

# LocalChains: A Decentralized Ledger with a tiny global state

Orfeas Stefanos Thyfronitis Litos and Aggelos Kiayias

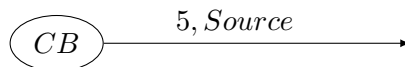
University of Edinburgh  
o.thyfronitis@ed.ac.uk, akiayias@inf.ed.ac.uk

**Abstract.**

## 1 Introduction

## 2 Transaction DAGs

There exists a genesis transaction that contains all coins to ever exist. This transaction is hardcoded in the protocol and is by definition *valid* and *trustworthy* for all nodes.



**Fig. 1:** Coinbase transaction

All other transactions must have the following attributes to be valid:

- Inputs that correspond to valid unspent outputs (UTXOs) of previous transactions.
- Outputs that have a total value that is less than or equal to the total value of the inputs, with a public key corresponding to each output. Each public key can exist only once in the valid and trustworthy DAG of a node<sup>1</sup>.
- One signature for each input, made with the private key that corresponds to the public key of the corresponding previous output.

At any point in time, *Alice* considers a DAG  $X$  of transactions as *valid* and *trustworthy*. She may be informed by Charlie about the existence another DAG of transactions,  $Y$ . To consider this DAG valid, *Alice* must verify that

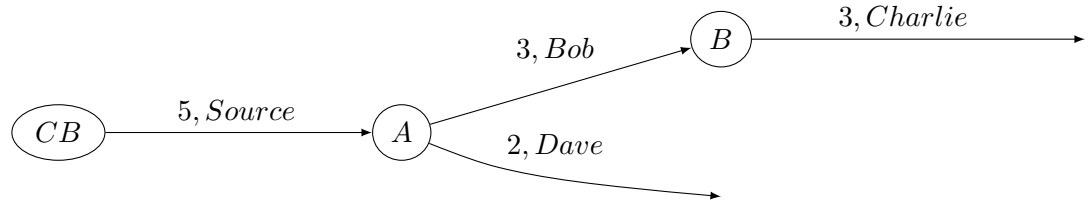
---

<sup>1</sup> This is to ensure that a transaction input cannot be connected with two different transaction outputs. It is not entirely clear that this limitation is necessary though.

- all the transactions are valid and
- all root transactions of  $Y$  are connected to UTXOs in  $X$ .

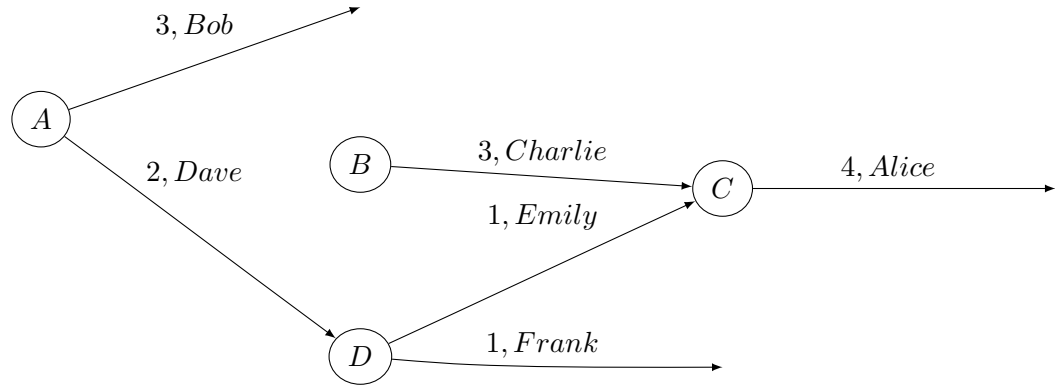
We will later see what course of action she is expected to take in case the second requirement does not hold.

For example, let  $X$  be:

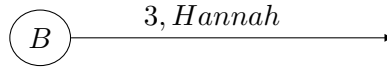


**Fig. 2:** Base DAG  $X$

Then *Alice* would consider  $Y_1$  valid, but  $Y_2$  invalid:



**Fig. 3:** Good DAG  $Y_1$



**Fig. 4:** Bad DAG  $Y_2$

$X$  and  $Y_2$  cannot be valid at the same time because there is a conflict regarding which private key can spend the output of the transaction  $B$ .

The existence of the two versions of the transaction  $B$  is proof that  $Bob$  attempted to double spend his output from transaction  $A$ .  $Alice$  and  $Charlie$  just discovered a proof of fraud committed by  $Bob$  and can use this proof of fraud so that one of the two can keep the coins in the transaction DAG and the other can get a refund from the network. Here is where the global part of the infrastructure comes into play.

### 3 Global Trust Ledger

Observe that in the previous example, it was never said that  $Alice$  thinks that  $Y_1$  is *trustworthy*;  $Alice$  only perceives  $Y_1$  as *valid*. The lack of a global transaction ledger makes it possible that  $Charlie$ ,  $Dave$  or  $Emily$  have double spent the new transactions  $Alice$  learned about, thus  $Alice$  has no reason to assume that she exclusively owns the coins at the output of transaction  $C$ . In order to be insured in case a previous transaction has been double spent, there has to be some kind of global, public commitment from each of the players in question (or someone vouching for them) that a refund will be given in case fraud is committed.