LocalCoin: An Ad-hoc Payment Scheme for Areas with High Connectivity

Dimitris Chatzopoulos, Sujit Gujar, Boi Faltings and Pan Hui

Abstract—The popularity of digital currencies, especially cryptocurrencies, has been continuously growing since the appearance of Bitcoin. Bitcoin's security lies in a proof-of-work scheme, which requires high computational resources at the miners. Despite advances in mobile technology, no cryptocurrencies have been proposed for mobile devices due to the lower processing capabilities of mobile devices. In this work, we propose LocalCoin, an alternative cryptocurrency that requires minimal computational resources, produces low data traffic and works with off-the-shelf mobile devices. LocalCoin replaces the computational hardness that is at the root of Bitcoin's security with the social hardness of ensuring that all witnesses to a transaction are colluders. Localcoin features (i) a lightweight proof-of-work scheme and (ii) a distributed block chain. We analyze LocalCoin for double spending for passive and active attacks and prove that under the assumption of sufficient number of users and properly selected tuning parameters the probability of double spending is close to zero. Extensive simulations on real mobility traces, realistic urban settings, and random geometric graphs show that the probability of success of one transaction converges to 1 and the probability of the success of a double spending attempt converges to 0.

Index Terms—P2P, Ad-hoc networks, cryprocurrency

1 Introduction

Bitcoin, proposed by Nakamoto in 2009, is the most popular online cryptocurrency [1] [2] [3] [31]. Numerous cryptocurrencies have been proposed thereafter and have attracted the attention of both financial and technological industries as well as academia [9] [11] [27]. All the digital currencies that were proposed before Bitcoin, follow the client/server model with transactions possible only between the currency provider and the users (PAYPAL, VISA, MASTERCARD, etc). Bitcoin, in contrast, works in a decentralised manner.

Decentralised cryptocurrencies have to deal with three main challenges: **(i) Proof of ownership**- users should be able to prove they have the amount of money they claim to have. **(ii) Double spending avoidance** - a defense mechanism against double spending. (Users are not able to spend the same money more than once). **(iii) Incentives** - for its stakeholders. Common characteristics of all the existing cryptocurrencies are: **(i)** Internet based **(ii)** use computationally expensive techniques to deal with double spending attacks and **(iii)** require lots of data storage. To become part of the Bitcoin peer network anyone can contribute their resources and work as a *miner*. Bitcoin, as well as other less popular cryptocurrencies, require their miners to employ devices with high computational capabilities and to be interconnected via the Internet. These requirements play a vital role in the quality and the guarantees of the protocols as well as in the miners' revenue. All the transactions are stored in a public ledger named **blockchain** in sets of blocks that are created by the miners [4]. Bitcoin requires from miners to solve cryptographic puzzles, which can only be solved by brute force SHA-256 hashing, in order to generate new blocks for the blockchain [22] [34]. Each block has size of 1 MB and two consecutive blocks are created with 10 minutes time difference, on average (1-3.5 typical-size transactions are verified per second). Miners earn bitcoins whenever they successfully mine a new block and put it in the blockchain. The probability of a miner to mine a block is proportional to the portion of the computational resources of the Bitcoin network he controls.

Cryptocurrencies are inferior to conventional currencies, because users cannot exchange money without the use of an Internet connection. Mobile devices are practically unable to partake as peers in any decentralised cryptocurrency, because of **(i)** their lower processing capabilities compared to conventional CPUs and GPUs and the specialised for mining hardware [39] and **(ii)** their unstable connectivity to the Internet compared to ordinary wire-line access protocols. For example, consider a university campus which might be spread across an area of some km^2 with several thousands of users with smartphones. These smartphones can be used to deploy Bitcoin-like currency. However, these devices can not compete the Bitcoin miners in the block creation process and this fact gives no incentives to their owners to employ them. Despite that, with widespread usage of smartphones, which are equipped with technologies such as WiFi-direct, NFC and so on, such users can be interconnected very easily.

The problem that we address in this paper is whether we can develop a cryptocurrency suitable for such mobile ad-hoc networks with high connectivity and we propose a location based, ad-hoc and peer-to-peer cryptocurrency that

- *Dimitris Chatzopoulos is with the Department of Computer Science and Engineering, The Hong Kong University of Science and Technology, Hong Kong.*
E-mail: dcab@cse.ust.hk
- *Sujit Gujar is with the International Institute of Information Technology (IIIT), Hyderabad, India.*
E-mail: sujit.gujar@iiit.ac.in
- *Boi Faltings is with the Ecole Polytechnique Federale de Lausanne (EPFL), Switzerland.*
E-mail: boi.faltings@epfl.ch
- *Pan Hui is with the Department of Computer Science and Engineering, The Hong Kong University of Science and Technology, Hong Kong.*
E-mail: panhui@cse.ust.hk

Manuscript received: date; revised: date

requires neither an Internet connection nor devices with high computational capabilities and is based on the connectivity between users that opportunistically exchange messages. In particular, the following are our contributions.

1.1 Our Contributions

We propose *LocalCoin*, a scheme that replaces the *computational* hardness that is at the root of Bitcoin's security with the *social* hardness of ensuring that all witnesses to a transaction are colluders (users assisting the malicious user to double spend). Where computational hardness provides a *weakest-link* security guarantee - it suffices to break the scheme once - the social hardness provides a *strongest-link* guarantee: if just one witness to the transaction is not cooperating, the scheme cannot be broken. This makes it possible to apply the same idea in mobile environments without sufficient computation power or internet connectivity, while taking advantage of its distributed nature [38].

(1) We are dealing with the proof of ownership issue by proposing a distributed block chain and requiring users to at least store the blocks containing their transactions.

(2) Regarding double spending attacks, we consider the location of each user who verifies the creation of a new block. A block is accepted only when the average euclidean distance of the nodes agreeing for the block to be accepted is higher than a certain threshold. This ensures that the information regarding each transaction is spread sufficiently in the network. We show that if the network of the users of LocalCoin is dense enough, the probability of a double spending attempt to be successful is upper bounded by the inverse of the square of the number of users (Theorem 2). We also prove that the probability of a double spending attempt to be successful by a malicious user who can hire colluders to assist him in the attack, is also very low (Theorem 3).

(3) We propose an incentive scheme based on transaction and block fees that are adjusted to the ad-hoc networks in order to encourage message exchange.

In addition to the theoretical analysis, we validate our claims regarding the spread of transactions, the transaction rates, and the double spending by extensive experiments on *(i)* static graphs using tools from Random Geometric Graph theory, *(ii)* city scale simulations with mobile users with the help of the ONE simulator [24] as well as *(iii)* real data from Infocom'05, Infocom'06 and Humanet datasets [6], [36].

1.2 Applicability of *LocalCoin*

We envision LocalCoin as a location based cryptocurrency that enables small payments. Although the provided guarantees against double spending are probabilistic and leave a small chance for a double spending attack to be successful, the cost of manipulating the protocol by having a set of colluders in the proper locations outweighs the gains when the transactions are small. Apart from conventional money transactions, *LocalCoin* can also be applied to mobile computing/networking applications such as computation offloading or downloading/streaming services. Device-to-device (D2D) ecosystems, have attracted the research interest and various *serverless* architectures and frameworks have beed proposed. However, all of them either do not consider incentives for the mobile users that contribute their resources or they imply the existence of a centralised server that keeps track of the reliability and the helpfulness of each user. *LocalCoin* can fill this gap and complement any distributed credit-based incentive scheme for D2D ecosystems.

2 Related Work

After Nakamoto's original paper [31], many research groups worked on various perspectives of the Bitcoin protocol. Tschorsch and Scheuermann in their tutorial present the existing contributions and results triggered by the proposal of Bitcoin [40]. Garay *et al.* discussed applications, such as the Byzantine agreement, that can be built on top of the Bitcoin core network [19]. Darkcoin, Zerocoin and CoinShuffle, motivated by the fact that a few transaction deanonymization attacks have been reported, focus on the security and privacy aspects of Bitcoin and propose extensions to fully anonymize transactions [14] [29] [35]. Also, CoinJoin employs a multi-signature scheme to enhance the transactions' privacy [28]. CommitCoin shows a commitment scheme that harnesses the existing computational power of the Bitcoin network [8]. Miller *et al.* present a formal model of anonymous and synchronous processes that communicate using one-way public broadcasts and prove that the Bitcoin protocol achieves consensus in this model in almost any case [30].

Authors of [23] analyse the security of using Bitcoin for fast payments, where the time between the exchange of currency and goods is short. Furthermore, [37] investigates the restrictions on the transaction processing rate in Bitcoin as a function of both the bandwidth available to users and the network delay, both of which lower the efficiency of Bitcoin's transaction processing. The security analysis done by Bitcoin's creator assumes that block propagation delays are negligible compared to the time between the creation of two consecutive blocks. This assumption fails when the protocol is required to process transactions at high rates. Eyal *et al.* proposed Bitcoin-NG, the 'next generation' of Bitcoin the design of which is based on scalability [16]. In more detail, the latency is limited only by the propagation delay of the network and the bandwidth of the capabilities of the miners.

Moreover, [17], [26] argue that the Bitcoin protocol is not incentive-compatible and after presenting a game theoretic analysis they also present an attack in which colluding miners obtain a revenue larger than their fair share. Also, [32] introduces a new defence against this 51% attack via (i) presenting a block header, (ii) introducing some extra bytes, and (iii) utilising the time-stamp more effectively in the hash generation. According to [12], Bitcoin only provides eventual consistency. They propose PeerCensus, a new system, built on the Bitcoin block chain, which enables strong consistency and acts as a certification authority, manages peer identities in a peer-to-peer network, and ultimately enhances Bitcoin and similar systems with strong consistency.

3 Proposed Approach

As discussed earlier, decentralised cryptocurrencies have to address three main challenges. We, first, explain how Bitcoin addresses these challenges and then how LocalCoin encounters them.

3.1 Bitcoin

3.1.1 *Proof of ownership*

Bitcoin's main achievement is its ability to reach a consensus about a valid transaction history in a totally decentralised fashion. Bitcoin deals with the proof of ownership problem by using the concept of *block chain* based on a Merkle tree data structure. The block chain consists of a sequence of blocks connected in a hash chain, where every block imprints a set of transactions that have been collected from the network. Every miner is aware of the creation of a new block and consequently is able to validate the proof of ownership of a claimed Bitcoins. Users can employ their bitcoins by using a set of verified transactions. In order for one transaction to be counted as verified it has to belong to a block which is at least six blocks away from the current mined block in the block chain.

3.1.2 *Double spending avoidance*

Bitcoin overcomes double spending by using a proof-of-work mechanism that imposes a delay on the verification of the transaction. In order to overcome this mechanism, one has to solve a hard problem with input that takes approximately 10 minutes for a brute force algorithm to solve. There are three main ways to attempt double spending in Bitcoin protocol: (*i*) race attack, (*ii*) Finney attack and a (*iii*) 51% attack. Waiting for some new blocks to be created based on the current one can easily prevent the first two attacks; One block in the case of a race attack and six in the case of a Finney attack. However a 51% attack can collapse the whole Bitcoin network but this is extremely costly. Also, it has been proven by Eyal and Sirer that proof-of-work blockchains are vulnerable to selfish mining by attackers that control more than 1/4 of the network's mining power [17].

3.1.3 *Incentives*

Each miner gets a reward of 25 bitcoins for mining a block [10]. However, this reward halves every 4 years. Another concern is, as more and more users join as miners, the probability of mining a successful block reduces. To partially overcome this issue, miners create mining pools and they share the earnings whenever one of them solves a cryptographic puzzle [15].

3.2 LocalCoin

In this work, we propose a new Bitcoin-like cryprocurency protocol, namely *LocalCoin*, for mobile ad-hoc networks in urban areas with high device density.

3.2.1 *Proof of ownership*

LocalCoin uses a *lightweight* storage architecture by extending the concept of block chain in a distributed fashion, where each user can store as many blocks as she wants (Figure 1). The proposed distributed block chain has a redundancy factor between the users. LocalCoin, similarly to Bitcoin, stores transactions into blocks. All the transactions in the same block are collectively verified. *The size of each block* is denoted by BS. In order for one block to be created *a minimum number of users to verify each transaction*, denoted by mVu, is needed (i.e., at least $BS \cdot mVu$ users are informed about each

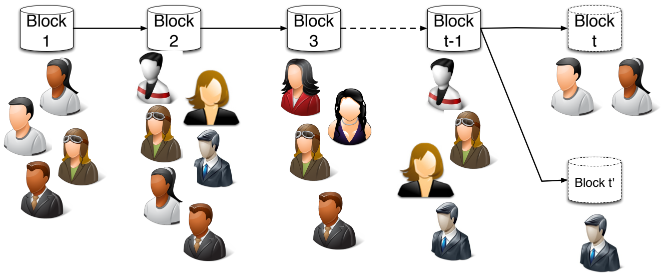


Fig. 1: Distributed block chain. Every block is stored to more than one but not to every user. Users who own at least one transaction in a block they have to store it.

transaction on one block). The relationship of these variables with the total amount of users affects the time needed to verify one block and prove the ownership of all the users that own these transactions.

3.2.2 *Double spending avoidance*

LocalCoin nullifies Bitcoin's computation overhead via incorporating a novel protocol, which is designed for the ad-hoc environment. Bitcoin's proof of work is based on the fact that cheating is improbable because a malicious user has to solve hard problems at a faster rate than the total remaining users. In LocalCoin, cheating is made very difficult because a malicious user has to misinform the majority of a set of trusted users. Every user in the LocalCoin protocol selects the users she trusts. LocalCoin avoids double spending in two ways. **(i)** The receiver of one transaction will accept the transaction if and only if she receives the transaction signed by at least *a minimum number of trusted users* of her trusted network, denoted by mTr. This constraint imposes a useful delay that spreads the transaction message to more users and increases the probability of one trusted user to detect the same input to another transaction. It is worth mentioning that any initiated transaction is signed by the sender and we assume that it is impossible for a malicious user to fake a transaction by pretending to be another user. **(ii)** During the block creation process, every participant checks for double spending attempts. To avoid fake block creation attempts by a set of collaborative malicious users, LocalCoin enforce the *average distance between the users that will verify the creation of a new block* to be more than aVd. This last constraint allows the block creation messages to be scattered to as many users as possible.

3.2.3 *Incentives*

We extend the transaction fee schema in order to motivate mobile users to participate. We propose *transaction fees* to motivate users to forward messages and *block fees* to motivate them to store as many blocks from the distributed block chain as possible. Transaction fees are important because mobile users are competing for them and they broadcast any received transaction. Every transaction includes an amount of localcoins that are collected by the mobile user who will first inform the receiver of the transaction about the transaction. Block fees are important because users store the created blocks in order to be able to verify the creation of new ones. Whenever a block is created, the mobile users that verified each transaction because they where aware of it share the localcoins that were included in these transactions as block fees.

LocalCoin is a lightweight protocol because: **(i)** any user has to only forward messages, **(ii)** a small subset of the total users are checking for the validity of a transaction message and **(iii)** users have to store the blocks that include their own transactions. The validity of LocalCoin is proved using concepts from random geometric graph (RGC) theory and its performance is depicted with the help of static graphs (Section 7.1), real trances from mobile users (Section7.2) and simulations with mobile users in a city scale (Section 7.3). In order to explain LocalCoin, we assume that it is deployed as a service in an area[1].

4 Problem Formulation

We assume a set of mobile users $\mathcal{U}$ who are registered to LocalCoin service. Any user can be malicious and in general is self-interested. Each user $i \in \mathcal{U}$ can utilise the service if she is inside the geographical area, $d_i \in \mathcal{D}$. We assume that any user i can change d_i only by moving to another location and not by manipulating it. We discuss further this assumption in Section 8.1. Two users can communicate only if their distance is within a threshold, so if a malicious user try to manipulate his location, he will be detected.

The availability of the service depends on the existence of other mobile users. Any user $i \in \mathcal{U}$ is able to exchange localcoins with another user $j \in \mathcal{U}$ by creating one transaction $t_{i\to j}$. User i needs to broadcast the transaction message that determines its characteristics. Any user j has a set of trusted users TN_j and this selection is based on social interaction between users as well as on other device to device interactions [7]. A realistic requirement of our protocol can be to force users to select their trusted peers by pairing via NFC. The selection of TN_j depends on j and they are responsible for guaranteeing that any received transaction with j as a destination should be examined before being broadcasted. The forwarding procedure is explained in detail in Section 5.1.

User i owns some localcoins and in order to prove this ownership she remembers all the transactions in which she was the receiver. The transactions are stored in blocks and each block contains more than one transaction. Each block is based on a previous block by creating a block chain. We call this chain *distributed block chain* because each block is duplicated to more than one but not to every user. An abstract instance of the distributed block chain is presented in Figure 1.

The mobile ad-hoc network nature of our service allows any other user k in the area to detect the transaction message, collect the information and contribute to this transaction. Any collected third party transaction can be used in the future by user k to earn money in terms of transaction fees and block fees. Transaction fees motivate mobile users to forward any received transaction message while block fees motivate mobile users to store collected transaction messages.

Each transaction $t_{i\to j}$ is described by a set of inputs and outputs as presented in Table 1. TN_i are mobile users trusted by i and $h(t_{*\to i})(\cdot)$ is the hash of a block that contains a transaction from anyone to user i. The outputs are: the transferred amount to user j, o_j, the transaction fees $trf_{i\to j}$, the block fees $bf_{i\to j}$, any possible change o_i and the amount of money user i owns, b_i. Transactions are verified in blocks via a mechanism that is presented in Section 5.2. Observe that a transaction $t_{i\to j}$ additionally: (i) returns, if any, change to i, (ii) transfers transaction fees to appropriate users, and (iii) pays block fees if needed. From now on, whenever we are using a past transaction as an input to a new one, where user i transfers some money, we assume that o_i can be of any of the four previous types.

Input	Output
TN_i	o_j
$h(t_{*\to i}(1))$	o_i
$h(t_{*\to i}(2))$	trf_{ij}
$\dots$	bf_{ij}
$h(t_{*\to i}(L_{i\to j}))$	b_i

TABLE 1: Transaction Template

Each user i keeps a transaction database $\mathcal{T}_i$, which contains a subset of the distributed block chain and a set of pending/unverified transactions. This database contains all the blocks user i needs to verify her own localcoins $\mathcal{B}_i \subseteq \mathcal{T}_i$ as well as other blocks in which she was present. At any time, there are two types of transactions in the network, the verified ones that can be used as an input to a new transaction and are stored in the block chain and the unverified ones. Unverified transactions are verified in bunches by a distributed consensus protocol and added to the distributed block chain.

5 Protocol

We present the basic functionalities of the LocalCoin protocol, which are categorised into three main categories; Transaction messages (Section 5.1), block creation messages (Section 5.2), and block management messages (Section 5.3).

5.1 Transaction Messages

send(i,j,$t_{i\to j}$): User i broadcasts a transaction, as described in Table 1, in order to transfer a number of localcoins to user j. The *send* command will broadcast $t_{i\to j}$ to all the single hop users (*neighbors*). Any user l operates based on the functionality of the *receive($t_{i\to j}$)* procedure as described in Algorithm 1. Whenever she receives a new transaction, she checks the sender and receiver and if she is not familiar with either of them she forwards the message hoping to collect the transaction fees. If she belongs to the trusted users of the receiver of the transaction, she examines the input transactions and signs the message if she is able to validate all of them. If she receives the message by a trusted user, she updates her transaction database according to the signed message. If she is the receiver of the message, she processes it as explained in procedure *process($t_{i\to j}$,k)*. After some iterations every user close to the sender will receive the forwarded message. If j is in the same component she will receive this message too but the transaction cannot be accepted if at least mTr of her TN_j trusted users have not signed and forwarded this message to her. The first user who forwards this message to j, regardless of being in her trusted users, will receive the amount of trf_{ij}

1. Terms "service" and "protocol" appear in this paper and the former is an implementation of the later.

Algorithm 1 Sudocode of the transaction processes

```
1: procedure SEND_$t_{i\to j}$
2:     $t_{i\to j}$=create_transaction()
3:     t=sign($t_{i\to j}$)
4:     broadcast(t,$my\_id$)
5: end procedure
1: procedure RECEIVE($t_{i\to j}$,$k$)
2:     if $t_{i\to j}$ is a send message then
3:         if $my\_id$ == j then
4:             process($t_{i\to j}$,$k$)
5:         else
6:             if $j \in TN_{my\_id}$ then
7:                 $aware\_of\_blocks \leftarrow check(t_{i\to j})$
8:                 if $aware\_of\_blocks == true$ then
9:                     t=sign($t_{i\to j}$)
10:                end if
11:            end if
12:            if $k \in TN_{my\_id}$ and $t_{i\to j}$ signed by $k$ then
13:                $update\_my\_blockchain()$
14:                t=sign($t_{i\to j}$)
15:            end if
16:            broadcast(t,$my\_id$)
17:            Update_pending_List()
18:        end if
19:    else
20:        Match_pending_List()
21:        $pending \leftarrow pending + 1$
22:    end if
23: end procedure
1: procedure PROCESS($t_{i\to j}$,$k$)
2:     if $t_{i\to j}$ is new then
3:         $first\_notifier(t_{i\to j}) \leftarrow k$
4:         $trusted(t_{i\to j}) \leftarrow 0$
5:     end if
6:     if $k \in TN_{my\_id}$ then
7:         $trusted(t_{i\to j}) \leftarrow trusted(t_{i\to j}) + 1$
8:     end if
9:     if $trusted(t_{i\to j}) > mTr$ then
10:        ack($i, j, t_{i\to j}, first\_notifier(t_{i\to j})$)
11:    end if
12: end procedure
```

if the transaction is going to be accepted by j and verified by the network. By accepting the transaction only if a subset of the trusted network signs and forwards the message to the receiver, the protocol addresses sybil attacks.

ack(i,j,$t_{i\to j}$): If j receives the message from mTr users of her trusted network TN_j, then she broadcasts an acknowledgement message. This message also determines the address of the user that will receive the transaction fees. This message is also forwarded in the similar way to that of the *send* command. There is no need for a third round because user j can only assign trf_{ij} to someone else. Everyone who receives the acknowledgement updates the knowledge about which accounts are participating in the transaction.

5.2 Block Creation Messages

build($\mathbf{BLK}(t_{i\to j}, t_{i'\to j'}, \ldots)$): Whenever a user l collects BS transactions (both *send* and *ack*), that are not yet verified, she runs a distributed consensus round [18]. Her signed message also contains her location, d_l and a current value of average distance vector $\mathbf{d}$. The distance vector has BS entries and each entry has the average distance between the users who verified the transaction.

Algorithm 2 Sudocode of block creation processes

```
1: procedure BUILDBLOCK($\mathbf{BLK}(t_{i\to j}, t_{i'\to j'}, \ldots)$)
2:     $block \leftarrow t_{i\to j}, t_{i'\to j'}, \ldots$
3:     $locations \leftarrow 0$
4:     $locations(0) = my\_GPS$
5:     broadcast($block, locations$);
6: end procedure
1: procedure RECEIVE($block, locations$)
2:     $[is\_valid, signed] \leftarrow Verify(block)$
3:     if $is\_valid$ then
4:         $avdist \leftarrow Average\_Distance(locations)$
5:         if $avdist > aVd$ then
6:             $broadcast(created\_block)$
7:         end if
8:     else
9:         broadcast("double spending attempt detected")
10:    end if
11: end procedure
```

verify($\mathbf{BLK}'(t_{i\to j}, t_{i'\to j'}, \ldots)$): The first mVu users who verify all the transactions in the *create* message and have average distance between each other bigger that aVd will share the block fees. Every user k who receives a *verify* message checks her database for unverified transactions and if she has any of the included in the message she signs them and forwards the message. Before forwarding the message, user k updates the distance entries which she has signed. If she detects a double spending attempt she deletes her entry if it has a later time-stamp or she signs her entry and adds it into the message if it has an earlier time-stamp. In case of double spending detection, user k sets the entry for the corresponding transaction to 0 and attaches and signs her detected pair with a newer timestamp. Whenever a user receives a *verify* message with the location of the user not being in her coverage radius, *(i)* she verifies any transaction she can verify, *(ii)* she notes that the location of the receiver is not in her coverage radius and she marks the location entry as false and then *(iii)* she broadcasts the message.

create($\mathbf{BLK}''(t_{i\to j}, t_{i''\to j''}, \ldots)$): Users who receive a message with transactions that are verified mVu times and have average distance bigger than aVd apart from forwarding the message they also broadcast a *create* message that defines the users who will share the block fees. Before broadcasting the *create* message, they examine if there is any entry of the mVu that has been marked by another user as false and in such case, these entries are not considered in the block creation.

5.3 Block Management Messages

delete(i,$t_{*\to i}$): We propose a garbage collection functionality that deletes every transaction that is not useful. The *delete* command is triggered after the *create* command in order to delete all the input transactions to the freshly verified ones since they can not be used any more. After deleting all the transactions of one block, the whole block is deleted. The

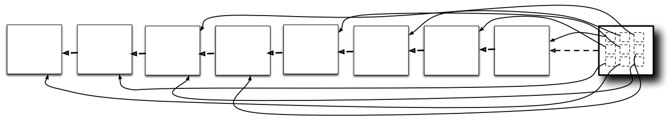


Fig. 2: A zoom view of one block with the pointers of each transaction to its parent block(s). For the sake of presentation we show that each transaction has only one parent.

motivation behind this process is to keep the size of the distributed block chain as storage efficient as possible because the mobile devices are not able to dedicate significant amount of storage for that[2].

sync(t): Any user can call *sync* function by giving only the time-stamp of her last update. By doing so, any nearby trusted user will send the newly verified transactions as well as the hash of the ones that have been deleted.

5.4 The Block Chain Evolution

If we have Λ transaction pairs per time unit, then we will have, in the long term, Λ/BS blocks per time unit. If one transaction $t_{i\rightarrow j}$ uses on average $L_{i\rightarrow j}$ transactions as an input then one block deletes $\sum_{k=1}^{BS} L_{i\rightarrow j}^{k}$ links to past blocks. In order to become one block orphan, all its transactions need to be unpointed. Each transaction is pointed by 4 links, so this block will be deleted approximately when $4 * BS$ transactions that point to that block are deleted. However, we cannot predict after how many block creations this will happen. Whenever a new block is created, the garbage collection process updates the past blocks on which the inputted transactions where placed. For any used transaction, in the creation of the new block, we delete the pointers to the parents of the used transactions. Figure 2 is a pictorial view of where one transaction of the nine in the rightmost block is used as an input to a new transaction. All the links to the parent blocks of this transaction will then be deleted. If a block has no child pointers pointing at it, it is deleted and the block that is after it then points to the one before it.

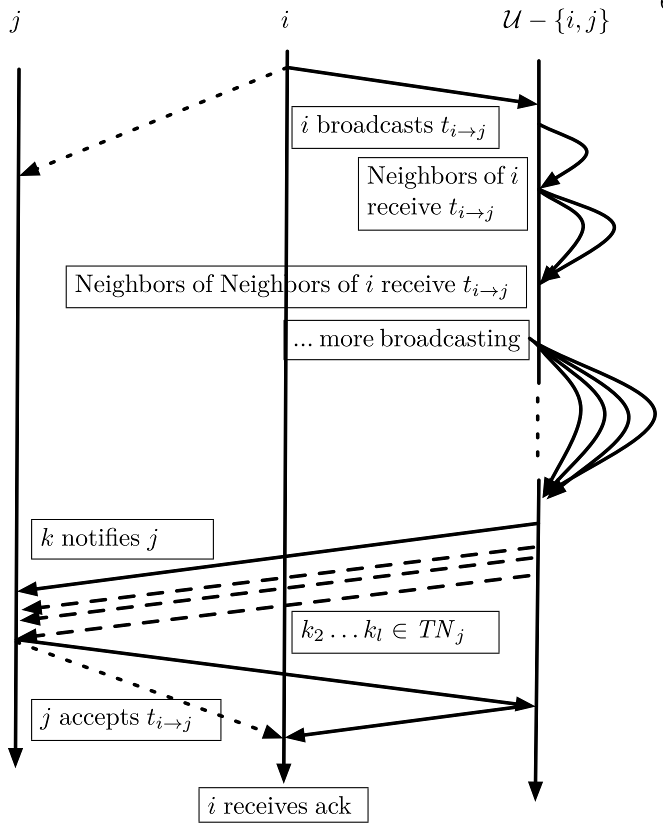
5.5 Transaction Example

Figures 3 and 4 depict the temporal and spatial evolution of one transaction. User i broadcasts $t_{i\rightarrow j}$ to her neighbors who forward $t_{i\rightarrow j}$ because they hope to get the transaction fees. Their neighbors also forward the transaction for the same reason and then user k_1 is the first to forward the message to j. If we assume for this example that $mTr = 2$, after the fourth reception of $t_{i\rightarrow j}$ user j will broadcast her ack message and she will announce user k as the receiver of the transaction fees. This pair of messages will be stored by at least i and j and potentially more users that participated in the forwarding and will be used in the block creation process.

5.6 Parameters of LocalCoin

The performance of LocalCoin depends on multiple parameters. User availability and position determine the connectivity between the users and the ability to verify new transactions but these parameters are not regulatable by LocalCoin. On

2. Bitcoin's blockchain is increasing with a rate of more than 200 MB per day [5].



j
i
$\mathcal{U} - \{i, j\}$
i broadcasts $t_{i\rightarrow j}$
Neighbors of i receive $t_{i\rightarrow j}$
Neighbors of Neighbors of i receive $t_{i\rightarrow j}$
... more broadcasting
k notifies j
$k_2 \dots k_l \in TN_j$
j accepts $t_{i\rightarrow j}$
i receives ack

Fig. 3: Temporal visualization of a transaction. In this general scenario we assume that user i wants to give some localcoins to user j who is not in her coverage area

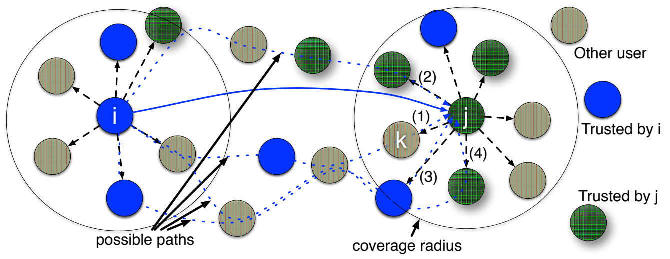


Fig. 4: Spatial representation of a simple transaction flooding scenario. For the sake of presentation we show only the coverage radius of i and j.

the other hand, the number of the trusted users needed for one transaction to be accepted (mTr), the amount of the transactions in each block (BS), the number of the users need to verify the creation of one block (mVu) and the average distance between the users who verify the block (aVd), affects the performance of LocalCoin and characterises the trade-off between the time needed to verify a new transaction, the security against double spending and the increase in the stored data. However, since LocalCoin can be categorised as a location based service, all four parameters can be adjusted based on the required transaction speed and the maximum risk of double spending as well as the rate of the creation of a new block.

6 Analysis

In this section we analyse and validate the performance of LocalCoin. Section 6.1 introduces the connectivity conditions under which the protocol is applicable and Section 6.2 presents the circumstances where a malicious user is able to successfully double spend some localcoins. Given that users' mobility increases the capacity of multihop wireless networks [20], we examine the worst-case performance of LocalCoin by considering non moving users.

6.1 Reachability

Each user i is located at $d_i \in \mathcal{D}$ and any other user j located in the coverage area of i, D_i (i.e $j \in D_i$) is able to receive any message i broadcasts. Practically, the coverage area of each user has a radius comparatively close to the coverage area of WiFi direct. We assume that every user has the same normalised coverage radius and we denote it as $r_{cov} = \frac{Wifi\_direct\_coverage}{Area\_of\_the\_supported\_service}$. Given the location of each user and the normalised coverage radius we produce a 2-dimensional random geometric graph (RGC) $G_{\mathcal{D}}(\mathcal{U}, r_{cov})$. The number of connected components of $G_{\mathcal{D}}(\mathcal{U}, r_{cov})$ depends on $|\mathcal{U}|$ and r_{cov} and has a subcritical and a supercritical phase. In the subcritical phase the number of connected components is large while in the supercritical phase it converges to one. A well known result for d-dimensional random geometric graphs is the following [21], [33]:

Lemma 1. For $|\mathcal{U}|r^d_{cov} \geq 2\log|\mathcal{U}|$ the $G^d_{\mathcal{D}}(|\mathcal{U}|, r_{cov})$ is
(1) connected with probability at least $1 - \frac{1}{|\mathcal{U}|^2}$,
(2) r-regular and
(3) the degree of every user, with high probability, is $\frac{\pi^{d/2}}{\Gamma(1+d/2)} nr^d(1 + o(1))$.

Where $\Gamma(\cdot)$ is the Gamma function and given that we consider a 2-dimensional graph ($d = 2$), $\Gamma(2) = 1$. We can rewrite the lemma as:

If $\frac{|\mathcal{U}|}{\log|\mathcal{U}|} \geq \frac{2}{r^2_{cov}}$, $G(|\mathcal{U}|, r_{cov}$ is regular with degree $d_{\mathcal{D}} = \pi|\mathcal{U}|r^2_{cov}(1 + o(1))$. Given a β-expander $d_{\mathcal{D}}$-regular graph, for every set $S \subset \mathcal{U}$, $|S| \leq |\mathcal{U}|/2$, holds $out(S) \geq \beta|S|$. Where:

$$out(S) = |\{\{u, v\}|\{u, v\} \in G_{\mathcal{D}}(|\mathcal{U}|, r_{cov}), u \in S, v \notin S\}|$$

Authors of [13], [25] state that if the nodes of the random geometric graph are produced by Poisson point process in the 2 dimensions, its density should be in the spectrum of $[0.696, 3.372]$. Simulation results converge to 1.44. If for example the subscribed users are 1000 and the coverage radius of wifi-direct is 200 meters the users will be able to form a connected graph with probability 0.999999 if the area of the supported service is less than $\pi\sqrt{\frac{2}{3}10^{11}} \approx 0.8km^2$.

Suppose user i wants to give some localcoins to user j. User i has $d_{\mathcal{D}}$ neighboors and by the properties of the expander graphs, there are $(1 + d_{\mathcal{U}})(1 + \beta)^l$ users at l hops from i. We continue expanding from i until the reachable set of users V_i has more than $|\mathcal{U}|/2$ users. User j may not be among them. However, if we expand from user j in the same way, we eventually obtain a set V_j of more than $|\mathcal{U}|/2$ users reachable from j. The sets V_i and V_j both contain more than $|\mathcal{U}|/2$ users so they must overlap. The overlap contains users on a path from i to j. In this way, we have shown that:

Symbol	Meaning
d_i	Location of user i in area of consideration $\mathcal{D}$.
r_{cov}	Normalised coverage area.
$\mathcal{U}(A)$	The users that are located in area A.
$\mathcal{M}$	Set of colluders. The users that help a malicious user m to double spend a localcoin.
$\mathcal{R}$	The area that is controled by malicious user m who tries to double spend a localcoin.

TABLE 2: Notation table with frequently used symbols.

Theorem 1. For any pair of users i and j, in the same connected component, there is a path of length at most $2(l + 1)$ from i to j, where $l = log_{(1+\beta)} \frac{\mathcal{U}}{2d_{\mathcal{D}}}$. The larger the value of β, the shorter the path between any two users.

In LocalCoin protocol, j has to be connected with at least $mTr \in TN_j$ in order to accept the transaction. Theorem 1 ensures that if user i wants to transfer some localcoins to user j the only requirement is that user j be connected with at least mTr users of her trusted network. The probability of the transaction to be successful depends on two main factors. The most important factor is both i and j must belong in the same component c, that is : $p(i \in c) \cdot p(j \in c) = |\%c| \cdot |\%c| = |\%c|^2$. Where $|\%c|$ is the fraction of the users that belong to component c. The second factor is to have at least $mTr \subset TN_j$ of j's trusted users in the component. This probability equals:

$$\sum_{l=mTr}^{|TN_j|} \binom{TN_j}{l} (p \cdot |\%c|)^l (1 - p \cdot |\%c|)^{TN_j - l} \qquad (1)$$

Where p is the probability of one user who belongs to the trusted network of j to be able to sign i's message. It is worth mentioning that in the case of moving users, the probability of having a successful transaction is increasing because mobile users can forward the received transactions whenever they make new neighbours.

6.2 Robustness Against Double Spending

Let us assume that a malicious user m wants to double spend a localcoin. We consider two types of attacks:

Passive: The attacker initiates two transactions with two different receivers and broadcasts them to two different connected components of the network. If there is only a single connected component, such attempt will be detected easily and hence, there must be at least two disjoint components for m be be successful in double spending. In Section 6.2.1 we show that probability of such attack decreases quadratically in $|\mathcal{U}|$.

Active: Another possible attack is the one where m is able, with the help of some colluders ($\mathcal{M}$), to control an area $\mathcal{R}$ and split $\mathcal{D}$ into more than one disconnected parts artificially. We discuss such attack in Sections 6.2.2 and 6.2.3.

6.2.1 A Passive double spending attack

In order for double spending to be successful, m has to employ enough colluders in order to cheat at least $2mVu$ other users. Each recipient of the fake transaction will wait until

mVu of her trusted nodes will forward her the fake message. Depending on BS, mVu and aVd the probability of double spending is changing. However, the wireless medium does not allow m to only select a number of users. Given that mTr users from both receivers are aware of m's ability to initiate this transaction, we examine how the remaining parameters affect the difficulty of double spending:

BS**:** The lower the number of transactions in one block the faster each block can be created and this allows m to try to double spend the same input and create two new blocks. If the connectivity graph between the users is partitioned, double spending is possible.

mVu**:** The higher the number of users needed to verify one transaction the more m needs to collaborate with him. For that, Lemma 1 can not hold and the connectivity graph has to be in the subcritical phase.

aVd**:** The higher the value of aVd the most difficult it is for m to double spend. Each user has on average $d_{\mathcal{D}}$ neighbours and any two users can communicate if the distance between them is less than r_{cov}, then if $\lambda = aVd/r_{cov}$, $\lambda \cdot d_{\mathcal{D}}$ users will receive the request for fake block creation.

We can paraphrase Lemma 1 and state that:

Theorem 2.

$$\text{If} \quad |\mathcal{U}|\left(\frac{aVd}{\lambda}\right)^2 \geq 2\log|\mathcal{U}|$$

$$\text{and} \quad \lambda d_{\mathcal{D}} > \frac{|\mathcal{U}|}{2},$$

double spending is possible with probability at most $1/|\mathcal{U}|^2$.

6.2.2 *Virtual-cut attack: An active double spending attack in static graphs*

Let us assume that the users' positions, $\{d_i\}$, are distributed uniformly and are static. As proved in the previous subsection, it is difficult to double spend a localcoin in the induced random graph as with high probability it consists of one major component. However, a malicious user may have detailed knowledge of the graph topology and he may artificially create a *virtual cut* by controlling users that transmit messages selectively to one part of the graph only. If a malicious user is able to double spend a localcoin by such trick, we say he is successful in a *virtual-cut attack*.

To complete a transaction in both components, each must contain enough users to verify the transaction and are at least aVd apart. This provides a lower bound on the number of users that must be controlled to create a suitable cut. Suppose a malicious user manages to induce an artificial cut as shown in Figure 5. He partitions the users into two components, A_1 with $\mathcal{U}(A_1)$ users and A_2 with $\mathcal{U}(A_2)$ users by controlling users in region B. Let $A = A_1 \setminus B$ and the number of the users in A are $|A| = \alpha\frac{|\mathcal{U}|}{2}$. Also, let the average distance between any pair of users within A to be γaVd. That is,

$$\sum_{i,j|i,j\in A} \frac{|d_i - d_j|}{|A|/2(|A|-1)} = \gamma aVd. \tag{2}$$

Let the average distance of a user in region B with a user in regions A or B to be ζaVd. The malicious user has to ensure

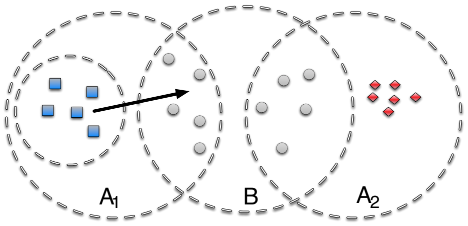


Fig. 5: Area partitioning for attempting a double spending attack.

that the average distance of any pair of users who agree the transaction is at least aVd. Let malicious user selects $\mathcal{M}$ users from region B for block creation with $\mathcal{U}(A_1)$. She needs to ensure:

$$\frac{\alpha\zeta|\mathcal{M}|\frac{|\mathcal{U}|}{2}aVd + \zeta\frac{|\mathcal{M}|^2}{4}aVd + \gamma\frac{\alpha^2|\mathcal{U}|^2}{4}aVd}{\alpha|\mathcal{M}|\frac{|\mathcal{U}|}{2} + \frac{|\mathcal{M}|^2}{4} + \frac{\alpha^2|\mathcal{U}|^2}{4}} > aVd \Leftrightarrow$$
$$2\alpha|\mathcal{M}||\mathcal{U}|(\zeta-1) + |\mathcal{M}|^2(\zeta-1) + \alpha^2|\mathcal{U}|^2(\gamma-1) > 0$$

If the malicious user chooses to (i) increase ζ or (ii) decrease α and γ, the region B will enlarge and thus he will need to add more users into $\mathcal{M}$. Intuitively, to decrease $|\mathcal{M}|$ he needs, higher ζ, or smaller values of α which in turn again need to control a bigger percentage of $\mathcal{U}$ and increase $|\mathcal{M}|$.

Example: For $\alpha = 1, \gamma = 1/2$ and $\zeta = 1.5$, m has to control at least $|\mathcal{M}| \geq (\sqrt{2}-1)|\mathcal{U}|$ or over 41% of the users.

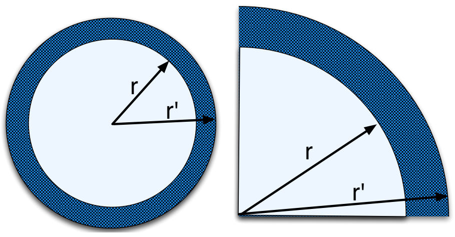


Fig. 6: Area controling for attempting a double spending attack.

Note that the above double spending attack is one of the possible attack and an attacker can isolate a region $\mathcal{R}$ by controlling all the users in a specific region. If he manages to control all the users in the shaded region, as shown in Figure 6, he can successfully create two virtual disconnected components in the connected graph. In order to block any message flow between the regions, $r' > r + 2r_{cov}$ has to hold. That is, he needs to cover an area the size of at least

$$\begin{aligned} \mathcal{R} &= \pi\left(r'^2 - (r' - 2r_{cov})^2\right) \\ &= 4\pi r_{cov}(r' - r_{cov}) \end{aligned} \tag{3}$$

The attacker can further optimize the attack by locating one of the two virtual components in a corner, so as to reduce the area he needs to control (Fig. 6). In a region A, the average distance between any two users placed with a uniform distribution is $\tilde{d}r'$ (Under extensive simulations $\tilde{d} = 0.45$). In

LocalCoin protocol, a transaction will be completed in A if the average distance between any two users is at least aVd. That is, $\tilde{d}r' > aVd$. The attacker can reduce the area that he needs to control by trying to push A further into a corner but he cannot reduce $r' < \frac{aVd}{\tilde{d}}$. Thus to create two successful transactions, one on A and one in the remaining area $\mathcal{D} \setminus A$, before he gets detected, he needs to control all the users in an area the size of:

$$\begin{aligned} \mathcal{R} &= \frac{1}{4}\pi\big((r')^2 - (r' - 2r_{cov})^2\big) \\ &= \pi r_{cov}(r' - r_{cov}) \\ &> \pi r_{cov}(\frac{aVd}{\tilde{d}} - r_{cov}) \end{aligned} \tag{4}$$

We assume that the users are uniformly distributed on a unit square. The average distance between any two users on any disc of radius r inside this unit square is $\bar{d}r$ (Under extensive simulations $\bar{d} = 0.903$). In LocalCoin protocol, we desire the information about each transaction to reach at least 50% of the users at the time of accepting a transaction. Thus, $\pi r^2 > 0.5$ which translates to $aVd > 0.36$. A higher value of aVd will slow down transactions' verification rate but it will increase security.

Example: For $aVd = \frac{1}{3}$ and $r_{cov} = 0.05$, the malicious user needs to control a region which is 0.1085 of $\mathcal{D}$. As all the users are uniformly distributed on $\mathcal{D}$, it amounts to controlling 10.85% of $\mathcal{U}$.

Theorem 3. To be able to double spend a LocalCoin by a virtual cut, an attacker needs to control at least $\mathcal{R} > \pi r_{cov}(\frac{aVd}{\tilde{d}} - r_{cov})$ fraction of the users, under the assumption that all the users are uniformly distributed and are static.

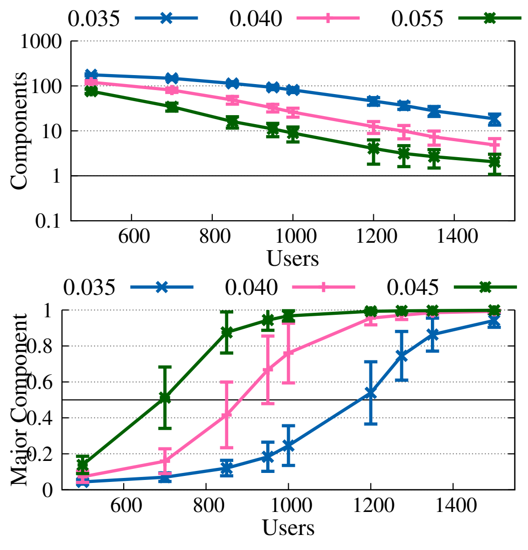
6.2.3 *Virtual-cut Attack: An active double spending attack in dynamic networks*

In reality, even if the attacker colludes with $\mathcal{M} : |\mathcal{M}| > \frac{|\mathcal{R}|}{|\mathcal{D}|}$ other users and places them appropriately to create two virtual components in the network. The remaining $\mathcal{U}\backslash\mathcal{M}$ users may be moving dynamically. So at the time when he plants a double spending attack, for the attack to be successful, none of these $|\mathcal{U}| - |\mathcal{M}|$ users should be placed in $\mathcal{R}$. Thus the probability that such the double spend attack by controlling large number of users to be successful is

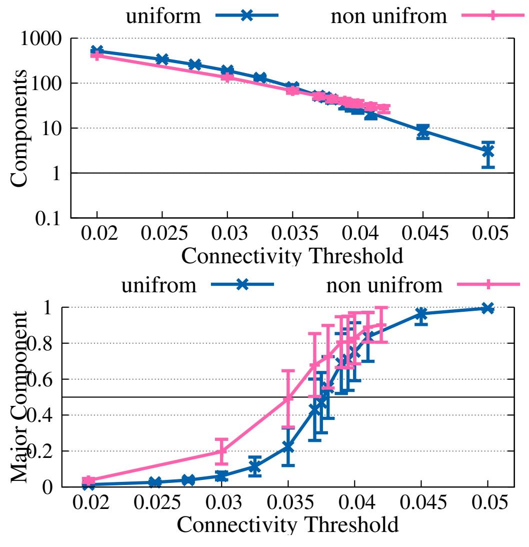
$$\frac{|\mathcal{R}|}{|\mathcal{D}|}^{(|\mathcal{U}|-|\mathcal{M}|)} \tag{5}$$

Example: For $\frac{|\mathcal{R}|}{|\mathcal{D}|} \approx 10\%$ and $|\mathcal{U}| - |\mathcal{M}| \geq 100$, the probability of a successful attack is at most 10^{-6}.

The assumptions made on the analysis of *LocalCoin* are discussed in Section 8.1.



(a) Different $|\mathcal{U}|$



(b) Different r_{cov}

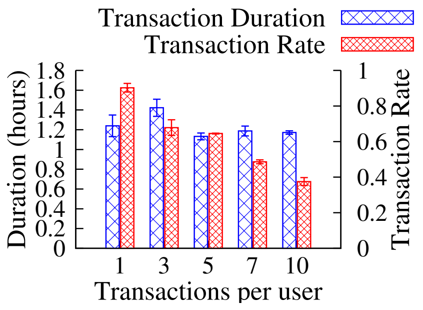
Fig. 7: The connected components and the fraction of the users in the major component in a $[0, 1] \times [0, 1]$ area.

7 Performance Evaluation of LocalCoin

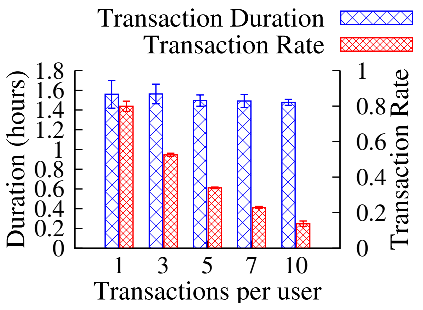
We conduct a set of simulations in static and dynamic graphs. The static random geometric graphs are produced with MATLAB (Section 7.1). We implement an event driven simulator in JAVA for dynamic graphs using real mobility traces (Section 7.2) and in order to scale up the number of the mobile users, we use the ONE simulator [24] (Section 7.3). After proving, in Section 6, that double spending is improbable in fully connected mobile ad-hoc networks, we investigate scenarios where the users are not fully connected in order to depict the robustness of LocalCoin.

7.1 Evaluation with RGGs

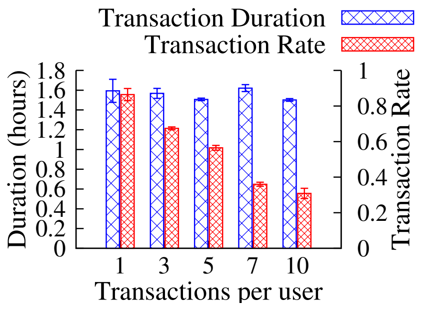
In the static analysis, we focus on the characteristics of the produced RGGs and their affect on LocalCoin. The simulations on static graphs are important because in the case where users are not moving, a malicious user m has the



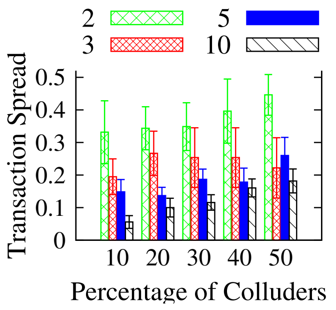
(a) Infocom 05



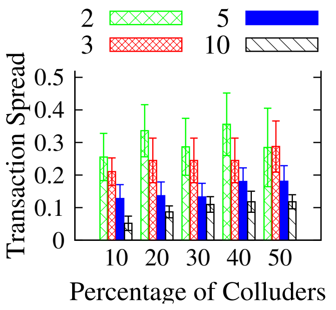
(b) Infocom 06



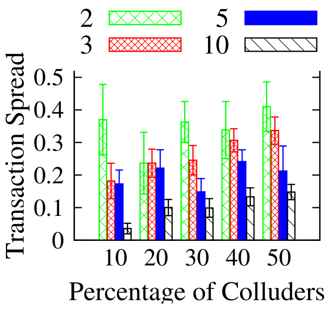
(c) Humanet



(d) Infocom 05



(e) Infocom 06



(f) Humanet

Fig. 8: Analysis of LocalCoin using users' mobility from Infocom 05, Infocom 06 and Humanet traces.

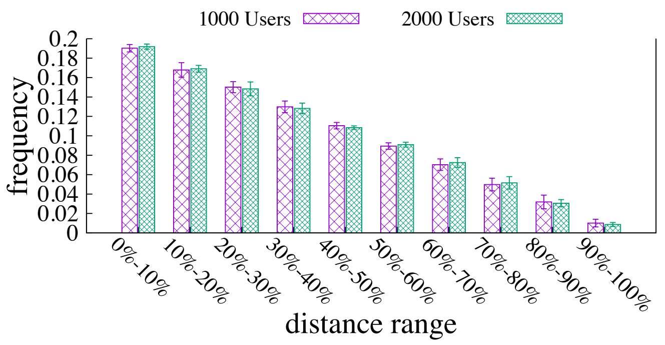


Fig. 9: Average distance between a randomly selected user with all the other users.

highest chances to successfully double spend some localcoins. We distribute uniformly the users $\mathcal{U}$ in a $[0,1]\times[0,1]$ area and we study the effect of $|\mathcal{U}|$ and r_{cov} on the number of connected components and the fraction of users in the largest connected component.

Figure 7a shows how $|\mathcal{U}|$ affects the number of connected components and the size of the major connected component for three different values r_{cov}. Figure 7b shows how r_{cov} affects the aforementioned quantities for two different user placements. Given that the uniform distribution of $\mathcal{U}$ is not a realistic case, we split the examined area into a 10 by 10 grid of equally spaced 100 cells, where the number of users in each cell is determined by a Poisson distribution with different parameter. Figure 10 depicts the used grid and the darkness of each cell depicts its populatiry. The average number of the total users is still 1000. In this non uniform case, the major component is formed for smaller values of r_{cov}. In the case of static graphs, double spending is possible when the number of connected components is more than 1, mVu is smaller than the number of the users in the components selected to double spend and aVd is small enough to apply for the users in each component. This requires at least two large components with roughly equal size. From our simulations, for $|\mathcal{U}| = 1000$ and $r_{cov} = 0.5$, the major component contains more than 90% of the users. As only one big component is getting formed the information about each transaction will spread in the network very easily.

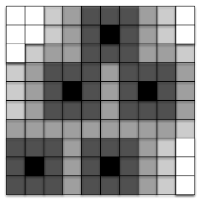


Fig. 10: Non-uniform distribution of users.

Figure 9 presents the distribution of the users in the examined area and it is useful to understand the impact of the constraint of the average distance between the users who verify the creation of a new block on the spread of the block creation process in the whole area. In order to produce the figure, we placed randomly 1000 and 2000 users and we selected randomly one of them and we measure the distance of all the other users with the selected one. Figure 9 shows that even if the imposed threshold in the creation of a block is less that 30% of the maximum possible value, more than 50% of the users will be informed about the creation process.

In order to examine LocalCoin in more realistic cases, we consider mobile users in the next two subsections.

7.2 Evaluation with Mobility Traces

We implement an event-driven simulator in Java in order to depict the performance of *LocalCoin*. We used three datasets, Infocom'05 and Infocom'06 from the Haggle project [36] and Humanet [6], which contain user mobility traces in different environments. The duration of the simulation is one day. We select the first day of the first two datasets, while Humanet is one day long. We considered all the mobile users, which are 41, 78 and 56 respectively.

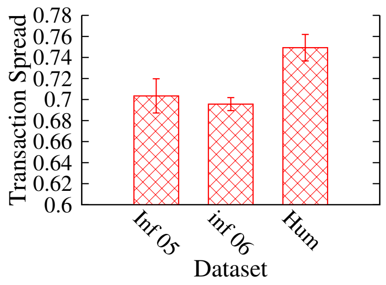


Fig. 11: The fraction of the transactions that were delivered to the destination.

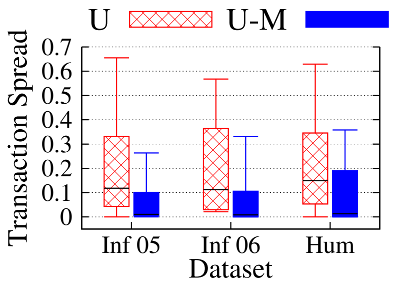


Fig. 12: Spread of Fake Transactions.

We introduce the datasets using the concepts of **Transaction Rate** and **Transaction Spread**. We define transaction rate as the fraction of the completed transactions and the transaction spread as the average fraction of the users that have stored the transaction. Figures 8a, 8b and 8c show the average time needed for one transaction to reach its destination and the transaction rate for different number of transactions per user. The receiver and the time of the transaction occurrence are generated uniformly between the users and the day. Figure 11, illustrates the transaction spread in the case where each user initiates one transaction at the beginning of the day. Note that all datasets are sparse with small number of users and hence, the transaction spread is slow resulting into the small values of transaction rate and transaction spread. This set of figures is needed to explain and understand the results produced by the simulation of a malicious user in these three traces.

Next, we examine the chances a malicious user (m) has to deliver multiple transactions with the same input (**fake transactions**) to more than one users. m tries to double spend by making at least two of the receivers of his fake transactions to accept them. However, double spending will not be successful before the creation of two blocks that contain these fake transactions, which is not possible if aVd is large enough. To simulate a double spending attack: m creates 2, 3, 5 or 10 fake transactions. Figures 8d, 8e and 8f show the average transaction spread of the fake transactions for variable number of colluders ($\mathcal{M}$). Multiple copies of the same transaction decrease the average spread of the fake transaction because the normal users ($\mathcal{U} \setminus \mathcal{M}$) receive at least two fake transactions with higher probability. Furthermore, most of these duplicates are stored by the colluders and not by the normal users. Figure 12 shows the spread of the duplicates in $\mathcal{U}$ and in $\mathcal{U} \setminus \mathcal{M}$ for the case of 5 fake transactions and $|\mathcal{M}| = 0.5|\mathcal{U}|$. On average, less than 2% of the normal users receive the fake transactions.

Only the first copy has spread like a normal transaction

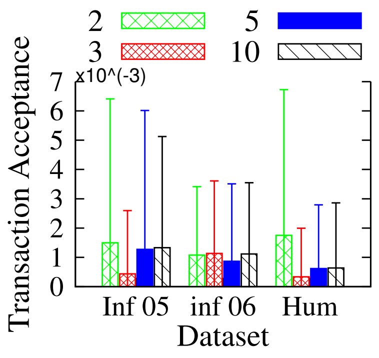


Fig. 13: Accepted Fake Transactions.

while the spread of others is decreasing dramatically since normal users are familiar with the first ones. Moreover, Figure 13 shows the probability of at least one of the receivers of the fake transactions to accept the transaction. For analysis purpose, we make it easier for m by using $mTr = 0$. That is, the receiver does not wait for a transaction approval by her trusted network. Even in this setting, the chance of m being successful in making a receiver to accept a fake transaction is $< 1\%$. Note the successful transaction does not mean that the transaction is accepted in block chain but it is only considered as pending.

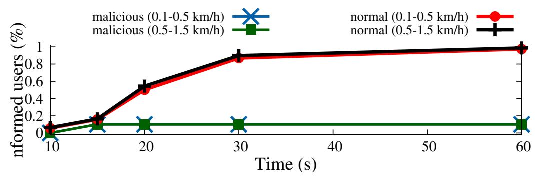
7.3 Evaluation with ONE simulator

We implement LocalCoin on ONE simulator to scrutinize its behaviour on a larger scale and examine how users' speed and coverage radius affect the chances of a malicious user to double spend. We consider a $4km^2$ area in the center of a Metropolitan city and a $|\mathcal{U}| = 1000$ mobile users. As a mobility pattern we use shortest path map based movement.

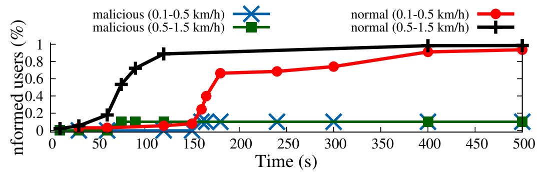
Figures 14a and 14b demonstrate the spread of a normal transaction. In Figure 14a, the coverage radius of every broadcast is 100 meters while in Figure 14b it is 50 meters[3]. Figure 14a shows that even if a malicious user has $|\mathcal{M}| = 100$ colluders to forward his fake transaction he does not have enough time to create a second fake transaction because in less than a minute almost all the users in the area will be informed of her transaction. The walking speed of the users has little influence on this case. However, if the coverage radius is 50 meters (Figure 14b), the walking speed matters. In case of the slow movement (0.1-0.5 km/h) it takes around 3 minutes for one broadcasted message to reach more than half of the users while in the case of normal walking (0.5-1.5 km/h) it takes less than 1 minute.

In order to examine more thoroughly the case when cheating is still possible, we consider the case of normally walking users (0.5 - 1.5 km/h) and coverage radius of 50 meters. A malicious user m creates two fake transactions and randomly selects the receivers of them. We examine three settings that differ in the time between the creation of the fake transactions. Figure 14c depicts how the fraction of the colluders $\frac{|\mathcal{M}|}{|\mathcal{U}|}$ affects m's chances to deliver the two fake transactions. In the first setting, m is not able to deliver 2 fake transactions, regardless of $|\mathcal{M}|$, because the time difference in the creation of the 2 them is only 10 seconds and m is in contact with the same

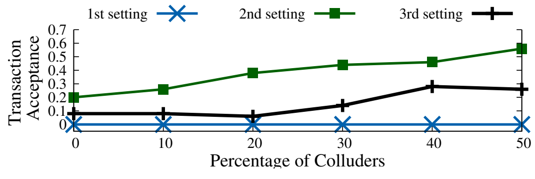
3. Wifi-direct supports a coverage radius of 200 meters but we expect it to drop radically in crowded urban areas.



(a) coverage radius = 100 meters



(b) coverage radius = 50 meters



(c) Prob. of Accepting a Fake Transaction

Fig. 14: Analysis of LocalCoin protocol using ONE.

users. In the second setting the time difference is 1 minute and m is able to successfully deliver two fake transactions but this happens either because he selects two users who have not met yet or have not met the same other users. In the third setting, m initiates the two transactions with 2 minutes difference. In this setting, m, delays the creation of the second fake transaction as much as possible in order to move as far as possible from the users that have the first transaction. Then, he is able to find a remote user that does not have the first transaction $\sim 60\%$ of the times, when half of the users in the network are assisting him. But If he wait a bit more (e.g. 30 more seconds), the first transaction will be spread in all the network, as shown in Figure 14b (after 150 seconds $> 99\%$ have received the first transaction).

It is worth reminding that, in the examined scenarios, we present the cases when a mobile user can be cheated by a malicious one in terms of accepting the transaction. However, this does not mean that the malicious user managed to double spend some localcoins. In order to succeed in that, two users have to initiate block creations and succssfully verify both of these blocks. The necessity for high density comes from this need to allow the block creation procedure to both manage to spread in a big part of the network and detect the attempts for double spending.

In summary, all three evaluation subsections are focused on describing the **area with high connectivity**, in which we argue that LocalCoin is applicable. This applicability depends on whether a normal transaction is properly spread to the whole network and a fake one is not spread and not verified. We examined both the case where the users are not moving and when they are moving either based on some available traces or by simulating their movement with the ONE simulator. In more detail, when users are static, we show that for a reasonable scenario, i.e., $|\mathcal{U}| = 1000, r_{cov} = 0.05$ the area consists of one major component implying feasibility of LocalCoin and impossibility of double spending attacks. When the users are walking, we show that with an average movement speed of 1kmph, they can complete their transactions in seconds while malicious users, even if they manage to find the proper time to initiate a fake transaction, a few minutes after that they will be detected in the block creation phase.

8 Discussion

We analysed mathematically LocalCoin by considering static graphs that are formed as random geometric graphs of which nodes are users' locations and the coverage area of the used wireless technology compared with the deployment area dictates whether two nodes are close enough to be connected. Static networks were preferred because they provide a lower bound in the performance of the protocol. In the case of mobile users, LocalCoin performs better because the transactions are more easily flooded to more users and any malicious user who wants to perform an attack has to not only consider the locations of the normal users but also their mobility, which is a very difficult task. An analysis of mobile users with different mobility patterns is part of our future work. It is notable that a conservative selection of the LocalCoin parameters (i.e., high values for mTr, BS, mVu and aVd) can guarantee that double spending is not possible, however the performance of the protocol, in terms of time, will be hindered.

8.1 Discussion about the assumptions of the analysis

The analysis presented in Section 6 is based on three major assumptions:

(A1) The mobile users are uniformly distributed in the service area.
(A2) The mobile users can not hack their location.
(A3) The colluders of a malicious user are preselected and a malicious user is not able to bribe a normal user during the double spending attack.

In the rest of this subsection we discuss the reasons that drove us to these three assumptions.

8.1.1 Assumption 1

The motivation behind distributing uniformly the mobile users in the examined area is that it makes the functionality of *LocalCoin* more challenging. Although it would be easier for a malicious user to identify critical locations to occupy if the users were not uniformly distributed, it would be also easier during the implementation of *LocalCoin* and the selection of its tuning parameters, to make such locations insufficient to create a virtual cut that can lead to a successful double spending attack. Figure 7, as explained in Section 7.1, shows that a major connected component can be formed more easily in the case where the users are not uniformly distributed.

8.1.2 Assumption 2

Although users' actual location is a core component of *LocalCoin* and a malicious user may consider asking a set of colluders to lie about their location and help him on double-spending, this is against his interest because the non-colluding neighbours of the colluders will immediately detect the attack and discard the block. Any normal user knows her location and her average coverage radius, so *LocalCoin* can handle such attacks by forcing each user to check the locations that her neighbours put during the block creation process. A parallel to the block verification distributed mechanism can detect the location manipulation attempts and intercept the double spending attacks. Although the incentives for each user to participate in such mechanism are not presented and analysed in detail, a simple defence mechanism that detects and punishes all the participants every time a location manipulation has been detected can be enough to create the necessary bias against the malicious users that try to manipulate their location.

8.1.3 Assumption 3

Our assumption is based on the fact that we envision *LocalCoin* to work on off-the-shelf mobile devices whose owners are using them on their needs while *LocalCoin* is running on the background. Mobile users are not aware of message exchange and the only way for a malicious user to double-spend is to implement another version off *LocalCoin* and install it in his colluders.

8.2 Getting assistance from a fixed network

It is worth mentioning that *LocalCoin* can benefit from fixed networks in the small scale (university campus scale) to increase the speed of the message forwarding and decrease the importance of user's density. The access points of a fixed network can be treated as normal users who do not have storage capabilities (i.e. $\mathcal{T}_{AP} = \emptyset$, $\forall$ access point) and do not compete for the transaction fees but broadcast all the incoming messages to the other nodes of the fixed network who broadcast the messages to the associated mobile devices. Moreover, the access points can be used for the block creation process as a guarantee for the average distance between the mobile users. In more detail, each user who verify the creation of a new block can attach a recently signed message that she received from the closest AP in order to provide a robust estimation of her location. However, this extra functionalities that are provided by a fixed network can only improve the performance of *LocalCoin* on the practical level.

Device to device architectures, with main representative being the 5G, are becoming more and more popular. The Wifi-direct technology is getting more mature and is adapted by many customer products, while, the design of LTE-direct is moving in this route. Motivated by advances in this direction and considering that any existing infrastructure, like an institutional network in a university campus, can only improve the coverage of LocalCoin, we argue that protocols like LocalCoin are applicable and implementable.

9 Conclusion and Future Work

In conclusion, *LocalCoin* is a decentralised cryptocurrency built on principles similar to Bitcoin but avoids any need of the Internet and computing overheads. We used a distributed block chain structure to store the produced transactions in the form of blocks and we proposed a set of four parameters to show the existing trade-offs in our proposal due to the mobile ad-hoc nature. These 4 parameters can be adjusted to provide the required security guarantees, in terms of double spending, and at the same time affect the maximum transaction rate. LocalCoin is applicable in dense areas of mobile users and if the users are fully connected, the double spending is impossible. Our future extensions of *LocalCoin* will be focused on building defence mechanisms against other types of attacks where mobile users form coalitions and perform abnormally without their incentives being aligned with the incentives of the normal mobile users.

References

[1] BITCOIN COMMUNITY. Bitcoin Protocol rules. https://en.bitcoin.it/wiki/Protocol_rules. Online; accessed 2 August 2016.

[2] BITCOIN COMMUNITY. Bitcoin Protocol specification. https://en.bitcoin.it/wiki/Protocol_documentation. Online; accessed 2 August 2016.

[3] BITCOIN COMMUNITY. Bitcoin source. https://github.com/bitcoin/bitcoin. Online; accessed 2 August 2016.

[4] Blockchain. Blockchain info. http://blockchain.info. Online; accessed 2 August 2016.

[5] Blockchain info. Blockchain size. https://blockchain.info/charts/blocks-size. Online; accessed 2 August 2016.

[6] J. M. Cabero, V. Molina, I. Urteaga, F. Liberal, and J. L. Martin. CRAWDAD data set tecnalia/humanet (v. 2012-06-12). http://crawdad.org/tecnalia/humanet/, June 2012.

[7] X. Chen, B. Proulx, X. Gong, and J. Zhang. Social trust and social reciprocity based cooperative d2d communications. In *Proceedings of the Fourteenth ACM International Symposium on Mobile Ad Hoc Networking and Computing*, MobiHoc '13, pages 187–196, New York, NY, USA, 2013. ACM.

[8] J. Clark and A. Essex. Commitcoin: Carbon dating commitments with bitcoin. In A. Keromytis, editor, *Financial Cryptography and Data Security*, volume 7397 of *Lecture Notes in Computer Science*, pages 390–398. Springer Berlin Heidelberg, 2012.

[9] CoinDesk. CoinDesk. http://www.coindesk.com/. Online; accessed 2 August 2016.

[10] CoinDesk. How Bitcoin Mining Works. http://www.coindesk.com/information/how-bitcoin-mining-works/. Online; accessed 2 August 2016.

[11] Coinmarketcap. Crypto-Currency Market Capitalizations. http://coinmarketcap.com. Online; accessed 2 August 2016.

[12] C. Decker, J. Seidel, and R. Wattenhofer. Bitcoin meets strong consistency. *CoRR*, abs/1412.7935, 2014.

[13] O. Dousse, M. Franceschetti, and P. Thiran. Information theoretic bounds on the throughput scaling of wireless relay networks. In *INFOCOM 2005. 24th Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings IEEE*, volume 4, pages 2670–2678 vol. 4, March 2005.

[14] E. Duffield and K. Hagan. Darkcoin: Peertopeer cryptocurrency with anonymous blockchain transactions and an improved proofofwork system. 2014.

[15] I. Eyal. The miner's dilemma. In *36th IEEE Symposium on Security and Privacy, S&P 2015*, 2015.

[16] I. Eyal, A. E. Gencer, E. G. Sirer, and R. V. Renesse. Bitcoin-ng: A scalable blockchain protocol. In *13th USENIX Symposium on Networked Systems Design and Implementation (NSDI 16)*, pages 45–59, Santa Clara, CA, Mar. 2016. USENIX Association.

[17] I. Eyal and E. G. Sirer. Majority is not enough: Bitcoin mining is vulnerable. *CoRR*, abs/1311.0243, 2013.

[18] M. J. Fischer. The consensus problem in unreliable distributed systems (a brief survey). In *Proceedings of the 1983 International FCT-Conference on Fundamentals of Computation Theory*, pages 127–140, London, UK, UK, 1983. Springer-Verlag.

[19] J. Garay, A. Kiayias, and N. Leonardos. *The Bitcoin Backbone Protocol: Analysis and Applications*, pages 281–310. Springer Berlin Heidelberg, Berlin, Heidelberg, 2015.

[20] M. Grossglauser and D. N. C. Tse. Mobility increases the capacity of ad hoc wireless networks. *IEEE/ACM Trans. Netw.*, 10(4):477–486, Aug. 2002.

[21] P. Gupta and P. R. Kumar. The capacity of wireless networks. *Information Theory, IEEE Transactions on*, 46(2):388–404, 2000.

[22] A. Juels and J. G. Brainard. Client puzzles: A cryptographic countermeasure against connection depletion attacks. In *NDSS*, volume 99, pages 151–165, 1999.

[23] G. O. Karame. Two bitcoins at the price of one? double-spending attacks on fast payments in bitcoin. In *In Proc. of Conference on Computer and Communication Security*, 2012.

[24] A. Keränen, J. Ott, and T. Kärkkäinen. The ONE Simulator for DTN Protocol Evaluation. In *SIMUTools '09: Proceedings of the 2nd International Conference on Simulation Tools and Techniques*, New York, NY, USA, 2009. ICST.

[25] Z. Kong and E. M. Yeh. On the critical density for percolation in random geometric graphs. In *Proceedings of IEEE International Symposium on Information Theory, ISIT*, pages 1–7. Citeseer, 2007.

[26] Y. Lewenberg, Y. Bachrach, Y. Sompolinsky, A. Zohar, and J. S. Rosenschein. Bitcoin mining pools: A cooperative game theoretic analysis. In *Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2015, Istanbul, Turkey, May 4-8, 2015*, pages 919–927, 2015.

[27] Mapofcoins. Explore the visualized history of the cryptocurrencies from their whitepapers up to present days. http://mapofcoins.com. Online; accessed 2 August 2016.

[28] G. Maxwell. Coinjoin: Bitcoin privacy for the real world. In *Post on Bitcoin Forum. Available online: https://bitcointalk.org/index. php*, 2013.

[29] I. Miers, C. Garman, M. Green, and A. D. Rubin. Zerocoin: Anonymous distributed e-cash from bitcoin. In *Proceedings of the 2013 IEEE Symposium on Security and Privacy*, SP '13, pages 397–411, Washington, DC, USA, 2013. IEEE Computer Society.

[30] A. Miller and J. J. LaViola Jr. Anonymous byzantine consensus from moderately-hard puzzles: A model for bitcoin. *Retrieved from Anonymous Byzantine Consensus from Moderately-Hard Puzzles: A Model for Bitcoin*, 2014.

[31] S. Nakamoto. Bitcoin: A peer-to-peer electronic cash system, 2009.

[32] G. Paul, P. Sarkar, and S. Mukherjee. Towards a more democratic mining in bitcoins. In A. Prakash and R. Shyamasundar, editors, *Information Systems Security*, volume 8880 of *Lecture Notes in Computer Science*, pages 185–203. Springer International Publishing, 2014.

[33] M. Penrose. *Random geometric graphs*, volume 5. Oxford University Press Oxford, 2003.

[34] R. L. Rivest, A. Shamir, and D. A. Wagner. Time-lock puzzles and timed-release crypto. Technical report, Cambridge, MA, USA, 1996.

[35] T. Ruffing, P. Moreno-Sanchez, and A. Kate. *CoinShuffle: Practical Decentralized Coin Mixing for Bitcoin*, pages 345–364. Springer International Publishing, Cham, 2014.

[36] J. Scott, R. Gass, J. Crowcroft, P. Hui, C. Diot, and A. Chaintreau. CRAWDAD data set cambridge/haggle (v. 2006-01-31). Downloaded from http://crawdad.org/cambridge/haggle/, Jan. 2006.

[37] Y. Sompolinsky and A. Zohar. Accelerating bitcoin's transaction processing.

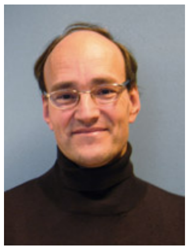
[38] E. Syta, I. Tamas, D. Visher, D. I. Wolinsky, and B. Ford. Decentralizing authorities into scalable strongest-link cothorities. *CoRR*, abs/1503.08768, 2015.

[39] M. B. Taylor. Bitcoin and the age of bespoke silicon. In *Proceedings of the 2013 International Conference on Compilers, Architectures and Synthesis for Embedded Systems*, CASES '13, pages 16:1–16:10, Piscataway, NJ, USA, 2013. IEEE Press.

[40] F. Tschorsch and B. Scheuermann. Bitcoin and beyond: A technical survey on decentralized digital currencies. *IEEE Communications Surveys Tutorials*, PP(99):1–1, 2016.

Dimitris Chatzopoulos received his Diploma and his Msc in Computer Engineering and Communications from the Department of Electrical and Computer Engineering of University of Thessaly, Volos, Greece. He is currently a PhD student at the Department of Computer Science and Engineering of The Hong Kong University of Science and Technology and a member of HKUST-DT System and Media Lab. During the summer of 2014, he was a visiting PhD student at Ecole polytechnique federale de Lausanne (EPFL). His main research interests are in the areas of mobile computing, device–to–device ecosystems and cryptocurrencies.

Sujit Gujar is an Assistant Professor at the Machine Learning Laboratory@IIITH. Prior to this, he was a Sr. Research Associate at Indian Institute of Science. He worked as a post-doctoral researcher at Ecole polytechnique federale de Lausanne (EPFL). He also worked as a research scientist with Xerox Research Centre India where he contributed in developing a technology that enables enterprises to use crowdsourcing as a complimentary workforce. His research interests are Game Theory, Mechanism Design, Machine Learning, and Cryptography applied to modern web and AI applications such as Auctions, Internet Advertising, Crowdsourcing, and multi-agent systems. His doctoral thesis was awarded alumni medal for best doctoral thesis in the Department of Computer Science and Automation at Indian Institute of Science. He was a recipient of Infosys fellowship for his doctoral research. He has co-authored 5 journal publications, 1 book chapter and 22 conference/workshop papers. He has 11 patents on his name.



Boi Faltings is a full professor of computer science at the Ecole Polytechnique Federale de Lausanne (EPFL), where he heads the Artificial Intelligence Laboratory, and has held visiting positions at NEC Research Institute, Stanford University and the HongKong University of Science and Technology. He has co-founded 6 companies in e-commerce and computer security and acted as advisor to several other companies. Prof. Faltings has published over 300 refereed papers and graduated over 30 Ph.D. students, several of which have won national and international awards. He is a fellow of the European Coordinating Committee for Artificial Intelligence and a fellow of the Association for Advancement of Artificial Intelligence (AAAI). He holds a Diploma from ETH Zurich and a Ph.D. from the University of Illinois at Urbana-Champaign.

Pan Hui received his Ph.D degree from Computer Laboratory, University of Cambridge, and earned his MPhil and BEng both from the Department of Electrical and Electronic Engineering, University of Hong Kong. He is currently a faculty member of the Department of Computer Science and Engineering at the Hong Kong University of Science and Technology where he directs the HKUST-DT System and Media Lab. He also serves as a Distinguished Scientist of Telekom Innovation Laboratories (T-labs) Germany and an adjunct Professor of social computing and networking at Aalto University Finland. Before returning to Hong Kong, he has spent several years in T-labs and Intel Research Cambridge. He has published more than 150 research papers and has some granted and pending European patents. He has founded and chaired several IEEE/ACM conferences/workshops, and has been serving on the organising and technical program committee of numerous international conferences and workshops including ACM SIGCOMM, IEEE Infocom, ICNP, SECON, MASS, Globecom, WCNC, ITC, ICWSM and WWW. He is an associate editor for IEEE Transactions on Mobile Computing and IEEE Transactions on Cloud Computing.