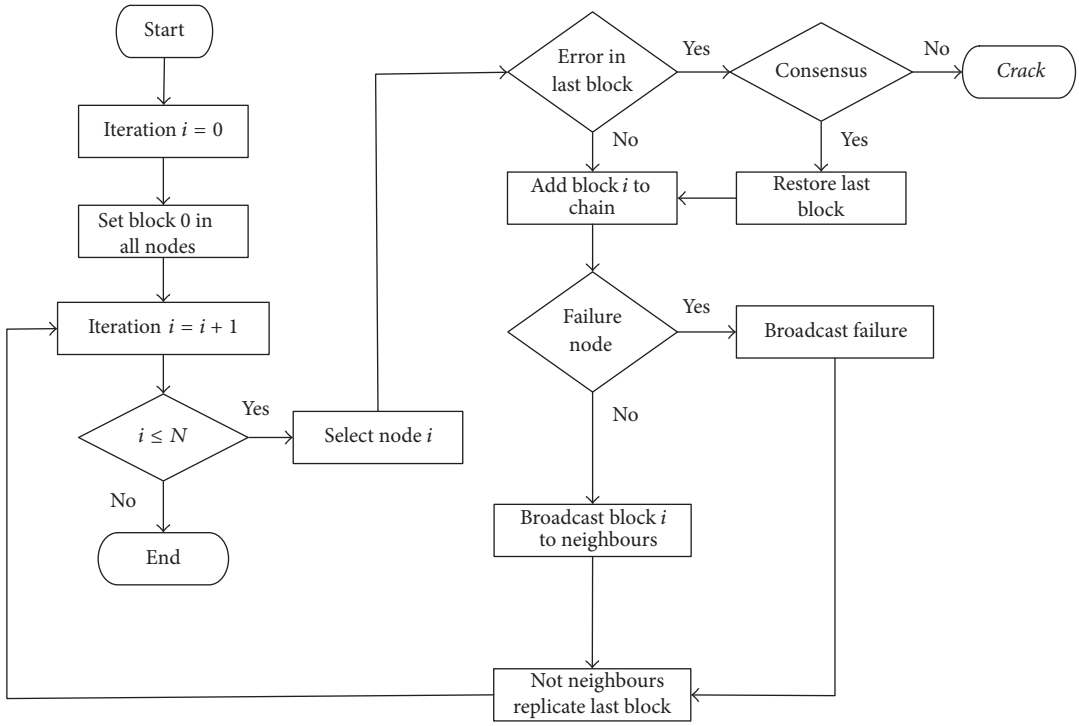


Figure 3: Blockchain dynamics workflow. At step 0 all sites own the 0-*genesis block*. At step i if miner n_i finds no error in the last block of his local instance of the chain he solves the next block and broadcasts the solution to neighbours. The nonreached sites simply replicate their state. However, if the sending site is a failure node, it will broadcast a failure to one of his neighbours. In this case, if the affected block finds the error in his solving step, he still has a chance to restore the block upon consensus from his acquaintances. In case this consensus is not possible the blockchain collapses.

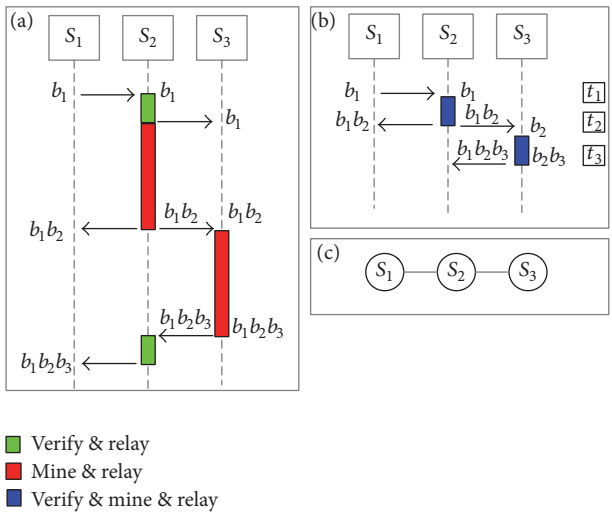


Figure 4: Sequence diagram comparison between Bitcoin blockchain (a) and the sequential model proposed in this work (b) for a simple miners network (c). In (a) if site S_1 at time t_1 sends a block b_1 to S_2, this miner will forward it to S_3 after a short verification lag t_2. Then, S_2 will release b_2 after a much larger mining delay. In (b) however, since there is no POW, S_2 will broadcast b_2 to neighbours pretty soon at epoch t_2.

With the graph model of size N described in Section 2.1 we represent each information block (or measure state in general) at site n_α at the ith iteration as $s_\alpha(i)$. As stressed above, all sites start from the 0-genesis block: $s_\alpha(0) = 0,\ \forall\alpha$. Then, following the flow depicted in Figure 3, at iteration i node n_i, checks its state and adds a block to the chain. We collect the number of sites matching the current block in the variable X_i, which is equal to the node degree k_i plus a noise term $\sigma_i \in \{0, 1\}$. If n_i sends an error signal to n_{i+1} which cannot be reverted to the state $s_{i+1}(i) = i$, then $\sigma_i = 0$ and $\sigma_i = 1$ in any other case. The performance ratio per iteration $m_i = X_i/N = (k_i + \sigma_i)/N$ is a measure of the consensus level reached at step i. Depending on whether consensus is reached or not the whole chain may collapse. In an ensemble of chains Ω we define both the failure and matching random variables $F : \Omega \to \{0, 1\}$, $M : \Omega \to R$, and $\omega \mapsto M(\omega) = (1 - F(\omega)) \sum_{i=1}^{N} m_i/N$, respectively. $F = 1$ in case there is one or more steps where consensus is not possible. Hence the ensemble mean for M can be expressed as

$$\langle M \rangle = \frac{\overline{k} + \overline{\sigma}}{N} \left(1 - P_F\right), \tag{3}$$

where $\overline{k} = 2 + p(N - 3)$—with p as connection probability—represents the network average degree, $\overline{\sigma} \equiv \sum_{i=1}^{N} \sigma_i/N$, and $P_F = P(\text{“}F = 1\text{”})$ stands for the failure probability. Since a chain failure can only happen after verification, $P_F = P_{F|V} P_V$, where $P_V = P(\text{“}V\text{”})$ is the verification probability and $P_{F|V} = P(\text{“}F\text{”} \mid \text{“}V\text{”})$ the respective conditional probability.

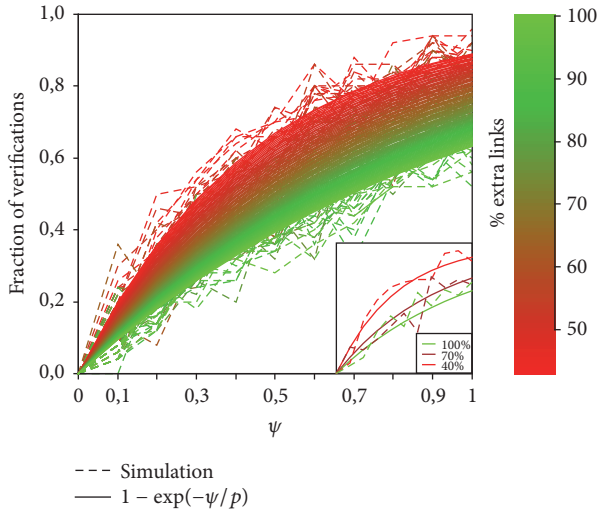


FIGURE 5: Exponential behaviour of the verification probability. As the ratio of attacking nodes increases, verifications grow exponentially. Graph connectivity decreases the verification rate. The inset shows the lower and upper connectivity bounds (40% and 100%) along with an intermediate connectivity of 70%.

Notice that even in the failure-free case there is an upper bound in the mean efficiency $M_0 = (1/N)(\overline{k} + \overline{\sigma})$ imposed by the lack of full connectivity (full connectivity and full recovery with $\overline{\sigma} = 1$ would result in the limit $\langle M \rangle = 1$ (i.e., 100% efficiency)). Hence, both size and connectivity limit network performance due to the partial sequential diffusion, specific for the BC architecture. Next, we look into chain failure probability.

Firstly, it is clear that failure can only happen when at iteration i the last block of node n_{i+1} is an error state. This requires (a) the emisor node to be an attack node with probability ψ and (b) the receiving node is indeed n_{i+1}. Since connections in G are uniformly random, the verification probability at step i can be expressed as $P_V(i) = \psi/k_i$. Also, because the chain is verified if at least one step needs verification, consequently the probability of blockchain verification is

$$P_V = 1 - \prod_{i=1}^{N} \left(1 - \frac{\psi}{k_i}\right). \tag{4}$$

3. Main Results

3.1. Mean Field Approximation for the Chain Verification Probability. By using a *mean field* approximation [11] we replace node degree by the mean network degree $\overline{k}$. In this case, for large N, P_V renders

$$P_V = 1 - \exp\left(-\frac{\psi}{p}\right). \tag{5}$$

In Figure 5 we compare expression (5) with Monte Carlo simulation. For a ring lattice of size $N = 60$ we generated 10^5 synthetic networks with increasing connections and attack strength until graph saturation. Each experimental point (50 runs with the same parameters) represents the fraction of networks that reported a verification step. As the ratio of attacking nodes increases verifications grow exponentially, like the epidemics in [11]. As expected, graph connectivity (measured with the percentage of additional links until saturation) decreases the verification rate.

It is important to provide this estimate because a large number of verification steps translate directly to cost and efficiency in real implementations. Next, we investigate in detail the probability of a chain failure after a verification step.

3.2. Network Consensus Mechanisms. As stressed, if node n_i sends an error code to node n_{i+1} at iteration i, there is a chance for node n_{i+1} to revert this error provided that the consensus reached among its neighbours is over a given threshold. The problem can be formulated as follows. Let U_j represent the neighbours of node n_j. Notice that, at iteration i, nodes in $\widetilde{U}_{i+1} \equiv \{i\} \cup (U_{i+1} \cap U_i)$ have value i while the remaining nodes in U_{i+1} can attain any value from the set $\Gamma \equiv \{E\} \cup \{0, 1, \ldots, i-1\}$. Given a consensus threshold $z \in \mathbb{N}$, let $\chi \equiv \max_{\gamma \in \Gamma}\{\sum_{x \in U_{i+1}} \mathbf{1}_\gamma(x)\}$ denote the maximum frequency of values in U_{i+1} which are different than i. There is agreement when

$$\max\left\{\left|U_{i+1} \cap U_i\right| + 1, \chi\right\} \geq \left\lfloor \frac{k_{i+1}}{z} \right\rfloor, \tag{6}$$

where the notation $\lfloor x \rfloor$ stands for the floor value of $x \in \mathbb{R}$. Notice that $z = 2$ defines a simple majority based consensus among the U_{i+1} sites.

Consequently, inspired by the antientropy and rumor mongering concepts [11], we split the consensus problem of (6) in two mechanisms: clustering and random infection (we use the mathematical epidemiology terminology for infected nodes as those receiving a given state. Notice that in our case infection is not necessarily a negative phenomenon unless the broadcasted quantity is an attack). In the former, n_{i+1} neighbours get an update from n_i to value i. In the latter case n_{i+1} neighbours eventually agree on a value $\neq i$ arriving from other sites different than n_i or from their own replications along the preceding steps in the blockchain cycle.

Notice that the number of symbols in Γ increases with the number of iterations. Therefore, it is increasingly less likely to reach consensus by random infection. On the other hand, the link consensus mechanism does not decrease with the iterations. Hence, the link consensus will dominate over random consensus in the *thermodynamic limit* $N \gg 1$. For a reasonable network size (say $N > 50$) this enables us to neglect the random term contribution to the failure probability. Below we elaborate more on this stochastic approximation.

3.3. Stochastic Network Failure in the Thermodynamic Limit. As demonstrated before, cluster consensus occurs when there are at least $\lfloor k_{i+1}/z \rfloor$ sites out of $k_{i+1} - 1$ possible nodes (the –1 term is because site n_i already holds an i state) with state i. Equivalently, it can be assumed that n_i is connected to at least $\lfloor k_{i+1}/z \rfloor$ nodes in $U_{i+1} \setminus \{i\}$. In this way, we can model the process as a Bernoulli trial (akin to [25]) where the success

variable follows the binomial $\sim B(k_{i+1}-1, p)$. The resulting failure probability renders

$$P_{F|V}(i,z) = \sum_{x=0}^{\lfloor k_{i+1}/z \rfloor} \binom{k_{i+1}-1}{x} p^x (1-p)^{k_{i+1}-1-x}. \quad (7)$$

Since the blockchain failure probability can be expressed as

$$P_F = 1 - \prod_{i=1}^{N} \left(1 - P_F(i,z)\right), \quad (8)$$

by using (8) and (3) and $P_F(i,z) = P_V(i)P_{F|V}(i,z) = \psi P_{F|V}(i,z)/k_i$ we arrive to the expression

$$\langle M \rangle = M_o \exp \sum_{i=1}^{N} \log\left(1 - \frac{\psi P_{F|V}(i,z)}{k_i}\right). \quad (9)$$

Now, provided that the quantity $\psi P_{F|V}(i,z)/k_i$ is small compared to 1, we approximate the logarithm in the last expression by its first-order series expansion. By implementing the same mean field approximation as for P_V in the preceding section we obtain the equation

$$\left\langle M^{MF} \right\rangle = M_o \exp\left(-\frac{\psi}{p} P_{F|V}^{MF}(z)\right), \quad (10)$$

where $P_{F|V}^{MF}(z)$ denotes the corresponding mean field approximation for $P_{F|V}(i,z)$. In Figure 6 we show the mean field approximation to the blockchain performance measured as the average network consensus for $N = 60$ and $z = 2$. As for P_V we generated 10^5 synthetic networks with increasing connections and attack strength until graph saturation. For each network instance, we monitored the number of sites with value i at iteration i within the blockchain cycle. This gives us an empirical estimate for the network match per iteration m_i. Then, we averaged the m_i quantities over the cycle, which results into a measure for the network performance (i.e., consensus level). Finally, we obtain the mean value of this quantity from our Monte Carlo dataset. Each experimental point represents 50 runs with the same parameters.

Notice that 100% performance—blockchain limit—can only be achieved for full connectivity $p \to 1$. The M_o upper bound (black straight line) limits the network match for partial connectivity; as we increase the link probability the performance increases according to (10). Also, stronger attack strategies (larger ψ values) result in lower performance as expected.

A remarkable result in Figure 6 is that beyond a critical value of connectivity p_c; consensus is only limited by information spreading and not by failure. This fact motivates us to look closely at possible estimates of p_c.

3.4. Estimate for the Attack Tolerance Critical Connectivity. Noticing that $P_{F|V}^{MF}(z)$ is nothing else than the cumulative distribution function for the binomial $B(\overline{k}-1, p)$, we use the normal approximation for the binomial distribution as

$$P_{F|V}^{MF}(z) \approx \frac{1}{2}\left[1 + Erf\left(A(p)\right)\right], \quad (11)$$

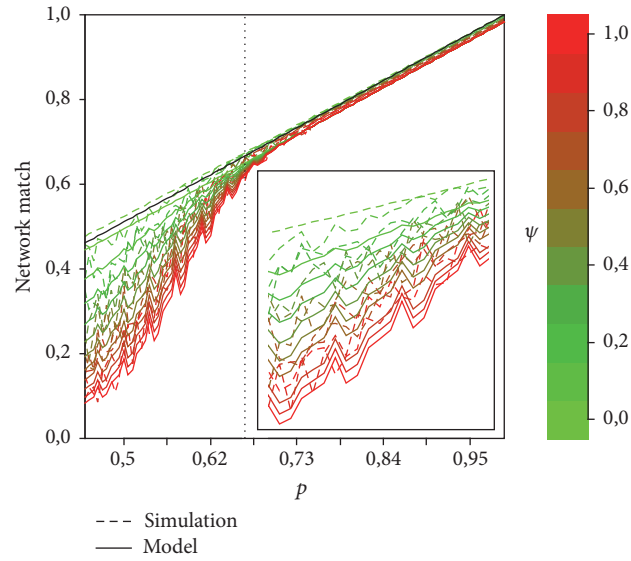


Figure 6: Mean field approximation to the blockchain performance for $N = 60$ and $z = 2$. Starting from the complete graph limit in the top right corner, as connectivity p decreases and attack strength ψ increases, the network match decreases according to (10). The black line represents the M_o upper bound limit. The vertical dotted line at $p = 0.66$ represents an estimate (see the last part of Section 3.3) for the limit $\langle M \rangle \to M_0$ where performance starts to be independent of ψ. The inset shows a zoom for the cut $p \in [0.5, 0.6]$.

where $Erf(x)$ is the error function and

$$A(p,z) = \frac{\lfloor \overline{k}/z \rfloor - p\left(\overline{k}-1\right)}{\left(2p(1-p)\left(\overline{k}-1\right)\right)^{1/2}}. \quad (12)$$

If $\epsilon \in \mathbb{R}$ denotes a small quantity, we inquire the conditions under which $\langle M \rangle$ tends to M_0, or more specifically $|\langle M \rangle - M_0| \leq \epsilon M_0$. To this end, we derive conditions for equality in this expression from (11) and (12). Also, by using the $\log(1-\epsilon) \approx -\epsilon$ approximation, the following condition holds:

$$A(p,z) = Erf^{-1}\left(\frac{2p\epsilon}{\psi} - 1\right). \quad (13)$$

In the large N limit $\overline{k} \approx pN$ and also assuming $\lfloor \overline{k}/z \rfloor \approx \overline{k}/z$, $A(p,z)$ can be approximated as

$$A(p,z) \approx \frac{1 - pN + N/z}{\sqrt{2N(1-p)}}. \quad (14)$$

From (14) one could attempt to solve (13) for ϵ, N, p, and ψ. But this is not possible because the function $Erf^{-1}(x-1)$ diverges for $x = 0$. Still, an interesting case occurs when $\epsilon = \psi/(2p)$. At this limit, (13) only makes sense if $A(p,z)$ vanishes, or, the equivalent, if $p = p^* \equiv (1/z + 1/N)$. However, this value does not provide the asymptotic limit we are looking for.

If we express ϵ in terms of $\alpha \in \mathbb{N}$ through the rescaling $\epsilon \equiv \psi/(p\alpha)$ and we also rewrite (13) in terms of $p - p^*$ we obtain

$$\frac{p - p^*}{\sqrt{1-p}} = \Phi(\alpha), \tag{15}$$

where we have introduced the function: $\Phi(\alpha) \equiv (2/N)^{1/2} Erf^{-1}(1 - 2/\alpha)$.

An operative approximation is possible by considering $\Phi(\alpha)^2 \ll 1$. Then, by using the corresponding solution $p \approx p^* + \Phi(\alpha)$ and for large N we find

$$p_c = \frac{1}{z} + \sqrt{\frac{2}{N}} Erf^{-1}\left(1 - \frac{2}{\alpha}\right). \tag{16}$$

This is nothing more than a useful parametrisation of (13). For $\alpha = 2$ we recover the case $p_c = p^*$. However, larger α values allow us to explore the limit $\langle M \rangle \to M_0$ closely. For instance, for $\alpha = 10$, $N = 60$, and $z = 2$ we arrive at the solution $p_c = 0.66$. This means that, for maximum attach strength ($\psi = 1$), beyond $p = p_c$, the percentage of the deviation of $\langle M \rangle$ from M_0 with respect to M_0 is lower than 15%. By setting other attack tolerance thresholds the p_c value can be adjusted in different realisations of the blockchain network. The value $p_c = 0.66$ represented in Figure 6 can then be conceived as a reasonable threshold for minimum network connectivity ensuring attack tolerance with the above parameters.

4. Proof-of-Concept Example

Notice that the mathematical model addressed in this work abstracts the specifics about transactions, blocks, network architectures, communication protocols, and so on. The implementer must therefore provide definitions for (a) what is a transaction, (b) criterion for consistent ordering of transactions (this is equivalent to defining the analogous to the double-spending problem), (c) how transactions can be packed into blocks, and (d) how is the information spread over the network. When these specifications are provided there are at least two possible scenarios where the findings addressed in this work can be applied: Wireless Sensor Networks and the Internet-of-Things.

As stressed, there are fundamental discrepancies between the proposed model and the current blockchain protocol implementation in cryptocurrencies. In particular, in our approach the information is not transmitted immediately to the miners once blocks are created; it is sequentially diffused as shown in Figure 4. This has its pros and cons depending on the application domain.

When there is no Proof-of-Work requirement the block mining lags tend to zero and the verification and generation delays become comparable. This way the blockchain construction speed is dominated by network latency. Therefore, in the absence of POW, one can reschedule agent's diffusion to save network operations. In the following example we show a proof-of-concept example in the IoT domain where we compare our model with an asynchronous diffusion scheme akin to the conventional blockchain implementation. In the context of IoT consider the problem of human mobility tracking where two individuals leave rooms A and B to reach rooms D, E through hall C (Figure 7). Five presence sensors A–D are continuously capturing data of the form $x_i = \{S_{\mathrm{ID}}, t, v\}$ where S_{ID} identifies the sensor, t represent the measurement time, and $v \in \{0, 1\}$ stands for the presence event. Measures are collected at ΔT intervals and then checked for consistency. Within ΔT, time is split into δt length subintervals. These quantities represent the minimum displacement time between home areas or any other relevant time scale. In general they will be functions of the sensor sampling rates. Therefore, we discretise the continuous variable t into measurement *epochs* n implicitly defined as

$$0 \leq \left(\frac{\delta t}{\Delta T}\right) t - n \leq 1. \tag{17}$$

This allows preprocessing raw data x_i into a dataset $\mathbb{D}$ with entries of the form $\{S_{\mathrm{ID}}, n\} \in \mathbb{D}$, where we also drop $v = 0$ values. Maintaining our cryptocurrency metaphor, we define *transactions* as ordered pairs in $\mathbb{D} \times \mathbb{D}$: $e^{nk}_{XY} \equiv ((X, n), (Y, k))$. For instance, $e^{12}_{\mathrm{AC}} = ((\mathrm{A}, 1), (\mathrm{C}, 2))$ represents the movement of a person from room A at epoch 1 to the hall C at epoch 2. Some transactions do not represent real movement (e.g., e^{22}_{AB}). A possible criterion for the validness of a transaction e^{nk}_{XY} is $X \neq Y$ if $k > n$. This restricts the type of movements allowed in a specific way, but any other criterion can also be defined.

Next we define a *path* P as an ordered sequence of transactions. If E denotes the set of possible transactions among the measurements in $\mathbb{D}$ collected in ΔT, consider two possible paths:

$$\begin{aligned} P &: \left(e^{01}_{\mathrm{AC}}, e^{12}_{\mathrm{CD}}, e^{23}_{\mathrm{BC}}, e^{34}_{\mathrm{CE}}\right), \\ P' &: \left(e^{01}_{\mathrm{AC}}, e^{12}_{\mathrm{CD}}, e^{23}_{\mathrm{CE}}, e^{34}_{\mathrm{BC}}\right). \end{aligned} \tag{18}$$

Both paths represent the movement of two individuals from A, B to D, E. However, P' is not consistent, since the person in B intends to move from C to E before reaching C.

Since we neglect POW, we can consider blocks containing one transaction only which can therefore be generated immediately. The order criterion provides means for building the information chain avoiding the type of order inconsistencies commented above.

We also consider a minimal set of three distributed agents (*miners* in our analogy) which will build the chain. Depending on the network architecture and the communication protocol the information flow among agents can be defined in different ways. However, the model provided in Section 2 allows a considerable reduction of network operations which is more amenable for an IoT implementation. In the bottom panel of Figure 7 we use simplified sequence diagrams to compare the information flow of blockchain (a) and sequential diffusion (b) models as we did in Figure 4. In the lower part, we have also included a summary of the local information stored at each node.

Without loss of generality the mining ordering can be mapped to nodes 1–3 (again, as in Figure 4, we use a single thread for verification and mining processes in the nodes, since mining times are much larger than verification times).

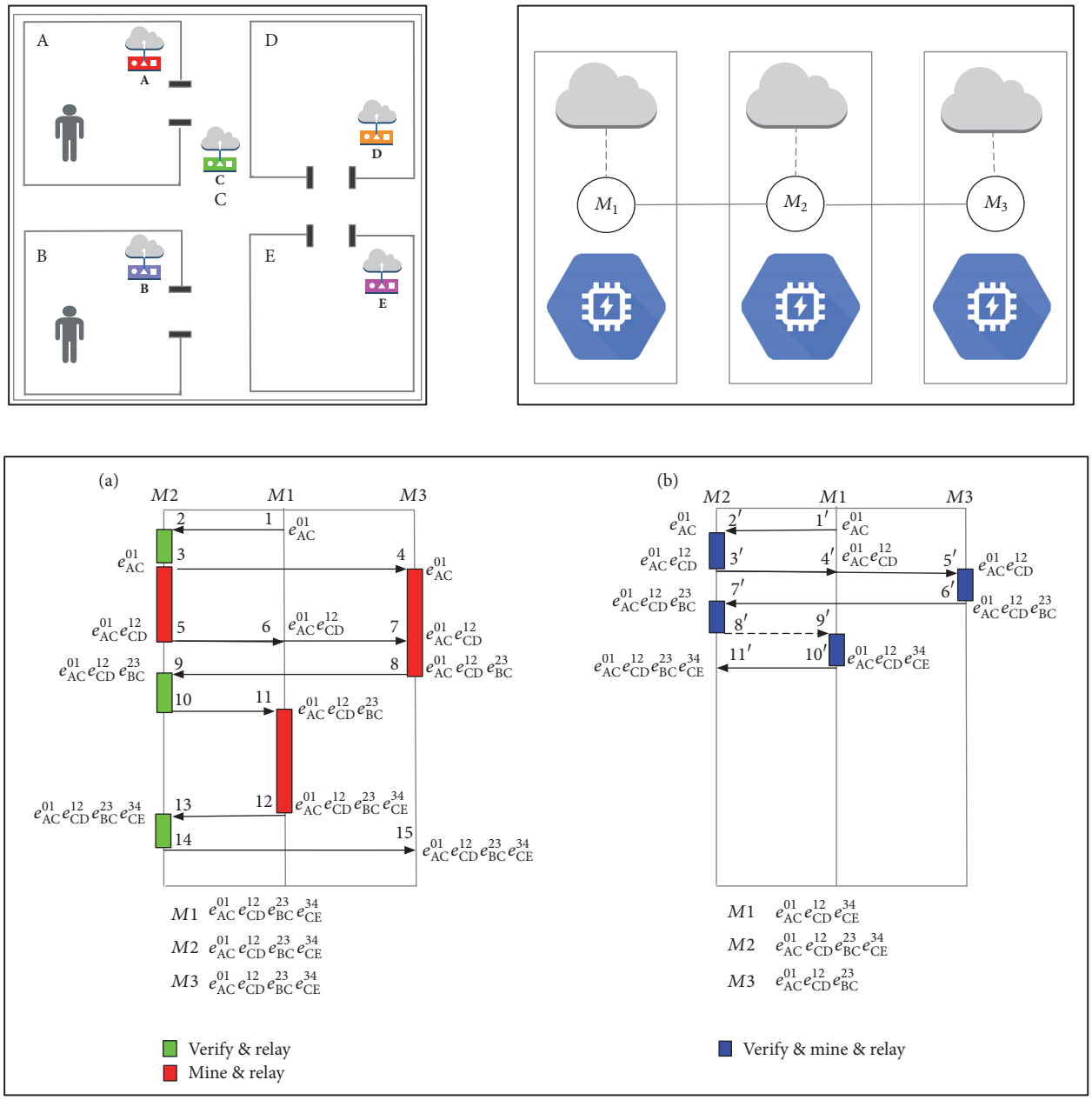


FIGURE 7: Schematic of a possible application of the model developed in this work. Five presence sensors monitor the movement from rooms A, B to D, E (top left panel). A minimal network of three distributed agents—*miners* (top right panel)—build consistent orderer aggregations of measurements. In the bottom panel we compare the sequence diagrams from the real blockchain and the sequential model.

In (a), first M_1 extracts and validates transaction e^{01}_{AC} from E and broadcasts the corresponding block to the network (1-2). After validating the block, M_2 in turn forwards it to M_3 (3-4). At a later time, M_2 validates e^{12}_{CD}, adds it to its local copy (5), and distributes the information among other nodes (6, 7). Next node 3 has itself mined e^{23}_{BC} (8) which is then validated and sent to the network (9–11). Finally, node 1 only finds it consistent to add e^{34}_{CE} to its local chain (12) and then it broadcasts the information to the network for its validation and transmission (13–15).

However in the sequential diffusion model (b), as stressed, agents do not immediately forward transactions/blocks as they receive them; nodes propagate information when they generate new blocks. In the absence of POW, agents can synchronise to save unnecessary communication processes. This way, node 2 does not forward e^{01}_{AC} ($2'$) to node 3 after receiving it from node 1 ($1'$); the information is sent when packing e^{12}_{CD} ($3'$) and so on. This reduces the network traffic considerably. When a miners round is completed node M_2 sends a sync message (dotted line from $8'$ to $9'$) to the next

first mining node (M_1 in this case) until there are no more transactions to verify. If there are N_M agents and transactions, the number of messages grows as $O(N_M^2)$ in (a) and as $\sum_i^{N_M} k_i$ in (b), where k_i is the degree of each node in the agents' network. The maximum overhead is attained for the full graph where $k_i = N_M - 1$ and both models coincide.

Since WSNs and IoT have in general very low battery capacities, this dramatically limits the size of network traffic. Therefore the model addressed here can add value to these situations.

5. Summary and Discussion

In this paper we have analysed, both theoretically and numerically, the conditions under which distributed sequential consensus is possible in presence of partial connectivity and uncertainty. A minimum sequential diffusion model consisting of the superposition of a ring lattice with a random graph along with ordered infection rules allowed us to capture key blockchain elements: partial connectivity, sequential diffusion, and failure spreading.

In our setting a mean field approximation for network degree was helpful in deriving closed-form expressions for the probability of chain verification once errors are detected. We found that verifications grow exponentially with attack. As expected, graph connectivity reduces verification rates. This is a remarkable result because a large number of verification steps translate directly to cost and efficiency in real implementations.

We have also provided expressions for the probability of network breakdown when consensus is not possible. To this end, we have investigated analytically the constituents of the consensus problem in blockchains. We found that clustering dominates over random infection in the large network size limit. This allowed us to derive an expression for the average network performance as a function of connectivity and failure strength. We validated this expression by Monte Carlo simulation. As expected, 100% performance—blockchain limit—can only be achieved for full connectivity. Furthermore, there is an upper bound for network match for partial connectivity. Stronger attack strategies result in lower performance.

The resulting expressions allow us to derive connectivity thresholds above which networks can tolerate attack. Beyond that, lower bound consensus is only limited by information spreading and not by failure. A set of reasonable assumptions and algebraic manipulations allowed us to derive operational expressions for this bound. Specifically, for $N = 60$ simple majority based consensus, we arrived at the solution; $p_c = 0.66$. This means that in a scenario with maximum attach strength, beyond p_c, the percentage deviation of blockchain consensus with respect to the upper connectivity bound is lower than 15%.

Clearly this contribution is just a first step in the understanding of partially connected blockchains; the problem still needs further elaboration in order to foster more robust implementations. For instance, we have neglected some communication issues such as delay or bandwidth limitations. In a future work we will research other topological models such as scale-free and spatial networks with heterogeneous links. Multiplex networks will also allow us to inquire into different attack patterns and the possible counterattacking strategies.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This research was partially supported by the Regional Ministry of Education from Castilla y León (Spain) and the European Social Fund under the *MOVIURBAN* project with Ref. SA070U16.

References

[1] S. Seebacher and R. Schüritz, "Blockchain Technology as an Enabler of Service Systems: A Structured Literature Review," in *Exploring Services Science*, vol. 279 of *Lecture Notes in Business Information Processing*, pp. 12–23, Springer International Publishing, Cham, 2017.

[2] S. Amini, F. Pasqualetti, and H. Mohsenian-Rad, "Dynamic Load Altering Attacks Against Power System Stability: Attack Models and Protection Schemes," *IEEE Transactions on Smart Grid*, pp. 1-1.

[3] S. Sicari, A. Rizzardi, L. A. Grieco, and A. Coen-Porisini, "Security, privacy and trust in Internet of Things: the road ahead," *Computer Networks*, vol. 76, pp. 146–164, 2015.

[4] H. Menashri and G. Baram, "Critical infrastructures and their interdependence in a cyber attack - the case of the u.s," *Military and Strategic Affairs*, vol. 7, no. 1, p. 22, 2015.

[5] R. Roman, J. Zhou, and J. Lopez, "On the features and challenges of security and privacy in distributed internet of things," *Computer Networks*, vol. 57, no. 10, pp. 2266–2279, 2013.

[6] A. Chakravorty, T. Wlodarczyk, and C. Rong, "Privacy preserving data analytics for smart homes," in *Proceedings of the 2nd IEEE Security and Privacy Workshops, SPW 2013*, pp. 23–27, usa, May 2013.

[7] S. Nakamoto, *Bitcoin: A Peer-to-Peer Electronic Cash System*, https://bitcoin.org/en/.

[8] A. Gervais, G. O. Karame, K. Wüst, V. Glykantzis, H. Ritzdorf, and S. Čapkun, "On the security and performance of Proof of Work blockchains," in *Proceedings of the 23rd ACM Conference on Computer and Communications Security, CCS 2016*, pp. 3–16, aut, October 2016.

[9] A. Dorri, S. S. Kanhere, R. Jurdak, and P. Gauravaram, "Blockchain for IoT security and privacy: The case study of a smart home," in *Proceedings of the 2017 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)*, pp. 618–623, Kona, Big Island, HI, USA, March 2017.

[10] M. Swan, *Blueprint for a New Economy*, O'Reilly Media, 1st edition, 2015.

[11] A. Demers, D. Greene, C. Hauser, W. Irish, and J. Larson, "Epidemic algorithms for replicated database maintenance," in *Proceedings of the the sixth annual ACM Symposium*, pp. 1–12, Vancouver, British Columbia, Canada, August 1987.

[12] A. Vespignani, "Modelling dynamical process in complex sociotechnical systems," *Nature Physics*, vol. 8, no. 1, pp. 32–39, 2012.

[13] A. J. Bray and G. J. Rodgers, "Diffusion in a sparsely connected space: a model for glassy relaxation," *Physical Review. B. Condensed Matter. Third Series*, vol. 38, no. 16, part A, pp. 11461–11470, 1988.

[14] D. J. Watts and S. H. Strogatz, "Collective dynamics of 'small-world' networks," *Nature*, vol. 393, no. 6684, pp. 440–442, 1998.

[15] M. E. Newman and D. J. Watts, "Renormalization group analysis of the small-world network model," *Physics Letters A*, vol. 263, no. 4-6, pp. 341–346, 1999.

[16] R. Monasson, "Diffusion, localization and dispersion relations on "small-world" lattices," *The European Physical Journal B*, vol. 12, no. 4, pp. 555–567, 1999.

[17] H. Mehlhorn and F. Schreiber, "Small-world property (encyclopedia of systems biology)," in *Encyclopedia of Systems Biology*, W. Dubitzky, O. Wolkenhauer, K.-H. Cho, and H. Yokota, Eds., pp. 1957–1959, Springer, NY, USA, 2013.

[18] R. Guerraoui, M. Hurfinn, A. Mostefaoui, R. Oliveira, M. Raynal, and A. Schiper, "Consensus in Asynchronous Distributed Systems: A Concise Guided Tour," in *Advances in Distributed Systems*, vol. 1752 of *Lecture Notes in Computer Science*, pp. 33–47, Springer Berlin Heidelberg, Berlin, Heidelberg, 2000.

[19] A. Babaee and M. Draief, "Distributed Binary Consensus in Dynamic Networks," in *Information Sciences and Systems 2013*, vol. 264 of *Lecture Notes in Electrical Engineering*, pp. 57–65, Springer International Publishing, Cham, 2013.

[20] R. Wattenhofer, *The Science of the Blockchain*, CreateSpace Independent Publishing Platform, 2016.

[21] D. Braha, A. A. Minai, and Y. Bar-Yam, *Complex Engineered Systems*, New England Complex Systems Institute series on complexity, Springer, Berlin, Heidelberg, 2006.

[22] J. Scheffran, "The complexity of security," *Complexity*, vol. 14, no. 1, pp. 13–21, 2008.

[23] R. Karp, C. Schindelhauer, S. Shenker, and B. Vocking, "Randomized rumor spreading," in *Proceedings of the 41st Annual Symposium on Foundations of Computer Science*, pp. 565–574.

[24] R. Albert, H. Jeong, and A.-L. Barabási, "Error and attack tolerance of complex networks," *Nature*, vol. 406, no. 6794, pp. 378–382, 2000.

[25] F. P. Junqueira and K. Marzullo, "Synchronous consensus for dependent process failures," in *Proceedings of the 23th IEEE International Conference on Distributed Computing Systems*, pp. 274–283, usa, May 2003.

[26] V. Sood, T. Antal, and S. Redner, "Voter models on heterogeneous networks," *Physical Review E: Statistical, Nonlinear, and Soft Matter Physics*, vol. 77, no. 4, Article ID 041121, 2008.

[27] C. Decker and R. Wattenhofer, "Information propagation in the Bitcoin network," in *Proceedings of the 13th IEEE International Conference on Peer-to-Peer Computing, IEEE P2P 2013*, ita, September 2013.

[28] I. Eyal and E. G. Sirer, "Majority Is Not Enough: Bitcoin Mining Is Vulnerable," in *Financial Cryptography and Data Security*, vol. 8437 of *Lecture Notes in Computer Science*, pp. 436–454, Springer Berlin Heidelberg, Berlin, Heidelberg, 2014.

[29] J. Göbel, H. P. Keeler, A. E. Krzesinski, and P. G. Taylor, "Bitcoin blockchain dynamics: The selfish-mine strategy in the presence of propagation delay," *Performance Evaluation*, vol. 104, pp. 23–41, 2016.

[30] F. Prieto-Castrillo, A. Astillero, and M. Botón-Fernández, "A Stochastic Process Approach to Model Distributed Computing on Complex Networks," *Journal of Grid Computing*, vol. 13, no. 2, pp. 215–232, 2015.

[31] J. Bajo, J. F. De Paz, G. Villarrubia, and J. M. Corchado, "Self-organizing architecture for information fusion in distributed sensor networks," *International Journal of Distributed Sensor Networks*, vol. 2015, Article ID 231073, 13 pages, 2015.

[32] A. M. Antonopoulos, *astering Bitcoin: Unlocking Digital Crypto-Currencies*, O'Reilly Media, Inc., 1st edition, 2014.