星云链技术白皮书

Nebulas Inc.

August 8, 2017

Abstract

白皮书的摘要，让大家看完摘要后，就对星云链要做的事情一目了然。

TODO 是蓝色

注释是红色

引用的例子，如下 GNU pthread [**pth**], PARSEC [**bienia2009parsec**]

Contents

1 简介

1.1 区块链技术简介

简单介绍一下区块链技术的概要，和最流行的开源区块链技术特点。

1.2 商业和技术挑战

当前公有链遇到的技术挑战：智能合约无法被简单搜索；用户以及智能合约的价值贡献无法被评估和计算；PoW 共识速度慢，消耗大量电能，PoS 共识使得拥有大量代币资产的用户获得记账资格，形成马太效应，带来另外的不公平忧虑；目前的激励机制仅仅奖励矿工，有一定的局限性，不能鼓励优秀的智能合约开发者；智能合约一经部署，无法升级，不适应普通的企业应用需求；区块链核心协议如果需要升级，容易引起硬分叉，对区块链社区带来伤害；

1.3 星云链解决的问题

介绍星云链的贡献：NR 排序算法，核心协议的可升级、PoR 共识（出块速度和安全性）、DIP 开发者激励、智能合约升级设计、

1.4 星云链设计基本原则

阐述星云链设计的基本原则：完全兼容以太坊、为区块链设计价值尺度、更快更合理的共识设计、对区块链生态的有效激励、图灵完备可升级的智能合约、可自我演化的区块链、更好用的区块链基础服务。

2 星云链代币 NAS

robin

3 账号和地址

类似于比特币和以太坊，星云链的帐号也是由椭圆曲线算法作为其基础加密算法。用户的私钥是随机生成的 256 位二进制数，通过椭圆曲线乘法生成 64 字节的公钥。比特币和以太坊的地址都是由公钥通过确定性的哈希算法运算得到，不同的是：比特币地址有校验码设计，防止用户输错了几个字符而把比特币误发给其它用户，以太坊却没有校验码设计。我们认为从用户的角度看，校验码的设计是合理的，因此星云链的地址也包含了校验码，具体计算方法如下：

```
Data = sha3_256(Public Key)[-20:]
CheckSum = sha3_256(Data)[0:4]
Address = "0x" + Data + CheckSum
```

公钥的 SHA3-256 摘要的后 20 位字节作为地址的主要组成部分，对其再做一次 SHA3-256 摘要，取前四位字节做校验码，相当于以太坊的地址尾部加上四字节的校验码。以太坊地址包括 0x 前缀共 42 个字符，星云链地址共 50 个字符，增加了 8 个字符的校验码。除了支持标准的 50 字符的标准地址，为了用户转账的安全，我们还支持扩展地址。借鉴了传统银行转账的设计：转账时除了验证对方的银行卡账号，汇款人还必须输入对方的姓名，只有银行卡账号和姓名都匹配，转账才能正确进行。扩展地址的生成算法如下：

```
Address = "0x" + Data + CheckSum
Ext = sha3_256(Data + nick)[0:2]
ExtAddress = Address + Ext
```

扩展地址在标准地址的尾部追加了 2 字节的扩展验证，共 54 个字符。加入扩展信息，使得在星云链钱包应用中可以增加另一个要素验证，比如：Alice 的钱包标准地址是 0x5d65d971895edc438f465c17db6992698a52318d5c17db69，加入“alice”后的扩展地址是 0x5d65d971895edc438f465c17db6992698a52318d5c17db69aabb；Alice 告诉 Bob 扩展地址 0x5d65d971895edc438f465c17db6992698a52318d5c17db69aabb 和 alice；Bob 在 Wallet 应用中，输入 0x5d65d971895edc438f465c17db6992698a52318d5c17db69aabb 和 alice。Wallet 应用验证钱包地址的一致性，避免 Bob 错误的输成了其它人的帐号。标准地址和扩展地址均可以用于转账，我们的星云链钱包应用将会强制使用扩展地址，转账时需要提供对方的昵称，验证地址一致性，进一步加强安全校验。

4 区块和交易

Wenbo

5 Nebulas Rank 算法

5.1 Introduction

Ranking nodes in complex network has been a fundamental concern in various applications. One canonical example is PageRank[**Brin2010**][**page1999pagerank**], which is the core algorithm for Google and other search engines[**langville2011google**]. Besides, by ranking algorithm, people also want to find out the most influential spreaders in epidemic and information network[**doerr2012rumors**][**Kitsak2010**], the most acknowledged scientist by the citation network or co-author network[**walker2007ranking**][**chen2007finding**][**Radicchi2009**], the most important cities in the transportation network[**guimera2005worldwide**], the most important vertices in metabolic network[**ivan2010web**], the top VC firms by co-investment[**Bhat2012**] etc. And when it comes to designing **Nebulas Rank** algorithm for blockchain world, decades of researches have enlightened us with many measurements like degree centrality[**freeman1979set**], eigenvector centrality[**bonacich1972factoring**], Katz centrality[**katz1953new**], PageRank[**Brin2010**], HITS[**kleinberg1999authoritative**], closeness centrality[**sabidussi1966centrality**], betweenness centrality[**freeman1977set**][**freeman1978centrality**][**freeman1991centrality**] etc. Before building our algorithm on the top of them, however, we still need to answer two questions:

1. What properties are embedded in the network?
2. What value should the rank indicate?

For the first question, **Nebulas Rank** uses the transaction graph of blockchain system, which is generated by the transaction history during the past period. We compare blockchain transaction graph with others in three aspects. First, as node representing accounts and edges representing money transferring, basically, the graph is a weighted directed graph, bearing large difference from social network[**Ugander2011**] and similarity with webpage network[**page1999pagerank**] in terms of topology. An asymmetric edge indicates the imbalanced ability of controlling money between two nodes. And edges' weight also characterizes the difference among links' quality. Thus directly applying standard algorithms for unweighted or undirected graph could discard a lot important information. Second, since the graph is derived by the trajectory of money flow, it could be presumed that Exchanges are highly ranked, whereas such accounts are willing to swap money with any client. Thus anyone can acquire unlimited links from those existing important nodes without much cost. Along with the anonymity of typical blockchain systems, sybil attackers could make large amount of transferring with a big Exchange account, in order to improve his influence such as PageRank. (It is still hard to receive money from a large number of non-sybil nodes, though.) This is an essentially difference from applications previously studied. For example, in an online social network with subscription relation, it costs no effort to follow an opinion leader, while attracting a verified opinion leader's attention is not an trivial thing. Third, being different from Bitcoin[**Nakamoto2008**], the newly invented blockchain systems such as Ethereum[**Wood2014**] introduces "smart contract" as a new type of account. After a normal account invoking some method of a contract, a sequence of consequent callings will be raised forming part of a call graph. Unlike Bitcoin

transaction graph, which only contains money transferring, Ethereum call graph/network also represents dynamic programming calling. We believe such network embeds more information and should be useful to measure a DApp or smart contract's value.

For the second question, **Nebulas Rank** aims to measure the value of users, and smart contracts in blockchains. For normal accounts, we define value by two aspects: **Liquidity**, which stresses the ability to control digital assets flow of high quality; **Propagation**, which focus more on the spreading influence. For smart contracts, we also consider **Interoperability** as a measurement. There are three-fold purposes in **Nebulas Rank**: 1) to be a good metric for blockchain accounts and smart contract search engine; 2) to provide a trustful criteria in PoR consensus protocol (see §**??**) , where only high ranked accounts should be eligible to become a validator; 3) to help to build DIP mechanism (see §**??**). [1].

More clearly, **Nebulas Rank** should be based on the formal concept of network flow. As revealed in [**Borgatti2005**], most centralities can be classified by their type of network flow. From the dimension of diffusion mechanism, traffic flow can spread by different kinds of duplication or transfer. Another spectrum is the trajectory of flow, which can be either paths, trails or walks. Essentially, blockchain transaction graph is the trajectory of money exchange, which falls into the classification of "transfer walk". Imagine an amount of money enters the network. Then the owner node divides the money and transfers to neighbors or keep it. To clarify, the money is divisible, imperishable and non-replicable and each step's direction is random due to the limit of local information. This rules out some measurements for us. For example, **freeman1977set**'s betweenness centrality, since it implies geodesic optimal paths.

Following are our solutions to the challenges described above.

First, in order to turn transactions into a graph, we keep the transfer value as edges' weight and embed the transfer timestamp into nodes' coinage property. Then we exploit coinage and other properties to fix the edge weight:

- We set a time window of T days and aggregate the largest K transactions as an edge's weight. (see §5.3.2;
- We reduce each edge's weight by its target's "coinage". In order to get higher coinage, an owner needs to let the money stay in place for a while, which slows down the speed of sybil attacks such as loop attack. (see §5.3.3)
- Edge weight is also reduced by its target's outgoing amount(see §5.3.6) as well as an encouragement function (see §5.3.4), which helps to mitigate some undesired effect.

Second, with graph generated, we measure each node's importance based on Weighted LeaderRank algorithm[**Chen2013**][**Li2014**]. LeaderRank is a simple variant of PageRank, which adds a ground node into the network and connect the ground node with each non-ground node. It substitutes PageRank's teleportation parameter by links throughout ground node, which is said to be more effective in computing, robust against manipulations and noise than PageRank algorithm[**Chen2013**]. The intuition for both LeaderRank and PageRank is random walk and Markov Chain. By PageRank, from each node, the

[1]Note the in this section, our transaction data don't include smart contracts. More about DIP is described in §**??**

probability of jumping to an arbitrary node is the same (or equal to 1 if there is no out links [**Kim2002**]). Whereas by LeaderRank[**Li2014**][**Chen2013**], different nodes adopt different arbitrary transition probabilities. For example, we could allow a node with more in-links to receive more teleportation probability. This is more plausible in the context of blockchain, since an account with little money transferred in is less trustable. Also, if a node receives much money but hardly spend it, we suppose it has more "surplus value" and assign with more teleportation probability from it. Intuitively, LeaderRank indicates the flux over a node in the dynamic equilibrium of money exchange network flow. From another perspective, mode flux means more control. The LeaderRanks algorithm matches both our goals of measuring **Liquidity** and **Propagation**. We will talk about the details of LeaderRank scheme in §5.4.

~~Our experiment results shows that .....~~

The rest of paper is organized as follows. §5.2 introduces the related works. Then in §5.3, we define the network topology and weight based on blockchain transactions. And in §5.4, LeaderRank with schemes designed for **Nebulas Rank** is introduced. In §5.5, we show our experiment results. And Finally we give all discussions and conclusions in §5.6

5.2 Related Works

Centrality, the core ranking index, is a most studied concept in network science since decades ago[**newman2010networks**]. There are a body of literatures introducing various centralities, including degree centrality[**freeman1979set**], eigenvector centrality[**bonacich1972factoring**], Katz centrality[**katz1953new**], closeness centrality[**sabidussi1966centrality**], betweenness centrality[**freeman1977set**][**freeman1978centrality**][**freeman1991centrality**] PageRank[**Brin2010**], HITS[**kleinberg1999authoritative**], SALSA[**Science2001**], etc. It is fundamental to clearly classify these measurements by a unified framework. **Borgatti2005** adopts a network flow based view to classify the centrality measurements by two categorical dimensions: material flowing by parallel duplication, serial duplication and transfer; and trajectory following geodesics optimum, path, trail and walk. **Borgatti2006** propose a unified framework with four dimensions from the perspective of graph theory. **Lu2016** review representative centrality algorithms and classified them into those only based on structural information, those driven by Markov dynamics, those by looking at the effect of removing nodes, those with dynamics-sensitivity and those trying to identify more than one node. With a hierarchical understanding of centrality algorithms, we are able to choose appropriate strategy according to the network scenarios. **Nebulas Rank**'s scenario is the money exchange flow network mentioned in [**Borgatti2005**].

Since Bitcoin[**Nakamoto2008**] system released in 2009, researchers have done some statistical and empirical analysis on Bitcoin's transaction graph[**Ron**][**Haslhofer**][**NielKondor2014**][**Baumann2014**], and some use the transaction graph structure to discuss anonymity in Bitcoin[**Meiklejohn2013**][**Ober2013**][**pham2016anomaly**][**Fleder2015**][**Fer** After other cryptocurrencies emerged and become popular, transaction graph analysis is conducted with more blockchains[**Chang2017**][**Anderson2016**]. **Nebulas Rank** adopts their transaction graph concept, i.e. Entity Graph in [**Tschorsch2015**], with minor revisions. That is, each account, or set of

accounts belonging to the same people, is mapped as a node. And each directed edge represents the intensity of transferring between two accounts. Actually before blockchain system like Bitcoin was invented, scientists have tried to study some financial networks among banks and global trading entities[**propper2008towards**][**Boss2004**][**Serrano2007**][**Bech2008**][**Fagiolo2009**][**Morten2006**][**Boss2004a**][**Krempel2002**][**Serrano2003**]. Comparing with blockchain transaction networks, these early studied finical networks are defined not only by transferring activities, but also by lending-based relationship. Moreover, the scale of these networks is much smaller. To conclude, there is rarely research work proposing custom ranking method for large scale transaction graph, especially blockchain transaction graph.

The most relevant work with **Nebulas Rank** is NEM[**nem**]'s Proof-of-Importance scheme. It adopts NCDawareRank[**Nikolakopoulos2013**], which exploits the clustering effect of network topology, as the ranking algorithm, with clustering algorithm based on SCAN algorithm[**xu2007scan**][**shiokawa2015scan**][**chang2017mathsf**]. And **Fleder2015** uses PageRank[**Brin2010**][**page1999pagerank**] as an assisting metric to discover interesting addresses and analyze their activities. However, both NCDawareRank and PageRank are ranking algorithms for webpage network. As we already mentioned in §5.1, blockchain transaction graph is very different from webpage network. And although community structure does exist in transaction graph and should be helpful to handle with spam nodes, it does not suit the consensus purpose mentioned in §5.1. Because in order to compute "unforgeable" node importance, accounts controlled by a single "real world" entity should be guaranteed to be mapped to the same cluster. However it is key difficulty to connect blockchain world with the "real world", and thus there is no proper objective definition for clustering problem. Therefore current clustering algorithms cannot provide meaningful and trustful result. Moreover, [**Fleder2015**]'s work does not provide an automated framework to identify important nodes. Instead, with the help of PageRank, it still take manual analysis as core method, which does not match **Nebulas Rank**'s context.

The algorithm we choose is LeaderRank[**Chen2013**][**Li2014**]. It is a simple variant of PageRank[**Brin2010**][**page1999pagerank**]. In PageRank, initially every node gets one unit rank value. Then at each iteration, every node distributes its rank value equally to its directed neighbors. To deal with dangling node problem, there is a damping factor, where every node distribute a specific proportion of its rank value to all nodes equally in the network. **Chen2013** propose a simple yet effective modification on PageRank's damping factor and call it LeaderRank. Then **Li2014** extend LeaderRank to weighted case and further improve its performance. By weighted LeaderRank[**Li2014**], an additional ground node is added and a bidirectional link is added between every node and ground node. Every edge targeting to ground node is of same weight and every edge from ground node is weighed positively proportional to target node's in-degree. LeaderRank is more resistant against manipulation and noisy data than PageRank[**Chen2013**][**Li2014**][**Lu2016**]. In terms of computation, LeaderRank can be seen as PageRank with one more node and set damping factor to be zero. And thus it is easy to implement and very scalable.

Other than LeaderRank, there are also other algorithms modifying PageRank's damping factor mechanism. For example, **Baeza-Yates2006** proposed a damping function decreasing with distance. Be-

sides, there are some betweenness based algorithms mentioned in [**Borgatti2005**], such as flow betweenness[**freeman1991centrality**] and random walk betweenness (aka. current flow betweenness)[**newman2005measure**]. Although they may be more suitable to represent the flow controlling ability, these centralities are quite computational intensive. So they are not suitable for **Nebulas Rank**. Out of all the current algorithms, we think LeaderRank is a relatively simple yet effective one.

5.3 Transaction Graph

5.3.1 Transactions

The input data for **Nebulas Rank** are all the transaction records, i.e. token transferring, during the past T days, denoted by a set of tuples:

$$T_{xs}^{all} = \{(s, t, \tau, a), \tau = Today - T \dots Today\}$$

, where s, t and a are the source account, target account and amount of an transfer, respectively.

Further, we filter transactions, providing that self transfer and zero amount transfer are excluded:

$$T_{xs} = \{(s, t, \tau, a) | s \neq t \wedge a > \Phi \wedge (s, t, \tau, a) \in T_{xs}^{all}\}, \Phi = 0 \tag{1}$$

5.3.2 Transactions Aggregation

Based on transactions defined above, we construct the directed weighted transaction graph $G = (V, E, W)$, where node set, edge set and weight on edges are denoted by V, E and W respectively. We also denote that $N = |V|$ and $M = |E|$. For simplicity, all nodes are denoted by integer numbers from 1 to N.

Each vertex $v \in V$ represents one individual account's address. Each edge represents the transferring intensity between two accounts. Consider $e = (s, t) \in E$, this edge is directed, and naturally, the weight of it should be determined by all related transactions, i.e. $(s, t, \tau, a) \in T_{xs}$. To compute edge (s, t)'s weight, we take the sum of top K amounts out of all related transactions:

$$w_e = \sum_{i=1}^{K} a_i, s.t. a_i \in \{a | (s, t, \tau, a) \in T_{xs}\} \wedge a_1 \geq a_2 \dots \tag{2}$$

By this mean, the link between two nodes is bi-directed and asymmetric, with top K transactions along each direction aggregated to become the weight. This is different from NEM, which all transfers amounts between two nodes are aggregated into one unilateral edge's weight[**nem**]. We presume NEM's solution is vulnerable to manipulations, since only a simple triangle loop will enhance edges' weight into infinity. Additionally, it is also not truthful to take the average or quartile of all relation transactions, since this forces accounts to transfer very cautiously. We will show the advantage of our aggregation method in §**??** ~~experiment confirming why doing so~~

5.3.3 Temporality Embedding

We noticed that the transactions happens with timestamps. So we try to embed this temporal information as a property of nodes. For each account, we calculate its coinage by the following pseudo-code.

~~formula~~ ~~Defined as C_v normalized by max~~ coinage 的公式和解释

The intuition of coinage is ~~insights~~.

Besides, we conjecture that reducing each transaction's contribution according to its block height, like NEM does[**nem**], encourages users to postpone their transferring until the last day of period, which will cause unnecessary confusion. Instead, **Nebulas Rank** treats each transaction equally, which encourages every account to keep active all the time.

We will talk about the coinage exploitation in §5.3.5. And we will show the advantage of our solution in §?? ~~experiments confirming the effect of coinage~~.

5.3.4 Encouragement Function

~~formula and intuition~~ ~~defined as B_v normalized by max~~ encouragement function 的公式和解释

We will talk about how we apply the encouragement function in §5.3.5. And we will show its advantage in §?? ~~experiments confirming the effect of encouragement function~~.

5.3.5 Exploiting Nodes' Property

We defined two node properties C_v and B_v in §5.3.3 and §5.3.4, respectively. Then we reduce each edge's weight by its target node's properties:

$$w_{(.,v)} \leftarrow w_{(.,v)} \times ln(1 + \frac{C_v + B_v}{2}) \tag{3}$$

5.3.6 Mitigating Dormant Effect

Consider a node receiving a large amount of money but does not spend any. This node forces its money to be "dormant " and prevents money from being circulated, which contradicts with **Nebulas Rank**'s purpose(§5.1). Thus we need to mitigate this dormant effect. In detail, we consider 1-hop local information of each node, limiting the amount of its in-transfers by the total amount of its out-transfers[2]:

$$w_{(.,v)} \leftarrow \frac{w_{(.,v)}}{\sum_u (w_{(u,v))}} min\{\sum_u w_{(v,u)}, \sum_u w_{(u,v)}\}. \tag{4}$$

Intuitively, such restricting method is reasonable: 1) Imagine two phases for a piece of blockchain token. First it is made out of thin air, which is as the reward of the system. Second it is either circulated around the whole network, which almost never stops being transferred from accounts to accounts, or enters dormant state, which the last owner does not spend it out. Formula 4 does not affect the first

[2]Note that formula 4 is applied after formula 3. Formula 3 is applied after definition 2

活跃币龄(Active Coinage)

活跃币龄：在当前周期内，某个地址新接收和新发出的币为活跃币，这些币的币龄为该地址的活跃币龄

```
def calculate(cur, genesis, nodes):
    for node in nodes:
        # cal coinage according to the active coins
        if node.balance < 0:
            coinage = (genesis.no - cur.no) * node.balance
            node.coinage += coinage
            node.balance = 0
        else:
            node.coinage += node.balance

def cal_coinage(nodes, transactions):
    # first block
    genesis = transactions[0].block
    for trans in transactions:
        # next block
        if cur.no < trans.block.no
            # cal coinage till current block
            calculate(cur.no, genesis.no, nodes)
            cur = trans.block
        # record balance
        edge.from.balance -= edge.coins
        edge.to.balance += edge.coins
    # cal coinage till current block
    calculate(cur, genesis, nodes)
```

phase, as edges weights can only be reduced as in-links. And in the second phase, accounts are encouraged to spend enough money in order to improve their in-link quality. 2) From the perspective of money flow, only circulated money should be counted. Nodes with dormant money does not control much of the network flow. That is, deleting these nodes does not affect interactions among other nodes. So formula 4 conforms with **Nebulas Rank**'s Liquidity value (§5.1).

We will show the how **Nebulas Rank** benefits from mitigating dormant effect in §**??**.

~~will this affect wgc????~~

5.4 LeaderRank

5.4.1 LeaderRank Weighting Scheme

We build our scoring algorithm based on LeaderRank[**Li2014**][**Chen2013**]. It does minor modification on the famous PageRank algorithm[**Brin2010**][**page1999pagerank**]. The modification is to add a ground node into the network, in place of PageRank's damping factor. Our method is as follows.

We add one ground node $\mathcal{G}$, numbered as $N+1$, into the network, and double link it with every other node. First every non-ground node sends and receives an amount of "Altruist" money to the the ground node, denoted by A_v. Then every non-ground receives an amount of "Charity" money from the ground node, denoted by C_v. Besides, each node sends an amount of "Surplus" money, denoted by S_v, to the ground node and receives an amount of "Bonus" money, denoted by B_v, from the ground node. Formally, the weighting scheme is given by formula 5 and 6:

$$\forall v \in V, w_{(v,\mathcal{G})} \leftarrow \alpha A_v + \mu S_v \tag{5}$$

$$\forall v \in V, w_{(\mathcal{G},v)} \leftarrow \beta C_v + \lambda B_v \tag{6}$$

We define the altruist and charity money for each node to be equal. And roughly, they should be proportional to the average transaction amount of all nodes:

$$\forall v \in V, A_v = \frac{\sum_{e \in E} w_e}{N}, C_v = \frac{\sum_{e \in E} w_e}{N} \tag{7}$$

Next we define "Surplus" money as the incoming amount of one node minus its outgoing amount:

$$\forall v \in V, S_v \leftarrow \sum_{(u,v) \in E} w_{(u,v)} - \sum_{(v,u) \in E} w_{(v,u)} \tag{8}$$

The intuition is to fix §5.3.6's side effect: after edges' weight are reduced by its targets' dormant effect, there are some nodes whose outgoing amount is less than their incoming amount. Theses nodes don't bring about dormant effect themselves though, they transferred money to some neighbors with dormant money. We suppose these nodes contain "surplus" values, which should not be kept by themselves. Thus all nodes should transfer their "surplus" value to the ground node. This is an extension of **Li2014**'s weighted LeaderRank algorithm.

And we define "Bonus" money as the total amount transferring to the node:

$$\forall v \in V, B_v \leftarrow \sum_{(u,v)\in E} w_{(u,v)} \tag{9}$$

The Intuition is that nodes with mode incoming transfers should have higher probability to receive money from ground node. This is the same scheme as **Li2014** designed.

With the ground node and corresponding edges added, the ranking algorithm can be understood as a Markov chain. States are nodes. Transition probability is proportional to the weight of some node's out-edge. It can be described by an iterative process. Initially all node's rank score is the same except ground node, then every node distribute their rank score among their neighbors:

$$p_i^{t+1} = \sum_u p_j^t \times \frac{w_{(j,i)}}{\sum_k w_{(j,k)}}; p_i^0 = \frac{1}{N} \tag{10}$$

, where p_i^t is node i's rank at the end of t-th iteration.

Equivalently, with the form of matrices,

$$P^{t+1} = H \times R^t; P^1 = [\frac{1}{N}, \frac{1}{N}, \ldots, \frac{1}{N}, 0]^T \tag{11}$$

$P^t \in \mathbb{R}^{N+1}$, represents the rank score for all nodes. And H is a $(N+1) \times (N+1)$ matrix representing transition probabilities of a Markov chain. The element in i-th row and j-column is the probability that a random walk hops to node i from node j, computed as

$$h_{ij} = \frac{w_{(j,i)}}{\sum_k w_{(j,k)}} \tag{12}$$

Since every node is connected with $\mathcal{G}$, the sum of each column of H is equal to one. The convergence of LeaderRank algorithm can be calculated by power iteration. Literatures[**Li2014**][**Chen2013**] show more mathematical details.

After convergence, we get P^*, then distribute ground node's rank among every other node evenly:

$$\forall v \in V, P_v^* \leftarrow P_v^* + \frac{P_{\mathcal{G}}^*}{N} \tag{13}$$

5.4.2 Comparing with PageRank

Comparing with PageRank, we think LeaderRank is more reasonable in the context of **Nebulas Rank**. LeaderRank replace PageRank's teleportation parameter[**Brin2010**][**page1999pagerank**] with ground node mechanism. Teleportation parameter cannot be explained directly from perspective of network flow, while ground node is more understandable.

On one hand, LeaderRank actually enable us to assign different "teleportation parameter" for each node. On the other hand, in the sense of money flow, by PageRank, each node contributes a proportion of their income to the public, and receives the same amount from it. So PageRank adopts a different ground node weighting scheme from our ranking algorithm. And by evenly distributed arbitrary "surfing" probability, PageRank's scheme grants nodes earning lower income with "friendly" ranking. But since **Nebulas Rank** aims to provide some truthfulness, a nodes with lower "income" is more likely to be a sybil

node and thus should be devalued by some more conservative ranking algorithm. There is same problem in NCDawareRank[**Nikolakopoulos2013**], which is also friendly to new nodes with less incoming edges. Together with survey in §5.2, we can conclude that original PageRank and NCDawareRank are not suitable for blockchain transaction graph. This challenges some of the previous studies[**Fleder2015**][**nem**].

5.5 Experiments

5.5.1 Ethereum Stats

~~degree - avg neighbor degree and dynamics; hhi~~

5.5.2 Exploitability of 1-hop Local Information

5.5.3 Noise Resistance

5.5.4 Sybil Attack Resistance

1. top k vs total
2. coinage vs no coinage
3. no reduction by date vs reduced
4. encouragement vs no encouragement
5. no dormant vs dormant
6. leaderRank with all vs others

1. loop
2. star
3. loop star including exchanges
4. random send money and exchanges
5. random graph including exchanges

~~all with comparison~~

5.6 Discussions And Conclusions

6 PoR(Proof of Reputation) 共识算法

6.1 常用共识算法的缺陷

经典 PoW(Proof of Work) 共识算法为零和博弈，采用竞争性哈希计算来确定记账人，导致了整个生态每次出块时都有大量电能在竞争中被无端消耗，挖矿成本高。如果把公链参与者作为整体来看，PoW 协议下生态维持平稳出块的成本将会持续升高。不断增加挖矿难度的 Bitcoin 早晚需要面临矿机收益入不敷出的情形，而 Ethereum 则早已在考虑使用新的 PoS 共识算法 Casper[**casper**] 来逐步取代现阶段的 PoW 共识 [**buterin2013ethereum**]。

PoS(Proof of Stake) 共识算法试图采用资产的多寡来取代算力的作用，按照币龄或者押金数额来分配获得记账权的概率，现阶段 Peercoin[**king2012peercoin**] 和 Ethereum 的 Casper 协议都采用了 PoS 共识算法。这种算法解决了 PoW 高能耗的弊端，但夸大了资本对记账权概率分配的影响，相较于 PoW，在 PoS 下大资本更容易占据生态的话语权，形成大集团垄断，同样不利于公链生态的自由发展。

PoI(Proof of Importance) 共识算法最早由 Nem 提出 [**nem**]，不同于 PoS，PoI 中引入了账户重要程度的概念，使用账户重要性评分来分配记账权的概率。这种算法解决了 PoW 的高能耗弊端，减缓了 PoS 的资本垄断危机，但确暴露了 nothing at stake 的问题，作弊者制造新区块的成本很低，公链受到 51% 攻击的风险较高。

考虑到常用共识算法的缺陷，Nebulas 提出了基于账户声望的 PoR 算法，从强调重要性的 PoI 共识算法出发，融合 PoS 中的经济惩罚，利用 PoS 强化了 PoI 的安全性，而 PoI 则反向遏制了 PoS 的垄断性，为生态自由发展助力。

6.2 PoR 算法设计

6.2.1 新区块产生

PoR 选取生态中有声望的账户来参与产生新区块 (block)，采用 Nebulas Rank 算法，排名 Top N 的账户被视为有影响力声望高的账户，这些账户自愿缴纳一定数量的 Nas 作为押金后则有资格成为新区块的验证者 (validator), 参与记账。

给定验证者集合 (validators set) 之后，PoR 算法通过伪随机数来决定验证者集合中谁是新的区块的提议者 (proposer)，即所有高声望的账户拥有相同的概率获得新区块的提议权，防止概率倾斜衍生垄断。

6.2.2 验证者集合

PoR 为应对验证者集合的动态变化，将验证者集合按照朝代 (dynasty) 做划分。若当前朝代为 D, 那么任何一个有资格的账户只能申请加入或者退出 D+2 朝代的验证者集合。

在 PoR 中定义每 100 个区块为一个 Epoch，在一个 Epoch 中验证者朝代不做变更，每个 Epoch 开始时，考察上一个 Epoch 的第一个区块，如果该区块到达了 Finality 状态，那么当前 Epoch 进入下一个朝代，否则延续上一个朝代不变，如图1所示。

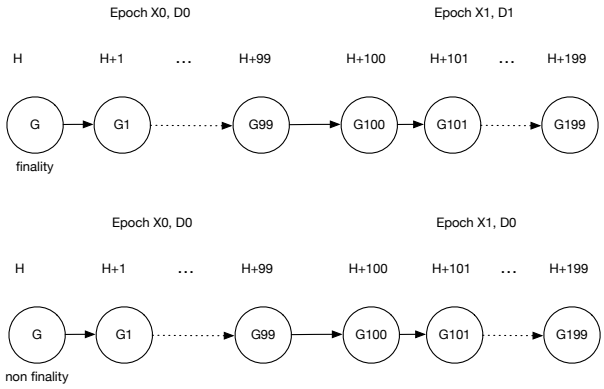

Figure 1: 验证者朝代更迭

6.2.3 共识过程

新的区块被提出后，所有验证者将会参与一轮 BFT(Byzantine Fault Tolerant) 方式的投票，来确定此区块的合法性。在投票最开始，每一个参与此区块共识的验证者将会被收取 2x（x 为激励奖金比例）的罚金，然后进入两阶段的投票过程。

第一阶段，所有验证者需要对新区块投 Prepare 票，投完 Prepare 票的验证者将获得 1.5x 的奖励，如果在当前朝代中有超过 2/3 的押金总额的验证者对新区块投了 Prepare 票，那么该区块进入投票的第二阶段。

第二阶段，所有验证者需要对新区块投 Commit 票，投完 Commit 票的验证者，可以再获得 1.5x 的奖励，如果在当前朝代中有超过 2/3 的押金总额的验证者对新区块投了 Commit 票，那么该区块到达 finality 状态。

为了加速整个生态向前延展，对于任意在高度 H 的区块 b，如果当前最新区块高度大于等于 H+100，那么区块 b 将在共识过程中被视为过期，在过期的区块上产生的任何共识活动都将会被忽略。

6.2.4 分叉选择（**Fork Choice**）

PoR 算法以每个高度上区块的得分来选择权威链，总是选择得分最高的区块加入权威链，在高度 h 的区块 b 的得分如下，

$$Score(b,h) = \sum_{(b',h') \in children(b)} Score(b',h') + \sum committeddepositinb \tag{14}$$

即为该区块及其所有后代区块收到的 commit 票对应的押金总和，如图**??**所示。

6.2.5 投票规则

为了避免共识过程被恶意破坏，导致共识过程没法完成，阻碍生态发展，PoR 参考 Casper 的最小惩罚规则来约束验证者的共识活动。

假设共识过程中的 Prepare 和 Commit 票结构如下，

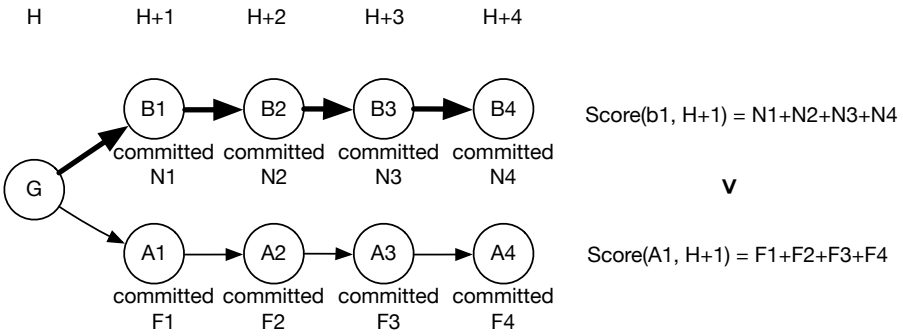

Figure 2: 分叉选择

a) $Prepare(H, v, vs)$，其中 H 为当前区块 hash，v 表示当前区块高度，vs 表示 v 的某个祖先区块高度

b) $Commit(H, v)$，其中 H 为当前区块 hash，v 表示当前区块高度

PoR 算法为整个投票过程制定了如下 4 条基本规则，

1) 单个区块的两阶段共识过程存在严格的先后顺序，只有在第一阶段 $Prepare(H, v, vs)$ 票总权值达到 2/3 后，验证者们才可以投出第二阶段的 $Commit(H, v)$ 票，

2) 多区块间不强制一个区块共识结束后才能开始后一个区块的共识，允许交织共识，但是不能完全没有秩序，只有高度 vs 完成了第一阶段过程，拥有 2/3 的 $Prepare(H_anc, vs, vs')$ 后，才可以基于 vs 对其后代区块投 $Prepare(H, v, vs)$ 票，保证交织稳步向前

3) 为了避免有节点利用交织共识恶意跨多区块投票，要求基于高度 u 投出了 $Prepare(H, w, u)$ 票之后，对于高度在跨度 u 和 w 之间的所有区块，不能再投出 $Commit(H, v)$ 票，保证共识过程的高效有序

4) 为了制止节点用同一笔押金在多个分支上同时下注，导致 Nothing at Stake 的问题，要求在一个高度投出 $Prepare(H1, v, vs1)$ 票之后，不能再投出不一样的 $Prepare(H2, v, vs2)$ 票

违反上述规则的验证者一旦被举报核实，将会被罚掉所有押金，举报者们将会共享罚金的 4% 作为奖励，罚金的剩余部分将会被销毁。

6.3 PoR 经济分析

6.3.1 激励分析

参与 PoR 算法的验证者，在每一个合法区块上可以获得 1x 的星云币奖励，如果网络不畅或者有人作弊导致 Prepare 阶段没有办法完成进入 Commit 阶段，那么所有验证者将损失 0.5x。因此成为验证者的价值节点在保持网络畅通，不参与作弊的情况下，将共享大量记账收益。

6.3.2 作弊分析

双重支付攻击 (double spend)

假设商户 merchant 等到新区块到达 finality 状态就确认交易发货，那么 fraud 要在 PoR 共识算法下完成双重支付攻击实现零成本购物要付出的最小代价如下：

首先，fraud 需要提高自己的 Nebulas Rank 到 Top N，然后缴一定数的 NaS 作为押金成为验证者，并申请参与 D+2 朝代区块的验证。

然后，fraud 需要被伪随机算法选中为新区块的提议者，此时 fraud 提出两个高度相同的新区块，一个哈希值为 hash1 包含 fraud 向 merchant 的转账交易，另一个哈希值为 hash2 包含 fraud 向 fraud 自己的转账交易。

最后，为了让 hash1 和 hash2 区块都到达 finality，如图3所示，fraud 至少需要花费所有押金的 1/3 来贿赂 1/3 的验证者，让他们给两个区块都投 Commit 票。

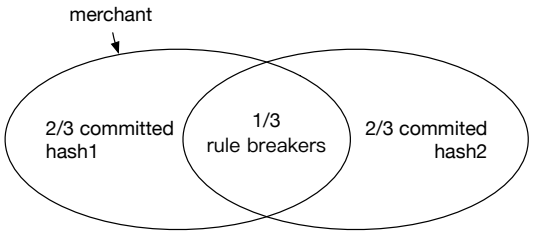

Figure 3: 双重支付经济惩罚

所以要完成一次成功的双重支付攻击，fraud 需要花费一定的资金来提升自己的 Nebulas Rank 排名（见 xxx Nebulas Rank 排名经济分析），然后等到幸运地被选为提议者时，至少花费总押金的 1/3 来让两个块同时到达 finality 状态。

51% 攻击 (51% attack)

在 PoW 中要发起 51% 攻击需要 51% 的算力，在 PoS 中则需要 51% 的押金，而在 PoR 中，则需要验证者集合中 51% 的账户，这意味着拥有足够多的高声望用户进入 Nebulas Rank 的 Top N，并且需要支付对应的押金，因此在 PoR 中 51% 攻击将更为困难。

短程攻击 (short-range attack)

PoR 中的每个高度上的区块都有共识有效期，如果某个高度距离最新高度超过 100 时，该高度的所有区块在共识过程中将被视为过期，那么这些区块上的所有新的共识活动将会被直接忽略。因此要在 PoR 中完成长程攻击 (long-range attack) 几乎不可能，但是在有效期内依旧存在发起短程攻击的可能性。

短程攻击者 Attacker 试图在高度 H+1 的区块还没有过期的情况下，伪造 A 链来替代 B 链成为权威链，Attacker 需要让区块 A1 的得分比 B1 更高。由于多投会被严惩，所以 Attacker 将不可避免地要贿赂

验证者，否则无法完成短程攻击。为了展现 PoR 共识算法的安全性，下面分别分析使不同数量的区块失效时，Attacker 需要付出的代价。

如果 Attacker 想要使 B1 失效，最小代价的情况如图4，就相当一次双重支付攻击，Attacker 幸运地成为了 H+1 高度的区块提议者，那么至少需要贿赂朝代 D0 中 1/3 的验证者多投使 A1 达到 finality，最小代价为所有押金的 1/3。

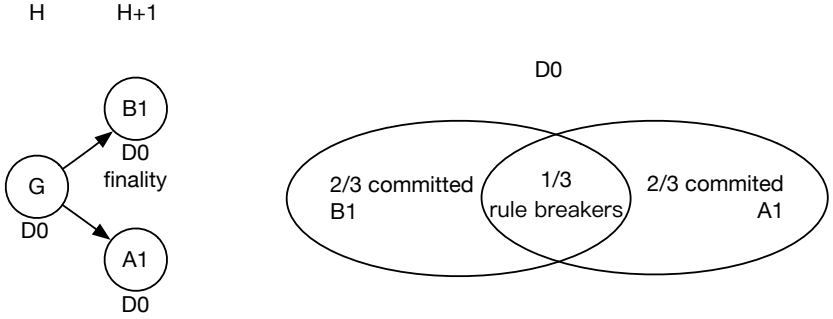

Figure 4: 短程攻击使一个区块失效的情形

如果 Attacker 想要使 B1 B2 失效，假设 B1 和 B2 都已到达 finality，块中交易都已生效，为了让这些交易失效，这里考虑两种情况。第一种如图5中 (a) 所示，高度 H+1 和 H+2 在同一个 Epoch 中，朝代相同，那么 Attacker 首先需要贿赂 D0 中 1/3 的验证者使 A1 达到 finality，此时这 1/3 的验证者将会被惩罚，押金被罚完。在 A2 的验证中整体押金总和只有 A1 中的 2/3，此时 Attacker 想要让 A2 到达和 B2 同价值的 committ 票，需要贿赂剩下所有没有作弊的验证者，完成一次这样的攻击至少需要损失总押金的 3/3，即使如此也不能保证 A1 得分比 B1 高，攻击失败风险高。第二种情况如图5中 (b) 所示，高度 H+1 和 H+2 正好在不同的 Epoch 中，且朝代不相同，那么此时 Attacker 需要贿赂 D0 中的 1/3 来让 A1 到达 finality，然后贿赂 D1 中的 1/3 来让 A2 达到 finality，完成一次这样的攻击至少需要损失总押金的 2/3。综上，想要发起短程攻击导致两个 finality 区块失效，至少需要花费总押金 2/3 的代价。

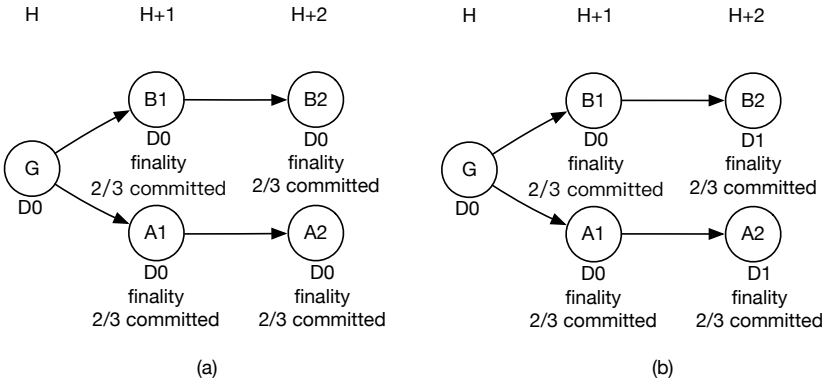

Figure 5: 短程攻击使两个区块失效的情形

如果 Attacker 想要使 B1 B3 失效，如图6所示，Attacker 首先需要贿赂 D0 中 1/3 的人完成 A1 的

finality，然后贿赂 D1 中 1/3 的人完成 A2 的 finality，最后需要贿赂 D1 中剩下 2/3 中的所有人来完成 A3 的 finality，综上至少要损失总押金的 4/3。要完成这些攻击准备将会十分困难，而且即使有幸做到了，也不定能保证 A1 的得分比 B1 高，攻击也可能会失败。

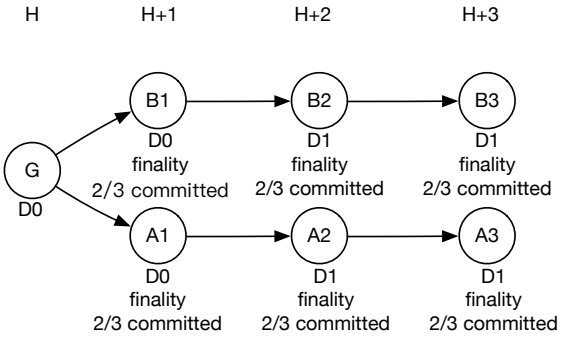

Figure 6: 短程攻击使三个区块失效的情形

如果 Attacker 想要使 B1 BN 失效，其中 N 受到区块共识有效期的限制，不会很大，由于 $N = 3$ 时所有验证者的押金就会被全部罚完，所以 $N >= 4$ 时，将没法完成攻击让 B1 得分比 A1 高，使 B1 BN 失效，发起这样的攻击没有任何意义。

7 DIP 开发者激励协议

在星云链中，我们提出面向智能合约开发者的 DIP(Developer Incentive Protocol 开发者激励协议)，周期性对星云链生态中所有智能合约的价值做评估，通过星云币的奖励来感谢为生态助力的优秀开发者。

DIP 的设计借助周活跃用户价值总和的概念，每周进行一次，和 Nebulas Rank 计算周期一致。对于智能合约 C，假设本周活跃账户地址集合为 wad(weekly active addresses)，其中根据第六章的 Nebulas Rank 分数，计算周活跃地址的 NR 之和作为合约 C 的贡献值。

$$Score(C) = \sum_{addr \in wad} NR(addr) \tag{15}$$

每周贡献值排名 Top N 的智能合约开发者将按比例瓜分 M 个星云币作为奖励，为了避免恶意刷榜，DIP 的分配曲线被设计得较为平均。

$$Coin(C) = kln(N + 1 - Rank(C)) + b \tag{16}$$

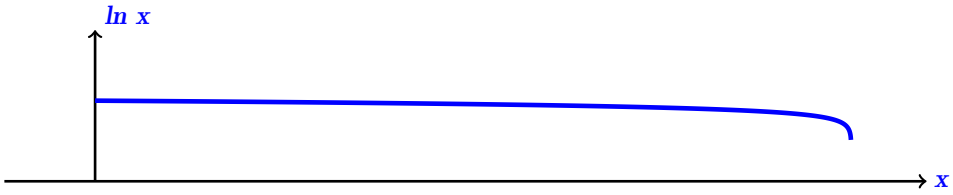

Figure 7: N=3000, M=3000

为了鼓励星云链生态智能合约的多样性，DIP 规定每个智能合约同一个版本最多可以接受 K 次奖励，当使用星云原力对合约做版本更新后，新版本将可以再次接受最多 K 次奖励。DIP 的奖励将会由各个节点单独计算发放，假设星云链平均每 S 秒出一个区块，那么一周将会有 24*7*3600/S 个区块，各个节点将会计算一次 DIP 的奖励，并且发给对应的智能合约的提币地址中。

8 Nebulas Force

wenbo

9 智能合约

Wenbo

10 星云链基础服务及开发工具