CAGE

Voting over a distributed ledger: an interdisciplinary perspective

CAGE working paper no. 416

June 2019

(Revised August 2020)

Amrita Dhillon
Grammateia Kotsialou
Peter McBurney
Luke Riley

Previously titled:
Introduction to voting and the blockchain: some open questions for economists

WARWICK
THE UNIVERSITY OF WARWICK

UKRI
Economic and Social Research Council

Voting over a distributed ledger: An interdisciplinary perspective

Amrita Dhillon* Grammateia Kotsialou† Peter McBurney‡
Luke Riley§

King's College London, UK

August 24, 2020

Abstract

This work discusses the potential of a blockchain based infrastructure for a decentralised online voting platform. When compared to paper based voting, online voting can vastly increase the speed that votes can be counted, expand the overall accessibility of the election system and decrease the cost of turnout. Yet despite these advantages, online voting for political office is subject to fraud at various levels due to its centralised nature. In this paper, we describe a general architecture of a centralised online voting system and detail which areas of such a system are vulnerable to electoral fraud. We then proceed to introduce the key ideas underlying blockchain technology as a decentralised mechanism that can address these problems. We discuss the advantages and weaknesses of the blockchain technology, the protocols the technology uses and what criteria a good blockchain protocol should satisfy (depending on the voting application). We argue that the decentralisation inherent in the blockchain technology could increase the public's trust in national elections, as well as eliminate voter impersonation and double voting. We conclude with a discussion regarding

*Department of Political Economy, email: `amrita.dhillon@kcl.ac.uk`
†Department of Political Economy, email: `grammateia.kotsialou@kcl.ac.uk`
‡Department of Informatics, email: `peter.mcburney@kcl.ac.uk`
§Department of Informatics, email: `luke.riley@kcl.ac.uk`

how economists and social scientists can collaborate with the blockchain community in a research agenda on the design of efficient blockchain protocols and new voting systems such as liquid democracy.

Contents

1 Introduction

Despite elections being critical to the democratic process, their integrity around the world is continuously questioned both by independent observers and the voters themselves. Major examples include the latest Election Integrity Project review [45] where independent researchers from Harvard and Sydney universities ranked only 19.5% of countries very high for election integrity. Similarly, in the 6th Round (2010-2014) of the World Values Survey [40] more than 25% of the individuals questioned (in about 76% of the countries surveyed) stated that they believe that election officials are often unfair (biased).

The issue of questionable election integrity can affect both developing and developed countries. In the developing world, the Honduras general election (26th November 2017) suffered from major irregularities at the vote counting stage, which led the Organization of American States (OAW) to recommend that the election should be rerun [43]. Elections in Albania continue to suffer from vote buying allegations [22]. In India the composition of the team of electoral officers can causally shift votes towards favoured parties with magnitudes large enough to change election outcomes, as shown by Neggers in [40]. In the developed world, examples include the 2014 mayor election in the Tower Hamlets constituency of London in UK that had to be rerun due to the discovery (after a court ordered investigation) of individuals voting multiple times and of votes casting from false addresses [41]. Another historical example from a country ranked highly on institutional independence, is the 1984 grand jury investigation into voter fraud in New York, USA. This investigation uncovered large scale and systematic fraud in the primaries of two of the borough's congressional districts between 1968 and 1982 (where 1000 to 2000 bogus registrants were discovered [35]). More recently, a US government study states that a weakness of the American system is that poll workers are not dependable or sufficiently trained [44].

There is as yet no consensus on how to measure voter fraud - presumably it is the most sophisticated fraud that is the most difficult to detect therefore relying on cases brought to the courts is an imperfect indicator. Klimek et al. [31] develop new methods from statistical physics to detect ballot stuffing and conclude that Duma and presidential elections in Russia in 2011 and 2012 suggest much ballot stuffing. Measures of voter fraud in US elections however suggest that at least double voting or voter impersonation is quite rare [3,26]. In the UK, there is no consensus over the degree of voter fraud. Besides direct measures of fraud however, there is the issue of "trust" in elections which can be eroded if there is even a small incidence of fraudulent voting, leading to lower turnouts of honest voters. Much of the academic literature assumes that election authorities are honest and assume that it is at the level of voting where there is any chance of fraud, not at higher levels. This assumption may of course not hold in many democracies.

Our analysis will be guided by a few desirable criteria for a voting system. *Accessibility* and *trust* in the voting system seem to be two minimal properties of a good voting system. However, the more accessible a system may be, the higher the risk for fraud can be. On the other hand, forcing voters to go through exhaustive security checks (to maintain trust in the system) can make voting less appealing and less accessible. Despite the conflicting nature of these two objectives, an election system must be able to balance the need for accessibility with the need to establish trust in order to provide a high level of election integrity. More specifically, election authorities must be able to show that eligible voters can easily register and vote, especially for countries with compulsory voting where accessibility is of even greater importance. But the public's trust levels can disturbingly decrease when election fraud incidents occur. Such incidents can arise at multiple levels during the whole voting process, even from collusion between officials (entrusted with authority to run the election) such as ballot box monitors or other election insiders.

To further explore this issue, we analyse and view a voting system as a sequence of four main processes, which we refer to throughout this paper:

- Voter registration
- Voter authentication
- Vote casting
- Vote counting

Note that each one of the mentioned sub-processes is vulnerable to some type of manipulation. Therefore, trust in a voting system (as a whole) implies that the possibility of manipulation should be minimised at each one of these steps. For example, the voting system needs to be able to show that no individual can be fraudulently added to the electoral roll (to achieve trust in the voter registration and authentication stages) while also showing that each vote has been accurately recorded and counted (to achieve trust in the vote casting and vote counting stages). But the more exhaustive the combined security checks for each stage are, the less accessible a voting system may become. For this reason, one of the main challenges of modern voting systems is to achieve a satisfactory level for both of these features (accessibility and trust) without compromising on one in favour of the other.

As additional desirable features of a voting system, we propose *speed* and *cost-efficiency* due to the following reasons. All paper ballot elections use an important amount of time and energy for the counting process, where an extreme example of this is Australia' s House of Representatives and Senate vote counting, which takes an average of two weeks [8]. Using the single transferable voting system [50], Australia compromises on the speed of the election results to achieve fairer results with respect to the proportional representation of citizens in the elected body. Lastly, organising and securing an entire election can incur a very large monetary cost to countries (especially to those running elections over multiple days). In India e.g. Electronic voting machines (EVM) were introduced in 1982 for the first time. An EVM takes about 3 hours to complete a vote count as opposed to paper ballots which could take 30-40 hours[1]. Therefore, to conclude, an election should cost as little as possible but without compromising on security, the speed to finalise the outcome or the fairness properties of the chosen voting mechanism.

In this paper, we argue first that electronic voting can improve accessibility, leading to some positive outcomes as shown by Fujiwara [23] for the case of Brazil. Fujiwara shows how the introduction of electronic voting in Brazil led to de facto enfranchisement (via greater accessibility) of less educated voters with a correspondingly more responsive government. It can also lead to faster counting as discussed above and can be cost efficient. Second, we document the various problems with centralised electronic voting systems and finally we show how the blockchain can potentially overcome these problems. We introduce the concept of distributed ledger technology (DLT) (blockchains are a special case of DLT) and how they can improve both the accessibility and trust properties of an online voting system.

This article is organised as follows. In Section 2, we focus on centralised online voting systems (i.e. that do not use distributed ledger technology), where we describe their general architecture and outline their vulnerable areas for manipulation. Section 3 describes from scratch the distributed ledger technology and how its promising features can be used for online voting. Section 4 focuses on a special case of distributed ledgers, called blockchains, and analyses the multiple ways (consensus protocols) on reaching agreement on voting data. In Section 5, we discuss a possible conceptualisation on using a blockchain based infrastructure for voting systems. More specifically, we analyse its potential for increasing trust in future voting systems, we present an illustration of how ballots can be submitted on such a system and describe possible challenges that

[1] `https://tinyurl.com/y6p4hhhr`

may require careful consideration during the development. In Section 6, we present existing blockchain based voting systems by categorising them according to the extend that they use this technology, concluding with details of a recent academic implementation. Finally, Section 7 concludes this work with open questions for economists and other social scientists in this area.

2 Centralised electronic voting

In this section, we discuss electronic voting and describe why the currently available (non-distributed ledger) electronic voting systems are considered to be centralised systems. Afterwards, we detail the areas of these systems that are vulnerable to manipulation or attack.

Electronic voting is the procedure of voting through the use of electronic devices and can be split into two main categories: *offline voting* and *online voting*. The first category is usually expressed by the use of electronic voting machines (which do not require an online connection), while the second category implies voting using devices connected to the Internet. Electronic Voting Machines (EVMs) can be positioned at polling stations in private booths to digitally record votes of citizens and, therefore, can replace the traditional paper ballot voting system. The main drawback of EVMs is that, unlike a paper ballot system where a voter can physically see the ballot that she casts, a voter has no guarantee that her vote was given to her preferred candidate. To solve this, EVMs can be extended to include: (i) Voter-Verifiable Paper Audit Trails (VVPAT), which gives the voters a non-digital way to verify that their vote was recorded correctly, and (ii) End-to-End verifiability (E2EV), which means that a voter has the capability to make sure that their vote has been properly cast, recorded and tallied in the election system [30]. For election authorities, EVMs can be beneficial for the speed of collection and counting of the citizens' votes. Adding perfectly working EVMs to an election would substantially reduce human errors, result in more accurate outcomes, prevent fraud in polling stations, and minimise staffing costs that occurs when manually performing the election [51]. Additionally, enormous amount of paper and energy is saved, which would otherwise be used on ballots printing, their transportation and storage. Since EVMs allow for the automation of vote counting, the final results can be announced faster than in a paper ballot election. Note that same day results in paper-based elections can occur only with large numbers of counting staff. In some cases, manual vote counting can take even more than a week to complete due to the complexity of the counting algorithm used. For instance, the

Australian national elections use the *single-transferable-vote* system for a large number of candidates (sometimes greater than 100), which may take up to a month to manually count.

Currently the majority of experts agree that EVMs can be built and operated in a satisfactory manner as long as they include VVPAT and a sufficient level of auditing (possibly with E2EV for the voters themselves) [25]. Despite ongoing issues with EVMs still in operation (which may not include VVPAT and/or E2EV), studies show that both in India [20] and Brazil [24] there was reduced electoral fraud and even better public good provision (due to reduced government spending for elections), when EVMs were introduced. We argue however that EVMs are still a centralised online voting system (i.e. controlled and operated by one entity) where there is scope for fraud by insiders as well as by hacking of the EVMs. We return to this issue later.

The use of the Internet has brought tremendous voting possibilities. Citizens can now vote digitally from any geographical location using a device connected to the Internet[1]. One of the main advantages of online voting is accessibility for everyone: people with special needs can be part of the voting community by casting votes from their own home, or people abroad can avoid the stress related to posting their vote (searching for nearby post offices, missing postal vote letters, etc.). Despite the tremendous convenience of using the Internet to vote, online voting for political office is difficult to implement due to the complex computer science issues that need to be resolved [25]. Such a system not only needs to maintain the integrity of the election, but it must do so in a way to convince the losing candidates that the winner is legitimate. At the same time, the system needs to maintain the secrecy of the voters' choices, to prevent voter coercion and minimise the possibility of vote buying. Indeed, the main complications arising from online voting have to do with the following conflicting objectives: how to maintain voter secrecy and at the same time to allow auditability of the votes without revealing voters' identities. In contrast, when the secrecy of transactions from the single organising (central) authority is not important, applications such as online banking already exist and flourish. Banks keep many transaction records and audit logs, some of which are available to the user. These records allow both the users and the banks to check for fraudulent activity. In voting, the more records an online system keeps on a voter, the more likely the voter secrecy will be compromised. Additionally, banks can use their transaction records and audit logs to undo the fraudulent transactions while any money lost can be written off as the cost of doing business. On the

[1] There are currently multiple areas around the world without Internet coverage, however this is an active research direction with promising technological advances, e.g. see https://x.company/loon/.

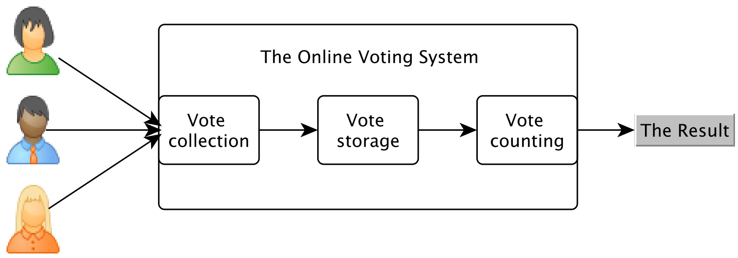


Figure 1: General architecture of an online voting system.

other hand, undoing votes and writing off hacks in an online voting system for a political office is a much more delicate issue.

2.1 General architecture of centralised online voting systems

An online voting system comprises of a series of steps which the user cannot physically observe or interact with due to its online centralised nature (the system is controlled and accessed only by a few people). To understand the main problems of online voting, we need to first understand the foundational infrastructure. We present in a simple form (Figure 1) the general architecture of an online voting system and discuss possible areas that can be manipulated by malicious entities. Observe that eligible voters submit their votes to a server through a single point of entrance. At this stage, there is a collection of all the votes and confirmation of their validity (we assume that voters have registered onto the electoral roll before the election has commenced, therefore there is a list of valid voters available). Then the collection of valid votes is stored in another part of the server until they are counted at a fixed time after the end of the election. The favourite candidate is the output result of the software system.

2.1.1 Areas of attack or manipulation in centralised systems

In this section, we highlight areas that are vulnerable to attack during the sequential process of online voting using a centralised system (that is controlled by one authority).

1. Virus on a voter's device: Online voting implies the usage of a device connected to the Internet so that the voter is able to connect with the online

voting server described previously. However, devices such as laptops or mobile phones can contain types of malware (digital viruses), which can delete or change a citizen's vote before it is even sent to the server. Note that this is also a weakness of decentralised systems as it is not actually about the system but about what data (valid or not) enter the system.

2. Denial of Service: All votes are received by the online voting server from a single point of entrance. A malicious entity can interfere with the election process at this stage by sending an excessive number of requests to the server. Overloading the server can cause large delays or even cease the collecting of new votes during the period of the attack.

3. Untrustworthy election authority: The programming code used to execute the different stages of an online voting system (collection, storage and counting of the votes) is usually kept private. Sometimes online voting providers open source (publish) their code however there is no guarantee that the published code matches the live version, which means that a malicious election authority can insert or remove code to manipulate votes without being noticed. An untrustworthy centralised authority can even use the voting data for profit by selling this data to third parties without the knowledge of the voters.

4. Invited auditors: Auditors may have been invited to survey the codebase. Some auditors will be more rigorous in their evaluation but still any audit process is not a guarantee that no problems will occur, especially if the central authority provide access to a different codebase. Additionally, during the counting stage, there may be a limited number of stakeholders/auditors present to observe the process. Even if they are technically knowledgeable, complex malicious acts would require a lot of time to identify. For these reasons, audits can only minimise the probability of identifying and solving a problem.

5. A single source of failure: If a centralised voting system goes offline (usually due to technical issues), no votes during this period will be registered. Even if this issue lasts only for a few minutes, it can cause confusion over the voting data (incorrect tally) and frustration for the citizens (lowering voter turnout). Furthermore, if there is an insufficient backup process, the voting data maybe fully or partially deleted (intentionally or otherwise), which massively jeopardises the integrity of any election outcome.

The more of these identified issues defended against, the higher the confidence in the voting system and, subsequently, the final outcome. But is there any way that efficiently deals with all of these types of issues? How can we minimise the probability of them occurring? In the next section we describe and discuss the potential of how distributed ledger technology can be used to tackle such issues when voting online.

3 Distributed ledger technology (DLT) as a decentralised solution

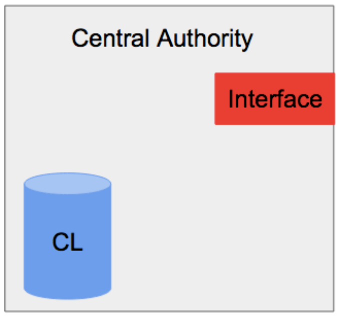
We start with defining what a ledger is. A ledger can be viewed as a file that continuously stores transactional data (assets that are assigned to users). For voting, the transactional data corresponds to the ballots that are assigned to their owners (each voter owns the ballot she cast). Therefore, a centralised voting system can be seen as a ledger (together with an interface) controlled by a centralised authority (see Figure 2a). Thus it exhibits the same vulnerabilities as the ones mentioned in the previous section. To solve these issues, we can modify this (centralised) technology by allowing multiple active stakeholders (which we call decentralised authorities). Each decentralised authority (or node) would have their own copy of the ledger and their own interface (see Figure 2b). But how do the multiple ledgers of each decentralised authority remain consistent over time and, which ledger do we choose in the worst case scenario where they become inconsistent? On top of this, since the voting system is now distributed over multiple authorities, how do different nodes of the network agree on the accurate voting data when there are issues such as failures on computer devices, network messages or malicious actors?

Network message failures

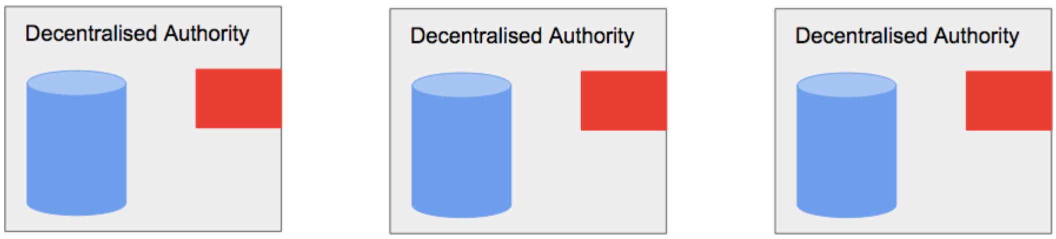
Computer failures

Malicious entities

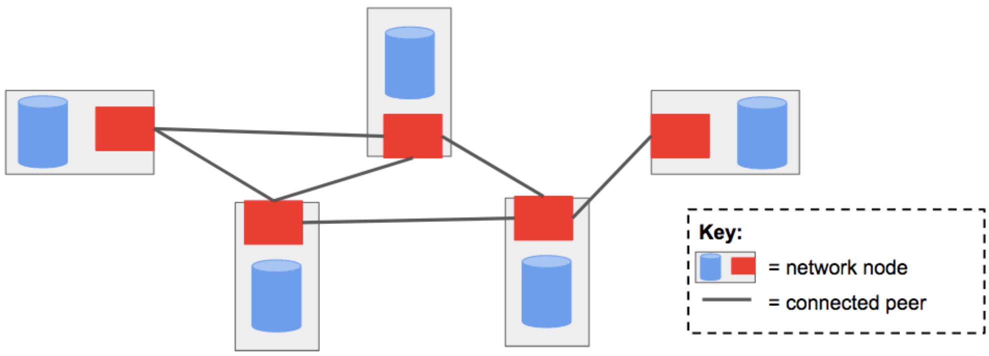
Fortunately, allowing multiple decentralised authorities having a copy of the ledger and a method to agree on which (voting) data is accurate under any scenario is mainly what distributed ledger technology allows, as we discuss in this section.



(*a*) A central authority controlling a centralised ledger (CL) and a generic interface.



(*b*) Each decentralised authority (or node) has its own copy of the ledger and its own interface.



(*c*) A distributed ledger system example

Figure 2: Moving from a centralised system to a decentralised system

3.1 What is a distributed ledger composed of?

A distributed ledger system (see Figure 2*c*) is a peer-to-peer network of nodes independent of each other, where each node is connected to a few others (not necessarily to all). Each node has a copy of the distributed ledger (the blue cylinder in our diagram) that stores the data, and an interface (the red rectangular in our diagram) for users or other nodes to connect. Additionally to the peer-to-peer network, a distributed ledger (DL) has a decentralised identity management, transactions and consensus protocols. We describe these components and how they are linked to each other as follows:

- DECENTRALISED IDENTITY MANAGEMENT: As mentioned previously, each piece of data in a distributed ledger (DL) is linked to an account (otherwise known as an address) to establish ownership. To transfer ownership of an asset (e.g. votes) from one account to another account, we use transactions (see next component). Each account is connected to and controlled by a public and private key (parameter) pair. We can view the public key as a more complicated representation of a DL account and the private key as an associated (to the public key) long password that cannot be changed. As the names suggest, a private key must not be shared with others than the primal key holder, whereas a public key can be shared with anyone. In this context, a private key and a message (e.g. a vote) can be given as input to a cryptographic signing (signature generation) algorithm that produces a signed message (e.g. a signed vote). To check if the owner of the key pair has indeed sent the signed message, we can perform the following process. Consider another user (e.g. a government officer) who uses a cryptographic signature verifying algorithm. Such an algorithm takes as input the signed message and a public key. If the algorithm returns *false*, it is implied that someone was trying to impersonate the owner of the key pair. If the algorithm returns *true*, the owner of the key pair has indeed sent this message (assuming that the private key has not been shared).
- TRANSACTIONS: Using a DL identity, a user is able to digitally sign a transaction to authorise actions on the network (e.g. casting a vote). All transactions must be from a sender account (e.g. voter) to a receiver account (e.g. a preferred candidate). A transaction may have a fee which is charged to the transaction sender and paid to the node of the network that adds this transaction to the distributed ledger. The voter signs the transaction (e.g. her vote) using her private key and the signed transaction is then published to the network as follows. All nodes verify a received transac-

tion and propagate it to others nodes on the DL network (if it passes the verification rules e.g. "is the signature correct?"). When a transaction is verified by a node, it enters that DL node's unconfirmed transaction pool and remains there until the transaction is added to the distributed ledger in a manner according to the consensus protocol (see next component).

- CONSENSUS PROTOCOLS: These protocols are sets of rules for the nodes to agree on what, how and when transactions are added to the distributed ledger (see more details in Section 4.2).

3.1.1 Nodes of a DL voting system

DL nodes may differ even in the same distributed ledger, e.g. nodes with or without full permissions. The following includes the main parts of a node with full permissions:

- *Distributed ledger*: it contains the node's copy of the DL, the blocks (if this DL is a blockchain - see Figure 3) and the confirmed transactions.
- *Current network state*: the up to date voting data of the DL, e.g. What is each voter's most recent ballot? What is each candidate's current vote balance? What is the associated smart contract code of the current voting system?
- *Wallet*: private keys for the voting accounts that this node controls and a mapping to the corresponding public key/address.
- *Unconfirmed transaction pool*: the transactions (vote submissions) that this node receives (propagated from other nodes or sent from a user directly to this node) and are yet to be confirmed in the distributed ledger section of the node.
- *Network routing aspects*: it describes how this node receives transactions and blocks and how it connects to the other nodes.
- *Consensus protocols*: rules for when and how the node knows that transactions (and/or blocks if the DL is a blockchain) are verified and confirmed (discussed in Section 4.2).

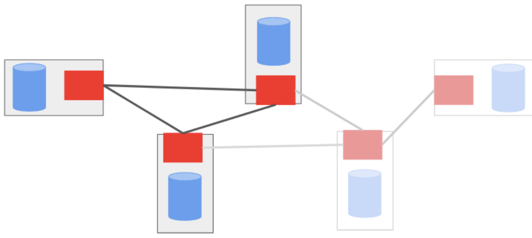
Note that a node can provide an external interface into one or more of these aspects. For example, the network routing aspects require an external interface to allow incoming connections, e.g. from other nodes joining the network. A node may also provide an external interface for users to interact with any other section, e.g. to read details on the current state of the distributed ledger.

3.2 What a distributed ledger can offer to voting

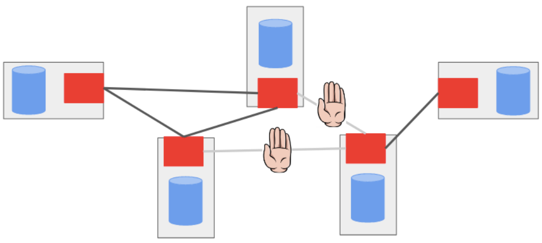
The main features that distributed ledgers can offer to online voting are as follows:

1. NO SINGLE SOURCE OF FAILURE
   If a few nodes get compromised, then the network is resistant to spreading that failure to the whole network, which would otherwise result in the compromise of the online voting system. More specifically, distributed ledger networks can tolerate crash, partition and Byzantine faults which we explain below.

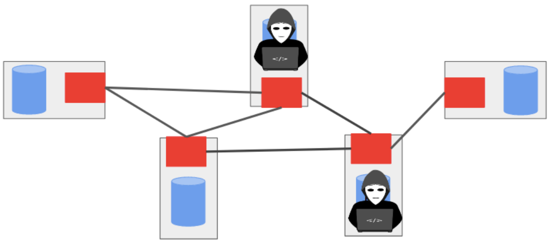
   - Crash Fault Tolerance (CFT): If some nodes crash or go offline (as the 2 greyed out nodes are in the given example), then the rest of the nodes can carry on operating as normal and continue adding (voting) data to the distributed ledger.

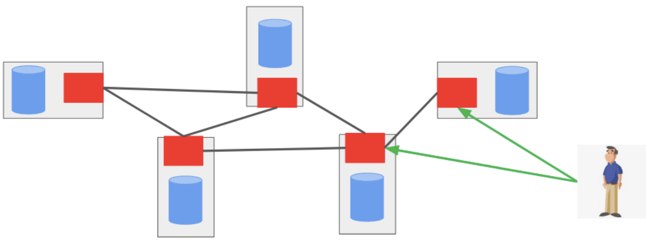
   - Partition Tolerance (PT): If there is a partition in the network (the connections greyed out with hands on them), then the network is temporarily separated into two sub-networks (with three nodes and two nodes in our example respectively). When the lost connections are re-established, the separated sub-networks synchronise their ledgers to create one overall distributed ledger.

   - Byzantine Fault Tolerance (BFT): A distributed ledger network can handle Byzantine (malicious) entities hacking or controlling multiple nodes, without compromising on the validity of data stored in the distributed ledger. However, note that this is true conditional on the event that these malicious entities compromise up to a certain percentage threshold of the network nodes (up to 50% is a common threshold).

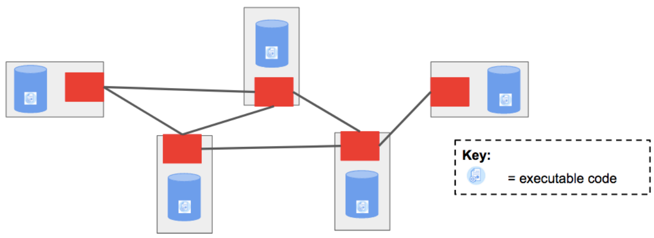


2. MULTIPLE ACCESS POINTS
   Every decentralised authority running a node in a DL voting system knows that the voting data in their node is correct (due to the properties of the technology). However, the average voter most likely would not be running a node (either because they do not have permission, or due to the cost and/or technical proficiency required to run one). Since this is the most likely scenario for the average voter, it is an advantage that distributed ledgers allow voters to access the system from multiple different access points (by connecting to the interfaces of different nodes). Checking with multiple nodes is advantageous as the voter can spot if it is interacting with a malicious or faulty node (providing the voter with incorrect data). A voter can spot such activity by comparing the requested data returned from one node, e.g. information on her voting receipt, to the same data returned from another node(s). Note that when a voter sends a data request to a node's interface of a centralised voting system, there is no guarantee where the response comes from (the original ledger or a manipulated version of it). With multiple access points, a voter can resend her data request to as many of the nodes in the network as she wants (the higher number of nodes returning the same result, then the higher confidence the voter can have in the returned piece of data being the actual data stored on the distributed ledger).



3. Users own their voting data
   There exist voting applications where each vote needs to be associated with the voter's ID, e.g. the voter's name. Examples of such applications include voting within a parliament, proxy voting and electronic petitions. The current structure of a centralised voting platform can allow interested third parties to have access and use this data (usually for targeted advertisement to special groups of citizens). For instance, if someone uses a centralised voting platform to vote on a petition in favour of stopping the general use of chemical pesticides, this platform could sell this voter's data (without her permission) to an organic products company. On the other hand, if the voting data is stored in a distributed ledger voting system, then a data ownership concept can be embedded as follows. Each piece of data has its own location which is controlled only by its owner. The owner determines who has access to the data in this location, the length of the access period and whether this access requires a fee or not. In this way, data commercialisation goes first through the hands of the data owners (voters) by getting their permission for any use of their own data.

4. Smart Contracts
   A final advantage of a distributed ledger voting system is that the distributed ledger can also contain automatically executing code, called smart contracts. Smart contracts trigger computational functions when certain data has been added to the ledger. For example, if a voter submits a vote, an embedded smart contract could automatically update the vote count at every decentralised authority (node), so that all nodes can keep track of the winning candidate.



3.3 Distributed Ledgers vs Distributed Databases

One of the most common questions regarding distributed ledgers is about their difference to distributed databases (see Figure 3). A distributed database (e.g.

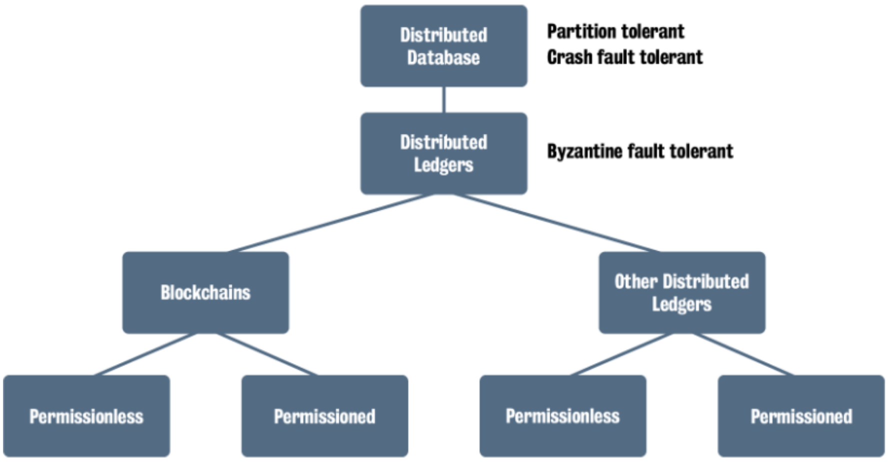


Figure 3: A categorisation of distributed ledgers

Google's BigTable used in applications such as Google Earth) stores all types of data while a distributed ledger stores data as transactions. In a distributed ledger, all assets are automatically assigned to a user/account but this is not the case with a distributed database. Since distributed ledgers assign assets to users, they should not tolerate malicious behaviour in the network, so the byzantine fault tolerance (BFT) property plays a key role characterising this special case of distributed databases. On the other hand, distributed database nodes are assumed to be all under the control of a single entity, thus they only enforce the crash and partition fault tolerance properties (CFT and PFT, respectively). To conclude,

$$\text{Distributed ledger} \Rightarrow \text{Distributed database},$$

which implies that there are cases where

$$\text{Distributed ledger} \not\Leftarrow \text{Distributed database}$$

as distributed databases do not necessarily satisfy BFT.

3.4 Permissioned & Permissionless Distributed Ledgers

In this section, we discuss how voting systems with a distributed ledger (DL) network infrastructure can have different levels of openness. The choice of which type of DL network to use should depend on the voting application.

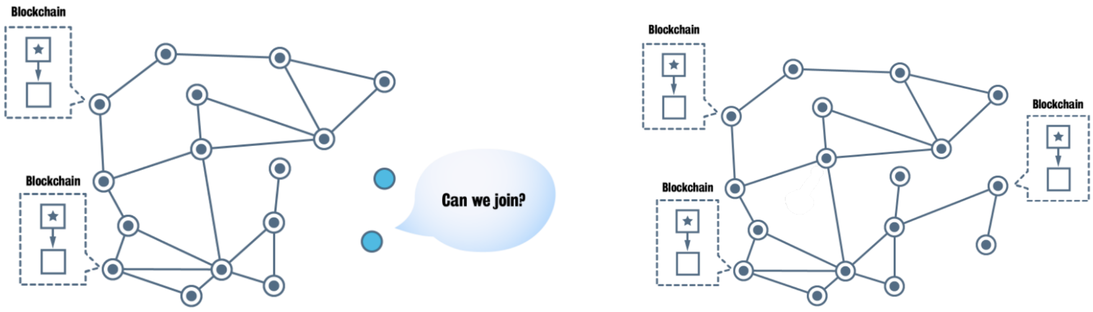


Figure 4: In a permissionless distributed ledger network, anyone interested in becoming a node can join the network.

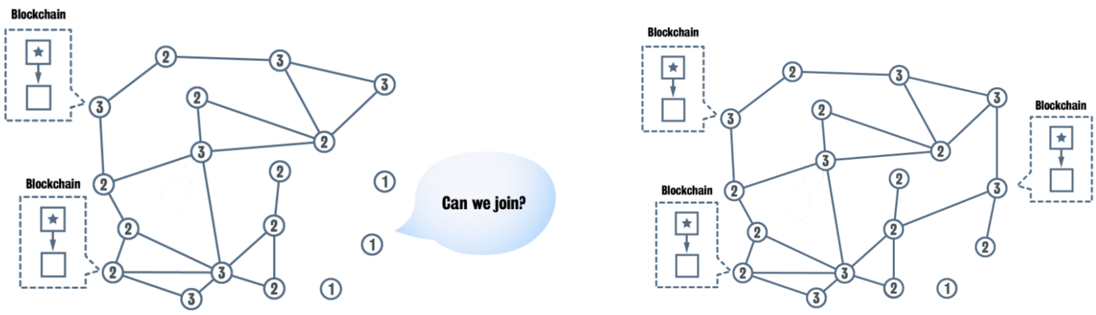


Figure 5: A permissioned DL network where circles numbered 1 represent users out of DL network interested to join, circles numbered 2 represent DL nodes with full permissions and circles numbered 3 are DL nodes with restricted permissions.

In a permissionless DL network, anyone interested in becoming a node of the DL network can do this by downloading the node software from any freely available source. Figure 4 shows an example of a permissionless DL network, where the DL nodes are represented as grey circles[1], while the two blue circles represent two new potential DL nodes interested in joining. As can be seen in the example, both users represented by blue circles are allowed to become apart of the network and therefore read and write to the shared distributed ledger. In permissioned distributed ledgers, by definition, potential new nodes need to be

[1]The dashed rectangular bubble shows that different DL nodes share the same ledger (in the form of blocks connected together) and each DL node has the same topmost block of data.

given permission to join the DL network. Without such permission, every external stakeholder of the network relies on a single or a group of nodes to get data stored on the distributed ledger. To become a DL node of a permissioned network, access needs to be granted by a one or more nodes of the current DL network (depending on the governance framework). Even when access is granted, nodes of the same DL network may have different permissions, for example (i) DL read-only permissions, (ii) DL read and write permissions or (iii) DL read, write and governance permissions. Figure 5 shows an example where two of the three users requesting to join have been allowed to join the network, each with different permissions.

Permissioned DL networks are suited for situations where there is a desire to decentralise the control and management of the voting process up to a certain degree but not completely, i.e. moving from a single central authority model to a consortium. The nodes of the permissioned DL network could be run by different constituencies or different candidates. In this way there can be more transparency of the online voting system between a wider variety of stakeholders than the centralised online voting system model. But note that full transparency for every member of the public will not be provided by permissioned DL networks (as the average voter will not have permission to run a node - and thus have full access to the ledger data - in this model). If the goal is to achieve a publicly open voting system with maximum possible decentralisation, then a permissionless distributed ledger is more appropriate. This model allows for full transparency and auditability benefits as any individual in the electorate can access the voting data and possibly the associated algorithms implemented as smart contracts (detailing for example, the cryptography used in the system and the counting method). A complication of this model is that it is more difficult - but not impossible - to keep all sensitive data private.

4 Blockchains - a special case of a distributed ledger

In this section, we discuss how blockchains (the most popular type of a distributed ledger) order and confirm data with respect to voting applications. Before we go into a more detailed analysis, we first give an overview of the features that distinguise blockchains from distributed ledgers. A blockchain contains groups of confirmed and valid data (blocks) that are sequentially linked to each other using cryptography, so that these data blocks eventually form an ordered chain [36, 39]. The idea of a blockchain is that data is recorded into blocks in an

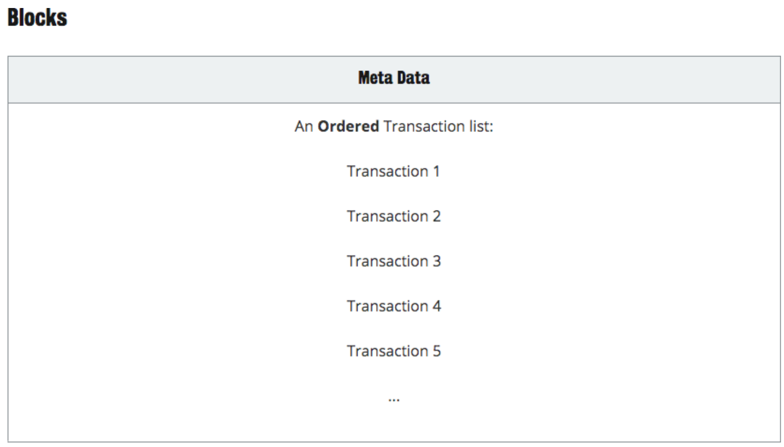


Figure 6: Blocks in a distributed ledger voting system contain a list of transactions in the form of votes being assigned to candidates. Additionally, blocks contain some meta data that includes the cryptography to link these blocks together.

"append only" manner. Once data is added onto the blockchain, everyone with a copy of this blockchain assumes the data stored on it is true as it is extremely difficult to modify when compared to storing data using a different method (e.g. central or distributed database). Blockchains, like distributed ledgers are created for a distributed network of nodes (computers) with no single authority in charge and possibly malicious entities in the network. Each node in the network contains a copy of the blockchain and the rules on how the nodes reach an agreement on which data is saved in it (which we describe in Section 4.2.3).

4.1 Blocks

Recall (from Section 3) that a ledger can be seen as a file that continuously stores transactional data (assets assigned to users) and that, for voting applications, the transactional data corresponds to the ballots assigned to their owners (each voter owns the ballot she cast). All transactions that appear in a (data) block (see Figure 6) are assumed to have passed protocol verification checks, for example “Is the digital signature for this user correct?" and “Are all required transaction fields completed?". The transactions inside a block are in an ordered list and

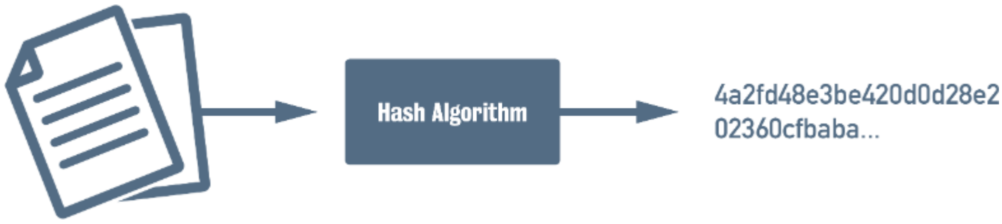


Figure 7: A digital object converted into a string of characters (hash output).

an individual block is of a small size[1]. The ordered transaction list implies that all nodes execute transactions in the same sequence and that transaction execution is deterministic (when they are grouped into a block). But, where do transaction orders come from?

4.1.1 Transaction selection

When a user/voter creates a transaction, it does not immediately go into a block. The transaction is firstly checked for validity at the receiving node. If it passes this check, the transaction is placed into the node's unconfirmed transaction pool and a copy of it is propagated to other nodes of the DL network, who will also check the transaction validity and, when passing, place the transaction into their unconfirmed transaction pool as well as continue to propagate it around the network. When creating a block, transactions are selected from the unconfirmed transaction pool in any order. It is the node(s) nominated as the block creator (sometimes known as block producer or miner) who decides on the transaction order in a block. Block creators can select any transactions they want from the unconfirmed transaction pool. If there are a balanced variety of nodes allowed to be block creators then this should allow all types of valid transactions to eventually become part of some block.

4.1.2 Chain

To understand how blocks are chained together, we briefly describe hashing. Hashing converts a digital object of arbitrary length (e.g., a document, an im-

[1]Bitcoin allows a maximum size of 1MB for blocks (https://bitinfocharts.com/comparison/bitcoin-size.html). Therefore, blockchains are not ideal for storing large data files (eg videos). However, large files could be referenced on the blockchain by storing it somewhere else (perhaps a distributed database) and then adding an identification marker to a transaction of a distributed ledger.

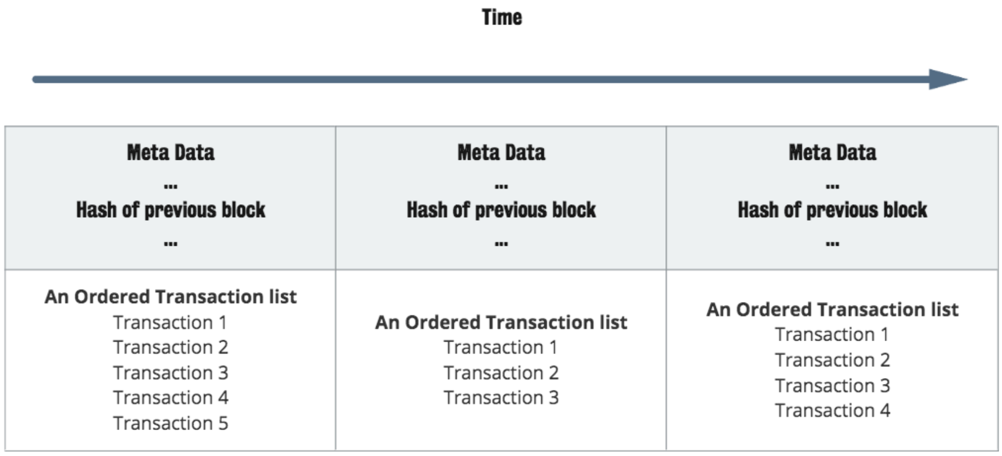


Figure 8: How hashes are used in a blockchain to interlink multiple blocks. Note that each block has an ordered list of transactions from 1 to n.

age,...) into a single character string (see Figure 7), which we call the hash output. Hashing algorithms are deterministic (hashing the same digital object always gives the same hash output) and very hard to reverse-engineer as hashing similar digital objects results in very different hash outputs - thus they are considered as a form of encryption. To show that it is very difficult to work out the input of a hash function by looking at similar hash outputs, see the following example which hashes very similar character strings but produces very different hash outputs:

I am Satoshi Nakamoto0 ⇒ a80a81401765c8eddee25df36728d732...
I am Satoshi Nakamoto1 ⇒ f7bc9a6304a4647bb41241a677b5345f...
I am Satoshi Nakamoto2 ⇒ ea758a8134b115298a1583ffb80ae629...
I am Satoshi Nakamoto3 ⇒ bfa9779618ff072c903d773de30c99bd...
I am Satoshi Nakamoto4 ⇒ bce8564de9a83c18c31944a66bde992f...
I am Satoshi Nakamoto5 ⇒ eb362c3cf3479be0a97a20163589038e...
I am Satoshi Nakamoto6 ⇒ 4a2fd48e3be420d0d28e202360cfbaba...

4.1.3 Hashes chain blocks together

To see how hash outputs are used to chain blocks together, we use the example illustrated in Figure 8. Each block has some metadata and a list of ordered transactions that the block creator has selected. Note that blocks do not have

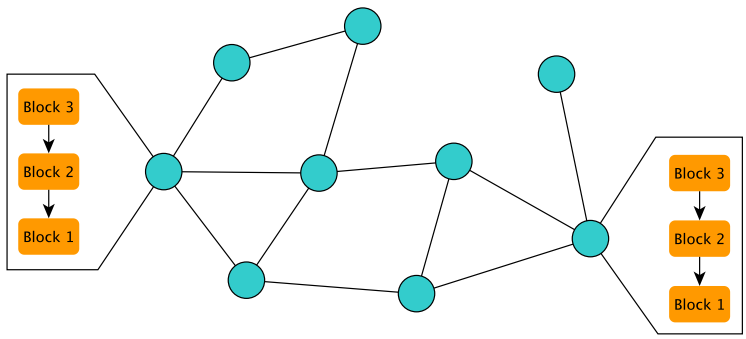


Figure 9: Every node in the network has a copy of the entire blockchain. In this example, the blockchain consists of three blocks.

to have the same number of transactions. Each block contains a hash link to the previous block. The hash of the previous block is generated by inputting the data of the previous block into a pre-agreed hashing algorithm, selected as part of the distributed ledger's data agreement rules. When multiple blocks are linked together via hashes in this way, we get a chain of blocks (known as blockchain). A copy of this chain is included in every node.

4.1.4 Hashes make blocks difficult to modify

How difficult is it to change a transaction recorded in the block of the chain? Using an example, we show that it is very difficult. Lets assume that we want to change Transaction 2 of Block 1 (to the left of Figure 8). To do this, we would have to correctly modify the hashes of all the blocks following Block 1 (i.e. Block 2 (middle) and Block 3 (to the right)). Recall that even a minor change to the input data of the hash algorithm can result in a large change to the resulting hash output (string of characters). If a user was maliciously trying to change the transaction (a specific vote in our case), then he would have to perform this change on every node of the DL network at the same time without anyone spotting all these changes, which is an exceptionally hard task. Therefore, voting data stored in a blockchain is significantly harder to modify (for a network of a reasonable size) compared to a centralised system.

4.1.5 What is the current state of the network?

Given a large number of transactions and blocks, how can we understand what is the current state of the network, i.e. what is the most up to date data? To clarify this, we process the transactions in order. Then the current network state is made up of the last valid transaction about each unique piece of voting data. For example, if users were allowed to re-cast their ballot and there were multiple transactions from the same user on the same election recorded into the blockchain, only the most recent transaction is prioritised and recorded in the overall current state.

4.2 Reaching consensus on data

How do nodes agree on what data to store in the distributed ledger? How can we trust the voting data stored in the distributed ledger? These questions around trust issues are answered by using distributed ledger consensus protocols that are Byzantine fault tolerant. For example, consider a set of computers that want to reach an agreement about a ballot received. The set of computers need to know the answer to the following question: Did voter X vote for candidate Bob? Or did voter X vote for candidate Eva? Using a consensus protocol, the set of computers can then come to one agreed decision, such as: voter X voted for candidate Eva.

4.2.1 What is a consensus protocol?

The set of rules on how the users of a blockchain can reach agreement on what data is in the blockchain are known as *consensus protocols* [6]. A consensus protocol consists of two key parts, an *anti-sybil control mechanism* and a *data agreement rule*.

Anti-Sybil mechanisms: The anti-Sybil mechanism gives no advantage to users who create multiple identities. The main proposed anti-Sybil mechanisms for distributed ledgers are the following.

- Proof of work
- Proof of stake
- Delegated proof of stake
- Proof of authority

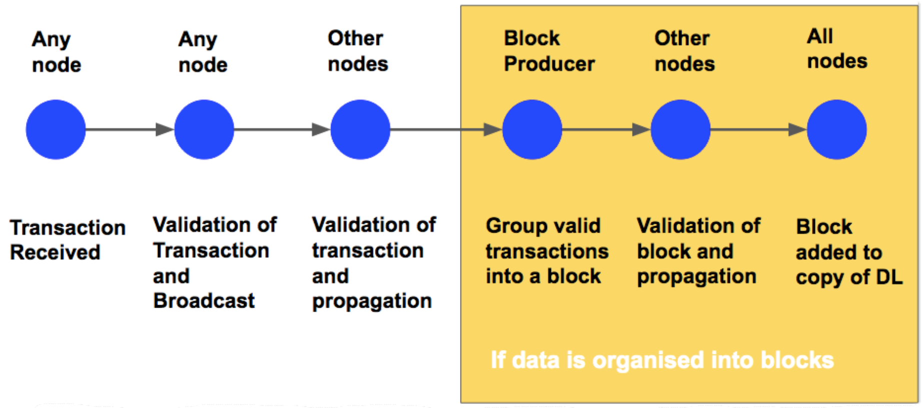


Figure 10: The general steps for every consensus protocol for blockchains, where the first three steps hold for any distributed ledger. Note the actions (on the bottom) and their participants (on the top).

Data agreement rules: The consensus protocol data agreement rules detail exactly how the DL nodes should operate. Examples of such rules include

- How exactly to decide who produces the next block
- How to make a valid block
- How to verify a transaction
- How to communicate between nodes

DL consensus protocols are commonly referred to by the name of their anti-Sybil mechanism, even though the actual data agreement rules may differ. For example, both Bitcoin and Bitcoin Cash blockchains use a proof-of-work consensus protocol but they differ in their data agreement rules[1]. Figure 10 shows the generic steps of any consensus protocol of blockchains.

4.2.2 Why follow a consensus protocol?

Regardless of the consensus protocol, if a node successfully produces the next valid collection of transactions on a permissionless DL then it will receive a reward in cryptocurrency (that it can exchange for another currency). This reward

[1]As bodes of the Bitcoin Cash network are allowed to create larger blocks of data.

Permissionless DL	Permissioned DL
Cryptocurrency reward 	Off-DLT incentives (e.g. reputation & legal responsibilities) 

Figure 11: Incentives to follow the data agreement rules of the consensus protocol.

comes from either (i) the generation of new cryptocurrency coins and/or (ii) collecting transaction fees from the verified transactions[1]. On the other hand, if a node successfully produces the next valid collection of transactions on a permissioned DL, it could also receive a cryptocurrency reward, however, the most common reason to follow the consensus protocol in a permissioned distributed ledger is due to the off-chain incentives, such as reputation or legal responsibility (see Figure 11). For example, if each node of a permissioned DL voting system (for national elections) represent a political party, then it would be in each party's best interest to correctly announce incoming (vote data) transactions to keep the citizens' trust and their own reputation level high.

4.2.3 Multiple ways to reach consensus on data

There are many consensus protocols available and each distributed ledger implements one. Each consensus protocol must specify how block creators are chosen. In this section, we focus on one of the two main components of a consensus protocols (see Section 4.2.1), the sybil control mechamisms. We describe main features of various such mechanisms and discuss how they can be combined with a data agreement protocol as well as their potential applicability to voting.

- PROOF OF WORK (POW): This is the first blockchain sybil control mechanism as it is used by Bitcoin [39], the first blockchain implementation.

[1]Transaction fees, are usually only charged for permissionless distributed ledgers and differ depending on the distributed ledger used and how congested the network is.

The idea behind this sybil control mechanism is that nodes race to solve a complex computational problem, where the first node to solve it is allowed to publish a block of data to be added to the blockchain. The likelihood of any node solving the computational problem first is proportional to the computational power that this node has access to (when compared with the total computational power of the network). This means that creating multiple nodes gives a person no advantage as computational power cannot be used in parallel by multiple nodes.

The Proof of Work data agreement protocol, consists of details on the exact computational puzzle to be solved[1], how to handle multiple blocks being released at the same time[2], and rules to identify what are valid blocks and transactions, e.g.: each block must be of a certain maximum size; each sender of cryptocurrency must have the cryptocurrency in the first place; a sender of cryptocurrency cannot attempt to send the same coins to multiple different places; etc.

In Proof of Work, anyone can become a block creator if they win the race to be the first to solve complex computational Proof of Work puzzle. Nodes are incentivised to attempt solving this complex computational problem by receiving some amount of the native cryptocurrency of the blockchain if they win the race. But every node who loses this race, essentially has no gain from all of the energy they have wasted. Budish [10] shows that con-

[1] The computational problem to solve in Bitcoin is cryptographic in nature and is a follows. A node runs a pre-agreed hash algorithm, that takes as input the data of the block the node wants to create, and attempts to find a hash output where the first x digits are zeros. This hash output links this block with the previous block in the chain. The connection is achieved as one of the input of the hash algorithm is the hash output of the previous block added in the chain. The following shows the main inputs in the hashing algorithm, where the nonce (the only non-fixed input that a node can adjust to get the desirable hash output) is an integer number chosen by the node running the hash algorithm.

HASH ALGORITHM INPUT
· Hask key of previous block · Timestamp · Transactions · Nonce ⋮

$$\Longrightarrow \quad \underbrace{00\ldots0}_{x}\text{1ac39bf2}$$

[2] Bitcoin uses the longest chain rule. If multiple blocks have been released at the same time, each node should continue to attempt to build on the blockchain it believes is the longest (or randomly select a chain if it thinks there are multiple chains of equal size). If at any point a node receives a longer chain than the one it is currently building on, the node should switch to the new longer chain.

ceptually an equilibrium requires a zero profit condition to be fulfilled: nodes enter and attempt to create the next block until there are no profits to be made, as well as an incentive compatibility condition which ensures that no one wants to carry out a majority attack. The latter implies that the cost (in terms of computing power) of doing so must be higher than the gains. Budish shows that these two conditions imply that the equilibrium per block payment for block creation must be large relative to the one shot benefits of an attack. This constraint puts an economic limit on the applicability of this consensus protocol.

- PROOF OF STAKE (POS): In this Sybil control mechanism, the likelihood of a node being assigned to create the block is random and proportional to their cryptocurrency holdings (compared with the total cryptocurrency stake in the network). This means that creating multiple nodes gives a person no advantage as their cryptocurrency holdings would have to be split between the accounts associated with their different nodes.

  For Proof of Stake data agreement protocols, there does not have to be a complex computational problem for the nodes to solve and so they can be significantly more energy efficient than Proof of Work versions. A possible data agreement protocol for Proof of Stake can be a lottery among nodes where the winner's prize is to produce the next block and each node has as many lottery tickets as it has cryptocurrency coins. Thus nodes with large stake have a higher chance to produce the next block. More complex Proof of Stake data agreement protocols are Tendermint [4,9], Casper [11], Oroborous [5, 19, 29] and Avalanche [47].

  As in Proof of Work, nodes are incentivised to have a stake in the system because, should they create a block, they will be rewarded with some amount of the native cryptocurrency of the blockchain. Unlike Proof of Work, if a node is not chosen to create a block, they do not waste energy, they only have to consider the opportunity cost of having their capital locked into this distributed ledger. This implies in particular that the problem pointed out by Budish [10] could be overcome with a different design.

- DELEGATED PROOF OF STAKE (DPOS): In this Sybil control mechanism, the node assigned to create the next block depends on the amount of votes received from other nodes with cryptocurrency holdings, where new elections can occur every few minutes. Note that this Delegated Proof of Stake election is separate from any further voting system built on top of the distributed ledger technology. In a Delegated Proof of Stake election there is no advantage to creating multiple nodes as firstly one cryptocurrency coin corresponds to allowed ballot, and secondly creating two candidate accounts will split the electorates' votes between these two candidates.

Possible data agreement protocols for Delegated Proof of Stake could be a round robin format between the k nodes with the highest vote in the previous election, where each of the elected nodes take turns producing the next block (in a way that is not computationally difficult to decide and order) and the other $k-1$ elected nodes decide via another binary vote (with 'accept' or 'reject' options) whether the newly produced block is valid or not. If a pre-agreed threshold number of 'accept' votes is reached, the block is added to the chain. Once the election cycle has been completed (after each block creating node has produced a pre-agreed number of blocks), then the next set of block producing nodes are elected and this process starts again. Additionally, the k elected nodes could reuse a Proof of Stake data agreement protocol where the only participants in this election cycle are themselves.

- PROOF OF AUTHORITY (POA): In this Sybil control mechanism, a node can create a block only if it contained an identified (approved) account, where it can only be approved by the current nodes with authority in the blockchain. This approval will be linked to a real life identification, meaning creating multiple accounts will not be possible. Possible Proof of Authority data agreement protocols can then follow a simple path to the round robin or lottery formats described previously. However only nodes with certain permissions can vote to decide if a new node can become a block producer or if a current node loses its block producing status.

  Note that this method of consensus is suitable for a permissioned network as the trust is centralised to one node or a group of nodes (as opposed to the others where trust is distributed). Such a consensus protocol should only be used when there is a high level of trust in each block creating node and each node responsible for the governance of the system. However, one of this mechanism's drawback is that by adding more nodes, the performance of the network can dramatically slow down. This is due to the large amount of messages that the nodes exchange between them if they vote on whether they accept a new block or not [9]. A block is considered valid (accepted) only when a specific number of nodes (usually two thirds plus one) vote in favour of the block. Note that this is in contrast to Proof of Work, where a block is considered valid until its validity is challenged.

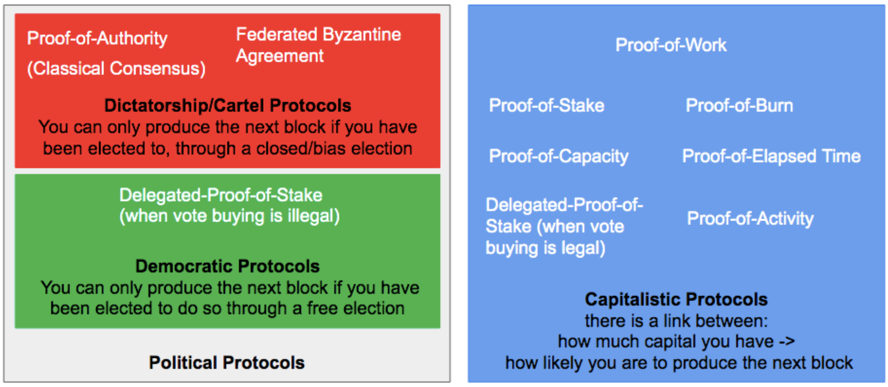


Figure 12: A consensus protocol family categorisation

We conclude this section by presenting the main criteria for a desirable consensus protocol for voting applications:

(1) contains an effective anti-sybil mechanism

(2) allows sufficiently many nodes to disincentivise a malicious attack (including collusive attacks),

(3) minimises computing power (to avoid unnecessary energy wastage) and

(4) rewards the block producing nodes (if a permissionless DL system is used)

At this point, we can observe that there is a strong parallel between mechanism design and designing the consensus protocol. The objective is to minimise the cost of running the system (including rewarding the block creating nodes) subject to incentive compatibility (including for coalitions of stakeholders) and individual rationality conditions. In contrast, for permissioned DL networks, the main question is how to choose the approved nodes, e.g. how to ensure representation of all political parties in an election. Last, we present a categorisation (see Figure 12) of consensus protocols with respect to what type of process they follow in order to choose the next block creator. We divide the consensus protocols into capitalistic and political ones, with the latter category being further divided to dictatorship and democratic style protocols.

5 Using blockchain technology in voting: a possible conceptualization

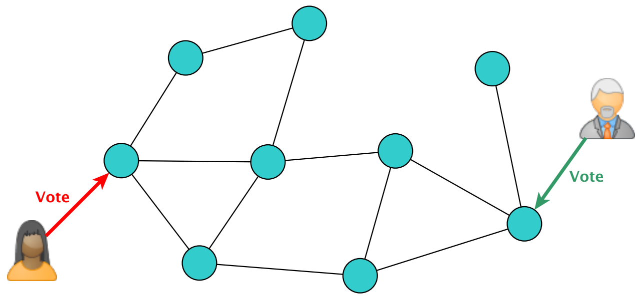
A blockchain based voting system has not been implemented yet in a large election trial. In this section, we give a high level illustration of how the principles of blockchain technology can apply to the special case of voting. The simplest possible design of such a system can be as follows. First, a voter needs to connect with one of the nodes of the blockchain network to submit her vote as a transaction (see Figure 13 (*a*)). Upon receiving the user's vote transaction, the chosen node forwards the transaction to the rest of the blockchain network nodes (see Figure 13 (*b*)) where it joins each node's unconfirmed (vote) transaction pool. The next block to be added in the chain will include a subset of the unrecorded vote transactions. Every new block (with newly confirmed votes) that is created is sent to every other node of the network so that all nodes update their copy of the blockchain (see Figure 13 (*c*)), allowing them to agree on what valid votes have been recorded.

Connecting to a node is possible in two different ways: (*i*) If a voter knows the location of a particular node, then she can directly connect to that node to cast her vote. This node could be on her PC (if the voter has permission to run a DL node) or on some server[1]. (*ii*) A voter can sign into a website interface that connects to the blockchain network. Since blockchain node networks are distributed, there could be multiple different interfaces connecting into the same blockchain network (via different nodes), therefore there would be multiple access points for the citizens to vote.

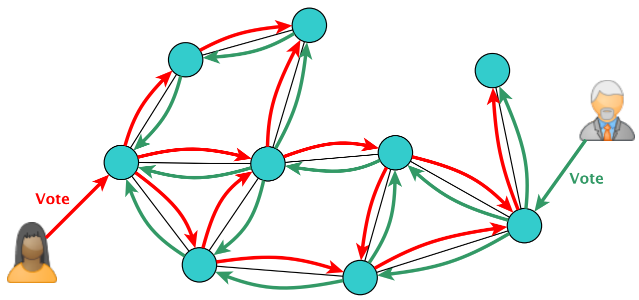
For the rest of this section, we discuss the main advantages that a blockchain infrastructure can offer to online voting when compared to a centralised system and suggest possible extensions to the simple conceptualisation described up to this point. Subsequently, we discuss the main challenges we have observed that such a technology would face before becoming accepted.

In Section 2.1.1, we see that one of the main problems with centralised online voting is the vulnerability of the voting data to malicious attacks (even from the central election authority). In contrast, voting data stored on a blockchain is very difficult to censor, due to the distributed network of nodes (that may be run by many different stakeholders of the system) and the way these nodes communicate between them to reach agreement on data (the consensus protocols). For instance, if a user sends a vote transaction to a node, and the user

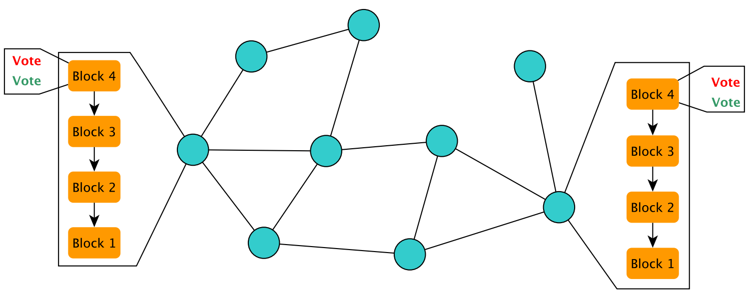
[1] This connection method is only recommended for users with advanced technical skills.



(*a*) Users can submit their vote from multiple access points.



(*b*) This data is being distributed in the whole network.



(*c*) If voting data is valid, it is added in the blockchain.

Figure 13: An illustration of vote submission to a blockchain-based online voting system.

discovers[1] that this vote is subsequently not recorded into the blockchain (i.e. the vote transaction has been lost or censored), then the user can send the vote transaction to another node of the network until the vote is recorded. Note that this re-sending process can be performed automatically (by applications) to avoid overburdening the user.

Unlike a centralised system, once a vote transaction is recorded in a block, it is significantly difficult to modify it. To do this, a malicious node would have to accurately edit not only the vote transaction and the meta data associated to the vote transaction's block but also every link between successive blocks in the chain. The malicious node would have to perform this procedure on every node of the network at the same time to avoid detection, which is more difficult as more nodes join the blockchain network. This is in line with our earlier observation that electoral fraud is prevented in the blockchain by increasing the difficulty of changing a single vote transaction. This feature also implies that the cost of changing multiple votes is increasing and convex, which solves the problem 3 of Section 2.1.1 (assuming that there are sufficiently many nodes).

Compared to standard centralised voting systems (where there is a single authority monitoring actions), a blockchain based voting system allows some level of decentralisation on the monitoring and implementation of the system's procedures. Examples where multiple nodes can have a role include: (i) the collection of votes; (ii) the validation of votes; and (iii) the counting of votes. Of course, as mentioned before, the composition of the stakeholders who have the nodes plays a key role to a transparent voting system. We suggest stakeholders with conflicting interests, such as different parties, to be running nodes to disincentivise collusive attacks (see discussion for challenge 3 in Section 5.1). This feature can also help with problem 3, discussed in Section 2.1.1 when the election authority is not independent from the executive. Furthermore, problem 4 of 2.1.1 can be dealt with blockchain technology since there can be multiple auditors with conflicting interests who can check on the validity of incoming data, which itself cannot be modified by any single node. Recall that the main difference between permissionless and permissioned blockchains is that in a permissionless blockchains: anyone can run a node on the network to read the blockchain data and create new blocks, and anyone can send and receive transactions recorded in the blocks. In contrast, in a permissioned blockchain, both running a node and being able to send a transaction require special per-

[1]Owners of nodes can provide a function (application) for the general voter to be able to read (restricted or all) the voting data recorded on the blockchain. In this way, a voter can discover whether her vote is correctly recorded or not. In contrast to this, in a centralised system there is no guarantee that the data a voter reads using such a function (provided by the single authority) are valid.

missions. By their nature, permissionless blockchains are characterised by a cryptocurrency reward system for nodes that create new blocks. A permissionless blockchain is not ideal for elections where voting data privacy is desirable and election management locations need to remain within specific geographical borders e.g. national elections of a country. However, permissionless blockchains could be well suited for voting on matters with a global interest. Examples can include petitions on matters where any citizen of the world has an interest, e.g. air pollution, global warming, people's rights, etc.

Even if we identify voting applications well suited for using permissionless blockchains, the question of which consensus protocol should be used remains and depends on the desirable features of the particular application. For example, consensus protocols using the Proof of Work sybil control mechanism have significant drawbacks due to the low number of transactions that can be recorded per second and the enormous energy level that they require. On the other hand, even if more energy efficient and faster mechanisms (compared to Proof of Work) are used such as the (delegated) Proof of Stake mechanism, we need to carefully design how to incentivise nodes to act truthfully in permissionless blockchain networks. Budish [10] shows that there is an economic limit[1] on how important the Bitcoin blockchain could be. Therefore an interesting open question is to design energy efficient consensus protocols for permissionless blockchain voting system where nodes (are always incentivised to) create and validate blocks that contain only the valid votes cast.

Until the question of existence of an efficient permissionless system is resolved, permissioned blockchains seem to be much more plausible to run important elections because the incentivisation scheme for stakeholders running nodes is easier to understand. For example, if a node acts maliciously, then other nodes can publicly announce this information, ruin an identified node's reputation and change its permissions so that the malicious node could no longer create blocks. A main possible drawback of a permissioned blockchain based voting system is that all, or the majority, of the nodes in the network could still be under the control of a single authority. In this case, the difference with the centralised online voting systems is minor. Alternatively, a permissioned blockchain with a large variety of stakeholders operating a node can allow for some degree of decentralisation of control. An open question on this topic is therefore: How many different stakeholders are required before an acceptable level of decentralisation is reached? Who should these stakeholders be? And how does this acceptable decentralisation level change for different voting ap-

[1] Above this limit, disruption in the system could be worth more than the gains from preserving it.

plications?

Similar questions occur for corporate voting. In most corporate shareholding structures, minority shareholders do not usually have voting rights due to the large amount of bureaucracy associated with each shareholder voting. For this reason, decisions within a company are usually taken only by the main shareholder(s), which creates a more centralised and maybe undemocratic environment for the minority shareholders especially within companies where there is a large number of minority shareholders. However, if all shareholder rights (including voting rights) can be automatically managed by a permissioned blockchain based platform[1], this bureaucratic burden would theoretically be minimised.

Even though conceptually blockchain based voting starts at the point where the voter submits a vote, smart contracts can be inserted to provide additional functions to reduce fraud before and after votes are securely recorded on the blockchain. Such functions include (i) creating, storing and updating the list of eligible voters, (ii) hiding interim results, (iii) decentralising the counting[2] of votes and (iv) counting encrypted votes without even decrypting them [37]. In the rest of this section, we discuss voting issues that such functions (operated by smart contracts) can resolve.

Related to point (i), a key property underlining the integrity of democratic elections, is the concept of "one person, one vote". Unfortunately, there currently exist electoral systems which do not guarantee this. Fraudulent votes can be cast due to exploiting either the legal allowance of individuals being registered in multiple constituencies (e.g. see [15, 42]), or inaccuracies in the electoral roll (e.g. see [14, 42]). For example, a recent study in UK by the Electoral Commission [16] states that the UK electoral roll is approximately 9% inaccurate as it records incorrect or old address for some citizens. Furthermore, authors of [38] highlight that these inaccuracies may lead to voter turnout misreports (in terms of under-representing it) by almost 10%. The source of this problem seems to be the multiple centralised databases (where each database is controlled by one constituency) which do not communicate with each other. A centralised system across constituencies is not suitable for this. Citizens change addresses across different constituencies without always updating their new residency status. For this reason, multiple constituencies' electoral databases have incorrect data stored, such as non-eligible voters still registered as eligible in multiple areas and/or eligible voters non registered at all in the area they live.

[1] Possible ways to do this are mentioned in [33, 46].

[2] Every node can check that the advertised tally of the election is correct according to the voting data stored in the blockchain.

Updating a residency status (adding an individual's details to the electoral databse of the new constituency) and manually deleting the previous one (from the electoral database of the old constituency) for every citizen that move to a new address can be resource-wise exhausting and inefficient. On top of this, accidental or malicious acts such as erasing eligible voters lists or even the whole database, are more likely to occur due to the centralised nature of the current infrastructure and the non-permanent history back up of data. In contrast, a blockchain based voter registration system could be ideal to resolve such issues, as it can offer synchronised eligible voter databases which allow automatic alterations under consensus among constituencies. For example, when a voter moves from constituency A to constituency B, then constituency B confirms this move with constituency A before proceeding to data alterations. In addition, note that (*a*) all such actions would be recorded as transactions on the blockchain, thus there would be a history of the address changes of an individual, which can subsequently minimise the electoral roll inaccuracies due to incorrect addresses, (*b*) every constituency can send vote casting notifications of its own residents (to all nodes of the blockchain network) so that an individual can only vote in one place, thus voter impersonation or double voting cases can be eliminated.

Another point of interest is how blockchain based voting address the issue of vote anonymity (secrecy). In a paper ballot system, or an EVM based system vote secrecy is maintained simply by making sure that the voter does not have any identification on the ballot. In an online (centralised or decentralised) system, this may be a challenge since the voter logs in using some form of validated identity from the election authority. For example, if votes enter the database of a centralised system in a non-encrypted form, then the votes anonymity is compromised since voting data can be read by the election authority. Similar, if votes enter a blockchain (of a blockchain based system) in a non-encrypted form then vote anonymity is compromised since votes are transparent (and 'permanently' recorded in the block as unencrypted) and available to read (to any interested stakeholder). To ensure that voters are not intimidated or votes cannot be bought, vote secrecy can be maintained in an online system if votes are encrypted before[1] the votes enter the system. Then encrypted votes can be counted without being decrypted (using smart contracts[2] on a blockchain or special counting applications for a database).

[1]For example, a vote can be encrypted on a user's PC (before it is submitted).

[2]Note that a smart contract is a computational protocol that enforces an agreement between parties. If it's only placed within a centralised system, this agreement is not guaranteed to execute as defined (as it can be manipulated).

Compared with a centralised system, recall that data stored in a blockchain are transparent and extremely difficult to modify - so even if votes can be seen (encrypted or not) they cannot be modified. On the other hand, even if votes are submitted as encrypted in a centralised voting system, there is no guarantee that the counting process includes only the validated vote submissions. For example, if an election authority suspects the loss of her desirable outcome (by watching the interim results), it can delete or add enough 'encrypted' submissions to ensure the winning of her desirable outcome, which is a problem as discussed in point 3 of Section 2.1.1. Summing up, in principle, blockchain based voting can even securely count encrypted votes without compromising the counting process.

5.1 Challenges

Despite the new tremendous possibilities this technology bring, we outline a few areas that may need careful consideration during the development of such a voting platform. Note that some of these areas refer to existing security flaws that centralised voting architectures also cannot solve.

1. Network interfaces: We mentioned that the average user can connect with the blockchain network by signing into a website interface that is connected to a blockchain node. Voting in this manner leaves an open window for malicious entities to create interfaces that may change or steal votes before they arrive at a blockchain node. Note that this is an existing problem (e.g. how to securely connect to application platforms) and not a weakness of the DL technology itself. A way to get around this current problem, using the blockchain concept, is for a voter to connect to multiple random nodes (even using different electronic devices). In this way, voters can confirm that their vote is correctly cast and recorded on the blockchain and could minimise threats from viruses (as opposed to centralised systems which have a single source of data entrance - problem 1 of Section 2.1.1). If a voter finds their vote incorrectly recorded, they could either report it to a third party (which is not possible with a centralised authority) or resolve the situation themselves by changing[1] their vote.
2. Partially blockchain-based voting systems: It is obvious that to get the full benefits of this technology, it needs to be used in as many parts of the

[1]Depending on the voting application, this is a feature that could be implemented using smart contracts.

voting process as possible. By restricting its use to sub-parts, it may not do much better than centralised online voting. For example, storing only the final election result in the blockchain implies that there will be a secure (permanent) record of the election result. However, this does not guarantee that the election result is not already manipulated before it is calculated (and stored in a blockchain). Recall that a blockchain can be viewed as an underlying level of additional security to the current online voting systems and current online voting systems may have flaws. If we fail to use blockchain technology to its full extent, the security flaws of centralised online voting systems can still occur in the sections of the system where blockchain has not been used.

3. Majority attack: This is the most interesting type of challenge that is unique to the blockchain concept due to its decentralised nature. In a blockchain network, a single node or a coalition of nodes may have the majority of the network power. Depending on the consensus protocol used, blockchain network power can be defined with different methods, e.g. computational power or economic stake held. If malicious nodes have the majority of power in a blockchain voting system, then they also have the ability to discard valid votes or add invalid votes into a block. However, the rest of nodes (honest nodes) can be set to publicly announced malicious acts so that they can be investigated by independent third parties (and re-run the process if neccessary). Therefore, even if majority attacks can occur in a blockchain based voting application, these attacks can be used as a tool for identifying what type of manipulation attempts have occurred and by whom. In this way, malicious actors risk their reputation level and their participation to the system as nodes, which can considerably reduce their incentive to repeat (or do in the first place) data manipulative acts. In contrast, this malicious act identifying tool cannot be developed in a centralised voting system as by definition there is a single authority which, if malicious, can alter data without any consequences.

4. Running Cost: Developing and deploying a centralised system should generally be less costly compared to developing and deploying a decentralised system. However, the latter can improve the trust in the system simply because the same data can be checked and evaluated by multiple parties. This can be interpreted as a trade off between running cost of decentralisation and the additional trust in the system that it can bring. Therefore an open question to be answered is the following: what level of cost is acceptable to allow the voting system to have an acceptable level of

trust? Note that the level of allowed decentralisation in systems using blockchain technology depends also on the consensus protocol used.

To summarise, blockchain and distributed ledger technology can be viewed as a way to increase the difficulty of electoral fraud. For example, if we focus on the validation of votes - the difference between a centralised system and blockchain is that the latter can have multiple independent validators operating in parrellel who would all be involved in making sure that votes are valid. Similarly, for vote storage, it would be much more difficult to modify results in a blockchain system than a centralised system simply because it would require the attacker to control a large percentage of the resources on the network. The key features of the blockchain, namely the reliance on multiple independent block creators and validators, as well as the "append only" nature of the transactions, together with every node having access to the live code can create big disincentives to potential attacks. While collusive attacks are possible, it is very difficult to do so without being spotted by another stakeholder. There are still some open questions about the consensus protocols, such as how to maximally incentivise block creators and how to design the protocols in a way to discourage collusion between the nodes.

5.2 Fairer voting and more democratic political systems

In this section, we describe ways in which a blockchain based infrastructure can open new ways for the implementation of fairer voting mechanisms and new political systems. Below we focus on proxy voting, liquid democracy and the single-transferable-vote and discuss their applicability potential using this technology.

There are many voting applications where individual voters who are not able to vote can delegate their voting rights to someone else they trust (proxy) and then vote on their behalf, e.g. members of parliament not being able to attend a voting procedure, shareholders of a company represented by their proxy in corporate voting, and citizens who vote in national elections by posting their proxy information at least one week before the election. A more general concept of proxy voting which allows multiple transitive steps of vote delegation is called liquid democracy. More specifically, liquid (or delegative) democracy allows a citizen to either vote directly on a topic or transfer her voting rights to a trustee of her choice. Such a political governmental structure would work better on continuous referenda, e.g. the political structure of Switzerland which is known for its (semi) direct democracy where citizens are called to directly vote on some issues[1].

A liquid democracy model theoretically seem to help increasing voters turnout as those who are unsure of what to vote for can now delegate their voting rights and therefore participate in elections through their trustee. Apart from the promising turnout level improvement, it can give the electorate more options as they can delegate their vote to different representatives for different topics and even declare a preference order over other representatives (in case their first choice trustee does not vote) [32].

While the notion of creating a more engaging with citizens democratic structure sounds appealing, to officially name a proxy is usually a timely inefficient process, which makes the implementation of a liquid democracy style mechanisms almost impossible. However, we believe that it is at least worth exploring how such a system could be implemented in the future. For example, would a direct democracy model (e.g. Swiss system) be improved with a liquid democracy adoption? Opening new research areas for social scientists and developers, we propose a blockchain based infrastructure for the exploration of implementing a liquid democracy governmental model in the future. Due to its ability of keeping a highly immutable record of any action within a system, a blockchain can keep a safe and transparent record of all the transitive steps of delegations of voting rights of an individual (the chain among voters who continuously delegate voting rights). Therefore naming a proxy (who can then name another proxy, etc) could become a much more efficient[2] process when compared to the current bureaucratic paper work.

Apart from the automatic delegation of voting rights, smart contracts can also achieve automatic withdrawal of vote rights from their final destinations (candidates) so that they can be transferred to other candidates of the election. Observe that the latter feature would be ideal for the implementation of the single-transferable-vote (STV), where the votes received by one candidate need to be redistributed in the case where this candidate is eliminated in a round. Note that STV mechanism outcomes are more proportionately representative of what a society wish, as opposed to other voting mechanisms such as first past the post (FPTP). For this reason, the STV mechanism is considered as a fairer mechanism. Despite this feature, STV is not mainly adopted in elections as it is considered complex and inefficient. Would the development of a blockchain infrastructure allow the adoption of STV in more national elections? Note that Australia already uses this mechanism, where currently the counting might take up to a month (due to its complex vote counting algorithm [13]).

[1]After certain threshold number of signatures is reached for a petition.

[2]Using smart contracts, the delegation of voting rights to individuals can become automatic.

6 Categorising existing blockchain based voting systems

Even though there has been no large scale implementation[1] of a blockchain based voting platform such as in national elections, our analysis showed that blockchain technology has potential to address current electronic voting issues:

- Securely storing large voting data
- Tracking votes using accounts
- Using smart contracts

For each one, we describe their importance and describe multiple small real-life trials. Last, we detail the first, to the best of our knowledge, academic implementation of a blockchain online voting system.

6.1 Securely storing large voting data

Our first category focuses on the ability of systems using this emerging technology to secure large voting data in an efficient way. Firstly, securely storing large data is particularly important after elections, where data sometimes is used for other surveys, e.g. on voters behaviour, to re-confirm the election result or even to investigate future allegations on potential fraud.

Using the current centralised systems, voting records may have been censored from entering the database, fake vote records may have been added by the central authority (or someone hacking the central authority) and voting records may have been deleted or destroyed (see issue in Georgia US state [28]). On the other hand, when a voting record is added to a blockchain (either as plain or encrypted text[2]) this creates a permanent and transparent record of this information. By storing each individual vote on a blockchain, either anyone (for permissionless blockchains) or certain stakeholders (for permissioned blockchains) can inspect the entire voting record. If these voting records were stored in plain text then anyone inspecting them will be able to check that the

[1]The first blockchain based voting system was created on the Bitcoin blockchain and is still in use at the time of writing. This system has a restricted electorate (only Bitcoin nodes that create blocks can vote) and a restricted topic (they can only vote on improvement proposals for Bitcoin). Even if this is a great start for blockchain voting, the requirement of being a node in order to vote is too restrictive for real world voting (as it is technically complex to set one up and can be quite costly to run). This section focuses on blockchain voting systems where a voter does not have to be a node.

[2]If an encryption algorithm is used, this is run before the votes are added to the blockchain.

tally of the votes on the blockchain match the advertised tally. If these voting records were stored in encrypted form, a person inspecting them will have to have the correct decryption keys to check further information. Additionally, due to the properties of blockchain, we can say that a complete voting record stored in this manner is much more secure than stored in a centralised system, as there is no centralised point to hack and no single authority who can add/edit/delete a voting recording.

But when the electorate is large, storing every individual vote on a blockchain can put a strain on the number of votes that can be processed per second[1]. To allow for scalability (to store large amount of data in a reasonable time-frame), instead of adding all of the voting records onto the blockchain, subsections of the complete vote record can be grouped together and this data can be hashed (see Section 4.1.2) before it is written onto the blockchain. This means that each group of vote records can be represented by only one hash string of characters on the blockchain. Using the hashing technique, we can compress and securely store an audit trail of the entire voting record using only a few hash character strings, which requires significantly less storage[2] space. Even though the hashing process can also be occurred in a centralised system, there is no guarantee that the hashes will not be altered (or even completely deleted) during or after the election (problem 3 of Section 2.1.1).

The first attempts that we have identified on using blockchains only for their feature to securely store large data are the following. In February 2016, the Blockchain Tech Corp worked with Republican Presidential candidate Rand Paul to record the Iowa caucus results onto a blockchain for long term storage, via their VoteWatcher product [17]. More specifically, VoteWatcher hashed the votes onto the Bitcoin blockchain (while concurrently posting them in anonymised plain text to the Florincoin blockchain). In April 2016, the Blockchain Tech Corp were once again involved with American political parties, when they recorded the results of the Liberation party's Texas convention onto the same blockchain [21]. The largest occurrence to date for using a blockchain in this way occurred in 2017, when X0.1 conducted a practice election to test their SecureVote software by anchoring over one billion votes onto the Bitcoin blockchain [27]. This occurred by splitting the whole record into groups of approximate sizes of 140,000 votes each, and running the same hash algorithm over each group. Finally, they

[1] For example, the Bitcoin blockchain can take as little as 7 transactions per second, whereas some permissioned blockchains can handle around 10,000 transactions per second.

[2] Note that the hashing technique is not completely ideal though as it would require the voting record to be stored somewhere outside of the blockchain (in multiple other independent places) so that people can check that the advertised audit trail of the system matches the real voting data.

added each resulting hash output onto the Bitcoin blockchain (via a transaction). In this way, they efficiently created a permanent record of over a billion votes (using as little storage space as possible) which can be re-assessed[1] at any time in the future.

6.2 Tracking votes using accounts

Our second category focuses on the ability of blockchain voting systems to securely track ownership of votes. For example, we can use blockchain technology to prove that a voter sent a vote to a candidate or delegated a vote to a proxy.

To achieve this, every user (including voters and candidates) is assigned an account (a blockchain address). Recall that an account is a unique identifier[2] on the blockchain network from which users can send and receive transactions. Then votes can be transformed to digital tokens, which are initially placed in a voter's account. When a voter casts a vote token, this token is sent as a (recorded blockchain) transaction to the account of the selected candidate. Implementing an election in this way provides a higher level of trust in the final outcome (compared to using this technology only to store the voting record - the previous category) because transactions from an account require signature with a private key, which only the account owner should possess. Note that such an implementation (election with tracking of votes) can also be feasible with a centralised system in a similar way. However, using a centralised system, it requires trusting the election authority to not filter any undesired votes and/or add her own vote transactions from an account she owns or an account of another person (problem 3 of Section 2.1.1).

Tracking votes and securely recording each step within a vote's path can allow for implementations of transparent voting systems where votes are public knowledge[3] and voters can even observe the interim results. On top of this, securely tracking ownership of voting rights can help not only on more efficient implementations of proxy (or liquid democracy) voting but also on efficient implementations of the single-transferable-vote (STV) mechanism. Recall that both liquid democracy and STV require multiple delegations of voting rights (either from a voter to another voter or from an eliminated candidate to a remaining of this round candidate). However, to achieve implementations of such complex voting models, tracking votes in this way is not enough. For

[1]Any modifications to a group of hashed votes would be immediately detectable, as it would produce a different hash output.

[2]An account is protected by a *private key* operating as the blockchain address's password.

[3]Nodes are able to see which candidate a voter selects and when this occurs.

example, a vote (token) may move from candidate to candidate during an STV counting phase. When a vote (token) is in a candidate's account, only this candidate can move the tokens to another account (because only the candidate will hold the private key controlling his account). Also to add an extra layer of complexity, the eliminated candidate would have to know to which accounts (candidates) to transfer all of the vote tokens in her possession. To resolve such issues, we also need smart contracts, which are discussed in the next category.

To the best of our knowledge, using blockchain accounts to track votes has not been used in a real trial so far. However, technical examples of how an election could be modeled in this way are given in the NEM blockchain [12] and the counterparty protocol (built on top of Bitcoin) [18]. This protocol allows any stakeholder to create a vote and assign vote tokens to others. Both the NEM and Counterparty protocols allow votes and interim results to be public knowledge[1].

6.3 Using smart contracts for election implementation

Smart contracts are pieces of code that are triggered by transactions when placed on the blockchain. If the logic of a voting system is coded into a smart contract (e.g. how votes are authorised, collected, stored, tallied and if votes need to be encrypted), then they can be used to increase the general knowledge on how the voting system operates and, in turn, promote the public's trust in it. Smart contracts encourage this because anyone connected to the blockchain can see its code.

Voting systems using this feature allow key aspects of the election to be controlled by smart contracts. There are many variations of this model but the majority follows this simple pattern: a smart contract lists the election information (including the candidates) and the voters select their preferred candidate by calling a function[2] of the smart contract code. The key differentiator of smart contracts compared to centralised systems is that every blockchain node owner can see the exact piece of the (smart contract) code being run when a blockchain transaction triggers it. This level of transparency cannot be achieved for code running on a centralised system, because only the centralised authority is running the code and proving to a third party what code was run on a centralised system (when a vote is received) is near-impossible.

[1] To avoid individual votes and interim results becoming public knowledge, [49] describes a protocol to build an election on top of the ZCash blockchain (which makes use of the more private features of this specific blockchain technology).

[2] This occurs by adding a transaction onto the blockchain that details this information.

The ability of blockchain based voting systems to integrate smart contracts is arguably the one with the most potential due to the transparency of the code and the multiple additional features that smart contracts can allow for. Indications of this large potential are the number of real-world examples, which include the following. In April and May of 2017, a parallel French election using the majority judgment voting system [7] occurred on a blockchain voting platform, named Cocorico [34]. A total number of 52809 votes were collected in the first round while 15251 votes were collected in the second and final round of voting. Cocorico allows valid voters to send their cast ballot to its own web server that would subsequently add it, in a non encrypted manner, to a smart contract on the Ethereum permissionless blockchain [1]. This smart contract contained the list of eligible voters and who they voted for in a non-encrypted manner. In the smart contract Cocorico had special permissions to control who could vote in the election, but the electorate could check that their vote was recorded (if they had the technical expertise to connect to the Ethereum blockchain themselves). In December 2017, the Moscow government upgraded their online voting platform called ActiveCitizen[1] to include a permissioned blockchain voting system based on Ethereum [27], where all votes were added to a smart contract in the blockchain. Users must go onto the ActiveCitizen website to send their vote to the ActiveCitizen servers before it gets logged onto the blockchain. Therefore, due to the fact that ActiveCitizen has a permissioned blockchain implementation, their electorate must trust that at least one of the nodes is trustworthy enough to accurately report if a person's vote has been recorded. Another prominent example is [37], which produced open source code to show how votes can be held in encrypted form in the smart contract and re-tallied by anyone after the election has ended (without decrypting the original votes).

6.3.1 A DLT voting implementation

In April 2019, there was the first academic trial of a new voting platform, called Verify My Vote (VMV) and run by Electoral Reform Services Ltd, University of Surrey and King's College London [2, 48]. This version upgrades Electoral Reform Service's current implementation with an additional level of security through distributed ledger technology and is described in detail as follows. The general idea is that VMV makes use of distributed ledger technology as a component of

[1] ActiveCitizen (originally released in 2014) allows citizens to influence management decisions related to Moscow's urban design.

the implementation of the e-voting protocol Selene[1] to: (a) manage the commitments to ballot tracker numbers in advance of the election (ballot tracker numbers are used to allow voters to check their ballot was recorded correctly), (b) record the verifiability evidence produced during the election, and (c) record the evidence of correct ballot mixing and decrypting to not reveal the identity of any voter. VMV aims to achieve individual[2] and universal verifiability[3] without modifying the core voting system, and also without significantly changing the voter's experience.

By using Selene as the e-voting protocol, VMV fulfils two main attractive features. First, the votes cast by voters are collected and made available on a permissioned[4] distributed ledger (DL) in a plain text form and are paired with suitably anonymised tracking numbers. This is in contrast with many other e-voting systems in the literature, where only an encrypted/hashed version of the votes is published publicly. Having plaintext votes published on the distributed ledger means voters are not required to trust (or having any particular knowledge of) the cryptographic processes utilised in the election to be confident that their vote was correctly recorded and tallied. Since votes are published in plain text on the distributed ledger, individual verifiability is guaranteed (anyone can check their intended vote against the plain text vote recorded in the distributed ledger through looking up the vote paired with their tracking number) and so is universal verifiability (anyone can compute the tally from the plaintext votes). Second, VMV also provides a coercion mitigation mechanism, via the Selene protocol, which voters can exploit in case of coercion.

For the verifiability purpose of VMV, a permissioned distributed ledger is used to store encrypted votes alongside the encrypted tracking numbers during the election, and to store plain text votes alongside the anonymised tracking numbers after the election ends. Instead of having only one election authority

[1] Selene is a new end-to-end verifiable voting protocol. It is designed for a greater ease of understanding for the voters when it comes to verifying their ballots as after the election they can see their vote in plaintext recorded next to their ballot tracker number. Furthermore, Selene proposes an approach which provides coercion-resistance for the system. It achieves this through recording commitments to ballot tracker numbers issued to voters in such a manner that voters cannot provide proof on which ballot tracker number they have been assigned. This is enabled by using some sophisticated cryptography in the Selene protocol. Last, the Selene protocol does not require any additional voter action during the ballot casting stage, which greatly simplifies the voter's experience.

[2] Any voter is able to check that her or his ballot is correctly recorded.

[3] Anyone can determine that all of the ballots in the electronic system have been correctly counted.

[4] The Quorum distributed ledger platform is used to create a permissioned distributed ledger in the VMV. Quorum is an enterprise-focused version of Ethereum which offers high speed and high throughput processing of private transactions within a permissioned group of known participants.

controlling the whole election and being responsible for publishing the election results, VMV distributes control of an election over several election trustees via the permissioned distributed ledger. This means that the election authorities should all agree on the election information that needs to be published in the distributed ledger, increasing the level of integrity and trust in the election. On the other hand, distributed ledgers offer the additional property where the stored information cannot be changed, so the voter can ensure that her vote recorded during the voting period is never changed or removed during the tally process (after the election finishes). The VMV distributed ledger offers read and write access granted to the election trustees and allows read only access to voters and auditors. The known participants in the permissioned distributed ledger group include the election authority and other trusted parties such that they cannot violate the integrity of the distributed ledger unless a threshold of them collude. The chosen consensus protocol is proof of authority (POA). Note that VMV uses smart contracts in the following way. Publicly viewable data of the voters are sent via transactions to a smart contract recorded at a specific blockchain address (in such a way that a ballot is not directly linked to a voter). Therefore, external stakeholders interested in verifying the result can call a function of that smart contract to download the available data and verify that an individual's vote is correctly recorded and that the announced election result corresponds to the downloaded data.

To conclude, scholars have identified many flaws in online voting systems and blockchain technology has a promising potential in solving these problems. However, this is still very much a developing technology and there are many questions that need to be answered before we deploy a large scale online voting system dependent on it. Most recent small scale implementations using blockchain-based online voting systems have been focused on limited aspects of what this technology can achieve, e.g. gathering vote data from users and storing them on a blockchain. We believe that the whole process of an online voting platform (voter registration, voter authentication, vote collection and vote counting) should be underlined by a blockchain based infrastructure in order to increase security compared to the platforms without such an infrastructure. If some areas remain exposed (without being underlined by a blockchain), then these areas are as vulnerable to attacks as the current online platforms. In other words, if blockchain technology is partially used, it does not guarantee any improvement towards making online voting secure.

From the environmental and governmental point of view, we need to identify what would be the most energy efficient consensus protocol that satisfies desirable properties for our election systems, such as: finding an acceptable level of decentralisation of the system so that public trust can be established, and how election systems can be recovered in case of a majority attack. Given that we have good indicative answers for the above, the next step should be to make this transition smooth for organisations and governments that want to use blockchain-based online voting, i.e. what should the transition steps be from a centralised election authority (possibility using paper ballots) to a decentralised technological implementation (blockchain) in online voting in order to minimise disruption?

For corporate voting, implementing an online voting system is an easier use case due to the fact that votes are normally public. Therefore, even if we are a few steps away from a blockchain implementation for political office elections, corporate blockchain based online voting systems are more developed [33, 46]. Last, we believe that smart contracts are going to play an important role in this area, as they can offer multiple additional features for any voting platform using a blockchain based infrastructure.

7 Conclusion and open questions

In this paper we describe how online voting and in particular the blockchain technology can help to reduce electoral fraud. Our main objective is to provide a conceptual basis for understanding the key ideas behind the design of this technology. Essentially, the blockchain technology is a decentralised ledger which has the property that all transactions in the ledger must be approved by multiple nodes in the network and these transaction records cannot be modified once they are accepted in a block (therefore in the blockchain), unless all of the relevant nodes agree. In a sense, something is true only when the threshold of agreement is reached among the nodes. Unlike an electronic voting machine or a centralised voting system, it is much more costly to commit electoral fraud because one needs to hack not just one server but many independent servers, all at the same time. When the costs of doing so are high relative to the benefits, then fraud is discouraged. The system also provides transparency since each node has access to the code being run (whereas in a centralised system the code is run only from a single location). The combination of the above key features could significantly contribute to increase the public's trust on an online voting platform as well as the platform's accessibility levels. The latter is not only due to its online nature but also due to the multiple nodes (entry points) that voters can use to cast their vote.

The ideas underlying the workings of such a decentralised system raise many open questions for economists and social scientists in general: How to design the consensus protocol in a way to minimise costs while taking into account strategic incentives of block creators? Who should be the stakeholders in elections (where permissioned blockchains are likely to be the norm)? Finally, a blockchain based infrastructure opens up the possibility of exploring new voting systems, e.g. it allows secure and transparent delegations of voting rights. Of course, a blockchain-based voting system is not yet implemented for a large scale election but, in the meantime, economists have the opportunity to help in designing such a system.

References

[1] *Ethereum: A secure decentralised generalised transaction ledger eip-150 revision (1e18248 - 2017-04-12)*, <https://bit.ly/2QTq6VV>.

[2] *An introduction to verify my vote platform*, <https://vmv.surrey.ac.uk/introduction>.

[3] John S. Ahlquist, Kenneth R. Mayer, and Simon Jackman, *Alien abduction and voter impersonation in the 2012 us general election: evidence from a survey list experiment*, Election Law Journal: Rules, Politics, and Policy **13** (2014), no. 4, 460–475.

[4] Yackolley Amoussou-Guenou, Antonella Del Pozzo, Maria Potop-Butucaru, and Sara Tucci-Piergiovanni, *Correctness of Tendermint-Core Blockchains*, 22nd International Conference on Principles of Distributed Systems (OPODIS 2018), vol. 125, 2018, pp. 16:1–16:16.

[5] Christian Badertscher, Peter Gazi, Aggelos Kiayias, Alexander Russell, and Vassilis Zikas, *Ouroboros Genesis: Composable proof-of-stake blockchains with dynamic availability*, Proceedings of the 25th ACM SIGSAC Conference on Computer and Communications Security, CCS, 2018, pp. 913–930.

[6] Arati Baliga, *Understanding blockchain consensus models*, 2017, <https://goo.gl/SD1kM9>.

[7] Michel Balinski and Rida Laraki, *Majority judgment: Measuring, ranking, and electing*, MIT Press, 2011.

[8] Adrian Beaumont, *How the votes are counted*, <https://goo.gl/quJ2j6>.

[9] Ethan Buchman, *Tendermint: Byzantine fault tolerance in the age of blockchains*, Master's thesis, The University of Guelph, 2016.

[10] Eric Budish, *The economic limits of bitcoin and the blockchain*, 2018.

[11] Vitalik Buterin and Virgil Griffith, *Casper the friendly finality gadget*, CoRR, vol. abs/1710.09437, 2017.

[12] Sergi Canal, *Releasing the NEM voting module*, 2017, <https://goo.gl/MycvUb>.

[13] The Australian Electoral Commission, *Counting the votes*, <https://goo.gl/jHrsbz>.

[14] The Electoral Commission, *Accuracy and completeness of electoral registers*, <https://goo.gl/Tp2eU9>.

[15] ______, *I have two homes. can i register to vote at both addresses?*, <https://goo.gl/ajykJA>.

[16] ______, *The december 2015 electoral registers in great britain*, 2016, <https://goo.gl/rwmhvi>.

[17] Blockchain Technologies Corp, *2016 Iowa caucus results forever documented on blockchain*, 2016, <https://goo.gl/4wubZG>.

[18] Counterparty, *Verifiable voting with tokens*, 2017, <https://goo.gl/FBVjMM>.

[19] Bernardo David, Peter Gazi, Aggelos Kiayias, and Alexander Russell, *Ouroboros Praos: An adaptively-secure, semi-synchronous proof-of-stake blockchain*, 37th Annual International Conference on the Theory and Applications of Cryptographic Techniques, 2018, pp. 66–98.

[20] Sisir Debnath, Mudit Kapoor, and Shamika Ravi, *The impact of electronic voting machines on electoral frauds, democracy, and development*, (under review) (2019).

[21] Michael del Castigio, *Libertarian party of Texas to store election results on three blockchains*, 2016, <https://goo.gl/RBtKQK>.

[22] Office for Democratic Institutions and Human Rights, *OSCE & ODIHR election observation mission final report*, 2017, <https://goo.gl/yxjGWB>.

[23] Thomas Fujiwara, *Voting technology, political responsiveness, and infant health: evidence from brazil*, Econometrica **83** (2015), no. 2, 423–464.

[24] ______, *Voting technology, political responsiveness, and infant health: evidence from brazil*, Econometrica **83** (2015), no. 2, 423–464.

[25] J. Paul Gibson, Robert Krimmer, Vanessa Teague, and Julia Pomares, *A review of e-voting: the past, present and future*, Annales des Télécommunications **71** (2016), no. 7-8, 279–286.

[26] Sharad Goel, Marc Meredith, Michael Morse, David Rothschild, and Houshmand Shirani-Mehr, *One person, one vote: Estimating the prevalence of double voting in u.s. presidential elections*, American Political Science Review (Forthcoming).

[27] Sarah Holder, *Can the blockchain tame moscow's wild politics?*, 2017, <https://www.citylab.com/life/2017/12/can-the-blockchain-tame-moscows-wild-politics/547973/>.

[28] The Independent, *Hackers may have gained 'almost total control' of an election server in georgia, report says*, 2020, <https://tinyurl.com/sqxvdwc>.

[29] Aggelos Kiayias, Alexander Russell, Bernardo David, and Roman Oliynykov, *Ouroboros: A provably secure proof-of-stake blockchain protocol*, 37th Annual International Cryptology Conference, 2017, pp. 357–388.

[30] Aggelos Kiayias, Thomas Zacharias, and Bingsheng Zhang, *End-to-end verifiable elections in the standard model*, 34th Annual International Conference on the Theory and Applications of Cryptographic Techniques, 2015, pp. 468–498.

[31] Peter Klimek, Yuri Yegorov, Rudolf Hanel, and Stefan Thurner, *Statistical detection of systematic election irregularities*, Proceedings of the national academy of Sciences of the United States of America **109** (2012), no. 41, 16469–16473.

[32] Grammateia Kotsialou and Luke Riley, *Incentivising participation in liquid democracy with breadth first delegation*, CoRR, vol. abs/1811.03710, 2018.

[33] Grammateia Kotsialou, Luke Riley, Amrita Dhillon, Toktam Mahmoodi, Peter McBurney, Paul Massey, and Richard Pearce, *Using distributed ledger technology for shareholder rights management*, Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems, AAMAS 2018, Stockholm, Sweden, July 10-15, 2018, 2018, pp. 1986–1988.

[34] LaPrimaire, *The presidential election at the majority judgment - results*, 2017, <https://goo.gl/ySHm2i>.

[35] Frank Lynn, *Boss tweed is gone, but not his vote*, 1984, <https://goo.gl/GEYkRN>.

[36] Daniele Magazzeni, Peter McBurney, and William Nash, *Validation and verification of smart contracts: A research agenda*, IEEE Computer **50** (2017), no. 9, 50–57.

[37] Patrick McCorry, Siamak F. Shahandashti, and Feng Hao, *A smart contract for boardroom voting with maximum voter privacy*, Financial Cryptography and Data Security - 21st International Conference, FC 2017, Sliema, Malta, April 3-7, 2017, Revised Selected Papers, 2017, pp. 357–375.

[38] Jonathan Mellon, Geoffrey Evans, Edward Fieldhouse, Jane Green, and Christopher Prosser, *Opening the can of worms: Most existing studies of aggregate level turnout are meaningless*, SSRN's eLibrary (2018).

[39] Satoshi Nakamoto, *Bitcoin: A peer-to-peer electronic cash system*, 2008, <http://bitcoin.org/bitcoin.pdf>.

[40] Yusuf Neggers, *Enfranchising your own? experimental evidence on bureaucrat diversity and election bias in India*, American Economic Review **108** (2018), no. 6, 1288–1321.

[41] BBC News, *Tower Hamlets election fraud mayor Lutfur Rahman removed from office*, 2015, <https://goo.gl/gR79u5>.

[42] The PEW Center on the States, *Inaccurate, costly, and inefficient: Evidence that America's voter registration system needs an upgrade*, <https://goo.gl/BPGJWe>.

[43] Gustavo Palencia, *OAS says Honduran presidential election should be redone*, 2017, <https://goo.gl/P7BCdM>.

[44] Nathaniel Persily, Robert F. Bauer, Benjamin L. Ginsberg, Brian Britton, Joe Echevarria, Trey Grayson, Larry Lomax, Michele Coleman Mayers, Ann McGeehan, Tammy Patrick, and Christopher Thomas, *he American voting experience: Report and recommendations of the presidential commission on election administration*, Presidential Commission on Election Administration (2014).

[45] Thomas Wynter Pippa Norris and Sarah Cameron, *Corruption and coercion: The year in elections*, 2017, <https://goo.gl/PYwAoe>.

[46] Luke Riley, Grammateia Kotsialou, Amrita Dhillon, Toktam Mahmoodi, Peter McBurney, and Richard Pearce, *Deploying a shareholder rights management system onto a distributed ledger*, Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems, AAMAS 2019, Montreal, Canada.

[47] Team Rocket, *Snowflake to avalanche: A novel metastable consensus protocol family for cryptocurrencies*, 2018, <https://goo.gl/oEWysZ>.

[48] Muntadher Sallal, Steve Schneider, Matthew Casey, Catalin Dragan, Francois Dupressoir, Jinguang Han, Luke Riley, Helen Treharne, and Joe

Wadsworth, *VMV: Augmenting internet voting systems with selene verifiability using permissioned distributed ledger*, (under review) (2019).

[49] Pavel Tarasov and Hitesh Tewari, *Internet voting using Zcash*, IACR Cryptology ePrint Archive **2017** (2017), 585.

[50] Nicolaus Tideman, *The single transferable vote*, Journal of Economic Perspectives **9** (1995), no. 1, 27–38.

[51] Web Roots Democracy, *Cost of voting*, 2017, <https://goo.gl/nhv3Dw>.