Blockchain, smart contracts and their potential insurance applications

Dimitri Semenovich

Abstract: Many popular accounts of the original Bitcoin blockchain and its newer derivatives leave out the essential concepts required to understand these systems from first principles. In this paper I have tried to explore foundations underlying blockchain technology, hoping to highlight those aspects that indeed promise unprecedented opportunities for innovation.

Keywords: Blockchain, Bitcoin, decentralised insurance markets.

Acknowledgements: Author thanks Tiberio Caetano and Chris Dolman for helpful discussions.

1 Introduction

Many popular accounts of the original Bitcoin blockchain and its newer derivatives leave out the essential concepts required to understand these systems from first principles. White papers talk about "automated audit", "unprecedented financial services innovation", "sales disruption" and the like, claims that are likely to leave a jaded practitioner at best skeptical. In this respect we have a situation not dissimilar to the still recent flurry of "big data" projects, albeit one where the subject matter is better amenable to concrete definitions.

In this paper we will try to explore the foundations underlying blockchain technology, hoping to highlight those aspects that indeed promise unprecedented opportunities for innovation.

First thing to note is that the original Bitcoin[1] protocol combined together several (in themselves quite complex) ideas to create the complete system:

- direct application of public key cryptography and related ideas to the facilitation of financial transaction (rather than their transmission, storage &c),
- a significant innovation in distributed consensus algorithms,
- smart contracts.

Thus treating the "blockchain" as monolithic can be somewhat unhelpful as different applications may require only some of the above (in particular, smart contracts and cryptography are often of independent value from distributed consensus).

As all of these building blocks are likely to be novel to the readers with a background in financial services, I've tried to focus on them individually before describing their interaction.

[1] https://bitcoin.org/bitcoin.pdf

Beyond Bitcoin itself, several blockchain native applications have emerged that are particularly relevant to insurance: distributed lotteries and prediction markets. These schemes offer a fascinating glimpse into what a truly decentralised organization might look like. While it appears quite unlikely that such "from the ground up" designs will be able to replace traditional insurance products, there is considerable scope for new products that fit well with the technology.

Blockchain protocols encourage one to rethink many of the processes taken for granted in financial services. One such example is audit — it is possible, for example, for a Bitcoin exchange to create public proof of the funds it controls without engaging an external party to validate its accounts.

Another example is "smart contracts" — traditionally, core systems in insurance and banking are very inflexible and require long and complex projects even for minute changes. Smart contracts invert that pattern — all of "business logic" is included in the transaction itself. New generation of flexible "core systems" that focus only on executing smart contracts can dramatically reduce complexity associated with IT change management and allow for a much greater degree of product customization, automated settlement &c.

In the following sections we review individual components of blockchain technology and describe some of the promising applications.

2 Cryptographic hash functions

The basic primitive underlying blockchain systems is a class of mathematical functions known as *cryptographic hash functions*. Hash functions generally accept a string of characters of any length and return a fixed length output, e.g. 16 bytes. Such functions should also be fast to compute — complexity of evaluation should not grow more than linearly in the size of the input.

Cryptographic hash functions need to satisfy some additional properties — *collision resistance* and *information hiding*.

Collision resistance means that it is difficult to find two such distinct values x and y that cause a "collision", namely that $H(x) = H(y)$. While very many such pairs exist (by Dirichlet's principle), finding a collision for a given x should be computationally infeasible. This property means that cryptographic hash functions can be used to generate "digests" of arbitrary information, protecting the data against tampering — once the hash value of a file is computed, all one needs is this hash to verify the authenticity of the original file. It will not be possible to generate another distinct file that would result in a matching hash value.

A series of files or data blocks where each block contains data as well as a hash of the previous block is called a *blockchain*. This structure prevents tampering with any of the blocks as long as we know the hash of the latest block and new blocks can be added as required. Bitcoin uses the blockchain data structure to store its transaction history.

Hiding property means that given the output of the hash function $v = H(x)$ it is impossible to discover what the input x was. This clearly doesn't quite work if the set X from which x is drawn is relatively small — it is then easy to discover $x \in X$ such that $v = H(x)$ by simple enumeration. To correct for this, we require instead that it is not possible to find x given $v = H(r + x)$, where $+$ denotes concatenation and $r \in$ R is drawn uniformly at random from a sufficiently large set R.

3 Cryptographic signatures

This is another essential building block of blockchain technology. Digital signature schemes are slightly more involved than hashing and we will again merely describe their properties without going into details of implementation.

First the party that intends to use the scheme must generate a pair of values (s, p) jointly satisfying certain properties. Here s is the *secret key* that needs to be kept private and is used to sign messages. The *public key* p is made available and can be used to verify signatures.

The process can then be described using the following two operations:

- $v = \mathrm{sign}(s, m)$ — signing function takes a message m and a secret key s and returns signature v.
- $\mathrm{validate}(p, m, v)$ — validation function takes a public key p, a message m, and a signature v and evaluates to "true" if and only if the signature was indeed generated on the same message using the matching secret key.

In other words these operations satisfy the condition:

$$\mathrm{validate}\left(p, m, \mathrm{sign}(s, m)\right) = \mathrm{true}\,.$$

Final requirement is that it is not computationally feasible to fake signatures — an opponent who knows a public key and has obtained any number of signed messages can not validly sign an unseen message that would validate against that public key.

Public keys can be used to identify a person — in order for someone to speak for that identity they must have access to the corresponding secret key. Identity creation in this situation is completely decentralised, anyone can create (s, p) pairs at any time without notification. In Bitcoin and other similar system, public keys are used as payment addresses or account numbers.

4 Cryptocurrency

Let's consider how to implement a simple centralised payment system using the ideas presented so far.

We will need two types of transactions — one to *create* new "coins" by fiat and another to *pay* or transfer coins from current owner to a new owner. Transactions will be recorded in a blockchain data structure.

Only the scheme operator is authorised to issue new coins. To do so, they sign a transaction containing coin id, its value and the public key to which it is assigned. Every other participant recognises the new coin to be valid as the originating transaction is signed by the scheme operator. There must exist a shared expectation that only a limited number of coins is to be created for them to maintain value.

The second type of transaction is to pay coins. Anyone who owns any coins is authorized to create a transaction of this type. It "deletes" some existing coins and creates new coins of the same total value, assigned to a new set of public keys (note that "change" may be created by sending some of the value back to the address of the owner). The transaction is signed by the owner of the coins "transferred".

The scheme operator recognises a transaction as valid and appends it to the blockchain if the following conditions are satisfied:

- the transferred coins are active — they were created in a previous transaction and not already deleted,
- the value of the coins deleted equals to the value of the coins created,
- the transaction is signed by the secret key corresponding to the private key to which the coins were previously assigned (only the operator can create new coins by fiat).

The operator then publishes and signs the new block containing the transaction. Every participant is able to validate that the transaction is valid and consistent with the previously published transactions. The operator is unable to modify the history of transactions as it would invalidate the blockchain or to make payments on behalf of those public keys where they don't also control the secret key.

The above already presents quite a powerful pseudonymous payment system — the only downside is the power invested in the operator — while it is difficult for them to "steal" coins outright, they can block payments from certain parties or stop processing payments altogether.

5 Distributed consensus

The major breakthrough of Bitcoin was a decentralised version of the protocol described in the previous section. This eliminates the single point of failure in the operator.

Here we give a simplified version of Bitcoin consensus (ignoring coin creations for the moment):

- new payment transactions are broadcast to all participants,
- each participant collects new transactions into a block,
- at regular intervals a randomly chosen participant broadcasts its block,
- other participants accept the block only if all transactions are valid (in the sense described earlier),
- participants demonstrate that they have accepted the block by including it in their version of the blockchain (i.e. its has will appear in the following block that they collect).

This scheme eliminates the dependency on the central authority and overall has reasonable properties — invalid transaction blocks (e.g. someone trying to spend coins they don't own) get rejected by other honest participants and do not make it onto the consensus version of the blockchain. It is also possible to deal with splits in the blockchain (different participants ending up with divergent histories, e.g. due to issues with network connectivity) using some simple heuristics, such as nodes discarding their version of the blockchain in favour of a longer one should such exist.

There are still two problems — we have no way to create new coins and there is no reliable way to randomly choose a participant to nominate new block or for that matter ascertain how many participants there actually are — if the creation of new identities is sufficiently easy (as is the case with public keys or IP addresses) the system can get overwhelmed by "sock puppets" controlled by a single malicious party — a so called *sybil attack* in which

case we are no better off than having a completely centralised system with the attacker in charge.

Bitcoin offered ingenious solutions to both of these questions.

Instead of participants being chosen randomly to announce the next block, they instead compete at solving cryptographic puzzles. This is called a *proof-of-work* scheme. This makes sybil attacks much more difficult as the cost of assembling sufficient computing power would be prohibitive.

For a new block to be valid, it must contain a fixed length number, called *nonce*, such as that the hash of the nonce appended to the rest of the block falls below certain target value:

$$H(\text{nonce} + \text{new\_block}) < \text{target}\,.$$

Since there is no better known strategy for finding nonces than simple enumeration, the participants win the competition essentially randomly, with the chance of individual participant winning each round proportional to the computational power the can bring to the problem. Competing to find nonces is also called *mining*.

Finally we get to how new coins are created. To incentivise miners to solve hash puzzles, Bitcoin protocol allows them to include a transaction in each new block that creates a coin of predefined value by fiat and assigns it to the address of miner's choice. This rewards miners for spending resources on block craetion and generates a predictable supply of new coins.

6 Smart contracts

Bitcoin does not simply store addresses of coin recipients for payment transactions but instead allows for small custom programs to be executed, check validity of the supplied public key. This is a very powerful idea on its own as it makes transactions themselves programmable — behavior can be implemented that wasn't thought of by original designers. Examples include schemes that allow funds only to be claimed if k out of n potential beneficiaries supply their signatures, decentralised lotteries and betting.

These programmable transactions are known as *smart contracts*.

7 Blockchain applications

There are a few existing applications of blockchain technologies, Bitcoin and newer alternatives, that should be particularly relevant to insurance. In this section we survey a few of these.

7.1 Automated audit

An entity that holds deposits of Bitcoins, such as a Bitcoin exchange, can use a simple technique to generate a *proof of assets* — that it in fact retains control of a given fraction of deposits.

To do so the entity publishes a transaction sending the required funds to a new address (i.e. a public key) and signs an arbitrary message with the secret key associated with the public key to which the funds were sent.

This is quite attractive even though such schemes do not strictly prove ownership but merely temporary control or ability to obtain cooperation from a third party.

It is also possible to construct "proofs of liabilities" but the mechanism is substantially more involved and we do not discuss it here.

7.2 Decenrtalised lotteries

A lottery is not all that dissimilar to an insurance pool — a large number of people deposit their money with a single counterparty who then disburses most of the money collected to a few randomly chosen individuals after taking an often substantial fee. A degree of trust is placed in the lottery organiser as they collect the money from participants prior to paying out prizes.

What makes lotteries different from insurance is the mechanism by which the beneficiaries are chosen — in the case of lotteries it is a random allocation without reference to any specific real world contingency. Lotteries also do not follow the principle of "insurable interest" with winners being strictly better of if they had been otherwise.

Bitcoin and similar public blockchain systems offer a fascinating way to implement distributed lotteries without a trusted central party (and potentially at much lower cost). There are also centralised lotteries using Bitcoin but they merely use it as a payment mechanism and are inherently less interesting.

Here we give a high level description of a simple protocol that can be implemented natively using Bitcoin scripts.

Let's say we have only 3 participants, called A, B and C. They send the money to a specially crafted Bitcoin script which then pays out the total contribution back to one of them at random. The algorithm is as follows:

- Each participant picks a large number A chooses x, B chooses y and C chooses z. They then communicate $H(x)$, $H(y)$ and $H(z)$ together with their payment.
- A, B and C create a new transaction now incorporating x, y, and z. The payment is made to the $(x + y + z)\ \%\ 3$ participant.

It has certain undesirable properties but rectifying them is quite involved. More fully featured smart contracts, such as those available on Ethereum blockchain, make implementation substantially easier compared to Bitcoin. Realistic proposals already exist and it is likely we will see fully featured distributed cryptocurrency based lotteries before long.

7.3 Prediction markets

Another significant area of interest are so called *prediction* or *betting markets*. These offer a way for people to place bets on outcomes of a diverse range of events, from sports to elections and corporate financial results. Prediction markets are also favoured by economists as a way to efficiently aggregate information on a particular topic, such as success of a project, from multiple sources. The ultimate speculative application of prediction markets in this vein is to make decisions e.g. having created a sufficiently active market betting on share price of a

company in 1 year's time conditional on it launching or not launching a new product line, the decision to proceed can be made by comparing the market odds.

There is currently a lot of interest in implementing decentralised prediction markets on blockchain technology, primarily due to the difficulties with their legal status through associations with gambling. Building a decentralized prediction market is a daunting task and would test the current limits of blockchain technology. Augur on Ethereum blockchain is the most advanced implementation at present.

Prediction markets are also nearly equivalent to insurance markets — we will discuss the differences in the next section, but all the considerations carry over.

Main components of a prediction market are as follows:

- A mechanism to accept funds into *escrow*, make payouts according to event outcome. This broadly corresponds to payment processing and treasury or capital management function of an insurance company.
- An *arbitration process* for determining the outcomes in question. Arbitration can be both decentralized (by consensus of market participants, another group such miners or even another market) and centralized. Further questions arise when the outcomes are ambiguous or not a matter of public record. In insurance context this would correspond to claims assessment.
- An *order book* or similar mechanism (such as a market maker relying on a scoring rule) for participants to find counterparties to trade with. An order book contains bids and asks. A bid is a buy order and ask is a sell order. Typically the ask price is higher than the bid, otherwise two participants are matched up and a trade occurs, eliminating one of the orders. This has no close analoque in insurance outside of risk securitasion schemes.

All of the above require an in depth treatment of their own but we will focus on arbitration since it's one of the areas where established firms can play a significant role.

Easiest way to perform arbitration is for the market to nominate a trusted party to provide the outcomes. This arbitrator doesn't event need to know that their services are being used by a particular market, as long as they make the data (such as stock market feeds, election results &c) available with an appropriate digital signature — these can then be validated by the market and acted on automatically. It is then be possible to create contracts for arbitrary contingencies that can be expressed through a combination of these trusted data feeds.

A more decentralised arbitration system where either several data providers vote or market participants themselves vote may be ultimately more preferable, however at this stage these remain experimental.

Following is an example of how a simple bet can be implemented using the Bitcoin protocol:

- An arbitrator creates two pairs of keys (s_0, p_0) and (s_1, p_1) for "No" and "Yes" outcomes of a certain event and publishes the public keys. Once the outcome is known, they also publish the corresponding secret key.
- A wishes to bet on a "Yes" outcome and B on a "No" outcome. They deposit money to a bitcoin script from which payments can be withdrawn either using signatures from A and "Yes" or B and "No".

8 Possible insurance use cases

We have focused primarily on the basic functions of public blockchain technology and some of its proposed uses. In this section we go through several insurance specific use cases applying these ideas. It is worth noting, however, that only one explicitly requires the public blockchain and the rest can be implemented by other means.

Cryptographic certificates of insurance — digital signatures can be applied to the creation of cryptographic certificates of insurance in situations where the purchase is intermediated or evidence needs to be presented to third parties. The issuing insurer simply digitally signs the message containing policy details, such as the details of the policy holder and the effective dates. This message can then be verified either by the policyholder to confirm that an intermediary has correctly processed their application or by a third party (e.g. if insurance is compulsory).

Insurance pools — a basic decentralised insurance pool is not fundamentally different from a lottery as discussed earlier, all what is needed is a mechanism to determine who qualifies for payment at the end and how surplus or shortfall is allocated. All of these tasks can in principle be carried out by a third party trusted by all the participants to make the final determination. At the end of the nominated period, they would authorize the transactions dispersing the pool to appropriate beneficiaries. Another option is to have the members vote on the outcomes and act on a consensus.

These design also covers the so called "savings clubs", where "winning" members rotate on a predetermined schedule.

Insurance markets — again there is no fundamental difference between a prediction market and an insurance market. Indeed, prediction markets today can be used to buy insurance against election outcomes and similar (thus potentially moving market odds away from true probabilities). Distinctions are premiarily in the types of outcomes that can be arbitrated, with insurance gravitating towards very large number of relatively small scale events of no or limited public interest (property damage &c) and the approach to capital management.

A close analogue of prediction markets in insurance is found in secondary markets for catastrophe bonds with parametric triggers. It appears likely that any early successes of lower-end insurance markets that combine origination and placement via an order book mechanism will follow a similar template and will be related to outcomes that are relatively few in number and are generally either a matter of reliable public record, such as weather, flight delays &c or can be reliably ascertained programmatically e.g. when a mobile phone is placed remotely in "lost" mode.

On capital side, prediction markets are always fully funded — when a bid and an ask are matched, the funds are placed in escrow for the duration of the bet. With insurance markets this may not always be practical due to high asymmetry in maximum losses between two parties. An insurance contract sold for $1000 may have a limit of $10m. This will either require market wide or facultative reinsurance or a more creative approach to funds escrow. Benefits of diversification can be recognised by splitting contracts into smaller slices and gradually relaxing the requirement for claim obligations to be fully funded as an investor buys enough slices in "independent" contracts.

Another issue worth pointing out is the "insurable interest" constraint placed on traditional insurance contracts — these essentially prevent one from buying "naked" positions in negative outcomes. Similar considerations, however, emerge in prediction markets. Often it is

easier to bring about a negative value outcome (e.g. when a market tracks project success) and it is desirable to make sure that the negative position one can take does not exceed the loss through other means (such as performance bonus) in case of project failure.

Core systems for smart contracts — traditional core IT systems in insurance and banking are quite inflexible, with implementation costs often dwarfing commercial value of boutique or customized products. The use of smart contracts as the fundamental primitive can significantly change this dynamic. This way the logic governing a product is contained within the transaction itself and is implemented using a generic scripting capability. This allows to accommodate a very large range of possible products without anticipating them in advance and the core systems merely need to provide a robust mechanism to execute these smart contracts.

9 Conclusion

Public blockchain technology offers potential that goes well beyond the imagination of this author. For those interested in insurance applications, lotteries and prediction markets are likely to prove a great source of ideas.

Many plausible insurance use cases, however, do not require full public blockchain and can be dramatically simplified if decentralization is not required — these appear to be areas where established players will be more likely to gain traction.