Trust Is Risk:
A Decentralized Financial Trust Platform

Orfeas Stefanos Thyfronitis Litos[1] and Dionysis Zindros[2,⋆]

[1] National Technical University of Athens
[2] National and Kapodistrian University of Athens
olitos@corelab.ntua.gr, dionyziz@di.uoa.gr

Abstract. Centralized reputation systems use stars and reviews and thus require algorithm secrecy to avoid manipulation. In autonomous open source decentralized systems this luxury is not available. We create a reputation network for decentralized marketplaces where the trust each user gives to the other users is quantifiable and expressed in monetary terms. We introduce a new model for bitcoin wallets in which user coins are split among trusted associates. Direct trust is defined using shared bitcoin accounts via bitcoin's 1-of-2 multisig. Indirect trust is subsequently defined transitively. This enables formal game theoretic arguments pertaining to risk analysis. We prove that risk and maximum flows are equivalent in our model and that our system is Sybil-resilient. Our system allows for concrete financial decisions on the subjective monetary amount a pseudonymous party can be trusted with. Risk remains invariant under a direct trust redistribution operation followed by a purchase.

1 Introduction

Online marketplaces can be categorized as centralized and decentralized. Two examples of each category are ebay and OpenBazaar. The common denominator of established online marketplaces is that the reputation of each vendor and client is typically expressed in the form of stars and user-generated reviews that are viewable by the whole network.

The goal of "Trust Is Risk" is to offer a reputation system for decentralized marketplaces where the trust each user gives to the other users is quantifiable in monetary terms. The central assumption used throughout this paper is that trust is equivalent to risk, or the proposition that *Alice*'s *trust* in another user *Charlie* is defined as the *maximum sum of money Alice* can lose when *Charlie* is free to choose any strategy. To flesh out this concept, we will use *lines of credit* as proposed by Sanchez [1]. *Alice* joins the network by explicitly entrusting some money to another

⋆ Research supported by ERC project CODAMODA, project #259152

user, say her friend, *Bob* (see Fig. 1 and 2). If *Bob* has already entrusted some money to a third user, *Charlie*, then *Alice* indirectly trusts *Charlie* since if the latter wished to play unfairly, he could have already stolen the money entrusted to him by *Bob*. We will later see that *Alice* can now engage in economic interaction with *Charlie*.

To implement lines-of-credit, we use Bitcoin [2], a decentralized cryptocurrency that differs from conventional currencies in that it does not depend on trusted third parties. All transactions are public as they are recorded on a decentralized ledger, the blockchain. Each transaction takes some coins as input and produces some coins as output. If the output of a transaction is not connected to the input of another one, then this output belongs to the UTXO, the set of unspent transaction outputs. Intuitively, the UTXO contains all coins not yet spent.

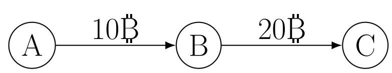

Fig. 1: A indirectly trusts C 10₿

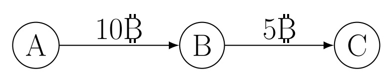

Fig. 2: A indirectly trusts C 5₿

We propose a new kind of wallet where coins are not exclusively owned, but are placed in shared accounts materialized through 1-of-2 multisigs, a bitcoin construct that permits any one of two pre-designated users to spend the coins contained within a shared account [3]. We use the notation $1/\{Alice, Bob\}$ to represent a 1-of-2 multisig that can be spent by either *Alice* or *Bob*. In this notation, the order of names is irrelevant, as either user can spend. However, the user who deposits the money initially into the shared account is relevant – she is the one risking her money.

Our approach changes the user experience in a subtle but drastic way. A user no more has to base her trust towards a store on stars or ratings which are not expressed in financial units. She can simply consult her wallet to decide whether the store is trustworthy and, if so, up to what value, denominated in bitcoin. This system works as follows: Initially *Alice* migrates her funds from her private bitcoin wallet to 1-of-2 multisig addresses shared with friends she comfortably trusts. We call this direct trust. Our system is agnostic to the means players use to determine who is trustworthy for these direct 1-of-2 deposits. Nevertheless, these deposits contain an objective value visible to the network that can be used to deterministically evaluate subjective indirect trust towards other users.

Suppose *Alice* is viewing the listings of vendor *Charlie*. Instead of his stars, *Alice* sees a positive value calculated by her wallet representing the maximum value she can safely pay to purchase from *Charlie*. This value, known as indirect trust, is calculated in Theorem 2 – Trust Flow.

Indirect trust towards a user is not global but subjective; each user views a personalized indirect trust based on the network topology. The indirect trust reported by our system maintains the following desired security property: If *Alice* makes a purchase from *Charlie*, then she is exposed to no more risk than she was already taking willingly. The existing voluntary risk is exactly that which *Alice* was taking by sharing her coins with her trusted friends. We prove this in Theorem 3 – Risk Invariance. Obviously it is not safe for *Alice* to buy anything from any vendor if she has not directly entrusted any value to other users.

In Trust Is Risk the money is not invested at the time of purchase and directly to the vendor, but at an earlier point in time and only to parties that are trustworthy for out of band reasons. The fact that this system can function in a completely decentralized fashion will become clear in the following sections. We prove this in Theorem 5 – Sybil Resilience.

We make the design choice that an entity can express her trust maximally in terms of her available capital. Thus, an impoverished player cannot allocate much direct trust to her friends, no matter how trustworthy they are. On the other hand, a rich player may entrust a small fraction of her funds to a player that she does not extensively trust and still exhibit more direct trust than the impoverished player. There is no upper limit to trust; each player is only limited by her funds. We thus take advantage of the following remarkable property of money: To normalise subjective human preferences into objective value.

A user has several incentives to join. First, she has access to otherwise inaccessible stores. Moreover, two friends can formalize their mutual trust by directly entrusting the same amount to each other. A company that casually subcontracts others can express its trust towards them. Governments can choose to directly entrust citizens with money and confront them using a corresponding legal arsenal if they make irresponsible use of this trust. Banks can provide loans as outgoing and manage savings as incoming direct trust. Last, the network is an investment and speculation field since it constitutes a new area for financial activity.

Observe that the same physical person can maintain multiple pseudonymous identities in the same trust network and that multiple independent trust networks for different purposes can coexist.

Trust Is Risk is not just a theoretical conception, but can be deployed and applied in existing decentralized markets such as OpenBazaar. All the necessary bitcoin constructs such as multisigs are readily available. Our only concern pertains to the scalability of such an implementation, but we are confident that such difficulties can be overcome.

2 Mechanics

We now trace *Alice*'s steps from joining the network to successfully completing a purchase. Suppose initially all her coins, say 10Ƀ, are under her exclusive control.

Two trustworthy friends, *Bob* and *Charlie*, persuade her to try out Trust Is Risk. She installs the Trust Is Risk wallet and migrates the 10Ƀ from her regular wallet, entrusting 2Ƀ to *Bob* and 5Ƀ to *Charlie*. She now exclusively controls 3Ƀ. She is risking 7Ƀ to which she has full but not exclusive access in exchange for being part of the network.

A few days later, she discovers an online shoes shop owned by *Dean*, also a member of Trust Is Risk. She finds a nice pair of shoes that costs 1Ƀ and checks *Dean*'s trustworthiness through her new wallet. Suppose *Dean* is deemed trustworthy up to 5Ƀ. Since 1Ƀ < 5Ƀ, she confidently proceeds to purchase the shoes with her new wallet.

She can then see in her wallet that her exclusive coins have remained 3Ƀ, the coins entrusted to *Charlie* have been reduced to 4Ƀ and *Dean* is entrusted 1Ƀ, equal to the value of the shoes. Also, her purchase is marked as pending. If she checks her trust towards *Dean*, it still is 5Ƀ. Under the hood, her wallet redistributed her entrusted coins in a way that ensures *Dean* is directly entrusted with coins equal to the value of the purchased item and that her reported trust towards him has remained invariant.

Eventually all goes well and the shoes reach *Alice*. *Dean* chooses to redeem *Alice*'s entrusted coins, so her wallet does not show any coins entrusted to *Dean*. Through her wallet, she marks the purchase as successful. This lets the system replenish the reduced trust to *Bob* and *Charlie*, setting the entrusted coins to 2Ƀ and 5Ƀ respectively once again. *Alice* now exclusively owns 2Ƀ. Thus, she can now use a total of 9Ƀ, which is expected, since she had to pay 1Ƀ for the shoes.

3 The Trust Graph

We now engage in the formal description of the proposed system, accompanied by helpful examples.

Definition 1 (Graph). *Trust Is Risk is represented by a sequence of directed weighted graphs* $(\mathcal{G}_j)$ *where* $\mathcal{G}_j = (\mathcal{V}_j, \mathcal{E}_j)\,, j \in \mathbb{N}$*. Also, since the graphs are weighted, there exists a sequence of weight functions* (c_j) *with* $c_j : \mathcal{E}_j \to \mathbb{R}^+$*.*

The nodes represent the players, the edges represent the existing direct trusts and the weights represent the amount of value attached to the corresponding direct trust. As we will see, the game evolves in turns. The subscript of the graph represents the corresponding turn.

Definition 2 (Players). *The set $\mathcal{V}_j = \mathcal{V}(\mathcal{G}_j)$ is the set of all players in the network, otherwise understood as the set of all pseudonymous identities.*

Each node has a corresponding non-negative number that represents its capital. A node's capital is the total value that the node possesses exclusively and nobody else can spend.

Definition 3 (Capital). *The capital of A in turn j, $Cap_{A,j}$, is defined as the number of coins that belong exclusively to A at the beginning of turn j.*

The capital is the value that exists in the game but is not shared with trusted parties. The capital of A can be reallocated only during her turns, according to her actions. We model the system in a way that no capital can be added in the course of the game through external means. The use of capital will become clear once turns are formally defined.

The formal definition of direct trust follows:

Definition 4 (Direct Trust). *Direct trust from A to B at the end of turn j, $DTr_{A \to B,j}$, is defined as the total finite amount that exists in $1/\{A,B\}$ multisigs in the UTXO in the end of turn j, where the money is deposited by A.*

$$DTr_{A \to B,j} = \begin{cases} c_j(A,B), & \textit{if } (A,B) \in \mathcal{E}_j \\ 0, & \textit{else} \end{cases} . \tag{1}$$

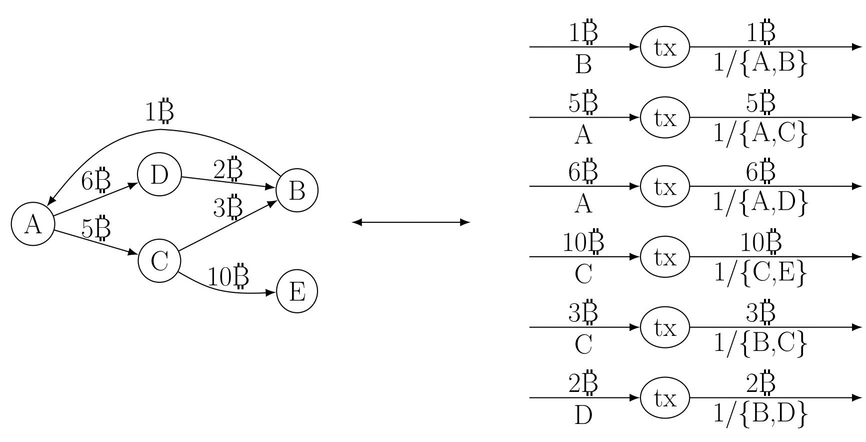

Fig. 3: Trust Is Risk Game Graph and Equivalent Bitcoin UTXO

The definition of direct trust agrees with the title of this paper and coincides with the intuition and sociological experimental results of Karlan et al. [4] that the trust *Alice* shows to *Bob* in real-world social networks corresponds to the extent of danger in which *Alice* is putting herself into in order to help *Bob*. An example graph with its corresponding transactions in the UTXO can be seen in Fig. 3.

Any algorithm that has access to the graph $\mathcal{G}_j$ has implicitly access to all direct trusts of this graph.

Definition 5 (Neighbourhood). *We use the notation* $N^+(A)_j$ *to refer to the nodes directly trusted by* A *at the end of turn* j *and* $N^-(A)_j$ *for the nodes that directly trust* A *at the end of turn* j*.*

$$\begin{aligned} N^+(A)_j &= \{B \in \mathcal{V}_j : DTr_{A \to B,j} > 0\} \ , \\ N^-(A)_j &= \{B \in \mathcal{V}_j : DTr_{B \to A,j} > 0\} \ . \end{aligned} \tag{2}$$

These are called out- and in-neighbourhood of A *on turn* j *respectively.*

Definition 6 (Total In/Out Direct Trust). *We use* $in_{A,j}, out_{A,j}$ *to refer to the total incoming and outgoing direct trust respectively.*

$$in_{A,j} = \sum_{v \in N^-(A)_j} DTr_{v \to A,j} \ , \qquad out_{A,j} = \sum_{v \in N^+(A)_j} DTr_{A \to v,j} \ . \tag{3}$$

Definition 7 (Assets). *Sum of A's capital and outgoing direct trust.*

$$As_{A,j} = Cap_{A,j} + out_{A,j} \ . \tag{4}$$

4 Evolution of Trust

Trust Is Risk is a game that runs indefinitely. In each turn, a player is chosen, decides what to play and, if valid, the chosen turn is executed.

Definition 8 (Turns). *In each turn* j *a player* $A \in \mathcal{V}, A = Player(j)$*, chooses one or more actions from the following two kinds:*
Steal(y_B, B)*: Steal value* y_B *from* $B \in N^-(A)_{j-1}$*, where* $0 \leq y_B \leq DTr_{B \to A,j-1}$*. Then set* $DTr_{B \to A,j} = DTr_{B \to A,j-1} - y_B$*.*
Add(y_B, B)*: Add value* y_B *to* $B \in \mathcal{V}$*, where* $-DTr_{A \to B,j-1} \leq y_B$*. Then set* $DTr_{A \to B,j} = DTr_{A \to B,j-1} + y_B$*.*

$y_B < 0$ *amounts to direct trust reduction, while* $y_B > 0$ *to direct trust increase.*

Let Y_{st}, Y_{add} be the total value to be stolen and added respectively by A. The capital is updated in every turn: $Cap_{A,j} = Cap_{A,j-1} + Y_{st} - Y_{add}$. For a turn to be valid we require $Cap_{A,j} \geq 0$ and $DTr_{A \to B,j} \geq 0$ and $DTr_{B \to A,j} \geq 0$. A player cannot choose two actions of the same kind against the same player in one turn. $Turn_j$ denotes the set of actions in turn j. The graph that emerges by applying the actions on $\mathcal{G}_{j-1}$ is $\mathcal{G}_j$.

Definition 9 (Prev/Next Turn). *Let $j \in \mathbb{N}$ be a turn with $Player(j) = A$. Define $prev(j)/next(j)$ as the previous/next turn A is chosen to play. Formally, let*

$$P = \{k \in \mathbb{N} : k < j \wedge Player(k) = A\} \text{ and}$$
$$N = \{k \in \mathbb{N} : k > j \wedge Player(k) = A\} \ .$$

Then we define $prev(j), next(j)$ as follows:

$$prev(j) = \begin{cases} \max P, & P \neq \emptyset \\ 0, & P = \emptyset \end{cases} \ , \ next(j) = \min N \ .$$

Definition 10 (Damage). *Let j be a turn such that $Player(j) = A$.*

$$Dmg_{A,j} = out_{A,prev(j)} - out_{A,j-1} \ . \quad (5)$$

We say that A has been stolen value $Dmg_{A,j}$ between $prev(j)$ and j. We omit turn subscripts if they are implied from the context.

Definition 11 (History). *We define History, $\mathcal{H} = (\mathcal{H}_j)$, as the sequence of all tuples containing the sets of actions and the corresponding player.*

$$\mathcal{H}_j = (Player(j), Turn_j) \ . \quad (6)$$

Knowledge of the initial graph $\mathcal{G}_0$, all players' initial capital and the history amount to full comprehension of the evolution of the game. Building on the example of Fig. 3, we can see the resulting graph when D plays

$$Turn_1 = \{Steal(1, A), Add(4, C), Add(-1, B)\} \ . \quad (7)$$

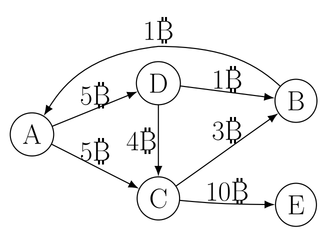

Fig. 4: Game Graph after $Turn_1$ (7) on the Graph of Fig. 3

We now define the Trust Is Risk Game formally. We assume players are chosen so that, after her turn, a player will eventually play again later.

`Trust Is Risk Game`
`j = 0`
`while (True)`
  `j += 1;` $A \xleftarrow{\$} \mathcal{V}_j$
  `Turn = strategy[`A`](`$\mathcal{G}_0$`,` A`,` $Cap_{A,0}$`,` $\mathcal{H}_{1\dots j-1}$`)`
  `(`$\mathcal{G}_j$`,` $Cap_{A,j}$`,` $\mathcal{H}_j$`) = executeTurn(`$\mathcal{G}_{j-1}$`,` A`,` $Cap_{A,j-1}$`, Turn)`

`strategy[`A`]()` provides player A with full knowledge of the game, except for the capitals of other players. This assumption may not be always realistic. `executeTurn()` checks the validity of `Turn` and substitutes it with an empty turn if invalid. Subsequently, it creates the new graph $\mathcal{G}_j$ and updates the history accordingly.

5 Trust Transitivity

In this section we define some strategies and show the corresponding algorithms. Then we define the Transitive Game, the worst-case scenario for an honest player when another player plays maliciously.

Definition 12 (Idle Strategy). *A player plays the idle strategy if she passes her turn.*

`Idle Strategy`
`Input : graph` $\mathcal{G}_0$`, player` A`, capital` $Cap_{A,0}$`, history` $(\mathcal{H})_{1\dots j-1}$
`Output :` $Turn_j$
`idleStrategy(`$\mathcal{G}_0$`,` A`,` $Cap_{A,0}$`,` $\mathcal{H}$`) :`
  `return(`$\emptyset$`)`

The inputs and outputs are identical to those of `idleStrategy()` for the rest of the strategies, thus we avoid repeating them.

Definition 13 (Evil Strategy). *A player plays the evil strategy if she steals all incoming direct trust and nullifies her outgoing direct trust.*

`evilStrategy(`$\mathcal{G}_0$`,` A`,` $Cap_{A,0}$`,` $\mathcal{H}$`) :`
  `Steals =` $\bigcup_{v \in N^-(A)_{j-1}} \{Steal(DTr_{v \rightarrow A,j-1}, v)\}$
  `Adds =` $\bigcup_{v \in N^+(A)_{j-1}} \{Add(-DTr_{A \rightarrow v,j-1}, v)\}$
  $Turn_j$ `= Steals` $\cup$ `Adds`
  `return(`$Turn_j$`)`

Definition 14 (Conservative Strategy). *A player plays conservatively if she replenishes the value she lost since the previous turn by stealing from others who directly trust her as much as she can up to Dmg_A.*

```
consStrategy($\mathcal{G}_0$, $A$, $Cap_{A,0}$, $\mathcal{H}$) :
  Damage = $out_{A,prev(j)}$ - $out_{A,j-1}$
  if (Damage > 0)
    if (Damage >= $in_{A,j-1}$)
      $Turn_j$ = $\bigcup_{v \in N^-(A)_{j-1}} \{Steal\left(DTr_{v \to A,j-1}, v\right)\}$
    else
      $y$ = SelectSteal($G_j$, $A$, Damage) #$y$ = $\{y_v : v \in N^-(A)_{j-1}\}$
      $Turn_j$ = $\bigcup_{v \in N^-(A)_{j-1}} \{Steal\left(y_v, v\right)\}$
  else $Turn_j$ = $\emptyset$
  return($Turn_j$)
```

`SelectSteal()` returns y_v with $v \in N^-(A)_{j-1}$ such that

$$\sum_{v \in N^-(A)_{j-1}} y_v = Dmg_{A,j} \quad \wedge \quad \forall v \in N^-(A)_{j-1}, y_v \leq DTr_{v \to A,j-1} \ . \quad (8)$$

Player A can arbitrarily define how `SelectSteal()` distributes the $Steal\,()$ actions each time she calls the function, as long as (8) is respected.

The rationale behind this strategy arises from a real-world common situation. Suppose there are a client, an intermediary and a producer. The client entrusts some value to the intermediary so that the latter can buy the desired product from the producer and deliver it to the client. The intermediary in turn entrusts an equal value to the producer, who needs the value upfront to be able to complete the production process. However the producer eventually does not give the product neither reimburses the value, due to bankruptcy or decision to exit the market with an unfair benefit. The intermediary can choose either to reimburse the client and suffer the loss, or refuse to return the money and lose the client's trust. The latter choice for the intermediary is exactly the conservative strategy. It is used throughout this work as a strategy for all the intermediary players because it models effectively the worst-case scenario that a client can face after an evil player decides to steal everything she can and the rest of the players do not engage in evil activity.

We continue with a possible evolution of the game, the Transitive Game.

```
Transitive Game
Input : graph $\mathcal{G}_0$, $A \in \mathcal{V}$ idle player, $B \in \mathcal{V}$ evil player
Angry = Sad = $\emptyset$ ; Happy = $\mathcal{V} \setminus \{A, B\}$
for ($v \in \mathcal{V} \setminus \{B\}$) $Loss_v$ = 0
j = 0
while (True)
  j += 1; $v \xleftarrow{\$} \mathcal{V} \setminus \{A\}$              # Choose this turn's player
  $Turn_j$ = strategy[$v$]($\mathcal{G}_0$, $v$, $Cap_{v,0}$, $\mathcal{H}_{1 \ldots j-1}$)
  executeTurn($\mathcal{G}_{j-1}$, $v$, $Cap_{v,j-1}$, $Turn_j$)
  for (action $\in Turn_j$)
    action match do
      case $Steal$(y, $w$) do                          # For each $Steal$,
        exchange = y                                   #
        $Loss_w$ += exchange                           # pass on $Loss$
        if ($v$ != $B$) $Loss_v$ -= exchange           #
        if ($w$ != $A$)                                # and change the
          Happy = Happy $\setminus \{w\}$              # mood of the
          if ($in_{w,j}$ == 0) Sad = Sad $\cup \{w\}$  # affected player
          else Angry = Angry $\cup \{w\}$
  if ($v$ != $B$)
    Angry = Angry $\setminus \{v\}$                    # Change the mood of
    if ($Loss_v$ > 0)  Sad = Sad $\cup \{v\}$          # the active player
    if ($Loss_v$ == 0) Happy = Happy $\cup \{v\}$
```

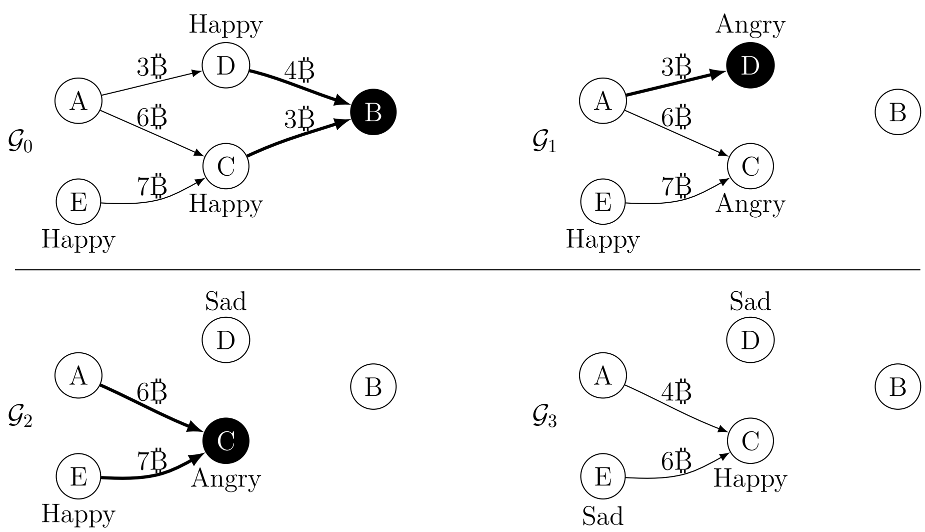

Fig. 5: B steals 7Ƀ, then D steals 3Ƀ and finally C steals 3Ƀ

In turn 0, there is already a network in place. All players apart from A and B follow the conservative strategy. The set of players is not modified throughout the Transitive Game, thus we can refer to $\mathcal{V}_j$ as $\mathcal{V}$. Each conservative player can be in one of three states: Happy, Angry or Sad. Happy players have 0 loss, Angry players have positive loss and positive incoming direct trust (line 17), thus are able to replenish their loss at least in part and Sad players have positive loss, but 0 incoming direct trust (line 16), thus they cannot replenish the loss. An example execution can be seen in Fig. 5.

Let j_0 be the first turn on which B is chosen to play. Until then, all players will pass their turn since nothing has been stolen yet (see the Conservative World theorem in Appendix A of the full version [5]). Moreover, let $v = Player(j)$. The Transitive Game generates turns:

$$Turn_j = \bigcup_{w \in N^-(v)_{j-1}} \{Steal\,(y_w, w)\} \ \text{, where} \tag{9}$$

$$\sum_{w \in N^-(v)_{j-1}} y_w = \min\,(in_{v,j-1}, Dmg_{v,j}) \ \ . \tag{10}$$

We see that if $Dmg_{v,j} = 0$, then $Turn_j = \emptyset$. From the definition of $Dmg_{v,j}$ and knowing that no strategy in this case can increase any direct trust, we see that $Dmg_{v,j} \geq 0$. Also $Loss_{v,j} \geq 0$ because if $Loss_{v,j} < 0$, then v has stolen more value than she has been stolen, thus she would not be following the conservative strategy.

6 Trust Flow

We can now define indirect trust from A to B.

Definition 15 (Indirect Trust). *Indirect trust from A to B after turn j is defined as the maximum possible value that can be stolen from A after turn j in the setting of* `TransitiveGame(`$\mathcal{G}_j$`,`A`,`B`)`.

Note that $Tr_{A \to B} \geq DTr_{A \to B}$. The next result shows $Tr_{A \to B}$ is finite.

Theorem 1 (Trust Convergence Theorem).
Consider a Transitive Game. There exists a turn such that all subsequent turns are empty.

Proof Sketch. If the game didn't converge, the $Steal\,()$ actions would continue forever without reduction of the amount stolen over time, thus they would reach infinity. However this is impossible, since there exists only finite total direct trust. □

Proofs of all theorems can be found in Appendix A of the full version [5].

In the setting of `TransitiveGame(`$\mathcal{G}$`,`A`,`B`)` and j being a turn in which the game has converged, we use the notation $Loss_A = Loss_{A,j}$. $Loss_A$ is not the same for repeated executions of this kind of game, since the order in which players are chosen may differ between executions and conservative players can choose which incoming direct trusts they will steal and how much from each.

Let G be a weighted directed graph. We investigate the maximum flow on it. For an introduction to maximum flows see Introduction to Algorithms, p. 708 [6]. Considering each edge's capacity as its weight, a flow assignment $X = [x_{vw}]_{\mathcal{V}\times\mathcal{V}}$ with source A and sink B is valid when:

$$\forall (v,w) \in \mathcal{E}, x_{vw} \leq c_{vw} \text{ and} \tag{11}$$

$$\forall v \in \mathcal{V} \setminus \{A,B\}, \sum_{w \in N^+(v)} x_{wv} = \sum_{w \in N^-(v)} x_{vw} \ . \tag{12}$$

The flow value is $\sum\limits_{v \in N^+(A)} x_{Av} = \sum\limits_{v \in N^-(B)} x_{vB}$. We do not suppose skew symmetry in X. There exists an algorithm $MaxFlow\left(A,B\right)$ that returns the maximum possible flow from A to B. This algorithm needs full knowledge of the graph and runs in $O\left(|\mathcal{V}||\mathcal{E}|\right)$ time [7]. We refer to the flow value of $MaxFlow\left(A,B\right)$ as $maxFlow\left(A,B\right)$.

We will now introduce two lemmas that will be used to prove one of the central results of this work, the Trust Flow theorem.

Lemma 1 (MaxFlows Are Transitive Games).
Let $\mathcal{G}$ be a game graph, let $A,B \in \mathcal{V}$ and $MaxFlow\left(A,B\right)$ the maximum flow from A to B executed on $\mathcal{G}$. There exists an execution of `TransitiveGame(`$\mathcal{G},A,B$`)` such that $maxFlow\left(A,B\right) \leq Loss_A$.

Proof Sketch. The desired execution of `TransitiveGame()` will contain all flows from the $MaxFlow\left(A,B\right)$ as equivalent $Steal\left(\right)$ actions. The players will play in turns, moving from B back to A. Each player will steal from his predecessors as much as was stolen from her. The flows and the conservative strategy share the property that the total input is equal to the total output. □

Lemma 2 (Transitive Games Are Flows).
Let $\mathcal{H} =$`TransitiveGame(`$\mathcal{G},A,B$`)` for some game graph $\mathcal{G}$ and $A,B \in \mathcal{V}$. There exists a valid flow $X = \{x_{wv}\}_{\mathcal{V}\times\mathcal{V}}$ on $\mathcal{G}_0$ such that $\sum\limits_{v\in\mathcal{V}} x_{Av} = Loss_A$.

Proof Sketch. If we exclude the sad players from the game, the $Steal\,()$ actions that remain constitute a valid flow from A to B. $\square$

Theorem 2 (Trust Flow Theorem).
Let $\mathcal{G}$ be a game graph and $A, B \in \mathcal{V}$. It holds that

$$Tr_{A \rightarrow B} = maxFlow\,(A, B) \quad .$$

Proof. From lemma 1 there exists an execution of the Transitive Game such that $Loss_A \geq maxFlow\,(A, B)$. Since $Tr_{A \rightarrow B}$ is the maximum loss that A can suffer after the convergence of the Transitive Game, we see that

$$Tr_{A \rightarrow B} \geq maxFlow\,(A, B) \quad . \tag{13}$$

But some execution of the Transitive Game gives $Tr_{A \rightarrow B} = Loss_A$. From lemma 2, this execution corresponds to a flow. Thus

$$Tr_{A \rightarrow B} \leq maxFlow\,(A, B) \quad . \tag{14}$$

The theorem follows from (13) and (14). $\square$

Note that the maxFlow is the same in the following two cases: If a player chooses the evil strategy and if that player chooses a variation of the evil strategy where she does not nullify her outgoing direct trust.

Further justification of trust transitivity through the use of $MaxFlow$ can be found in the sociological work by Karlan et al. [4] where a direct correspondence of maximum flows and empirical trust is experimentally validated.

Here we see another important theorem that gives the basis for risk-invariant transactions between different, possibly unknown, parties.

Theorem 3 (Risk Invariance Theorem). *Let $\mathcal{G}$ be a game graph, $A, B \in \mathcal{V}$ and l the desired value to be transferred from A to B, with $l \leq Tr_{A \rightarrow B}$. Let also $\mathcal{G}'$ with the same nodes as $\mathcal{G}$ such that*

$$\forall v \in \mathcal{V}' \setminus \{A\}, \forall w \in \mathcal{V}', DTr'_{v \rightarrow w} = DTr_{v \rightarrow w} \quad .$$

Furthermore, suppose that there exists an assignment for the outgoing direct trust of $A, DTr'_{A \rightarrow v}$, such that

$$Tr'_{A \rightarrow B} = Tr_{A \rightarrow B} - l \quad . \tag{15}$$

Let another game graph, $\mathcal{G}''$, be identical to $\mathcal{G}'$ except for the following change: $DTr''_{A \rightarrow B} = DTr'_{A \rightarrow B} + l$. It then holds that

$$Tr''_{A \rightarrow B} = Tr_{A \rightarrow B} \quad .$$

Proof. The two graphs $\mathcal{G}'$ and $\mathcal{G}''$ differ only in the weight of the edge (A, B), which is larger by l in $\mathcal{G}''$. Thus the two $MaxFlow$s will choose the same flow, except for (A, B), where it will be $x''_{AB} = x'_{AB} + l$. $\square$

A can reduce her outgoing direct trust in a manner that achieves (15), since $maxFlow\,(A, B)$ is continuous with respect to A's outgoing direct trusts.

7 Sybil Resilience

One of our aims is to mitigate Sybil attacks [8] whilst maintaining decentralized autonomy [9]. We begin by extending the definition of indirect trust.

Definition 16 (Indirect Trust to Multiple Players). *Indirect trust from player A to a set of players, $S \subset \mathcal{V}$ is defined as the maximum possible value that can be stolen from A if all players in S are evil, A is idle and everyone else $(\mathcal{V} \setminus (S \cup \{A\}))$ is conservative. Formally, let choices be the different actions between which the conservative players choose, then*

$$Tr_{A \to S, j} = \max_{j' : j' > j, choices} \left[out_{A,j} - out_{A,j'} \right] \ . \tag{16}$$

We now extend the Trust Flow theorem to many players.

Theorem 4 (Multi-Player Trust Flow).
Let $S \subset \mathcal{V}$ and T be an auxiliary player such that, for the sake of argument, $\forall B \in S, DTr_{B \to T} = \infty$. It holds that

$$\forall A \in \mathcal{V} \setminus S, Tr_{A \to S} = maxFlow\,(A, T) \ .$$

Proof. If T chooses the evil strategy and all players in S play according to the conservative strategy, they will have to steal all their incoming direct trust since they have suffered an infinite loss, thus they will act in a way identical to following the evil strategy as far as $MaxFlow$ is concerned. The theorem follows thus from the Trust Flow theorem. $\square$

We now define several useful notions to tackle the problem of Sybil attacks. Let Eve be a possible attacker.

Definition 17 (Corrupted Set). *Let $\mathcal{G}$ be a game graph and let Eve have a set of players $\mathcal{B} \subset \mathcal{V}$ corrupted, so that she fully controls their outgoing and incoming direct trusts with any player in $\mathcal{V}$. We call this the corrupted set. The players $\mathcal{B}$ are considered legitimate before the corruption, thus they may be directly trusted by any player in $\mathcal{V}$.*

Definition 18 (Sybil Set). *Let $\mathcal{G}$ be a game graph. Participation does not require registration, so Eve can create unlimited players. We call the set of these players $\mathcal{C}$, or Sybil set. Moreover, Eve controls their direct and indirect trusts with any player. However, players $\mathcal{C}$ can be directly trusted only by players $\mathcal{B} \cup \mathcal{C}$ but not by players $\mathcal{V} \setminus (\mathcal{B} \cup \mathcal{C})$, where $\mathcal{B}$ is the corrupted set.*

Definition 19 (Collusion). *Let $\mathcal{G}$ be a game graph. Let $\mathcal{B} \subset \mathcal{V}$ be a corrupted set and $\mathcal{C} \subset \mathcal{V}$ be a Sybil set. The tuple $(\mathcal{B}, \mathcal{C})$ is called collusion and is controlled by Eve.*

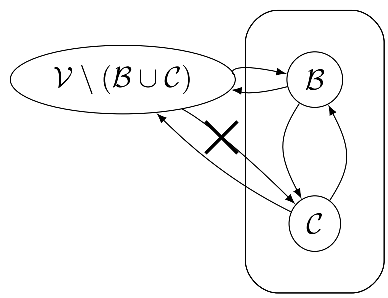

Fig. 6: Collusion

From a game theoretic point of view, players $\mathcal{V} \setminus (\mathcal{B} \cup \mathcal{C})$ perceive the collusion as independent players with a distinct strategy each, whereas in reality they are all subject to a single strategy dictated by Eve.

Theorem 5 (Sybil Resilience).
Let $\mathcal{G}$ be a game graph and $(\mathcal{B}, \mathcal{C})$ be a collusion of players on $\mathcal{G}$. It is

$$Tr_{A \to \mathcal{B} \cup \mathcal{C}} = Tr_{A \to \mathcal{B}} \quad .$$

Proof Sketch. The incoming trust to $\mathcal{B} \cup \mathcal{C}$ cannot be higher than the incoming trust to $\mathcal{B}$ since $\mathcal{C}$ has no incoming trust from $\mathcal{V} \setminus (\mathcal{B} \cup \mathcal{C})$. $\square$

We have proven that controlling $|\mathcal{C}|$ is irrelevant for Eve, thus Sybil attacks are meaningless. Note that the theorem does not reassure against deception attacks. Specifically, a malicious player can create several identities, use them legitimately to inspire others to deposit direct trust to these identities and then switch to the evil strategy, thus defrauding everyone that trusted the fabricated identities. These identities correspond to the corrupted set of players and not to the Sybil set because they have direct incoming trust from outside the collusion.

In conclusion, we have delivered on our promise of a Sybil-resilient decentralized financial trust system with invariant risk for purchases.

8 Related Work

Webs-of-trust can be used as a basis for trust as shown by Caronni [10]. PGP [11] implements one and Pathfinder [12] explores its transitive closure. Freenet [13] implements a transitive web-of-trust for fighting spam. Mui et al. [14] and Jøsang et al. [15] propose ways of calculating trust towards distant nodes. Vişan et al. [16] calculate trust in a hierarchical way. CA- and Byzantine-based [17] PKIs [18] and Bazaar [19] require central trusted third parties or at least authenticated membership. FIRE [20], CORE [21], Grünert et al. [22] and Repantis et al. [23] do not prove any Sybil resilience. All these systems define trust in a non-financial manner.

We agree with Gollmann [24] in that the meaning of trust should not be extrapolated. We adopted their advice and urge our readers to adhere to the definitions of *direct* and *indirect* trust as defined here.

Beaver [25] includes a trust model that, to discourage Sybil attacks, relies on fees, something we chose to avoid. Our motivating application for exploring trust in a decentralized setting is OpenBazaar, where transitive financial trust has previously been explored by Zindros [9]. That work however does not define trust as a monetary value. We are strongly inspired by Karlan et al. [4] who give a sociological justification for the central design choice of identifying trust with risk. We appreciate the work in TrustDavis [26], which proposes a financial trust system with transitivity and in which trust is defined as lines-of-credit, similar to us. We extended their work by using the blockchain for automated proofs-of-risk, a feature not available to them at the time.

Our conservative strategy and Transitive Game are similar to the mechanism proposed by Fugger [27] which is also financially transitive and is used by Ripple [28] and Stellar [29]. IOUs in those correspond to reversed edges of trust in our system. The critical difference is that our trust is expressed in a global currency and there is no money-as-debt. Furthermore, we proved that trust and maximum flows are equivalent, a direction not explored in their papers, even though it seems to hold for their systems as well.

9 Further Research

When a purchase is made, outgoing direct trust must be reduced such that (15) holds. Trust redistribution algorithms for this will be discussed in a future paper.

Our game is static. In a future dynamic setting, users should be able to play simultaneously, freely join, depart or disconnect temporarily from the network. An interesting analysis would involve modelling repeated purchases with the respective edge updates on the trust graph and treating trust on the network as part of the utility function. Other types of multisigs, such as 1-of-3, can be explored.

MaxFlow in our case needs complete network knowledge, which can lead to privacy issues [30]. Calculating the flows in zero knowledge remains an open question. SilentWhispers [31] and its centralized predecessor, PrivPay [32], offer insight into how privacy can be achieved.

A wallet implementation of our game on any blockchain is welcome. Experimental results can be harvested by a simulation or implementation of Trust Is Risk. Afterwards, our system can be used in decentralized social networks, such as Synereo [33], and other applications.

References

1. Sanchez W.: Lines of Credit. `https://gist.github.com/drwasho/2c40b91e169f55988618#part-3-web-of-credit` (2016)
2. Nakamoto S.: Bitcoin: A Peer-to-Peer Electronic Cash System (2008)
3. Antonopoulos A. M.: Mastering Bitcoin: Unlocking Digital Cryptocurrencies. O'Reilly Media, Inc. (2014)
4. Karlan D., Mobius M., Rosenblat T., Szeidl A.: Trust and social collateral. The Quarterly Journal of Economics: pp. 1307–1361 (2009)
5. Thyfronitis Litos O. S., Zindros D.: Trust Is Risk: A Decentralized Financial Trust Platform. IACR: Cryptology ePrint Archive (2017)
6. Cormen T. H., Leiserson C. E., Rivest R. L., Stein C.: Introduction to algorithms. MIT Press and McGraw-Hill: 3rd ed. (2009)
7. Orlin J. B.: Max Flows in O(nm) Time, or Better. In STOC '13 Proceedings of the forty-fifth annual ACM symposium on Theory of computing: pp. 765–774: ACM, New York: doi:10.1145/2488608.2488705 (2013)
8. Douceur J. R.: The Sybil Attack. International workshop on Peer-To-Peer Systems (2002)
9. Zindros D.: Trust in Decentralized Anonymous Marketplaces (2015)
10. Caronni G.: Walking the web of trust. In Enabling Technologies: Infrastructure for Collaborative Enterprises, IEEE 9th International Workshops: pp. 153–158 (2000)
11. Zimmermann P.: PGP Source Code and Internals. The MIT Press (1995)
12. Penning H. P.: PGP pathfinder. `pgp.cs.uu.nl`
13. Clarke I., Sandberg O., Wiley B., Hong T. W.: Freenet: A Distributed Anonymous Information Storage and Retrieval System. In H. Federrath (editor), Designing Privacy Enhancing Technologies: pp. 46–66: Springer-Verlag Berlin Heidelberg, Berkeley, USA (2001)
14. Mui L., Mohtashemi M., Halberstadt A.: A Computational Model of Trust and Reputation. In Proceedings of the 35th Annual Hawaii International Conference on System Sciences: pp. 2431–2439: IEEE (2002)
15. Jøsang A., Ismail R.: The Beta Reputation System. In Proceedings of the 15th Bled Electronic Commerce Conference (2002)

16. Vişan A., Pop F., Cristea V.: Decentralized Trust Management in Peer-to-Peer Systems. In 10th International Symposium on Parallel and Distributed Computing: pp. 232–239 (2011)
17. Lamport L., Shostak R., Pease M.: The Byzantine Generals Problem. In ACM Transaction on Programming Languages and Systems: vol. 4.3: pp. 382–401 (1982)
18. Adams C., Lloyd S.: Understanding PKI: concepts, standards, and deployment considerations. Addison-Wesley Professional (2003)
19. Post A., Shah V., Mislove A.: Bazaar: Strengthening User Reputations in Online Marketplaces. In Proceedings of NSDI'11: 8th USENIX Symposium on Networked Systems Design and Implementation: p. 183 (2011)
20. Huynh T. D., Jennings N. R., Shadbolt N. R.: An Integrated Trust and Reputation Model for Open Multi-Agent Systems. Autonomous Agents and Multi-Agent Systems: vol. 13(2), pp. 119–154 (2006)
21. Michiardi P., Molva R.: Core: a Collaborative Reputation Mechanism to Enforece Node Cooperation in Mobile Ad-hoc Networks. In Advanced Communications and Multimedia Security: pp. 107–121: Springer US (2002)
22. Grünert A., Hudert S., Köning S., Kaffille S., Wirtz G.: Decentralized Reputation Management for Cooperating Software Agents in Open Multi-Agent Systems. ITSSA: vol. 1(4), pp. 363–368 (2006)
23. Repantis T., Kalogeraki V.: Decentralized Trust Management for Ad-hoc Peer-to-Peer Networks. In Proceedings of the 4th International Workshop of Middleware for Pervasive and Ad-hoc Computing: p. 6: MPAC: ACM (2006)
24. Gollmann D.: Why trust is bad for security. Electronic notes in theoretical computer science: vol. 157(3), pp. 3–9 (2016)
25. Soska K., Kwon A., Christin N., Devadas S.: Beaver: A Decentralized Anonymous Marketplace with Secure Reputation (2016)
26. DeFigueiredo D. D. B., Barr E. T.: TrustDavis: A Non-Exploitable Online Reputation System. CEC: vol. 5, pp. 274–283 (2005)
27. Fugger R.: Money as IOUs in Social Trust Networks & A Proposal for a Decentralized Currency Network Protocol. http://archive.ripple-project.org/decentralizedcurrency.pdf (2004)
28. Schartz D., Youngs N., Britto A.: The Ripple protocol consensus algorithm. Ripple Labs Inc White Paper: vol. 5 (2014)
29. Mazieres D.: The stellar consensus protocol: A federated model for internet-level consensus. Stellar Development Foundation (2015)
30. Narayanan A., Shmatikov V.: De-anonymizing Social Networks. In Proceedings of the 30th Symposium on Security and Privacy: pp. 173–187: IEEE: 10.1109/SP.2009.22 (2009)
31. Malavolta G., Moreno-Sanchez P., Kate A., Maffei M.: SilentWhispers: Enforcing Security and Privacy in Decentralized Credit Networks (2016)
32. Moreno-Sanchez P., Kate A., Maffei M., Pecina K.: Privacy preserving payments in credit networks. In Network and Distributed Security Symposium (2015)
33. Konforty D., Adam Y., Estrada D., Meredith L. G.: Synereo: The Decentralized and Distributed Social Network (2015)