Towards Dependable, Scalable, and Pervasive Distributed Ledgers with Blockchains

Kaiwen Zhang, Hans-Arno Jacobsen
Middleware Systems Research Group
École de technologie supérieure, Canada
University of Toronto, Canada
Technical University of Munich, Germany

Abstract—**Distributed blockchain ledgers are on the verge of becoming a disruptive technology, capable of profoundly impacting a wide range of industries and established applications, such as cryptocurrency, and allowing for novel use cases in both the public sector (e.g., eGovernment, eHealth, etc.) and the private sector (e.g., finance, supply chain management, etc.). Blockchains promise the ability to maintain critical information in a trustworthy repository without any centralized management. The reliability of blockchain-enabled applications is based on the innate immutability of stored data, maintained through cryptographic means, which enables blockchains to provide transparency, efficiency, auditability, trust, and security. As the technology is still in its infancy, a number of pain points must be addressed in order to make distributed ledgers more dependable, scalable, and pervasive. In this paper, we present the research landscape in distributed ledger technology (DLT). To do so, we describe a taxonomy of blockchain applications called *blockchain generations*. We also present the DCS properties (Decentralization, Consistency, and Scalability) as an analogy to the CAP theorem. Furthermore, we provide a general structure of the blockchain platform which decomposes the distributed ledger into six layers: Application, Modeling, Contract, System, Data, and Network. Finally, we classify research angles across three dimensions: DCS properties impacted, targeted applications, and related layers.**

1. Introduction

A blockchain system, also known as distributed ledger technology (DLT), handles and shares transaction records across a network of users. The records can be verified by each user, and transactions are tied together using cryptography so that altering the records is nearly impossible. Blockchains allow for transparency, efficiency, immutability of records, auditability, and security, which reduce problems of system component and database redundancy, fraud, misuse, and many cybersecurity challenges.

The most popular application of blockchains remains its original purpose as the cryptocurrency Bitcoin, originally devised in 2008 [1]. Newer systems, such as Hyperledger [2] and Ethereum [3], focus on extended processing capability to execute arbitrary code (i.e., Smart Contracts [4]) in order to support a wide range of decentralized applications (ĐApps). DLT is on the verge of becoming a disruptive technology, and hold an enormous potential to profoundly impacting a wide range of industries and established applications, such as cryptocurrency, and allowing for novel use cases in both the public sector (e.g., eGovernment, eHealth, etc.) and the private sector (e.g., finance, supply chain management, etc.).

Although there has been a tremendous amount of hype around blockchains in the past few years, few industries have adopted it so far. One of the reasons for this reluctance is a lack of trust in this new technology. Another is the perceived difficulty of integrating blockchain with existing systems and processes that companies have spent years building around mandated regulations. ĐApps are also susceptible to security and privacy concerns due to the lack of formal verification or guarantees provided by the DLT platform, as evidence by earlier incidents (e.g., DAO Attack and the Parity Multisig Bug [5]). Finally, current blockchain platforms do not provide the throughput or scalability necessary to adequately support the needs of today's applications. Thus, there is a clear need for fundamental research in involved disciplines, such as distributed systems, cryptography, and software engineering, to alleviate these pain points and unlock the full potential of blockchains.

In this paper, we provide a comprehensive effort at structuring all aspects of blockchain research. Since the technology is still in its development stage, it is important to provide a clear roadmap to the future effort necessary to improve DLT and facilitate their adoption in a wide variety of verticals. By providing a clear structure, blockchain researchers can better position their works and understand the likely benefits of their results with respect to potential application domains. Furthermore, we argue that major players require a comprehensive package detailing blockchain technologies and their underpinnings from first principles, rather than the piecemeal and ad hoc approaches currently appearing at a staggering rate.

To the best of our knowledge, this paper is the first comprehensive attempt at structuring academic blockchain research. Previous works either survey a limited scope (e.g., smart contracts [6], consensus [7], or security [8]). Existing vision papers list challenges in a particular domain (e.g., Internet of Things [9], scalability [10], and Business Process

Management [11]) or without providing a clear structure to classify research [12].

The contributions of this paper are as follows:

1) Provide a clear and precise definition of key terms and the DCS properties (Decentralization, Consistency, and Scalability) as an analogy to the CAP theorem (Section 2),
2) Create a taxonomy of blockchain applications called *blockchain generations* (Section 3),
3) Model modern DLT platforms as a blockchain stack of six layers (Section 4),
4) Classify research challenges across three dimensions: DCS properties impacted, targeted applications, and related layers (Section 5).

2. Core concepts

In this section, we review core concepts related to blockchain technologies. In particular, we clearly define key terms which will be employed throughout the rest of the paper, and list the various components that form a blockchain.

2.1. Key terms

A *distributed ledger* is a collection of records, commonly financial transactions, which is recorded as a log, where new data is appended at the end of the ledger. The ledger is replicated across multiple nodes, where each maintains a consistent copy of the data.

At its core, a distributed ledger consists of three major components: the blockchain *data structure*, a P2P *network* of servers maintaining the blockchain, where peers execute a *consensus protocol* which regulates how new data is added. Various alternatives are possible for each component, depending on the specifications of the application. Figure 1 illustrates a basic working model of a distributed ledger.

Broadly speaking, we can categorize distributed ledgers as either *public* or *private*. In a public ledger, anyone is free to join the network as a peer and maintain the ledger. Conversely, a private ledger restricts access to a set of machines, usually belonging to a consortium of organizations. From a technical point of view, the major difference between both types of ledgers concerns the amount of trust between nodes. In a public ledger with no trust among peers, each node is capable of behaving arbitrarily, hence additional measures (i.e. incentives) are required to tolerate malicious attackers. On the other hand, a private ledger assumes some level of trust between peers, which means the underlying failure model is weaker. Private ledgers can therefore obtain better performance (throughput and scalability) than their public counterparts in exchange for limited decentralization capabilities.

The follow subsections now describe in details the three major components of distributed ledgers.

2.2. Data structure

The most commonly used data structure for distributed ledgers is the *blockchain*. Each block contains a set of records added to the ledger and is immutable once generated. A newly created block is inserted in the blockchain by linking it to the last block in the chain. To prevent tampering of information found in existing blocks, the integrity of each block can be verified using a hash function which takes into consideration all preceding blocks. Hence, in order to successfully alter an older block, one must also modify all following blocks, which is considered unfeasible or unlikely. This implies that the amount of trust in the information contained in a block depends on the block age (i.e., the number of blocks following it).

The internal structure of each block varies greatly between systems. It commonly consists of one or more Merkle trees, which are hash-based data structures. Merkle trees are advantageous as they provide fast lookups of transaction inclusion for lightweight clients, who do not possess a full copy of the ledger. For instance, Bitcoin employs Merkle trees for the Simple Payment Verification protocol [1]. Figure 2 summarizes the Bitcoin data structure.

Due to its strong guarantees, blockchains are the most prevalent form of storage for ledgers. For the remainder of the paper, we will focus exclusively on blockchain-based ledgers.

2.3. P2P network

In order to exchange information about the ledger, peers communicate using a network. The network topology is not often disclosed or well understood in popular blockchain systems. In general, an unstructured overlay network is employed, where each peer is connected to a variable set of neighbors. Gossiping is employed to broadcast data, such as new transactions and blocks, among the peers using multiple rounds of message exchanges.

2.4. Consensus

The consensus protocol is the most varied and studied in the blockchain community. It is most frequently used by the peers to agree on the content of the next block (i.e., confirmed transactions) to be added to the blockchain. In some instances, consensus is also required to elect a leader, agree on the outcome of smart contract executions, and the recipients of mining rewards or transaction fees.

We provide two categories of consensus protocols: *Proof-based* and *Leader-based*.

For proof-based consensus, two components are required: a *block proposal* algorithm and a *branch selection* algorithm. A *block proposal* algorithm allows any peer to propose a block to the rest of the network, which can be quickly validated by the other peers. As multiple blocks can be proposed simultaneously (e.g., due to network latencies), branches can occur where peers operate on different versions of the blockchain. To reconciliate branches, a *branch*

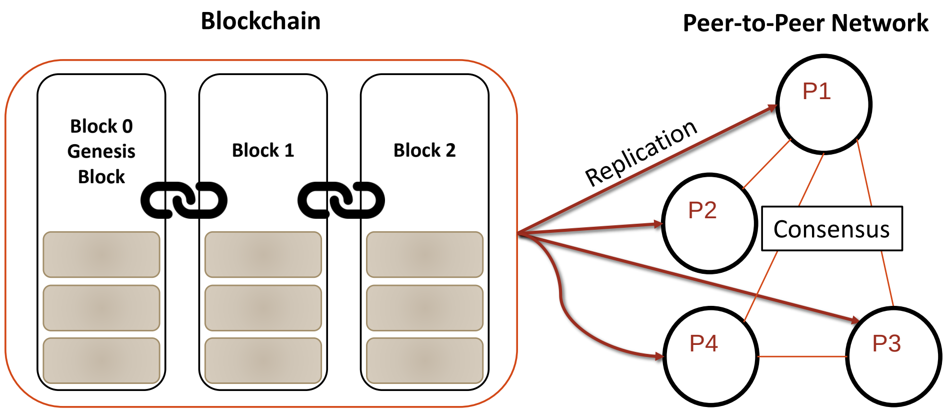


Figure 1. Distributed ledger: Basic architecture

selection algorithm is used by peers to decide which branch to accept, which ensure that all peers eventually converge to the same state.

The original blockchain consensus algorithm, pioneered by Satoshi Nakamoto for the Bitcoin system, employs a block proposal mechanism called *Proof-of-Work* to enforce data immutability [1]. In order to insert a new block to the chain, a computational puzzle must be solved which takes as input the entire blockchain. In addition, Proof-of-Work is coupled with the longest chain branch selection policy (also known as *Nakamoto consensus*, where non-faulty peers are always operating on the longest known branch. These two mechanisms together enforce the immutability of the blockchain data, as it takes an attacker a large volume of computational resources (e.g., more than 51% of the entire network) in order to alter existing data and rewrite all blocks following it to generate consistent proofs. Transactions are submitted by client users not actively involved in the ledger, which are then pooled into blocks to be inserted to the blockchain. Due to concurrency and network delays, multiple blocks with a valid proof (e.g., two blocks $N+1$) may be generated based on the current blockchain (e.g., ending at block N). However, the longest chain policy will eventually force all peers to converge on the state of the blockchain and choose one block and reject all other candidates.

Due to the immense computational costs, which are economically and environmentally prohibitive, many alternatives have been proposed to Proof-of-Work. One example is Proof-of-Stake (e.g., in PeerCoin [13]), which requires participants to commit a share of the digital currency in order to forge new blocks, which substantially reduces the computational efforts required to preserve safety.

Public ledgers also require an incentive system to encourage peers to join the network and maintain the blockchain up-to-date with incoming transactions. In Bitcoin, this is accomplished by rewarding the peer (also called miner) who successfully adds a block to the blockchain. In addition to this block reward, a miner is also allowed to collect all fees associated to the transactions in the new block. Some consensus protocols combine proof-based consensus with leader-based, such as Bitcoin-NG [14]. Proof-of-Work is employed to determine the next leader, who can then propose the next sequence of blocks to be inserted.

In private ledgers, where a certain level of trust is maintained among peers, leader-based consensus is more commonly employed. For instance, Hyperledger employs an ordering service to determine the order of incoming transactions [2]. This ordering service can be either centralized (i.e. static leader) or distributed (i.e. with periodic leader election). The ordering service has full control of the block proposal process: there is thus no possibility of branching possible, and no branch selection algorithm is therefore required. As the ordering service does not execute transactions, the committing peers which receive blocks from the orderer must then execute a Practical Byzantine Fault-Tolerance protocol (PBFT) to agree on the outcome of the transactions.

2.5. Smart contracts

More recently, smart contracts have been introduced to support general applications beyond cryptocurrency. Smart contracts are programs automatically executed by the blockchain miners whenever their encoded conditions are triggered. Smart contracts are also transparent since they can be reviewed and agreed upon by the interacting parties prior to inserting them to the blockchain. Finally, the correct execution of smart contracts is guaranteed due to the immutability properties of a blockchain, which prevents it from being corrupted. Two notable systems which support smart contracts are Ethereum and Hyperledger.

The following is a sample “Hello World” application written in Solidity for Ethereum [15]:

```
pragma solidity \begin{lstlisting}^0.4.4;

contract HelloWorld {
 string public greeting;
 function HelloWorld(string _greeting) {
  greeting = _greeting;
 }
```

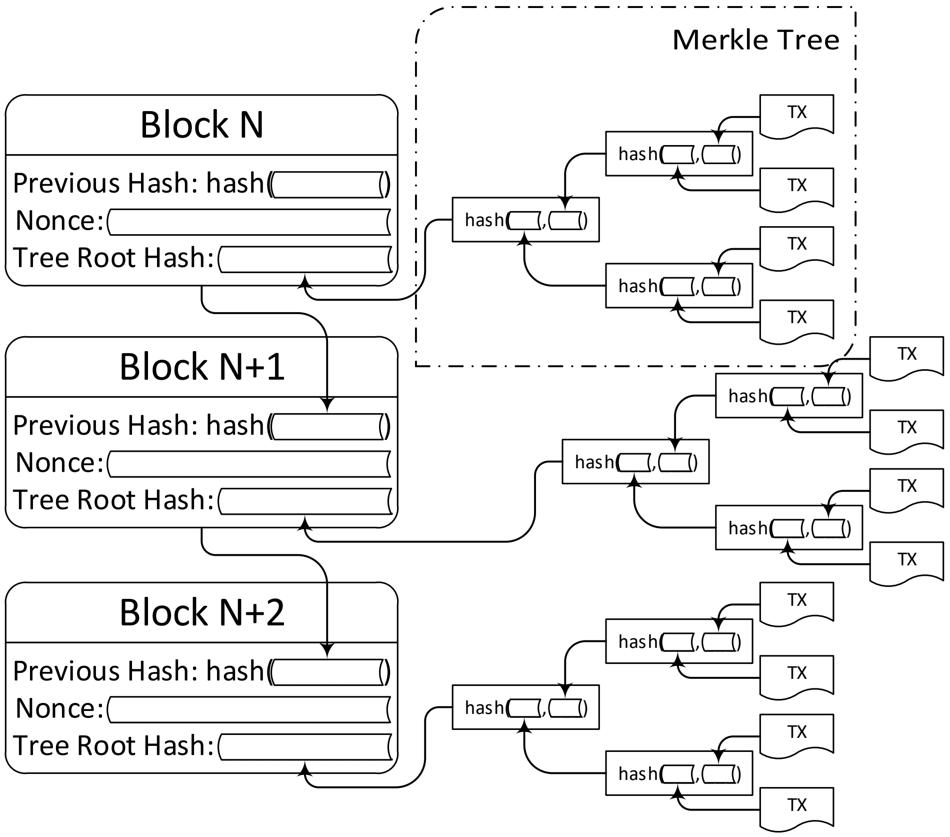


Figure 2. Block structure

```
  function setGreeting(string _greeting) {
    greeting = _greeting;
  }

  function say() constant returns (string) {
    return greeting;
  }
}
```

The function $HelloWorld()$ initializes a new object, while the function $setGreeting()$ writes a new message. Both of these require a transaction to execute and cost some *gas*, which is given to the miner who includes the transaction in a block. The function $say()$ is *constant*: it does not cost gas to execute, since it only reads existing information. It can therefore be executed without sending a transaction.

2.6. Distributed databases

Blockchain systems can be considered a specialized type of distributed database management systems (DDBMS). DDBMS are commonly used to enable large scale web applications, and exist in many forms, such as sharded relational databases (from Oracle, IBM, etc.), NoSQL key-value stores (HBase, Cassandra, MongoDB), and NewSQL data stores (Google Spanner). Blockchains differ in their emphasis on decentralized management through the use of cryptographic mechanisms to ensure security and privacy. Whereas common DDBMS are employed to store data across a number of data centers, all controlled by a single organization, blockchains are meant to be employed in applications running in environments where the network participants are assumed to belong to unknown or competing parties, and which therefore require intrinsic proofs of data integrity without having to rely on third-party verification. Thus, blockchains can simplify the deployment of a shared DDBMS at low cost and alleviate friction between parties when exchanging critical documents.

Due to the computational requirements of security and privacy, blockchain systems cannot match the velocity and throughput of traditional DDBMS. Further research is therefore required to reconcile the performance of blockchain systems with other data systems without sacrificing privacy and security.

Blockchain systems are also related to peer-to-peer file sharing networks such as BitTorrent and Freenet. While these systems also provide decentralization and censorship resistance, they lack a consensus algorithm for the online insert of new content, such as Proof-of-Work for Bitcoin transactions.

2.7. DCS properties

Generally speaking, blockchain platforms strive to maintain the following three desirable properties:

Decentralization: Blockchains are inherently decentralized, do not rely on a trusted third party for reliability, and provide censorship resistance for data and ĐApps. Furthermore, blockchain data should remain confidential and anonymous when operating in a trustless environment.

Consistency: The blockchain data should be exactly identical at all peers and at all times. Previously committed transactions are immutable. The current state of the blockchain should be completely verifiable using the entire history of the blockchain. The results of blockchain queries should be identical no matter which peer is contacted.

Scalability: The performance (i.e. higher throughput, lower latency) and availability of the system should increase as the amount of resources allocated (e.g., number of peers, computational power, etc.) increases. The system should be able to handle increasing volume of concurrent users, smart contracts, and queries.

We observe that there exists a trade-off between the DCS properties, which can be formulated similarly to the CAP theorem [16]: “A blockchain system can only simultaneously provide two out of the three properties”. We present three examples which corroborate with this observation.

Bitcoin can be considered a DC system, as its main focus lies on providing a consistent blockchain state in a decentralized environment. As a consequence of using Proof-of-Work in an incentive-driven network, Bitcoin is not a scalable system: over time, miners are incentivized to increase their hash power to increase their chance of successfully mining a block. Yet, Bitcoin does not yield increased performance despite the increase in power: the system is fixed to mine one block every 10 minutes, producing a throughput of 7 transactions per second.

Ethereum is also a DC system, as it produces a consistent blockchain in a public network. As with the original CAP theorem, the DCS properties constitute a spectrum rather than binary options. Hence, Ethereum provides a higher performance than Bitcoin by reducing the block time from 10 minutes to 10-40 seconds. This however negatively impacts consistency, as the occurrence of branches increases. Ethereum mitigates this issue by employing the GHOST protocol as the branch selection algorithm [17].

Hyperledger is a CS system, as it sacrifices decentralization by opting for a permissioned network. In exchange, Hyperledger relies on an efficient ordering service instead of Proof-of-Work, with a throughput above 10K transactions per second [18].

In general, we argue that any optimization of a blockchain system will improve some of its properties while decreasing others. Thus, it appears that “one size does not fit all” in the context of blockchain applications. It is therefore possible to tune blockchain systems to achieve the right balance of DCS properties suitable for a particular application.

3. Blockchain Application Generations

In light of the DCS properties, we now present a taxonomy of blockchain applications. Classifying potential use cases is necessary in order to understand objectives research projects should pursue to satisfy the requirements of the targeted applications.

We propose a taxonomy based on the concept of three *blockchain generations*: Blockchain 1.0, 2.0, and 3.0 [19]. Generations are ordered by time, although previous generations do not end when new ones begin. Instead, all three generations of applications are evolving at the same time. We argue that each generation has its own set of challenges, and that there are worthwhile research directions to pursue for each one.

3.1. Blockchain 1.0: Cryptocurrency

Blockchain 1.0 contains cryptocurrency applications. The distributed ledger is used to store transactions as exchanges of digital assets between the *wallets* of multiple clients. The semantics of validating transactions are hard-coded into the system.

Blockchain 1.0 is the oldest and most popular class of blockchain applications which currently exist. Beside Bitcoin, there exists over 600 alternative cryptocurrencies which are publicly usable [20].

Although 1.0 applications are currently being used for investment, cryptocurrency is ultimately designed to replace traditional fiat currency. To accomplish this goal, 1.0 blockchain systems must be made more dependable and scalable while retaining decentralization. Furthermore, any changes must be incrementally deployable to support legacy systems such as Bitcoin. One important challenge is the presence of *hard forks* when new versions of blockchain code are incompatible with previous ones. When a hard fork occurs, the userbase is divided when there is resistance to update the code.

3.2. Blockchain 2.0: ĐApps

Blockchain 2.0 is characterized by smart contracts, which support decentralized applications (ĐApps). Blockchain platforms for 2.0 applications provide a Turing complete smart contract language, and a public cloud infrastructure for ĐApps developers to host their code. For instance, Ethereum allows developers to create contract accounts to store their ĐApps, after paying an initial cost to store the contract code. Ethereum users can then choose to employ any of the ĐApps hosted on the Ethereum network by sending transactions which triggers these contracts. The cost of executing a transaction is calculated when executing the code, and is paid to the miner in a form known as *gas*.

2.0 applications go beyond traditional cryptocurrency as the semantics which regulate the transfer of digital assets are defined per application. Examples include crowdfunding, charity donation, forecasting, and decentralized autonomous organizations (DAOs) [19].

For 2.0, the research focus should be on the safety of smart contracts, which must be formally verified before they are permanently committed to the blockchain. Furthermore, scalability mechanisms for 2.0 differ fundamentally from 1.0 as they must explicitly consider the cost of retrieving smart contracts and executing them.

3.3. Blockchain 3.0: Pervasive apps

Blockchain 3.0 applications involve entire industries and the public sector. Due to their large scale, 3.0 applications must be deployed using a dedicated infrastructure rather than a public cloud. Furthermore, 3.0 applications are likely to interact with the physical world and therefore involve the Internet of Things (IoT). Example of applications include land registries [21], eHealth [22], and supply chain management [23].

Due to the efforts required to support a 3.0 application, clean-slate blockchain designs can be employed for superior scalability. In some context, decentralization can be toned down when the application use cases are restricted to a consortium of organizations, which can be leveraged to improve consistency and scalability. Example of 3.0 systems include Hyperledger and Corda [24].

4. Blockchain layers

We propose a layered reference architecture for blockchains, encompassing all aspects related to blockchains from networking among peers and clients to applications (see Figure 3). We provide a description for each layer and associated challenges from top to bottom.

4.1. Application layer

The application layer focuses on developing blockchain solutions for use within and across specific applications and industries. Research directions include a methodology for translating application requirements from stakeholders into concrete specifications which are then satisfied by a tailored blockchain system. To do this, a comprehensive survey must be made of the application landscape, of blockchain "success" and "failure" stories, in order to better understand the challenges faced by blockchain researchers and developers. Feasibility studies should be conducted to determine the suitability of blockchain technologies and to elicit requirements from end-users in order to select the appropriate blockchain system design. These requirements also serve as motivations for research focusing on improving, extending, and possibly redesigning aspects of the blockchain in the lower layers. Finally, the development of application prototypes demonstrate the feasibility and benefits of using blockchains. Research challenges lie in determining which application characteristics will unequivocally determine the use of specific sets of blockchain features.

4.2. Modeling layer

In order to facilitate the generation of smart contracts, which are automatically executed to enforce application semantics, modeling approaches are required to express workflows. The best modeling languages (e.g., BPMN etc.) for blockchains must be identified via a thorough investigation of the technical layers. These models allow for existing and new applications to be expressed so as to permit blockchain integration. Extensions of existing models are also possible to be more conductive to blockchain deployment. In particular, security requirements and privacy policies should be incorporated such that they will be correctly reflected in the lower layers. Finally, there exists standardization efforts for modelling applications specifically designed to be executed with smart contracts [25].

4.3. Contract layer

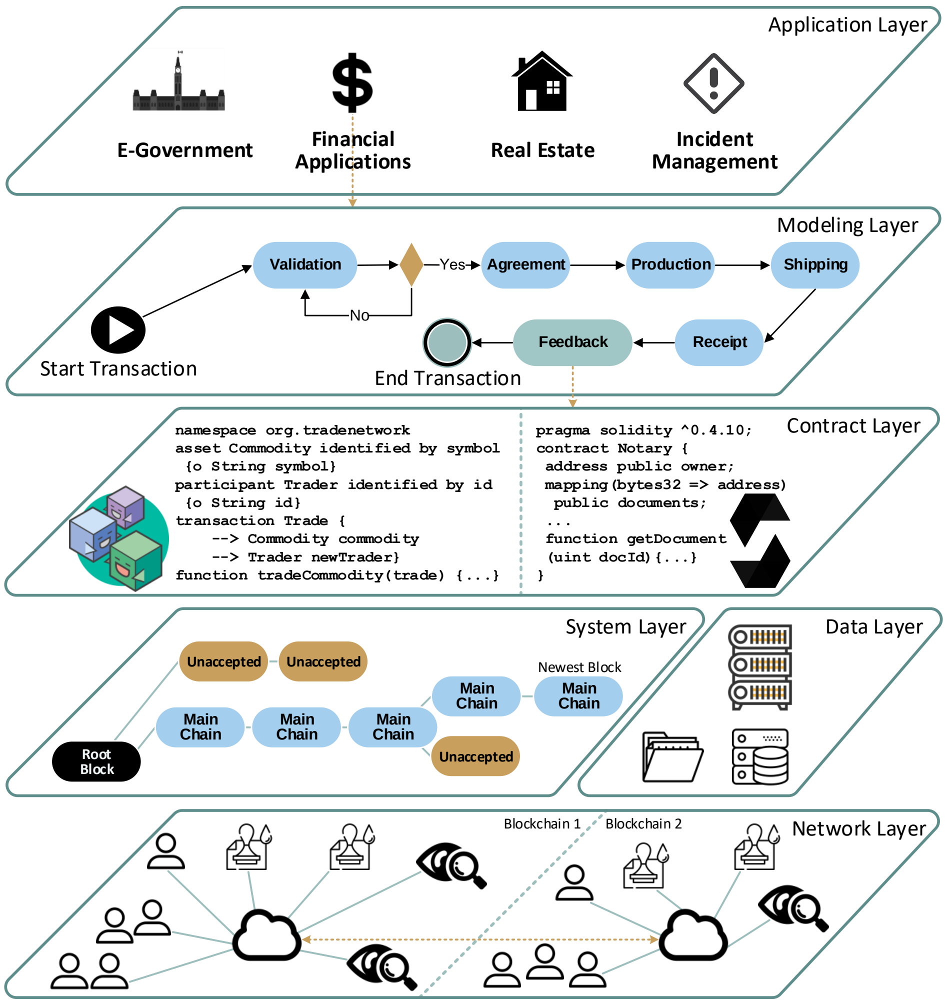
Smart contracts are still in early development. They must be made verifiable, reliable and secure before being employed at large scale. The main challenge is to generate smart contracts which are validated and tested prior to deployment in a live blockchain, as there are financial repercussions for incorrectly executed contracts. The contract language itself must be secure and devoid of potential weaknesses, to avoid attacks. Reusable services and middleware components can be expressed as smart contracts and integrated across industries.

4.4. System layer

The system layer consists of the core components, that is, the consensus protocol and other associated subsystems used to maintain the blockchain. Research challenges at this layer include the improvements of both proof-based and leader-based algorithms with respect to scalability and consistency. The system must also guarantee privacy in order to support applications with confidential information.

4.5. Data layer

The data layer refers to the management of information stored both on-chain (within blocks) and off-chain (within a database). Off-chain data storage is a recent development offered by systems like Ethereum and Hyperledger in order to reduce the amount of information stored in the blockchain, lowering the storage overhead required by the peers maintaining the blockchain (each one must keep a complete copy of the blockchain). The trade-off is that off-chain information is no longer durable or immutable. The decision to store data either on-chain or off-chain is challenging in terms of privacy and security as well as performance. A related challenge is the optimal placement of data either on- or off-chain.



```
namespace org.tradenetwork
asset Commodity identified by symbol
 {o String symbol}
participant Trader identified by id
 {o String id}
transaction Trade {
    --> Commodity commodity
    --> Trader newTrader}
function tradeCommodity(trade) {...}
```

```
pragma solidity ^0.4.10;
contract Notary {
 address public owner;
 mapping(bytes32 => address)
  public documents;
 ...
 function getDocument
 (uint docId){...}
}
```

Figure 3. Blockchain Stack

4.6. Network layer

Blockchains operate over networks, since peers must communicate to share information about the state of the blockchain. This feature of blockchains is often overlooked and not well described in existing approaches, however, we believe that it is crucial to investigate the network conditions and their impacts on the blockchain in terms of security and privacy and performance. Further effort is required to investigate how to best achieve cross-blockchain communication in the network in order to support the interoperation of different blockchains and cross-application interactions.

5. Main Research Challenges

In this section, we now outline a landscape for blockchain research which is classified across the following dimensions: DCS properties impacted, targeted applications, and related layers.

5.1. Applicability of blockchains

To improve the adoption of blockchains, case studies should be conducted in promising domains. A methodology should be designed for the investigation of use cases. We propose the following template for describing use cases:

1) Name
2) Intent: What is the problem solved?
3) Actors (users):
    - Who is sending transactions?
    - Who is creating the smart contracts?
    - Which actors are known and/or trusted, if any?
    - Who is maintaining the blockchains (verifying transactions, ledger)?
    - Who is only querying the data (vs. clients submitting transactions)?
4) Data objects:
    - What is stored on the blockchain?

- What is executed on the blockchain (i.e. smart contracts)?
- What kind of queries are served on the blockchain?

5) Relationships between actors and objects:
    - Which actor is exposed (e.g., creates, uses, etc.) to what object?
    - Which actor has what kind of permissions (executing contracts, writing, reading transactions)?
    - Which actor is liable for which object?
6) Performance requirements:
    - What is the expected number of actors of each type?
    - What is the expected throughput of transactions?
    - What is the expected latency?
    - What is the anticipated system growth?

Ultimately, the goal of this research angle is to clearly understand the benefits of employing blockchains, defining which applications benefit the most from it, and which platform is suitable for which applications.
DCS properties: This topic does not directly impact the DCS properties.
Applications: This topic mostly considers 3.0 applications, which are the least understood. However, 2.0 applications are also useful since they involve smart contracts. 1.0 applications are already popular, well-studied, and concern only a very narrow scope of possibel applications.
Related layers: This research aspect mostly focuses on the application layer, as well as the modeling layer, which can assist in the exploration process of new use cases. However, in-depth knowledge of the lower layers is necessary in order to understand the properties of various blockchain platforms and their relative suitability to the targeted use cases.

5.2. Blockchain middleware

A number of common elements are found across applications and domains. The development of reusable blockchain middleware will lead to more robust blockchain applications. Currently, the state-of-the-art in blockchain middleware is the development of cloud computing services which are directly integrated with blockchains, such as IBM BlueMix (using Hyperledger) [26], Microsoft Blockchain on Azure (using Ethereum) [27], and Deloitte Rubix (using a custom blockchain) [28]. These projects are providing Blockchain-as-a-Service (BaaS), by offering a managed blockchain platform to customers who do not need to deploy their own resources. We see these efforts as abstracting the lower layers of our vision (system/data layer and below). However, we argue that reusable components and services are required at the modelling and contract layers in order to simplify development and strengthen the reliability of the system. To date, we note one example currently being developed, the Application Blockchain Interface (ABCI), which allows applications to use the underlying blockchain system to tolerate failures by replicating the state across multiple machines [29].

We envision that blockchain middleware will be developed for the following services: messaging and event notification, identity management, data integration (especially with physical sources, e.g. sensors), and analytics.
DCS properties: This topic does not directly impact the DCS properties.
Applications: Blockchain 2.0 and 3.0 will benefit the most from blockchain middleware. In particular, 3.0 applications, which involve the physical world, all require a data integration service which takes into account the constraints of the physical world. For instance, real-life sensors can be tampered with or produce inaccurate readings, which must be taken into account when stored on the blockchain. We also envision 1.0 applications to require specialized blockchain services, such as off-chain payment networks [30] and cross-platform cryptocurrency exchanges [31].
Related layers: Blockchain middleware will mostly be implemented at the contract layer, as a smart contract. However, some middleware may be installed as pluggable subsystems at other layers.

5.3. Security and privacy

Security and privacy concerns are a barrier for the adoption of blockchains [32], [33]. In Bitcoin, while transactions are encrypted and accounts are anonymized, it is still possible to trace users based on their activity, which is fully exposed since every transaction is recorded and accessible on the blockchain [34]. As a result, Bitcoin is not a perfectly fungible system, since some coins might be linked to addresses known to be used for fraudulent activities [35]. “Clean” coins with little or no history are therefore worth slightly more than older coins which have been part of many transactions. Newer systems address these privacy concerns by introducing mixer networks to hide the transaction history [36].

Furthermore, the issue of privacy is further complicated in smart contract platforms, where it is desirable to keep the contract code confidential, yet still allow transactions to be validated on a public network. Cryptographic techniques such as Zero Knowledge Proofs (and particularly zk-SNARK) can be employed to accomplish this objective [4].

In some industrial use cases, encryption of data is not enough to satisfy privacy requirements. There is a need to explicitly guarantee that the information will not be stored outside of defined boundaries, for legal purposes. In these situations, the blockchain platform must support such privacy domains and yet still remain consistent. One such proposed approach is called multi-channel [37].

Finally, the security of smart contracts is an important factor which must be improved in order to facilitate their widespread use. There is a need to develop validation tools which can formally analyze smart contracts for bugs and incorrect behavior. New languages could also be developed

to improve readability of smart contract code, which can potentially be then directly reviewed by lawyers.
DCS properties: This topic mainly focuses on decentralization and consistency, by making blockchain systems more resilient to attacks. However, scalability will most likely be impacted by proposed techniques, especially if they are computationally intensive or incur additional latency (such as mixer networks).
Applications: This topic is important across generations, but on different aspects. 1.0 focuses on the privacy and traceability of transactions, 2.0 on the reliability of smart contracts, and 3.0 on data privacy.
Related layers: This topic requires research in the contract, system, and data layers. The network layer is also involved in the use of onion networks (as a mixer).

5.4. Scalable system innovations

Scalability is an important concern to ensure large-scale adoption of blockchain systems. At the moment, the main focus remains the consensus algorithm and to replace Proof-of-Work with scalable alternatives that are environmentally friendly, such as Proof-of-Stake [13].

Furthermore, the throughput of a blockchain is limited if it grows linearly and peers are forced to execute transactions serially. The blockchain also employs full replication of the entire system. The performance of the system can be improved by introducing parallelism, such as sharding [38] and side-chains [39].

Another possibility is to offload transactions outside the blockchain, as in the Lightning network [30].

As the number of users and smart contracts grow, so does the size of the state of the blockchain, which is stored in the main memory of peers. New data structures and data management solutions must be proposed and tested to ensure fast transaction validation and query response. Examples include the IAVL+ tree [40] and the Merkle Patricia tree. Furthermore, a more efficient protocol is needed to bootstrap new miners when they join the network without requiring a full download of the blockchain, since this data will continue to grow over time.

Finally, new technologies could be leveraged to improve the performance of the system. For instance, Hyperledger Sawtooth relies on Intel SGX to provide Proof-of-Elapsed-Time as its consensus algorithm [41].
DCS properties: Any improvement in scalability will impact either decentralization or consistency. Solutions which operate in a permissioned system do not offer full decentralization, while those in a permissionless system introduce branches which must be merged, or partition the data to be replicated across the network. In the latter, data availability issues might become a problem if there is no guarantee that smart contracts or blocks will be stored by any peer.
Applications: All three generations benefit from scalability, but they each have unique constraints which limit the applicability of proposed approaches. 1.0 systems must be incrementally maintained: any improvement is likely to be minimum, such as a re-parametrization of existing protocols (cf. Segwit2x for Bitcoin [42]). 2.0 systems are public cloud networks, and therefore must maintain a certain degree of decentralization. Only systems for 3.0 applications could potentially be built from scratch and benefit from the most optimal designs found.
Related layers: This topic heavily involves the system layer, as it revolves around the consensus protocol. The data layer will also be affected since it must also be made scalable.

6. Conclusions

Distributed ledger technology has an enormous potential to impact the world in a variety of applications. However, there is a clear need for fundamental research in a variety of aspects in order to ensure its widespread adoption as dependable, scalable, and pervasive systems.

In this paper, we provided a comprehensive structure for all aspects of blockchain research. We first presented the properties of Decentralization, Consistency, and Scalability (DCS) and demonstrated how blockchain systems must achieve the right balance of the three. We also classified blockchain applications under three generations, 1.0 (cryptocurrency), 2.0 (ÐApps), and 3.0 (pervasive apps). Our layered diagram shows the full structure of a blockchain stack and its different components: Application, Modeling, Contract, System, Data and Network layers. Finally, we presented a roadmap for blockchain research centered around four topics: applicability of blockchains, blockchain middleware, security and privacy, and scalable system innovations.

To summarize, we made the following observations:

1) Each generation of applications have unique challenges which must be addressed separately.
2) There exist a wide spectrum of research possible, involving every layer of the blockchain stack, which requires different fields of expertise, such as networking, data management, distributed systems, cryptography, and software engineering.
3) The DCS properties are governed by trade-offs, in a similar manner to the original CAP theorem. Therefore, the design of blockchain systems is largely driven by use cases, in order to achieve the right balance of properties required to satisfy the needs of the application. In the blockchain world, “one size does not fit all”!

Acknowledgments

This research is supported by the Alexander von Humboldt foundation.

References

[1] S. Nakamoto, “Bitcoin: A peer-to-peer electronic cash system, 2008,” *URL: http://www.bitcoin.org/bitcoin.pdf*, 2012.

[2] *Hyperledger Whitepaper*, http://www.the-blockchain.com/docs/Hyperledger%20Whitepaper.pdf, may 2017.

[3] G. Wood, "Ethereum: A secure decentralised generalised transaction ledger," 2017.

[4] A. Kosba, A. Miller, E. Shi, Z. Wen, and C. Papamanthou, "Hawk: The blockchain model of cryptography and privacy-preserving smart contracts," in *Security and Privacy (SP), 2016 IEEE Symposium on*. IEEE, 2016, pp. 839–858.

[5] E. Hildenbrandt, M. Saxena, X. Zhu, N. Rodrigues, P. Daian, D. Guth, and G. Rosu, "Kevm: A complete semantics of the ethereum virtual machine," Tech. Rep., 2017.

[6] N. Atzei, M. Bartoletti, and T. Cimoli, "A survey of attacks on ethereum smart contracts (sok)," in *International Conference on Principles of Security and Trust*. Springer, 2017, pp. 164–186.

[7] L. S. Sankar, M. Sindhu, and M. Sethumadhavan, "Survey of consensus protocols on blockchain applications," in *Advanced Computing and Communication Systems (ICACCS), 2017 4th International Conference on*. IEEE, 2017, pp. 1–5.

[8] I.-C. Lin and T.-C. Liao, "A survey of blockchain security issues and challenges," *IJ Network Security*, vol. 19, no. 5, pp. 653–659, 2017.

[9] A. Dorri, S. S. Kanhere, and R. Jurdak, "Blockchain in internet of things: challenges and solutions," *arXiv preprint arXiv:1608.05187*, 2016.

[10] K. Croman, C. Decker, I. Eyal, A. E. Gencer, A. Juels, A. Kosba, A. Miller, P. Saxena, E. Shi, E. G. Sirer *et al.*, "On scaling decentralized blockchains," in *International Conference on Financial Cryptography and Data Security*. Springer, 2016, pp. 106–125.

[11] J. Mendling, I. Weber, W. Van Der Aalst, J. v. Brocke, C. Cabanillas, F. Daniel, S. Debois, C. Di Ciccio, M. Dumas, S. Dustdar *et al.*, "Blockchains for business process management-challenges and opportunities," *arXiv preprint arXiv:1704.03610*, 2017.

[12] Z. Zheng, S. Xie, H.-N. Dai, and H. Wang, "Blockchain challenges and opportunities: A survey," *Work Pap.–2016*, 2016.

[13] S. King and S. Nadal, "Ppcoin: Peer-to-peer crypto-currency with proof-of-stake," *self-published paper, August*, vol. 19, 2012.

[14] I. Eyal, A. E. Gencer, E. G. Sirer, and R. Van Renesse, "Bitcoin-ng: A scalable blockchain protocol," in *13th USENIX Symposium on Networked Systems Design and Implementation (NSDI 16)*. USENIX Association, 2016, pp. 45–59.

[15] Y. Okada. (2017, nov) Helloworld.sol. [Online]. Available: https://github.com/parakeety/solidity-hello-world/blob/master/contracts/HelloWorld.sol

[16] E. A. Brewer, "Towards robust distributed systems," in *PODC*, vol. 7, 2000.

[17] Y. Sompolinsky and A. Zohar, "Secure high-rate transaction processing in bitcoin," in *International Conference on Financial Cryptography and Data Security*. Springer, 2015, pp. 507–527.

[18] A. Bessani, J. Sousa, and M. Vukolić, "A byzantine fault-tolerant ordering service for the hyperledger fabric blockchain platform," in *Proceedings of the 1st Workshop on Scalable and Resilient Infrastructures for Distributed Ledgers*. ACM, 2017, p. 6.

[19] M. Swan, *Blockchain: Blueprint for a new economy*. " O'Reilly Media, Inc.", 2015.

[20] K. Sedgwick. (2018, jan) New website ranks 600 cryptocurrencies by github activity. [Online]. Available: https://news.bitcoin.com/new-website-ranks-600-cryptocurrencies-github-activity/

[21] P. Rizzo, "Sweden tests blockchain smart contracts for land registry," *URL: http://www. coindesk. com/sweden-blockchain-smart-contracts-land-registry*, 2016.

[22] A. Ekblaw, A. Azaria, J. D. Halamka, and A. Lippman, "A case study for blockchain in healthcare:âĂIJmedrecâĂİ prototype for electronic health records and medical research data," in *Proceedings of IEEE Open & Big Data Conference*, vol. 13, 2016, p. 13.

[23] S. Underwood, "Blockchain beyond bitcoin," *Communications of the ACM*, vol. 59, no. 11, pp. 15–17, 2016.

[24] R. G. Brown, J. Carlyle, I. Grigg, and M. Hearn, "Corda: An introduction," *R3 CEV, August*, 2016.

[25] H. Wang, K. Chen, and D. Xu, "A maturity model for blockchain adoption," *Financial Innovation*, vol. 2, no. 1, p. 12, 2016.

[26] IBM, "IBM blockchain," https://www.ibm.com/blockchain/offerings.html, 2018.

[27] Microsoft, "Azure solutions," https://azure.microsoft.com/en-us/solutions/blockchain/, 2018.

[28] Rubix, "Blockchain-powered enterprise and government solution," http://rubixbydeloitte.com/, 2018.

[29] Tendermint, "Serverside blockchain API," https://github.com/tendermint/abci, 2018.

[30] J. Poon and T. Dryja, "The bitcoin lightning network: Scalable off-chain instant payments. 2016," *URl: https://lightning. network/lightningnetwork-paper. pdf (visited on 2016-04-19)*, 2015.

[31] M. Herlihy, "Atomic cross-chain swaps," *arXiv preprint arXiv:1801.09515*, 2018.

[32] V. Buterin. (2016, jan) Privacy on the blockchain. [Online]. Available: https://blog.ethereum.org/2016/01/15/privacy-on-the-blockchain/

[33] C. Wilkins and G. Gaetz. (2017, may) Could dlt underpin an entire wholesale payment system? [Online]. Available: https://www.theglobeandmail.com/report-on-business/rob-commentary/could-dlt-underpin-an-entire-wholesale-payment-system/article35106771/

[34] E. Androulaki, G. O. Karame, M. Roeschlin, T. Scherer, and S. Capkun, "Evaluating user privacy in bitcoin," in *International Conference on Financial Cryptography and Data Security*. Springer, 2013, pp. 34–51.

[35] R. Pérez-Marco, "Bitcoin and decentralized trust protocols," *arXiv preprint arXiv:1601.05254*, 2016.

[36] D. Hopwood, S. Bowe, T. Hornby, and N. Wilcox, "Zcash protocol specification," Tech. rep. 2016-1.10. Zerocoin Electric Coin Company, Tech. Rep., 2016.

[37] C. Cachin, "Architecture of the hyperledger blockchain fabric," in *Workshop on Distributed Cryptocurrencies and Consensus Ledgers*, 2016.

[38] J. Poon and V. Buterin, "Plasma: Scalable autonomous smart contracts," *White paper*, 2017.

[39] W. Li, A. Sforzin, S. Fedorov, and G. O. Karame, "Towards scalable and private industrial blockchains," in *Proceedings of the ACM Workshop on Blockchain, Cryptocurrencies and Contracts*. ACM, 2017, pp. 9–14.

[40] Tendermint, "Iavl+ trees," https://github.com/tendermint/tendermint/wiki/Merkle-Trees, 2018.

[41] L. Chen, L. Xu, N. Shah, Z. Gao, Y. Lu, and W. Shi, "On security analysis of proof-of-elapsed-time (poet)," in *International Symposium on Stabilization, Safety, and Security of Distributed Systems*. Springer, 2017, pp. 282–297.

[42] Segwit2x, "B2x," https://b2x-segwit.io, 2018.