Scenarios propagation information through a large network of nodes.

Dissemination of information to a large network of nodes can occur for various scenarios. In some cases, all the nodes have different incentives to disseminate information. Special attention is paid to decentralized electronic currency, including Bitcoin, which is a radical new approach to the monetary system [1-6]. Its cryptographic fundamentals have largely held up even as its usage has become increasingly widespread. However, it exhibits a fundamental problem of a different nature, based on how its incentives are structured. A modification to the protocol that can fix this problem is proposed. Bitcoin relies on a peer-to-peer network to track transactions that are performed with the currency. For this purpose, every transaction a node learns about should be transmitted to its neighbors in the network. As the protocol is currently defined and implemented, it does not provide an incentive for nodes to broadcast transactions they are aware of. In fact, it provides a strong incentive not to do so. The solution is to augment the protocol with a scheme that rewards information propagation. The proposed scheme succeeds in setting the correct incentives, that it is Sybil-proof, and that it requires only a small payment overhead, all this is achieved with iterated elimination of dominated strategies. Lower bounds on the overhead that is required to implement schemes with the stronger solution concept of dominant strategies, indicating that such schemes might be impractical are provided. The scheme must be carefully designed so that it does not provide incentives for the attack.). If no such attacks were made, one possible explanation is the short time span of the challenge and its non-commercial, scientific essence. It seems quite plausible that if the challenge is repeated several times such attacks will become common. The aim is to develop schemes to encourage the dissemination of information to counteract the obstacles resulting from competition from other sites, and with low overhead. Similar is the task of developing a lottery with numbered tickets. Each ticket has the same probability of winning, and the prize is always allocated. As more tickets are sold, the probability that a certain ticket will win decreases. In this case again, there is a clear tension between the organizer of the raffle, who wants as many people to find out about the raffle, and the participants who have already purchased tickets and want to increase their individual chances of winning. The lesson here is simple to make raffles more successful participants should be incentivized to spread the word. One example of a raffle already implementing this is friend trips, in which the more friends you recruit the bigger your probability of winning. As apparent from the previous examples, the tension between information propagation and an increased competition is a general problem. Analyze such conflicts is convenient for example protocol Bitcoin. Bitcoin is a decentralized electronic currency system proposed as an alternative to current government-backed currencies [7]. There are additional properties that some consider as benefits: Bitcoins are not controlled by any government, and its supply will

eventually be fixed. Additionally, it offers some degree of anonymity (although this fact has been contested [8]. Bitcoin has been actively running since 2009. Its appeal lies in the ability to quickly transfer money over the internet, and in its relatively low transaction fees. As of November 2011, it has 7.5 million coins in circulation (called Bitcoins) which are traded at a value of approximately 3 USD per bitcoin. Bitcoin relies on a peer-to-peer network to verify and authorize all transactions that are performed with the currency. Suppose that Alice wants to reserve a hotel room for 30 bitcoins. Alice cryptographically signs a transaction to transfer 30 bitcoins from her to the hotel, and sends the signed transaction to a small number of nodes in the network. Each node in the network propagates the transaction to its neighbors. A node that receives the transaction verifies that Alice has signed it and that she does indeed own the bitcoins she is attempting to transfer. The node then tries to authorize the transaction by attempting to solve a computationally hard problem (basically inverting a hash function). In fact, a node authorizes a set of several transactions together. To keep the presentation simple, this simplified description considers only a single transaction. Proposed solution applies to the case of multiple transactions as well.

Once a node successfully authorizes a transaction, it sends the "proof" (the inverted hash) to all of its neighbors. They in turn, send the information to all of their neighbors and so on. Finally, all nodes in the network agree that Alice's bitcoins have been transferred to the hotel. In compensation for their efforts, nodes are offered a payment in bitcoins for successful authorizations. The system is currently in its initial stages, in which nodes are paid a predetermined amount of bitcoins that are created out of thin air. This also slowly builds up the bitcoins supply. But Bitcoin's protocol specifies an exponentially decreasing rate of money creation that effectively sets a cap on the total number of bitcoins that will be in circulation. As this payment to nodes is slowly phased out, bitcoin owners that want their transactions approved are supposed to pay fees to the authorizing nodes. This is where the incentive problem manifests itself. A node in the network has an incentive to keep the knowledge of any transaction that offers a fee for itself, as any other node that becomes aware of the transaction will compete to authorize the transaction first and claim the associated fee. For example, if only a single node is aware of a transaction, it can eliminate competition altogether by not distributing information further. With no competition, the informed node will eventually succeed in authorizing and collect the fee. The consequences of such behavior may be devastating: as only a single node in the network works to authorize each transaction, authorization is expected to take a very long time. Bitcoin's difficulty level is adjusting automatically to account for the total amount of computation in the network. The expected time of a single machine to authorize a transaction is currently on the order of 60 days, instead of the 10 minutes it is expected to take if the entire network competes to authorize. It is important to note that false identities are an important problem Bitcoin. In fact, the Bitcoin protocol is built around the assumption that nodes can create false identities, and considers a transaction fully approved only once nodes that control a majority of the

CPU power in the network have accepted it, rather than just a majority of the nodes. The latter is vulnerable to Sybil attacks. Therefore any reward scheme for transaction distribution must discourage such attacks. Propose a model in Bitcoin. The presentation here is a bit. For simplicity assume that only one transaction needs to be authorized. Authorization process is modeling as consisting of two phases: the first phase is a distribution phase. The second one is a computation phase, in which every node that has received the transaction is attempting to authorize it. In general, it is common to think of an efficient peer to peer network as a random graph, in which messages propagate quickly, multiplying the number of recipients as the message is sent to each additional layer. In the general case of random graphs the problem is unsolvable. It is easier to solve, assuming that the network consists of a forest of d-ary directed trees, each of them of height B. The intuition for this simplification is that when d is small relatively to the number of nodes n, message propagation in a random graph resembles a tree. In some sense, the case of trees is harder than general graphs as each node monopolizes the flow of information to its descendants. In the initial phase distribution of the buyer sends the transaction details to the root of the trees (seeds). SEED is a block cipher developed by the Korean Information Security Agency. It is used broadly throughout South Korean industry, but seldom found elsewhere. It gained popularity in Korea because 40-bit SSL was not considered strong enough (see 40-bit encryption), so the Korean Information Security Agency developed its own standard. However, this decision has historically limited the competition of web browsers in Korea, as no major SSL libraries or web browsers supported the SEED algorithm, requiring users to use an ActiveX control in Internet Explorer for secure web sites.[9] As of late 2009, the NSS software security library in Mozilla's Gecko platform has implemented support for SEED and Mozilla Firefox as of 3.5.4 supports SEED.[10] Unfortunately support for SEED alone is not enough to allow for secure transactions with Korean web services. SEED is a 16-round Feistel network with 128-bit blocks and a 128-bit key. It uses two 8 × 8 S-boxes which, like those of SAFER, are derived from discrete exponentiation (in this case, x247 and x251 – plus some "incompatible operations"). It also has some resemblance to MISTY1 in the recursiveness of its structure: the 128-bit full cipher is a Feistel network with an F-function operating on 64-bit halves, while the F-function itself is a Feistel network composed of a G-function operating on 32-bit halves. However the recursion does not extend further because the G-function is not a Feistel network. In the G-function, the 32-bit word is considered as four 8-bit bytes, each of which is passed through one or the other of the S-boxes, then combined in a moderately complex set of boolean functions such that each output bit depends on 3 of the 4 input bytes. SEED has a fairly complex key schedule, generating its thirty-two 32-bit subkeys through application of its G-function on a series of rotations of the raw key, combined with round constants derived (as in TEA) from the Golden ratio. The information about the transaction is flowing from the root towards the leaves. Assumptions about the total number of nodes can be found in [1]. In the

distribution phase, each node v can send the transaction to any of its neighbors after adding any number of fake identities of itself before sending to that neighbor. All v's fake identities are connected to the same set of children. A node can condition its behavior only on the length of the chain above it, which can possibly include false identities that were produced by its ancestors. In the computation phase each node that is aware of the transaction tries to authorize it. If there are k such nodes, each of them has the same probability of 1/k to authorize it first. When the node succeeds in authorizing a transaction some have a winning chain of the seeds. Each node on the path in the tree from the seed to the authorizer appears on this chain (nodes are not able to remove their predecessors from the chain due to the use of cryptographic signatures), so the observed chain is a superset of the real path in the tree (it potentially includes duplicates). For allocating rewards on the winning chain we look at it in reverse order, reward is allocated to the authorizing node and rd to the seed. Some assume that there exists a minimal payment c that at least covers the expected computation cost for a single node to authorize the transaction by itself such that any node that is promised a reward of c will attempt to authorize the transaction. Thus, some require that the authorizer reward rd is at least c. In the actual Bitcoin protocol many transactions are authorized together, and thus the minimum reward rd per transaction is relatively small since the computation cost is split between many transactions. For a smaller reward, nodes will refuse to participate in the first place (due to individual rationality). Some normalize c to be 1. Every reward scheme some consider in the paper is completely defined by non-negative rewards rd for every seed s and every $1 \leq i \leq b$. The reward rd is the reward given to the i-th identity on the i,h winning chain of length b that starts at the authorizing node. A reward scheme is individually rational for B in a tree rooted by s if for every $1 \leq b \leq B$ some have that rewards ≥ 1. Award scheme will encourage the dissemination of information without duplication. That is, it will be in a node's best interest to distribute the transaction to all its children without duplicating itself, as well as never duplicating when it authorizes. Main goal is to have information of the transaction reach the majority of the nodes in the network. It may be achieved with small rewards, while minimizing the number of SEEDs (so the burden of the initial distribution is as low as possible). New Reward Schemes is the main result [1]. It is the Hybrid scheme. A reward scheme requires only a constant number of seeds and a constant overhead. Start is introducing a new family of schemes: almost uniform schemes. Each member in the family is parameterized by a height parameter B and a reward parameter β. Let v be the node that authorized the transaction and suppose that v is the l'th node in the chain. If $l > B$ no node is rewarded (so nodes far from the seed will not attempt to authorize the transaction). Otherwise, each node in the chain except v gets a reward of β, and v gets a reward of $1 + (B - l + 1)\cdot\beta$. If there are some seeds, only strategy profiles that exhibit information propagation and no duplication survive every order of iterated removal of dominated strategies. Furthermore, there exists an order in which no other strategy profiles survive. This gives us two interesting schemes, for

two different values of β. The first is when β = 1. In this case the (1, B)-almost-uniform scheme requires only a constant number of seeds and the total payment is always O(B). The second scheme is the (1/B, B)-almost-uniform scheme. This scheme works if the number of seeds is Ω(B). Its total payment is 2. Combine both schemes to create a reward scheme, which has both a constant amount of seed and fixed overhead costs. The hybrid scheme first distributes the transaction to a constant number of seeds using the almost-uniform scheme. Then we can argue that at least B nodes are aware of the transaction. This fact enables us to further argue that the (1/B, B)-almost uniform scheme B guarantees that the transaction is distributed to trees of height B. At the end of the distribution phase, most of the nodes that are aware of the transaction receive a reward of 1/B if they are in the successful chain, so the expected overhead is low. In the hybrid rewarding scheme, if the number of seeds $t \geq 14$, the only strategies that always survive iterated elimination of dominated strategies exhibit information propagation and no duplication. In addition, there exists an elimination order in which the only strategies that survive exhibit information propagation and no duplication. Furthermore, the expected total sum of payments is at most 3. Notice that this scheme exhibits in equilibrium low overhead, Sybil proofness, and provides the nodes with an incentive to propagate information. Iterated removal of dominated strategies is the following common technique for solving games: first the set of surviving strategies of each player is initiated to all its strategies. Then at each step, a strategy of one of the players is eliminated from the set. The strategy that is eliminated is one that is dominated by some other strategy of the same player (with respect to the strategies in the surviving sets of all other players). This process continues until there is no strategy that can be removed. The solution concept prescribes that each player will only play some strategy from his surviving set. In general the surviving sets can depend on the order in which strategies are eliminated. Yet, shows that in our case, regardless of the order of elimination, profiles of strategies in which every node propagates information and never duplicates, survive. Moreover, for a specific order of elimination they are the only strategies that survive. The intuition behind the elimination process is that decreasing competition by your own descendants might be unprofitable due to the distribution rewards. Consider one particular node. If the amount of external competition from non-descendent nodes is small, it prefers not to distribute the transaction, and thus increase the probability that it receives the reward for authorization. However, if sufficiently many non-descendent nodes are aware of the transaction, the node prefers to duplicate itself one less time, and thus distribute the transaction to its children and increase its potential distribution reward. Once all nodes increase distribution, the arms race begins: the competition each node faces is greater, and again it prefers to duplicate itself one less time and distribute all the way to its grand-children. As this process continues, all nodes eventually prefer to distribute fully and never to duplicate. Iterated removal of dominated strategies is a strong solution concept, but ideally we would like our rewarding scheme to achieve all desired properties in the

stronger notion of dominant strategies equilibrium. However, shows that the dominant pattern in each strategy scheme either the amount that the scheme must pay in equilibrium is huge, or the number of initial seeds t must be very large. Every individually rational reward scheme that propagates information to at least half of the network, and in which no-duplication and information-propagation are a dominant strategy for all nodes, has expected definite payment. Notice that for the sum of total rewards to be constant the number of seeds t has to be a significant part of the network. This implies that dominant strategy schemes are quite impractical. The
paper describing Bitcoin's principles was originally published as a white paper. A protocol was since developed in an open-source project. To date, no formal document describes the protocol in its entirety, although the open source community maintains wiki pages devoted to that purpose. The basic setup of electronic transactions relies on public key cryptography. When Alice wants to transfer 50 coins to Bob, she signs a transaction using her private key. Hence, everyone can verify that Alice herself initiated this transaction (and not someone else). Bob, in turn, is identified as the target of the transfer using his public key. For the money to be actually transferred from Alice's account to Bob's account, some entity has to keep track of the latest owner of the coins, and to change this owner from Alice to Bob. Otherwise, Alice could double spend her money – First transfer the coins to Bob, then transfer the same coins again to Charlie. Traditionally, this role was fulfilled by banks. In return, banks tended to charge high fees, for example in international transfers. Agreeing on the History by Majority of CPU power Bitcoin suggests a different solution to this problem. A peer to peer network is used to validate all transactions. Nodes in the network agree on a common history using a majority of CPU power mechanism. A mechanism cannot rely on a numerical majority of the nodes, as it is relatively easy to create additional identities in a network, for example by spoofing IP addresses. . Let us assume again that Alice wants to transfer 50 coins to Bob. Alice will send her signed transaction to some of the nodes in the network. Next, these nodes will forward the transaction to all of their neighbors in the network and so on. A node that receives the transaction first verifies that this is a valid transaction (e.g. that the money being transferred indeed belongs to Alice). If successful, the node adds this transaction to the block of transactions it attempts to authorize (the specific details on the structure of this block are omitted). To authorize a block, a node has to solve an inverse Hash problem. More specifically, its goal is to add some bits (nonce) to the block such that the Hash value of the new block begins with some predefined number of zeroes.
The number of zeroes is adjusted such that the average time it takes the network to sign a block is fixed. The protocol uses a clever method to aggregate transactions (a Merkle tree) which assures that the size of the block to be hashed is also fixed. Additional information that is included in the block is the hash of the previously authorized block. So, in fact, the authorized blocks form a chain in which each block identifies the one the preceded it. When a node authorizes a block it broadcasts to the network the new block and the proof of work (the string which is added to the block

to get the desired hash). If a node receives two different authorized blocks it adopts the one which is part of the longer chain. Let us argue briefly and informally why the history is determined by the majority of the CPU power ([14]). That is, the probability that a group of malicious nodes manages to change the history decreases as the fraction of CPU they control decreases. Assume that a group of malicious nodes, that does not have the majority of the CPU power, wants to change the chain that has currently been adopted by the other nodes. Since the malicious nodes own less CPU power, their authorization rate is slower than the authorization rate of the majority of the network, which is honest. Therefore, the longer chains will be signed by the majority of the network (with high probability) and these are the ones that will be adopted by the honest nodes in the network. In fact, once a block has been accepted into the chain, the probability that a longer chain that displaces it will appear decreases exponentially with time (as the chain that follows it grows longer it is harder to manufacture a longer alternative one). To incentivize nodes to participate in the peer to peer network and invest effort in authorizing transactions, rewards have to be allocated. Currently, in the initial stages of the protocol this is done by giving the authorizer of a block a fixed amount of bitcoins (this also the only method for printing new bitcoins). This fixed amount will be reduced every few years at a rate determined by the protocol. As the money creation is slowly phased out, there will be a need to reward the nodes differently. The protocol has been designed with this in mind. It already allows the transaction initiator to specify a fee for authorizing her transaction. This introduces an incentive problem since nodes prefer to keep the transaction to themselves instead of broadcasting it. Some research has been conducted with regards to its privacy [5]. The subject of incentives for information dissemination, especially in the context of social and peer-to-peer networks has received some attention in recent years. Kleinberg and Raghavan [11] consider incentivizing nodes to search for information in a social network. A node that possesses the information is rewarded for relaying it back, as are nodes along the path to it. A key difference between their model and [1], is that nodes either possess the sought-after information or do not. If they do, then there is no need for further propagation, and if they do not, they cannot themselves generate the information, and so do not compete with nodes they transmit to (they do however compete for the propagation rewards with other unrelated nodes). Emek et. al. [12] considers reward schemes for social advertising scenarios. In their model, the goal of the scheme is to advertise to as many nodes as possible, while rewarding nodes for forwarding the advertisement. Unlike their scenario, the scheme is eventually only aware of a chain that results in a successful authorization, and only awards nodes on the path to the successful authorizer. Other works consider propagation in social networks without the aspect of incentives, for example [13], which considers ways to detect propagation events and examines data from cellular call data. A challenging open question is to consider the setting where the network is modeled as a random d-regular graph. Another interesting extension to consider is one in which nodes have different computation power.

References

1. Dobzinski S., Oren S., Zohar A. On Bitcoin and Red Balloons. // http://arxiv.org/pdf/1111.2626.pdf (дата обращения: 13.07.2012).
2. Introduction to an electronic currency, network payment and electronic commerce.//**URL: http://www.econf.rae.ru/article/6950.** (дата обращения: 11.08.2012).
3. The electronic currencies networks: a pseudo-anonymous electronic currency system// **http://www.econf.rae.ru/article/6953.** (дата обращения: 14.08.2012).
4. An analysis of a Heisenbot uncertainty problem: a large anonymous online marketplace. // URL: http://econf.rae.ru/article/6957 (дата обращения: 17.08.2012).
5. Surowiecki J. Cryptocurrency. Technology Review, MIT. September/October, 2011 http://www.technologyreview.com/computing/38392/ . (дата обращения: 22.08.2011).
6. Davis J. The crypto-currency: Bitcoin and its mysterious inventor. The New Yorker, October 10, 2011.
7. The Bitcoin wiki. https://bitcoin.it
8. Reid F., Harrigan M. An analysis of anonymity in the Bitcoin system. arXiv/1107.4524, 2011. (дата обращения: 14.08.2012).
9. Gen Kanai (2007-01-26). "The Cost of Monoculture". 2007-01-29.
10. Mozilla Corporation (2009-08-09). "Bug 478839 - Firefox should support South Korean SEED crypto cipher suites". 2009-08-09.
11. Kleinberg J., Raghavan P. Query incentive networks. In FOCS ’05: Proc. 46th IEEE Symposium on Foundations of Computer Science, pages 132–141, 2005.
12. Emek Y. et al. Mechanisms for multi-level marketing. In Proc.12th ACM conference on Electronic commerce, EC ’11, pages 209–218, NY, USA, 2011.
13. Dyagilev K. et al. Generative models for rapid information propagation. Proc. First Workshop on Social Media Analytics, SOMA ’10, pages 35–43, NY, USA, 2010.
14. Satoshi Nakamoto.Bitcoin: A peer-to-peer electronic cash system. // http://bitcoin.org/bitcoin.pdf (дата обращения: 17.08.2011).