Efficient topology control of blockchain peer to peer network based on SDN paradigm

Varun Deshpande, Hakim Badis, Laurent George

Peer-to-Peer Networking and Applications
https://doi.org/10.1007/s12083-021-01248-6

Efficient topology control of blockchain peer to peer network based on SDN paradigm

Varun Deshpande[1,2] · **Hakim Badis**[1,2] · **Laurent George**[1,2]

Received: 29 July 2020 / Accepted: 23 September 2021

Abstract
Blockchain allows to securely store, using cryptography functions, validated transactions and other data across its peer to peer (P2P) network. This P2P network is generated and maintained by the participating peers in a distributed manner through peer discovery, neighbor selection and managing inbound/outbound connections. As these tasks require extra consumption of network bandwidth, energy, memory and CPU usage, resource-limited devices may not support inherent blockchain applications. Further, changes in the topology control functions or reconfiguration should be considered by all peers to fully benefit from the new performance, which can take time. The aim of this paper is to build a blockchain network in a manner that greatly reduce topology control overhead while guaranteeing the properties such as high flexibility, fast reconfigurability, connectivity, small diameter and clustering. For this, we propose to use the Software-Defined Networking (SDN) paradigm to manage the blockchain P2P network. This way, the topology control tasks are moved off the peers to a secure overlay layer composed of multiple servers having synchronized databases. This new layer is responsible for building and managing the topological structure of the P2P network layer based on random *r-out* digraphs. Next, we mathematically discuss *r-out* digraphs generation using binomial distribution and preferential attachment models. Then, and in order to reduce the number of connections per peer, we establish lower and minimum upper bounds on outbound and inbound connections respectively that still guarantee the P2P network feasibility and connectivity. Further, we investigate topological properties of blockchain P2P network such as connectivity, diameter and clustering. Finally, we provide extensive simulation and numerical results to verify the efficiency of our approach and illustrate the effects of centralized topology control on network performance.

Keywords Blockchain · Peer to peer network · *r-out* digraph · SDN paradigm · Topology control · Uniform distribution model · Binomial distribution model · Preferential attachment model · Network properties · Connectivity

1 Introduction

With blockchain already becoming mainstream, enthusiasts have moved on to more complex and perpetual Blockchain 3.0 [1]. Blockchain's reach has not been only limited to the financial sector. A simple market review lands us with a plethora of solutions [2], whether their classification is based on the type of blockchain or its protocol. Blockchain architectures combine many eclectic technologies like peer to peer (P2P) network, Cryptography, Data Storage (Shared Ledger) into Blocks which are chained together. The persistence and immutability of data are achieved by maintaining this chain of data blocks across each participating node. The coherency for the same is achieved through blockchain communication protocol (which manages information dissemination) using the underlying P2P network. Most, if not all, performance metrics in a blockchain are directly related to the performance of underlying P2P network.

In general sense, P2P Network is a virtual overlay network that is built on top of a physical network topology. Each peer is able to directly communicate (over one or multiple physical links) with all other peers on the network. This property has facilitated the design of many distributed applications such as distributed file storage/sharing, distributed

✉ Varun Deshpande
varun.deshpande@esiee.fr

Hakim Badis
hakim.badis@univ-eiffel.fr

Laurent George
laurent.george@esiee.fr

[1] ESIEE Paris, Noisy-le-Grand, France

[2] LIGM, Université Gustave Eiffel, Champs-sur-Marne, France

Published online: 06 October 2021

games, distributed e-commerce, distributed social networking, etc. Based on how the peer (having a desired resource) is located, P2P networks can be classified into three types: unstructured, structured and hybrid. In an unstructured P2P network, the search query is disseminated to the rest of the network mainly using flooding or gossiping protocols. Both protocols are robust, but they suffer from high flooding overhead and high latency respectively. In structured networks, the search query is efficiently routed to the peer that has the desired resource. Most of the structured P2P networks are based on a Distributed Hash Table (DHT). However, structured networks suffer from high cost of advertising/discovering resources. In hybrid networks, the search query is transmitted to a centralized server that helps finding peers having the desired resource. Consequently, each peer should inform the centralized server about its shared resources. This P2P network type is fast, reliable, and has low overhead, but is susceptible to single point of failure (SPoF).

Generally, blockchain P2P networks are unstructured, where each peer randomly selects a limited number of neighbors from its peers' lists to establish and maintain outbound connections. The initially empty peers' lists are first populated by some bootstrap nodes, and then progressively filled by peers found through a distributed peer discovery mechanism such as Peers' list Exchange Protocol (PEX) [3] and adjusted Kademlia DHT protocol [4]. Unfortunately, these operations consume additional resources such as bandwidth, energy, memory and CPU usage. Consequently, resource-limited peers (e.g., IoT devices, smartphones, etc.) are impacted, thus, reducing their ability to perform some functions in a blockchain network. Additionally, the maintenance of multiple TCP outbound and inbound connections has an impact on the consumption of CPU resources as well. To enable resource-limited peers' interaction on the blockchain network and saving resources, several blockchain frameworks have developed the concept of light peers [5] (e.g., wallets) on one hand, and on the other hand, lightweight consensus algorithms and mining strategies. However, peers (light or full) are still required to perform a peer discovery process and maintain outbound connections (with or without accepting inbound connections).

The efforts made by each individual peer to provide distributed topology control in a blockchain P2P network requires extra consumption of resources. Furthermore, replicating any changes in topology control functions or reconfiguration (to enhance the overall network performance) of all participating peers is generally performed manually. In other words, each peer has the responsibility to upgrade its blockchain client software. As some peers might take more time to upgrade their client software on top of the information dissemination time, the desired performance optimizations of the overall blockchain P2P network requires extra time to take effect. To address these points, we propose in this paper to use an architecture based on the Software-Defined Networking (SDN) paradigm [6]. The SDN paradigm is still in an early stage of development and market entry. SDN paradigm works on the principle of separating the control plane that carries the signaling traffic, from the data plane that carries the data traffic. The control plane is handled by a centralized controller who has an entire view of the network topology. The control functions are excluded from the network devices and are aggregated and placed onto the centralized SDN controller. With this architecture, SDN paradigm aims to improve the control of networks, offering agile reconfiguration and flexibility.

The architecture of our proposed blockchain network comprises of two separate layers: a control layer dedicated to the topology control of the P2P network, and a data layer dedicated to P2P data communications between (transactions, blocks and contracts). The control layer includes multiple secure servers (to avoid SPoF) with synchronized databases. Its objective is to build and maintain, in a centralized manner, the P2P network topology. In our proposal, the P2P network topology follows the random *r-out* digraphs model, where *r* is the number of outbound connections. This choice is justified by the fact that it is already being used by most of existing blockchain frameworks (Section 2). The number of the outbound connections is specified in the blockchain software clients. We study both approximate and exact *r-out* digraphs generations using binomial, uniform and preferential attachment models. We also determine the feasibility conditions of constructing *r-out* digraphs in terms of number of inbound and outbound connections. Finally, we investigate and evaluate some *r-out* digraph properties (as a function of network size and no. of outbound connections) that highly impact the blockchain P2P network performance. These properties concern network connectivity, diameter and clustering.

The remaining paper is organized as follows: Section 2 provides a literature review on related works and sets the preface for the rest of the paper. Section 3 deals in detail about constraints and problems in building an optimal P2P network for blockchain. Section 4 elaborates different *r-out* graph generation techniques while Sect. 5 explains the chosen centralized topology management. After this, we explain blockchain P2P network modeling in Sect. 6. Section 7 elaborates on random *r-out* digraph properties. We show in Sect. 8, the simulation methodology used for our described model and also the results for the same. Finally, we conclude this work in Sect. 9 with insights on future work.

2 Related work and background

Even though legacy blockchains (like Bitcoin) struggle to meet higher transactions per second rate, their economics have scaled up resulting in the commercial introduction of numerous other crypto-currencies. As of 2020, there are

more than 5000 cryptocurrencies with an estimated market cap of more than $200 billion [7]. This itself shows that the concept of crypto currency has been perceived in a good way by the masses. With the discovery of new use-cases in the domains other than finance or economics, a wide variety of blockchain frameworks have been developed. However, research in this field is still at a nascent level and many shortcomings (like scalability, low throughput, redundancy) need to be addressed before its large-scale adoption [8].

As blockchain's core lies with P2P network, it has benefited from previous extensive work on P2P systems popularly used in file sharing applications to boost the performance and avoid SPoF. Moreover, blockchain technology has improved some aspects of the P2P Network to provide high-efficiency consensus protocols [9–11], identity privacy [12, 13] and security of transactions [14]. The distributed topology controlling in blockchain's P2P network is one of the aspects studied. It includes peers discovery, outbound neighbors selection and maintenance. Thus, before launching the peer discovery process, new peers that want to join an existing blockchain P2P network should initialize their peer lists by bootstrap nodes obtained by querying a hard-coded set of DNS server or boot nodes [15, 16]. To discover other peers and populate the peers list, multiple distributed peer discovery protocols are used by blockchains such as PEX [3] and adjusted Kademlia DHT protocol [4]. Based on local peers' list, each participating peer selects randomly r number of outbound TCP connections based on certain metrics (e.g., freshness of IP, ping time, nearest neighbor). Table 1 gives a standard outbound connection count r for some well-known blockchain frameworks. The last task in topology control consists to maintain these r number of active outbound connections.

As each peer performs topology control, extra consumption of network bandwidth, energy, memory and CPU usage are needed [17, 18]. Thus, resource-limited peers (e.g., IoT end devices and smartphones) may not have the capabilities to perform all blockchain functions like mining, routing, database and even wallet [19, 20]. In order to deal with resources limitations, in recent years, blockchains have undergone significant structural changes. For integrating blockchain with IoT, a detailed literature review was conducted in [21], where the authors give a general and most common architecture for blockchain of Things (applied BCoT) and how to integrate the IoT devices. In this architecture, resource-constrained IoT devices such as sensors and actuators are proposed to play the role of lightweight nodes. Thereby, they can verify transactions without downloading or storing the whole blockchain with the help of full nodes. To this end, they should still perform peers discovery to find full nodes. In [22], the authors propose another blockchain architecture where low resource devices in a smart home benefit from a centralized manager that manages all incoming and outgoing connections. In [23], the authors propose to use a consensus protocol like Practical Byzantine Fault Tolerance (PBFT) [24] instead of Proof of Work (PoW) for permissioned blockchain. Moreover, mining or more appropriately block confirmations are carried out on edge devices only which posses more resources than IoT end systems.

For supporting the blockchain operations on smartphones, there exist several blockchain applications for smartphones to enable wallets (e.g., Ledger Nano X [25], Edge [26], Atomic Wallet [27]) and mining (e.g., MinerGate [28], CoinHive [29]). However, it has been shown that mining using smartphones is nonprofitable [30] due to the computing power and battery limitations. A comprehensive overview of the impact of blockchain operations on smartphones was reported in [31]. The authors also propose a novel consensus protocol based on a combination of PoW, Proof of Activity (PoA), and Proof of Stake (PoS) algorithms to render smartphones mining-friendly. Based on the possibility of mining and wallets, recently, several major smartphone manufacturers such as Samsung, HTC, and Pundi X, are designing their smartphones to be blockchain-enabled.

Our work can be seen as complementary to existing solutions for resource-constrained peers in blockchains. It proposes to move the topology control functionality off of the peers and into a separate secure overlay layer composed of multiple servers, which drastically reduces the high control topology overhead and its negative impact on peer resources. This solution is inspired from SDN paradigm [6] to allow, in addition to reducing topology control overhead, high degree of programmability and hence high flexibility in P2P network [32, 33].

Table 1 Default no. of outbound connections for various blockchain frameworks

Blockchain	r value
Bitcoin	8
Ethereum	13
Litecoin	125
Neuorochain	3
Stellar	8
Monero	8
Tezos	100

3 Problem statement

In blockchain technology, block and transaction messages are propagated through a distributed P2P network. Each participating peer performs local actions to create and maintain a global topology. These actions include peer discovery, neighbor selection, establishing and maintaining outbound connections, accepting or rejecting inbound connections. The blockchain P2P network properties like topology, size

(maximum number of participating peers) and performance (connectivity, diameter, etc.) are highly impacted by the blockchain type, the consensus protocol used, the number of inbound/outbound connections per peer (specified by the blockchain client software) and the neighbor selection process. For example:

- Permissioned blockchains that use voting based consensus generally require a fully/highly connected P2P network between voting peers with small size (ten to hundreds peers). The topology of the clique formed by voting peers can be modeled by complete or nearly complete digraphs. A complete digraph can be considered as a special case of $(n-1)$-*out* digraphs where n is the digraph size and $n-1$ is the out-degree of each vertex.
- Permissioned blockchains that use lottery based consensus require a small size P2P networks (but larger than those used by voting based consensus) where all peers have the same small number of random outbound connections. Thus, the topology of these networks can be viewed as random *r-out* digraphs, where r is the out-degree of each vertex.
- Non-permissioned blockchains that use proof based consensus support a large number of nodes (thousands) in P2P networks where all peers have the same small number of random outbound connections. Thus, the topology of these networks can be also viewed as random *r-out* digraphs.

Therefore, many existing blockchain frameworks' P2P network can be viewed as random *r-out* digraphs. The number of outbound connections, r, is specific to each blockchain framework (Table 1). However, some important questions arise like: How is the default value of r defined? What is its impact on the blockchain performance? and what are the impacts of the in/out degree distribution, of the peer discovery protocol and of the neighbor selection process on the blockchain performance? On which, to the best of our knowledge, no deep investigation is conducted till date.

The main objective of our work is to build an *r-out* digraph for blockchains that can be used by resource-limited devices such as IoT devices, smartphones, etc. The value of r as well as the in/out degree distribution should be dynamically adapted depending on the desired blockchain P2P network performance in terms of connectivity, diameter and clustering. To the best of our knowledge, almost all blockchain frameworks follow random *r-out* digraphs as they specify default value of outgoing connections in their client software implementations.

However, for the case of blockchain frameworks where the underlying p2P network is not *r-out* graph, our work can be adapted accordingly, by selecting an appropriate graph type and generator and using it in the secure overlay layer for the topology construction as explained in the next sections of this chapter. Next, we describe in detail the key topics we consider for enhancing the blockchain P2P network P2P performance.

3.1 Topology control

The topology control mainly includes two operations: topology construction and topology maintenance. The topology construction is related to peers discovery and neighbors selection while topology maintenance is related to reconstructing the topology when nodes join/leave the network or when changes are proposed in existing blockchain frameworks to build the global blockchain P2P network. When a peer wants to join a blockchain P2P network, it starts by discovering other participating peers that are already connected. The IP addresses of the peers found are stored locally in a list. Then, the peer randomly selects r neighbors from the list to establish and maintain its outbound connections. When an outbound connection fails, another neighbor is selected and a new outbound connection is initiated. A distributed topology control suffers from two difficulties: overhead and lack of flexibility.

3.1.1 Overhead

The distributed topology control operations involve extra overhead in terms of communication traffic which results in significant consumption of network bandwidth, energy, memory and CPU usage. Thus, resource-limited devices such as IoT devices or smartphones do not support blockchain capabilities or, at best, run a blockchain with limited functionalities. For example, LORaWAN [34] or SigFox [35] end-devices have a small battery, a low data rate (limited number of exchanged messages per day and short payload size) and a low computing power and storage. They cannot perform by themselves the topology control operations. To deal with this issue, we propose a centralized topology control where a secure overlay layer composed of distributed and synchronized servers handles and manages the topology control operations instead of the participating peers. The vertical plane (peers-servers) is dedicated to the topology control traffic while the horizontal plane (inter-peer) is dedicated to data traffic control.

3.1.2 Flexibility

In a blockchain ecosystem, the ability to adapt and include changes on-demand is defined as flexibility. In traditional blockchain P2P network, any change in the topology control functions/configuration such as the peer discovery process, the number of outbound/inbound connection or the neighbor selection strategy, is handled by each

participating peer. This operation requires more time, especially for large-scale public blockchains, to allow all peers updating the installed blockchain software. When some peers (including key peers) take time to update the configurations, the global blockchain P2P network properties can be impacted and the performance can be degraded.

The idea of re-configurable networks has received considerable attention in recent years due to the emergence of the Software-Defined Networking (SDN) paradigm where a separation is done between the data plane and the control plane. The control plane is placed in a centralized controller. Thus, any changes in the topology control functions/ configuration will only be implemented at the controller and the resulting configuration is injected in the peers. In this paper, we propose to use the SDN paradigm where the centralized controller is an overlay layer of distributed and synchronized servers.

3.2 Connectivity

For any underlying P2P network, *connectivity* is necessary for a number of reasons. Proper connectivity helps not only in faster information dissemination but also to maintain a single coherent blockchain by reducing the probability of fork formation that can be the source of attacks as explained in the following example. Figure 1 shows a simple view of a blockchain after the P2P network becomes disconnected and forms two independent sub-networks. We can see that each independent sub-network adds new subsequent blocks and considers the new blockchain fork as the main blockchain. In this case, a malicious peer can efficiently perform a double-spending attack on each sub-network. Moreover, it is possible that the isolated sub-network lacks even one full node, thereby making the blockchain fork invalid within that sub-network as full transaction history from genesis block is absent. Further, when the number of isolated sub-networks within a P2P network increases, full nodes are unequally split across them. This increases the success rate of Distributed Denial-of-Service (DDoS) attack which works by flooding full nodes of a targeted isolated sub-network.

When the two isolated sub-networks are merged into one network, one blockchain appears with two forks, out of which, the longest chain is considered in the main chain. This effectively reverses the transactions in the other fork which may lead to financial repercussions for the businesses depending on blockchain.(e.g., real-world goods delivery, on-demand services, etc.) In this paper, as our blockchain P2P network topology is a *r-out* digraph, we derive the smallest value of r to guarantee the network connectivity and avoid such forks altogether.

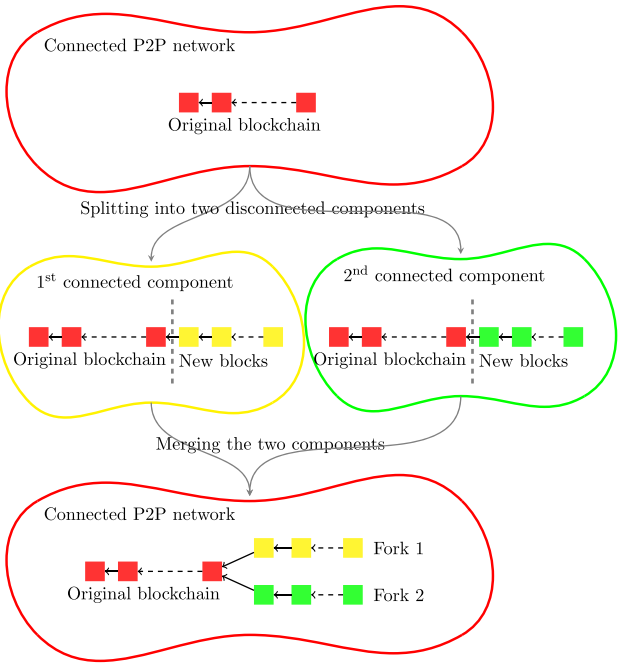


Fig. 1 Example of a blockchain evolution when splitting and merging a connected P2P network

3.3 Smaller diameter

Smaller diameter of the P2P network helps in quicker information dissemination and thereby minimizes the number of transient forks in a given blockchain. Fewer forks increase overall trustability of the blockchain and reduce the mining time in the blockchain (and also to assure that the transaction is indeed included in the main chain). Smaller diameter can also ensure smaller block propagation time thereby enabling the participants either to add more blocks in the given period or increase the block size, both without increasing the number of transient forks. It also aids in quicker consensus time.

3.4 Consensus efficiency

The goal of blockchain consensus protocols is to maintain the exact same copy of the ledger on all honest peers, tolerating a bounded number of byzantine faults [36]. To reach consensus, all consensus algorithm require at least a connected P2P network [10]. The existence of a solution to a consensus problem depends on the hypothesis we make on communication delays and on processor speeds:

– Synchronous: the worst case communication delays are bounded and the bounds are known. The processor speeds are bounded and known.

- Partially Synchronous: The worst case communication delays are bounded but the bounds are not known, or the bounds are known but after an unknown period of time, or the bounds are valid during a sufficient period of time for the consensus to be completed. Same for the processors speeds.
- Asynchronous: The communication delays are not bounded and can be infinite. The processor speeds are unknown.

An important result shows that the consensus problem cannot be solved in the asynchronous case (assuming no global time reference is available, i.e. no clock synchronization protocol) even in the case of a single node crash [37]. In this paper, we therefore consider the partially synchronous case. We also suppose message authentication. In the case of partially synchronous communications with authentication, several consensus protocols have been proposed [38]. These protocols vary with different blockchain networks [10], and can be classified as voting-based consensus and proof-based consensus algorithms.

The voting-based consensus algorithms are based on a cooperative approach between a subset of *identity-verified* participating peers, and are mainly (but not only) applied for permissioned or private blockchain networks. Examples of such algorithms are PBFT (tolerates at most f faulty/malicious nodes for $3f + 1$ total nodes) [24], Crash fault Tolerance (CFT) [39], etc. The proof-based consensus algorithms are based on a competition approach between participating peers, and are mainly (but not only) applied for non-permissioned blockchain networks. Examples of such algorithms are PoW [40], PoS [40], Proof of Elapsed Time (PoET) [41], Proof of Burn [42], etc.

Indeed, while voting-based consensus assume fully connected P2P network between voting peers [10], proof-based consensus algorithms assume (simply) connected P2P networks [43, 44]. In our work, in order to keep the consensus efficiency, we maintain the connectivity conditions according to the blockchain type. Further, despite our proposed architecture being two-layered, there are no additional security implications arising out of this. This is because the main blockchain functioning is still carried out in a single layer (lower-layer) only thereby making no difference w.r.t. consensus application in blockchain's working.

3.5 Clusters

In graph theory, clustering refers to the task of grouping nodes into clusters, so that the edge density is higher within clusters and relatively lower between clusters. Generally, the clusters appear in large-scale public blockchains due to the presence of peers (called super-peers) having more resources to perform the consensus algorithm, relay data between clusters, etc. In our work, we propose to use a centralized network generator at the topology control layer to manage the blockchain P2P network clustering. Thus, the number of clusters can be dynamically adjusted according to the desired performance.

4 Generating *r-out* digraphs

The key topics mentioned in Sect. 3 are important to optimize blockchain topology w.r.t a given use-case scenario. These key topics are therefore to be modeled using a structure close to the actual P2P network. For this, we use graph theory to model blockchain's P2P network. In the context of blockchain, the underlying P2P network construction starts by finding peers with a peer discovery method. For peer discovery, distributed or centralized methods can be applied. With distributed method, each peer performs neighbor discovery independently using different techniques (e.g., asking default servers for the neighbor list, listening to address advertisement messages, using default neighbor addresses) [45]. Contrarily, with the centralized method, each peer registers on the centralized server(s), who alone has the vision of all online peers.

Next, we consider a special class of graph called *r-out* digraph for modeling the P2P network. Our motivation comes from the fact that most blockchains' P2P network, at any given time, maintain r outbound connections. Further, we consider *r-out* graph generation techniques for both settings i.e. centralized as well as distributed, to incorporate peer discovery properties. The *r-out* digraphs can likewise be generated in two ways i.e. centralized and distributed.

4.1 Distributed *r-out* graph generation

The distributed method of *r-out* graph generation is the most commonly used in blockchain architectures. In this type of generation, each individual node selects its own r peer neighbors. The main advantage for such type of generation is that it is very simple without the involvement of any centralized entity. Due to this, such distributed *r-out* graph generation are used in almost all non-permissioned/public blockchain.

Further, the distributed method is more resilient against failures of other peers but is more resource intensive. The resulting blockchain architecture is immune to attacks like DDoS because, in a non-permissioned/public blockchain, any node can participate in without a centralized access control and the graph generation is distributed. This makes distributed *r-out* graph generation the default choice as higher resiliency results in higher reliance on the financial transactions from the blockchain.

However, there are certain disadvantages associated with the distributed *r-out* graph generation like difficulty in controlling topology. For e.g., in an open and unregulated network, clusters are bound to form around the nodes that are more active and with more resources. However, as the topology generation/graph is distributed, any controlling action results in a very high overhead. Another issue with the distributed *r-out* graph generation is the resources overhead related to the neighbor discovery protocol.

As there is no centralized database, the participants periodically transmit the messages of the new peer participants so as to keep the network as uniform as possible when new connections are added. Further, as the network participation is non-permissioned in most blockchain frameworks that utilize distributed method, it is more difficult to verify data transactions as the owner cannot be effectively verified.

4.2 Centralized *r-out* graph generation

The peer discovery using centralized *r-out* graph generation is fast without requiring additional resources (bandwidth, power consumption, and storage of all neighbors information). In the context of IoT, this is advantageous for mobile peers and remote nodes with limited resources. However, this protocol is prone to DoS attacks and is less resilient against the failure of the centralized server (SPoF). Hence, a group of backup servers are necessary. This allows us to mitigate SPoF and DoS problems.

Topology control is easier with the centralized approach than with the distributed approach for *r-out* graph generation as each peer has a direct communication link with the centralized server(s) and the centralized server(s) can quickly detect when a node leaves/joins the network. As a result, the topology reconstruction is fast and efficient. As centralized server(s) is/are in-charge of the topology control, it further adds to the flexibility of the network and on-demand required changes to manage temporary surge can be effectively and quickly implemented.

Although, it introduces hierarchy in the topology management, the blockchain functions still remains decentralized. Another advantage of the centralized method is that since all the node addition/removal goes through centralized server(s), the identity of each node can be ascertained using technologies like Secure Element (SE) [46]. SE has been already to guarantee *root of trust* in Industry 4.0 [47], IoT [48], Smart Vehicles [49], etc. SE has also been used in blockchain transaction security [50] which was implemented in the context of marketplace transactions in Demand Response [51]. In our work, we propose to use SE and extend its application to include the functionality of authentication of lower layer peers, verifying data transactions and overall maintain trust in the blockchain network.

4.3 Selecting the appropriate graph generation technique

A neighbor selection process may result in two types of P2P networks i.e., structured and unstructured. In an unstructured P2P network, the topology is not optimized with respect to parameters like neighbor-degree, network overlay, physical position of peers and metrics like bandwidth, link-latency, computing, and storage capacities of peers [52]. Moreover, network properties like connectivity, small diameter, specific clustering coefficient, cannot be guaranteed. Hence, in our work, we focus on a structured P2P network to optimize the topology according to given network properties specifications.

As constructing an optimized topology structure using distributed protocols is an arduous task, centralized protocols are preferred [53] which is also the point of view of our work. For a given P2P network structure, there can be different centralized graph generators and selecting a simple, fast and robust generator is of prime importance. In our work, we consider two types of *r-out* graphs i.e. approximate (average r outbound connections per node) and exact (exactly r outbound connections per node).

5 Centralized topology management and maintenance

In this Section, we describe the general architecture of our proposed 2-layer network where the upper layer helps to control the topology of the lower layer in a centralized manner. The lower layer forms the P2P network of the blockchain and implements all the blockchain protocols and their functions except topology control protocols (neighbor discovery, selection, and maintenance).

5.1 General architecture

Our hybrid P2P network is organized into two layers: the upper tier - servers and the lower tier - peers (Fig. 2). In our work, we consider that the blockchain is built on the lower layer of a hybrid P2P network infrastructure deployed over the Internet. The infrastructure is formed by peers and servers. The peer devices can be mobile IoT devices like delivery drones, in-vehicle electronic control units, survey robots, or static IoT devices like weather stations, automatic toll plazas, disaster alerting systems, embedded devices (for various applications).

Each peer has the capability to communicate with other peers that are indicated as neighbors by upper layer servers.

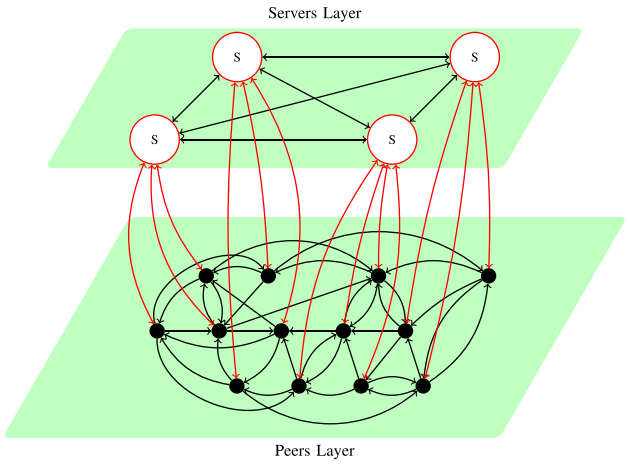


Fig. 2 General architecture of the two layers

The servers' role is to collect information from peers and elect a designated server responsible for selecting and maintaining outbound and inbound neighbors of all peers according to their profiles. We distinguish two types of profiles for peers: contributors and non-contributors. A contributor can be seen as router who allows information forwarding by accepting inbound connections. Non-contributor does not accept inbound connections. Once a peer has information on its neighbors through its local server, it can directly communicate with them.

All communications are done over TCP/IP protocols. Each peer is identified by its IP address and should initiate outbound connections with other peers. Only contributor peers accept inbound connections. To allow inbound connections for contributor peers belonging to private networks, different ways can be applied:

- Setting up explicit TCP port forwarding rules on the contributor (router).
- Using NAT traversal solutions like: Relaying via Rendezvous Servers, Connection Reversal, TCP Hole Punching [54].

This "2-tier architecture" is helpful to satisfy the constraints on full connectivity, easy topology control and increases network flexibility. As the upper layer contains multiple servers acting as mirrors, a failure in the designated server can be resolved by re-selecting a new server from the server pool. Also, as the peers layer (lower layer) can be public and open, its size in terms of the number of participating peers can be large. Its maximum size limit depends on the capacity of the servers layer (upper layer) to support the down layer. Indeed, when the peers layer size increases, the topology control traffic load handled by the servers layer will increase which can negatively affect the servers functioning. Examples of such negative impacts are longer delivery time for control data, traffic bottlenecks, security and even crashes. To minimize these potential troubles, the servers layers should also be scaled up in terms of size and computing capacity on one hand, and on the other hand, the incoming traffic on the servers layer should be well distributed and balanced among the appropriate servers using solutions such as dedicated multiple reverse proxy servers and load balancers. Based on this analysis, the servers layer and its entry points should be dynamically dimensioned and configured according to the peers layer size. Thus, with good dimensioning practice, the size of the P2P network is theoretically unbounded, subject to practical limitations and considerations.

5.2 Upper layer operations

The principal upper layer operations are used to maintain the described architecture and to control the topology of the lower layer.

5.2.1 P2P network topology control

The role of this task is to discover online peers, build and maintain the lower layer P2P network topology. In peer discovery, when a peer is online, it first does the authentication with the associated server (using SE). Then it indicates its available resources (computing power, storage, battery autonomy, etc.) and its profile: contributor (allows inbound connections) or non-contributor (no inbound connections). The associated server then forwards the collected information to the designated server.

Based on the information provided by peers, the designated server builds the topology using a graph generator. To this end, it pseudo-randomly selects inbound and outbound neighbors for each contributor and only outbound neighbors for non-contributor. When a new peer joins or when an old peer leaves the network, designated server updates the network topology accordingly.

5.2.2 Server-pool maintenance

This task consists of (pseudo-randomly) selecting a designated server from the server-pool and maintaining the synchronized view of the topology of the lower layer across server-pool. The server-pool consists of regional servers which serve a specific region independently. The regional server is acting as an intermediary between regional peers and the designated server. It is responsible for collecting information from regional peers, forwarding it to

the designated server and finally informing its regional peers about their neighbors. If any server (designated or regional) fails, another is selected from the server-pool as a replacement.

5.3 Lower layer operations

In this layer, peers are equal and follow a flat hierarchy. However, we distinguish them in two types: contributors (accepting inbound connections) and non-contributors (rejecting inbound connections). The whole blockchain resides in this layer in true sense. As a result, there are no additional security implications arising out of 2-layer structure. Contributing peers support blockchain database, routing, mining and are a source of new information. This makes them indispensable for proper blockchain functioning. Non-contributing peers, on the other hand, are the only source of new information and don't contribute for blockchain's upkeep and maintenance. The functionality of peer discovery and peer selection are delegated to the upper layer.

To keep the blockchain P2P topology (*r-out* digraph) updated, the centralized server(s) should be aware of the liveliness of peer nodes and their outgoing connections state. All peer nodes in the blockchain P2P network send periodic *keep_alive* messages to affirm their liveliness. Each *keep_alive* message contains the emission interval field that represents the time before the transmission of the next *keep_alive* message, and the validity time field that indicates for how long the centralized server(s) must consider the sender alive. In general, the validity time is expressed as a function of the emission interval. For example, when validity time = emission time $\times 2$, the peer node would be still considered alive even if only one of the two consecutive *keep_alive* messages is received. Both time values are configured locally by each peer node according to how it is powered (battery, self-powered or electrical grid), device type (phone, IoT or device), mobility, applications it is running, the expected blockchain performance, etc. :

- If the *keep_alive* emission interval of a given peer node is very short, the centralized servers can quickly detect the offline state and update the network topology. However, the peer will consume more energy. Consequently, for those powered by battery or self-powered (IoT devices, mobile phones, etc.), the life time will decrease. For those powered by electrical grid, the cost will increase. Thus, there is a trade-off between energy consumption and updating the P2P network topology;
- The outbound connections of mobile peer nodes would not be interrupted on the condition that the mobile peer keeps its public IP address (through the Network Address Translation gateway). Consequently, mobility doesn't affect directly the P2P network topology. However, as mobility requires portable power solution, *keep_alive* emission interval should be low/reduced to bring down the power consumption and increase life time;
- The tasks performed by peer nodes on the blockchain may demand high resources such as processing power or high bandwidth links, causing significant battery usage. In such case(s), *keep_alive* emission interval should be increased;
- When a node is offline, its neighbors (inbound connections sources) can detect this event when attempting data exchange with it as no *ack* message is received. Thus, the number of outbound connections for the neighbors is reduced (below the threshold r). Subsequently, they will query the centralized server(s), through outbound connections state messages, to select other neighbor(s). The outbound connections state messages (generated once there is a change) combined with *keep_alive* messages, allow to reduce the duration of the window when the knowledge of the P2P network structure at the centralized server(s) is not exact. During this time window, the *r-out* digraph may become approximate or worse, if the number of peer nodes going offline is high. Consequently, this may directly impact the properties of the original P2P network in terms of connectivity, diameter, clustering, etc.

Further, each peer node should also transmit once per day an outbound connections state message to the centralized server(s) to confirm and or update the knowledge of the centralized server(s) (similar to the bitcoin protocol). This functionality also helps the centralized server(s) to detect anomalies related to misbehaving nodes that detect changes in their outbound connections and do not inform. Consequently, as the server(s) should receive at least one outbound connections state message from each peer by day, some misbehaving nodes can be detected and excluded. The periodicity, once per day, is also to reduce power consumption.

Note: The proposed solution of two layers does not affect the distributed properties of the blockchain P2P network. This is because the P2P network generation is still random and based on the whole network knowledge. In other words, for a given peer, its neighbors are selected randomly among all the network participants and not from a subset/partial network view, unlike the current blockchain P2P networks. In addition, with our proposed solution, the overlay layer can be queried, to select neighbors for a given node from a small subset of randomly selected nodes. This would make the constructed topology more closer to the existing blockchain topology generation techniques.

Further, we recall that the blockchain operations(block proposal, transactions and block propagation, block validation by consensus algorithms) are still per-

formed in the distributed P2P network of the nodes. Only the P2P network topology generation and maintenance operations are ensured by the secure overlay layer to remove the excess network topology control/ maintenance overhead and increase configurability.

5.4 Cost of the centralized topology control

In our architecture, redundant servers and redundant storage are necessary at the servers layer to provide resilience and recoverability. The redundant servers, running the same applications, can be implemented in two ways: mirror cluster and farm cluster. In a mirror cluster, only the primary server is active while the secondary servers remain passive with real-time synchronous replication and failover. In a farm cluster, there is no primary or secondary servers, and all redundant servers are simultaneously active. The incoming traffic is load balanced across all the active servers according to the performance of each server. Both solutions (mirror and farm clusters) can be hosted on-site which requires high cost and effort to buy and maintain the hardware/software of the servers and the network equipments, or off-site by public or private service providers which save upfront hardware costs ($5-$200 against $1000-$3000) and time. It is important to note that the cost of both solutions increases with increasing number of redundant servers, however, "virtualization" may reduce the number of the required physical redundant servers and thus the cost to some extent.

For the redundant storage part, a range of solutions exist such as local/manual backup storage, cloud backup storage and fully managed backup storage. The performance of each technology type is related to its capability to reduce the backup latency, increase the backup reliability and the amount of data recovery. While local/manual backup has high backup latency (few days), medium reliability and a limited amount of data recovery; cloud and fully managed backup solutions offer a short backup latency (few minutes), high reliability and a high amount or entire of data recovery. In our architecture, the servers layer should mainly the store data related to the lower layer such as the list of peers and their state, outbound/inbound connections of each peer, etc. As this data mainly consists of text arranged in simple tables, the magnitude of storage requirement remains low and so its cost, compared to the redundant servers is negligible.

6 Blockchain P2P network modeling

In this Section, we model the P2P network described in the Sect. 5 by a special class of digraphs, named *r-out* digraphs. We discuss three models to generate such random digraphs.

Further, we suppose that the P2P network we model has n peers. Among them, k non-contributor peers exist having *null* inbound connections. The other peers are contributors and accept inbound connections up to a limit γ. Each peer, regardless of its profile, initiates and maintains r outbound connections.

6.1 Definitions and notations

Throughout this paper, we use the following notations and definitions: Let $D = (V, \vec{E})$ be a simple labeled digraph (directed graph), where $V = \{v_1, v_2, \dots, v_n\}$ is the set distinguishable vertices of size n, and $\vec{E} = \{(u, v) : u, v \in V\}$ is the set of directed edges. The digraph D is called simple if it does not contain self-loops (no directed edge between a vertex and itself) or multiple directed edges in the same direction. The set of all in-neighbors and out-neighbors of a vertex u are denoted by $N_D^-(u)$ and $N_D^+(u)$, respectively. Their cardinalities are the in-degree and the out-degree of u, denoted by $deg_D^-(u)$ and $deg_D^+(u)$, respectively. The sum of out-degree and in-degree of u gives its degree, denoted by $deg_D(u)$. The maximum in-degree of D, denoted by $\Delta^-(D)$, is the largest in-degree of its vertices. We define the degree sequence of D as the list of its in-degree and out-degree pairs, i.e.,

$$\left((deg_D^{+(v_1)}, deg_D^{-(v_1)}), \dots, (deg_D^{+(v_n)}, deg_D^{-(v_n)})\right)$$

Since each directed edge in a digraph D increases out-degree of its head and in-degree of its tail by one, we have

$$\sum_{u \in V} deg_D^{+(u)} = \sum_{u \in V} deg_D^{-(u)} = |\vec{E}|. \tag{1}$$

A digraph, D, is called *r-out* digraph if each vertex in V has the same out-degree r. More formally: $\forall u \in V : deg_D^+(u) = r$. Its total number of arcs is indicated by the following Lemma:

Lemma 1 *If D is an r-out digraph then, $|\vec{E}| = rn$.*

Proof Let $D = (V, \vec{E})$ be a simple *r-out* digraph of order n. Based on equation 1, we have:

$$|\vec{E}| = \sum_{u \in V} deg_D^+(u) = rn.$$

□

6.2 Model description and motivation

We model the lower layer P2P network by a simple *r-out* digraph. This model is motivated by the fact that it is sufficient to set $r = 2$ to obtain with high probability a fully

connected network as its size grows to infinity [55]. Consequently, the full connectivity constraint can be satisfied. Also, the justification for selecting a digraph instead of an undirected graph is that connections in our blockchain protocol can be inbound or outbound, based on who initiated the connection. Since each peer should maintain exactly r outbound connections to distinct target peers, the resulting P2P network topology can be represented as an *r-out* digraph. Further, only single inbound and/or outbound connection between two peers is supported, a simple *r-out* digraph satisfies this restriction.

In order to consider non-contributor, we assume that the simple *r-out* digraph contains k, $0 \leq k \leq n-2$, vertices with null in-degree. All the other vertices, representing contributor peers, have up to γ inbound degrees which correspond to the maximum number of inbound neighbors. We denote such digraph by $\mathbb{D}^{\gamma}_{n,k,r\text{-}out}$. The limits on outbound and inbound degrees are mainly to prevent excessive bandwidth occupation.

6.3 Satisfying necessary conditions of feasibility

To generate a simple *r-out* digraph $\mathbb{D}^{\gamma}_{n,k,r\text{-}out}$, some necessary (but not sufficient) conditions on its parameters n, k, r, γ needs to be satisfied for its realization. All the parameters are correlated. In general, n and k are fixed according to the use-case for which the blockchain is used. Based on n and k, range for parameters r and γ are derived and expressed.

For parameters n and k, it is obvious that $n \geq 2$ to create a network of at least 2 participating peers and $0 \leq k \leq n-2$ to have at least 1-*out* digraph. For the parameter r, the total number of directed edges of a *r-out* digraph should be less than or equal to total possible edges of a complete digraph for same value of n and k. As the total number of directed edges of a *r-out* digraph is $r \times n$ (from Lemma 1) and the total possible edges of a complete digraph with n nodes and k non-contributors is $2\binom{n}{2} - k(n-1)$, we find that:

$$r \leq \frac{(n-1) \times (n-k)}{n} \tag{2}$$

For γ, let $d = \big((r, r_1), (r, r_2), \ldots, (r, r_n)\big)$ be the degree sequence of digraph $\mathbb{D}^{\gamma}_{n,k,r\text{-}out}$, where $r = deg^{+}_{\mathbb{D}^{\gamma}_{n,k,r\text{-}out}}(v_i)$ and $r_i = deg^{-}_{\mathbb{D}^{\gamma}_{n,k,r\text{-}out}}(v_i)$ for $v_i \in V(\mathbb{D}^{\gamma}_{n,k,r\text{-}out})$ and $i = 1, \ldots, n$. Our objective is to find the optimized range of γ satisfying $r_i \leq \gamma$ for $i = 1, \ldots, n$. The following Lemma gives such range.

Lemma 2 *$\mathbb{D}^{\gamma}_{n,k,r\text{-}out}$ is realizable if the inbound degree limit γ satisfies*

$$\frac{rn}{n-k} \leq \gamma \leq n-1.$$

Proof As the maximum number of in-neighbors of a contributor peer is $n-1$ (excluding itself), the highest value of γ, $\gamma_{\max}$, is $n-1$. On the other hand, the lowest value of γ, $\gamma_{\min}$, is reached when all the contributors have the same inbound degree. Consequently, $\gamma_{\min}$ can be written as $\sum_{i=1}^{n-k} \gamma_{\min} = r \times n$ and therefore, $\gamma_{\min} = \frac{rn}{n-k}$. □

6.4 Generating simple random *r-out* digraphs

A simple random *r-out* digraph $\mathbb{D}^{\gamma}_{n,k,r\text{-}out}$ is a digraph sampled out from a set of all *r-out* digraphs ($\mathscr{D}^{\gamma}_{n,k,r\text{-}out}$) according to the uniform distribution. For a given vertex, other probability distributions can also be used to fine tune desired properties, like clustering, regularity, smaller diameter. We investigate three models: Binomial, Uniform, and Preferential Attachment. Exact simple *r-out* digraphs can be generated by Binomial and Preferential Attachment models whereas approximate simple *r-out* digraphs (mean out-degree is r) can be generated by Binomial and Uniform models. The rationale behind investigating approximate simple *r-out* digraphs is to show if it is possible to use them to keep the exact *r-out* digraph properties and speed up the topology construction.

6.4.1 Approximate *r-out* digraph generation using binomial model

A binomial model has two parameters, the number of vertices n and the probability p ($0 \leq p \leq 1$) assigned independently to each of $2\binom{n}{2}$ possible directed edge. Let $\mathbb{D}_{n,p}$ be a binomial random digraph. $\mathbb{D}_{n,p}$ is considered as an approximate *r-out* digraph $\mathbb{D}_{n,k,r\text{-}out}$ if the average out-degree of its vertices is r and the number of vertices having null in-degree is k. Let us consider $\mathbb{D}_{n,k,p}$ as an approximation of $\mathbb{D}^{\gamma}_{n,k,r\text{-}out}$. The average number of directed edges of $\mathbb{D}_{n,k,p}$, $|\overrightarrow{E}(\mathbb{D}_{n,k,p})|$, should be equal to the number of edges of $\mathbb{D}_{n,k,r\text{-}out}$, $|\overrightarrow{E}(\mathbb{D}^{\gamma}_{n,k,r\text{-}out})|$. To ensure this equality, the probability p is given in the following Lemma:

Lemma 3 *The equality $|\overrightarrow{E}(\mathbb{D}_{n,k,p})| = |\overrightarrow{E}(\mathbb{D}_{n,k,r\text{-}out})|$ holds true when $p = \frac{rn}{(n-1)(n-k)}$.*

Proof As the total number of directed edges of complete digraph on n nodes and k as non-contributors among them is $2\binom{n}{2} - k(n-1)$, the average number of directed edges of $\mathbb{D}_{n,k,p}$ is: $|\overrightarrow{E}(\mathbb{D}_{n,k,p})| = \left(2\binom{n}{2} - k(n-1)\right) \times p$. We also have $|\overrightarrow{E}(\mathbb{D}_{n,k,r\text{-}out})| = r \times n$ (from Lemma 1). By resolving the equality, the probability p can be expressed as: $p = \frac{rn}{(n-1)(n-k)}$ □

As the average out-degree of nodes is $p \times (n-k)$, the probability $p = \frac{rn}{(n-1)(n-k)}$ ensures that the average out-degree of nodes tends to r when the number of nodes n goes to ∞. Thus, $\mathbb{D}_{n,k,p}$ sustains the concept of *r-out* digraph at average.

Let D_0 be a given digraph generated by the model $\mathbb{D}^{\gamma}_{n,k,r\text{-}out}$. An approximate digraph of D_0 can be generated by the $\mathbb{D}_{n,k,p}$ model with the following probability:

$$\mathbb{P}(\mathbb{D}_{n,k,p} = D_0) = p^{r\times n} \times (1-p)^{(n-1)(n-k)-rn}. \tag{3}$$

The generation process of an approximate random digraph $\mathbb{D}^{\gamma}_{n,k,r\text{-}out}$ using a $\mathbb{D}_{n,k,p}$ model consists of multiple iterations. Starting with an empty digraph of n nodes and *zero* arcs, performing $2\binom{n}{2} - k(n-1)$ independent Bernoulli trails to add arcs each with probability $p = \frac{rn}{(n-1)(n-k)}$.

Based on this model, the resulting P2P network may not follow the fixed input parameters (r, k, γ). It may have some peers with more/less r outbound connections or more than γ inbound connections and non-contributors accepting inbound connections. If we accept this network topology, we don't strictly enforce the fixed parameters. However, it benefits from the *r-out* digraph properties. The following Lemma shows that each of the *r-out* digraphs can be generated with same probability when using binomial random digraph generator with out-degree constraints.

Lemma 4 *A random digraph $\mathbb{D}_{n,p}$ subject to the constraint that each node has exactly the same out-degree r, is equally likely to be one of the $\binom{n-1}{r}^n$ digraphs of $\mathscr{R}$-out.*

Proof Let D_0 be an *r-out* digraph. Based on the remark that $\{\mathbb{D}_{n,p} = D_0\} \subseteq \{\mathbb{D}_{n,p} \in \mathscr{R}\text{-out}\}$, we have

$$\begin{aligned}\mathbb{P}(\mathbb{D}_{n,p} = D_0 | \mathbb{D}_{n,p} \in \mathscr{R}\text{-out}) &= \frac{\mathbb{P}(\mathbb{D}_{n,p} = D_0)}{\mathbb{P}(\mathbb{D}_{n,p} \in \mathscr{R}\text{-out})} \\ &= \frac{\left(p^r \times p^{(n-1)-r}\right)^n}{\left(\binom{n-1}{r} p^r \times p^{(n-1)-r}\right)^n} \\ &= \binom{n-1}{r}^{-n}\end{aligned}$$

□

6.4.2 Approximate *r-out* digraph generation using uniform model

This graph model has two parameters, the number of vertices n and the number of directed edges m, where m takes values from zero to the maximum number of directed edges derived from complete simple digraphs. The digraph also contains k vertices with *zero* in-degree. Therefore, the maximum number of directed edges are $2\binom{n}{2} - k(n-1)$. This can be simplified further into the form $(n-1)(n-k)$. Therefore, the number of directed edges m is within the interval $\{0, (n-1)(n-k)\}$. Let $\mathscr{D}_{n,m}$ be the family of all possible digraphs having n nodes and exactly m directed edges, and $\mathbb{D}_{n,m}$ be a random digraph chosen uniformly at random from $\mathscr{D}_{n,m}$. The cardinality of $\mathscr{D}_{n,m}$ is $|\mathscr{D}_{n,m}| = \binom{2\binom{n}{2}}{m}$. Each digraph $D \in \mathscr{D}_{n,m}$ is chosen uniformly with equal probability:

$$\mathbb{P}(\mathbb{D}_{n,m} = D) = \begin{cases} \frac{1}{\binom{2\binom{n}{2}}{m}} & \text{if } |\vec{E}| = m \\ 0 & \text{if } |\vec{E}| \neq m \end{cases} \tag{4}$$

The digraph generation process consists of multiple iterations, starting with a graph of n vertices and *zero* directed edges, adding arcs till total edges are m thereby the resulting graph is equally likely to be one of the $\mathscr{D}_{n,m}$ set.

From Lemma 1, we know that an *r-out* digraph has exactly $r \times n$ directed edges. By taking $m = rn$, the set of all *r-out* digraphs can be sampled using the uniform random directed graph $\mathbb{D}_{n,rn}$. The cardinality of this set can be given by the following Lemma:

Lemma 5 *Let $\mathscr{R}$-out be the family of all possible r-out digraphs. Then, $|\mathscr{R}\text{-out}| = \binom{n-1}{r}^n$*

Proof For a given vertex, the number of ways to choose r out-neighbors from $n-1$ possible candidate is given by $\binom{n-1}{r}$. As the out-neighbor sets of all vertices(n) are mutually independent, the total number of all possible *r-out* digraphs can be given by $\binom{n-1}{r}^n$. □

Based on Lemma 5 and equation 4, the random digraph $\mathbb{D}_{n,r\times n}$ can be any *r-out* digraph with the following probability:

$$\mathbb{P}(\mathbb{D}_{n,rn} \in \mathscr{R}\text{-out}) = \frac{|\mathscr{R}\text{-out}|}{|\mathscr{D}_{n,rn}|} = \frac{\binom{n-1}{r}^n}{\binom{2\binom{n}{2}}{rn}} \tag{5}$$

The following Lemma shows that each *r-out* digraphs can be generated with same probability when using uniform random digraph generators with out-degree constraints.

Lemma 6 *A random digraph $\mathbb{D}_{n,r\times n}$ subject to the constraint that each node has exactly the same out-degree r, is equally likely to be one of the* $\binom{n-1}{r}^n$ *digraphs of* $\mathscr{R}$-out.

Proof Let D_0 be an *r-out* digraph. Based on the remark that $\{\mathbb{D}_{n,rn} = D_0\} \subseteq \{\mathbb{D}_{n,rn} \in \mathscr{R}\text{-out}\}$, we have

$$\mathbb{P}(\mathbb{D}_{n,r\times n} = D_0 | \mathbb{D}_{n,r\times n} \in \mathscr{R}\text{-out}) = \frac{\mathbb{P}(\mathbb{D}_{n,rn} = D_0)}{\mathbb{P}(\mathbb{D}_{n,rn} \in \mathscr{R}\text{-out})}$$

$$= \frac{\binom{2\binom{n}{2}}{rn}^{-1}}{\frac{\binom{n-1}{r}^n}{\binom{2.\binom{n}{2}}{rn}}}$$

$$= \binom{n-1}{r}^{-n}$$

□

6.4.3 Observations

Our objective is to find a simple random model generator to sample *r-out* digraphs. From Lemmas 6 and 4, we can see that both $\mathbb{D}_{n,r\times n}$ and $\mathbb{D}_{n,p}$ under the constraint that all vertices have the same out-degree (r), are equally probable in distribution and the resulting *r-out* digraph is selected uniformly at random from $\mathscr{R}$-out (the set of all *r-out* digraphs). In other words, uniform and binomial models subject to the above constraint, can generate all possible *r-out* digraphs, each being equally likely. Figure 3 illustrates such generation mechanisms from the sample space of all possible digraphs on n vertices. We know from random graph theory that random digraphs $\mathbb{D}_{n,rn}$ and $\mathbb{D}_{n,p}$ have the same asymptotic behavior for most of properties when the mean number of arcs in $\mathbb{D}_{n,p}$ is equal or close to $r \times n$ [56]. In other words, when $2\binom{n}{2}p = r \times n$ or $p = \frac{r}{n-1}$. In practice, $\mathbb{D}_{n,\frac{r}{n-1}}$ is easier to handle then $\mathbb{D}_{n,rn}$, i.e. approximate *r-out* graph generation using binomial model is computationally easy compared to its uniform model counterpart.

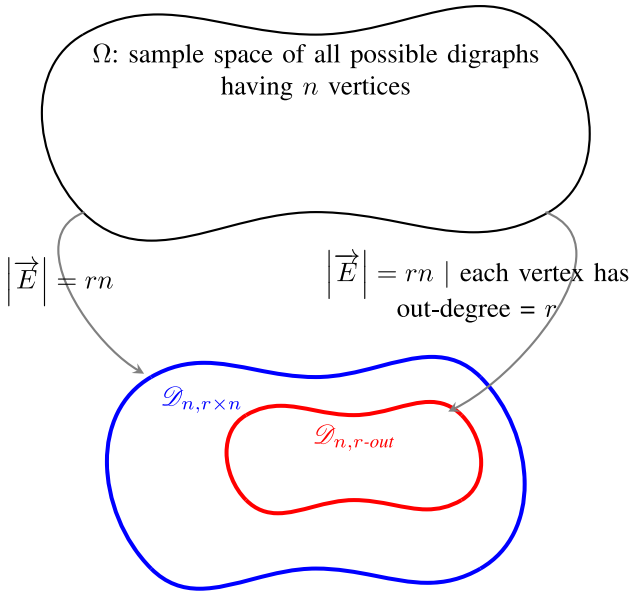


Fig. 3 Sample spaces for approximate digraphs

Based on the above analysis, the binomial random digraph model $\mathbb{D}_{n,p=\frac{r}{n-1}}$ subject to the constraint that all vertices have the same out-degree (r) is selected to sample our *r-out* digraphs rather than the uniform random digraph model $\mathbb{D}_{n,rn}$.

6.4.4 Exact *r-out* digraph generation using binomial model

To generate an exact *r-out* digraph $\mathbb{D}^{\gamma}_{n,k,r\text{-}out}$ by the $\mathbb{D}_{n,k,p}$ model, additional constraints are necessary. When a Bernoulli trial results in adding a directed edge between pair of vertices (the first vertex is called tail while the second one is called head), three additional independent conditions should be filled: a) the tail is a contributor, b) the tail has less than r out-neighbors, and c) the head has less than γ in-neighbors. Consequently, the probability to add a directed edge is a result of multiplying four probabilities: p, $\mathbb{P}(\text{tail} = \text{contributor})$, $\mathbb{P}(deg^+_D(\text{tail}) < r)$ and $\mathbb{P}(deg^-_D(\text{head}) < \gamma)$. The probability p is given by $\mathbb{D}_{n,k,p}$ model as $\frac{rn}{(n-1)(n-k)}$ (from Lemma 3). The probability for a vertex to be contributor is $\frac{n-k}{n}$. As the degree distribution is binomial, the probabilities $\mathbb{P}(deg^+_D(\text{tail}) < r)$ and $\mathbb{P}(deg^-_D(\text{head}) < \gamma)$ can be given respectively by $\sum_{i=0}^{r-1}\binom{n-k}{i}p^i(1-p)^{(n-k-i)}$ and $\sum_{i=0}^{\gamma-1}\binom{n-1}{i}p^i(1-p)^{(n-k-i)}$.

Let D_0 be a given digraph generated by the model $\mathbb{D}^{\gamma}_{n,k,r\text{-}out}$. The probability to sample D_0 using the $\mathbb{D}_{n,k,p}$ model with the additional constraints can be written as:

$$\mathbb{P}(\mathbb{D}_{n,k,p} = D_0) = \Bigg(p \times \frac{n-k}{n} \times \sum_{i=0}^{r-1} \binom{n-k}{i} p^i (1-p)^{(n-k-i)} \times \sum_{i=0}^{\gamma-1} \binom{n-1}{i} p^i (1-p)^{(n-k-i)} \Bigg)^{r \times n} \times \Bigg(1 - \Bigg[p \times \frac{n-k}{n} \times \sum_{i=0}^{r-1} \binom{n-k}{i} p^i (1-p)^{(n-k-i)} \times \sum_{i=0}^{\gamma-1} \binom{n-1}{i} p^i (1-p)^{(n-k-i)} \Bigg] \Bigg)^{(n-1)(n-k)-rn}. \tag{6}$$

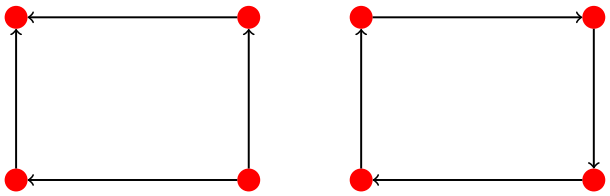
Based on Eqs. 3 and 6, we can see that, at average, the probability to generate an exact *r-out* digraph is less than the probability to generate an approximate one. In other words, at average, generation of an approximate *r-out* digraph is quicker compared to its exact counterpart (Section 8.5). Consequently, the generation process for an exact $\mathbb{D}^{\gamma}_{n,k,r\text{-}out}$ using the $\mathbb{D}_{n,k,p}$ model needs *at least* (best case scenario) the same number of iterations as for generating the approximate $\mathbb{D}^{\gamma}_{n,k,r\text{-}out}$. However, these iterations can largely be reduced by avoiding vertices whose in/out degrees are saturated w.r.t threshold. Subsequently, generation time is greatly minimized (Section 8.5). Another advantage of this model is that the P2P network inherits all the properties of $\mathbb{D}^{\gamma}_{n,k,r\text{-}out}$ digraphs.

6.4.5 Exact *r-out* digraph generation using preferential attachment

In random digraphs, preferential attachment refers to the process of constructing a digraph by adding oriented edges, one at a time. At each step, higher degree vertices are more likely to be incident to the next selected oriented edge. This mechanism can be applied to generate random $\mathbb{D}^{\gamma}_{n,k,r\text{-}out}$ digraphs. For this, the candidate tail and head vertices should also satisfy the conditions on out-degree and in-degree respectively. The proposed algorithm is illustrated below:

Algorithm 1 Preferential Attachment Random $\mathbb{D}^{\gamma}_{n,k,r\text{-}out}$ Digraph Generation.

Output: Random $\mathbb{D}^{\gamma}_{n,k,r\text{-}out}$ digraph;
Initialization
Create an empty digraph D with n vertices and k non-contributors;
Each vertex i in digraph D is assigned an initial weight α_i;
LOOP Process
while there exists vertices in D with out-degree $< r$ **do**
  Select a vertex, u, having $deg^{+}_{D}(u) < r$, uniformly at random from D;
  if there exists vertice(s) in D with in-degree $< \gamma$ **then**
    Select a vertex, v, having $deg^{-}_{D}(v) < \gamma$ **and** maximizing the probability: $\frac{\alpha_v}{\sum_{x=1}^{n} \alpha_x}$ **and** $v \notin k$;
    Form an edge $u \rightarrow v$ in D;
    Increment the weight of v: $\alpha_v = \alpha_v + 1$;
  else
    Digraph D unrealisable with current *seed* value;
    Exit;
  end if
end while
$\mathbb{D}^{\gamma}_{n,k,r\text{-}out} = D$;
return $\mathbb{D}^{\gamma}_{n,k,r\text{-}out}$;



(a) Weakly connected digraph

(b) Strongly connected digraph

Fig. 4 Connectivity in digraph

The resulting random $\mathbb{D}^{\gamma}_{n,k,r\text{-}out}$ digraph's distribution depends on the initial weight vector $\alpha = (\alpha_0, \alpha_1, \ldots, \alpha_n)$. Further, when the elements of the vector α are same and high (tends to ∞), the heads selection follow a random uniform distribution. As the tails selection also follows a uniform distribution, all the sampled digraphs, in this case, have the same uniform distribution. In contrast, when the elements of the vector α are same and low (tends to 0), head nodes are more likely selected as their in-degree increases (tends to γ) and bigger clusters appear more frequently. Further, a vertex with higher α value is more likely to be head of a higher number of peers (at most γ peers).

The initial weights given to each peer should be motivated by the P2P network topology control. When P2P network has some peers with more resources (e.g., miners, full nodes), they are more likely to have the capacity to handle higher in-degree. Such peers can be modeled by corresponding higher initial weight. When peers have equal resources, higher and equal initial weights are more appropriate to generate a uniform digraph.

7 Random *r-out* digraph properties

In this Section, we explain the following properties which play an important role in information dissemination and optimization of a blockchain P2P network.

7.1 Network connectivity

We distinguish between two types of connectivity in digraphs: weak and strong. A digraph is said to be weakly connected if its underlying undirected graph is connected. An underlying undirected graph is obtained by replacing each directed edge by an undirected edge. A digraph is said to be strongly connected if every vertex is reachable from every other vertex, i.e. there exist a path in each direction for each pair of vertices in the digraph (See Fig. 4).

In our work, we focus only on *weak connectivity* because in blockchain P2P network, even though the protocols make a distinction between inbound/outbound connections (based on who initiated the connection), the

information flow on both types of connection is bidirectional. This makes the P2P network undirected when information flow is considered. As we consider two types of *r-out* digraphs i.e., approximate and exact, the connectivity constraints are different for both.

7.1.1 Weak connectivity of exact *r-out* digraphs

For weak connectivity of exact *r-out* digraph, we use the theorem shown in [55] which states that:

$$\lim_{n\to\infty} \mathbb{P}(\textit{r-out digraph is weakly connected}) = \begin{cases} 0 & \text{if } r = 1 \\ 1 & \text{if } r \geq 2 \end{cases} \tag{7}$$

In other words, an *r-out* digraph is weakly connected for $r \geq 2$ with a very high probability when the number of nodes is high. This theorem is valid for all *r-out* digraphs and hence it can be applied to our $\mathbb{D}^{\gamma}_{n,k,r\text{-}out}$ digraph. This greatly simplifies the connectivity constraint on the exact *r-out* P2P network.

7.1.2 Weak connectivity of approximate *r-out* digraphs

W.r.t graph theory, as these digraphs are an approximation of exact *r-out* digraphs, the theorem of weak connectivity for exact *r-out* digraphs cannot be directly applied. Hence, to derive the conditions of weak connectivity for approximate *r-out* digraphs, we use connectivity conditions of binomial random graphs. By ignoring directed edges from a random $\mathbb{D}_{n,k,p}$ digraph, we obtain an undirected underlying random graph $\mathbb{G}_{n,k,p'}$ where the probability p' is computed based on p. This observation is illustrated by the following Lemma:

Lemma 7 *The underlying random graph of* $\mathbb{D}_{n,k,p=\frac{rn}{(n-1)(n-k)}}$ *is* $\mathbb{G}_{n,k,p'=p(2-p)}$.

Proof Let u and v two vertices from $V(\mathbb{D}_{n,k,p=\frac{rn}{(n-1)(n-k)}})$. We consider three events: e={there is an undirected edge between u and v}, e_1={there is a directed edge from u to v}, and e_2={there is a directed edge from v to u}. We know from $\mathbb{D}_{n,k,p=\frac{rn}{(n-1)(n-k)}}$ that the events e_1 and e_2 are independent and $\mathbb{P}(e_1) = \mathbb{P}(e_2) = p$. The event e can be written as $e_1 \cup e_2$. Therefore:

$$\begin{aligned}\mathbb{P}(e) = \mathbb{P}(e_1 \cup e_2) &= \mathbb{P}(e_1) + \mathbb{P}(e_1) - \mathbb{P}(e_1)\mathbb{P}(e_2) \\ &= 2p - p^2 = p(2-p)\end{aligned}$$

By applying the previous analysis for all pairs of vertices, we find that they are assigned the same probability value $p' = p(2-p)$. As undirected edges are independent, the resulting $\mathbb{G}_{n,k,p'=p(2-p)}$ graph has a binomial distribution. □

We know from connectivity of random binomial graphs, that the function $\frac{\ln(n)}{n}$ is a threshold function for the disappearance of isolated vertices and so for connectivity in $\mathbb{G}_{n,k,p'}$. In other words, $\mathbb{G}_{n,p'}$ is connected if:

$$p' > \frac{\ln(n)}{n}. \tag{8}$$

By replacing p' with $p(2-p)$ (from Lemma 7) and p with $\frac{rn}{(n-1)(n-k)}$ (from Lemma 3), the inequality 8 becomes:

$$\frac{n^2}{(n-1)^2(n-k)^2}r^2 - \frac{2n}{(n-1)(n-k)}r + \frac{\ln(n)}{n} < 0. \tag{9}$$

The range of possible solutions of inequality 9 which satisfy the conditions of feasibility can be expressed as:

$$r_c = \frac{(n-1)(n-k)}{n}(1 - \sqrt{1 - \ln(n)/n}) < r \leq n - 1 = r_{\max}. \tag{10}$$

The above equation is important as it establishes a relation between connectivity and the number of outbound connections r. This relation helps to model a connected blockchain P2P network with minimum possible outbound connections r_c (hence minimum resources).

7.2 Diameter

The diameter of a graph/digraph is the smallest-longest path between the vertices i.e., the maximum eccentricity of any vertex in the graph or the greatest distance between any pair of vertices in the graph. For optimally modeled blockchain P2P network, smaller diameter helps reducing information dissemination time. W.r.t diameter, exact and approximate *r-out* digraphs have the same asymptotic upper bound $\mathcal{O}(\log n)$ [57, 58]. It means, there exists a constant C_r such that the diameter $\leq C_r \times \log n$ with high probability when n tends to ∞ (Section 8.3).

7.3 Clustering coefficient

In graph theory, a clustering coefficient (CC) is a measure of the degree to which nodes in a graph tend to cluster together. This measure can be calculated for each vertex (local) and for the whole digraph (global). The local clustering coefficient of a vertex i is defined as the fraction of all neighbors of i that are also neighbors among themselves(triangles). It is obvious that if a vertex has less than two neighbors, its local coefficient is zero. The local coefficient clustering of a node i is given by: $C_i = \frac{|\{(j,l)\in\vec{E}(\mathbb{D}^{\gamma}_{n,k,r\text{-}out}): j,l\in N_D(i)\}|}{deg_D(i)(deg_D(i)-1)}$. The global clustering coefficient is the average of all local clustering coefficient of each participating vertex is given by: $CC = \frac{1}{n}\sum_{i=1}^{n} C_i$. As the value of CC alone is not sufficient to determine that the

digraph is highly or lowly clustered, a comparison with oriented edge density metric (ρ) is necessary. If CC is much higher than ρ, the digraph is considered highly clustered. Otherwise, the digraph is considered lowly clustered. The directed edge density is calculated by using the ratio between current number of directed edges and the total number of all possible edges. For approximate and exact $\mathbb{D}^{\gamma}_{n,k,r\text{-}out}$ digraphs, the oriented edge density is given by $\rho = \frac{r \times n}{(n-1)(n-k)}$.

7.4 Other properties

Following are some other properties which are important in improving information dissemination and optimization of blockchain P2P network:

- *Graph Generation Time*: It is the amount of time taken to generate a map of the given P2P network graph. This time is important when centralized protocols for topology construction are used. For distributed protocols, this property is irrelevant.
- *Maximum Inbound Connections*: It is the maximum inbound connections any node within the given P2P network has i.e., the node having the highest individual clustering coefficient. This property closely follows the clustering coefficient i.e., for higher clustering coefficient value, one can roughly expect a higher value of maximum inbound connection for a node within the given blockchain P2P network.

8 Simulation & results

In this Section, we first explain our simulation techniques and then summarize our findings in brief for all properties. Armed with these simulated trends, one can model any given blockchain P2P network with optimal parameter values.

8.1 Energy consumption analysis

Synopsis: Our proposition of centralized topology control consumes less energy than the prevalent distributed topology control. Our solution is more adapted for energy constrained peers.

Methodology: We estimate the average energy consumption required for topology control via averaging total no. of topology control packets exchanged per node in the time frame of 24 hours. We calculate this energy consumption for peers using bitcoin blockchain as a reference example for distributed topology control. Next, we compare it with energy consumption of our proposed centralized topology control method under similar settings.

In general, the total number of messages in a network of n nodes, each having r outbound connections, in Δt hours, can be given as (*number of messages/hour/node/connection*)$\times n \times r \times \Delta t$.

For the calculation of bitcoin topology control energy consumption, we analyze all the cases when a given bitcoin peer transmit/receive control message(s):

1. When a node (new or old) establishes a connection with a remote node, it transmits its address with *addr* message.
2. Each node (new or old) broadcast every 24 hours its own address with an *addr* message to all connected nodes.
3. When node (new or old) receives an *addr* message (solicited/unsolicited), it forwards it (under conditions that entries in its address database are < 10) to two neighbors as an unsolicited *addr* message.
4. When a node (new or old) establishes a connection with a remote node, it can transmit a *getaddr* message.

When mapping bitcoin topology [3], it is observed that there are 10000 nodes approximately with 1% of it (100 nodes) being replaced every hour. Considering the P2P network topology (see Sect. 2), bitcoin follows an *r-out* topology with $r = 8$ (see Table 1). Hence, for bitcoin, the total number of transmitted topology control messages:

- Case 1: No. of new nodes/reconnecting nodes every 24 hours*8 (24*100*8).
- Case 2: Total nodes*8 per day (10000*8).
- Case 3: ((Total nodes-1)*2)*(Case 1 + Case 2).
- Case 4: Assuming new/reconnecting nodes send at least 1 *getaddr* and get 1 *addr* response every 24 hours to complete their database (24*100*8*2).

Total messages/24 hours = Case 1 + Case 2 + Case 3 + Case 4

Average no. of messages per peer per day = Total messages per day / total peers.

Next, we also analyze all the cases for our proposed solution, when a peer transmit/receive control message(s) (vertically), for a similar network setting and size as that of bitcoin network and for same number of $r = 8$ outbound connections with 1% of nodes being replaced every 1 hour:

1. When a new node arrives in the network, it transmits to the server(s) *get_neighbors* message. Then, it will receive a set of *neighbors* message. Since 1% nodes are replaced every hour in the bitcoin network [3], we assume the same number for our proposed solution.
2. In our simulation, we have considered devices powered by batteries with few hours of work. To reduce the time window when the centralized server(s) is/are not

aware of offline peers without much overhead, we set the *keep_alive* emission interval and validity time to 30 minutes. Of course, *keep_alive* emission interval and validity time are configurable and we will deeply investigate this issue in our future work.

3. If a peer should be connected to r-neighbors and the current outbound connections are less than the critical threshold, then it will transmit *get_neighbors* request message again to the server(s). Consequently, it will receive a set of *neighbors* message from the server(s). Since this number is difficult to estimate and since 1% of network nodes are replaced every hour [3] in bitcoin network, we assume the number of nodes to have less than r outbound connections to be $1\% \times r$ assuming an *r-regular* digraph i.e., each node has equal number of inbound-outbound connections ($r_{out} = r_{in}$).
4. Every peer transmits once per day to the server(s) the status of its outbound connections. The objective of this task is to update *r-out* digraph once per day.

The total number of transmitted control messages using our solution are:

- Case 1: No. of new nodes/reconnecting nodes every 24 hours (24*100*2).
- Case 2: Total nodes*2*24 messages per day (24*10000*2).
- Case 3: Total nodes*probability to loose at least $r-2$ connections at a time per day*2 (24*10000*1%*2).
- Case 4: Total nodes (10000).

Average no. of messages in our solution per peer per day = Total vertical messages per day (Case 1 + Case 2 + Case 3 + Case 4)/ total peers.

For both cases, energy consumption per peer per day due to the control topology management = average no. of messages*energy consumption per message. As the total energy consumption per peer is directly proportional to the total number of control topology messages exchanged (assuming energy consumption per message is nearly constant for empirical analysis), the energy efficiency of our proposed centralized solution compared to distributed approach can be easily affirmed through the simulation results presented in Fig. 5.

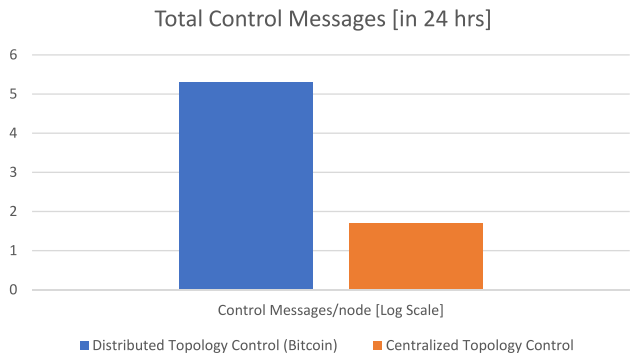


Fig. 5 Control Messages per 24 hours per node, Distributed vs Centralized

In Fig. 5, the x-axis represents the two approaches of topology control i.e. Distributed (Bitcoin) and Centralized. The y-axis is *log-scaled* and represents the total control messages sent per 24 hours per node for both distributed and centralized approaches. Each bitcoin node sends approximately $\approx 200k$ messages per day for topology control and maintenance alone while with our proposed solution, each node sends ≈ 54 messages per day. From Fig. 5, the centralized solution for topology control is $\approx 3.7k$ times energy efficient compared to the distributed approach. The overall efficiency of the centralized solution slightly reduces when power consumption of upper tier nodes (Section 5.1) is also considered. For both cases, the average throughput per peer per day for topology control management = average no. of messages per peer per day * average size of topology control message.

8.2 Other simulation parameter and synopsis

The results presented in this Section are a part of a larger group of simulation results. Therefore it is imperative to summarize our findings. For our *r-out* digraph generation techniques, we experimented with following types of models:

- Approximate *r-out* digraphs: Erdős-Rényi digraphs (Binomial)
- Exact *r-out* digraphs: Preferential Attachment digraphs ($\alpha = 1, 10, 20$), Binomial Distribution digraphs

The digraphs generated were evaluated under 5 broad parameters, viz., Diameter, Connectivity, Graph Generation Time, Degree of Clustering, Maximum Inbound Connections. These parameters helped us to find the best suitable algorithm for optimally modeled blockchain P2P network.

Table 2 Comparison of different types of digraphs, PA = Preferential Attachment

Digraph	Connectivity	Clusters*	Max IC*	Gen. Time*
Erdős-Rényi	3	3	1	4
PA ($\alpha = 1$)	5	2	2	1
PA ($\alpha = 10$)	5	5	3	1
PA ($\alpha = 20$)	5	5	4	1
Binomial Exact	5	5	5	5

For simulation, each type of sample digraph was generated varying the number of participating nodes n, the number of non-contributors k (%), and r outbound connections per node. This sampling was repeated 100 times, varying number of nodes n from 1000 to 10000 (step size = 1000). The percentage of wallets k was varied from 0 to 75% (step size = 25%). The outbound connections were varied with $r = 2, 4, 6$. The tools used to accomplish these simulations were Python 3.2 with NetworkX Library 2.1.

For easy understanding of each digraph's performance across different parameters, Table 2 illustrates the scores for each individual digraph type under each parameter. The higher the score, the more suitable the digraph type is. (* *indicates a higher score for lower entity value. Diameter is not compared as relative variation was statistically insignificant.*)

From Table 2, we can conclude that, within our simulation range, Binomial Exact *r-out* digraph excels across all categories with the highest score followed by Preferential Attachment with high α value. From our simulations, we also found that at higher α value, the Preferential Attachment Exact *r-out* digraph has properties similar to Binomial Exact *r-out* digraph.

8.3 Diameter

The variation of the diameter was observed varying the values of total nodes n, the percentage of non-contributors k and the outbound connections (r, above the critical value r_c). Synopsis of the simulated trends is presented in the graph below. To summarize it was found that when increasing the number of participating nodes n in the network, the diameter increased till a maximum threshold C_r, as indicated in Sect. 7.2. Contrarily, increasing the outbound connections r reduced the diameter. Increasing the number of non-contributors k had a similar effect on reducing the diameter, however, it was more drastic.

In Fig. 6, the percentage of non-contributors and number of outbound connections are varied for different network size. We can see that for the same percentage of non-contributors k across various network sizes, the diameter reduces by at least 1 when outbound connections r are increased by 2. This is because average connection probability increases when r is increased thereby making the digraph more densely connected. Further, for the same number of outbound connections r across various network sizes n, the diameter decreases at least by 2 when the percentage of non-contributors k is increased from 25% to 75%. This is because of the clustering effect (contributors at the center and non-contributors at the periphery), as a result of more non-contributors. Each individual plot saturated at a maximum upper bound value.

8.4 Network connectivity

With reference to Sect. 6, the network connectivity was tested for both types of *r-out* digraphs, i.e., Approximate and Exact. For the Exact *r-out* digraphs, we obtained a connected digraph when $r \geq 2$. For the Approximate *r-out* digraphs, when $r > r_c$ (from Eq. 10), we obtained a connected digraph with a very high probability ($\geq 90\%$).

Figure 7 depicts the histogram of Approximate *r-out* digraphs for $r > r_c$ and $r < r_c$. There are two peaks visible showing that most of digraphs are disconnected when $r < r_c$ and most of the digraphs generated were connected when $r > r_c$. Figure 8 shows the relative connectivity for the Approximate and Exact *r-out* digraph. The values of r tested were 2, 4 and 6. Binomial and Preferential Attachment digraphs ($\alpha = 1, 10, 20$) were simulated for Exact *r-out* digraphs and they were connected across

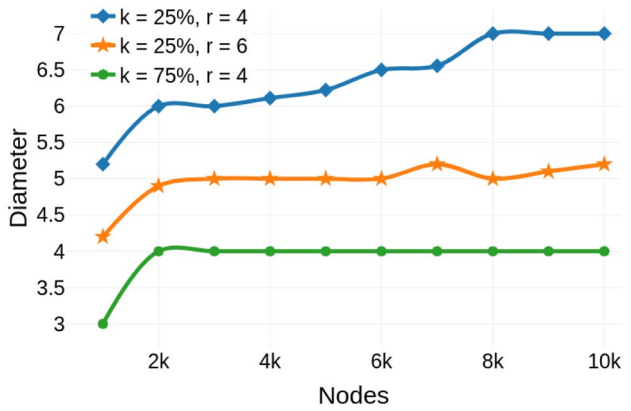


Fig. 6 Diameter Variation

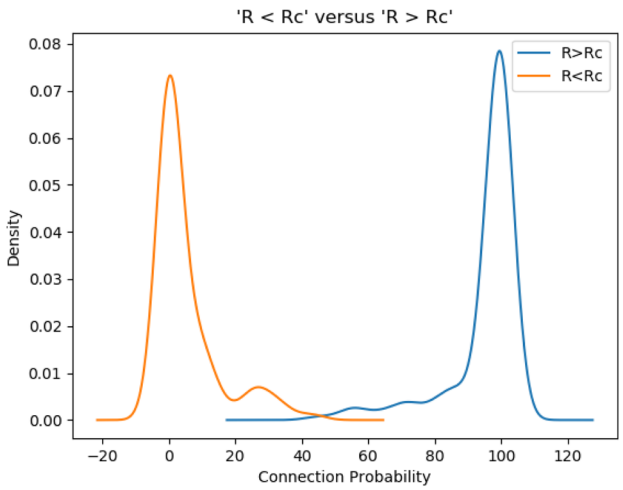
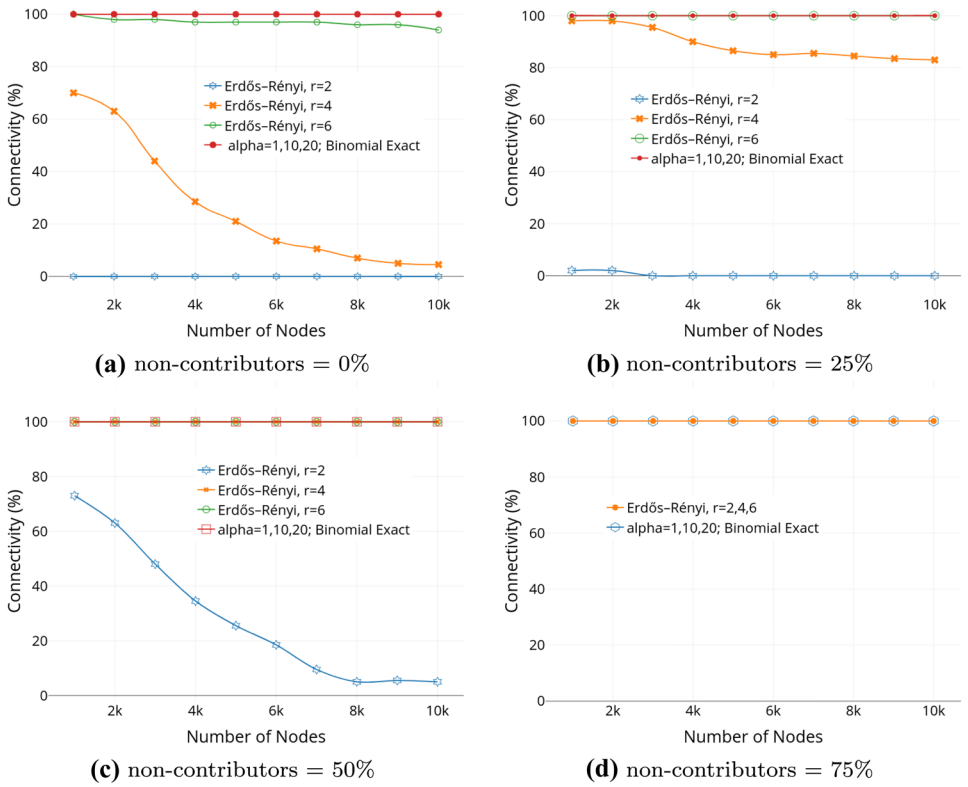


Fig. 7 Connectivity histogram for approximate *r-out* digraphs

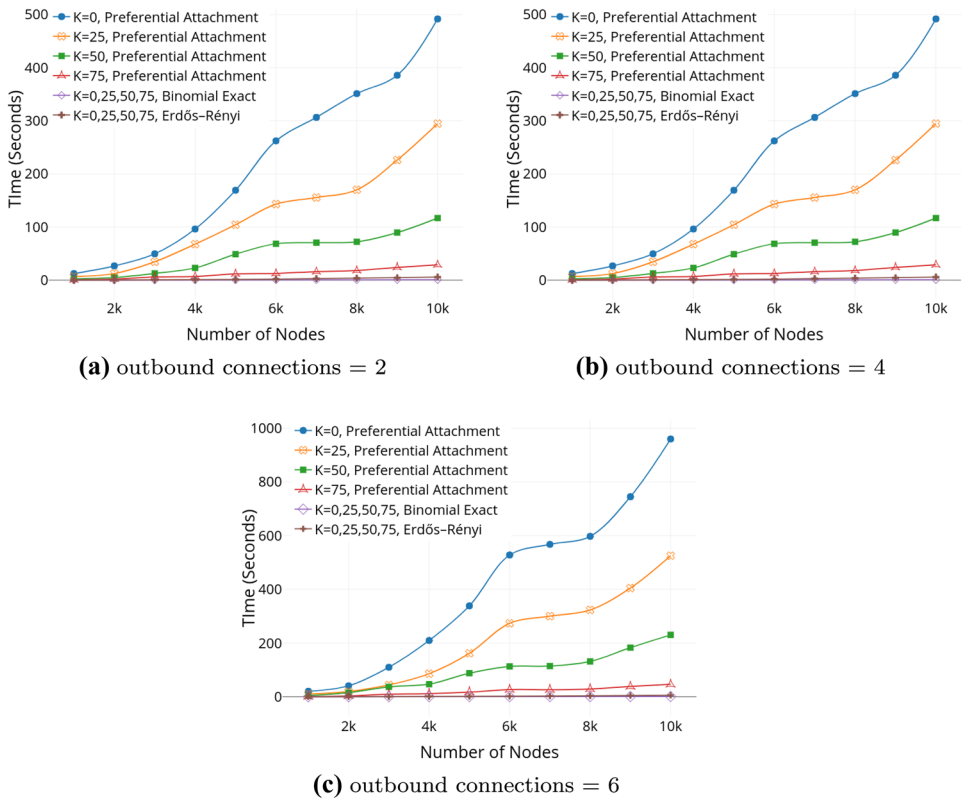


(a) non-contributors = 0% **(b)** non-contributors = 25%

(c) non-contributors = 50% **(d)** non-contributors = 75%

Fig. 8 Connectivity of *r-out* digraphs

all the simulations. The Approximate *r-out* digraphs (Erdős-Rényi) were connected with more probability when r value increased or when the number of wallets (non-contributors k) increased (attesting the value of r_c from Eq. 10 of Sect. 7.1.2).

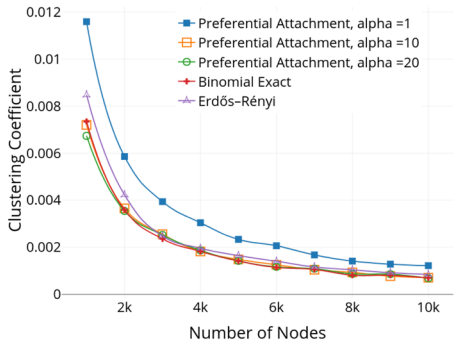


(a) outbound connections = 2 **(b)** outbound connections = 4

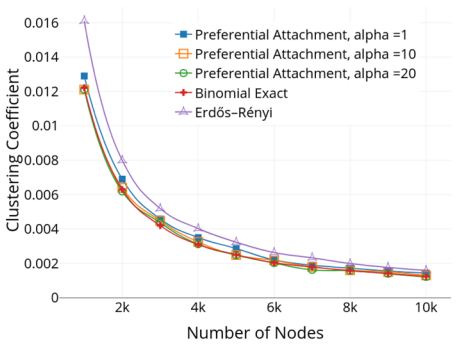
(c) outbound connections = 6

Fig. 9 Graph generation time (K expressed in %)

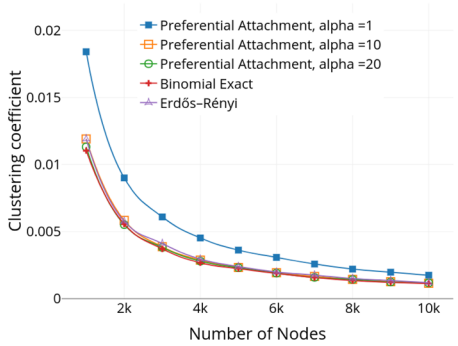
Fig. 10 Clustering Coefficient



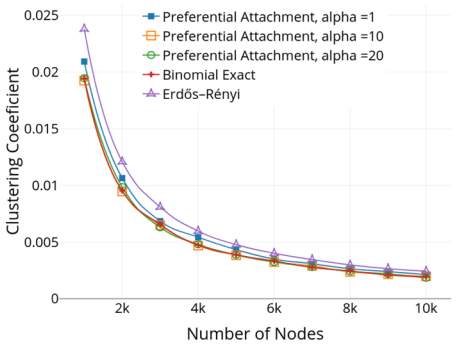
(a) outbound connections = 4, non-contributors= 0%



(b) outbound connections = 4, non-contributors= 50%

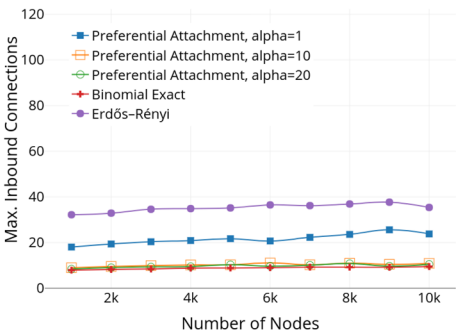


(c) outbound connections = 6, non-contributors= 0%

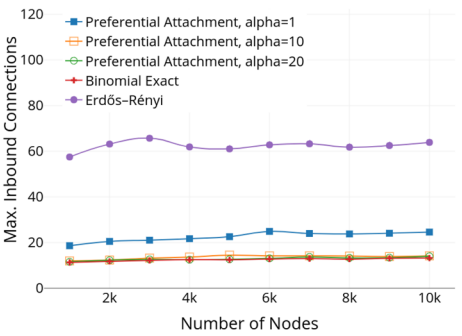


(d) outbound connections = 6, non-contributors= 50%

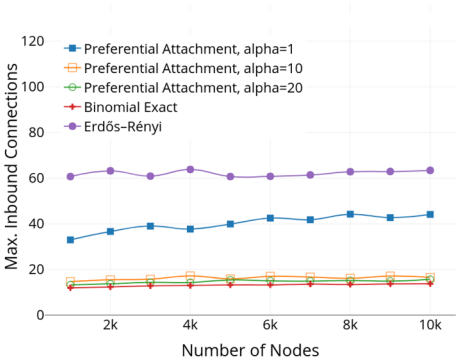
Fig. 11 Max inbound connections



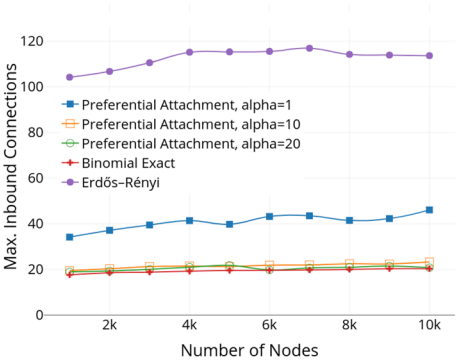
(a) outbound connections = 2, non-contributors= 0%



(b) outbound connections = 2, non-contributors= 50%



(c) outbound connections = 4, non-contributors= 0%



(d) outbound connections = 4, non-contributors= 50%

8.5 Graph generation time

The Fig. 9 represents the time in seconds required to generate digraphs. It is evident that time for graph generation increases with increase in the number of nodes n or number of outbound connections r as both causes increase in total edges to be formed in the P2P network. Increasing the number of non-contributors k reduces the graph generation time. The graph generation time for Approximate *r-out* graph (Erdős-Rényi) was less compared to Exact *r-out* graph (Preferential Attachment). This is obvious given the fact that there are additional constraints to make the digraph Exact *r-out*. However, this time can largely be reduced by avoiding vertices whose in/out degrees are saturated w.r.t threshold. Binomial Exact *r-out* digraph's reduced generation time illustrates this (Section 6.4.4). We can also observe that the change in the number of non-contributors k affects Preferential Attachment digraph only.

8.6 Clustering coefficient

Figure 10 shows the plots for clustering coefficient. The trends depicted are for outbound connections $r = 4, 6$ and for non-contributors $k = 0\%, 50\%$. The coefficient value *decreases* with an increase in the number of total participating nodes n while it *increases* when outbound connections r or non-contributors k are increased(more clustering). Preferential Attachment with lower α value has most clusters as the initial weight assigned to each node is low during the graph generation (Section 6.4.5). However, Approximate *r-out* digraph (Erdős-Rényi) tends to cluster more when non-contributors k in a given blockchain P2P network increase. The Exact *r-out* digraph (Binomial) performs better in this case as well with a low value of the clustering coefficient.

8.7 Maximum inbound connections

Figure 11 illustrates the plot of maximum inbound connections (maximum clustering) a node can have within the blockchain P2P network (when γ is not fixed). Here, in this case, Approximate *r-out* digraph (Erdős-Rényi) performs far worse compared to Exact *r-out* digraph. Maximum inbound connections are fairly constant and do not vary to a great extent when the number of participating nodes n is increased. However, this value *increases* when the number of non-contributors k or the outbound connections r is increased. For example, for Erdős-Rényi, this value nearly increases by 100% when non-contributors k increase by 50% (for outbound connections $r = 4$).

9 Conclusion and future work

The role of the underlying P2P network in blockchain is underrated even though most of the performance metrics for any blockchain are affected by it. When modeled properly, blockchain P2P network performance can be optimized so as to satisfy a given set of constraints like low resources overhead (for IoT devices), full connectivity, small diameter and a configurable number of clusters. We showed how the blockchain P2P network is affected by different topology control methods and when network size, outbound connections, and non-contributors are varied. We mathematically evaluated *r-out* topological structure using the binomial, uniform distribution models and preferential attachment model. Then, we established lower bounds on inter peer connections as a function of total participating nodes, outbound connections per node and non-contributors to guarantee full connectivity in the blockchain P2P network. Finally, we investigated asymptotic topological properties of blockchain P2P network.

As a future work, we plan to implement the findings of this paper by dimensioning the blockchain P2P network parameters (inbound connections, outbound connections, number of participants, number of non-contributors) on demand, according to the use-case(s) requirements (latency, data rate, reliability, availability, etc.) and participating nodes profiles (low energy, mobility, computational resources, memory, etc.). Our idea is to conceive an on-demand, secure, scalable and efficient blockchain tool that takes, as input, the application requirements and participating nodes profile and makes decision about P2P network parameters. We will also conduct a theoretical and experimental study to determine how the servers layer will be optimally dimensioned to meet the peers layer scalability.

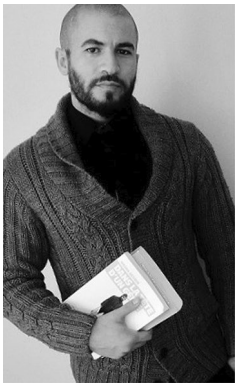
References

1. Zhao JL, Fan S, Yan J (2016) Overview of business innovations and research opportunities in blockchain and introduction to the special issue
2. Nagpal R (2017) 17 blockchain platforms - a brief introduction. Available: https://medium.com/blockchain-blog/17-blockchain-platforms-a-brief-introduction-e07273185a0b
3. Deshpande V, Badis H, George L (2018) Btcmap: Mapping bitcoin peer-to-peer network topology. In 2018 IFIP/IEEE International Conference on Performance Evaluation and Modeling in Wired and Wireless Networks (PEMWN). IEEE pp. 1–6
4. Maymounkov P, Mazières D (2002) Kademlia: A peer-to-peer information system based on the xor metric. In Revised Papers from the First International Workshop on Peer-to-Peer Systems (Berlin, Heidelberg), IPTPS '01, Springer-Verlag pp. 53–65

5. Ramachandran GS, Krishnamachari B (2018) Blockchain for the iot: Opportunities and challenges. arXiv preprint arXiv:1805.02818
6. Kreutz D, Ramos FM, Verissimo PE, Rothenberg CE, Azodolmolky S, Uhlig S (2014) Software-defined networking: A comprehensive survey. Proceedings of the IEEE 103(1):14–76
7. All cryptocurrencies (2018). Available: https://coinmarketcap.com/all/views/all/
8. Yli-Huumo J, Ko D, Choi S, Park S, Smolander K (2016) Where is current research on blockchain technology?-a systematic review. PLoS One 11(10):1–27
9. Bach LM, Mihaljevic B, Zagar M (2018) Comparative analysis of blockchain consensus algorithms. In 2018 41st International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO) pp. 1545–1550
10. Wang W, Hoang DT, Hu P, Xiong Z, Niyato D, Wang P, Wen Y, Kim DI (2019) A survey on consensus mechanisms and mining strategy management in blockchain networks. IEEE Access 7:22328–22370
11. Yu Z, Liu X, Wang G (2018) A survey of consensus and incentive mechanism in blockchain derived from p2p. In 2018 IEEE 24th International Conference on Parallel and Distributed Systems (ICPADS) pp. 1010–1015
12. Bernabe JB, Canovas JL, Hernandez-Ramos JL, Moreno RT, Skarmeta A (2019) Privacy-preserving solutions for blockchain: Review and challenges. IEEE Access 7:164908–164940
13. Wang D, Zhao J, Wang Y (2020) A survey on privacy protection of blockchain: The technology and application. IEEE Access 8:108766–108781
14. Huynh TT, Nguyen TD, Tan H (2019) A survey on security and privacy issues of blockchain technology. In 2019 International Conference on System Science and Engineering (ICSSE) pp. 362–367
15. Biryukov A, Khovratovich D, Pustogarov I (2014) Deanonymisation of clients in bitcoin p2p network. In Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security (New York, NY, USA), CCS '14, Association for Computing Machinery pp. 15–29
16. Decker C, Wattenhofer R (2013) Information propagation in the bitcoin network. In Peer-to-Peer Computing (P2P), 2013 IEEE Thirteenth International Conference on pp. 1–10
17. Brienza S, Cebeci SE, Masoumzadeh SS, Hlavacs H, Özkasap O, Anastasi G (2015) A survey on energy efficiency in p2p systems: File distribution, content streaming, and epidemics. ACM Comput Surv 48:3
18. Orgerie A-C, Assuncao MDD, Lefevre L (2014) A survey on techniques for improving the energy efficiency of large-scale distributed systems. ACM Comput Surv 46:4
19. Cho S, Lee S (2019) Survey on the application of blockchain to iot. In 2019 International Conference on Electronics, Information, and Communication (ICEIC) pp. 1–2
20. Doku R, Rawat DB, Garuba M, Njilla L (2019) Lightchain: On the lightweight blockchain for the internet-of-things. In 2019 IEEE International Conference on Smart Computing (SMARTCOMP) pp. 444–448
21. Dai H, Zheng Z, Zhang Y (2019) Blockchain for internet of things: A survey. IEEE Internet Things J 6(5):8076–8094
22. Dorri A, Kanhere SS, Jurdak R (2017) Towards an optimized blockchain for iot. In 2017 IEEE/ACM Second International Conference on Internet-of-Things Design and Implementation (IoTDI) IEEE pp. 173–178
23. Pan J, Wang J, Hester A, AlQerm I, Liu Y, Zhao Y (2018) Edgechain: An edge-iot framework and prototype based on blockchain and smart contracts. IEEE Internet Things J 6(3):4719–4732
24. Castro M, Liskov B (1999) Practical byzantine fault tolerance. In Proceedings of the Third Symposium on Operating Systems Design and Implementation (USA), OSDI '99, USENIX Association pp. 173–186
25. Ledger nano x. Available: https://shop.ledger.com/pages/ledger-nano-x
26. Edge blockchain wallet and security platform. Available: https://edge.app/
27. Atomic cryptocurrency wallet. Available: https://atomicwallet.io/
28. Drake N (2018) How to mine cryptocurrencies on your android smartphone. Available: https://techradar.com/how-to/how-to-mine-cryptocurrencies-on-your-android-smartphone
29. Coinhive android sdk. Available: https://github.com/theapache64/coin_hive_android_sdk
30. Is mobile mining profitable? (2018) Available: https://coincentral.com/is-mobile-mining-profitable/
31. Ometov A, Bardinova Y, Afanasyeva A, Masek P, Zhidanov K, Vanurin S, Sayfullin M, Shubina V, Komarov M, Bezzateev S (2020) An overview on blockchain for smartphones: State-of-the-art, consensus, implementation, challenges and future trends. IEEE Access 8:103994–104015
32. Gupta V, Kaur K, Kaur S (2016) Network programmability using software defined networking. In 2016 3rd International Conference on Computing for Sustainable Global Development (INDIACom) pp. 1170–1173
33. Zhang J, Yang C, Shi H, Wang L (2017) A survey of building efficient p2p streaming systems in sdn. In 2017 4th International Conference on Information Science and Control Engineering (ICISCE) pp. 120–125
34. de Carvalho Silva J, Rodrigues JJPC, Alberti AM, Solic P, Aquino ALL (2017) Lorawan - a low power wan protocol for internet of things: A review and opportunities. In 2017 2nd International Multidisciplinary Conference on Computer and Energy Science (SpliTech) pp. 1–6
35. Lavric A, Petrariu AI, Popa V (2019) Long range sigfox communication protocol scalability analysis under large-scale, high-density conditions. IEEE Access 7:35816–35825
36. Lamport L, Shostak R, Pease M (1982) The byzantine generals problem. ACM Trans Program Lang Syst 4(3):382–401
37. Fischer MJ, Lynch NA, Paterson MS (1985) Impossibility of distributed consensus with one faulty process. J ACM 32(2):374–382
38. Dwork C, Lynch N, Stockmeyer L (1988) Consensus in the presence of partial synchrony. J ACM 35(2):288–323
39. Baldoni R, Helary J-M, Raynal M (2000) From crash fault-tolerance to arbitrary-fault tolerance: Towards a modular approach. In Proceeding International Conference on Dependable Systems and Networks. DSN 2000 IEEE pp. 273–282
40. Nguyen G-T, Kim K (2018) A survey about consensus algorithms used in blockchain. J Inf Process Syst 14:1
41. Chen L, Xu L, Shah N, Gao Z, Lu Y, Shi W (2017) On security analysis of proof-of-elapsed-time (poet). In International Symposium on Stabilization, Safety, and Security of Distributed Systems. Springer pp. 282–297
42. Karantias K, Kiayias A, Zindros D (2020) Proof-of-burn. In International Conference on Financial Cryptography and Data Security. Springer pp. 523–540
43. Gervais A, Ritzdorf H, Karame GO, Capkun S (2015) Tampering with the delivery of blocks and transactions in bitcoin. In Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security (New York, NY, USA), CCS'15, Association for Computing Machinery pp. 692–705
44. Pasqualetti F, Bicchi A, Bullo F (2012) Consensus computation in unreliable networks: A system theoretic approach. IEEE Trans Autom Control 57(1):90–104
45. Satoshi client node discovery. Available: https://en.bitcoin.it/wiki/Satoshi_Client_Node_Discovery
46. GlobalPlatform, Inc. (2018) Introduction to secure element

47. Deshpande V, George L, Badis H (2019) Pulsec: Secure element based framework for sensors anomaly detection in industry 4.0. IFAC-PapersOnLine 52(13):1204–1209
48. Deshpande V, Das T, Badis H, George L (2019) Sebs: A secure element and blockchain stratagem for securing iot. In 2019 Global Information Infrastructure and Networking Symposium (GIIS) IEEE pp. 1–7
49. Deshpande V, George L, Badis H (2019) Safe: A blockchain and secure element based framework for safeguarding smart vehicles. In 2019 12th IFIP Wireless and Mobile Networking Conference (WMNC) IEEE pp. 181–188
50. Urien P (2018) Towards secure elements for trusted transactions in blockchain and blochchain iot (biot) platforms. invited paper. In 2018 Fourth International Conference on Mobile and Secure Services (MobiSecServ) IEEE pp. 1–5
51. Deshpande V, George L, Badis H, Desta AA (2020) Blockchain based decentralized framework for energy demand response marketplace. In NOMS 2020-2020 IEEE/IFIP Network Operations and Management Symposium IEEE pp. 1–9
52. Beverly R, Afergan M (2007) Machine learning for efficient neighbor selection in unstructured p2p networks. SysML 7:1–6
53. Srivatsa M, Gedik B, Liu L (2006) Large scaling unstructured peer-to-peer networks with heterogeneity-aware topology and routing. IEEE Trans Parallel Distrib Syst 17(11):1277–1293
54. Yu G, Shao-Hai H (2006) Establishing tcp connections between hosts behind nats. In Wireless, Mobile and Multimedia Networks, 2006 IET International Conference on IET pp. 1–3
55. Mauldin RD, Ed (1982) The Scottish Book: Mathematics from the Scottish Cafe, 1 ed. Birkhause
56. Frieze A, Karoński M (2016) Introduction to random graphs. Cambridge University Press
57. Barzdin IM (1973) Finite automata-behavior and synthesis. North-Holland Publishing Company
58. Flaxman AD, Frieze AM (2004) The diameter of randomly perturbed digraphs and some applications. In Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques. Springer pp. 345–356

Varun Deshpande Varun Deshpande is currently working on Embedded Secure Elements and Blockchain at METRON SAS, Paris. This is in collaboration and as a part of his studies of Ph.D. (CIFRE Equivalent) at Université Gustave Eiffiel (LIGM-ESIEE Paris).



Hakim Badis Associate Professor in Computer Science Department, Université Gustave Eiffiel. Researcher at LIGM, Université Gustave Eiffiel specialized in next generation wireless networks, multi-hop and IoT sensor networks, smart antennas (MIMO, etc.), discrete mathematics (graph theory, information theory, etc.), distributed algorithms and complexity, Blockchain and networks.

Laurent George Head of the Computer Science department at ESIEE Paris. Teacher-Researcher specialized in Networks, Real Time Embedded Systems, IoT and Security (Blockchain) for applications related to the smart city (autonomous vehicles, energy efficiency)