1 Introduction

The theory of evolution has been studied for many centuries as it is a bewildering thought that from the raw elements that make up our planet a living organism arose and evolved into our sophisticated, complex society. Since the ground-breaking work of Darwin some mechanisms that fuelled that evolution have been widely accepted in the scientific community, namely natural selection, the survival of the strongest random mutation of a species that inherits its phenotype to the next generation, creating a fitter generation. This theory about competition between individuals was the accepted truth about evolution but in the late 1960s doubts arose about the completeness of this theory. When looking at group behavior in species one will find that cooperation is a common theme among related individuals, yet there is no place for cooperation in the classic Darwin theory.[5] In their work Axelrod and Hamilton [5] analyze how to combine the seemingly inferior individual's strategy of cooperating with the goal to maximize fitness. At the basis of their experiments is a game called Prisoner's Dilemma [6], in which two players can choose to cooperate or defect: if both cooperate they fare best, if both defect they fare okay, yet if one cooperates and the other defects the cooperator farest the worst. This creates a social dilemma: not knowing what the other does, makes defecting the only reasonable strategy, yet if both would trust each other to cooperate, they would both win. Axelrod and Hamilton ran experiments on this game with multiple rounds with different strategies and found that if the game is played multiple rounds with the possibility of meeting the same partner again in the future, cooperation between players can be established and be superior. Later research showed that this direct form of reciprocity, the act of returning a deed, is only one form of cooperation found in species and human behavior. Nowak and Martin [11] defined five forms in which cooperation can occur: kin selection, direct reciprocity, indirect reciprocity, network reciprocity and group reciprocity. Conceptually these forms can be described like this:

- kin selection: we help those that share our genes
- direct reciprocity: I help you, you help me
- indirect reciprocity: I help you, somebody helps me
- network reciprocity: neighbors help each other
- group selection: A group, in which members help each other, survives

These mechanisms for fostering cooperation entail an important concept: trust. We trust those that are close to us and we trust those that we have had succesful interactions with before. If two entities trust each other, they are able to cooperate and as a result, prosper. If trust is broken both parties will fare worse.

Trust and cooperation lay the basis for human societies and have a long history. A prominent example is the evolution of economy as shown in Figure 1.1. In the pre-industrial age, most economy and trade was done in local communities with families that trusted each other over generations and traders that returned year after year. During industrial and post-industrial age companies have largely replaced local producers and are trusted by millions of customers based on their brand name. Nowadays, in the information age, internet companies allow people to connect, trade and cooperate directly, examples being eBay for trading physical goods, AirBnb for sharing houses and Uber for ride-hailing. How trust and the lack of it influence trade and markets has been studied in the well-known paper by Akerlof [4]. He describes the information asymmetry

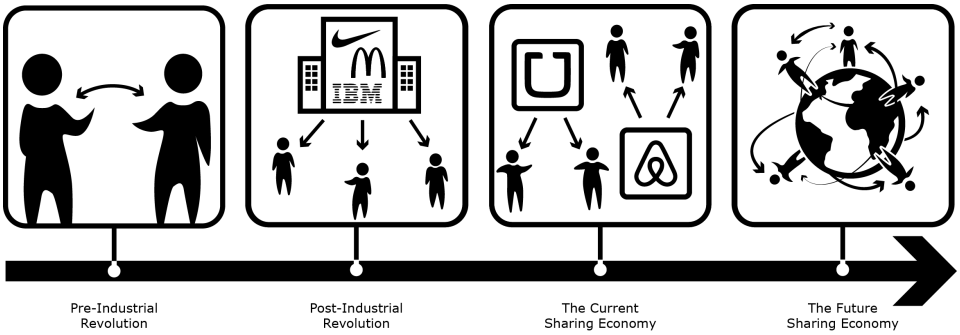

Figure 1.1: Evolution of the economy

between seller, who knows the quality of the goods which will be sold, and the buyer, who can only estimate that quality by some market statistic. The seller's incentive to sell goods of lesser quality than the average statistic leads to a decreasing statistic and thus price which in turn decreases the quality of the goods sellers are willing to offer for that lower price. Hence the market breaks down. Akerlof describes institutions to solve this problem namyly institutions such as guarantees, brand names and certifications. These are trust inducing institutions and can be generalized as reputation systems. If the sellers sell goods to many people and those people report or gossip the good quality of what they have bought, others can trust those sellers and both seller and buyers will thrive. On the other hand a bad reputation will lead to a seller getting out of business as buyers will mistrust. This closes the gap to the work of Nowak as this reputation is what makes indirect reciprocity possible: the seller is not taking advantage of the buyer's inconvenient situation but the buyer cannot directly return that favor. Only by gossipping the event to other potential buyers who are then more willingly to buy from the seller is the reciprocity circle closed. [11]

These analog, gossip-based reputation systems are what guide our decisions in buying cars, new or used, at which bank we store our money and at which restaurant we should have dinner. But reputation systems are also prevalent in the digital world: we make our decisions in buying used goods on eBay, renting a house to a stranger (or from a stranger), getting into a stranger's car (what mum told us not to) based on the reputation of the partner. The sharing economy or collaborative consumption is the rising star of economic concepts in the information age and it is power by reputation. A company offers a platform on which the two sides of a trade or transaction can find each other. With each encounter both parties can rate that interaction and it becomes part of their history. With a longer and more positive history the value of a profile increases as users see the reputation as security for a good interaction and are willing to pay for it. However, there are reasons for concern. What if the platform changes their rules in an almost unacceptable way or abuses the personal data their users have entrusted them with? Users cannot take their reputation and data to another platform because their reputation is actually owned by the platform facilitating the trades. Such abuses in which trusted companies act wrongly have been happening in many times in the past, examples being the dieselgate [1] and the facebook/cambridge analytica scandals. [2] These scandals show that eventhough millions of users trust them, central institutions do not neccessarily serve their customers or clients.

We envision a future in which collaborative consumption is possible without any intermediator. This future requires a reputation system which is application agnostic, owned by noone and ruled by everyone. A distributed reputation system as a layer directly on top of the internet. However this poses some challenges from a technical point of view. Distributed system are intrinsically hard to control and regulate, which is both blessing and curse. No party can impose unfair rules on other users but it is also hard to prevent malicious users from sending wrong information across the network. [7] reviews state-of-the-art reputation systems and finds that all commercial reputation systems are centralized. Some of the scientific reputation systems are decentralized like EigenTrust [8], P-Grid [3] and RateWeb [9], yet they have not been proven to work in settings where high throughput, global scaling are required which is the case for a global reputation system. Distributed, secure and globally scalable systems remain an unsolved problem.

This report is related to research done at the Blockchain Lab at TU Delft, whose ambition it is to be the first to create a working solution for this problem. In many years of research multiple milestones have been reached. In [10] we have solved the free-riding problem in the peer-to-peer file-sharing context with a reputation system that tracks uploads and downloads. With TrustChain [12] we have created our own blockchain fabric for bandwidth as a currency which builds on the previous work and adds tamper-proof recording and

immutable history to the reputation system. All work is implemented in Tribler, a BitTorrent client with Tor-like layered anonymity. The implementation allows for testing of research ideas with real users in production environments.

This work specifically will be concerned with the agreement on reputation in the network and closely related to that the dissemination of records of interactions. We will enlarge on the problem description in the next chapter. Afterwards the problem will be defined formally and analyzed in the bounds of the definition. Before proposing a solution, some existing approaches for recording and dissemination of data will be discussed in chapter. Next, we define a solution based on the TU Delft blockchain fabric TrustChain and propose a specific mechanism of using such a fabric. Finally, we prove the correctness and scalability properties of the fabric in experimental analysis, before concluding and making suggestions for further research.

2

Problem description

Abstract. Our audacious ambition is to design and create a layer of reputation on top of the core infrastructure of the internet that enables application agnostic trustful relationships between relative strangers. Such a global reputation systems needs to be distributed, scalable, tamper-proof, robust against strategic manipulation and misbehavior. Reputation system require public visibility of reputations however the state of large distributed systems are inherently unobservable. Together with a general subjectivity due to the nature of the concept of trust this creates a discrepancy between agents perception of the trustworthiness of their peers. The main question underlying this work is how diminish this discrepancy. locality?; trust vector or number; reputation or trust system?; automated trust system; quantifiable?;

In the introduction we have made a case for our audacious ambition to design and create a layer of reputation on top of the core infrastructure of the internet that enables application agnostic trustful relationships between relative strangers. This requires a distributed, scalable reputation system. Reputation systems appear in many forms but we are concerned with their digital form as only digital networks can reach global scale with fast information distribution. Reputation systems have previously been formally defined to include at least three components [14]:

- Long-lived identities
- Recording and distribution of feedback about interactions
- Use of feedback to guide interactions

We need entities to be identifiable and in existence for a long time to ensure that future interactions between known entities are likely. If changing of identities is easy, a bad reputation is easily discarded and exchanged for a clean slate. We need to capture and distribute feedback of interacitons such that entities are aware of the history and reputation of other entities on the system. Finally, users should actually make use of the feedback on not just ignore it.

Next to the requirements of reputation systems we also have requirements for specific usecase of a global reputation system without centralized institutations.

- distributed: no entity should be owner of the reputation of all people, no single point of failure should exist
- scalable: future applications similar to those that exist today with centralized reputation systems should be able to handle billions of users.
- robust against strategic manipulation: once reputation increases in worth users of the system will try to exploit the system by attacking it, alone or by colluding and the architecture needs to be robust against such attacks

This introduces additional challenges: enforcing long-lived identities is even harder without the assumption of a trusted central entity that can check the validity of new entities. Also creating a central distributed record with synchronization at scale is a topic of ongoing research and generally seen as an unsolved problem.

Finally, reputation system in general and distributed systems specifically are intrinsically weak in protection against malicious behavior, although they are robust in terms of complete failures as no single point of failure exists.

As of today, no single algorithm or architecture can provide a solution that conforms with all these requirements. Only by combining multiple components and iterating their design can we approach a reputation layer that is able to conform with the requirements. This layer needs to combine these components:

Identity. At the lowest level there is the identity layer that ensures an entity is identifiable for other entities in the network. The most basic version of this is a simple public- private key pair for signing and encrypting data. But creating a new key pair is cheap, therefore this is not enough and in a later iteration of this identity system digital entities will need to be bound to real-world, verified entites like government-issued passports or biometric identifiers.

Communication. The internet creates a global communication network with high connectivity across the world but it is currently not in a state that direct communication between devices is straight-forward. The large increase in connected devices expended the address space of IPv4 and IPv6 transition has been slow, thus network address translation creates subnetworks with local adress spaces. Connection from such a subspace to a server with a public adress is still simple like it is the case with most client-server applcations on the internet, but direct communication when both devices are behind NATs is still not standardized. Also new routing solutions are required to ensure communication based on the actual identity layer mentioned before instead of the IPv4 and DNS identity layer.

Record and distribution mechanism Reputation is based on feedback on interactions. This feedback needs to be recorded and distributed, such that other entities in the network have a chance to respond to the history of feedback of their peers. Without a central entity which is aware of all transactions, each node will record some transactions. It is a challenge to create a global record which is correct, tamper-proof and well distributed across the network.

Interpretation of records Based on the recorded feedback each agent can interpret them to form an opinion about other nodes. For a reputation system the records are seen as positive or negative behavior and each agent can output a ranking of reputations for all peers this agent knows of. Those rankings are calculated based on a reputation function. In the past our research group has analyzed different reputation functions like NetFlow [12], MaxFlow [10] and PageRank [13].

Application layer We imagine that the reputation system we are developing can be used for any type of application that requires two entities to trust each other. This reputation layer will be accessible for anyone however no application will be able to delete data or lock data into their proprietary platform.

Each layer adds another level of protection: if identities are expensive and hard to create, fake identities will be less easy to create, protecting the system against spamming. Creating an immutable record and distributing that information to everyone makes knowledge tamper-proof, unchangeable and forever, making all information reliable. On the interpretation layer additional securities can be enabled: we envision a concept of locality to secure against distributed attacks with global collaborations of malicious nodes - if we only trust agents with a certain level of latency attackers can only choose from nodes in the vicinity and supply of those nodes is limited.

The concept of trust is however vague. In the analog world, trust is more of a feeling than a rational calculation, yet computing systems are deterministic, precise and rational. The vagueness partly stems from the subjective interpretation of actions which in the digital system is the reputation function used, but also from the difference in what each person knows about another one, or in digitally speaking subsets of information that each one agent has. Yet, from a practical point of view it would be desirable if people could agree on the reputation and trustworthiness of agents, because it makes actions predictable and gives reputation its value.

To see this we should go back to the discussion of indirect reciprocity, which is one of the five forms of fostering cooperation which were introduced in chapter 1. People act prosocially at a personal cost in order to build an altruistic reputation which is rewarded with third-parties acting prosocially towards them. Reputation is valuable because people with higher reputation can expect more cooperation in future interactions. This indirect reciprocity mechanism works as long as agents agree on what is good and what is bad

reputation. Once there is ambiguity about the reputation of agents this value decreases as even people with a bad reputation could be seen as good people by others due to that ambiguity. Also only if reputation has actual value to agents, we can ensure prosocial behavior in the network. Therefore we need agreement on the reputation of agents.

Agreement on the interpretation layer can be achieved by defining a function for all agents to use which calculates a quantitative reputation from the history of feedback. Usually this history is public, visible to everyone, however this is difficult to achieve in a distributed system. Here the second problem comes into play. The information a network node acts upon is a different subset of complete information on the network for each agent; each agent is in a different state. This situation is undesirable but inherent to systems with the requirements stated in the previous section. Thus, a first step towards agreeing on the reputation of agents is if agents agree on which data should be used as an input to the reputation function. In other words we have to make sure that agents disseminate their knowledge and obtain knowledge from other agents such that information is well distributed and available.

However, in most contexts sharing and obtaining information comes at a cost which is not negligible. Thus agents may be reluctant to spend resources without any direct reward. There is an obvious network effect to agents knowing more about their peers but agents can also gain reputation by cooperating with agents with low reputation. Thus there is no incentive to obtain a better view of the network.

The question that we are trying to answer is therefore:

How can we design a scalable, distributed feedback recording and distribution mechanism that makes dissemination and verification of transaction records incentive compatible?

3

Formal Problem Definition and Analysis

In the previous two chapters we have introduced the problem of state-of-the-art commercial reputation systems and explained how a distributed reputation could solve this problem but requires among other things a strong transaction recording and distribution mechanimsm. In this chapter we formally define a model in which we can analyze this mechanism.

Each agent maintains a subset of the global information and calculates the perceived reputation of peers based on that. In order to approximate the true reputation of peers an agent needs more data or multiple rankings.

4

Related work

In the previous two chapters we have shown that a need exists for a decentralized accounting system in order to create a global infrastructure for secure, anonymous digital transactions that does not require control through a trusted third party. This need has been identified before and work has been performed both in the scientific community as well as the industry. In this chapter we will summarize those efforts, describe the short-comings of those approaches and define a basis for the work performed in this work.

4.1. Applications of decentralized accounting systems

The general concept of accounting is quite old as it is simply a recording of transactions between two or more parties. Before the digital age those recordings were simply written text on paper, nowadays those recordings are stored in databases. We are concerned with another type, namely decentralized accounting systems. We identified three types of applications for decentralized accounting systems: cryptocurrencies, distributed work systems and reputation systems.

4.1.1. Cryptocurrencies

In the years 2007 and 2008 the global financial crisis shattered the global economy, lead to many people loosing house and job and diminished the trust clients had in banks to keep their money safe. Politics discussed the problem and proposed to regulate the banks more but with little impact. However something else promised to change the banking world: the first white-paper for a decentralized digital currency without any need for a trusted third party, Bitcoin, was announced.

Bitcoin. Before the announcement of Bitcoin it was assumed that in order to verify the correctness of transactions between parties and prevent cheating with digital money a bank or credit card company was needed. Bitcoin proved them wrong by creating a hash-based chain of transaction blocks, a global ledger, that is shared among all users of the network. The acceptance of transactions is managed by a process called “mining” which ensures that only the majority of CPU power can publish new block. A blocks contains a fixed number of transactions and the Bitcoin network makes sure that a block is created once every 10 minutes. All mining node will execute the proof-of-work mechanism: in order to publish a block a value needs to be found that, when hashed with a certain hashing function like SHA-256, starts with a certain number of zeros. Depending on how many CPUs are active on the network the problem can be increased in difficulty by requiring more zeros at the beginning of the hashed value. Once a new block is published other nodes will validate the transactions and if they agree, will show their acceptance by working on creating the next block. This system ensures that as long as a majority of CPU power is owned by honest nodes, they will outpace the rest of the network in solving the hashing puzzle and creating valid blocks. Nodes will accept the longest chain and the transactions will be valid.

The Bitcoin approach solved many problems assuming that an honest majority exists: first and foremost the double-spending of funds is prevented because the Bitcoin blockchain creates one global order of valid transactions. Also the Sybil-attack is prevented by pairing the voting power to the available CPU power, which means Sybils can only run on real hardware, removing the advantage of fake identities. But these measures of attack prevention come at a price of efficiency. The surging price of Bitcoins especially in the year 2017 led

to a surge in transactions, transaction fees and energy usage. The increasing price of Bitcoins makes mining them more profitable which means more nodes are joining the mining operation. Therefore the difficulty for the proof-of-work problem is increased, such that it takes more computing power to find a correct value. This again increases the amount energy consumed in the whole network. At the same time the number of transactions processed is a constant of the Bitcoin currency, approximately 7 transactions per second. At the time of writing the energy conusmption is at least 2.55 GW which makes it comparable to contries such as Ireland. Summarized Bitcoin was a large step towards decentralized accounting but unsolved scalability issues still prevent it from being actually useful as an infrastructure such as the one we envision.

Alternative coins and improvement measures. Bitcoin served as a first proof-of-concept for trustless digital currencies or for our purposes, a "secure" decentralized accounting system, but the shortcomings were also obvious. Once the popularity increased, other enthousiasts, startups and incumbent companies started to create their own spin-off digital currency. Each of these so-called "alternative coins" used blockchains as a core technology to store transactions but tried to solve the scalability issues using different approaches. The discussion of all alternative coins goes beyond the scope of this chapter, therefore we will quickly introduce some of the main differences between the largest systems.

The block time is one parameter to tweak in order to increase transaction throughput. Ethereum, the second largest cryptocurrencies currently uses a block time of 15 seconds with a proof-of-work consensus. Also block size is a factor in the throughput rate, but increasing block time and size only creates a constant factor to the rate of transactions.

Ethereum is currently testing a proof-of-stake mechanism which should replace the energy intesive proof-of-work. In short this mechanism will require "minders" to put some amount of currency into a wallet in order to participate in the process. If a miner does not perform the validation of transactions correctly that "stake" will be lost for the miner. This will solve the energy consumption problem but it will not solve the overall scalability issue of the system.

Another feature in development in multiple currencies is the "Lightning network". The lightning network will allow two parties that expect to conduct multiple transactions with each other to create a "channel". Both parties store some funds in the channel and can then interact freely through this channel without needing to interact with the master network of the currency. Only the opening and netbalance at closing time will be writting to the chain while all other interactions are only recorded locally. This should increase the possible throughput significantly but due to the early stages of development the actual implications of large-scale use are not proven at the time of writing. But considering that Bitcoin has a transaction limit of 200000 transactions a day, it would still take 5000 days or 13.7 years to open one channel each for a billion people.

The IOTA project ...

Sharding ...

Conclusion

4.1.2. Distirubted work systems

In the field of distributed computing many applications include some mechanism in which a node is performing work for other nodes or the network in general. Seuken et al. call these distributed work systems. Some examples of distributed work systems are peer-to-peer file-sharing network, packet forwarding in mobile ad-hoc networks and volunteer scientific distributed computing. As our research group is mostly concerned with file-sharing networks and the concepts are similar in general we will stick to that example to discuss the latest developments.

Many different file-sharing networks have been built in the past, the most prominent being Napster, Gnutella and BitTorrent. In contrast to centralized file-sharing, in peer-to-peer systems there is no server that contains all data, but instead users share data directly, one peer downloading and one peer uploading. With no infrastructure needed, no costs and no single point of failure such a systems seems optimal. Talking in terms of distributed work systems, the act of uploading is equivalent of performing work while the act of downloading consumes work. There is, however, a social dilemma here: uploading to another node does not lead to an immediate reward for the uploading node, therefore, if we assume that bandwidth is a precious resource it is cheaper to not upload, yet if all agents on the network realize this, no agent will upload and thus no agent is able to download. The agents that do not upload any data are known as free-riders and free-rider protection in peer-to-peer file-sharing networks is a subject of ongoing research.

Accounting systems pose a possible solution to the free-riding problem. Let's first imagine a centralized accounting systems keeping track of all uploading and downloading behavior, uploading data increases the

balance of agents, downloading decreases the balance. Now, the accounting system can enforce that agents keep their balance around 0, so they upload approximately as much as they download. Therefore, an accounting system can solve the free-riding problem, however as mentioned multiple times, a decentralized accounting system is hard to implement. Accounting mechanisms have first been related with this subject in the DropEdge paper, however a lot of work has been done on the very related subject of reputation systems, which will be discussed in the next section. Seuken et al. define an incentive-compatible accounting mechanism which removes any advantage for users that misreport their own contributions in the network. They present their DropEdge algorithm and show that it's possible to increase the efficiency of BitTorrent clients using accounting. A negative result of their work is that an accounting mechanism cannot prevent sybil attacks. Some short-comings of the approach is strategic manipulations of data and dissemination of data.

4.1.3. Reputation systems

One of the reasons that decentralized accounting systems are hard to create is that agents in peer-to-peer applications do not have a complete view of the network and thus also not all information of the network, at least not without a global consensus mechanism. In the file-sharing example from the previous section agents decide to upload to other agents based on some partial knowledge of the network and contributions of agents. It can be argued that an accounting mechanism cannot be correct if it acts on partial information and instead the particular balance of an agent as seen by another agent is rather a reputation. The goal is then to create trust between users in order to facilitate cooperation. Such a system will be called a reputation system.

Whether reputation systems can be called an application of accounting systems can be argued about. In general accounting systems track transactions between accounts, the full history of transactions determines the state of the network. According the framework of Mui et al. trust is the expectation of reciprocation for an agent given that agent's history of behavior. So a reputation system can act on the data of an accounting system and add additional conclusions. The previous example of agents uploading and downloading helps to understand this. An accounting system keeps track of the transactions and calculates the balance of an agent, for example +10MB, for an agent that has uploaded 10MB more than downloaded. Also it is possible to account the total uploaded and downloaded data, for example 1010MB and 1000MB respectively. A simple accounting system stops at this point, the system behaves correctly when no error has been done in calculating the balances and the data is correct. A reputation system adds another layer of interpretation to this data. The simplest reputation function only checks whether the balance is positive or not, or if the choice is between multiple agents, whose balance is the most positive. Another reputation function might weight agents with a 0 balance but 10GB of uploaded (and downloaded) data more trustworthy than an agent with 10MB positive balance but only 100MB uploaded data. Thus we can see a reputation system as a layer on top of an accounting system.

Describe some reputation systems ...

4.2. TrustChain

TrustChain was built as a system to create trust between two strangers

4.2.1. Data structure

4.2.2. Accounting mechanism

Definition of trust and reputation

4.2.3. Subjective graph

4.2.4. Consensus

5

Agent state transparency

As previously shown, the TrustChain solution is a scalable, high-throughput, immutable append-only database. However, it is lacking state transparency which makes sharing of information not enforceable. We can define an extended architecture which adds state transparency to TrustChain 1. We have shown that the fact that states of agents are not recorded on the blockchain adds all kinds of ambiguities about their behavior such that we cannot enforce rules in the system, as we cannot verify their behavior. 2. By recording the root hash of the Merkle tree of the database on each block, agents will be required to prove of which blocks their database exists. However, just this is not enough as from the hash itself no other agent can verify that the database is consistent with the data received by other agents. 3. In order to solve this second problem, we have two options: 1. In each interaction in which blocks are transferred/exchanged, the sending party needs to sign a transfer block which includes the root hash of the Merkle tree of the blocks sent. 2. With each block transfer, both agents exchange all blocks such that they achieve the exact same state of the database (anti-entropy). In that case they don't need to record the exchanged blocks but can each calculate the hash of the database and if they agree, they know that their databases are the same. 4. This makes available some powerful verification tools: 1. First of all, the addition publicly shows the knowledge of an agent. That means an agent cannot lie about not having some information or having information. Telling the truth about his information is strategy-proof. 2. We can replay the life-time of an agent and completely verify the behavior including application specific rules.

6

Reputation consensus through anti-entropy

7

Experiments and results

8 Discussion

8.0.1. Strategy proofness
8.0.2. Attack resistance
8.0.3. Future research

Bibliography

[1] How volkswagen's "defeat devices" worked. https://www.nytimes.com/interactive/2015/business/international/vw-diesel-emissions-scandal-explained.html. Accessed: 2018-06-14.

[2] Facebook and cambridge analytica: What you need to know as fallout widens. https://www.nytimes.com/2018/03/19/technology/facebook-cambridge-analytica-explained.html. Accessed: 2018-06-14.

[3] Karl Aberer, Philippe Cudré-Mauroux, Anwitaman Datta, Zoran Despotovic, Manfred Hauswirth, Magdalena Punceva, and Roman Schmidt. P-grid: a self-organizing structured p2p system. *ACM SIGMOD Record*, 32(3):29–33, 2003.

[4] George A. Akerlof. The market for "lemons": Quality uncertainty and the market mechanism. *The Quarterly Journal of Economics*, 84(3):488–500, 1970. ISSN 00335533, 15314650. URL http://www.jstor.org/stable/1879431.

[5] R Axelrod and WD Hamilton. The evolution of cooperation. *Science*, 211(4489):1390–1396, 1981. ISSN 0036-8075. doi: 10.1126/science.7466396. URL http://science.sciencemag.org/content/211/4489/1390.

[6] A.R.A.M. Chammah, A. Rapoport, A.M. Chammah, and C.J. Orwant. *Prisoner's Dilemma: A Study in Conflict and Cooperation*. Ann Arbor paperbacks. University of Michigan Press, 1965. ISBN 9780472061655. URL https://books.google.nl/books?id=yPtNnKjXaj4C.

[7] Ferry Hendrikx, Kris Bubendorfer, and Ryan Chard. Reputation systems: A survey and taxonomy. *Journal of Parallel and Distributed Computing*, 75:184 – 197, 2015. ISSN 0743-7315. doi: https://doi.org/10.1016/j.jpdc.2014.08.004. URL http://www.sciencedirect.com/science/article/pii/S0743731514001464.

[8] Sepandar D Kamvar, Mario T Schlosser, and Hector Garcia-Molina. The eigentrust algorithm for reputation management in p2p networks. In *Proceedings of the 12th international conference on World Wide Web*, pages 640–651. ACM, 2003.

[9] Zaki Malik and Athman Bouguettaya. Rateweb: Reputation assessment for trust establishment among web services. *The VLDB Journal—The International Journal on Very Large Data Bases*, 18(4):885–911, 2009.

[10] Michel Meulpolder, Johan A Pouwelse, Dick HJ Epema, and Henk J Sips. Bartercast: A practical approach to prevent lazy freeriding in p2p networks. In *Parallel & Distributed Processing, 2009. IPDPS 2009. IEEE International Symposium on*, pages 1–8. IEEE, 2009.

[11] Martin A Nowak. Five rules for the evolution of cooperation. *science*, 314(5805):1560–1563, 2006.

[12] Pim Otte, Martijn de Vos, and Johan Pouwelse. Trustchain: A sybil-resistant scalable blockchain. *Future Generation Computer Systems*, 2017. ISSN 0167-739X. doi: https://doi.org/10.1016/j.future.2017.08.048. URL http://www.sciencedirect.com/science/article/pii/S0167739X17318988.

[13] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford InfoLab, 1999.

[14] Paul Resnick, Ko Kuwabara, Richard Zeckhauser, and Eric Friedman. Reputation systems. *Commun. ACM*, 43(12):45–48, December 2000. ISSN 0001-0782. doi: 10.1145/355112.355122. URL http://doi.acm.org/10.1145/355112.355122.