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. Through direct trust redistribution, the risk incurred from making a purchase from a pseudonymous vendor in this manner remains invariant.

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.

Our goal is to create 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 to be the *maximum sum of money* that *Alice* can lose when *Charlie* is free to choose any strategy he wants. To flesh out this concept, we will use *lines of credit* as proposed by Sanchez **?**. *Alice* joins the network by explicitly entrusting a certain amount of

⋆ Research supported by ERC project CODAMODA, project #259152

money to another user, say her friend, *Bob* (see Fig. 1 and 2). If *Bob* has already entrusted an amount of 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 **?**, 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.

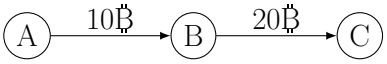

Fig.1: A indirectly trusts C 10₿

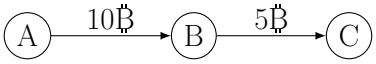

Fig.2: A indirectly trusts C 5₿

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 construction that permits any one of two pre-designated users to spend the coins contained within a shared account **?**. We will 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. This dubious kind of trust is confined to the direct neighbourhood of each player; indirect trust towards unknown users is calculated by a deterministic algorithm. In comparison, systems with global ratings do not distinguish between neighbours and other users, thus offering dubious trust indications for everyone.

Suppose that *Alice* is viewing the item listings of vendor *Charlie*. Instead of *Charlie*'s stars, *Alice* will see a positive value that is calculated by her wallet and represents the maximum monetary value that *Alice* can safely pay to complete a purchase from *Charlie*. This value, known as indirect trust, is calculated in Theorem 2 – Trust Flow. Note that 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 will not be safe for *Alice* to buy anything from *Charlie* or any other vendor if she has not directly entrusted any value to any other user.

We see that 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 find trustworthy to a great extent and still exhibit more direct trust than the impoverished player of the previous example. 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.

There are several incentives for a user to join this network. First, she has access to stores that would be inaccessible otherwise. Moreover, two friends can formalize their mutual trust by directly entrusting the same amount to each other. A large company that casually subcontracts other companies can express its trust towards them. A government can choose to directly entrust its citizens with money and confront them using a corresponding legal arsenal if they make irresponsible use of this trust. A bank can provide loans as outgoing and manage savings as incoming direct trust. Last but not least, the network can be viewed as a possible

investment and speculation field since it constitutes a completely new area for financial activity.

It is worth noting 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. On the other hand, the same pseudonymous identity can be used to establish trust in different contexts.

2 Mechanics

We will now trace *Alice*'s steps from joining the network to successfully completing a purchase. Suppose initially all her coins, say 10Ƀ, are stored in a way that she exclusively can spend them.

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 and is risking 7Ƀ in exchange for being part of the network. She has full but not exclusive access to the 7Ƀ entrusted to her friends and exclusive access to the remaining 3Ƀ, for a total of 10Ƀ.

A few days later, she discovers an online shoes shop owned by *Dean* who has also joined Trust Is Risk. She finds a nice pair of shoes that costs 1Ƀ and checks *Dean*'s trustworthiness through her new wallet. Suppose that *Dean* is deemed trustworthy up to 4Ƀ. Since 1Ƀ is less than 4Ƀ, she confidently proceeds to purchase the shoes by paying through her new wallet.

She can then see in her wallet that her exclusive coins have increased to 6Ƀ, the coins entrusted to *Bob* and *Charlie* have been reduced to 0.5Ƀ and 2.5Ƀ respectively and *Dean* is entrusted 1Ƀ, equal to the value of the shoes. Also, her purchase is marked as pending. If she proceeds to check her trust towards *Dean*, it will again be 4Ƀ. Under the hood, her wallet redistributed her entrusted coins in a way that ensures that *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 amount of value that exists in* $1/\{A, B\}$ *multisigs in the UTXO in the end of turn* j, *where the money is deposited by* A.

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

This definition agrees with the title of this paper and coincides with the intuition and sociological experimental results of **?** 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 below.

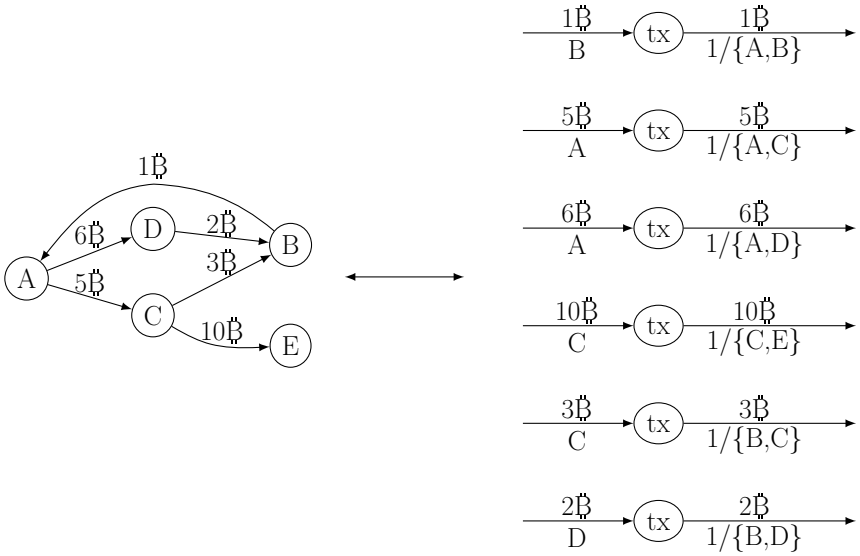

Fig.3: Trust Is Risk Game Graph and Equivalent Bitcoin UTXO

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.*

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

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

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:*

$$DTr_{B\to A,j} = DTr_{B\to A,j-1} - y_B \ .$$

***Add**(y_B, B): Add value y_B to $B \in \mathcal{V}$, where $-DTr_{A\to B,j-1} \leq y_B$. Then:*

$$DTr_{A\to B,j} = DTr_{A\to B,j-1} + y_B \ .$$

When $y_B < 0$, we say that A reduces her direct trust to B by $-y_B$. When $y_B > 0$, we say that A increases her direct trust to B by y_B. If $DTr_{A\to B,j-1} = 0$, then we say that A starts directly trusting B. A passes her turn if she chooses no action. Also, let Y_{st}, Y_{add} be the total value to be stolen and added respectively by A in her turn, j. For a turn to be feasible:

$$Y_{add} - Y_{st} \leq Cap_{A,j-1} \ . \tag{5}$$

The capital is updated in every turn: $Cap_{A,j} = Cap_{A,j-1} + Y_{st} - Y_{add}$.

A player cannot choose two actions of the same kind against the same player in one turn. The set of actions of a player in turn j is denoted by $Turn_j$. The graph that emerges by applying the actions on $\mathcal{G}_{j-1}$ is $\mathcal{G}_j$.

For example, let $A = Player(j)$. A valid turn can be

$$Turn_j = \{Steal(x, B), Add(y, C), Add(w, D)\} \ .$$

The *Steal* action requires $0 \leq x \leq DTr_{B\to A,j-1}$, the *Add* actions require $DTr_{A\to C,j-1} \geq -y$ and $DTr_{A\to D,j-1} \geq -w$ and the *Cap* restriction requires $y + w - x \leq Cap_{A,j-1}$.

We use $prev(j)$ and $next(j)$ to denote the previous and next turn respectively played by $Player(j)$.

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:*

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

$next(j)$ *is always well defined with the assumption that after each turn eventually everybody plays.*

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

$$Dmg_{A,j} = out_{A,prev(j)} - out_{A,j-1} \quad . \tag{6}$$

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.*

$$\mathcal{H}_j = (Player(j), Turn_j) \quad . \tag{7}$$

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

$$Turn_1 = \{Steal(1, A), Add(4, C), Add(5, B)\} \quad . \tag{8}$$

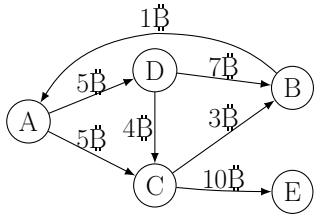

Fig.4: Game Graph after $Turn_1$ (8) on the Graph of Fig. 3

Trust Is Risk is controlled by an algorithm that chooses a player, receives the turn that this player wishes to play and, if this turn is valid, executes it. These steps are repeated indefinitely. We assume players are chosen in a way that, after her turn, a player will eventually play again later.

```
Trust Is Risk Game
1 j = 0
2 while (True)
3   j += 1; A ←$ V_j
4   Turn = strategy[A](G_0, A, Cap_{A,0}, H_{1...j−1})
5   (G_j, Cap_{A,j}, H_j) = executeTurn(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. For the routine code, see the Appendix.

5 Trust Transitivity

In this section we define some strategies and show the corresponding algorithms. Then we define the Transitive Game that represents the worst-case scenario for an honest player when another player decides to depart from the network with her money and all the money directly entrusted to her.

Definition 12 (Idle Strategy). *A player A is said to follow the idle strategy if she passes in her turn.*

```
Idle Strategy
Input : graph G_0, player A, capital Cap_{A,0}, history (H)_{1...j−1}
Output : Turn_j
1 idleStrategy(G_0, A, Cap_{A,0}, H) :
2   return(∅)
```

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 A is said to follow the evil strategy if she steals all incoming direct trust and nullifies her outgoing direct trust in her turn.*

```
1 evilStrategy(G_0, A, Cap_{A,0}, H) :
2   Steals = ⋃_{v∈N^−(A)_{j−1}} {Steal(DTr_{v→A,j−1}, v)}
3   Adds = ⋃_{v∈N^+(A)_{j−1}} {Add(−DTr_{A→v,j−1}, v)}
```

```
4    Turn_j = Steals ∪ Adds
5    return(Turn_j)
```

Definition 14 (Conservative Strategy). *Player A is said to follow the conservative strategy if she replenishes the value she lost since the previous turn, Dmg_A, by stealing from others that directly trust her as much as she can up to Dmg_A and she takes no other action.*

```
1  consStrategy(G_0, A, Cap_{A,0}, H) :
2    Damage = out_{A,prev(j)} - out_{A,j-1}
3    if (Damage > 0)
4      if (Damage >= in_{A,j-1})
5        Turn_j = ⋃_{v∈N^-(A)_{j-1}} {Steal(DTr_{v→A,j-1}, v)}
6      else
7        y = SelectSteal(G_j, A, Damage) #y = {y_v : v ∈ N^-(A)_{j-1}}
8        Turn_j = ⋃_{v∈N^-(A)_{j-1}} {Steal(y_v, v)}
9    else Turn_j = ∅
10   return(Turn_j)
```

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

$$\sum_{v \in N^-(A)_{j-1}} y_v = Dmg_{A,j} \ \wedge \ \forall v \in N^-(A)_{j-1}, y_v \leq DTr_{v \to A, j-1} \ . \quad (9)$$

Player A can arbitrarily define how `SelectSteal()` distributes the $Steal()$ actions each time she calls the function, as long as (9) is respected.

As we can see, the definition covers a multitude of options for the conservative player, since in case $0 < Dmg_{A,j} < in_{A,j-1}$ she can choose to distribute the $Steal()$ actions in any way she chooses.

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 very useful possible evolution of the game, the Transitive Game. In turn 0, there is already a network in place. All players apart from A and B follow the conservative strategy. Furthermore, the set of players is not modified throughout the Transitive Game, thus we can refer to $\mathcal{V}_j$ for any turn j as $\mathcal{V}$. Moreover, 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, thus are able to replenish their loss at least in part and Sad players have positive loss, but 0 incoming direct trust, thus they cannot replenish the loss. These conventions will hold whenever we use the Transitive Game.

```
Transitive Game
Input : graph G_0, A ∈ V idle player, B ∈ V evil player
1  Angry = Sad = ∅ ; Happy = V \ {A, B}
2  for (v ∈ V \ {B}) Loss_v = 0
3  j = 0
4  while (True)
5    j += 1; v <-$ V \ {A}
6    Turn_j = strategy[v](G_0, v, Cap_{v,0}, mathcalH_{1...j-1})
7    executeTurn(G_{j-1}, v, Cap_{v,j-1}, Turn_j)
8    for (action ∈ Turn_j)
9      action match do
10       case Steal(y, w) do
11         exchange = y
12         Loss_w += exchange
13         if (v != B) Loss_v -= exchange
14         if (w != A)
15           Happy = Happy \ {w}
16           if (in_{w,j} == 0) Sad = Sad ∪ {w}
17           else Angry = Angry ∪ {w}
18   if (v != B)
19     Angry = Angry \ {v}
20     if (Loss_v > 0)  Sad = Sad ∪ {v}       #in_{v,j} should be zero
21     if (Loss_v == 0) Happy = Happy ∪ {v}
```

An example execution follows:

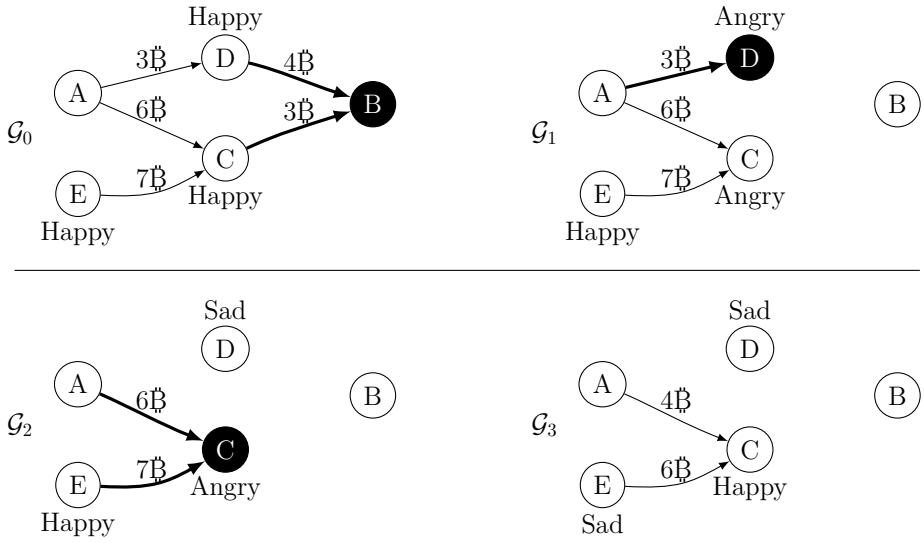

Fig.5: B steals 7Ƀ, then D steals 3Ƀ and finally C steals 3Ƀ

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 Appendix (Theorem 6) for a formal proof of this simple fact). Moreover, let $v = Player(j)$. The Transitive Game generates turns:

$$Turn_j = \bigcup_{w \in N^-(v)_{j-1}} \{Steal\,(y_w, w)\} \text{ , where} \tag{10}$$

$$\sum_{w \in N^-(v)_{j-1}} y_w = \min\left(in_{v,j-1}, Dmg_{v,j}\right) \ . \tag{11}$$

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, it is $Loss_{v,j} \geq 0$ 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 the indirect trust from A to B.

Definition 15 (Indirect Trust). *The 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 theorem shows that $Tr_{A\to B}$ 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. □

Full proofs of all theorems and lemmas can be found in the Appendix.

In the setting of `TransitiveGame(`$\mathcal{G}$`,`A`,`B`)`, we make use of the notation $Loss_A = Loss_{A,j}$, where j is a turn in which the game has converged. It is important to note that $Loss_A$ 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 the conservative players are free to choose which incoming direct trusts they will steal and how much from each.

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

$$\forall (v,w) \in \mathcal{E}, x_{vw} \leq c_{vw} \text{ and} \tag{12}$$

$$\forall v \in \mathcal{V} \setminus \{A,B\}, \sum_{w\in N^+(v)} x_{wv} = \sum_{w\in N^-(v)} x_{vw} \ . \tag{13}$$

We do not suppose any skew symmetry in X. The flow value is $\sum_{v\in N^+(A)} x_{Av}$, which is proven to be equal to $\sum_{v\in N^-(B)} x_{vB}$. There exists an algorithm that returns the maximum possible flow from A to B, namely $MaxFlow\,(A,B)$. This algorithm evidently needs full knowledge of the graph. The fastest version of this algorithm runs in $O\,(|\mathcal{V}||\mathcal{E}|)$ time **?**. We refer to the flow value of $MaxFlow\,(A,B)$ as $maxFlow\,(A,B)$.

We will now introduce two lemmas that will be used to prove the 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 $MaxFlow\,(A,B)$ the maximum flow from A to B executed on $\mathcal{G}$. There exists an execution of `TransitiveGame(`$\mathcal{G}, A, B$`)` such that $maxFlow\,(A,B) \leq Loss_A$.

Proof Sketch. The desired execution of `TransitiveGame()` will contain all flows from the $MaxFlow\,(A, B)$ as equivalent $Steal\,()$ 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 $\sum_{v\in\mathcal{V}} x_{Av} = Loss_A$.

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

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

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

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

$$Tr_{A\to B} \geq maxFlow\,(A, B) \quad . \tag{14}$$

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

$$Tr_{A\to B} \leq maxFlow\,(A, B) \quad . \tag{15}$$

The theorem follows from (14) and (15). □

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 conducted in **?** 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 $l \leq Tr_{A\to B}$. Let also $\mathcal{G}'$ with the same nodes as $\mathcal{G}$ such that*

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

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

$$Tr'_{A\to B} = Tr_{A\to B} - l \ . \tag{16}$$

Let another game graph, $\mathcal{G}''$, be identical to $\mathcal{G}'$ except for the following change:

$$DTr''_{A\to B} = DTr'_{A\to B} + l \ .$$

It then holds that

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

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 $x''_{AB} = x'_{AB} + l$. □

It is intuitively obvious that it is possible for A to reduce her outgoing direct trust in a manner that achieves (16), since $maxFlow\,(A, B)$ is continuous with respect to A's outgoing direct trusts. We leave this calculation as part of further research.

7 Sybil Resilience

One of the primary aims of this system is to mitigate the danger for Sybil attacks **?** whilst maintaining fully decentralized autonomy.

Here we extend the definition of indirect trust to many players.

Definition 16 (Indirect Trust to Multiple Players). *The 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 follow the evil strategy, A follows the idle strategy and everyone else ($\mathcal{V} \setminus (S \cup \{A\})$) follows the conservative strategy. More formally, let choices be the different actions between which the conservative players can choose, then*

$$Tr_{A\to S,j} = \max_{j':j'>j,choices} \left[out_{A,j} - out_{A,j'}\right] \ . \tag{17}$$

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 $\forall B \in S, DTr_{B \to T} = \infty$. It holds that

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

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. □

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 direct trusts to any player in $\mathcal{V}$ and can also steal all incoming direct trust to players in $\mathcal{B}$. We call this the corrupted set. The players $\mathcal{B}$ are considered to be 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. Since participation in the network does not require any kind of registration, Eve can create any number of players. We will call the set of these players $\mathcal{C}$, or Sybil set. Moreover, Eve can arbitrarily set the direct trusts of any player in $\mathcal{C}$ to any player and can also steal all incoming direct trust to players in $\mathcal{C}$. 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 a set of players corrupted by Eve.*

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, both controlled by Eve. The tuple $(\mathcal{B}, \mathcal{C})$ is called a collusion and is entirely controlled by a single entity in the physical world. 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 the controlling entity, Eve.*

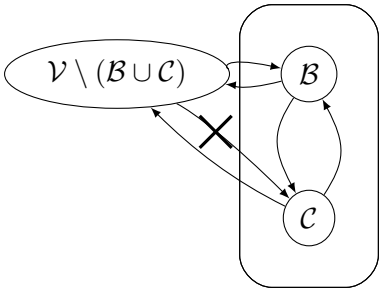

Fig.6: Collusion

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

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

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 $\mathcal{V} \setminus (\mathcal{B} \cup \mathcal{C})$. □
We have proven that controlling $|\mathcal{C}|$ is irrelevant for Eve, thus Sybil attacks are meaningless. We note that this theorem does not deliver reassurances against attacks involving deception techniques. More 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 successfully delivered our promise for a Sybil-resilient decentralized financial trust system with invariant risk for purchases.

8 Related Work

The topic of trust has been repeatedly attacked with several approaches: Purely cryptographic infrastructure where trust is rather binary and transitivity is limited to one step beyond actively trusted parties is explored in PGP **?**. A transitive web-of-trust for fighting spam is explored in Freenet **?**. Other systems require central trusted third parties, such as CA-based PKIs **?** and Bazaar **?**, or, in the case of Byzantine Fault Tolerance, authenticated membership **?**. While other trust systems attempt to be decentralized, they do not prove any Sybil resilience properties and hence

may be Sybil attackable. Such systems are FIRE **?**, CORE **?** and others **???**. Other systems that define trust in a non-financial way are **???????**.

We agree with the work of **?** in that the meaning of trust should not be extrapolated. We have adopted their advice in our paper and urge our readers to adhere to the definitions of *direct* and *indirect* trust as they are used here.

The Beaver marketplace **?** includes a trust model that relies on fees to discourage Sybil attacks. We chose to avoid fees in our system and mitigate Sybil attacks in a different manner. Our motivating application for exploring trust in a decentralized setting is the OpenBazaar marketplace. Transitive financial trust for OpenBazaar has previously been explored by **?**. That work however does not define trust as a monetary value. We are strongly inspired by **?** which gives a sociological justification for the central design choice of identifying trust with risk. We greatly appreciate the work in TrustDavis **?**, which proposes a financial trust system that exhibits transitive properties and in which trust is defined as lines-of-credit, similar to our system. We were able to extend 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 very similar to the mechanism proposed by the economic paper **?** which also illustrates financial trust transitivity and is used by Ripple **?** and Stellar **?**. IOUs in these correspond to reversed edges of trust in our system. The critical difference is that our denominations of trust are expressed in a global currency and that coins must pre-exist in order to be trusted and so there is no money-as-debt. Furthermore, we prove that trust and maximum flows are equivalent, a direction not explored in their paper, even though we believe it must hold for both our and their systems.

9 Further Research

When *Alice* makes a purchase from *Bob*, she has to reduce her outgoing direct trust in a manner such that the supposition (16) of Risk Invariance theorem is satisfied. How *Alice* can recalculate her outgoing direct trust 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. Other types of multisigs, such as 1-of-3, can be explored for the implementation of multi-party direct trust.

MaxFlow in our case needs complete network knowledge, which can lead to privacy issues through deanonymisation techniques **?**. Calculating the flows in zero knowledge remains an open question. **?** and its centralized predecessor, PrivPay **?**, seem to offer invaluable insight into how privacy can be achieved.

Our game theoretic analysis is simple. 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.

An implementation as a wallet on any blockchain of our financial game is most welcome. A simulation or actual implementation of Trust Is Risk, combined with analysis of the resulting dynamics can yield interesting experimental results. Subsequently, our trust network can be used in other applications, such as decentralized social networks **?**.

Appendix

1 Proofs, Lemmas and Theorems

Lemma 3 (*Loss* Equivalent to *Damage*).
Consider a Transitive Game. Let $j \in \mathbb{N}$ and $v = Player(j)$ such that v is following the conservative strategy. It holds that

$$\min(in_{v,j}, Loss_{v,j}) = \min(in_{v,j}, Damage_{v,j}) \quad .$$

Proof.
Case 1: Let $v \in Happy_{j-1}$. Then

1. $v \in Happy_j$ because $Turn_j = \emptyset$,
2. $Loss_{v,j} = 0$ because otherwise $v \notin Happy_j$,
3. $Damage_{v,j} = 0$, or else any reduction in direct trust to v would increase equally $Loss_{v,j}$ (line 12), which cannot be decreased again but during an Angry player's turn (line 13).
4. $in_{v,j} \geq 0$

Thus

$$\min(in_{v,j}, Loss_{v,j}) = \min(in_{v,j}, Damage_{v,j}) = 0 \quad .$$

Case 2: Let $v \in Sad_{j-1}$. Then

1. $v \in Sad_j$ because $Turn_j = \emptyset$,
2. $in_{v,j} = 0$ (line 20),
3. $Damage_{v,j} \geq 0 \wedge Loss_{v,j} \geq 0$.

Thus

$$\min\left(in_{v,j}, Loss_{v,j}\right) = \min\left(in_{v,j}, Damage_{v,j}\right) = 0 \ .$$

If $v \in Angry_{j-1}$ then the same argument as in cases 1 and 2 hold when $v \in Happy_j$ and $v \in Sad_j$ respectively if we ignore the argument (1). Thus the theorem holds in every case. □

Proof of Theorem 1: Trust Convergence

First of all, after turn j_0 player E will always pass her turn because she has already nullified her incoming and outgoing direct trusts in $Turn_{j_0}$, the evil strategy does not contain any case where direct trust is increased or where the evil player starts directly trusting another player and the other players do not follow a strategy in which they can choose to $Add\,()$ direct trust to E. The same holds for player A because she follows the idle strategy. As far as the rest of the players are concerned, consider the Transitive Game. As we can see from lines 2 and 12 - 13, it is

$$\forall j, \sum_{v \in \mathcal{V}_j} Loss_v = in_{E,j_0-1} \ .$$

In other words, the total loss is constant and equal to the total value stolen by E. Also, as we can see in lines 1 and 20, which are the only lines where the Sad set is modified, once a player enters the Sad set, it is impossible to exit from this set. Also, we can see that players in $Sad \cup Happy$ always pass their turn. We will now show that eventually the $Angry$ set will be empty, or equivalently that eventually every player will pass their turn. Suppose that it is possible to have an infinite amount of turns in which players do not choose to pass. We know that the number of nodes is finite, thus this is possible only if

$$\exists j' : \forall j \geq j', |Angry_j \cup Happy_j| = c > 0 \wedge Angry_j \neq \emptyset \ .$$

This statement is valid because the total number of angry and happy players cannot increase because no player leaves the Sad set and if it were to be decreased, it would eventually reach 0. Since $Angry_j \neq \emptyset$, a player v that will not pass her turn will eventually be chosen to play. According to the Transitive Game, v will either deplete her incoming direct trust and enter the Sad set (line 20), which is contradicting $|Angry_j \cup Happy_j| = c$, or will steal enough value to enter the $Happy$ set, that is v will achieve $Loss_{v,j} = 0$. Suppose that she has stolen m players. They, in their turn, will steal total value at least equal to the value stolen by v (since they cannot go sad, as explained above). However, this means that, since the

total value being stolen will never be reduced and the turns this will happen are infinite, the players must steal an infinite amount of value, which is impossible because the direct trusts are finite in number and in value. More precisely, let j_1 be a turn in which a conservative player is chosen and

$$\forall j \in \mathbb{N}, DTr_j = \sum_{w,w' \in \mathcal{V}} DTr_{w \to w',j} \ .$$

Also, without loss of generality, suppose that

$$\forall j \geq j_1, out_{A,j} = out_{A,j_1} \ .$$

In $Turn_{j_1}$, v steals

$$St = \sum_{i=1}^{m} y_i \ .$$

We will show using induction that

$$\forall n \in \mathbb{N}, \exists j_n \in \mathbb{N} : DTr_{j_n} \leq DTr_{j_1 - 1} - nSt \ .$$

Base case: It holds that

$$DTr_{j_1} = DTr_{j_1 - 1} - St \ .$$

Eventually there is a turn j_2 when every player in $N^-(v)_{j-1}$ will have played. Then it holds that

$$DTr_{j_2} \leq DTr_{j_1} - St = DTr_{j_1 - 1} - 2St \ ,$$

since all players in $N^-(v)_{j-1}$ follow the conservative strategy, except for A, who will not have been stolen anything due to the supposition.

Induction hypothesis: Suppose that

$$\exists k > 1 : j_k > j_{k-1} > j_1 \Rightarrow DTr_{j_k} \leq DTr_{j_{k-1}} - St \ .$$

Induction step: There exists a subset of the $Angry$ players, S, that have been stolen at least value St in total between the turns j_{k-1} and j_k, thus there exists a turn j_{k+1} such that all players in S will have played and thus

$$DTr_{j_{k+1}} \leq DTr_{j_k} - St \ .$$

We have proven by induction that

$$\forall n \in \mathbb{N}, \exists j_n \in \mathbb{N} : DTr_{j_n} \leq DTr_{j_1 - 1} - nSt \ .$$

However

$$DTr_{j_1-1} \geq 0 \wedge St > 0 \ ,$$

thus

$$\exists n' \in \mathbb{N} : n'St > DTr_{j_1-1} \Rightarrow DTr_{j_{n'}} < 0 \ .$$

We have a contradiction because

$$\forall w, w' \in \mathcal{V}, \forall j \in \mathbb{N}, DTr_{w \to w', j} \geq 0 \ ,$$

thus eventually $Angry = \emptyset$ and everybody passes. □

Proof of Lemma 1: MaxFlows Are Transitive Games

We suppose that the turn of $\mathcal{G}$ is 0. In other words, $\mathcal{G} = \mathcal{G}_0$. Let $X = \{x_{vw}\}_{\mathcal{V}\times\mathcal{V}}$ be the flows returned by $MaxFlow\,(A, B)$. For any graph G there exists a $MaxFlow$ that is a DAG. We can easily prove this using the Flow Decomposition theorem **?**, which states that each flow can be seen as a finite set of paths from A to B and cycles, each having a certain flow. We execute $MaxFlow\,(A, B)$ and we apply the aforementioned theorem. The cycles do not influence the $maxFlow\,(A, B)$, thus we can remove these flows. The resulting flow is a $MaxFlow\,(A, B)$ without cycles, thus it is a DAG. Topologically sorting this DAG, we obtain a total order of its nodes such that $\forall$ nodes $v, w \in \mathcal{V} : v < w \Rightarrow x_{wv} = 0$ **?**. Put differently, there is no flow from larger to smaller nodes. B is maximum since it is the sink and thus has no outgoing flow to any node and A is minimum since it is the source and thus has no incoming flow from any node. The desired execution of Transitive Game will choose players following the total order inversely, starting from player B. We observe that $\forall v \in \mathcal{V} \setminus \{A, B\}$, $\sum\limits_{w\in\mathcal{V}} x_{wv} = \sum\limits_{w\in\mathcal{V}} x_{vw} \leq maxFlow\,(A, B) \leq in_{B,0}$. Player B will follow a modified evil strategy where she steals value equal to her total incoming flow, not her total incoming direct trust. Let j_2 be the first turn when A is chosen to play. We will show using strong induction that there exists a set of valid actions for each player according to their respective strategy such that at the end of each turn j the corresponding player $v = Player\,(j)$ will have stolen value x_{wv} from each in-neighbour w.

Base case: In turn 1, B steals value equal to $\sum\limits_{w\in\mathcal{V}} x_{wB}$, following the modified evil strategy.

$$Turn_1 = \bigcup_{v\in N^-(B)_0} \{Steal\,(x_{vB}, v)\}$$

Induction hypothesis: Let $k \in [j_2 - 2]$. We suppose that $\forall i \in [k]$, there exists a valid set of actions, $Turn_i$, performed by $v = Player(i)$ such that v steals from each player w value equal to x_{wv}.

$$\forall i \in [k], Turn_i = \bigcup_{w \in N^-(v)_{i-1}} \{Steal(x_{wv}, w)\}$$

Induction step: Let $j = k + 1, v = Player(j)$. Since all the players that are greater than v in the total order have already played and all of them have stolen value equal to their incoming flow, we deduce that v has been stolen value equal to $\sum_{w \in N^+(v)_{j-1}} x_{vw}$. Since it is the first time v plays, $\forall w \in N^-(v)_{j-1}, DTr_{w \to v, j-1} = DTr_{w \to v, 0} \geq x_{wv}$, thus v is able to choose the following turn:

$$Turn_j = \bigcup_{w \in N^-(v)_{j-1}} \{Steal(x_{wv}, w)\}$$

Moreover, this turn satisfies the conservative strategy since

$$\sum_{w \in N^-(v)_{j-1}} x_{wv} = \sum_{w \in N^+(v)_{j-1}} x_{vw} \ .$$

Thus $Turn_j$ is a valid turn for the conservative player v.

We have proven that in the end of turn $j_2 - 1$, player B and all the conservative players will have stolen value exactly equal to their total incoming flow, thus A will have been stolen value equal to her outgoing flow, which is $maxFlow(A, B)$. Since there remains no Angry player, j_2 is a convergence turn, thus $Loss_{A,j_2} = Loss_A$. We can also see that if B had chosen the original evil strategy, the described actions would still be valid only by supplementing them with additional $Steal()$ actions, thus $Loss_A$ would further increase. This proves the lemma. □

Proof of Lemma 2: Transitive Games Are Flows

Let $Sad, Happy, Angry$ be as defined in the Transitive Game. Let $\mathcal{G}'$ be a directed weighted graph based on $\mathcal{G}$ with an auxiliary source. Let also j_1 be a turn when the Transitive Game has converged. More precisely, $\mathcal{G}'$ is defined as follows:

$$\mathcal{V}' = \mathcal{V} \cup \{T\}$$

$$\mathcal{E}' = \mathcal{E} \cup \{(T, A)\} \cup \{(T, v) : v \in Sad_{j_1}\}$$

$$\forall (v, w) \in \mathcal{E}, c'_{vw} = DTr_{v \to w, 0} - DTr_{v \to w, j_1}$$

$$\forall v \in Sad_{j_1}, c'_{Tv} = c'_{TA} = \infty$$

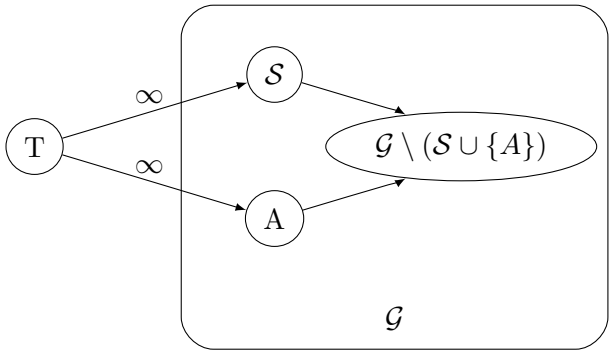

Fig.7: Graph $\mathcal{G}'$, derived from $\mathcal{G}$ with Auxiliary Source T.

In the figure above, $\mathcal{S}$ is the set of sad players. We observe that $\forall v \in \mathcal{V}$,

$$\begin{aligned} &\sum_{w \in N^-(v)' \setminus \{T\}} c'_{wv} = \\ &= \sum_{w \in N^-(v)' \setminus \{T\}} (DTr_{w \to v,0} - DTr_{w \to v,j_1}) = \\ &= \sum_{w \in N^-(v)' \setminus \{T\}} DTr_{w \to v,0} - \sum_{w \in N^-(v)' \setminus \{T\}} DTr_{w \to v,j-1} = \\ &= in_{v,0} - in_{v,j_1} \end{aligned} \tag{18}$$

and

$$\begin{aligned} &\sum_{w \in N^+(v)' \setminus \{T\}} c'_{vw} = \\ &= \sum_{w \in N^+(v)' \setminus \{T\}} (DTr_{v \to w,0} - DTr_{v \to w,j_1}) = \\ &= \sum_{w \in N^+(v)' \setminus \{T\}} DTr_{v \to w,0} - \sum_{w \in N^+(v)' \setminus \{T\}} DTr_{v \to w,j-1} = \\ &= out_{v,0} - out_{v,j_1} \ . \end{aligned} \tag{19}$$

We can suppose that

$$\forall j \in \mathbb{N}, in_{A,j} = 0 \ , \tag{20}$$

since if we find a valid flow under this assumption, the flow will still be valid for the original graph.

Next we try to calculate $MaxFlow\,(T, B) = X'$ on graph $\mathcal{G}'$. We observe that a flow in which it holds that $\forall v, w \in \mathcal{V}, x'_{vw} = c'_{vw}$ can be valid for the following reasons:

- $\forall v, w \in \mathcal{V}, x'_{vw} \leq c'_{vw}$ (Capacity flow requirement (12) $\forall e \in \mathcal{E}$)
- Since $\forall v \in Sad_{j_1} \cup \{A\}, c'_{Tv} = \infty$, requirement (12) holds for any flow $x'_{Tv} \geq 0$.
- Let $v \in \mathcal{V}' \setminus (Sad_{j_1} \cup \{T, A, B\})$. According to the conservative strategy and since $v \notin Sad_{j_1}$, it holds that

  $$out_{v,0} - out_{v,j_1} = in_{v,0} - in_{v,j_1} \ .$$

  Combining this observation with (18) and (19), we have that

  $$\sum_{w \in \mathcal{V}'} c'_{vw} = \sum_{w \in \mathcal{V}'} c'_{wv} \ .$$

  (Flow Conservation requirement (13) $\forall v \in \mathcal{V}' \setminus (Sad_{j_1} \cup \{T, A, B\})$)
- Let $v \in Sad_{j_1}$. Since v is sad, we know that

  $$out_{v,0} - out_{v,j_1} > in_{v,0} - in_{v,j_1} \ .$$

  Since $c'_{Tv} = \infty$, we can set

  $$x'_{Tv} = (out_{v,0} - out_{v,j_1}) - (in_{v,0} - in_{v,j_1}) \ .$$

  In this way, we have

  $$\sum_{w \in \mathcal{V}'} x'_{vw} = out_{v,0} - out_{v,j_1} \text{ and}$$

  $$\sum_{w \in \mathcal{V}'} x'_{wv} = \sum_{w \in \mathcal{V}' \setminus \{T\}} c'_{wv} + x'_{Tv} = in_{v,0} - in_{v,j_1} +$$

  $$+(out_{v,0} - out_{v,j_1}) - (in_{v,0} - in_{v,j_1}) = out_{v,0} - out_{v,j_1} \ .$$

  thus

  $$\sum_{w \in \mathcal{V}'} x'_{vw} = \sum_{w \in \mathcal{V}'} x'_{wv} \ .$$

  (Requirement 13 $\forall v \in Sad_{j_1}$)
- Since $c'_{TA} = \infty$, we can set

  $$x'_{TA} = \sum_{v \in \mathcal{V}'} x'_{Av} \ ,$$

  thus from (20) we have

  $$\sum_{v \in \mathcal{V}'} x'_{vA} = \sum_{v \in \mathcal{V}'} x'_{Av} \ .$$

  (Requirement 13 for A)

We saw that for all nodes, the necessary properties for a flow to be valid hold and thus X' is a valid flow for $\mathcal{G}$. Moreover, this flow is equal to $maxFlow\,(T, B)$ because all incoming flows to E are saturated. Also we observe that

$$\sum_{v\in\mathcal{V}'} x'_{Av} = \sum_{v\in\mathcal{V}'} c'_{Av} = out_{A,0} - out_{A,j_1} = Loss_A \ . \tag{21}$$

We define another graph, $\mathcal{G}''$, based on $\mathcal{G}'$.

$$\mathcal{V}'' = \mathcal{V}'$$

$$E(\mathcal{G}'') = E(\mathcal{G}') \setminus \{(T, v) : v \in Sad_j\}$$
$$\forall e \in E(\mathcal{G}''), c''_e = c'_e$$

If we execute $MaxFlow(T, B)$ on the graph $\mathcal{G}''$, we will obtain a flow X'' in which

$$\sum_{v\in\mathcal{V}''} x''_{Tv} = x''_{TA} = \sum_{v\in\mathcal{V}''} x''_{Av} \ .$$

The outgoing flow from A in X'' will remain the same as in X' for two reasons: Firstly, using the Flow Decomposition theorem **?** and deleting the paths that contain edges $(T, v) : v \neq A$, we obtain a flow configuration where the total outgoing flow from A remains invariant, [3] thus

$$\sum_{v\in\mathcal{V}''} x''_{Av} \geq \sum_{v\in\mathcal{V}'} x'_{Av} \ .$$

Secondly, we have

$$\left.\begin{array}{l} \sum\limits_{v\in\mathcal{V}''} c''_{Av} = \sum\limits_{v\in\mathcal{V}'} c'_{Av} = \sum\limits_{v\in\mathcal{V}'} x'_{Av} \\ \sum\limits_{v\in\mathcal{V}''} c''_{Av} \geq \sum\limits_{v\in\mathcal{V}''} x''_{Av} \end{array}\right\} \Rightarrow \sum_{v\in\mathcal{V}''} x''_{Av} \leq \sum_{v\in\mathcal{V}'} x'_{Av} \ .$$

Thus we conclude that

$$\sum_{v\in\mathcal{V}''} x''_{Av} = \sum_{v\in\mathcal{V}'} x'_{Av} \ . \tag{22}$$

Let $X = X'' \setminus \{(T, A)\}$. Observe that

$$\sum_{v\in\mathcal{V}''} x''_{Av} = \sum_{v\in\mathcal{V}} x_{Av} \ .$$

[3] We thank Kyriakos Axiotis for his insights on the Flow Decomposition theorem.

This flow is valid on graph $\mathcal{G}$ because

$$\forall e \in \mathcal{E}, c_e \geq c''_e \ .$$

Thus there exists a valid flow for each execution of the Transitive Game such that

$$\sum_{v \in \mathcal{V}} x_{Av} = \sum_{v \in \mathcal{V}''} x''_{Av} \overset{(22)}{=} \sum_{v \in \mathcal{V}'} x'_{Av} \overset{(21)}{=} Loss_{A,j_1} \ ,$$

which is the flow X. □

Theorem 6 (Conservative World Theorem).
If everybody follows the conservative strategy, nobody steals any amount from anybody.

Proof. Let $\mathcal{H}$ be the game history where all players are conservative and suppose there are some *Steal* () actions taking place. Then let $\mathcal{H}'$ be the subsequence of turns each containing at least one *Steal* () action. This subsequence is evidently nonempty, thus it must have a first element. The player corresponding to that turn, A, has chosen a *Steal* () action and no previous player has chosen such an action. However, player A follows the conservative strategy, which is a contradiction. □

Proof of Theorem 5: Sybil Resilience
Let $\mathcal{G}_1$ be a game graph defined as follows:

$$\mathcal{V}_1 = \mathcal{V} \cup \{T_1\} \ ,$$

$$\mathcal{E}_1 = \mathcal{E} \cup \{(v, T_1) : v \in \mathcal{B} \cup \mathcal{C}\} \ ,$$

$$\forall v, w \in \mathcal{V}_1 \setminus \{T_1\}, DTr^1_{v \to w} = DTr_{v \to w} \ ,$$

$$\forall v \in \mathcal{B} \cup \mathcal{C}, DTr^1_{v \to T_1} = \infty \ ,$$

where $DTr_{v \to w}$ is the direct trust from v to w in $\mathcal{G}$ and $DTr^1_{v \to w}$ is the direct trust from v to w in $\mathcal{G}_1$.
Let also $\mathcal{G}_2$ be the induced graph that results from $\mathcal{G}_1$ if we remove the Sybil set, $\mathcal{C}$. We rename T_1 to T_2 and define $\mathcal{L} = \mathcal{V} \setminus (\mathcal{B} \cup \mathcal{C})$ as the set of legitimate players to facilitate comprehension.

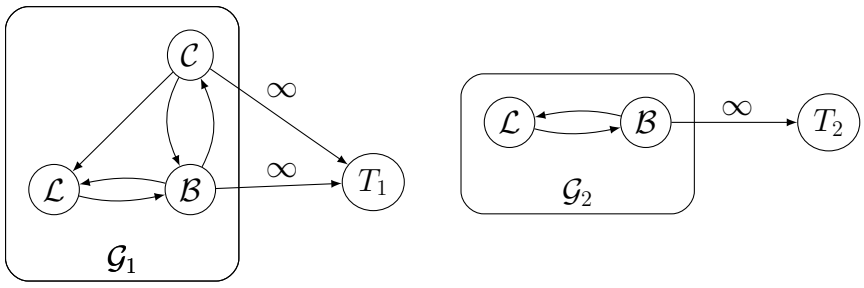

Fig.8: Graphs $\mathcal{G}_1$ and $\mathcal{G}_2$

According to theorem (4),

$$Tr_{A\to\mathcal{B}\cup\mathcal{C}} = maxFlow_1\left(A, T_1\right) \wedge Tr_{A\to\mathcal{B}} = maxFlow_2\left(A, T_2\right) \quad . \tag{23}$$

We will show that the $MaxFlow$ of each of the two graphs can be used to construct a valid flow of equal value for the other graph. The flow $X_1 = MaxFlow\left(A, T_1\right)$ can be used to construct a valid flow of equal value for the second graph if we set

$$\begin{aligned} \forall v \in \mathcal{V}_2 \setminus \mathcal{B}, \forall w \in \mathcal{V}_2, x_{vw,2} &= x_{vw,1} \ , \\ \forall v \in \mathcal{B}, x_{vT_2,2} &= \sum_{w \in N_1^+(v)} x_{vw,1} \ , \\ \forall v, w \in \mathcal{B}, x_{vw,2} &= 0 \ . \end{aligned}$$

Therefore

$$maxFlow_1\left(A, T_1\right) \leq maxFlow_2\left(A, T_2\right)$$

Likewise, the flow $X_2 = MaxFlow(A, T_2)$ is a valid flow for $\mathcal{G}_1$ because $\mathcal{G}_2$ is an induced subgraph of $\mathcal{G}_1$. Therefore

$$maxFlow_1\left(A, T_1\right) \geq maxFlow_2\left(A, T_2\right)$$

We conclude that

$$maxFlow\left(A, T_1\right) = maxFlow\left(A, T_2\right) \ , \tag{24}$$

thus from (23) and (24) the theorem holds. □

2 Algorithms

This algorithm calls the necessary functions to prepare the new graph.

```
Execute Turn
Input : old graph 𝒢_{j−1}, player A ∈ 𝒱_{j−1}, old capital
    Cap_{A,j−1}, TentativeTurn
Output : new graph 𝒢_j, new capital Cap_{A,j}, new history ℋ_j
executeTurn(𝒢_{j−1}, A, Cap_{A,j−1}, TentativeTurn) :
  (Turn_j, NewCap) = validateTurn(𝒢_{j−1}, A, Cap_{A,j−1},
      TentativeTurn)
  return(commitTurn(𝒢_{j−1}, A, Turn_j, NewCap))
```

The following algorithm validates that the tentative turn produced by the strategy respects the rules imposed on turns. If the turn is invalid, an empty turn is returned.

```
Validate Turn
Input : old 𝒢_{j−1}, player A ∈ 𝒱_{j−1}, old Cap_{A,j−1}, Turn
Output : Turn_j, new Cap_{A,j}
validateTurn(𝒢_{j−1}, A, Cap_{A,j−1}, Turn) :
  Y_st = Y_add = 0
  Stolen = Added = ∅
  for (action ∈ Turn)
    action match do
      case Steal(y, w) do
        if (y > DTr_{w→A,j−1} or y < 0 or w ∈ Stolen)
          return(∅, Cap_{A,j−1})
        else Y_st += y; Stolen = Stolen ∪ {w}
      case Add(y, w) do
        if (y < -DTr_{A→w,j−1} or w ∈ Added)
          return(∅, Cap_{A,j−1})
        else Y_add += y; Added = Added ∪ {w}
  if (Y_add - Y_st > Cap_{A,j−1}) return(∅, Cap_{A,j−1})
  else return(Turn, Cap_{A,j−1} + Y_st − Y_add)
```

Finally, this algorithm applies the turn to the old graph and returns the new graph, along with the updated capital and history.

```
Commit Turn
Input : old 𝒢_{j−1}, player A ∈ 𝒱_{j−1}, NewCap, Turn_j
Output : new 𝒢_j, new Cap_{A,j}, new ℋ_j
commitTurn(𝒢_{j−1}, A, NewCap, Turn_j) :
  for ((v, w) ∈ ℰ_j) DTr_{v→w,j} = DTr_{v→w,j−1}
```

```
3   for (action ∈ Turn_j)
4     action match do
5       case Steal(y, w) do DTr_{w→A,j} = DTr_{w→A,j−1} − y
6       case Add(y, w) do DTr_{A→w,j} = DTr_{A→w,j−1} + y
7   Cap_{A,j} = NewCap; H_j = (A, Turn_j)
8   return(G_j, Cap_{A,j}, H_j)
```

It is straightforward to verify the compatibility of the previous algorithms with the corresponding definitions.