# 1 Abstract

Reputation in centralized systems typically uses stars and review-based trust. These systems require extensive manual intervention and secrecy to avoid manipulation. In decentralized systems this luxury is not available as the reputation system should be autonomous and open source. Previous peer-to-peer reputation systems define trust abstractly and do not allow for financial arguments pertaining to reputation. We propose a concrete sybil-resilient decentralized reputation system in which direct trust is defined as lines-of-credit using bitcoin's 1-of-2 multisig. We introduce a new model for bitcoin wallets in which user coins are split among trusted friends. Indirect trust is subsequently defined using a transitive property. This enables formal game theoretic arguments pertaining to risk analysis. Using our reputation model, we prove that risk and max flows are equivalent and propose several algorithms for the redistribution of trust so that a decision can be made on whether an anonymous third party can be indirectly trusted. In such a setting, the risk incurred by making a purchase from an anonymous vendor remains invariant. Finally, we prove the correctness of our algorithms and provide optimality arguments for various norms.

# 2 Introduction

# 3 Tags/Keywords

decentralized, trust, web-of-trust, bitcoin, multisig, line-of-credit, trust-as-risk, flow

# 4 Related Work

# 5 Key points

# 6 Definitions

**Definition 6.1** (Players)**.**
The set $\mathcal{M} = V(G)$ is the set of all players in the network, otherwise understood as the set of all pseudonymous identities.

**Definition 6.2** ((In/Out) Neighbourhood of $A$, $N^+(A), N^-(A), N(A)$)**.**

1. Let $N^+(A)$ be the set of players $B$ that $A$ directly trusts with any positive value. More formally, $N^+(A) = \{B \in \mathcal{M} : DTr_{A\to B} > 0\}$. $N^+(A)$ is called out neighbourhood of $A$.
2. Let $N^-(A)$ be the set of players $B$ that directly trust $A$ with any positive value. More formally, $N^-(A) = \{B \in \mathcal{M} : DTr_{B\to A} > 0\}$. $N^-(A)$ is called in neighbourhood of $A$.
3. Let $N(A)$ be the set of players $B$ that either directly trust or are directly trusted by $A$ with any positive value. More formally, $N(A) = N^+(A) \cup N^-(A)$. $N(A)$ is called neighbourhood of $A$.
4. Let $N(A)_i$ (respectively $N^+(A)_i, N^-(A)_i$) be the $i$-th element of set $N(A)$ (respectively of $N^+(A), N^-(A)$), according to an arbitrary but constant enumeration of the set players.

**Definition 6.3** (Direct Trust from $A$ to $B$, $DTr_{A\to B}$)**.**
Total amount of value that exists in 1-of-$A$,$B$ multisigs in the utxo, where the money is deposited by $A$.

**Definition 6.4** (Capital of $A$, $Cap_A$)**.**
Total amount of value that exists in the utxo and can be spent by $A$.

**Definition 6.5** ($B$ steals $x$ from $A$)**.**
$B$ steals value $x$ from $A$ when $B$ reduces the $DTr_{A\to B}$ by $x$ and increases $Cap_B$ by $x$. This makes sense when $x \leq DTr_{A\to B}$.

**Definition 6.6** (Indirect trust from $A$ to $B$ $Tr_{A\to B}$)**.**
Value that $A$ will lose if $B$ steals the maximum amount she can steal (all her incoming trust) and everyone else follows the honest strategy.

**Definition 6.7** (Trust Reduction)**.**
Let $A, B \in \mathcal{M}, x_i$ flow to $N^+(A)_i$ resulting from $maxFlow(A, B), u_i$ current $DTr_{A\to N^+(A)_i}$, $u'_i$ new $DTr_{A\to N^+(A)_i}, i \in \{1, ..., |N^+(A)|\}$.

1. The Trust Reduction on neighbour $i, \delta_i$ is defined as $\delta_i = u_i - u'_i$.
2. The Flow Reduction on neigbour $i, \Delta_i$ is defined as $\Delta_i = x_i - u'_i$.

We will also use the standard notation for 1-norm and $\infty$-norm, that is:

1. $||\delta_i||_1 = \sum_{i\in N^+(A)} \delta_i$
2. $||\delta_i||_\infty = \max_{i\in N^+(A)} \delta_i$.

**Definition 6.8** (Restricted Flow)**.**
Let $A, B \in \mathcal{M}, i \in \{1, ..., |N^+(A)|\}$.

1. Let $F_{A_i\to B}$ be the flow from $A$ to $N^+(A)_i$ as calculated by the $maxFlow(A, B)$ $(x'_i)$ when $u'_i = u_i$, $u'_j = 0\ \forall j \in \{1, ..., |N^+(A)|\} \wedge j \neq i$.
2. Let $S \subset N^+(A)$. Let $F_{A_S\to B}$ be the sum of flows from $A$ to $S$ as calculated by the $maxFlow(A, B)$ $(\sum_{i=1}^{|S|} x'_i)$ when $u'_C = u_C\ \forall C \in S, u'_D = 0\ \forall D \in N^+(A) \setminus S$.

**Definition 6.9** (Turns)**.**
The game we are describing is turn-based. In each turn $i$ exactly one player $A \in \mathcal{M}$ chooses an action (according to a certain strategy or at random) that can be one of the following, or a finite combination thereof:

1. Do nothing (Pass).
2. Steal value $y_B, 0 \leq y_B \leq DTr_{B\to A,i-1}$ from $B \in N^-(A)$. $DTr_{B\to A,i} = DTr_{B\to A,i-1} - y_B$.
3. Add value $y_B, -DTr_{A\to B,i-1} \leq y_B$ to $B \in N^+(A)$. $DTr_{A\to B,i} = DTr_{A\to B,i-1} + y_B$. When $y_B < 0$, we say that $A$ reduces her trust to $B$ by $-y_B$, when $y_B > 0$, we say that $A$ increases her trust to $B$ by $y_B$.
4. Add value $y_B$ to $B \in \mathcal{M} \setminus N^+(A)$. Obviously $DTr_{A\to B,i-1} = 0, DTr_{A\to B,i} = y_B$. We say that $A$ starts directly trusting player $B$ by $y_B$.

Player $A$ is not allowed to choose two actions of the same kind against the same player in the same turn. For example, $A$ cannot steal $y_{B,1} \leq DTr_{B\to A,i-1}$ and $y_{B,2} \leq DTr_{B\to A,i-1}$ from $B \in N^-(A)$ in the same turn, but can instead steal $y_{B,1} + y_{B,2}$ given that $y_{B,1} + y_{B,2} \leq DTr_{B\to A,i-1}$. Also $A$ is allowed to steal $y_B \leq DTr_{B\to A,i-1}$ from $B$ and start directly trusting the same player $B$ by $w_B$ given that $B \in N^-(A) \wedge B \notin N^+(A)$.

**Definition 6.10** (Honest strategy)**.**
A player $A$ is said to follow the honest strategy if for any value $x$ that has been stolen from her since the previous turn she played, she substitutes it in her turn by stealing from others that trust her value equal to $min(x, \sum_{B\in\mathcal{M}} DTr_{B\to A})$ and she takes no other action.

**Definition 6.11** (Idle strategy)**.**
A player $A$ is said to follow the idle strategy if she passes in her turn.

**Definition 6.12** (Evil strategy)**.**
A player $A$ is said to follow the evil strategy if she steals value $y_B = DTr_{B\to A,i-1}\ \forall\ B \in N^-(A)$ (steals all incoming direct trust) and reduces her trust to $C$ by $DTr_{A\to C,i-1} \forall C \in N^+(A)$ (nullifies her outgoing direct trust) in her turn.

# 7 Theorems-Algorithms

**Theorem 7.1** (Saturation theorem)**.**
*Let $s$ source, $n = |N^+(s)|, x_i, i \in \{1, ..., n\}$, flows to $s$'s neighbours as calculated by the maxFlow algorithm, $u'_i$ new direct trusts to the $n$ neighbours and $x'_i$ new flows to the neighbours as calculated by the maxFlow algorithm with the new direct trusts, $u'_i$. It holds that $\forall i \in \{1, ..., n\}, u'_i \leq x_i \Rightarrow x'_i = u'_i$.*

*Proof.*

1. $\forall i \in \{1, ..., n\}, x'_i > u'_i$ is impossible because a flow cannot be higher than its corresponding capacity. Thus $\forall i \in \{1, ..., n\}, x'_i \leq u'_i$.
2. In the initial configuration of $u_i$ and according to the flow problem setting, a combination of flows $y_i$ such that $\forall i \in \{1, ..., n\}, y_i = u'_i$ is a valid, albeit not necessarily maximum, configuration with a flow $\sum_{i=1}^{n} y_i$. Suppose that $\exists j \in \{1, ..., n\} : x'_j < u'_j$ as calculated by the maxFlow algorithm with the new direct trusts, $u'_i$. Then for the new maxFlow $F'$ it holds that $F' = \sum_{i=1}^{n} x'_i < \sum_{i=1}^{n} y_i$ since $x'_j < y_j$ which is impossible because the configuration $\forall i \in \{1, ..., n\}, x'_i = y_i$ is valid since $\forall i \in \{1, ..., n\}, y_i = u'_i$ and also has a higher flow, thus the maxFlow algorithm will prefer the configuration with the higher flow. Thus we deduce that $\forall i \in \{1, ..., n\}, x'_i \geq u'_i$.

From (1) and (2) we conclude that $\forall i \in \{1, ..., n\}, x'_i = u'_i$. □

**Theorem 7.2** (Trust flow theorem - TOCHECK)**.**
*$Tr_{A \to B} = MaxFlow_{A \to B}$ (Treating trusts as capacities)*

*Proof.*

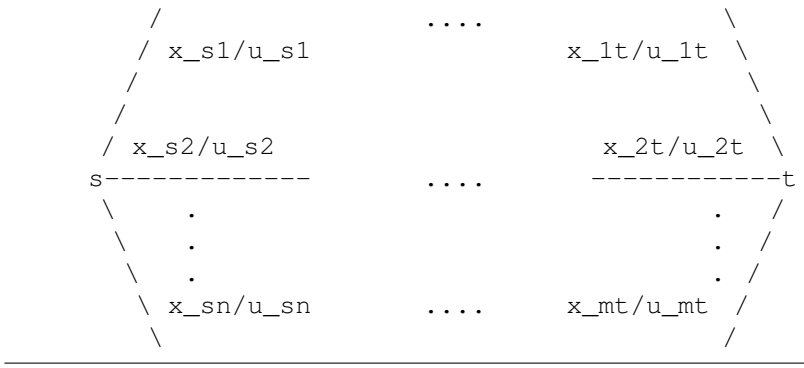
1. We will show that $Tr_{A \to B} \leq MaxFlow_{A \to B}$. We know that $MaxFlow_{A \to B} = MinCut_{A \to B}$. We will show that, if everybody except A and B follows the honest strategy, $Tr_{A \to B} \leq MinCut_{A \to B}$. Suppose that in round $i$ all the members of the MinCut, $P$, have stolen the maximum value they can from members that belong in the MaxFlow graph and nobody in the partition in which $A$ belongs has stolen yet any value. Let the total stolen value from the MinCut members be $St$. It is obvious that $St_i \leq MinCut_{A \to B}$, because otherwise there would exist $u \in P$ that doesn't follow the honest strategy, since they stole more than they were stolen from. The same argument holds for any round $i' > i$ because in each round an honest player can steal only up to the value she has been stolen. It is also impossible that the $St$ increase further due to stolen value from members of the partition of $B$ since members of $P$ disconnect the two partitions and have already played their turns, thus $\forall i' > i, St_{i'} \leq St_i$. There exists a round, $k$, when all the honest players stop stealing, so in the worst case $A$ will have been stolen $Tr_{A \to B} = St_k \leq MinCut_{A \to B} = MaxFlow_{A \to B}$.
2. We can see that $Tr_{A \to B} \geq MaxFlow_{A \to B}$ because the strategy where each one of the non-idle players steals value equal to the incoming flows from their respective friends is a valid strategy that does not contradict with the honest strategy, since for every honest player $w$ it holds that $\sum_{v \in N^-(w)} x_{vw} = \sum_{v \in N^+(w)} x_{wv}$ and according to the strategy each honest player will have been stolen value equal to $\sum_{v \in N^+(w)} x_{wv}$.

Combining the two results, we see that $Tr_{A \to B} = MaxFlow_{A \to B}$. □

**Theorem 7.3** (Honest world theorem)**.**
*If everybody follows the honest strategy, nobody steals any amount from anybody.*

*Proof.* Suppose that there exists a series of stealing actions represented by a vector where $action_i$ ="player $i$ steals value V $> 0$ from player $i + 1$". This vector must have an initial element, $action_1$. However, player 1 follows the honest strategy, thus somebody must have stolen from her as well, so player 1 cannot be the initial element. We have a contradiction, thus there cannot exist a series of stealing actions when everybody is honest. □

**Theorem 7.4** (Trust transfer theorem (flow terminology) - TOCHECK)**.**
*Let $s$ source, $t$ sink, $n = N^+(s)$*
*$X_s = \{x_{s,1}, ..., x_{s,n}\}$ outgoing flows from $s$,*
*$U_s = \{u_{s,1}, ..., u_{s,n}\}$ outgoing capacities from $s$,*
*$V$ the value to be transferred.*
*Nodes apart from $s$, $t$ follow the honest strategy.*
*Obviously $maxFlow = F = \sum_{i=1}^{n} x_{s,i}$.*




*We create a new graph where*

1. $\sum_{i=1}^{n} u'_{s,i} = F - V$
2. $\forall i \in \{1, ..., n\}\ u'_{s,i} \leq x_{s,i}$

*It holds that $maxFlow' = F' = F - V$.*

*Proof.* From theorem 7.1 we can see that $x'_i = u'_i$. It holds that $F' = \sum_{i=1}^{n} x'_i = \sum_{i=1}^{n} u'_i = F - V$. □

**Corollary 7.1** (Requirement for $\sum_{i=1}^{n} u'_{s,i} = F - V$, $u'_{s,i} \leq x_{s,i}$)**.**
*In the setting of 7.4, it is impossible to have $maxFlow' = F - V$ if $\sum_{i=1}^{n} u'_{s,i} > F - V \wedge \forall i \in \{1, ..., n\}, u'_{s,i} \leq x_{s,i}$.*

*Proof.* Due to 7.4, $maxFlow' = F - V$ if $\sum_{i=1}^{n} u'_{s,i} = F - V \wedge \forall i \in \{1, ..., n\}, u'_{s,i} \leq x_{s,i}$. If we create new capacities such that $\forall i \in \{1, ..., n\}, u''_{s,i} \leq x_{s,i}$, then obviously $maxFlow'' = \sum_{i=1}^{n} u''_{s,i}$. If additionally $\sum_{i=1}^{n} u''_{s,i} > F - V$, then $maxFlow'' > F - V$. □

**Theorem 7.5** (Trust-saving Theorem)**.**
$\forall i \in \{1, ..., |N^+(A)|\}, u'_i = F_{A_i \to B} \Leftrightarrow u'_i = u_i$

*Proof.* We know that $x_i \leq F_{A_i \to B}$, thus we can see that any increase in $u'_i$ beyond $F_{A_i \to B}$ will not influence $x_i$ and subsequently will not incur any change on the rest of the flows. □

**Theorem 7.6** (Invariable trust reduction with naive algorithms)**.**
*Let $A$ source, $n = |N^+(A)|$ and $u'_i$ new direct trusts. If $\forall i \in \{1, ..., n\}, u'_i \leq x_i$, Trust Reduction $||\delta_i||_1$ is independent of $x_i, u'_i$*
*$\forall$ configurations of $x_i$*

*Proof.* Since $\forall i \in \{1, ..., n\}, u'_i \leq x_i$ it is (according to 7.1) $x'_i = u'_i$, thus $\delta_i = u_i - x'_i$. We know that $\sum_{i=1}^{n} x'_i = F - V$, so we have $||\delta_i||_1 = \sum_{i=1}^{n} \delta_i = \sum_{i=1}^{n} (u_i - x'_i) = \sum_{i=1}^{n} u_i - F + V$ independent of $x'_i, u'_i$ □

**Theorem 7.7** (Dependence impossibility theorem)**.**
$\frac{\partial x_j}{\partial x_i} = 0$ *with* $x_i$ *the flow from MaxFlow* $\Rightarrow \forall x'_i \leq x_i, \frac{\partial x_j}{\partial x_i} = 0$ *ceteris paribus*

*Proof.* TODO □

Here we show three naive algorithms for calculating new direct trusts so as to maintain invariable risk when paying a trusted party.

**Algorithm 1:** First-come, first-served trust transfer

**Input :** $x_i$ flows, $n = |N^+(s)|$, $V$ value
**Output:** $u'_i$ capacities

1 $F \leftarrow \sum_{i=1}^{n} x_i$
2 **if** $F < V$ **then**
3 | **return** $\perp$
4 $Fcur \leftarrow F$
5 **for** $i \leftarrow 1$ *to* $n$ **do**
6 | $u'_i \leftarrow x_i$
7 $i \leftarrow 1$
8 **while** $Fcur > F - V$ **do**
9 | $reduce \leftarrow min(x_i, Fcur - F + V)$
10 | $Fcur \leftarrow Fcur - reduce$
11 | $u'_i \leftarrow x_i - reduce$
12 | $i \leftarrow i + 1$
13 **return** $U' = \bigcup_{i=1}^{n} \{u'_i\}$

*Proof of correctness for algorithm 1.*
We want to prove that $\forall i \in \{1, ..., n\} u'_i \leq x_i$ and that $\sum_{i=1}^{n} u'_i = F - V$ where $F = \sum_{i=1}^{n} x_i$.

- Let $i \in \{1, ..., n\}$. In line 6 we can see that $u'_i = x_i$ and the only other occurence of $u'_i$ is in line 11 where it is never increased ($reduce \geq 0$), thus we see that, when returned, $u'_i \leq x_i$.

- We will show that $\sum_{i=1}^{n} u'_i = F - V$.
  $F_{cur,0} = F$
  If $F_{cur,i} = F - V$, then $F_{cur,i+1}$ does not exist because the *while* loop breaks after calculating $F_{cur,i}$.
  Else $F_{cur,i+1} = F_{cur,i} - min(x_{i+1}, F_{cur,i} - F + V)$.
  If for some $i, min(x_{i+1}, F_{cur,i} - F + V) = F_{cur,i} - F + V$, then $F_{cur,i+1} = F - V$, so if $F_{cur,i+1}$ exists, then $\forall k < i, F_{cur,k} = F_{cur,k-1} - x_k \Rightarrow F_{cur,i} = F - \sum_{j=1}^{i} x_j$
  Furthermore, if $F_{cur,i+1} = F - V$ then $u'_{i+1} = x_{i+1} - F_{cur,i} + F - V = x_i - F + \sum_{j=1}^{i-1} x_j + F - V = \sum_{j=1}^{i} x_j - V$, $\forall k \leq i, u'_k = 0$ and $\forall k > i + 1, u'_k = x_k$.
  In total, we have $\sum_{j=1}^{n} u'_j = \sum_{j=1}^{i} x_j - V + \sum_{j=i+1}^{n} x_j = \sum_{j=1}^{n} x_j - V \Rightarrow \sum_{j=1}^{n} u'_j = F - V$.

□

*Complexity of algorithm 1.*

□

**Algorithm 2:** Absolute equality trust transfer($||\Delta_i||_\infty$ minimizer)

**Input :** $x_i$ flows, $n = |N^+(s)|$, $V$ value
**Output:** $u'_i$ capacities

$F \leftarrow \sum_{i=1}^{n} x_i$
**if** $F < V$ **then**
  **return** $\perp$
**for** $i \leftarrow 1$ *to* $n$ **do**
  $u'_i \leftarrow x_i$
$reduce \leftarrow \frac{V}{n}$
$reduction \leftarrow 0$
$empty \leftarrow 0$
$i \leftarrow 0$
**while** $reduction < V$ **do**
  **if** $u'_i > 0 \ \wedge \ x_i < reduce$ **then**
    $empty \leftarrow empty + 1$
    $reduce = reduce + \frac{x_i - reduce - u'_i}{n - empty}$
    $reduction \leftarrow reduction + u'_i$
    $u'_i \leftarrow 0$
  **else if** $x_i \geq reduce$ **then**
    $reduction \leftarrow reduction + u'_i - (x_i - reduce)$
    $u'_i \leftarrow x_i - reduce$
  $i \leftarrow (i + 1) mod\, n$
**return** $U' = \bigcup_{i=1}^{n} \{u'_i\}$

**Algorithm 3:** Proportional equality trust transfer

**Input :** $x_i$ flows, $n = |N^+(s)|$, $V$ value
**Output:** $u'_i$ capacities

$F \leftarrow \sum_{i=1}^{n} x_i$
**if** $F < V$ **then**
  **return** $\perp$
**for** $i \leftarrow 1$ *to* $n$ **do**
  $u'_i \leftarrow x_i - \frac{V}{F} x_i$
**return** $U' = \bigcup_{i=1}^{n} \{u'_i\}$

**Algorithm 4:** $||\delta_i||_\infty$ minimizer

**Input :** $X = \{x_i\}$ flows, $n = |N^+(s)|$, $V$ value
**Output:** $u'_i$ capacities

$F \leftarrow \sum_{i=1}^{n} x_i$
$\delta \leftarrow$ `BinSearch(`*0,V,F,n,X*`)`
**for** $i \leftarrow 1$ *to* $n$ **do**
  $u'_i \leftarrow max(u_i - \delta, 0)$
**return** $U' = \bigcup_{i=1}^{n} \{u'_i\}$

**Algorithm 5:** BinSearch function

**Input :** *bot*, *top*, $F$, $n$, $X$
**Output:** $\delta$

**if** *bot* = *top* **then**
  **return** *bot*
**else**
  **for** $i \leftarrow 1$ *to* $n$ **do**
    $u'_i \leftarrow u_i - \frac{top+bot}{2}$
  **if** $maxFlow < F$ **then**
    **return** `BinSearch(`*bot,* $\frac{top+bot}{2}$*,n,X,F*`)`
  **else**
    **return** `BinSearch(`$\frac{top+bot}{2}$*, top,n,X,F*`)`

*Proof of correctness - TOEXPAND.* In all three algorithms, we have $u'_i <= x_i$ because in the only case where $u'_i$ is altered after its initialisation, it is reduced. Furthermore, a total of $V$ is subtracted from all the $u'_i$, thus $\sum_{i=1}^{n} u'_i = F - V$. □

However, we need to minimize $\sum_{i=1}^{n}(u_i - u'_i) = ||\delta_i||_1$.

# 8 Further Research

# 9 References