Distributed Community Detection over Blockchain Networks Based on Structural Entropy

ABSTRACT

Blockchain technology provides a groundbreaking computing paradigm that tackles problems in a completely decentralised manner. As the underlying infrastructure and protocol of blockchain, blockchain networks convey communications and coordinations across all involving participants. In extensive application scenarios, conducting community detection over blockchain networks has potential effects on both discovering hidden information and enhancing communicating efficiency. However, the decentralised nature poses a restriction on community detection over blockchain networks. In coping with this restriction, we propose a distributed community detection method based on the Propose-Select-Adjust (PSA) framework that runs in an asynchronous way. We extend the PSA framework using the concept of structural entropy and aim to detect a community structure with low entropy. We test our entropy-based distributed community detection algorithm on both benchmark networks and bitcoin trust networks. Experimental results reveal that our algorithm successfully detects communities with low structural entropy.

KEYWORDS

blockchain, community detection, structural entropy, distributed computing

ACM Reference Format:

. 2019. Distributed Community Detection over Blockchain Networks Based on Structural Entropy. In *BSCI 2019: ACM International Symposium on Blockchain and Secure Critical Infrastructure, July 7–12, 2019, Auckland, New Zealand*. ACM, New York, NY, USA, 10 pages. https://doi.org/xx.xxxx/xxxxxxx.xxxxxxx

1 INTRODUCTION

Cryptocurrency has attracted massive attentions from industry, government institutions, and academia in recent years. As the first and most representative cryptocurrency, bitcoin has gained huge success in the worldwide capital market. The blockchain is the core technology and underlying mechanism for all kinds of cryptocurrency, which was first proposed in 2008 and implemented in 2009 [13]. Blockchain can be regarded as a public ledger of all cryptocurrency transactions that have ever been executed. It is constantly growing as new blocks are appended to it by miners (typically every 10 minutes) to record the most recent transactions. The key characteristics of blockchain include decentralisation, persistency, anonymity, and auditability, in which decentralisation is most outstanding. Through using technologies of cryptographic hash, digital signature (asymmetric cryptography) and distributed consensus mechanism, blockchain works in a completely decentralised environment, which allows transactions to be recorded and validated without central trusted devices. Beyond cryptocurrency and financial-related applications, the *generalised blockchain* has demonstrated immense potentials in fields of governance, health, science, literacy, culture, and art [22].

Blockchain utilizes the peer-to-peer (P2P) network as the underlying communication architecture and protocol, where participants are equally privileged in terms of communication. Based on the P2P network architecture, the generalised blockchain can be regarded as a complex network-based software connector, which provides communication, coordination (through transactions, smart contracts, and validation oracles) and facilitation services [24]. The two-layer blockchain model is depicted in Fig. 1, where each node has two layers, namely, the *blockchain layer* and the *application layer*. In the application layer, interactions among participants of the blockchain form various types of virtual relations such as trust [21] and transaction [3]. Each of these relations form a network.

Since there is no control by central servers, networks in the application layer are very much self-organising and exhibit some significant features [12, 25]. One of the most prevalent features is *community structure* which means that the network can be viewed as consisting of a number of sub-networks, called communities; the links within the same community are dense whereas between different communities are sparse. Naturally, the application layer networks of blockchain exhibits *community structure*. The *community detection problem* demands an automated approach to reveal the communities given the network structure.

Detecting communities over application-layer networks can solve some significant problems from a structural perspective. For example, one of the most straightforward applications in the context of blockchain is to track bitcoin users activity on bitcoin transaction networks [19]. Each node in a bitcoin transaction network represents an address and the network is partitioned into several communities. In this scenario, the detected communities are used to re-identify multiple addresses that belong to the same user. Another application amounts to facilitating anomaly detection in cryptocurrency transaction network [18]. The transaction network is divided into groups to locate anomalies as soon as possible to prevent them from harming the network's community and integrity. A third possible application scenario is to execute group validation over blockchain trust networks. A trust network records trust relationships of participants involving in blockchain applications. The trust network is not static, but rather, it undergoes continuous updates as users join and leave the network. This calls for a dynamic algorithm to perform community detection while the trust relationships

BSCI 2019, July 7–12, 2019, Auckland, New Zealand

ACM ISBN xxx-x-xxxx-xxxx-x/xx/xx…$xx.xx
https://doi.org/xx.xxxx/xxxxxxx.xxxxxxx

change as time passes. Such a dynamic algorithm would be useful when validating events: When an event proposed by a user is broadcast across the communication network, the groups that capture the message can diffuse the event within the respective group and provide group validations accordingly.

Existing methods to solve the community detection problem pose some severe challenges for the applications above on blockchain networks. Namely, most established methods for community detection are centralised approaches, where the algorithm needs to know in advance the entire network topology [5, 15, 17]. These methods produce highly accurate results. Such centralised methods prove to be difficult to scale to large decentralised networks where global information is inaccessible. Furthermore, blockchain networks are dynamic in the sense that nodes and links frequently come and go, and communities evolve continuously. In coping with such dynamic networks, classical approaches such as snapshots analysis may be ineffective, as snapshots may not smoothly track community evolution across different time stamps [16]. All the above pose new challenges and hence call for a novel approach for community detection. To attempt at designing a community detection method for blockchain, the following issues are needed to be settled:

- (Decentralisation) The method is capable to work on the decentralised network without the requirement of massive information exchanging. Instead of detecting communities using the entire network data, one would need to delegate the problem to many individual processors, who have only access to local data.
- (Dynamisation) The method is adaptive to the changes to the network. Namely, when the network undergoes continuous changes, the results of the algorithm would be updated efficiently.

Contribution. In light of these requirements, we propose a distributed community detection for blockchain networks based on the Propose-Select-Adjust (PSA) framework, which is a distributed community detection framework with asynchronous runs [10]. In the PSA framework, each node only requires its local information provided by the neighbourhood and functions without synchronisation from a central controller. The contributions of this paper are listed as follows:

- We propose a distributed community detection method for blockchain networks based on the PSA framework. This framework will settle the decentralisation and dynamisation problem above.
- We extend the original formulation of PSA framework [10] by utilizing the concept of *local structural entropy* [8]. This metric evaluates the amount of information that lies within a sub-network and is an appropriate guiding factor for community detection.
- To test our algorithm, we conduct experiments on both benchmark networks for community detection and bitcoin trust networks. Results show that our algorithm produces reasonable community structures.

The rest of this paper is organised as follows: In Sec. 2, we give some preliminaries on the community detection problem, structural entropy, and PSA framework, which provide the necessary background for the introduction of our distributed algorithm in Sec. 3. Next, in Sec. 4 we show the detailed mechanism of how the entropy-based PSA-system functions to detect communities in a distributed asynchronous manner. Sec. 5 shows the experimental results of our algorithm. Finally, the paper ends with conclusions and outlook in Sec. 6.

Related Work. Community detection is one of the most important questions in the study of complex networks. Conventional methods for community detection include the clique percolation algorithm which is agglomerative [17], Girvan and Newman's betweenness-based algorithm which is divisive [5], and approaches that aim to maximize modularity [15]. All of these classical methods are centralised approaches which requires the utilization of the entire network structure. This has to be achieved by a central controller which would mean that the algorithms would be insufficient for the case of blockchain networks that described above.

Using techniques from community detection to solve problems in blockchain, especially in the context of cryptocurrency, has attracted a lot of attention recently. For example, the authors in [19] propose several algorithms to detect communities on bitcoin transaction networks for predicting different addresses that belong to the same user. Community detection has also shown its potential in facilitating anomaly detection. For example, clustering methods such as k-means are used in [18] to detect and locate anomaly in cryptocurrency transaction networks. Moreover, in industrial applications, a blockchain-based dynamic community detection method is proposed in [20] for detecting P2P botnets in the Internet of things (IoT). Our work differs from these works in that we propose a distributed asynchronous community detection algorithm which provides a solution framework for a series of problem domains in the blockchain-based applications above.

This work is also related to structural entropy. The notion aims to extend information theory to networked data. However, the classical information entropy pioneered by Shannon fails to support analysis on communication networks since network topology is not taken into consideration. To better analyse communication networks from the perspective of information theory, Li et. al. in [8] propose the notion of structural entropy and further define multi-dimensional structural entropy in [9]. Their goal is to capture the information that are inherent to a network structure. Their series of work on structural entropy points out how structure structure impacts information resistance and information uncertainty on communication networks. Inspired by their work, we employ structural entropy in our proposed distributed community detection method, which is expected to find out community structures with low structural entropy.

2 PRELIMINARIES

2.1 Community Detection

We regard a complex network as an undirected unweighted graph $G = (V, E)$ where V, E are the sets of nodes and edges, respectively. We assume that V contains n elements $\{1, 2, \ldots, n\}$ and abuse the

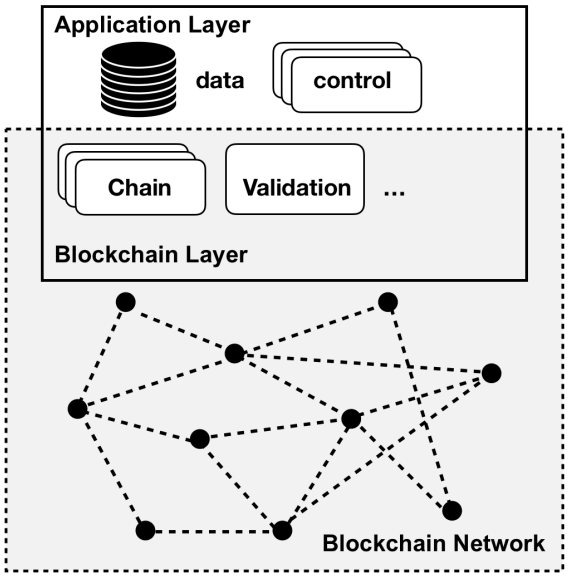

Figure 1: Overview of the two-layer blockchain model.

notation writing an undirected edge as $(u, v) \in V^2$. For any node $u \in V$, the *neighbourhood* of u, $N(u)$, is the set $\{v\} \cup \{v \in V \mid (u, v) \in E\}$ that contains v and all neighbours of v. For a subset $C \subseteq V$, $N(C)$ denotes $\bigcup_{u \in C} N(u)$.

Very often complex network structures exhibit a special property [23]. More precisely, nodes in the network can be partitioned into clusters, with a high density of edges within each cluster but low density of edges between these clusters [5]. Any graph with this property is named a *community structure* where each cluster mentioned above is called a *community*. A well-known example is the karate club network, as constructed out of the social relationship between members of a karate club in an American university in the 1970s. As can be seen in Figure 2, the network contains two clearly divided communities, which facilitates the prediction of real events [26].

Formally, the *community structure* of a graph $G = (V, E)$ refers to a partition $\mathcal{C}$ of the node set V, i.e.,

$$\mathcal{C} = \{C_1, C_2, \ldots, C_k\}$$

such that each C_i for $1 \leq i \leq k$ induces a connected subgraph, called a *community*. The requirement for these $C_1, \ldots, C_k$ is that they have high intra-cluster density and low inter-cluster density, as defined below.

Definition 1. Let $\mathcal{C}$ be a clustering of a graph $G = (V, E)$, and $C \in \mathcal{C}$ be a cluster. The *intra-cluster density* of C represents the edge connectivity within C; it is defined as

$$\delta_{\text{int}}(C) = \frac{|E \restriction C|}{|C|(|C| - 1)/2}, \qquad \text{if } |C| > 1$$

and $\delta_{\text{int}}(C) = 1$ if $|C| = 1$.

Let $D_1, D_2, \ldots, D_m$ be all communities in $\mathcal{C}$ such that $C \cap D_i = \varnothing$ and $D_i \cap N(C) \neq \varnothing$ for all $1 \leq i \leq m$. The *inter-cluster density* of C represents the edge connectivity between C and its neighbouring clusters; it is computed as

$$\delta_{\text{ext}}(C) = \frac{|C| \times |N(C) \setminus C|}{|C| \times (|D_1| + \cdots + |D_m|)}, \quad \text{if } m \geq 1$$

and $\delta_{\text{ext}}(C) = 0$ if $m = 1$.

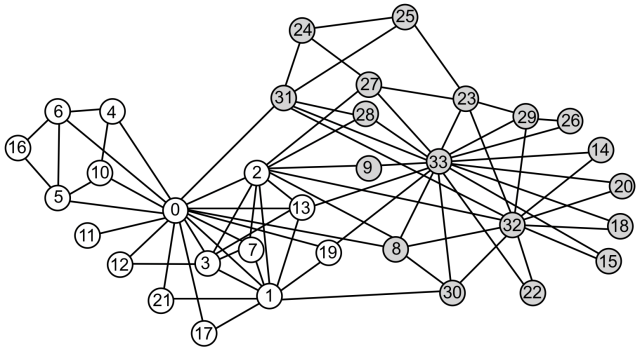

Figure 2: The community structure of an university karate club analyzed by Zachary. The two communities are in two different colours.

As networks vary greatly in the real-world, there has not been a universally accepted formal definition of communities in networks. On the other hand, most established definitions resort to the outcome of certain concrete algorithms. In general, an over-arching view of the community structure is that it reveals important global information about the network structure. We will exploit this intuition to provide an information-theoretic notion of community structure in subsequent sections.

The *community detection problem* aims to compute the community structure $\mathcal{C}$ of a given graph $G = (V, E)$ where each community in $\mathcal{C}$ has high intra-cluster density but a low inter-cluster density.

2.2 Structural Entropy

Shannon's information entropy measures the inherent uncertainty of a probabilistic distribution:

$$H(p_1, \cdots, p_n) = -\sum_{i=1}^{n} p_i \log_2 p_i.$$

This metric is well-known and widely used. Shannon's theory on entropy and the associated concept of noise, have provided rich insights for information science and technology. It is certainly desirable that this entropy measure can be applied to relational structures such as complex networks. However, this famous metric does not provide the appropriate tools for this purpose. The obstacle mainly arises due to the fact that Shannon's information entropy relies on a pre-defined probability distribution. However, a single probability distribution would normally not be sufficient to capture the information content of a complex network. For example, consider measuring the information entropy using the degree distribution of nodes in a network as follows:

Definition 2. The *Shannon structural entropy* of a graph G has the following form: Let $G = (V, E)$ be a connected graph where $|V| = n$

and $|E| = m$. For each vertex i, let d_i be the degree of i in G, and let $p_i = d_i/2m$. Then the vector $\vec{p} = (p_1, p_2, \ldots, p_n)$ is the stationary distribution of a random walk in G. By using this, we define the one-dimensional structural entropy or positioning entropy of G by:

$$\mathcal{H}(G) = H(\vec{p}) = H\left(\frac{d_1}{2m}, \ldots, \frac{d_n}{2m}\right) = -\sum_{i=1}^{n} \frac{d_i}{2m} \cdot \log_2 \frac{d_i}{2m}.$$

Two networks would have the same Shannon structural entropy if they have the same degree distribution. However, this notion would not be able to reflect the existence of community structure information, i.e., graphs with or without a clear community structure may have the same degree distribution. Therefore for the purpose of more refined network analysis, one would need a new framework that expresses structural entropy.

The authors of [9] argue that, from the perspective of data communication, the structural entropy of a network should in principle reflect a sense of resistance over information passing. A network structure with low structural entropy should mean lower uncertainty in terms of delivering a piece of information between two nodes, and thus such structure would be preferred in terms of facilitating communication. On the other hand, a higher structural entropy should denote the fact that the network structure contains "holes" [1] which forbids the passing of information. From social network theory, it is well-known that such holes correspond to certain closed clusters with few outside contacts. In this sense, a network with a higher structural entropy should denote the existence of a clear community structure.

Generally, given a clustering $\mathcal{C} = \{C_1, C_2, \cdots, C_k\}$ over G, we would need a notion of *local entropy* of a community C_j where $1 \le j \le k$. For any community C_j, let V_j denote the *volume* of the community, i.e., the sum of degree of nodes in V_j, and g_j denote the number of edges with exactly one endpoint lying in community C_j.

Definition 3. Let $G = (V, E)$ be a graph where $|V| = n$ and $|E| = m$, assume that $\mathcal{C} = \{C_1, C_2, \cdots, C_k\}$ is a community structure of V. Define the *local entropy* of C_j as:

$$\mathcal{H}^{C_j}(G) = \frac{V_j}{2m} H\left(\frac{d_1^j}{V_j}, \cdots, \frac{d_{n_j}^j}{V_j}\right) - \frac{g_j}{2m} \log_2 \frac{V_j}{2m},$$

where n_j is the number of nodes in community C_j, d_i^j is the degree of the i-th node in C_j, V_j is the number of edges with both endpoints in community C_j, and g_j is the number of edges with exactly one endpoint in community C_j.

Based on local entropy, we define the *structural entropy* as follow.

Definition 4 ([9])**.** The *structural entropy of a community structure* C of a graph G is defined by summing the local entropy of all communities:

$$\begin{aligned}\mathcal{H}^{\mathcal{C}}(G) &= \sum_{j=1}^{k} \mathcal{H}^{C_j}(G) \\ &= -\sum_{j=1}^{k} \frac{V_j}{2m} \sum_{i=1}^{n_j} \frac{d_i^j}{V_j} \log_2 \frac{d_i^j}{V_j} - \sum_{j=1}^{k} \frac{g_i}{2m} \log_2 \frac{V_j}{2m}.\end{aligned}$$

According to the definition of structural entropy, a community structure $\mathcal{C} = \{C_1, C_2 \ldots, C_k\}$ leads to a low entropy if every $C_i \in \mathcal{C}$ has a high intra-cluster density and low inter-cluster density, which is consistent with the intuition of a community structure. We therefore present the following formal definition of an optimal community structure.

Definition 5 ([9])**.** Given a network $G = (V, E)$, the *inherent community structure* of G is a community structure $\mathcal{C}^*$ with the minimum structural entropy, i.e.,

$$\mathcal{C}^* = \arg\min_{\mathcal{C}} \mathcal{H}^{\mathcal{C}}(G).$$

The inherent community structure is thus the goal of our community detection problem. In our entropy-based distributed community detection framework to be introduced below, we iteratively refine the clustering of a network in order to continuously reduce the structural entropy.

3 THE Propose-Select-Adjust FRAMEWORK

In [10], the authors put forward a framework that is appropriate for decentralised processing of graphs. Namely, the Propose-Select-Adjust (PSA) framework describes a general scheme that involves a network to self-organise into a community structure.

The scheme treats each node in the network as an independent processing unit, called an *agent*. The framework is based on the consensus formation process in a multi-agent system. Imagine a group of networked agent whose collective goal is to decide on dividing the group into several community, where every agent belongs to one and only one partition. Imagine further a setup where each agent only sees local information about her own connections. An agent may communicate with other through the connections, but no knowledge is shared among all agents. In this way, an individual can only make ego-centric judgements. The PSA framework aims to ensure that the agents in this context still arrive at a consensus through repeated interactions. Each agent performs the following three tasks:

(1) Propose: Each agent individually computes a list of agents and sends an *invitation* to every node on the list in the hope of forming a community.
(2) Select: An agent may also receive invitations from other agents. After the invitations are received, the agent evaluates the *quality* of each proposed community, and chooses to accept the more favourable invitation.
(3) Adjust: Once an agent accepts an invitation, she then deliberates her own community according to the accepted proposal. After every individual finishes this step, the whole group would have been divided into a number of communities, and thus a community is formed.

The resulting community structure may not be optimal. In this case, the agents should start another round of the three steps above, as illustrated in Fig. 3.

We now formally define a *cell* in the PSA-framework. The set Σ in the definition is the symbol set $\{\mathsf{p}, \mathsf{s}, \mathsf{a}\}$.

Definition 6. [10] Let $G = (V, E)$ be a network and v be a node in V. A PSA-*cell* defined on the node v is a tuple

$$\mathcal{M}_v = (\mathcal{P}, \mathcal{S}, \mathcal{A}, Q, (\delta_{\sigma,v})_{\sigma\in\Sigma}, F_v) \quad \text{where}$$

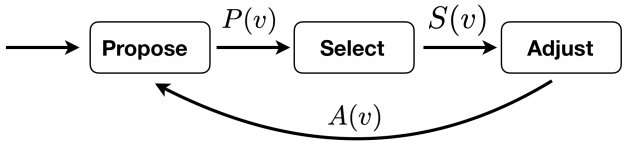

Figure 3: The Propose-Select-Adjust **Framework.**

- $\mathcal{P}, \mathcal{S}, \mathcal{A}$ are finite sets of *proposals*, *selections* and *solutions*, respectively; Q is a finite set of *control states*
- $\delta_{\mathrm{p},v} : (\mathcal{A} \times \mathcal{P} \times Q)^{|N(v)|} \to \mathcal{P} \times Q$ is the *propose function*
- $\delta_{\mathrm{s},v} : (\mathcal{P} \times \mathcal{S} \times Q)^{|N(v)|} \to \mathcal{S} \times Q$ is *select function*
- $\delta_{\mathrm{a},v} : (\mathcal{S} \times \mathcal{A} \times Q)^{|N(v)|} \to \mathcal{A} \times Q$ is *adjust function*
- $F_v : \Sigma \times Q^{|N(v)|} \to Q$ is the *change-step functions*.

As a general algorithmic framework, the definition above does not pinpoint the exact meaning of proposals, selections, and solutions but rather introduce them generically as sets. A PSA-cell runs by iteratively switching its states. The state of a PSA-cell is denoted as

$$(\sigma, P(v), S(v), A(v), q) \in \Sigma \times \mathcal{P} \times \mathcal{S} \times \mathcal{A} \times Q$$

where $P(v), S(v), A(v)$ are the current *proposal*, *selection* and *solution* of the cell v, respectively, $q \in Q$ and $\sigma \in \Sigma$ is called the *step* of v, which denotes one of propose, select and adjust, respectively. The cell v applies the three transition functions $\delta_{\mathrm{p},v}, \delta_{\mathrm{s},v}$ and $\delta_{\mathrm{a},v}$ repeatedly in turn to the appropriate components of its current state:

(1) The cell v starts in step p and applies the propose function $\delta_{\mathrm{p},v}$ to compute a proposal $P(v) \in \mathcal{P}$, according to current solutions and proposals (and control states) of cells in its neighbourhood $N(v)$.
(2) Then v moves to step s and applies the select function $\delta_{\mathrm{s},v}$ to compute a selection $S(v) \in \mathcal{S}$, according to proposals and selections (and control states) of cells in $N(v)$.
(3) Then v moves to step a and applies the adjust function $\delta_{\mathrm{a},v}$ to update its candidate solution $A(v)$, according to selections and solutions (and control states) of cells in $N(v)$. The cell then repeats the above cycle.

Formally, a PSA-cell v in state σ of the computation may change to the next state σ', where $(\sigma, \sigma') \in \{(\mathrm{p}, \mathrm{s}), (\mathrm{s}, \mathrm{a}), (\mathrm{a}, \mathrm{p})\}$, by performing a transition

$$(\sigma, P(v), S(v), A(v), q) \mapsto (\sigma', P(v), S(v), A(v), q')$$

only if $F_v(\sigma, q_1, \ldots, q_k) = q'$ where $q_1, \ldots, q_k$ are the current control states of cells in $N(v)$. We now formally introduce a PSA-system.

Definition 7.

- A PSA-*system* Γ is defined by

$$\Gamma = (V, E, (\mathcal{M}_v)_{v \in V}, \mathcal{P}, \mathcal{S}, \mathcal{A}, Q, p_0, s_0, a_0, q_0)$$

where (V, E) is a network, for each node $v \in V$, $\mathcal{M}_v$ is a PSA-cell with proposal set, selection set, solution set and control states $\mathcal{P}, \mathcal{S}, \mathcal{A}, Q$, respectively, $p_0 \in \mathcal{P}$, $s_0 \in \mathcal{S}$, $a_0 \in \mathcal{A}$ and $q_0 \in Q$.

- The initial state if any cell in the PSA-system is

$$(\mathrm{p}, p_0, s_0, a_0, q_0).$$

- A *configuration* of the PSA-system is defined as a function

$$c : V \to \Sigma \times \mathcal{P} \times \mathcal{S} \times \mathcal{A} \times Q.$$

- A *run* of the PSA-system is a sequence $c_0, c_1, \ldots$ of configurations in which each cell cycles through the processes of propose, select and adjust as describe above.

Let $A(c, v)$ denote the solution in the state $c(v)$ for a node $v \in V$. A run $\rho = c_0, c_1, \ldots$ is *stablising* if there is some $i \geq 0$ such that $\forall j \geq i \forall v \in V : A(c_i, v) = A(c_j, v)$. Stablising runs are important for a PSA as the *limit* would define the outcome of the system. Furthermore, it is desirable to have a PSA-system all of whose runs are stablising and have the same limit. This would then define the *outcome* of the system. In the subsequent section, we present an instantiation of the PSA-system all of whose runs are stablising and have the same limit.

4 COMMUNITY DETECTION WITH A ENTROPY-BASED PSA-SYSTEM

Our algorithm combines the PSA-system as introduced above with structural entropy. To instantiate the PSA-system, we need to specify the different ingredients in the definitions above.

We describe the proposal set $\mathcal{P}$ and the solution set $\mathcal{A}$ for Γ. The proposal set relies on the notion of a *tendency tree*. A *tree* is a tuple (T, f) where T is a set of nodes and $f : T \to T$ is an *edge function* such that $f(r) = r$ for exactly one node (the root) $r \in T$ and $\exists i > 0 : f^i(u) = r$ for all other nodes $u \neq r$.

Definition 8. A *tendency tree* of a graph (V, E) is a tree (T, f) where $T \subseteq V$ and for any $u \neq v \in T$, $f(u) = v$ implies $(u, v) \in E$.

The proposal set $\mathcal{P}$ and solution set $\mathcal{S}$ of the PSA-system Γ is the set of all tendency trees of (V, E) where the initial proposal and solution are both the empty tree. The selection set $\mathcal{S}$ is $V \cup \{\mathsf{null}\}$ where the initial selection is null. To present our distributed algorithm, we illustrate our algorithm on a classical benchmark network, namely, the karate club network.

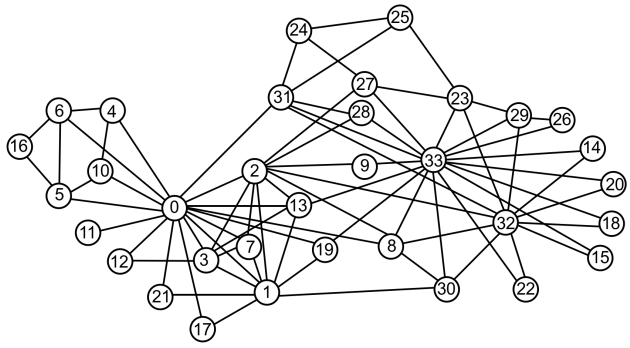

Figure 4: The initial configuration of the karate club network.

In the design of [10], a PSA-cell runs a number of *rounds* of the propose-select-adjust cycle, which are indexed by $1, 2, 3, \ldots$ and ω.

In particular, there are three different stages of rounds where a cell would perform different operations. These are round1, round $i > 1$, and round ω, as illustrated in Fig 5. We use the control states in Q to flag the current stage of a cell. We now describe each of these stages in detail.

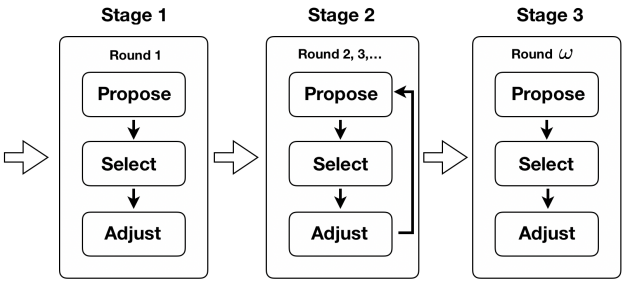

Figure 5: The general computation flow of a cell in the PSA-system Γ.

4.1 Stage 1

Propose. The propose step is for an individual cell to initiate the formation of a community. Here we use the notion of *core centrality* of nodes [2], which has been a classical tool in social network analysis. The intuition of a core resembles degree, i.e., the number of edges adjacent to a node. Unlike degree, this notion considers not only the local information of a node, but rather a set of nodes. More precisely, a *k-core* is an induced subgraph of G where all nodes have degree at least k. The *core centrality* of a node v is the largest k-core that contains v.

Definition 9. For $v \in V$, the *local core* of v is the set

$$K(v) = \{u \in N(v) \mid |N(u) \cap N(v)| \geq \kappa(v) + 1\}.$$

In the propose step, a cell $\mathcal{M}_v$ would set a tendency tree defined on the local $K(v)$ as the proposal $P(v)$. This proposal will be sent to all nodes in $K(v)$. The proposals for the nodes in the karate club network are shown in Figure 6.

Select. The cell v waits for all its neighbouring cells $N(v)$ to make a proposal and collects all received proposals, i.e.,

$$I(v) = \{u \in N(v) \mid v \in K(u)\}.$$

Note that every cell in $I(v)$ has proposed v to join with itself. To make a selection, v picks the most "favourable" proposal among all proposals $P(u)$ where $u \in I(v)$. For the next definition, we say that a *C-consistent set* for a community structure C is a union of communities in C. In the original PSA-system, the authors define the preference relation using directly intra-cluster density and inter-cluster density [10]. In this paper, on the other hand, we adopt a different preference relation using the concept of local structural entropy introduced in Sec. 2.

Definition 10. Let C be a community structure of the graph $G = (V, E)$. We define the *preference relation* $\preceq$ on all C-consistent sets such that $C \prec C'$ if

(1) $\mathcal{H}^C(G) > \mathcal{H}^{C'}(G)$; or

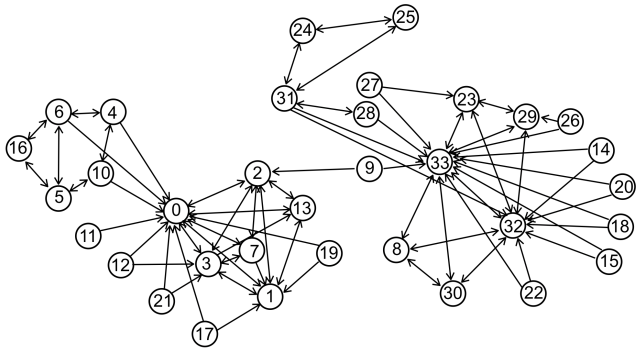

Figure 6: The proposal $P(v)$ made by a node v in the karate club network in Stage 1 consists of nodes in the local core $K(v)$. Self-loops in the tendency trees are omitted. E.g. the tendency tree of 0 contains nodes 0,1,2,3,7.

(2) $\mathcal{H}^C(G) = \mathcal{H}^{C'}(G)$ and $|C| < |C'|$.

The node v then selects the node $S(v) = u \in I(v)$ whose proposal is the most preferred. For example, Figure 7 displays the selections of each node in the karate club network.

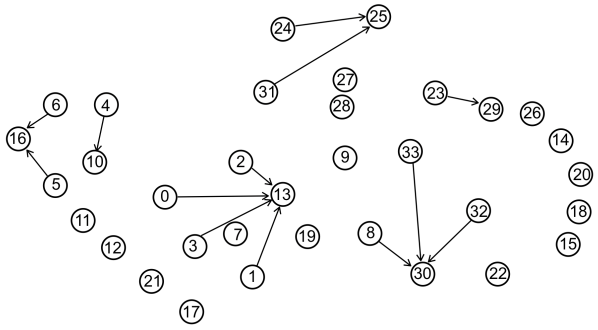

Figure 7: The selection of each node is labeled by an arrow. Self-selections are omitted.

Adjust. After all cells in $N(v)$ have made a selection, the cell v then updates its solution $A(v)$ to include all cells who have selected v, i.e., set $A(v) = \{u \in N(v) \mid S(u) = v\}$. After this step, all cells would have declared a community $A(v)$; all these communities form a community structure C_0 of the graph. Fig. 8 shows the resulting community structure in the karate club network.

4.2 Stage 2

After the communities are formed in Stage 1, cells in the same community bind into a single "meta"-cell and start to make collective decisions. To further reduce the structural entropy of the resulting community structure, the propose-select-adjust procedures are repeated several times. Here the operations are different from Stage 1 as the cells are no longer individual nodes, but rather they are communities formed in the previous round. In this stage, each cell

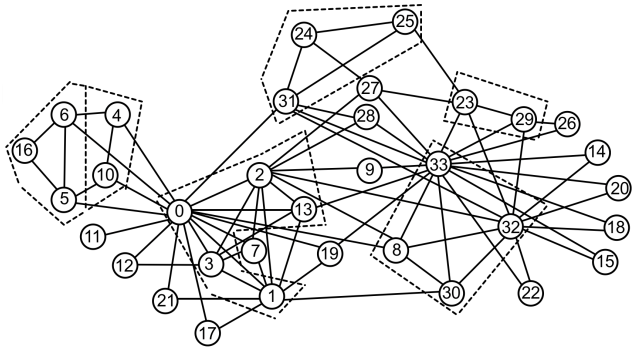

Figure 8: The resulting clustering in Zachary's karate club after the first round. Nodes belong to the same community have circled in dash line. An isolated nodes is a community comprises itself

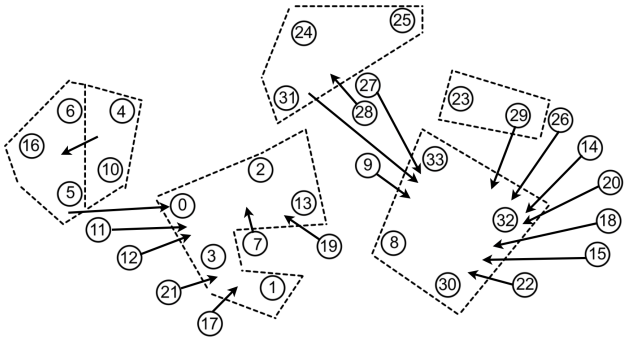

Figure 9: The final outcome of the PSA-system on the karate club network. The $\tau(C)$ of each community C is shown with an arrow. Self-loops are omitted from the diagram. The two sinks are the community of node 0 and the community of node 33. Hence the resulting community structure C_ω coincides with the ground truth community structure (Figure 2) as in Zachary's original work [26].

aims to lower the local structural entropy of its community until no improvement can be achieved. We present the intuitive ideas:

Propose. Recall that (T, f) is the current tendency forest $A(v)$ of v. The cell v examines all $u \in K(v)$ and compares the current solution $A(v)$ with the union of $A(v)$ and $A(u)$. The new proposal $P(v)$ contains the union of $A(v)$ and $A(u)$ that has the highest preference. This proposal is then propagated to the root of T, who makes a proposal based on proposals of its children and passes it down to all cells in T. In this way, all $v \in T$ will produce the same proposal.

Select. We say that a community $C \subseteq \mathcal{C}_0$ *receives* a proposal $P(u) \in \mathcal{P}$, if $P(u)$ contains C. Through propagation of information, the community of v examines all proposals it receives and chooses the proposal with the highest preference.

Adjust. For every community $C \in \mathcal{C}_0$, we define a *tendency cluster* $\tau(C) \in C$. There are two cases:

- Case 1: every cell $x \in C$ selects its parent $f(x)$ in its current tendency tree; in this case the community set $\tau(C) = C$.
- Case 2: Some cell $x \in C$ selects a node $u \notin C$; in this case the community C has decided to join with the community C' that contains u, and we set $\tau(C) = C'$.

Note that whenever $\tau(C) \neq C$, $\tau(C)$ always has higher utility vector than C. Hence for every $C \in \mathcal{C}_0$, there is some $j \in \mathbb{N}$ and $D \in \mathcal{C}_{i-1}$ such that $D = \tau^j(C) = \tau^{j+1}(C)$. We call D the *sink* of C.

We define the community structure $\mathcal{C}_1$ as follows: $C, C' \in \mathcal{C}_0$ bind into the same community in $\mathcal{C}_1$ whenever they have the same sink. Each node v then adjusts its solution $A(v)$ to its new community in $\mathcal{C}_1$. For this to happen, the tendency tree of $A(v)$ may need to be changed so that it is linked with another community in $\mathcal{C}_0$. The resulting tendency tree's root would be the root of its sink.

After all cells in $N(v)$ update their solutions, v then moves back to step p and starts another round. The node v moves on to Stage 3 when no change occurs to $A(v)$ after step a. In the karate club example, the community structures $\mathcal{C}_0$ and $\mathcal{C}_1$ are the same, so every node directly moves on to Stage 3 after one round in Stage 2.

4.3 Stage 3

The community structured obtained after Stage 2 has in a certain sense reached *local optimality* in the sense that they cannot achieve a lower structural entropy if combined with any neighbouring communities. However, at this point, the resulting community structures may still be quite distinct from the desired outcome. They are normally too small to reveal any global structure of the network. In real-world networks, communities tend to combine several such communities (e.g. the two communities in Fig. 2 are formed by combining several communities in Fig. 8). Hence we use Stage 3 to find such tendency and obtain the final clustering of the network.

Stage 3 is performed similarly to Stage 2, except that when a community $C \in \mathcal{C}_*$ generates proposals, it would send a proposal to every node in its neighbourhood. The clustering $\mathcal{C}_\omega$ is the outcome of the PSA-system and contains all the communities identified in this network. Figure 9 shows the result on the karate club network, which matches the groundtruth community structure in Fig. 2. This shows that the PSA-system correctly detected the communities in the karate club dataset. Fig. 9 shows tendency of each community $C \in \mathcal{C}_*$ in the karate club network.

5 EXPERIMENTAL RESULTS

5.1 Benchmark Networks

Besides the karate club network, we also run our PSA-system on two networks with ground truth community structure, dolphin social network and American college football league network, which are commonly used as benchmarks for community detection. The dolphin network was generated from observations of a group of 62 bottlenose dolphins. Nodes represent the dolphins, and edges represent associations between dolphin pairs occurring more often than expected by chance [11]. The American college football league network contains the network of American football games between Division IA colleges during regular season Fall 2000 [14]. Nodes

and edges represent teams and matches, respectively. There are 10 conferences and 8 independent teams in 2000.

Fig. 10 shows the resulting communities in the Doubtful sound bottlenose dolphin network, where edges are social interactions between dolphins. Fig. 11 shows the resulting communities in the American college football league network, where our algorithm accurately revealed conferences in the league.

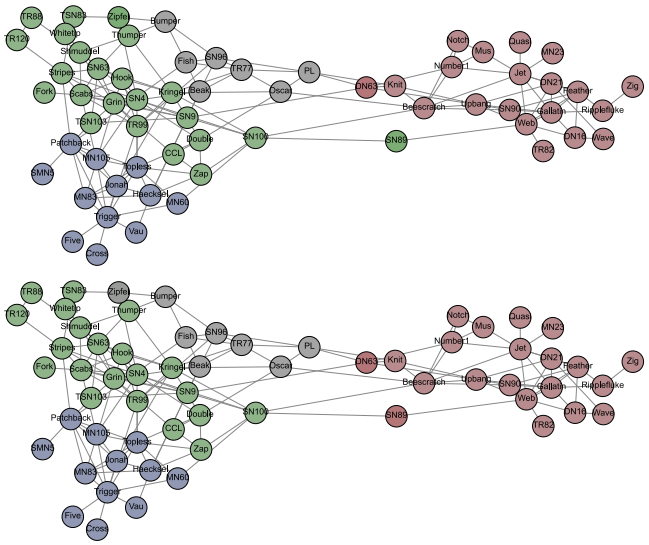

Figure 10: Running our algorithm reveals four communities in the dolphin social network. Our algorithm revealed four communities which matches the ground truth to a high precision. (Top: The ground truth community structure [11]. Bottom: The community structure detected by our algorithm.)

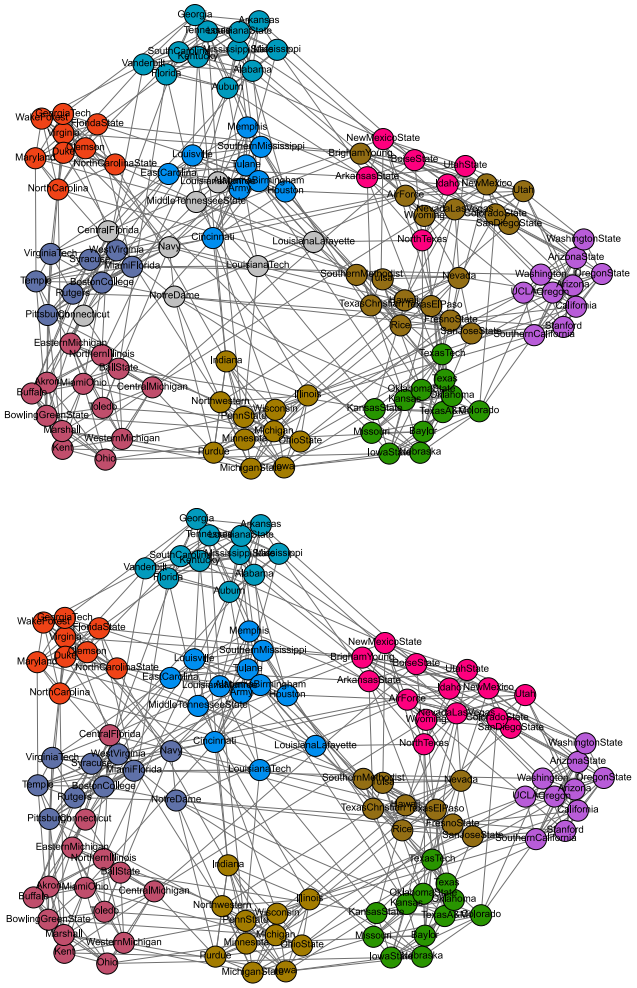

Figure 11: Running our algorithm reveals several communities in the American college football league, which match the actual conferences to a high precision. (Top: The ground truth community structure, where grey nodes indicate independent teams. Bottom: The community structure detected by our algorithm.)

5.2 Bitcoin Trust Networks

A network is dynamic when it undergoes continuous changes such as adding/deletion of edges. We next conduct experiments on two dynamic bitcoin trust networks, which we call BITCOIN1 and BITCOIN2 [6, 7], respectively. A bitcoin trust network records who-trusts-whom relationships of people who trade using bitcoin on a platform called bitcoin OTC, where users are anonymous. Nodes in bitcoin trust networks represent users and if user i trusts user j, there is an edge between i and j. Each edge is assigned with a timestamp, which represents the exact time when the corresponding edge occurs. The statistics of two bitcoin trust networks at the last timestamp are summarised in Table 1. To conduct simulations, we speed up the evolution of networks by 100 times, namely, every second in the experiment we append new edges occurring during an interval of 100s in reality.

We apply both the original PSA algorithms and our structural entropy-based PSA algorithm to two bitcoin networks. To demonstrate the outcome of our entropy-based algorithm, we select four timestamps and show the evolution of the community structure in BITCOIN2 (see Fig. 12).

We measure the quality of the resulting clusterings using two *quality functions*: let $C = \{C_1, \ldots, C_\ell\}$ be the clustering obtained by the algorithm on the bitcoin network at the last timestamp.

1) *Modularity*: This widely used quality function measures the proportion of in-cluster edges taking into account the expected proportion; it is defined as

$$\mathrm{Mod}(C) = \sum_{i=1}^{\ell} \left[\frac{|E[C_i]|}{|E|} - \frac{d_i^2}{4|E|^2} \right]$$

where d_i is the sum of degrees of nodes in C_i [4].

2) *Performance*: This function counts edges within communities and pairs of nodes that are not linked by edges between different

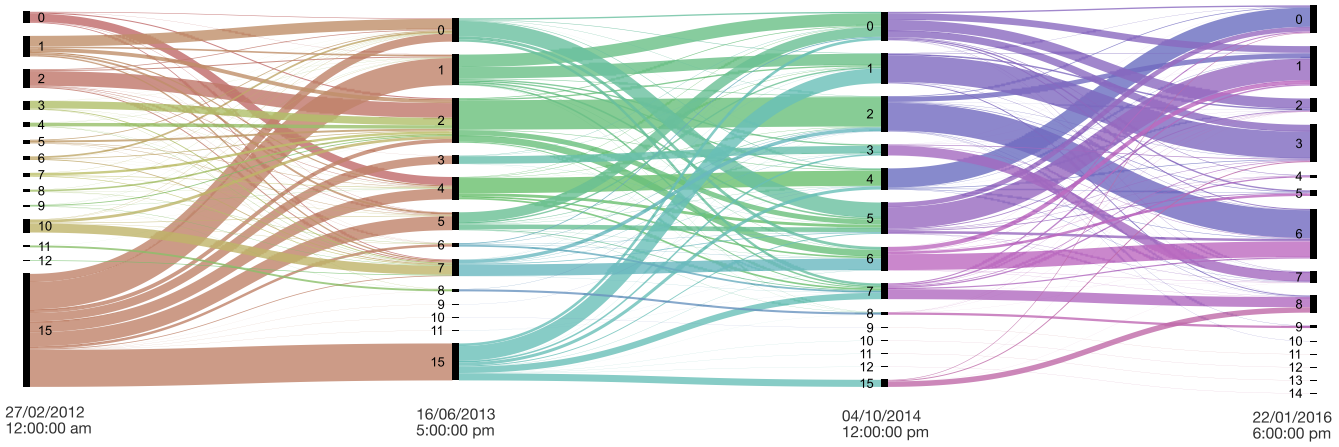

Figure 12: The evolution of the community structure of BITCOIN2 **output by our algorithm. The category 15 refers to the isolated nodes at corresponding timestamp. Our algorithm detectes 15 stable communities at the last timestamp, which represents 15 trust groups over bitcoin trading on the OTC platform.**

communities (the "correctly interpreted pairs") [4]; it is defined as

$$\frac{|X \cup Y|}{(|V|-1)|V|},$$

where $X = \{(u,v) \in E \mid C(u) = C(v)\}$, $Y = \{(u,v) \notin E \mid C(u) \neq C(v)\}$ and $C(x)$ is the community of x.

Fig. 13 shows the results of the quality of the resulting clusterings using two metrics introduced above. Moreover, we also compute the structural entropy of each bitcoin network under the corresponding obtained clustering structure at the last timestamp. Outcomes of experiments on performance and structural entropy are illustrated in Fig. 14.

Table 1: Statistics of two bitcoin trust networks (last timestamp)

	Nodes	Edges	Largest Component
BITCOIN1	5,881	35,592	5,876
BITCOIN2	3,783	24,186	3,775

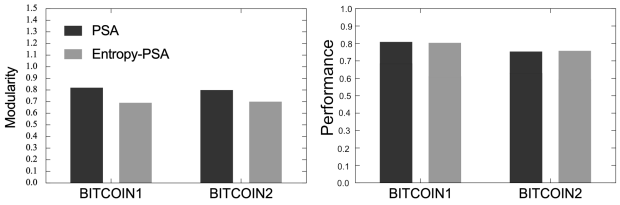

Figure 13: Results of modularity and performance by PSA **in [10] and the entropy-based** PSA **on two bitcoin trust networks.**

Experimental results reveal that entropy-based PSA detects a community structure with slightly lower modularity and considerable performance compared to the original PSA algorithm, but

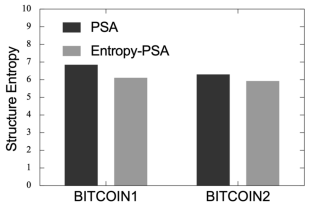

Figure 14: Results of structure entropy by PSA **in [10] and the entropy-based** PSA **on two bitcoin trust networks.**

reduces the structural entropy of the output community structure. This implies that blockchain networks logically partitioned by entropy-based PSA tend to have a lower resistance and uncertainty over information diffusion. As a result, the efficiency of the blockchain system undergoes an enhancement.

6 CONCLUSIONS AND OUTLOOK

In this paper, we propose a distributed asynchronous community detection algorithm based on PSA framework for blockchain networks. Using structural entropy as the selecting criteria, the algorithm can detect community structure with low structural entropy. Experimental results show that the algorithm works well on both benchmark networks and bitcoin networks. Our work provides a solution framework for categories of problems in blockchain-based applications.

Future research directions that have yet to be explored include: (a) Extend the current framework to detect overlapping communities in blockchain networks; (b) Explore more application scenarios where our proposed algorithm can be applied; (c) Embed our distributed community detection algorithm into the fundamental layer of blockchain technology to support the applications in higher layers.

REFERENCES

[1] Ronald S Burt. 2004. Structural Holes and Good Ideas. *Amer. J. Sociology* 110, 2 (2004), 349–399.

[2] Kousik Das, Sovan Samanta, and Madhumangal Pal. 2018. Study on centrality measures in social networks: a survey. *Social Network Analysis and Mining* 8, 1 (2018), 13.

[3] Michael Fleder, Michael S Kester, and Sudeep Pillai. 2015. Bitcoin transaction graph analysis. *arXiv preprint arXiv:1502.01657* (2015).

[4] Santo Fortunato. 2010. Community detection in graphs. *Physics reports* 486, 3-5 (2010), 75–174.

[5] Michelle Girvan and Mark EJ Newman. 2002. Community structure in social and biological networks. *Proceedings of the national academy of sciences* 99, 12 (2002), 7821–7826.

[6] Srijan Kumar, Bryan Hooi, Disha Makhija, Mohit Kumar, Christos Faloutsos, and VS Subrahmanian. 2018. Rev2: Fraudulent user prediction in rating platforms. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*. ACM, 333–341.

[7] Srijan Kumar, Francesca Spezzano, VS Subrahmanian, and Christos Faloutsos. 2016. Edge weight prediction in weighted signed networks. In *Data Mining (ICDM), 2016 IEEE 16th International Conference on*. IEEE, 221–230.

[8] Angsheng Li and Yicheng Pan. 2016. Structural information and dynamical complexity of networks. *IEEE Transactions on Information Theory* 62, 6 (2016), 3290–3339.

[9] Angsheng Li and Yicheng Pan. 2018. Structure Entropy and Resistor Graphs. *arXiv preprint arXiv:1801.03404* (2018).

[10] Jiamou Liu and Ziheng Wei. 2014. Community detection based on graph dynamical systems with asynchronous runs. In *Computing and Networking (CANDAR), 2014 Second International Symposium on*. IEEE, 463–469.

[11] David Lusseau and Mark EJ Newman. 2004. Identifying the role that animals play in their social networks. *Proceedings of the Royal Society of London B: Biological Sciences* 271, Suppl 6 (2004), S477–S481.

[12] Anastasia Moskvina and Jiamou Liu. 2016. Integrating networks of equipotent nodes. In *International Conference on Computational Social Networks*. Springer, 39–50.

[13] Satoshi Nakamoto. 2008. Bitcoin: A peer-to-peer electronic cash system. (2008).

[14] Mark EJ Newman. 2004. Fast algorithm for detecting community structure in networks. *Physical review E* 69, 6 (2004), 066133.

[15] Mark EJ Newman and Michelle Girvan. 2004. Finding and evaluating community structure in networks. *Physical review E* 69, 2 (2004), 026113.

[16] Gergely Palla, Albert-László Barabási, and Tamás Vicsek. 2007. Quantifying social group evolution. *Nature* 446, 7136 (2007), 664.

[17] Gergely Palla, Imre Derényi, Illés Farkas, and Tamás Vicsek. 2005. Uncovering the overlapping community structure of complex networks in nature and society. *Nature* 435, 7043 (2005), 814.

[18] Thai Pham and Steven Lee. 2016. Anomaly detection in bitcoin network using unsupervised learning methods. *arXiv preprint arXiv:1611.03941* (2016).

[19] Cazabet Remy, Baccour Rym, and Latapy Matthieu. 2017. Tracking bitcoin users activity using community detection on a network of weak signals. In *International Workshop on Complex Networks and their Applications*. Springer, 166–177.

[20] Gokhan Sagirlar, Barbara Carminati, and Elena Ferrari. 2018. AutoBotCatcher: Blockchain-based P2P Botnet Detection for the Internet of Things. In *2018 IEEE 4th International Conference on Collaboration and Internet Computing (CIC)*. IEEE, 1–8.

[21] David Shrier, Weige Wu, and Alex Pentland. 2016. Blockchain & infrastructure (identity, data security). *Massachusetts Institute of Technology-Connection Science* 1, 3 (2016).

[22] Melanie Swan. 2015. *Blockchain: Blueprint for a new economy.* " O'Reilly Media, Inc.".

[23] Duncan J Watts. 2004. *Small worlds: the dynamics of networks between order and randomness*. Vol. 9. Princeton university press.

[24] Xiwei Xu, Cesare Pautasso, Liming Zhu, Vincent Gramoli, Alexander Ponomarev, An Binh Tran, and Shiping Chen. 2016. The blockchain as a software connector. In *2016 13th Working IEEE/IFIP Conference on Software Architecture (WICSA)*. IEEE, 182–191.

[25] Bo Yan, Yang Chen, and Jiamou Liu. 2017. Dynamic relationship building: exploitation versus exploration on a social network. In *International Conference on Web Information Systems Engineering*. Springer, 75–90.

[26] Wayne W Zachary. 1977. An information flow model for conflict and fission in small groups. *Journal of anthropological research* 33, 4 (1977), 452–473.