Do you need a Blockchain?

Karl Wüst*, Arthur Gervais†
*karl.wuest@inf.ethz.ch, †arthur.gervais@inf.ethz.ch
Department of Computer Science
ETH Zurich, Switzerland

Abstract—Blockchain is being praised as a technological innovation which allows to revolutionize how society trades and interacts. This reputation is in particular attributable to its properties of allowing mutually mistrusting entities to exchange financial value and interact without relying on a trusted third party. A blockchain moreover provides an integrity protected data storage and allows to provide process transparency.

In this article we critically analyze whether a blockchain is indeed the appropriate technical solution for a particular application scenario. We differentiate between permissionless (e.g., Bitcoin/Ethereum) and permissioned (e.g. Hyperledger/Corda) blockchains and contrast their properties to those of a centrally managed database. We provide a structured methodology to determine the appropriate technical solution to solve a particular application problem. Given our methodology, we analyze in depth three use cases — Supply Chain Management, Interbank and International Payments, and Decentralized Autonomous Organizations and conclude the article with an outlook for further opportunities.

1 Introduction

Bitcoin and its blockchain have allowed mutually mistrusting entities to perform financial payments without relying on a central trusted third party while offering a transparent and integrity protected data storage [1]. Due to these properties, blockchain as a technology has gained much attention beyond the purpose of financial transactions – distributed cloud storage, smart property, Internet of Things, supply chain management, healthcare, ownership and royalty distribution, and decentralized autonomous organizations just to name a few. Contrary to Bitcoin's *permissionless* blockchain, where any writer and reader can join at any time, so-called *permissioned* blockchains have been proposed, where only an authorized set of entities is allowed to write and read the respective blockchain. A permissioned blockchain, however, shares similarities with a centralized database, and this naturally brings up the question whether a blockchain is better suited than a centralized database.

In this work, we analyze the properties of different blockchain types (i.e. permissioned and permissionless) and contrast these properties to those of a centrally managed database. We provide a methodology to identify whether a blockchain is useful depending on the problem requirements, and if so, what type of blockchain might be appropriate. Based on our methodology, we evaluate in detail three use cases, namely *(i)* supply chain management, *(ii)* interbank and international payments and *(iii)* decentralized autonomous organizations and argue if and which blockchain type make sense for the specific applications.

The remainder of this article is organized as follows. In Section 2, we briefly describe the most important background about blockchain. In Section 3 we provide a structured methodology to identify if a blockchain makes sense, and if yes, which type of blockchain would be appropriate. Based on our methodology, we analyze proposed use cases in detail in Section 4. In Section 5, we review related work in the area, and we conclude the article in Section 6.

2 Background on Blockchain

In the following section, we detail the required blockchain background and the involved parties. The name blockchain stems from its technical structure — a chain of blocks. Each block is linked to the previous block with a cryptographic hash. A block is a datastructure which allows to store a list of transactions. Transactions are created and exchanged by peers of the blockchain network and modify the state of the blockchain. As such, transactions can exchange monetary amounts, but are not restricted to financial transactions only and for example allow to execute arbitrary code within so-called smart contracts.

Before diving into the specific differences of permissionless and permissioned blockchains, we now describe the different participants of these networks. As applicable to any database system, we denote as *writer* any entity which writes state to the database. In a blockchain this would correspond to a participant that is involved in the consensus protocol and helps growing the blockchain. As such, a writer is able to accumulate transactions within a block and append this block to the blockchain. Related work might also denominate a writer as a validator. We denote a *reader* as any entity which is not extending the blockchain, but participating in either the transaction creation process, simply reading and analysing or auditing the blockchain. Note that we consider regulators and blockchain software maintainers to be outside of this scope.

Permissionless Blockchains Bitcoin [1] and Ethereum [2] are instances of permissionless blockchains, which are open and decentralized. Any peer can join and leave the network as reader and writer at any time. Interestingly, there is no central entity which manages the membership, or which could ban illegitimate readers or writers. This openness implies that the written content is readable by any peer. With the use of cryptographic primitives however, it is technically feasible to design a permissionless blockchain which hides privacy relevant information (e.g. Zerocash [3]).

Permissioned Blockchains To only authorize a limited set of readers and writers, so called-permissioned blockchains have been recently proposed. Here, a central entity decides and attributes the right to individual peers to participate in the write or read operations of the blockchain. To provide encapsulation and privacy, reader and writer could also run in separated parallel blockchains that are interconnected. The most widely known instance of permissioned blockchains are Hyperledger Fabric and R3 Corda [4].

2.1 Properties

In the following, we describe and compare the most relevant properties that distributed ledgers and centralized systems provide.

Public Verifiability allows anyone to verify the correctness of the state of the system. In a distributed ledger, each state transition is confirmed by verifiers (e.g. miners in Bitcoin), which can be a restricted set of participants. Any observer, however, can verify that the state of the ledger was changed according to the protocol and all observers will eventually have the same view of the ledger, at least up to a certain length. In a centralized system, different observers may have entirely different views of the state. As such, they might not be able to verify that all state transitions were executed correctly. Instead, observers need to trust the central entity to provide them with the correct state.

Transparency of the data and the process of updating the state is a requirement for public verifiability. The amount of information that is transparent to an observer, however, can differ, and not every participant needs to have access to every piece of information.

Privacy is an important property of any system. There exists an inherent tension between privacy and transparency. Privacy is certainly easier to achieve in a centralized system because transparency and public verifiability are not required for the functioning of the system.

Integrity of information ensures that information is protected from unauthorized modifications, i.e. that retrieved data is correct. The integrity of information is closely linked to public verifiability. If a system provides public verifiability, anyone can verify the integrity of the data; integrity can otherwise only be ensured if the centralized system is not compromised.

Redundancy of data is important for many use cases. In blockchain systems, redundancy is inherently provided through replication across the writers. In centralized systems, redundancy is generally achieved through replication on different physical servers and through backups.

Trust Anchor defines who represents the highest authority of a given system that has the authority to grant and revoke read and write access to a system.

2.2 Tensions between Transparency and Privacy

There exist an inherent tradeoff between transparency and privacy. A fully transparent system allows anyone to see any piece of information, i.e. no privacy is provided. Likewise, a fully private system provides no transparency. However, a system can still provide significant privacy-guarantees while making the process of state transitions transparent, e.g. a distributed ledger can provide public verifiability of its overall state without leaking information about the state of each individual participant. Privacy in a public system can be achieved using cryptographic techniques but typically comes at the cost of lower efficiency. The cryptocurrency Zerocash [3] for example makes use of computationally expensive cryptography to provide full anonymity while still providing sufficient transparency to publically verify the ledger state.

3 Where does a Blockchain make sense?

In general, using an open or permissioned Blockchain only makes sense when multiple mutually mistrusting entities want to interact and change the state of a system, and are not willing to agree on an online trusted third party.

To ease the decision making process, we provide a flow chart in Figure 1. We consider one or multiple parties that write the system state, i.e. a writer corresponds to an entity with write access in a typical database system or to consensus participant in a blockchain system.

If no data needs to be stored, no database is required at all, i.e. a blockchain, as a form of database, is of no use. Similarly, if only one writer exists, a blockchain does not provide additional guarantees and a regular database is better suited, because it provides better performance in terms of throughput and latency. If a trusted third party (TTP) is available, there are two options. First, if the TTP is always online, write operations can be delegated to it and it can function as verifier for state transitions. Second, if the TTP is usually offline, it can function as a certificate authority in the setting of a permissioned blockchain, i.e. where all writers of the system are known. If the writers all mutually trust each other, i.e. they assume that no participant is malicious, a database with shared write access is likely the best solution. If they do not trust each other, using a permissioned blockchain makes sense. Depending on whether public verifiability is required, anyone can be allowed to read the state (public permissioned blockchain) or the set of readers may also be restricted (private permissioned blockchain). If the set of writers is not fixed and known to the participants, as is the case for many cryptocurrencies such as Bitcoin, an permissionless blockchain is a suitable solution.

In Table 1 we contrast some properties of permissionless and permissioned blockchains, and a central database. In a centralized systems, the performance in terms of latency and throughput is generally much better than in blockchain systems, as blockchains add additional complexity through their consensus mechanism. For example, Bitcoin can currently only sustain a throughput of approximately seven transactions per second (which could be extended to approximately 66 without compromising security [5]), while a centralized system such as Visa can handle peaks of more than fifty thousand transactions. There is a tradeoff between decentralization, i.e. how well a system scales to a large number of writers without mutual trust, and throughput, i.e. how many state updates a system can handle in a given amount of time. When making the decision of whether to use a blockchain system or not, this tradeoff should be taken into account as well.

4 Case by Case

In the following Section, we outline several use cases where industrial efforts are advertising to use blockchain technology. Where possible, we evaluate objectively how a blockchain solution might make sense and what the technical, security and privacy implications would be.

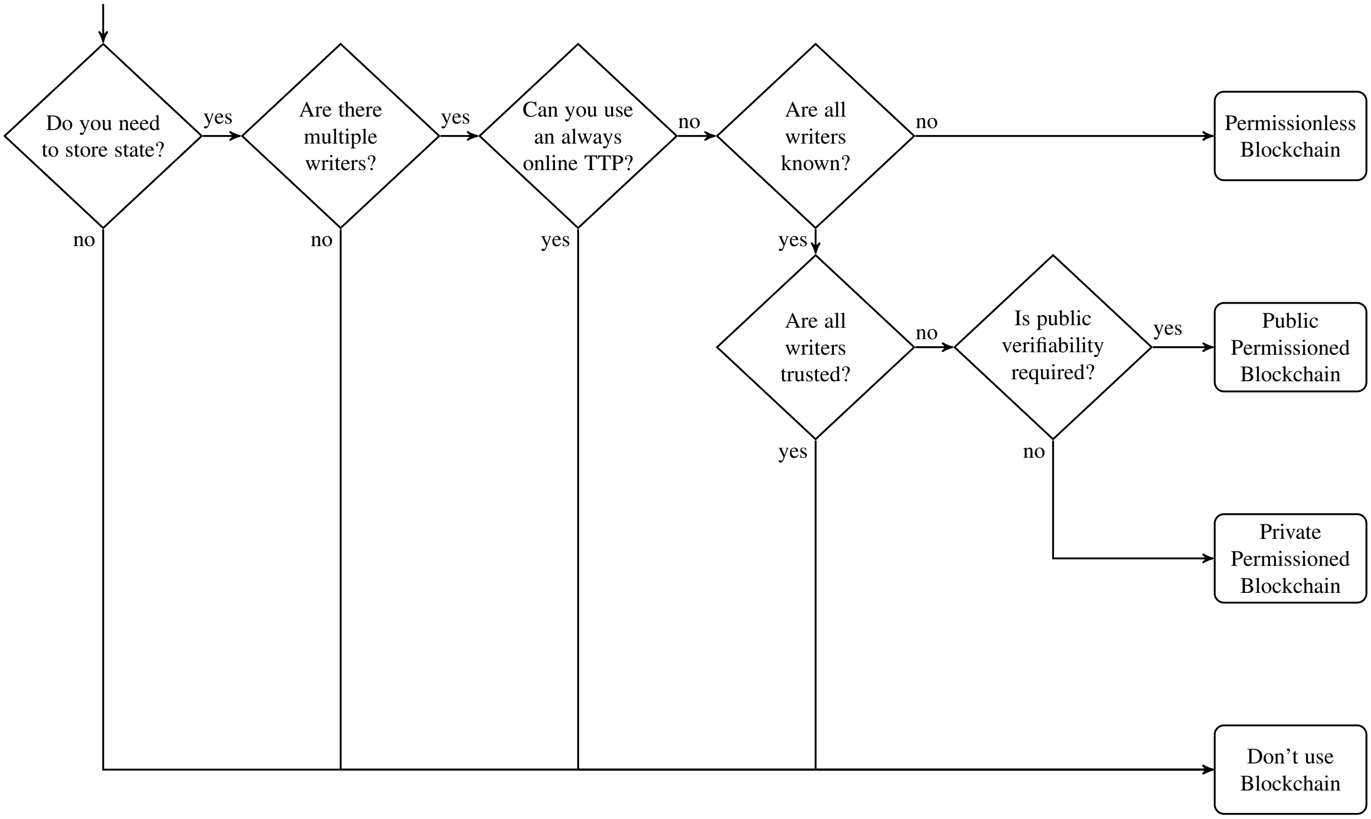

Do you need to store state?
yes
no
Are there multiple writers?
yes
no
Can you use an always online TTP?
no
yes
Are all writers known?
no
yes
Permissionless Blockchain
Are all writers trusted?
no
yes
Is public verifiability required?
yes
no
Public Permissioned Blockchain
Private Permissioned Blockchain
Don't use Blockchain

Fig. 1: A flow chart to determine whether a blockchain is the appropriate technical solution to solve a problem (Table 1 should be considered in the decision making process as well). Writers refer to entities with write access to the database/blockchain, i.e. in a blockchain setting, a writer corresponds to a consensus participant. If a trusted third party (TTP) is available that is not always online, this can be used to establish a known group of writers, i.e. the TTP can function as a certificate authority in such a setting. Public and private permissioned blockchains differ in that a public blockchain allows anyone to read the contents of the chain and thus verify the validity of the stored data, while a private blockchain only allows a limited number of participants to read the chain. Note that for any blockchain based solution it is possible to make use of cryptographic primitives in order to hide privacy-relevant content.

	Permissionless Blockchain	Permissioned Blockchain	Central Database
Throughput	Low	High	Very High
Latency	Slow	Medium	Fast
Number of readers	High	High	High
Number of writers	High	Low	High
Number of untrusted writers	High	Low	0
Consensus mechanism	Mainly PoW, some PoS	BFT protocols (e.g. PBFT [6])	None
Centrally managed	No	Yes	Yes

TABLE 1: We differentiate between permissionless, permissioned blockchains and a centralized database. Note that a permissioned blockchain can be public, for example if public verifiability of the content is desired.

4.1 Supply Chain Management

In Supply Chain Management (SCM), the flow of materials and services required in manufacturing a given product is managed, which includes various intermediate storage and production cycles until the delivery to the final point of consumption. Typically, multiple companies interact and trade on a global scale within a given supply chain. Due to this complexity, associated costs of managing the inventory, processes and failure detection are particularly expensive.

Several companies (e.g. Skuchain, Provenance, Walmart, Everledger) advertise to provide blockchain based solutions to improve the efficiency of supply chain management solutions. Some even claim that blockchain technology paves the way to *demand* instead of *supply* chains, where businesses will benefit from a greater flexibility in interacting with different markets and balancing the price risks.

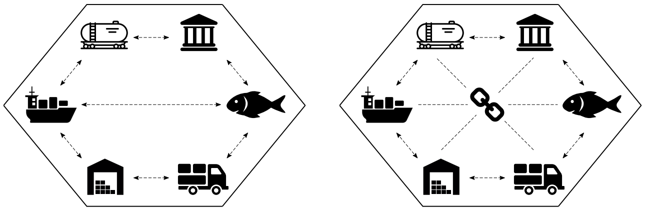
Traditional SCM is driven by planning and communication. The future demand is estimated based on the past and current demand, information is pushed to the involved stakeholders that hope to get the relevant information on time to respond to changes, delays or errors. Companies decide what product is released to the market at what time, and customers indirectly drive the demand.

In demand chain management (DCM), the customer's interest is at the core of the chain — reduced costs, performant customer service, and faster go-to-market from idea or minimum viable product (MVP), just to name a few examples. DCM allows for this increased flexibility by requiring all stakeholders to have a real-time visibility of what consumers want and purchase. All parties of the demand chain have therefore to be tightly connected within a network. Contrary to SCM, which "optimizes the flow" and might be based on incomplete and inaccurate market assessements, DCM requires companies to have a complete and accurate view of the market to proactively choose optimal production decisions. As such, the information flow in DCM's is *pull* based rather than *push* based: the stake holders do not need to wait for a notification, but can actively query the state of the chain management.

While SCM solutions certainly can and should be improved, it is unclear why blockchain in particular is a suitable technical solution. Skuchain for instance (cf. Figure 2) relies on IBM's Hyperledger Fabric as blockchain backend. Fabric's pluggable consensus options allow for a wide range of flexibility on how many nodes are actually taking part in the consensus process. Skuchain acknowledged (upon request in private correspondance) that for most supply chain management features a single source of truth would be sufficient — as such a single trusted database at Skuchain should be sufficient to satisfy most business needs.

Provenance aims to provide another blockchain based solution for more transparency in product supply chains. Provenance does not provide any details on their technical product but claims that *data can be accessed and verified by all actors*. Even if Provenance manages to hide the actor's identity (as claimed in the whitepaper), such data would leak a considerable amount of business critical information from the different actors — e.g., production volume and times.

Everledger has digitally certified over 1 mio. diamonds and records every diamond permanently in the Everledger blockchain to provide a clear audit trail for stakeholders. While Everledger does not provide technical details on their solution, Everledger claims to use a hybrid model between a public and a private blockchain to benefit from the permissioned controls in private blockchains.



(a) Traditional SCM. (b) Blockchain powered SCM.

Fig. 2: Traditional SCM (left) compared to blockchain-based Supply Chain Management (right). Traditional SCM is distributed, i.e. there is no central entity. A blockchain powered SCM maintains a distributed ledger where participant can update and read (pull) the current SCM state.

4.1.1 Outlook

The participants of a SCM vary greatly across different supply chains and the same peers might take different roles across different supply chains. The segmentation basis for different actors in the supply chain is typically defined by their respective ownership stake of the product that is being produced. This implies that a single blockchain would be required for every supply chain that a participant is involved in — which clearly deteriorates the performance of the final solution.

Following our methodology from Section 3, a SCM certainly requires to store data. Multiple writers are involved, i.e. the different participants of the SCM that own a certain share of the final product. Skuchain acknowledged to only require a single source of trust, which would however remove the decentralized component of the blockchain, and thus be equivalent to a trusted central server. Continuing our methodology, a SCM could technically likely always use an online TTP. If that is not possible, at least all writers will be known, which leaves us to choose between a permissioned or no blockchain.

This reasoning leaves us with the question whether all writers can be trusted. Supply chain management has the inherent problem of the interface between the digital and the physical world. A human, or some machine under the control of a single writer, typically is required to register that a certain good has arrived in a warehouse, and if for example its quality is appropriate. If there is no trust in the operation of these employees, then the whole supply chain is technically compromised as any data can be supplied by a malicious writer. If, on the other hand, all writers are trusted, a blockchain is not needed as a regular database with shared write access can be used instead. Note that if through some technical means, the connection between the digital and physical world could be realized in a secure manner, then the previous reasoning might change.

4.2 Interbank and International Payments

In this Section, we outline how interbank and international payments are currently performed in the banking system. In addition, we describe solutions based on distributed ledger technology that aim to simplify and replace the current system. Based on this understanding we explain the benefits and drawbacks of using distributed ledger technology to simplify interbank payments.

4.2.1 The Legacy System

Traditionally, in the current banking system, a transaction transferring money from an account at bank $\mathcal{A}$ to an account at bank $\mathcal{B}$ takes multiple steps. Contrary to cash transfers, debts in bank transfers are typically not immediately settled.

If Alice wants to transfer \$100 to Bob, Alice's account is debited with \$100 and Bob's account should be credited with the same amount. If the accounts are at the same bank, the bank can simply apply these changes to their books because the total debit and credit amount of the bank remains identical. If Alice however, has her account at bank $\mathcal{A}$ and Bob at bank $\mathcal{B}$, the total debit of bank $\mathcal{A}$ changes when debiting Alice's account. Similarly, if Bank $\mathcal{B}$ credits Bob's account without debiting another account with the same value, the sum of all debits and all credits at Bank $\mathcal{B}$ would no longer be equal. This can be solved, if each of the banks have an account with the other bank (commonly referred to as a *Nostro* account). Then, bank $\mathcal{A}$ could debit Alice's account and credit $\mathcal{B}$'s account while bank $\mathcal{B}$ would debit $\mathcal{A}$'s account and credit Bob's account while modifying the respective Nostro account.

In practice this would lead to large debts between banks which brings a large amount of risk. Banks therefore have accounts at a central bank, which is mirrored in a local account (*mirror account*) at the bank for bookkeeping, where they credit and debit the central bank. I.e., bank $\mathcal{A}$ debits Alice's account, informs the central bank of the payment and credits the mirror of their account at the central bank, the central bank debits the account of bank $\mathcal{A}$, credits bank $\mathcal{B}$'s account and informs $\mathcal{B}$ of the payment, who then debits their central bank mirror account and credits Bob's account. The central banks are used as settlement authorities for the payments in the currency for which they are responsible,

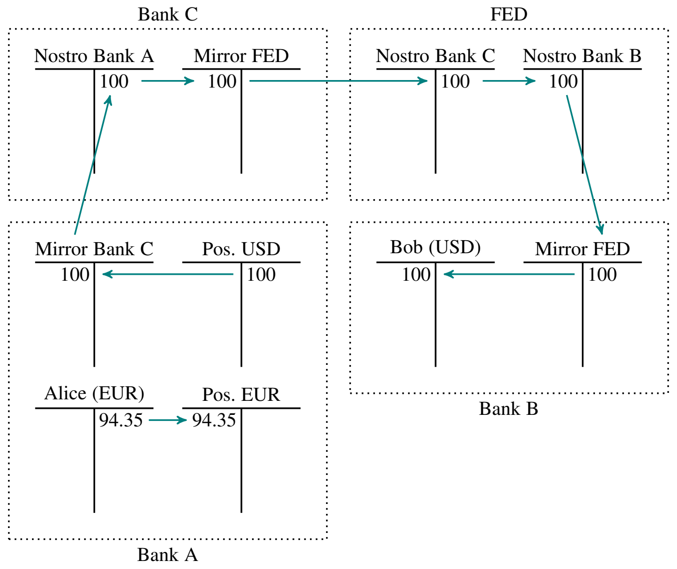


Fig. 3: The logical flow of money and accounting steps involved in a traditional international payment in which Alice from Europe pays USD 100 to Bob in the USA. At Bank A, Alice implicitly buys USD with EUR, i.e. her account gets debited while the banks Euro account is credited with the same amount (EUR 94.35). The banks USD account then gets debited with the bought USD, which are credited to the mirror account of Bank C, who then debit Bank A's Nostro account (i.e. the account that Bank A holds at Bank C) and credit the mirror account of the US central bank (FED). The FED then debits Bank C's Nostro account and credits Bank B, who then debits the central banks mirror account and finally credit Bob's account with the intended amount. Note that in this simplified example, fees that would occur in practice at intermediate steps are not shown.

since they are trusted to fulfill their debts (by issuing money if necessary).

Already, three banks are involved for a single payment and in practice, additional parties such as clearing houses take part, such that low value payments can be batched and the central bank does not need to be involved in every interbank payment. For international (i.e., inter currency) payments, even more parties need to be involved, e.g., if Alice has a Euro account at bank $\mathcal{A}$ located in the EU and bank $\mathcal{B}$ is located in the USA. For cross currency payments, there is no single central bank that is able to settle the payments and bank $\mathcal{A}$ does not have an account with the US central bank.

Instead, bank $\mathcal{A}$ has a USD account at some commercial bank $\mathcal{C}$ in the USA, which we assume to be distinct from $\mathcal{B}$ for this example. This bank $\mathcal{C}$ is called $\mathcal{A}$'s correspondent bank. This requires a trust relationship between banks $\mathcal{A}$ and $\mathcal{C}$. In our example, some amount of Euro is debited in Alice's account with which USD is implicitly bought by Alice at bank $\mathcal{A}$, i.e., $\mathcal{A}$'s Euro position increases while the USD position decreases by $100. The $100 are credited to the mirror account for $\mathcal{A}$'s account at Bank $\mathcal{C}$. Bank $\mathcal{C}$ then debits $\mathcal{A}$'s account at their bank and transfers the money to bank $\mathcal{B}$ using the US central bank (FED) for the settlement. This money transfer is depicted in Figure 3.

For money transfers in currencies for which a bank does not have a correspondent bank, additional intermediate hops may be required which adds complexity, more delays and as a consequence higher costs.

Overall, the main drawbacks of the correspondent banking system are the long transaction confirmation time, the cost caused by the multiple intermediate hops and the trust that is required between the banks in order for the system to work.

4.2.2 Distributed Ledger Technology for Interbank Payments

Due to the high costs entailed by the correspondent banking system, many put their hopes into distributed ledger technology to simplify interbank payments. Some central banks such as the Monetary Authority of Singapore (MAS) and the Bank of Canada are working on solutions to use distributed ledger technology for interbank payments [7], [8]. In the solution of the MAS, banks deposit some amount of money with the MAS and in return receive the same amount on the distributed ledger. The ledger can then be used to immediately transfer money between the banks. While this does not allow cross currency transfers, it simplifies interbank payments within a single currency and is a first step towards replacing the payment system.

Similarly, companies such as SWIFT [9] and Visa started to develop proof of concepts for international payments using blockchain technology. While these proof of concepts are not yet public and very little information about them is available, other solutions using distributed ledgers that aim to simplify cross-border payments are already more developed.

Ripple aims to provide a global settlement network based on a distributed ledger. Ripple only partially replaces the correspondent banking system. Banks can continue to use correspondent banks to process payments in cases where liquidity in the required foreign currency is available at low rates. Otherwise, banks can use third party liquidity providers to provide the required liquidity. Similar to the traditional correspondent banking system, a payment may require multiple hops if no trust relationship exists between the two banks that are parties in the transaction. Contrary to the traditional system, the payment is atomic, i.e., either all of the intermediate payments go through or none of them. In the traditional system, if something goes wrong for an intermediate payment, previous payments have to be reversed and sometimes manual intervention is required. Additionally, Ripple provides its own currency, XRP, which can be used as intermediate currency for transactions.

XRP is the only currency on the Ripple ledger for which transactions do not entail counterparty risk. Other currencies are "issued" by gateways that need to be trusted to settle the owed debts outside of the distributed ledger if a party chooses to withdraw a deposit. This means, for example, that not all USD have the same issuer and they are not backed by the central bank, i.e., an on-chain US Dollar is not a real US Dollar and, de facto, every issuer creates a new parallel currency. Because of this gateway system, Ripple does not remove the trust relationships required in the correspondent banking system but simply shifts them to other parties, the gateways.

This limitation could be removed if such a system would use central banks to act as gateways, since the currencies issued on Ripple would then actually correspond to the real currencies. This would remove all trust requirements for settlement other than the trust in the central banks, which is a necessity in any case when transacting in the corresponding currency.

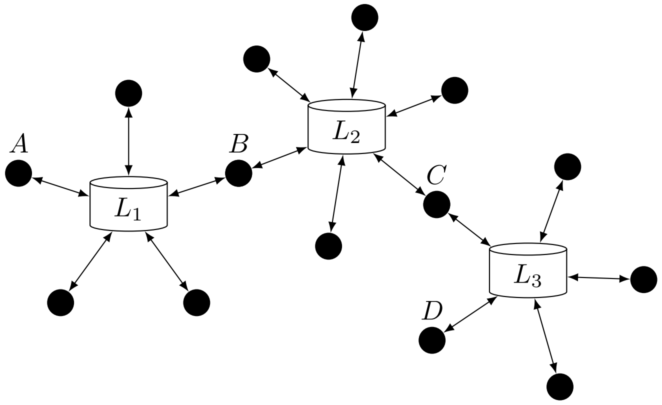


Fig. 4: Three individual ledgers L_1, L_2, and L_3 that are connected through nodes B and C, i.e. node B participates as writer in L_1 and L_2 and node C participates in both L_2 and L_3. If each of these ledgers is a blockchain for one currency, a payment from A to D can be routed through B and C as atomic transaction, where B and C provide currency exchange. This can be achieved for example through hashed timelock contracts [10].

4.2.3 Outlook

For financial applications, blockchain technology seems well suited in general, since parties are generally risk averse and do not want to rely on strong trust assumptions. We can evaluate the usefulness of blockchain technology for a given system with our methodology from Section 3. If we consider a system for interbank payments, we have multiple parties, the banks, that act as writers. If we only consider single currency systems, we do have a trusted third party, the central bank. The central bank may, however, not want to act as a verifier for every transaction and may only act as a certificate authority giving out licenses to banks to participate in the system. This means that all writers of the system are known and we can use a permissioned blockchain. Whether the chain should be publicly verifiable is a matter of opinion, i.e. the blockchain can either be public or private. On one hand, banks likely want to keep their monetary flows private, on the other hand, having public verifiability may increase the trust of the public in the monetary system. As mentioned in Section 2.2, this tension between transparency and privacy can be resolved at the cost of efficiency by using cryptographic techniques to provide privacy while also ensuring public verifiability.

While current systems (such as Ripple) are not yet able to provide trustless intercurrency money transfers, the future development in this area looks promising. Many central banks currently research the possibilities of using blockchain technology for interbank payments and with centrally issued on-chain currency, the value is defined by the actual value of the currency and thus interchangeable.

If countries collaborate in designing their blockchains for interbank payments, they can be designed in a way that allow interaction between chains, e.g. to provide atomic cross currency payments as shown in Figure 4. This can be done using techniques that are also used in off-chain payment networks such as hashed timelock contracts [10] or by instantiating the blockchains as satellite chains [11]. In such a system, banks that have accounts on multiple chains can be used to exchange currency and route payments atomically internationally while removing the trust requirements of the correspondent banking system.

4.3 Decentralized Autonomous Organizations

A *Decentralized Autonomous Organizations* (DAO) is an organization that is run autonomously through a set of smart contracts. In contrast to traditional organizations or companies, there is no central control or management. Instead, a DAO is defined by a set of rules encoded in smart contracts that define how the DAO behaves and how it evolves. Typically, a DAO has many investors that then decide by voting how the funds of the DAO should be invested. As the goal of such an organization is to be governed in a completely decentralized way and the investors generally don't know or trust each other, a permissionless blockchain is naturally a good fit for such a design: The system is required to store some state and multiple mutually distrusting and possibly unknown writers exist.

Decentralized autonomous organisations are, however, a special case. For some applications a dedicated permissioned blockchain may be useful for a single DAO. In most cases, however, DAOs do not require their own blockchain but are instead better suited to be build on top of an existing blockchain with an already existing currency (such as Ethereum [2]).

4.4 Other use cases

In the following section we discuss other use cases that have been suggested for blockchain technology.

4.4.1 Proof of Ownership

Proof of Ownership for intellectual property is an often proposed and straightforward use case for blockchains. If the creator of some digital object wants to prove ownership at a later time, he can use a public blockchain as a time stamping service by committing to the digital object together with his identity, e.g. with a hash, and publishing that commitment on the blockchain. This allows to later prove that the object existed at that time and was associated with the respective identity. While this does not fully prove ownership, it does provide evidence of ownership if no one else can show that the object was previously published. Instead of using a blockchain, a trusted third party could provide a proof of ownership, e.g. a patent office. A public blockchain, however, eases the process of providing a proof in a decentralized way and without disclosing details of the digital object.

4.4.2 E-Voting

E-Voting is a problem with many difficulties. Many of the desired e-Voting properties have trade-offs. On one hand, for example, privacy is a main requirement as votes should be anonymous to prevent coercion. On the other hand, e-voting should provide some sort of public verifiability, because otherwise, the provider of the e-voting solution – or someone who managed to compromise it – might be able to change votes at will. In e-voting, many parties are involved and these parties typically do not trust each other. At the same time, e-voting systems require public verifiability, and thus, many have proposed to base e-voting systems on blockchain technology. Due to the requirements, it seems reasonable that blockchain technology can help to achieve some of the desired properties. However, to the best of our knowledge, so far no solution has been proposed that has been shown to be secure, verifiable, and private and there are still many open challenges.

4.4.3 Smart Contracts

Smart Contracts [12] are digital contracts that are self enforcing or make it prohibitively expensive to break contract. Since a blockchain can be used as a distributed state machine without a trusted third party, the technology is well suited to support smart contracts. While Bitcoin already supports a limited set of smart contracts, Ethereum [2] was the first blockchain to support arbitrary code execution on the blockchain, allowing any kind of smart contract.

Since contract partners do not usually fully trust each other, blockchain technology is suitable for this application if the parties do not want to rely on a trusted third party, because it can simplify trustless protocols between multiple parties. Depending on the setting and the requirements, a permissionless blockchain or a permissioned blockchain can be used.

Because practical smart contracts are relatively new technology, it is not yet clear to what extent these are legally binding.

4.4.4 Internet of Things

Many have suggested possible use cases for blockchain technology in the Internet of Things (IoT) in combination with smart contracts with the aim to provide autonomous systems that pay for resources that they consume and get paid for resources that they provide. As the system is inherently decentralized with entities that do not trust each other, using a blockchain seems natural. However, as with supply chain management (cf. Section 4.1) the interface between the physical and the digital world poses a potential problem. If computers supply values that were read from sensors to the blockchain, the blockchain does not guarantee the correctness of these values, i.e. if smart contracts behave according to values supplied by sensors, the sensors – and whoever controls them – necessarily need to be trusted. For many cases, if e.g. only automation is desired, a blockchain may not be necessary if a trusted party can be used instead. In other cases, the specific trust assumption have to be studied and evaluated carefully to determine whether the use of a blockchain provides additional value.

4.4.5 Trading and Fair Exchange Protocols

Fair multi-party exchange protocols have been extensively studied in the literature. Due to the recent emergence of open and decentralized blockchains (e.g. Bitcoin and Ethereum), however, the design of fair exchange protocols has recently experienced a renaissance. The exchange of digital goods is likely to be feasible without trusted dispute mediator [13], while the exchange of physical goods still requires a trusted third party in case of disputes [14].

5 Related Work

Bitcoin [1], as the first open and decentralized blockchain, initiated a large development in the area. Other permissionless blockchains such as Zerocash [3] or Ethereum [2] build on the techniques used by Bitcoin and extend the possibilities through improved privacy or more expressive smart contracts. Other extensions such as hashed timelock contracts that are e.g. used in the lightning network [10] can be used to improve the throughput of blockchains or to allow transfers of digital assets between different blockchains.

Through the emergence of Bitcoin, many companies now develop their own permissioned blockchains (e.g. Corda [4], Hyperledger) where the participants are limited to a predefined set. Since the permissioned setting is simpler than a permissionless setting, these permissioned blockchains can use more efficient protocols for consensus that have been known for decades such as PBFT [6].

6 Conclusion

The choice between a permissionless, permissioned or centralized database is not trivial. While this question has been discussed before [15], to the best of our knowledge, we provide in this article the first structured methodology to decide which technological solution is the most appropriate depending on which application scenario. Our methodology takes into account the required trust assumptions, application requirements, involved parties and technical characteristics such as throughput and latency. We applied our methodology to three known application scenarios that have seen wider interest to adopt blockchain technology and further discussed other use cases. We conclude that depending on the application scenario, there are indeed valid use cases for each, permissionless and permissioned blockchains, and centralized databases that need to be determined carefully.

References

[1] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system. 2009.

[2] Gavin Wood. Ethereum: A secure decentralised generalised transaction ledger. *Ethereum Project Yellow Paper*, 151, 2014.

[3] Eli Ben Sasson, Alessandro Chiesa, Christina Garman, Matthew Green, Ian Miers, Eran Tromer, and Madars Virza. Zerocash: Decentralized anonymous payments from bitcoin. In *Security and Privacy (SP), 2014 IEEE Symposium on*, pages 459–474. IEEE, 2014.

[4] Richard Gendal Brown, James Carlyle, Ian Grigg, and Mike Hearn. Corda: An introduction. *R3 CEV, August*, 2016.

[5] Arthur Gervais, Ghassan O. Karame, Karl Wüst, Vasileios Glykantzis, Hubert Ritzdorf, and Srdjan Capkun. On the security and performance of proof of work blockchains. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, CCS '16, pages 3–16, New York, NY, USA, 2016. ACM.

[6] Miguel Castro and Barbara Liskov. Practical byzantine fault tolerance. In *Proceedings of the Third Symposium on Operating Systems Design and Implementation*, OSDI '99, pages 173–186, Berkeley, CA, USA, 1999. USENIX Association.

[7] Mas working with industry to apply distributed ledger technology in securities settlement and cross border payments, 2017. http://www.mas.gov.sg/News-and-Publications/Media-Releases/2017/MAS-working-with-industry-to-apply-Distributed-Ledger-Technology.aspx.

[8] Carolyn A. Wilkins. Fintech and the financial ecosystem: Evolution or revolution?, 2016. http://www.bankofcanada.ca/wp-content/uploads/2016/06/remarks-170616.pdf.

[9] SWIFT explores blockchain as part of its global payments innovation initiative, 2017. https://www.swift.com/news-events/press-releases/swift-explores-blockchain-as-part-of-its-global-payments-innovation-initiative.

[10] Joseph Poon and Thaddeus Dryja. The bitcoin lightning network: Scalable off-chain instant payments, 2015.

[11] Wenting Li, Alessandro Sforzin, Sergey Fedorov, and Ghassan O. Karame. Towards scalable and private industrial blockchains. In *Proceedings of the ACM Workshop on Blockchain, Cryptocurrencies and Contracts*, BCC '17, pages 9–14, New York, NY, USA, 2017. ACM.

[12] Nick Szabo. Formalizing and securing relationships on public networks. *First Monday*, 2(9), 1997.

[13] Wacław Banasik, Stefan Dziembowski, and Daniel Malinowski. Efficient zero-knowledge contingent payments in cryptocurrencies without scripts. In *European Symposium on Research in Computer Security*, pages 261–280. Springer, 2016.

[14] Steven Goldfeder, Joseph Bonneau, Rosario Gennaro, and Arvind Narayanan. Escrow protocols for cryptocurrencies: How to buy physical goods using bitcoin. 2017.

[15] Gideon Greenspan. Avoiding the pointless blockchain project, 2015. http://www.multichain.com/blog/2015/11/avoiding-pointless-blockchain-project/.