Consensus through arbitration: An application of Game Theory methods to distributed calculations in trustless environments.

Maxim Orlovsky*, Sabina Sachtachtinskagia†

2 November, 2017

Abstract

We use Games Theory methods to study the incentives of blockchain network participants that rent their hardware to provide distributed calculations. We propose how to design the network in such a way that all those participants, following their "egoistic" incentives and maximizing their own profits, end up contributing to network's success. As a result, it becomes possible to do distributed calculations reliably with very little work repeated for purpose of quality control. The nature of our solution is to encourage network participants to have skin in the game, tying meritocratic rewards to risks and contributions by means of smart contracts.

Keywords: Blockchain, Decentralization, Game Theory, Networks, Smart Contracts

JEL Codes: L14, L15, L17

1 Introduction

One strong modern trend is decentralization. Widely known are partially decentralized platforms like Amazon Marketplace, eBay, Etsy, Uber, Lyft, AirBNB and similar projects that share similar

*Department of Cytology, Bogomoletz Institute of Physiology, National Academy of Sciences, Ukraine
†Department of Economics, Athens University of Economics and Business, Greece

traits: they substitute many of the usual vertical chain relations with decentralized supply and demand. Even more heavily decentralized projects come from the space of Distributed Ledger techology and its popular application, the Blockchain, which is leading the world trends nowadays. These projects tend to operate without central governance at all - instead, the informal community leaders suggest the ways for the network development and the participants "vote with their feet". Such a high level of decentalization requires significant automation of tasks, so that the network's everyday operation does not require supervision and administrative involvement. This automation is supported by the emerging technology of smart contracts (currently most fully implemented on Ethereum protocol), which allows to create blockchain contracts that can be fullfilled automatically, depending on predetermined observable conditions of the network.

Decentralization usually brings multiple cost savings for various reasons, such as: spare resource sharing, outsourcing to countries with lower costs, crowdsourcing of creativity, network effects and synergies, less bureaucracy, leaner administration, no middleman services and greater transparency. However, one problem that plagues decentralized projects is the difficulty of quality assurance, and in other words this problem can be described as protecting the network from attacks. In distributed ledgers, and in particular in blockchains, there are additional restrictions that make this task of quality assuranse or network protection difficult. Blockchain networks a) operate in trustless environments which may be prone to moral hazard problems and disruption attacks, b) by design, decentralized networks cannot take cenralized actions, therefore the quality checking needs to be done in a decentralized way within the network[1], c) there are severe financial constraints because, since every blockchain transaction is autonomous, its cost cannot exceed its benefit for the network and subsidies between transactions are not possible.

We attempt to develop a model that would estimate the possible severity of the problem and provide a solution. We use Games Theory methods to study the incentives of network participants and propose ways to align their incentives with that of the network as a whole. The network is therefore redesigned in such a way that all participants, following their "egoistic" incentives and maximizing their own profits, arrive to contributing to network's success.

The proposed solution is twofold. Firstly, we introduce a more meritocratic approach, where network participants need to assume responsibility by putting something at stake if they want to

[1]Theoretically, quality checking / network protection could also potentially be outsourced to a third party, but that would raise many questions like: implicit centralization, trust for this third party and its motivations, and the complexity of creating a programmable contractual environment for connecting the third party to the network.

receive gains from the network. The greater the rewards that a participant opts for, the greater is the stake that will be lost if the participant does not produce contributions of acceptable quality. Secondly, we develop a process of arbitration, which is essentially a process of quality control and network protection applied specifically to distributed ledgers. Arbitration is about re-checking random parts of work done by network participants and resolving conflicting opinions whenever they occur. We also delevop instruments that help to estimate the probability and costs of attack on the network, whether for personal gain or for disruption of its services. Finally, we attempt to align the incentives of all network participants so that the network would be able to use quality control sparingly, e.g. checking only 10%-15% of all work done, and still achieve high quality results[2].

2 The Model

2.1 The motivating case

Our motivation for this model was the recent Pandora Boxchain project that is a market for big data, M.L. kernels and distributed calculations for A.I. on blockchain. We limit out study to one aspect only: the distributed calculations, since this case is quite representative of the industry. Like many other distributed ledger networks, Pandora has encountered a typical challenge: while providing significant cost-savings, it needed to develop an inexpensive and efficient way of quality control for its distributed calculations. Speaking more techincally, the network must have a way to reach a consensus without repeating (re-checking) too much work. Failure to reach a proper consensus would open the network to attacks, as network participants would be able to reap rewards without contributing adequately. The solution described in this paper was proposed by Pandora team members[3]. The model that we develop below uses distributed calculations as an example market, but it may be adapted virually for any blockchain economics sector.

[2] Checking only a small part of calculations by using economically incentivized, randomly selected validators is the most striking difference of our model from the recent work by Teutsch J. and Reitwiessner C. (2017) that describes the Truebit project. In contrast, their approach is to use a pool of validators that work simultaneously for a bounty. While adding to network's reliability, that raises the costs for checking to over 500% of the cost of the calculation itself according to the same authors estimation.

[3] Apart from the authors, other members of the Pandora team that participated in delevoping this solution are Andrey Sobol and Olga Ukolova.

2.2 Key characteristics of the blockchain setting

The setting that we study involves a multi-way platform, on which economic agents transact for distributed calculations[4]. The quality of calculation (whether they are right or wrong) is verifiable, but verification is costly because it requires redoing the calculation anew.

This platform is build on blockchain. The network is generated when nodes simultaneously run opersouce software.

The platform operates in a decentralized way, which means that there is technically no place for centralized authority that could act as "deus ex machine" for intervention or resolution of disputes. As a result, the solution to any arising problems must come endogenously from network protocol.

The platform operates in a trustless environment. This means that implicitly there is no trust between agents and no financial claims can be made outside of the protocol that sets the rules for the network. Such a setting requires careful aligning of the incetives of network participants with the network as a whole, so as to avoid problems like moral hazard, free riding etc.

The platform is transparent. This means that any record that is made cannot be altered everafter and, after a certain moment, will be visible to all. When calculation results are reported by any agent, a record is set. The network protocol keeps this record secret until it is considered as verified by another agent, and then releases it to be fully transparent and costlessly verifiable by anyone.

2.3 Types of agents and implementation of stakes

Concentrating solely on the distributed calculations market, the platform has three types of agents: clients, Worker nodes and Validator nodes[5].

Clients are buyers of calculations. They can be, for example, businesses or researchers.

Worker nodes and Validator nodes are owners of GPU hardware that is required to do the calculations. Nodes are asked to deposit into smart contract a collateral stake, denominated in network's tokens. If nodes are found Faulty, they lose all of their stake; therefore the role of stake is to allow for punishment in case of cheating. Also, smaller fines, also written in smart contracts, could help to build responsibility e.g. for staying within pre-agreed timeframe.

[4] These calculations are done on GPU hardware and use as input big data and A.I. kernels.

[5] We omit big data providers and A.I. kernel creators, because they have already given their inputs and are not active when distributed calculations are made.

Let the standard stake be w for Worker node and v for Validator node, where $v > w$. In this study, we will assume "standard" stakes, but in reality these levels can be minimal stakes, increased if the node wants so: more responsibility could lead to greater rewards, e.g. greater frequency of task assignment and more high-paid assignments.

In addition to stakes, a platform could implement reputation for nodes (and in fact the Pandora project does so). We will not use reputation in our main model here, but will examine briefly what effects that would have below in chapter 4.1

2.4 Operation of the network

A simplified presentation of typical transaction during the everyday operation of the network would be as follows.

Stage 1. The Client formally describes the calculation that he needs, provides the necessary big data and kernels, defines the level of quality of calculations that he desires, and deposits on smart contracts the payment for work. Consider s as such standartized payment for a typical task[6].

Stage 2. The network protocol sends the task to random Validator node. The Validator node, in turn, splits the task to smaller assignments and sends each assignment to a random Working node (randomness for both Validator and Worker is weighted by the nodes' stakes, where greater stake leads to higher chance of receiving a task and to higher chance of receiving a more high-paying task).

Stage 3. The Worker node does the calculations, incuring cost c (e.g. electricity bills and amortization of hardware equipment). Then it returns the result, protected cryptographically so that the Validator cannot read it yet, to blockchain. Alternatively, the Worker might report a fake result without doing calculations and without incuring cost. We call this option "cheating".

Stage 4. The Validator node receives the result, which is at the time cryptographically protected, and stores it. Then it does a partial checking of calculations. Let's call p the share of the calculations that are checked in validation. A logical viable level would be $p = (0.10, 0.30)$, i.e. 10%-30% of calculation duplication. Then, p is also the probability to discover a cheating, at the network level. Alternatively, the Validator might report a fake result without doing calculations and without incuring cost. However, the Validator node does not know the answer that the Worker node has

[6]It does not play a role in our model how exactly the amount of this payment is derived. In our motivation case of Pandora, it is the minimum payment that nodes are going to accept for their work, and that depends primarily on the cost of electricity and hardware.

given, until Validator's answer is published.

Stage 5. The answers of Validator and Worker are revealed. If they are the same, both nodes are paid for their work and the results are given to Client. If answers are not the same, then arbitration starts.

2.5 Arbitration procedure

The proposed arbitration procedure is as follows. Arbitration is done by Validator nodes in three rounds.

Round 1 is required to identify the problem. As described above, one random Validator node routinely checks selectively a small part of the assignment done by one single Worker node. If its calculations produce same results, the Worker node receives rewards for the whole task. If the Validator node finds results different to those given by the Worker node, then the protocol declares the Worker node to be Faulty. A Faulty Worker node loses its stake[7], unless it appeals. In order to appeal, it needs to raise its stake to match that of a Validator node - this way it contributes the resource to pay for additional calculations checking.

Round 2 is started if a Worker appeals by raising its stake to match the stake of a Validator node. In this case, a random College of arbitration is gathered that it comprised on any small number of Validator nodes, preferrably 3 (being a small and odd number). The purpose of this stage is to resolve cheaply and efficiently the dilemma whether the Worker node or the Validator node is faulty. Each Validator of the College reaches a verdict and then the verdicts are revealed simultaneously. If their verdict is unanimous, then the College is rewarded by splitting the stake which used to belong to the Faulty Validator node or to the Worker node[8]. However, if their verdict is not unanimous, then arbitration continues to the next round.

Round 3 is started automatically if the College of arbitration is not in agreement, i.e. if there is at least one disagreeing Validator. In such case, a serious and massive attack on network is suspected, since already at least two nodes have pledged their Validator stakes in disagreement (one from each previous round). To protect the network, exceptional measures are taken, which guarantee to resolve the problem. One simple solution would be to assign a large number of random

[7] This stake is transferred to the Validator who found that the Working node is Faulty. Therefore the Validator node has incentives to do the check thoroughly. On the other hand, it is easy to prove that the Validator node will not misreport Working nodes as Faulty, since that will lead to appeal and ultimately cost Validator its own stake.

[8] If the stake of the Worker was not raised at the moment of appeal, that could distort the College's motivation; however when stakes are equal, the College has no preference which of the nodes to strike down.

Validators on the task and pay them for work by dissolving the stake of the Faulty Validator(s) of previous stages. This solution is described in more detail in chapter 3.4 below. A second solution would be to assign some special, Master Validators whose very high stake and / or reputation ensures that they are loyal to the network and eager to protect its functionality. A third solution would be to crowdsource the verdict to the whole network. In any of these solutions, the final result would be similar: as long as most network stakeholders remain loyal to the network, the verdict would be the correct one.

3 Aligning of economic incentives

3.1 Two possible strategies for cheating

There are two possible strategies for cheating.

The first strategy is to create Work Nodes and hope to stay undetected. If detected, do not raise the stake for appeal, but let the node be destroyed.

The second strategy is to create both Validator Nodes and Work Nodes. If detected, raise the stake for appeal and attempt to win the arbitration.

In order to ensure that the network is safe, we need to check for both strategies to be unprofitable.

3.2 The basic rule for cheating to be unprofitable

In order for the network to operate, producing nearly perfect results in equilibrium, cheating must become unprofitable.

Let's calculate the profit from cheating.

The expected payoff when not cheating is $s-c$, i.e. standard payment for the assignment, minus the incured cost.

The expected payoff when cheating is $(1-p)s+p(-w)$, i.e. the expectation of standard payment s without any doing work minus the expectation to lose one's stake collateral deposit w.

Therefore, cheating is not profitable when $s - c > (1 - p)s + p(-w) \implies p(s + w) > c$

Observe that the result depends on all four parameters. It depends positively on the probability of detection p, the standard payment s and the stake w; it depends adversely on the cost of calculations c.

3.3 Quantification of results

In order to quantify the above results, we need to make reasonable assumptions about the relation between payment s and cost c. This depends on the "markup" (profit margin) $\frac{s-c}{c}$. If $\frac{s-c}{c} = 0.1$ then this markup is 10%, therefore, $s = 1.1c$. If $\frac{s-c}{c} = 1$ then this markup is 100%, therefore $s = 2c$.[9]

The table below uses the formula $p(s + w) > c$ and shows how high the stake w need to be, relative to calculation cost c, in order to deter cheating.

p	$\frac{s-c}{c}$	w
0.1	0.1	$8.9c$
0.1	1	$8c$
0.2	0.1	$3.9c$
0.2	1	$3c$

Note that the results are not too sensitive to a change in the markup. Instead, they are very sensitive to a change in probability of detection p.

Therefore we find that reasonable Worker stakes w are sufficient to make cheating non-profitable.

3.4 Cheating in arbitration round 2

Now that we have found how to make plain cheating unprofitable (cheating stragegy 1), it remains to prove that it stays unprofitable when the cheater drags the network into arbitration (cheating strategy 2). In order to get into arbitration, the cheater had to raise his stake, therefore his stake is now not w, but $v > w$. Obviously this strategy makes sense only when the cheater expects to pass through arbitration and emerge victorious. This can be true only if he owns a significant percentage of the network's Validator nodes, since to pass through round 2 of arbitration, he needs to get all 3 Validator nodes to give an unanimous verdict. In at least one Validator disagrees, round 3 of arbitration will start.

Suppose that the cheater owns n% of validators (e.g. $n = 0.5$ for owning half of validators). The chance to get all 3 validators selected from his ownership is approximately n^3. Then, the expected payoff is $n^3 * 0 + (1 - n^3) * (-v) = v\left(n^3 - 1\right)$. The alternative payoff that is gotten by not raising the stake and not going to arbitration is $-w$. Therefore, it is profitable to go to follow this strategy

[9] Note that any gains of the Working nodes from mining, a task that is quite common in blockchains, would simply equal to an increase in s.

after being detected, only if $w > v\left(1 - n^3\right)$. This proves that for some reasonable values of the parameters, cheating in round 2 (without cheating in round 3) is unprofitable.

For example, let's assume that a validator stake is equal to 2 worker stakes: $w = \frac{1}{2}v$

Then, $\frac{1}{2}v > v\left(1 - n^3\right) \Rightarrow n > 0.7\dot{9}$. This means that if we set a sufficiently high validator stake v at least 2 times the worker stake w, this cheating strategy is unprofitable unless the cheater controls over 79% of Validator nodes. Therefore, we may conclude that the network is safe.

3.5 Feasibility of arbitration round 3

Finally, we need to show that the round 3 of arbitration can work and give a true verdict, deterring cheaters who enter into round 2 from escalatung to round 3. To show this, we will use the simple assumption that arbitration is done by a large number or arbitrators - as many as can be financed by the stakes of Validators when they are found Faulty.

If the process of arbitration reaches round 3, it means that at least 2 Validator node stakes are about to be confiscated by the smart contract (one Faulty node from each previous round). Therefore, to show the feasibility of arbitration, it suffices to show that 2 Validator stakes are sufficient to pay for arbitration by multiple nodes. In other words, it needs to hold that $2v \geq ns$ where n is the desired number of validators in round 3 so as to consider the arbitration as successful[10]. This is the condition that makes cheating in round 3 unprofitable, because it guarantees that there will be sufficient funds to pay for the desired quality of cheking.

For example, suppose that $v = 3w$ and $w = 10c$, while the usual profit margin is 100%, i.e. $s = 2c$. Then, the 2 confiscated stakes v would be redistributed to the network as 15 standard payments s to spend on mass-verifying one calculation, attracting $n = 15$ random Validators that would give an estimate of what network thinks is a consensus.

As we have mentioned above in Chapter 2.5 there are other and possibly better (although more complex) ways to do the final arbitration, but we show here that it can work in principle and there is no problem of insufficient funds to sponsor the checking.

[10] The desired number n of randomly chosen Validators need to be (arbitrarily) large, so that is represents the consensus.

4 Possible extensions

4.1 The role of reputation

We are only going to describe briefly the implications of introducing reputation, but will not quantify it, since that would unnecessarily complicate our basic model; we might enrich our model with reputation effects in future works.

Suppose that each successful competion of the assignment adds to reputation of the node; it can be a number that is tracked by protocol. If reputation, just like stake, leads to greater rewards for the node and becomes a prerequisite for the Validator status, then reputation has value for the node. In that case, the threat of losing all reputation has a very similar effect to the threat of losing stake. Additionally, reputation is personal and non-transferrable, unlike the stake, and it is built very gradually, so it captures better the expected loyalty of a node to the network.

Since reputation mainy amplifies the effect that the stake has, it will suffice for now to study only the effect of the stake, which is more easily quantifiable. Any solutions that work adequately with the stake system, would work only better if reputation is added.

4.2 A better validation method

So far we have used a validation rule where a Validator was checking p percent of the whole task. However, usually cheating does not tend to be widespread on the network uniformly, instead usually concentrates on "recyclable" nodes with mininum stake / reputation. We should therefore, whenever technically feasible, validate not just a fixed percentage of one task, but a fixed percentage of node's assignment, becoming node-centric rather than task-centric. If a Worker node that does not perform any calculations, checking even a small portion of every assignment given to a node would easily lead to detection and punishment of Faulty nodes. Of course, the cheating nodes may then employ sophisticated strategies, e.g. doing only partial calculations, but still this validation rule centered on nodes would be quite effective.

4.3 Owners of multiple nodes

If the structure of node ownership is such that includes entities who own a big share of network's nodes, we need to account for strategic interaction between the random Validator and the random Worker node. Assume that an entity who owns many nodes can identify that the same task has arrived randomly to its Validator node and to its Worker node. In that case, the owner of both

nodes can cheat with impunity (provided that the Clients does not do his own checks). How severe is the problem? Irrespective of how many Worker nodes one owns, the probability of cheating with impunity depends solely on the percentage of Validator nodes owned.

The remedy could be twofold. Firstly, it is possible to let the Client to appeal - even a slim chance of that could make cheating unprofitable because of additional risks for the cheater. Secondly, it might be possible to make transactions between nodes zero-knowledge, so that the Validator cannot recognize what it validates.

In addition, the problem is less severe than it may look, because those who have invested heavily in stake and in reputation, enough to acquire multiple Validator nodes, already have their incentives aligned with the network's success and therefore would hesitate to cheat because that would endanger the network.

4.4 Protection against costly attacks

So far we have examined only cheating for profit. However, for various reasons (ranging from vandalism to competition motives), a network might be attacked in a way that does not bring immediate profits to the attacker. The model that we have used to calculate the impact of cheating also provides the tools to estimate how costly the attacks will be for the adversary. Often it suffices to take the functions that measure profit from cheating and use them to measure attacker's loss. We have identified a variety of possible attacks and have been checking for the robustness of the network, the results pending publication.

5 Conclusion

In this paper, we have examined the incentives of network participants to cheat the network by reaping rewards without doing the proper contribution. These observations are not limited to the market for distributed calculations, but are valid in any decentralized economic sector where free rider problem presents itself. We have searched for resolutions of this problem by means of the network itself, without any centralized intervention, and we recommend to introduce the skin in the game factor (node stakes). In addition, we recommend a three-step arbitration process that may have variations, but generally is centered on i) detection of the discrepancy in outcomes, ii) attempt to resolve the dilemma cheaply and efficiently, iii) if impossible, then calling to crowdsourcing to protect the network from attack.

Specifically in our model, we find that since detection of cheating is imperfect, it could be profitable to report fake calculations results without calculating, hoping to remain undetected. We also find that, once cheating is detected, arbitration works quite well and does not present a weak point. The most effective attacking strategy is to cheat, but surrender (and not raise) the collateral stake if caught. Therefore it is recommended to increase the network's ability to deter this cheating strategy. This can be done either by improving validation methods or by setting carefully the parameters of the network, primarily by keeping the stakes of nodes high enough, to ensure that those who reach for rewards have skin in the game.

References

[1] Ethereum http://Ethereum.org

[2] Teutsch J., Reitwiessner C.: "A scalable verification solution for blockchains", Working paper, 2017.