Validation of Decentralised Smart Contracts through Game Theory and Formal Methods

Giancarlo Bigi[1], Andrea Bracciali[2], Giovanni Meacci[3,4], and Emilio Tuosto[5] *

[1] Università di Pisa, IT. giancarlo.bigi@unipi.it
[2] Stirling University, UK. abb@cs.stir.ac.uk
[3] http://www.BitHalo.org giovannimeacci@gmail.com
[4] Coin Cube LLC, New York, NY, 11213
[5] Leicester University, UK. et52@leicester.ac.uk

Abstract. *Decentralised smart contracts* represent the next step in the development of protocols that support the interaction of independent players without the presence of a coercing authority. Based on protocols à la BitCoin for digital currencies, smart contracts are believed to be a potentially enabling technology for a wealth of future applications. The validation of such an early developing technology is as necessary as it is complex. In this paper we combine game theory and formal models to tackle the new challenges posed by the validation of such systems.

1 Introduction

The introduction of the BitCoin protocol in 2008 has strongly pushed forward the development of *decentralised distributed systems.* BitCoin is decentralised since it is not controlled by any central coercing authority. Rather, a computationally expensive distributed consensus over the internet certifies its transitions, for instance preventing the double expenditure of immaterial money. Due to the computational costs involved, the consensus of the whole BitCoin network over the internet cannot realistically be overturned. Although BitCoin has been highly volatile and associated to illegal activities, institutional players, including governments and banks, as well as the general public, have shown interest in it. BitCoin has started to appear as a potentially reliable and enabling technology.

Currently, the next step builds on top of BitCoin, aiming to introduce decentralised distributed technologies on a larger scale. One example of this are *decentralised smart contracts*, i.e. protocols designed to define self-enforcing contracts amongst untrusted and independent players. BitHalo is a paradigmatic, very recent and innovative example of a decentralised smart contract.

The validation of such protocols is clearly highly desirable and, as usual, very complex. Interestingly, the distributed and decentralised aspects of smart contracts add to complexity since free choices and gaming strategies come into play. Protocols are run by autonomous players, possibly mixing physical actions,

* Authors would like to thank David Zimbeck for useful discussions and for sharing information about BitHalo.

e.g. the shipment of goods, and computer-mediated ones, e.g. an electronic payment, without any possibility of a coercing central authority. Differently from more traditional protocols, a sort of socio-economical aspect becomes relevant for the validation of smart contracts.

In this paper, we analyse and validate DSCP, an idealised smart contract inspired by BITHALO. A distinguished feature of our approach is the combination of *game theory* and *formal methods* to suitably address the mentioned complexity of the analysis and validation of smart contracts. Game theory has been widely exploited to analyse how contracts are settled through bargaining procedures (see, for instance, [21]) but the analysis of protocols that enforce contracts is a novel area of application. Formal methods have been extensively used for protocol validation, from security protocols to the more recent behavioural contracts of application level protocols. However, it is worth remarking that such kind of contracts and the contracts supported by BITHALO exist in different contexts and for different purposes. Indeed, in behavioural contracts the main focus is to ensure that the parallel composition of distributed participants does not yield communication problems such as deadlocks (see, for instance, [3, 5, 6]) or to analyse communication misbehaviours in untrusted settings (see, for instance, [4]). Noticeably, BITHALO also embeds steps that depend upon decisions made by human players, which do not appear in application level protocols.

In our framework game theory and formal methods complement each other: the former caters for the study of the gaming strategy aspects, while the latter provide the grounds for a precise definition of the protocol and related working hypotheses. Furthermore, the probabilistic framework we adopted allows us to properly model uncertainty and non-determinism in players' behaviour, and to exploit effective automated techniques, like statistical model checking, to validate the properties of the smart contract. To the best of our knowledge, the proposed combined approach is here firstly applied to the validation of smart contracts.

A detailed analysis of DSCP is carried out both analytically from the viewpoint of game theory, and computationally, via the definition of a model, the properties of interest, and their validation through probabilistic model checking. Sensitivity of various parameters is studied, including monetary values and fraudulence profiles of the players. Our combined analysis formally and quantitatively clarifies the intended behaviour of the protocol, which relies on a deposit scheme to enforce trust. Sometimes, assumptions on the deposit scheme may result in being unrealistic. Our analysis explores the details of the system under the uncertainty introduced when the deposit enforcing trust assumption is weakened.

2 Bitcoin-based Smart Contracts

Smart contracts [26, 27] are protocols defining self-enforcing, digital contracts. The main aim of such contracts is to guarantee fair exchanges between untrusted and independent entities. The recent introduction of the BITCOIN protocol [1]

by Satoshi Nakamoto[6] in 2008 [22] allowed for decentralised smart contracts. BitCoin provides decentralised virtual monetary instruments that can support contracts which do not require intermediaries, central repositories or single administrators. The huge potential of these contracts calls for a formal analysis and validation of their properties.

BitHalo [29, 7] is a recently developed smart contract based on BitCoin. It is supported by a freely available software platform and, to the best of our knowledge, is the first off-blockchain, decentralized smart contract. A short introduction to BitCoin and BitHalo follows.

2.1 BitCoin: a protocol for decentralised applications

The first application of the BitCoin protocol has been the digital, decentralised, partially anonymous currency called *bitcoin* (BTC), which is not redeemable for gold, and not backed by any government or legal entity.

BTC total market capitalization ranges between three and four billion USD, depending on the BTC/USD exchange rate (June 2015). There are nearly nine million of BTC wallets (April 2015), reaching 100,000 transactions per day (February 2015). The New York State Department of Financial Services has released a regulatory framework for digital currencies [18] (June 2015). California and UK government are considering similar options. BitCoin venture capitalist investments are expected to reach 1 billion USD by the end of 2015 [28].

Digital currency predecessors of BitCoin used centralised clearinghouse systems in order to address the problem of fraudulent transactions, exactly like traditional banking systems. However, these centralised structures provided them with a potential single point of attack and failure, and they became easy targets of governments, hackers, and criminal entities, and eventually failed.

BitCoin combined previous inventions such as b-money [13] and HashCash [2], and introduced four critical innovations that eliminated the main weakness of its predecessors [1]: (1) a decentralised peer-to-peer network that allows users to transfer BTCs; (2) a trusted public ledger (called blockchain) with the list of all transactions that took place within the system; (3) a process called *mining* that let the BitCoin protocol act as decentralised clearinghouse and allows new BTCs to be created through the solution (called *proof-of-work*, POW) of a mathematical problem based on a cryptographic hash algorithm; (4) a decentralised transaction verification system.

These additions addressed the problem of double spending, where a coin is spent twice, and other fraudulent transactions. Let consider a simple practical example. Using (1) user A transfers to user B the money that he received in a previous transaction T. Because of (2) each user in the BitCoin network can check the blockchain and verify whether money from transaction T has been already spent by A or not. Correctness of the blockchain information is given in terms of consensus by the vast majority of the nodes in the network. This is

[6] This name is believed to be a pseudonym.

achieved with the points (3) and (4), which rely on distributed computational resources and can be informally summarised as follows.

Each BitCoin node keeps a record of the blockchain, which literally is a chain of blocks. Once mined and validated, blocks are assembled one after the other to form the blockchain. The difficulty to find the POW, i.e. mine a new block, is periodically adjusted to the current computing power of the participants in such a way that a single block validation happens on average each 10 minutes [12]. The system contains an incentive for *miners* to validate transaction and finding POWs in terms of BTCs and transaction fees.

Originally, the blockchain was just a long single chain of blocks, but more recently it has a more complex topology with bifurcations and even "orphan" isolated blocks. The longest chain, i.e. the chain which has the most POWs, is independently selected by every node as the main chain, i. e. the blockchain. In practice, if the block containing T is deep in the blockchain, the amount of computational power needed to force a fork and rebuild an alternative, longest chain with a modified T, makes invalidating T unfeasible.

Nakamoto's idea of POW-based consensus makes unnecessary any central trusted authority in charge of issuing currency and validating fast, secure, borderless, and commission-fees free, financial transactions. The approach is applicable to a variety of different fields, such as the registry of property (see, e.g., "The Property Rights Project" [25]), fairness of elections, lotteries, digital notarisation, storage of personal and sensitive data, smart contracts and more.

2.2 BitHalo: decentralised smart contracts

We consider here BitHalo as a paradigmatic example of smart contract. The term "smart contract", together with the idea of "smart property" [11], was introduced by computer scientist Nick Szabo, during the early 1990's [26].

The BitCoin scripting system already had a variety of script hashes aimed at supporting different kinds of smart contracts [10]. However, BitHalo doesn't rely on those hashes because of known drawbacks in the BitCoin smart contract protocol [10], as well as in some proposal based on it [20, 9].

The purpose of BitHalo is to create unbreakable trade contracts without the need of arbiters or escrow agents, lowering significantly the costs for the two parties involved in the contract. Since it does not require trust, nothing in the BitHalo system is centralised. It does not require a server, just the Internet. Its peer-to-peer communication system allows the two parties to use email, Bitmessage, IRC, or other methods to exchange messages and data. BitHalo is off-blockchain in the sense that the record of BitHalo contracts is not kept in the blockchain, and therefore the use of BitHalo will not bloat the blockchain.

BitHalo can be used for bartering, self-insuring, backing commodities, performing derivatives, making good-faith employment contracts, performing two-party escrow, and more general business contracts.

Transactions are insured by a *deposit* in one of the supported digital currencies (including BTC) on a joint account, double-deposit escrow. The BitHalo protocol forces each party to uphold the contract in order to achieve the most

economically optimal outcome. In a typical contract exchanging a payment for goods or services, the payment can be sent either separately, using checks, money transfer, crypto-currencies, etc., or paid directly with the deposit. The deposit will only be refunded to both parties on *shared consent*, which has to be expressed by both parties. In the lack of expression of shared consent, the joint account will self-destruct after a time-out. Time limits and deposit amounts are all flexible and agreed upon by both parties. Dissatisfaction about the outcome of the transaction by one of the parties, for instance because of theft or deception, will lead to the destruction of the deposit due to the lack of shared consensus. When the deposit exceeds the amount being transacted, the loss typically results larger than the benefits possibly obtainable by a fraudulent behaviour. However, deposits exceeding the transacted amount may be in some cases unfeasible. In some situations, smaller deposits may incentivate one or both parties to break the contract.

2.3 DSCP, a decentralised smart contract protocol

While the BitHalo platform enables users to interact through several variations of a core smart contract protocol, for the purposes of our analysis we fix here the details of DSCP, an idealised distributed and decentralised contracting protocol that mimics one of the possible interaction modalities of BitHalo. The results of the analysis of DSCP, carried out by means of the contribution of both game theory and formal verification, will be presented in the rest of the paper.

As standard, DSCP allows two parties, i.e. the two players of the protocol, to autonomously exchange money against goods without the need of a centralised arbiter. It is worth remarking that the two players are completely independent, not subject to any third party authority in the execution of the exchange protocol, and can, for instance, decide to leave the protocol at any time.

DSCP is based on the mentioned notion of "enforced trust" in the fact that none of the two parties will ever be in a position in which breaking the protocol is for them advantageous. We will see that this, as expected, will be properly enforced only when the deposit, whose payment is a pre-requisite for the execution of the protocol, exceeds the value of the goods.

DSCP allows payments to be made disjointly from the trust-enforcing deposit. This adds flexibility, e.g. users may want to pay in fiat currencies and use BTCs only for the deposit (which can only be in crypto-currencies), and makes the set of possible interactions richer and therefore more interesting to be analysed.

In our interpretation, we distinguish between the *money* m and the *value* v belonging to a player of the protocol. The former being the cash availability and the latter the asset value of the player. When a buyer (analogously a seller) successfully buys an item, their money will decrease, while the value of the goods they possess will increase of the same quantity, assuming fair prices. We assume that both price and value coincide, and the buyer and the seller give the same value to a given item (other choices are possible, see Footnote 7). In an ideal world with "robust" contracting protocols, the sum of money and value of each player and therefore the overall value and money in the system

should stay constant in time. We will show that such a "wealth preservation property" holds when the deposit exceeds the value/price[7] (and players do not behave against their interests). Autonomous players can decide to break the protocol because either it is advantageous for them or because of contingencies and free human behaviour, even if not necessarily advantageous. Examples are, respectively, breaking the protocol when the consequent loss of the deposit is less than the advantage obtained, and abandoning a transaction due to a too long response time of the other party and, perhaps, a not too constraining deposit. Players' free choices may clearly break the conservation of wealth in the system. A player may gain money while another may loose asset, or both may loose money because breaking the protocol leads to the loss of deposits.

We will not model time explicitly. Consequences of time-outs like a too long response are understood as the possibility for players to leave the protocol, and negate consent, at any time.

Furthermore, we will consider probabilities, together with parameters like money and value, to model players' behaviour and their choice capabilities. This will allow us to account for diverse player profiles.

These assumptions lead to a quite reach model where different aspects have to be accommodated in order to fully describe the protocol and its implications at various levels. These aspects range from the precise description of the possible interplay of the two autonomous players, to the psycho-economical forces that may drive their choices.

3 Game Theoretic Analysis of DSCP

Once the terms of the contract are agreed, the two players - the buyer and the seller - act independently of each other in the actual transaction, but the choices of the former affect the result and the behaviour of the latter and vice versa. Therefore, concepts and ideas from noncooperative game theory can be exploited to analyse the transaction protocol and, as a consequence, the quality of the agreement as well.

The agreement requires setting a price p for the item and the value of the safety deposits of the buyer and seller, namely d_b and d_s. Clearly, the buyer has to consider the item worth paying p: turning the utility of the item for the buyer into a (monetary) value v_b, it must be greater than p. Similarly, the value v_s that the seller assigns to the item has to be at most p. The determination of prices is a central topic of economics that goes far beyond the aim of this paper, so we will simply consider p as given and any pair of values $v_b \geq p$ and $v_s \leq p$.

The role of the deposits is more relevant for our analysis: they are meant to guarantee that the players will actually perform the transaction. The choice of their value requires some kind of pre-transaction analysis of the transaction

[7] We stick to a quite simple vision of trading. An interesting alternative would be assuming that $v < p$ for the seller and $p < v$ for the buyer. In this case both would have an incentive to come to the shared consent, both increasing their wealth. The wealth preservation property would not hold. This is scope for future work.

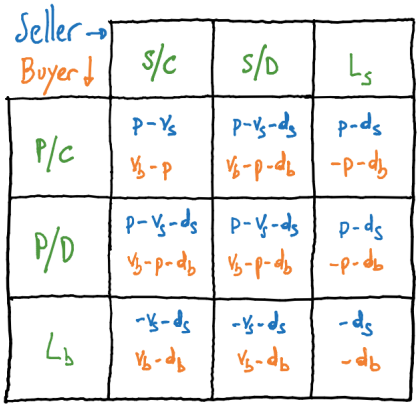


Table 1. "One shot" strategic game.

itself by both sides. Imagining that both decide their full strategies at the very beginning of the transaction in one shot, each player can analyse all the possible outcomes as the result of the behaviour of both. Relying on game theory, this analysis can be carried out modelling the transaction as a *strategic or normal game* of two players (see, for instance, Section 2.1 in [24]).

A priori, once that the agreement is settled and the deposits paid, the buyer can behave in the following ways: pay the item and confirm a satisfactory transaction, namely *strategy* $[P/C]$, pay and leave the system denying a satisfactory transaction $[P/D]$, leave the system without paying $[L_b]$. Similarly, the seller can ship the item and confirm a satisfactory transaction $[S/C]$, ship and leave the system denying a satisfactory transaction $[S/D]$, leave the system without shipping $[L_s]$. Each pair of strategies leads to an outcome given by a pair of *pay-offs*, one for each player. The nine possible outcomes of the transaction are given in Table 1 through a bimatrix where the top entry in each cell is the pay-off for the seller and the bottom entry for the buyer.

Comparing the strategies of the buyer, $[L_b]$ *dominates* $[P/D]$ since the former provides a better pay-off than the latter for any possible strategy of the seller: the buyer would never select $[P/D]$. Notice that $[P/C]$ *dominates* $[P/D]$ *weakly*, that is some payoffs are equal while none is better for the latter strategy. If $p > d_b$, then $[L_b]$ dominates $[P/C]$ as well and the buyer would select the strategy $[L_b]$: as a consequence, the seller shouldn't have agreed such a price and deposit. The comparison of the strategies of the seller is analogous: $[L_s]$ dominates $[S/D]$, $[S/C]$ dominates $[S/D]$ weakly; if $p > d_s$, then $[L_s]$ dominates $[S/C]$ as well and the seller would select the strategy $[L_s]$.

Indeed, the technique of iterated elimination of dominated strategies shows that, whenever $p > d_b$ or $p > d_s$, the strategy profile $([L_b], [L_s])$ is the unique Nash equilibrium of the game, that is the unique profile such that no player can improve their own pay-off changing strategy while the other does not.

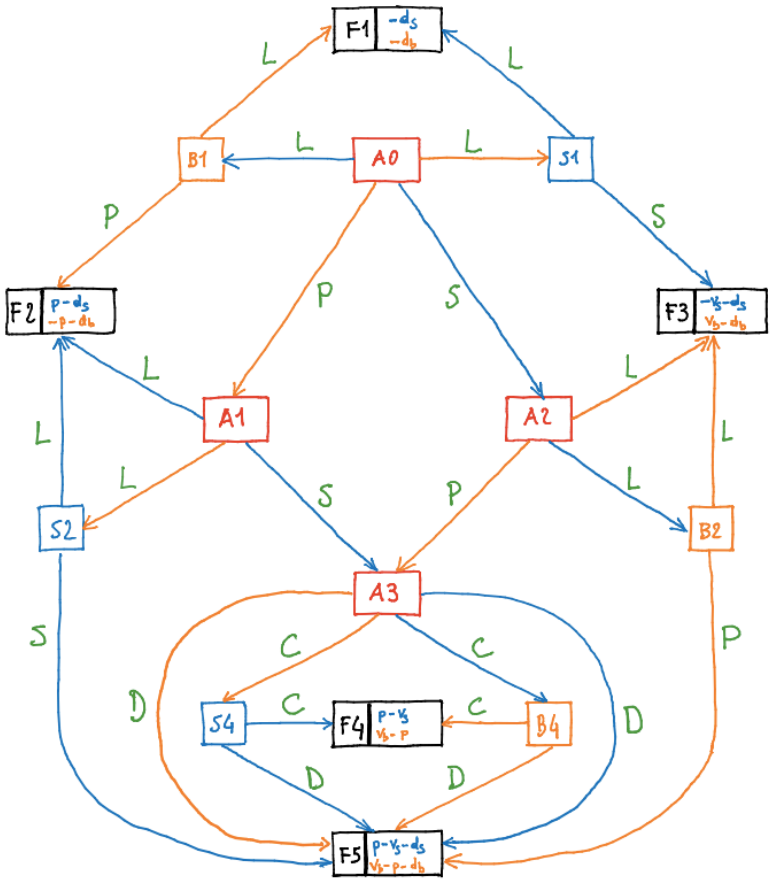


Fig. 1. The graph of the transaction protocol.

This basic analysis suggests that the players should agree deposits $d_b \geq p$ and $d_p \geq p$. Indeed, in this case the successful transaction $([P/C], [S/C])$ is a Nash equilibrium as well.

Though the above formulation as a strategic game can be useful for a preliminary analysis, it does not fully catch the nature of the actual transaction: the choices are not taken altogether at the very beginning but somehow sequentially. As the players are perfectly aware of the state of the system, this knowledge may and actually does influence their next choices.

The transaction can be described through the directed graph of Figure 1. Each node represents one possible state of the system, while each arc represents one action that can be taken by one of the players in the state at the tail node leading to the state at the head node. Node A0 represents the initial state (agreement settled), while the nodes labelled with F represent the end of the transaction with a specific outcome. Notice that there are five end (F) nodes

and they match the five different outcomes given in Table 1. The remaining nodes are labelled with the player or the players that may take an action. Notice that the same state/node may allow actions by both players: there is no player exclusively in charge of the next move and they could also act simultaneously. In this latter case no problem arises as considering arbitrarily one action before the other leads eventually to the same state. A node is labelled with A if both players can act, with B or S if only the buyer or the seller can act. The buyer can take the following actions: pay the item (arc label P), confirm (C) or deny (D) a satisfactory transaction after the item has been paid and shipped, and leave the system at any state (L). Similarly, the seller can ship the item (S), confirm (C) or deny (D) a satisfactory transaction, and leave the system (L). Each action can be taken at most once. Moreover, leaving the system is a final choice: if one player leaves the system after the other has shipped or payed, the next choice of the latter does not affect the outcome and hence it can be ignored. Similarly, shipping or paying, if not already done, is the unique sensitive choice after the other player has left the system.

This description does not fit perfectly the definition of a sequential or extensive game with perfect information (see Definition 89.1 in [24]), which requires a unique player to be in charge of the next at each state. Anyway, concepts and ideas can be borrowed from the theory of extensive games all the same.

In this framework a *strategy* of a player is a plan that provides one action for each state at which the player can take action. Notice that pairing one plan of the buyer and of the seller does not necessarily identify a unique outcome as the order of moves at common nodes may determine different paths. Nevertheless, *backwards induction* provides useful information on how the transaction is likely to happen: as the players are supposed to act rationally, at each node they are going to choose one action, if any exists, that necessarily leads to the best pay-off between those that can be still reached from the current state; therefore, all the "non-optimal" actions can be cancelled. Applying this idea backwards from the final states (F) to the initial one (A0) provides only likely paths from the latter to the former.

Figure 2 illustrates the result of the backwards induction supposing that the safety deposits are larger than the price of the item, i.e. $d_b > p$ and $d_s > p$. The picture shows that A1 is actually a time sensitive state for the buyer: after having paid, the buyer has no advantage to leave the system rather than waiting, potentially forever, for the seller to take some action. Similarly, A2 is time sensitive for the seller. Thus, if they both rule out leaving the system at their own time sensitive states, the unique paths left describe the ideal transaction: the buyer pays, the seller ships and both confirm a satisfactory transaction (no matter in which order).

If the safety deposits are equal to the price, i.e. $d_b = p$ and $d_s = p$, the same kind of analysis does not exclude any possible final state except F5 coming out as the result of the following unreasonable behaviour: the buyer pays, the seller ships and one of them denies a satisfactory transaction. Finally, if the deposits are smaller than the price, the backwards induction provides only the final state

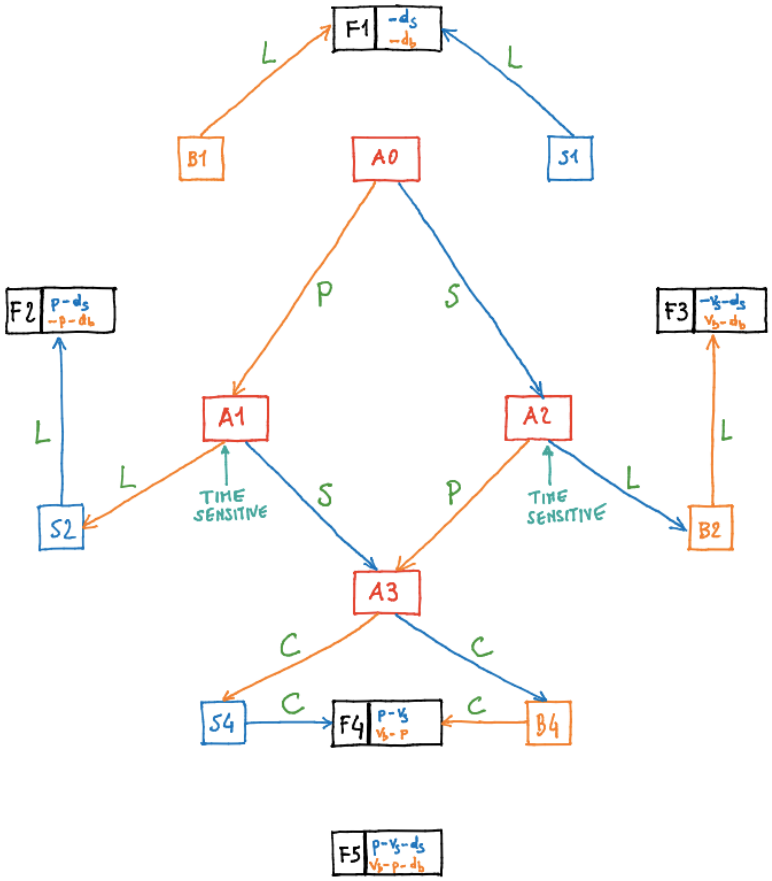


Fig. 2. The graph reduced by backwards induction (for $d_b, d_s > p$).

in which both leave the system without paying and shipping, as it was already suggested by the strategic game given in Table 1.

In conclusion, the above analysis based on game theory suggests that the players are likely to perform DSCP satisfactorily if the deposits are larger than the price, no matter how much bigger. It is worth stressing that the whole analysis is based on the assumption that the seller, if not leaving earlier, ships the right item and in turn the buyer, if not leaving earlier, pays exactly the agreed price. The analysis of this kind of unfair behaviours is beyond the aim of this paper.

4 Formal Verification of DSCP

Players' strategies under the hypotheses of a perfectly rational, utilitaristic and deterministic behaviour have been analysed by means of game theory. It has also

been proved that a deposit exceeding the value of the traded goods guarantees the fair execution of DSCP under the mentioned hypotheses.

However, as observed, the requirement on the deposit might be unrealistic, especially for expensive goods. It is therefore worth studying the protocol behaviour with smaller deposits, when players adopt a less strict and more realistic behaviour, and also when considering the impact of different players' profiles on the protocol, e.g. honest and fraudulent ones.

This is done by defining a probabilistic formal model of DSCP and exploiting the PRISM probabilistic model checker [17] to validate the properties of interest.

4.1 Protocols, contracts and formal verification

Communication and interaction protocols are not simple to design, verify, and implement. A paradigmatic case is that of the well studied security protocols, which can typically be described in terms of very few steps describing the participants involved in the communication, the sharing of secrets among them, and the information they generate and exchange during the protocol execution. Their simplicity is however only apparent; designing a provably correct protocol is very hard and there are several examples of security protocols found flawed after having been considered correct for a few years, the paradigmatic example being the Needham-Shroeder [23] protocol and the Lowe's attack to it [19], discovered by means of formal verification. Among the reasons behind such a complexity, is the difficulty in formally identify an "attacker model" (and often a precise definition of the security properties to guarantee/check).

Formal methods have been advocated as suitable tools for the rigorous specification and verification of security protocols and their attacker models and several formal approaches to the verification of security protocols have been successful. For instance, the precise definition of the attacker model of Dolev-Yao [14] allows the execution of protocols to be clearly described.

Various kind of contract protocols are being verified by means of formal techniques. A key difference between security protocols and smart contracts like DSCP is the fact that the properties of interest of the latter escape the usual domain of the properties of security protocols, hence providing new interesting research directions. Intuitively, DSCP aims to guarantee that one of the two parties involved in a financial transaction cannot "cheat without a penalty". Although intuitively simple, such property alone is not satisfactory; in fact, the penalty which one of the parties incurs in has to be compared to the advantage to cheat. Therefore, a crucial phase of DSCP is the determination of the penalties the two parties are agreeable with.

4.2 Markov Decision Processes

In the execution of DSCP under the mentioned hypotheses, players can synchronise their actions, exhibit probabilistic behaviour typically depending on the state in which they are, and make non-deterministic and probabilistic choices.

A *Markov Decision Process (MDP)* conveniently describes decision making processes that may depend on non-deterministic and random choices (see [15] for an introduction to MDPs, associated temporal logics and their use in PRISM).

Informally speaking, an MDP can be understood a state-based automata which first resolves the nondeterministic choice of the next action to be performed, and then resolves the probabilistic choice amongst the possible next states the chosen action may lead to.

PRISM supports MDPs and their analysis and simulation in process algebra like settings. For instance, PRISM caters for synchronisation, i.e. different automata synchronise on the execution of certain actions, which belong to some specified synchronisation set. By "unfolding" the effects that non-deterministic choices may have on probabilistic ones, non-determinism may be resolved and the MDP reduced to a Discrete Time Markov Chain (DTMC).

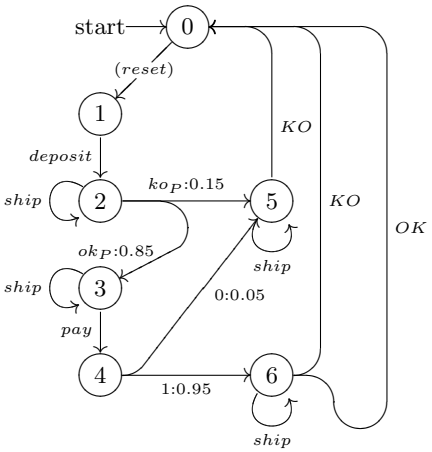
PRISM also supports Temporal Logics for expressing properties of an MDP, which can then be validated. For our purposes, validation will consist in an approximated statistical approach: a suitable number of system evolutions are explored in order to approximate the probability that a property of interest holds. Probabilistic properties have been expressed in the PCTL logic [16, 8], whose temporal operators include, amongst others, $X\phi$, i.e. property ϕ holds in the next state with a given probability, and $F\phi$, eventually a state satisfying ϕ will be reached with a given probability.

4.3 A probabilistic model of DSCP

The automaton in Figure 3(a) formalises the assumptions about the functioning of DSCP outlined in Section 2.3, and makes other details unambiguous, such as the choice that price and value coincide and are the same for both players. Furthermore, we do not distinguish here between leaving (L) and denying (D) a protocol. While such a distinction may be useful for more complete game theory analyses, the two actions have in the current settings the same effect.

The automaton describes the behaviour of the buyer. The whole model of DSCP, an MDP, consists of this automaton and a symmetrical one for the seller, which only swaps paying for shipping, highlighting the high symmetry of the two roles under the assumptions made.

From the initial state 0 only the transition to state 1 is enabled and used to reset the initial conditions of a protocol execution. Therefore it is not relevant to the player behaviour. From state 1 only the *deposit* transition to state 2 is enabled. This models the agreement and payment of the deposit and the actual start of the protocol. Here *deposit* is a synchronisation action that the two automata can only perform together. It is worth noting another synchronisation action, i.e. *ship*, that appears in several states in a loop transition. This represents the seller sending goods and has been modelled as a synchronisation, as the buyer must be aware of the shipment, too. In order to avoid deadlocks, the buyer is (almost) always ready to synch on *ship*. Symmetrically, the seller is (almost) always ready to accept payments by synching on *pay* (this avoids both being deadlocked on *ship*/*pay*, the other not being ready to synch).



(a) Buyer's automata

```
114 //S2
115         [ship]          s=2 & received = false  -> (received'=true) & (v'=v+p);
116
117         []              s=2                     -> (received? _p_ok : 0.85)   :(s'=3)  +
118                                                    (received? _p_ko : 0.15)   :(s'=5)
119                                                    & (__ko'=true);
120
121 //S3    PAY
122         [pay]           s=3                     -> (s'=4) & (m'=m-p);
123         [ship]          s=3 & received = false  -> (received'=true)
124                                                    & (v'=v+p);
125
126 //S4    DECIDE
127         []              s=4                     -> (received? 1:.95): (s'=6)  +
128                                                    (received? 0:.05): (s'=5) & (__ko'=true);
```

(b) Buyer's PRISM implementation

Manual exploration

Module/[action]	Probability	Update
Buyer	0.85	s'=3
Buyer	0.15	s'=5, __ko'=true
Seller	0.85	ss'=3
Seller	0.15	ss'=5, __ko'=true

(c) Actions's probabilities in state 2

Fig. 3. Model and implementation of the Buyer's player.

State 2 is actually one of the two nodes where players can make a decision (double arrow): either leave, and hence break the protocol, by moving to state 5 with loss of the deposit, or play their own part by moving to state 3, which in this case leads to the payment of the agreed price. There is not synchronisation here as this is a personal (a.k.a. internal) choice. The two branches of the arrow are labeled with $n : m$. These are the two probabilities associated with each branch for the case in which the buyer (seller, resp.) has (n) or has not (m) received the goods (payment, resp.).

The modelling of such a probabilistic choice is key. If the buyer has received, but not yet paid at this stage, the probabilities ok_P to follow the protocol and ko_P to abandon it (both players loose the deposit) take into account the ratio of the paid deposit over the price still to be paid (symmetrically, the deposit over the value of the goods to be sent):

$$ok_P = min(1, \left(\frac{d}{p}\right)^r) \qquad ko_P = 1 - ok_P$$

As expected, if $p \leq d$ the protocol is followed with probability 1 because there is no gain in stealing the goods and loosing the deposit. Otherwise, the probability decreases as much as the deposit is irrelevant with respect to the price/value. Furthermore, we have added the exponential r to model the player's attitude. With $r = 1/2$, say, the value d/p is amplified towards 1, i.e. reducing the attitude to steal, as an honest player would typically do. With $r \geq 1$ the effect is the opposite, increasing the probability to steal.

If goods have not yet been received (m value on the arrow), it is assumed that the buyer (seller) will proceed to payment (shipment) with $P = 0.85$, hence

following the protocol most of the times. This choice is worth some consideration. As detected by game theory, this is a critical choice: players can abandon now the protocol and loose the deposit, or proceed and be possibly driven to an even larger loss (see the pay-offs of Section 3 when $v_s = v_b = p$). In the game theory interpretation, then, abandoning now, after having paid the deposit, is a plausible choice. This has informed our choice to retain a minimal probability of abandoning. A possible alternative and indecisive $P = 0.5$, or any probability leaning towards abandoning, would invalidate the spirit of the protocol, neglecting the interest of the players to trade (see Footnote 7). Moreover, it should also be considered that, although we do not model *time* explicitly, players may decide to abandon the protocol and loose the deposit after a too long wait for the counterpart to act, possibly not wanting to make the first move (it is worth remarking here that in our idealised DSCP, participants do not communicate directly). This can be accounted for by the non-null 0.15 probability of abandoning the protocol at this stage. Such probability could be tailored on empirical data about protocol usage.

It is worth remarking how aspects such as the utility and attitude of players and some form of time-dependent events can be easily and clearly embedded in the model, as done in the transition described above.

Finally, the co-existence of non-determinism and probabilistic choices in state 2 has to be noted. Informally speaking, according to MDP theory, this is dealt with by first resolving the non-determinism between *ship* and the action regarding the probabilistic choice about abandoning or continuing the protocol. Several possibilities may arise, e.g. the seller might not be ready to synch on *ship*, making the choice deterministic, or the buyer might follow a specific policy to resolve non-determinism (the theory defines the concept of possible *adversaries*). A probabilistic choice will be made only if the probabilistic action has been selected when resolving non-determinism.

From state 3 only the *pay* action is enabled leading to the next choice state 4, with analogous conditions to state 2, the difference being that the player has paid/shipped. If the player has already received they are happy, $P = 1$, to express consent, i.e. move to state 6, otherwise, they are however more incentivised to reach consent, $P = 0.95$. The 0.05 probability of abandoning the protocol accounts for timeouts and other contingencies.

In state 5 the choice has been made and, although *ship* is still enabled, the only synchronising exit action is KO, abandoning the protocol with the mutual loss of deposits. Analogously for state 6, where it is still possible to synch on KO if the counterpart abandons, otherwise the preferred choice (carried out by a specific implementation of non-determinism) will be OK, back to state 0. Variables representing players' wealth are updated in the transitions from states 5 and 6.

Figure 3(b) shows a snapshot of the PRISM code that illustrates how the model is implemented. Each transition is labeled with the action name, if any, variable s represents the current state and some tests are performed in the selection of the next transition, e.g. the test `received = false`. Probabilistic choices are

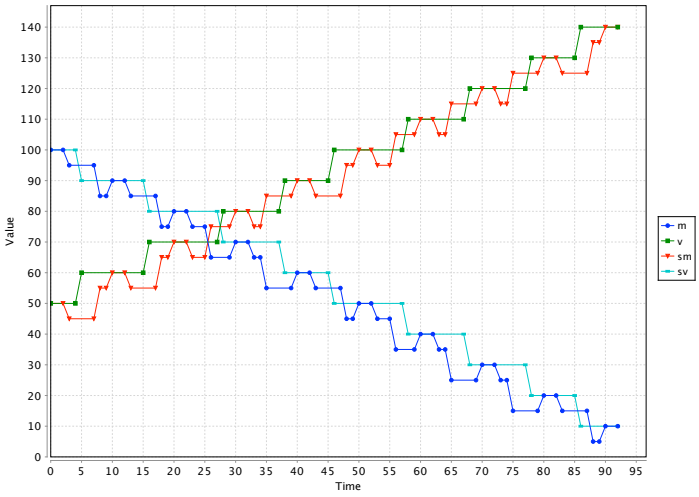


Fig. 4. Wealth preservation The figure shows a sample of protocol executions, with $m = 1000$ and $v = 500$, and $sm = 500$ and $sv = 1000$, the money and the value of the buyer and the seller, respectively. Both cannot run up a debt i.e. transactions would stop when one of the two runs out of money. The protocol is executed in "ideal" conditions: the ratio between the deposit and the price of objects does not make it convenient to break the protocol ($p = sp = 10$ and $d = sd = 10$), each player makes fully rational and protocol compliant choices all the time (e.g. there are no minimal probabilities to abandon the protocol as in the general model illustrated in Section 4.3). The graph shows the results of a series of about 100 transactions between a typical seller and a typical buyer, after which the sum of money and value is preserved for both players and therefore for the whole system.

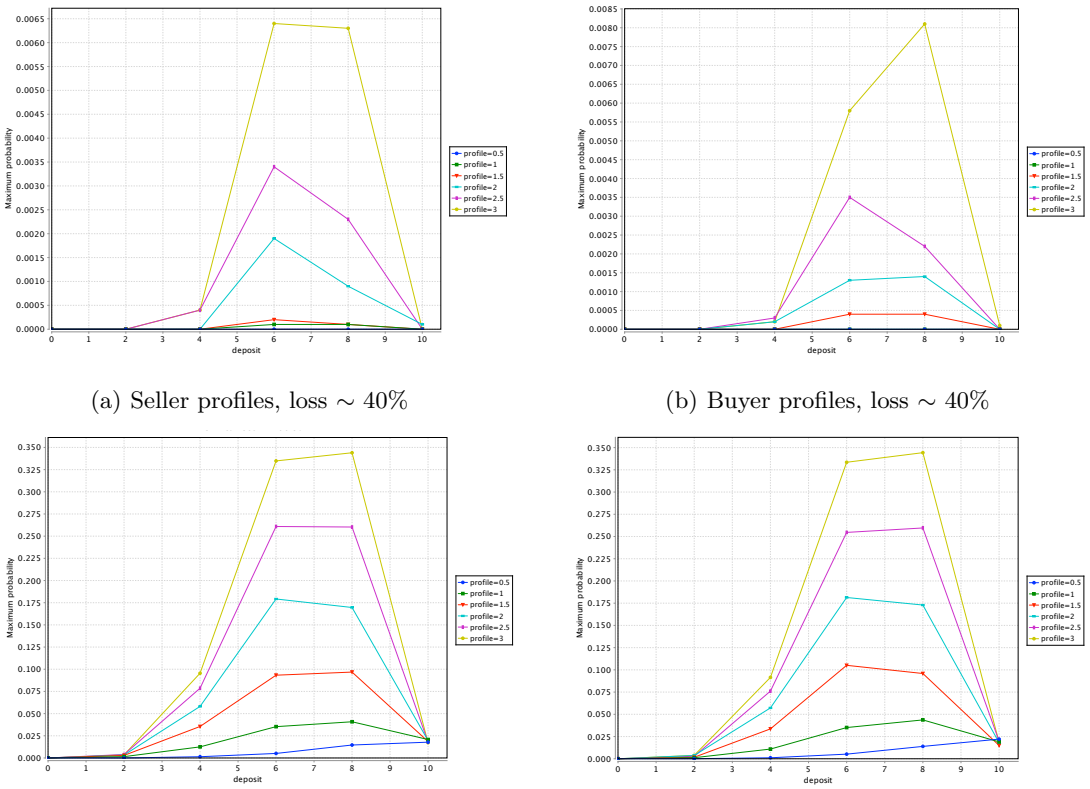
represented by the + operator and preceded by the associated probabilities, e.g. the expression `(received? _p_ok : 0.85)` in state 2.

Figure 3(c) shows the enabled actions in state 2 for both players. Here the choice is probabilistic for both. The action leading the seller to state 3 has been selected. Updates, including the state change, are shown for each action.

5 Validating DSCP

5.1 Model validation

A simple sanity check is presented in Figure 4 to provide validation of the defined model. This is based on the idea of wealth preservation within the system under the assumptions made and ideal players' behaviour (Section 2.3) and shows the expected fair execution of the protocol: all buyer's money is transformed into value and correspondingly seller's value is transformed into money with no loss.



(a) Seller profiles, loss $\sim 40\%$

(b) Buyer profiles, loss $\sim 40\%$

(c) Seller profiles, loss $\sim 30\%$

(d) Buyer profiles, loss $\sim 30\%$

Fig. 5. Deposit and profiles sensitivity Loss probability for different player profiles within 700 time units. Different curves represent different profiles. Profliles less than 1 tend to be honest, those bigger fraudulent.

5.2 Deposits, prices and players' profiles

Following the results of the game theory analysis and further considerations about adding realism to the model, the case of a deposit smaller than the price/value of goods is considered here. In this case, DSCP may loose some of its expected "enforced trust" effect on players. Under these assumptions it is also of interest to explore how players with different attitudes towards behaving fraudulently can perturbe the protocol.

Six profiles have been considered, for $r = 0.5$, inching towards honesty and trust, $r = 1$ simply depending on the ratio between d and p, and r assuming values in $\{1.5, 2, 2.5, 3\}$, a progressively more fraudulent attitude. The deposit assumes values in $\{0, 2, 4, 6, 8, 10\}$, against a price of 10.

Validation has been carried out using $PCTL$ logic to express the properties of interest and exploiting the statistical model checking facilities provided by

PRISM, [17]. Starting again from the idea of wealth preservation, we have investigated the probability of one of the two symmetric players, both with the same profile, reporting a loss of a certain percentage after a given amount of time. Specifically, the tested property was

$$Pmax =? [\ F < 700\ SellerLossX\ ]$$

which, informally, reads as *"What is the probability that the seller will loose about X% of their initial wealth within 700 time units?"* The predicate $SellerLossX$ identifies all the states where seller wealth has been reduced by $(X \pm 5)\%$. The operator F, *finally*, requires that the property $SellerLossX$ will eventually be satisfied in the states reached by the repeated execution of the protocol. This has to happen within 700 time units.

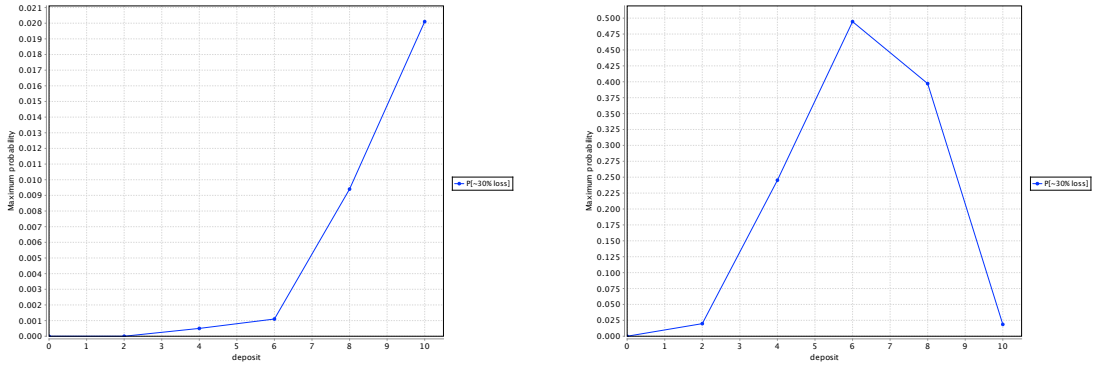
PRISM automatically validates such a formula by a statistical approach: a sufficiently large number of simulations is run in order to asses the desired probability, as required by the $Pmax =?$ operator (10,000 simulations in our case for each possible combination of profiles and deposits. Each simulation runs for at most 1500 time units).

Figures 5(a) and 5(b) report probabilities of loosing 40% of the initial wealth, while Figures 5(c) and 5(d) report probabilites of loosing 30% of the initial wealth for the seller and buyer, respectively. These results are about protocols run by players with the same profile. Seller and buyer results are symmetric in all the cases, as expected. The figures show a quite consistent probability of a 30% loss, up to about 0.3, and a tenfold lower probability for a 40% loss.

Quite interestingly, a very low deposit, e.g. from 0 to 2 in the 30% case, may incentivate fraudulent behaviour due to "lack of risk", however not much wealth is lost, because of the scarce impact of the loss of a minimal deposit. Analogously, a deposit close or equal to the price, 10 in this case, causes a phase switch as stealing becomes non-convenient (note the *min* operator in the definiton of the probability choices of players). A deposit close to the price, 3/4 of it say, results in being the most harmful situation, combining the probability of fraudulent behaviour with relevant loss following deposit loss.

Not surprisingly, profiles behave as expected. The honest one has a physiological minimal loss due to those minimal probabilities of abandoning the protocol even in potentially rewarding cases. Protocol executions in the presence of fraudulent players report larger losses, proportional to the r parameter.

These results, beyond explicating the details of the functioning of the protocol, can be used to determine preferred player behaviour when operating in untrustable environments. In those cases where a deposit equal to the price is unfeasible, some sort of trade-off analysis between deposit, profiles and levels of risk can be carried out through graphics like Figure 5. For instance one might wonder: *which is the maximum deposit within a range that allows keeping the risk of an X% loss below a given threshold? which is the minimum risk of such a loss for a range of deposits? which fraudulence profiles can be tolerated when one wants to keep the loss below a given threshold for a given deposit?*



(a) Fraudulent seller

(b) Honest buyer

Fig. 6. A fraudulent seller and an honest buyer Figures shows the probability of loosing about 30% for a fraudulent seller (a), who will leave the protocol whenever convenient with an high probability, and an honest buyer (b) who tend to follow the protocol. The curve is drawn for different deposits in $[0, 10]$ with 10 the price/value.

5.3 Being fraudulent pays off

As a final step, the different profiles are compared. This is done by letting an honest buyer ($r = 0.5$) interact with a quite fraudulent ($r = 1$) seller (symmetrical cases are analogous due to the symmetry of players). Results are reported in Figure 6 obtained by validating again the property of loosing about 30% of wealth within 700 time units.

The fraudulent seller, Figure 6(a), has an almost negligible probabilty of incurring in a 30% loss,[8] i.e. 0.02 in the worst case when $d = 10$ (which is when the "enforced trust" by the deposit forces players to behave fairly and loss are only due to the minimal probabilities of abandoning the protocol - Section 4.3).

For the honest buyer instead, Figure 6(b), the risk is consistently higher, with a peak half-way in the deposit scale, as expected. In the worst case the probability of incurring in a 30% loss is 0.5.

6 Conclusions

We analysed and validated DSCP, an idealised protocol for decentralised smart contracts, inspired by BitHalo. Several works can be found in literature on the security of virtual currencies such as BitCoin. Our focus is on the validation of players' behaviour while carrying out a smart contract. Our methodological

[8] It must be recalled that larger probabilities could hold for lesser-percentage losses, e.g. the seller could have a 0.1 probability of a 20%. Probabilities for different losses can be determined, if of interest.

approach to the validation of DSCP combines game theory and formal verification. Beyond the validation results, this kind of joint analysis applied to smart contracts is, to the best of our knowledge, an innovative aspect of the paper.

Game theoretic models provide an analysis of the behaviour of the players in the game/protocol under the assumption of perfect rationality and return maximization. Such models also suggest conditions for agreeing on the contract. The results are exploited to shape the actual behaviour of the system in a formal model for automated validation. Furthermore, formal methods support the modelling and simulation of aspects that game theory analysis calls out, noticeably, the cases in which players should not agree on the contract.

The reported results of the combined framework we adopted show that this is a promising approach, worth being further developed.

Several research directions are amenable to further investigations. More complex game theory models and tools could be considered such as repeated games, imperfect information, beliefs. As a consequence, smart contracts exhibiting more sophisticated behaviour could be modelled. For instance one could consider probability choices that may evolve as a result of the previous transactions, different levels of trust on the other players, and perceived utility of items larger/smaller than the agreed price. More complex and expressive models will also require a more sophisticated approach to parameter calibration. This is an interesting and open aspect of our approach, to be addressed in future work when larger datasets about the usage and performances of smart contracts will be available and hence usable to identify critical parameters.

Finally, smart contracts with multiple parties could be analysed through similar techniques as well.

References

1. Antonopoulos, A.: Mastering Bitcoin. O'Relly (2015)
2. Back, A.: Hashcash a denial of service counter-measure (2002), http://www.hashcash.org/papers/hashcash.pdf
3. Bartoletti, M., Cimoli, T., Zunino, R.: Compliance in behavioural contracts: a brief survey. In: Programming languages with applications to biology and security - Colloquium in honour of Pierpaolo Degano for his 65th birthday, C. Bodei, G.-L. Ferrari, C.Priami (eds) Lecture Notes in Computer Science, Springer. (2015), this volume.
4. Bartoletti, M., Scalas, A., Tuosto, E., Zunino, R.: Honesty by typing. In: Formal Techniques for Distributed Systems - Joint IFIP WG 6.1 International Conference, FMOODS/FORTE 2013, Held as Part of the 8th International Federated Conference on Distributed Computing Techniques, DisCoTec 2013, Florence, Italy, June 3-5, 2013. Proceedings. pp. 305–320 (2013), http://dx.doi.org/10.1007/978-3-642-38592-6_21
5. Basile, D., Degano, P., Ferrari, G.L.: Automata for analysing service contracts. In: Trustworthy Global Computing - 9th International Symposium, TGC 2014, Rome, Italy, September 5-6, 2014. Revised Selected Papers. pp. 34–50 (2014), http://dx.doi.org/10.1007/978-3-662-45917-1_3

6. Basile, D., Degano, P., Ferrari, G.L.: A formal framework for secure and complying services. The Journal of Supercomputing 69(1), 43–52 (2014), http://dx.doi.org/10.1007/s11227-014-1211-0
7. BitHalo: https://bithalo.org/
8. Bianco, A., de Alfaro, L.: Model checking of probabilistic and nondeterministic systems. In: Thiagarajan, P. (ed.) Proc. 15th Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS'95). LNCS, vol. 1026, pp. 499–513. Springer (1995)
9. BitcoinWiki: Atomic trading, https://en.bitcoin.it/wiki/atomic cross-chain trading
10. BitcoinWiki: Script, https://en.bitcoin.it/wiki/script
11. BitcoinWiki: Smart property, https://en.bitcoin.it/wiki/Smart_Property
12. Blockchain.info: Average transaction confirmation time, https://blockchain.info/charts/avg-confirmation-time
13. Dai, W.: b-money. http://www.weidai.com/bmoney.txt (1998)
14. Dolev, D., Yao, A.: On the security of public key protocols. IEEE Transactions on Information Theory 29(2), 198–208 (1983)
15. Forejt, V., Kwiatkowska, M., Norman, G., Parker, D.: Automated verification techniques for probabilistic systems. In: Bernardo, M., Issarny, V. (eds.) Formal Methods for Eternal Networked Software Systems (SFM'11). LNCS, vol. 6659, pp. 53–113. Springer (2011)
16. Hansson, H., Jonsson, B.: A logic for reasoning about time and reliability. Formal Aspects of Computing 6(5), 512–535 (1994)
17. Kwiatkowska, M., Norman, G., Parker, D.: PRISM 4.0: Verification of probabilistic real-time systems. In: Gopalakrishnan, G., Qadeer, S. (eds.) Proc. 23rd International Conference on Computer Aided Verification (CAV'11). LNCS, vol. 6806, pp. 585–591. Springer (2011)
18. Lawsky, B.M.: Bitlicense, http://www.dfs.ny.gov/legal/regulations/adoptions/dfsp200t.pdf
19. Lowe, G.: An attack on the Needham-Schroeder public-key authentication protocol. Information Processing Letters 56(3), 131–133 (November 1995)
20. Malinowski, D., Mazurek, L., Andrychowicz, M., Dziembowski, S.: Secure multiparty computations on bitcoin. IACR Cryptology ePrint Archive p. 784 (2013)
21. Muthoo, A.: Bargaining Theory with Applications. Cambridge University Press (1999)
22. Nakamoto, S.: Bitcoin: A peer-to-peer electronic cash system (2008), https://bitcoin.org/bitcoin.pdf
23. Needham, R., Schroeder, M.: Using encryption for authentication in large networks of computers. Communications of the ACM 21(12), 993–999 (December 1978)
24. Osborne, M.J., Rubinstein, A.: A Course in Game Theory. MIT Press (1994)
25. de Soto Polar, H.: The property rights project, http://www.ild.org.pe
26. Szabo, N.: Formalizing and securing relationships on public networks. First Monday (1997), http://firstmonday.org/ojs/index.php/fm/article/view/548/469
27. Szabo, N.: The idea of smart contracts (1997), http://szabo.best.vwh.net/smart_contracts_idea.html
28. VV.AA: Special report bitcoin. Bloomberg Briefs (2015), http://www.bloombergbriefs.com/
29. Zimbeck, D.: Two party double deposit trustless escrow in cryptographic networks and bitcoin (2014), https://bithalo.org/