TECHNISCHE UNIVERSITÄT WIEN
Vienna | Austria

FAKULTÄT FÜR !NFORMATIK
Faculty of Informatics

S&P SECURITY & PRIVACY GROUP

Atomic Multi-Channel Updates with Constant Collateral in Bitcoin-Compatible Payment-Channel Networks

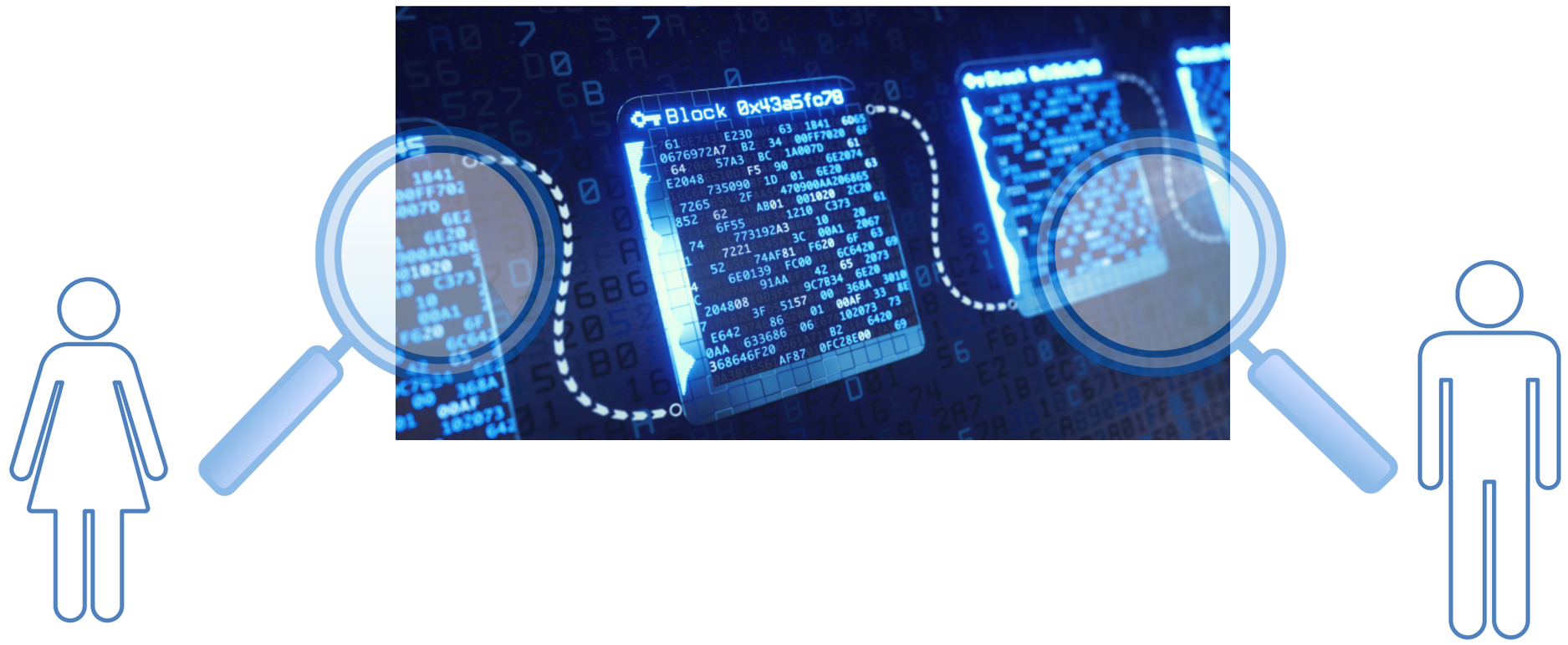
Christoph Egger[1], **Pedro Moreno-Sanchez[2]**, Matteo Maffei[2]

@siccegge @pedrorechez @matteo_maffei

[1]Friedrich-Alexander-University, Erlangen-Nuremberg
[2]TU Vienna

Scaling Bitcoin
Tel Aviv, Sep12th 2019

Scalability Issues



- Decentralized data structure recording each transaction in order to provide public verifiability
- Global consensus: everyone checks the whole blockchain

Bitcoin's transaction rate: ~10 tx/sec

Visa's transaction rate: ~10K tx/sec

Scalability Solutions?

- On-chain (tweak consensus)
  e.g., DAG Blockchain, sharding, ...
- Off-chain (use blockchain only for disputes)
  e.g., Payment Channel Networks

Lightning Network
(Bitcoin)

Raiden Network
(Ethereum)

Many other projects (Bolt, Perun, Liquidity Network...)

Scalability Solutions?

- On-chain (tweak consensus)
  e.g., DAG Blockchain, sharding, ...
- Off-chain (use blockchain only for disputes)
  e.g., Payment Channel Networks

Lightning Network
(Bitcoin)

Raiden Network
(Ethereum)

Many other projects (Bolt, Perun, Liquidity Network...)

Background on Payment Channels

Payment Channels: Open

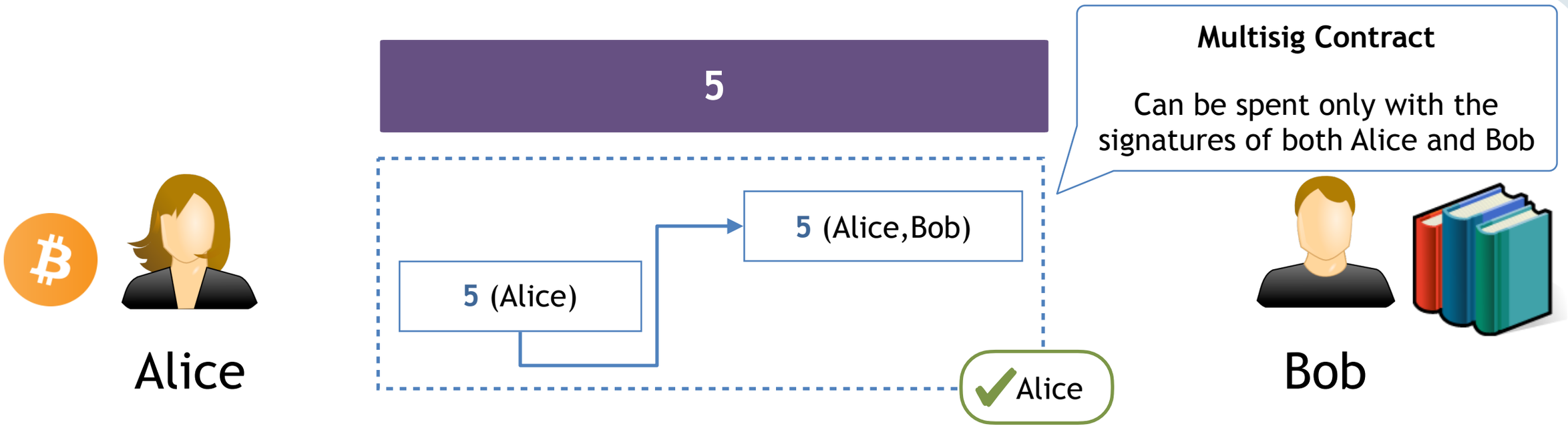
5

Alice

Bob

Blockchain

Payment Channels: Open

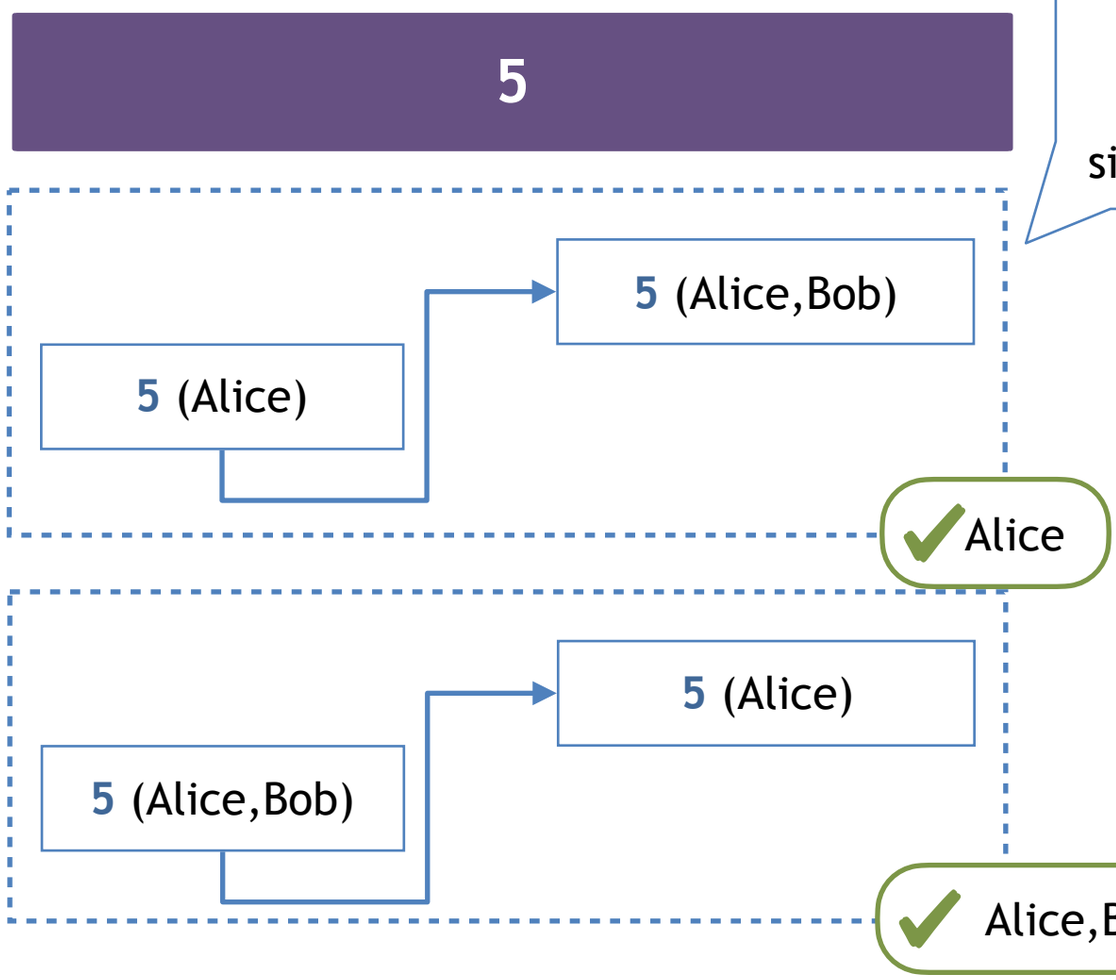


Blockchain

- Alice creates multisig contract to deposit money on the channel

Payment Channels: Open

Alice



Multisig Contract

Can be spent only with the signatures of both Alice and Bob

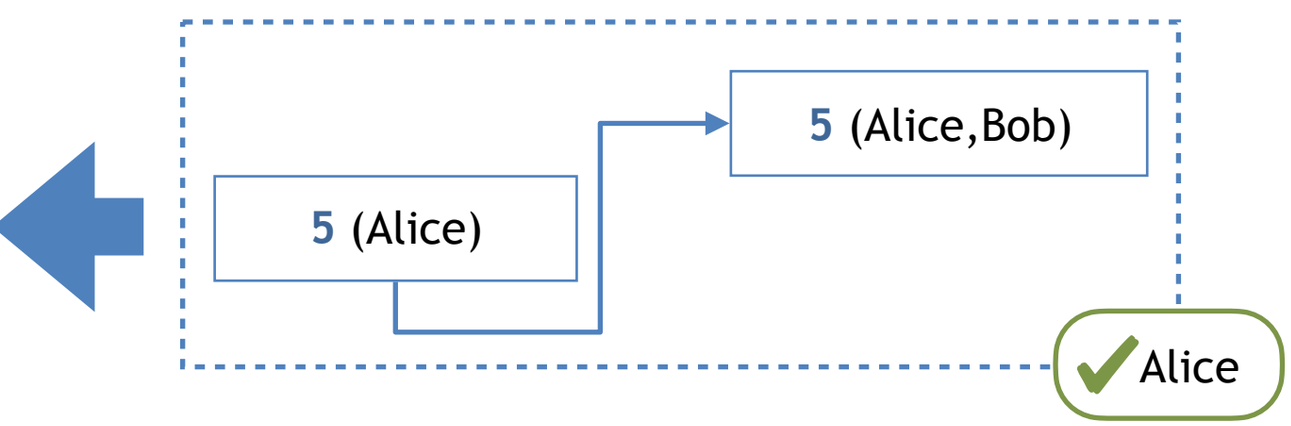
Bob

Blockchain

- Alice creates multisig contract to deposit money on the channel
- Alice lets Bob sign a refund transaction to unlock the money

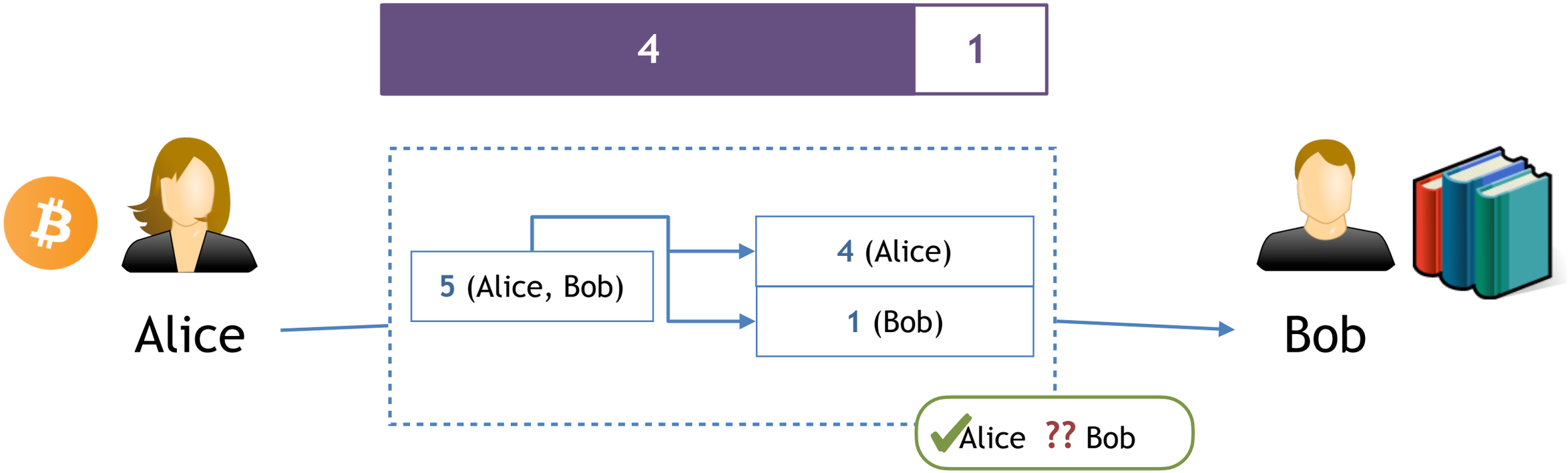
Payment Channels: Open

Blockchain

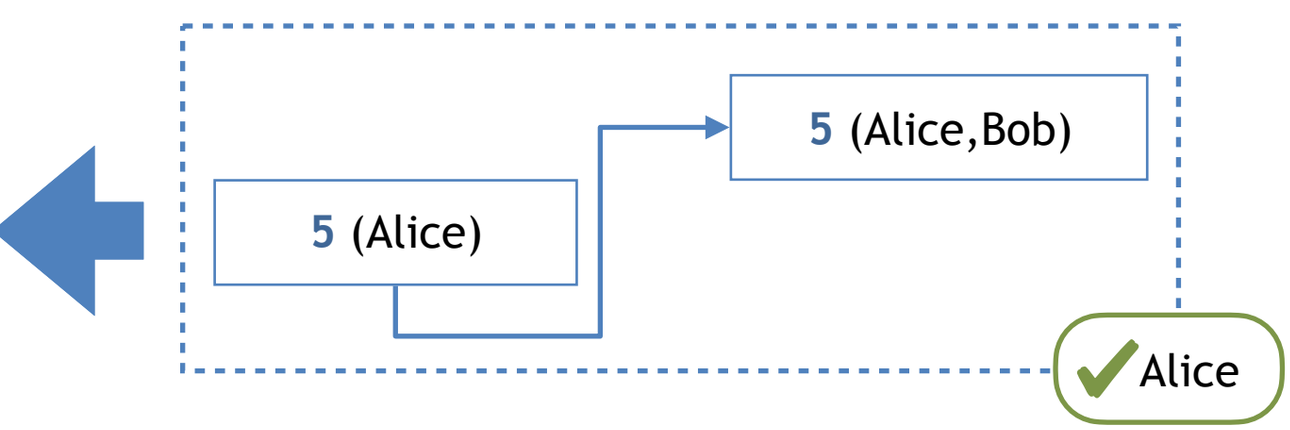


- Alice creates multisig contract to deposit money on the channel
- Alice lets Bob sign a refund transaction to unlock the money
- Alice places the multisig contract onchain

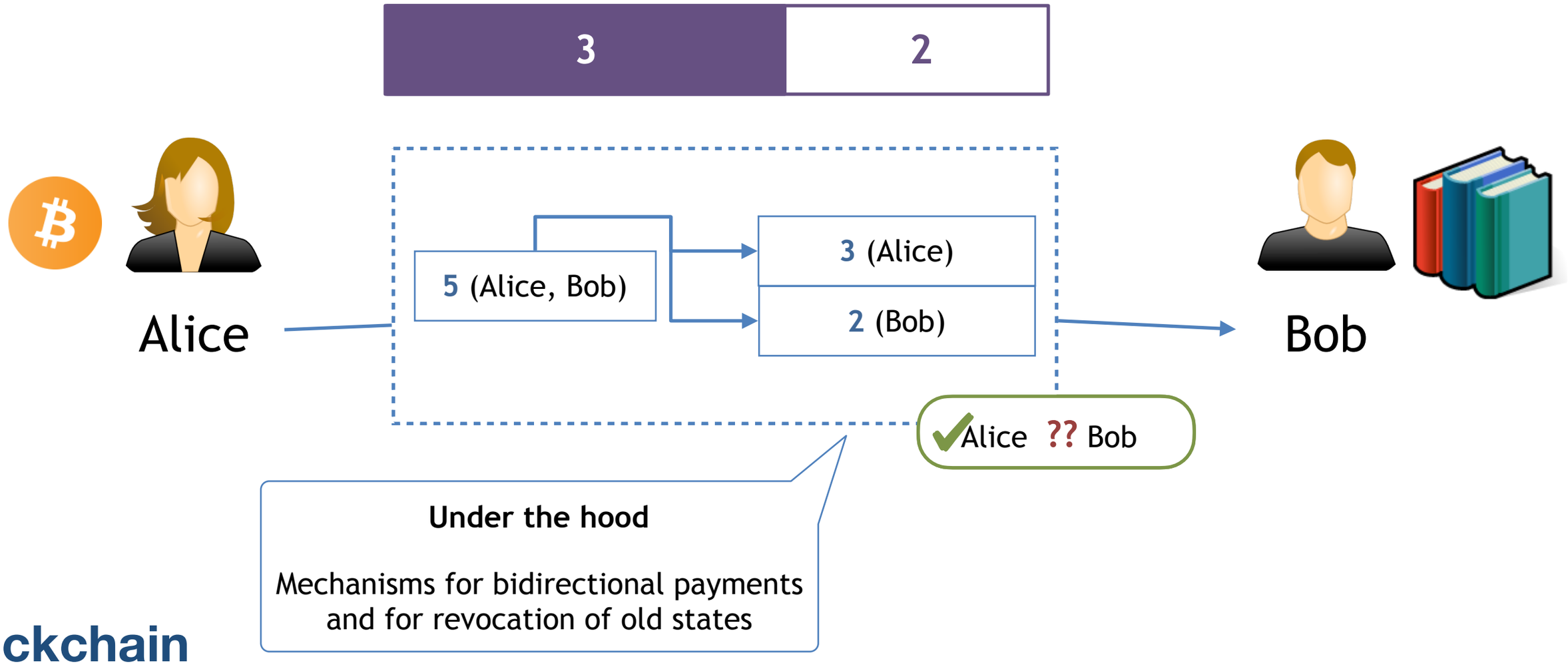
Payment Channels: Transactions



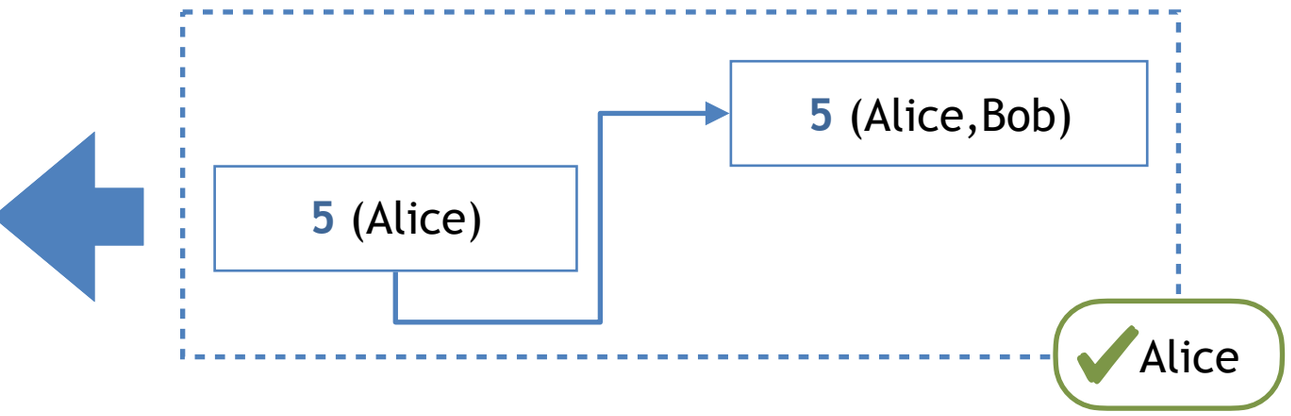
Blockchain



Payment Channels: Transactions

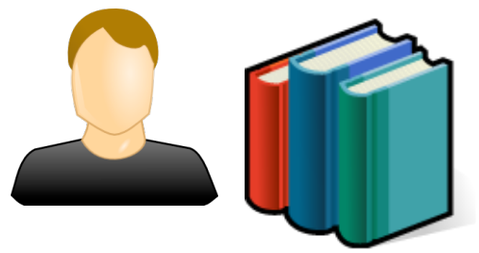


Blockchain



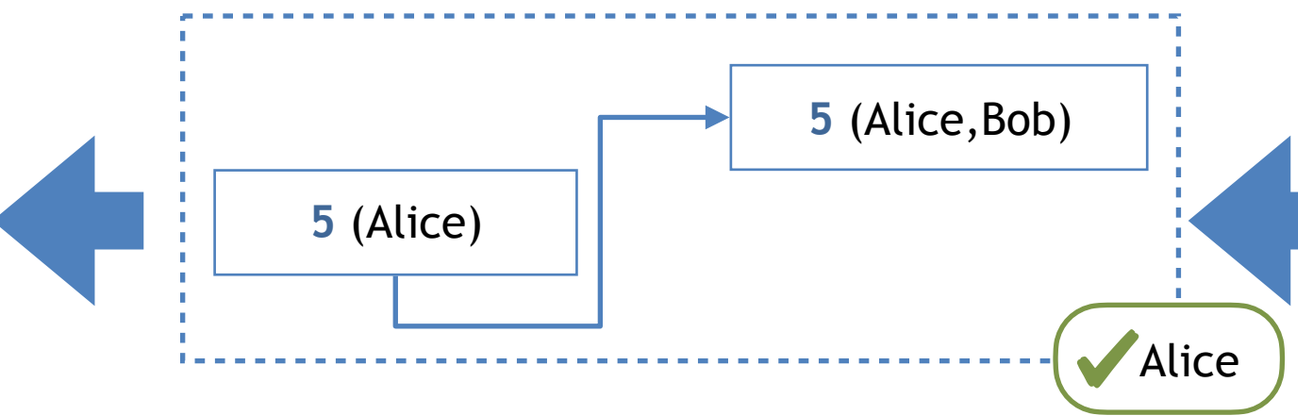
Payment Channels: Close

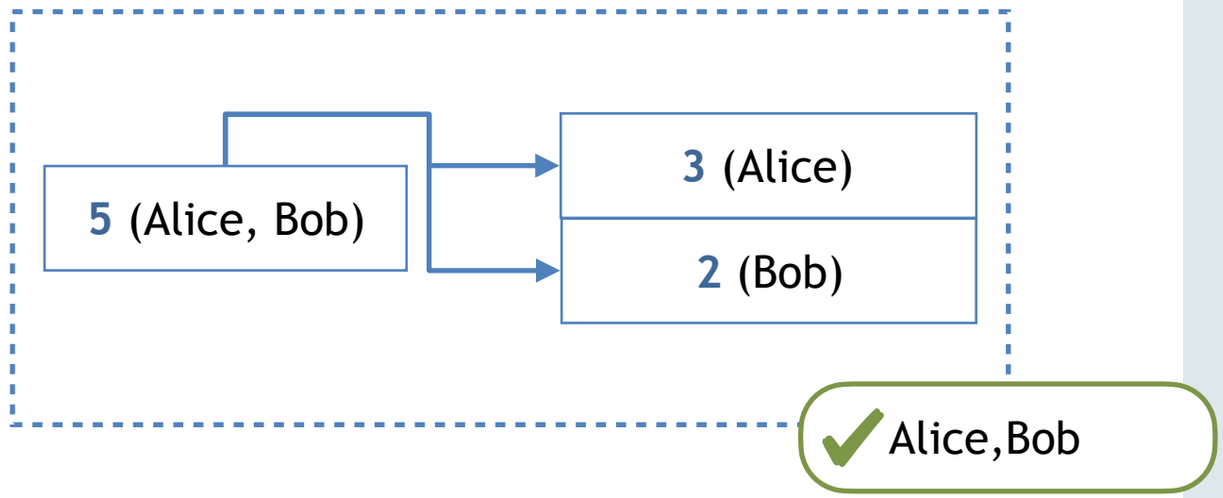
Alice



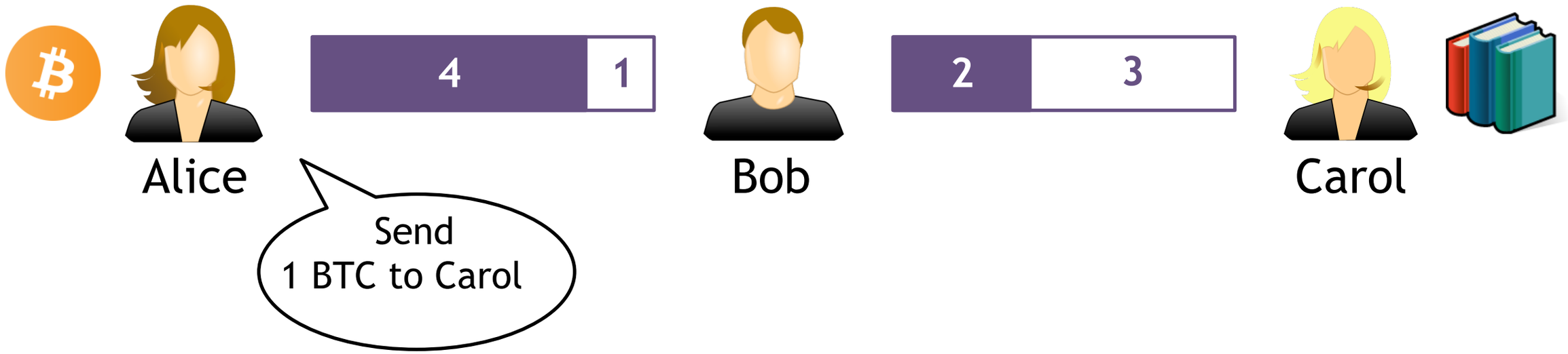
Bob

Blockchain



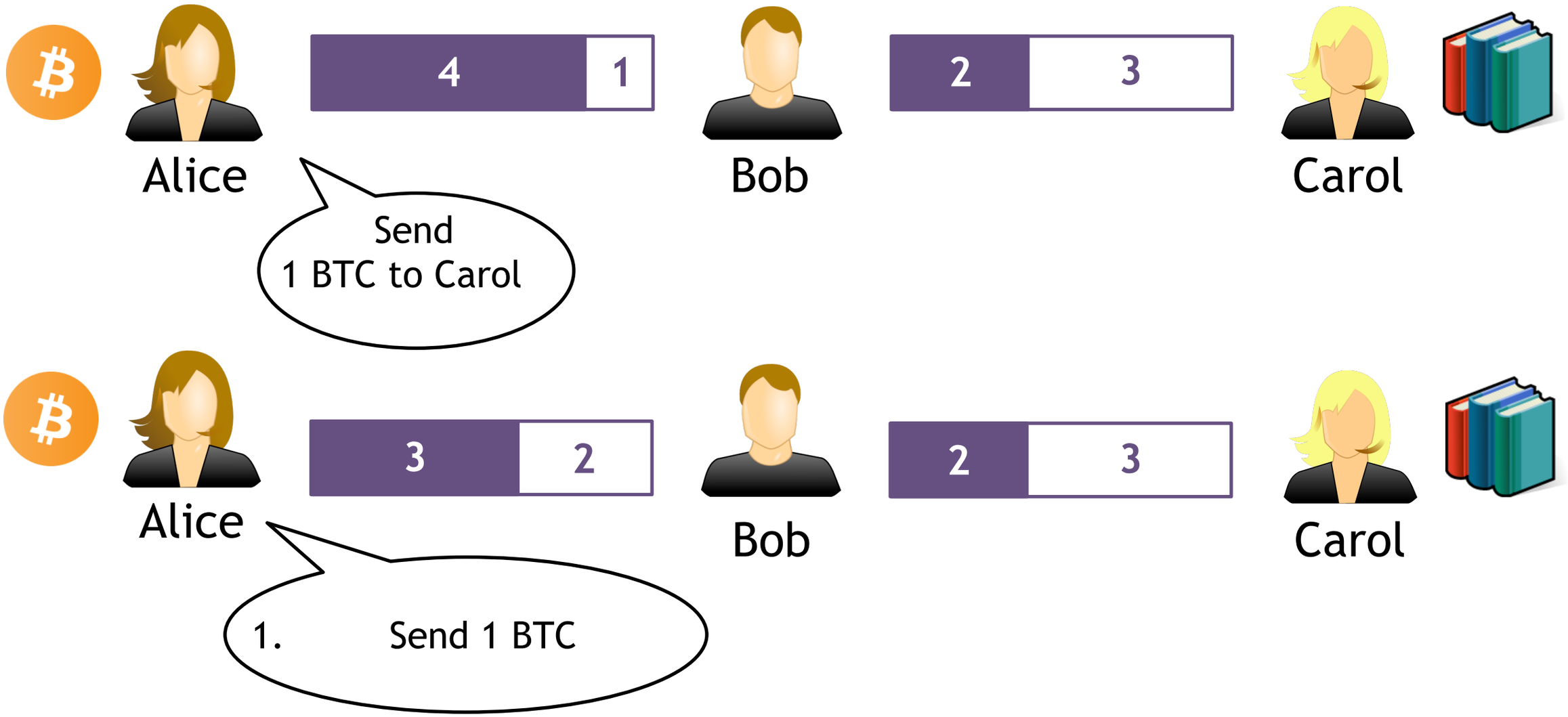


Payment Channel Networks (PCNs)



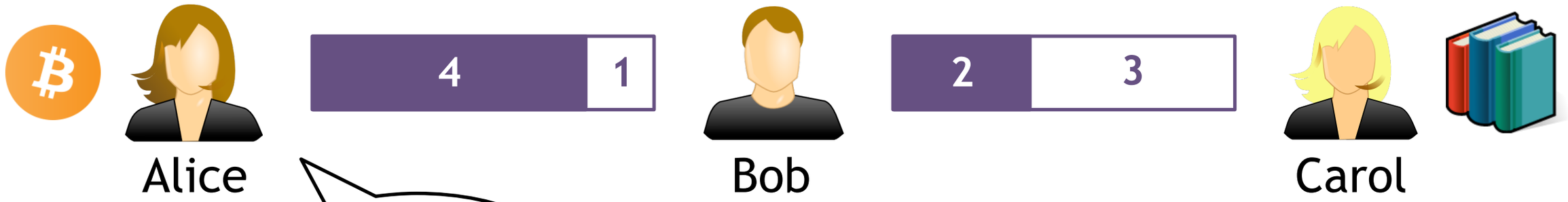
One cannot open channels with everyone...
⇒ exploit channel paths!

Payment Channel Networks (PCNs)

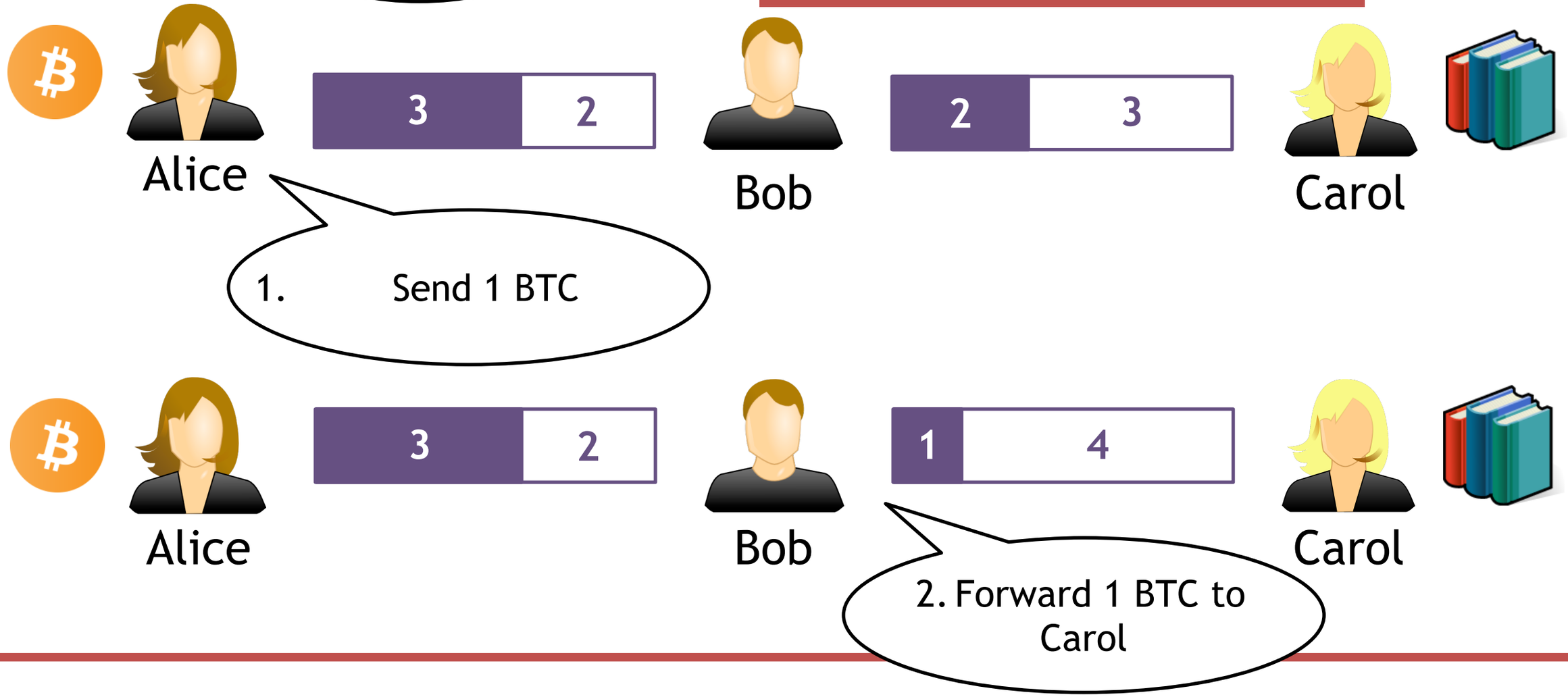


Payment Channel Networks (PCNs)

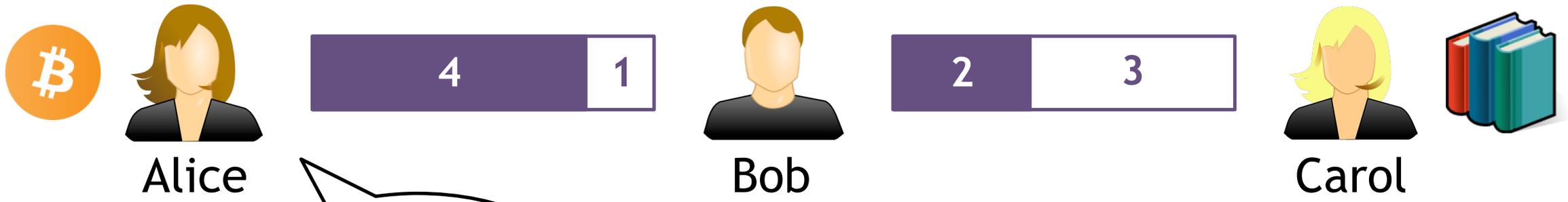
Payment Channel Networks (PCNs)



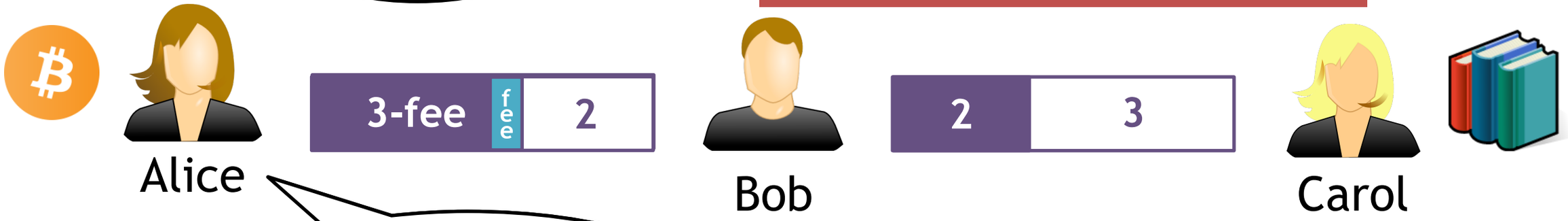
Should happen atomically

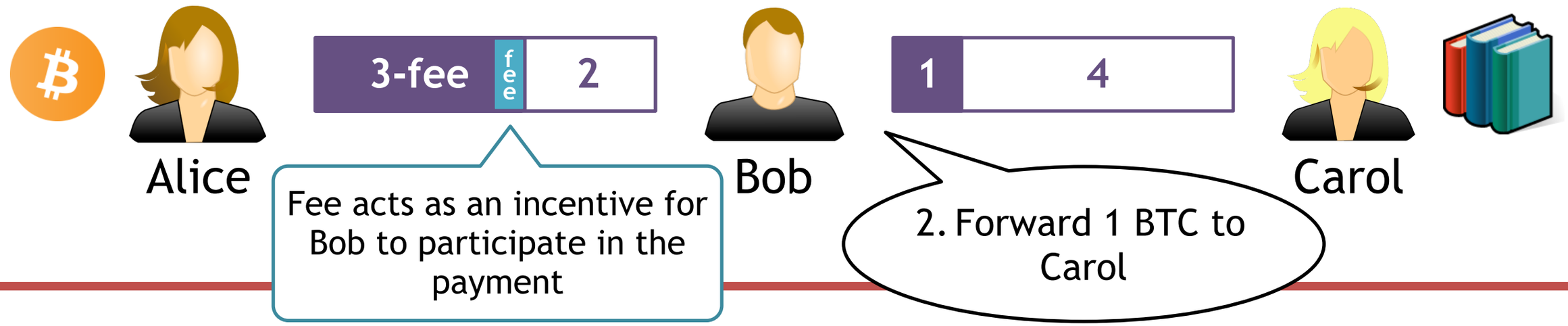


Payment Channel Networks (PCNs)

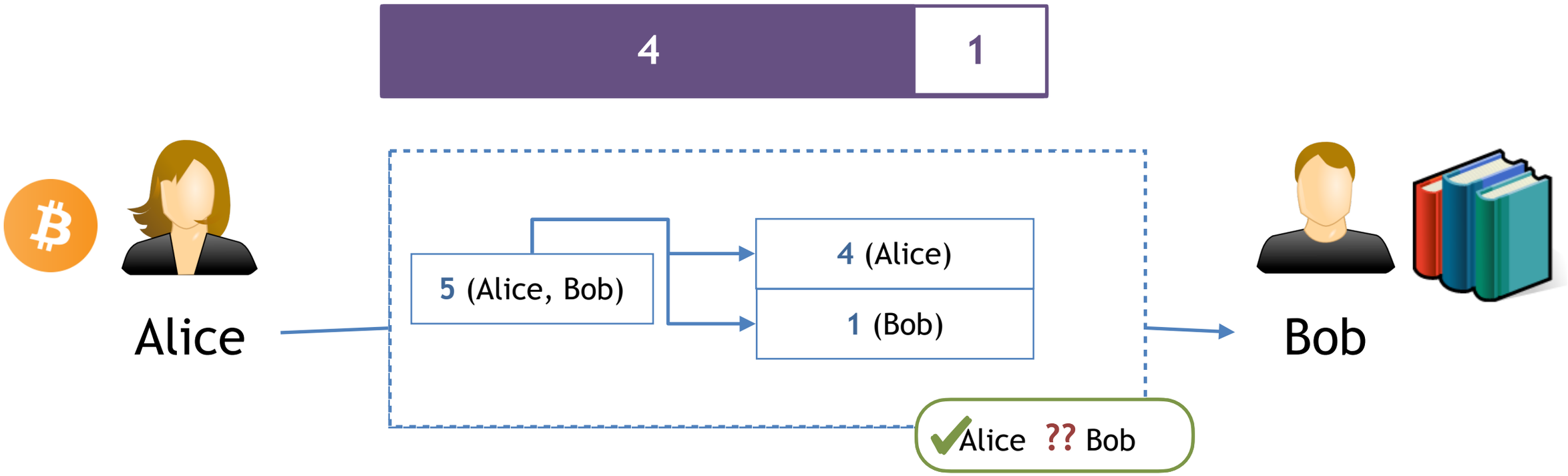


Should happen atomically

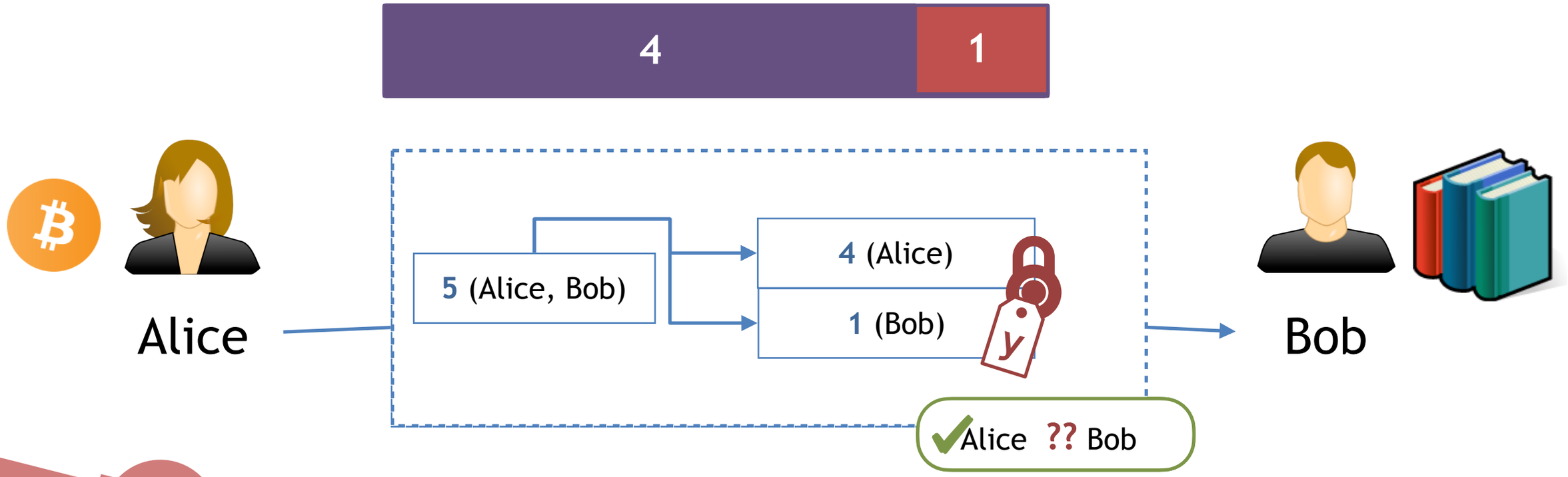


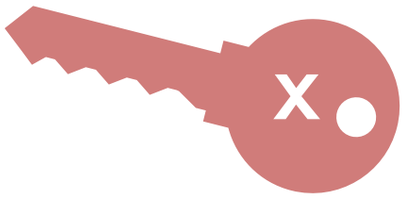


Hashtime Lock Contract (HTLC)



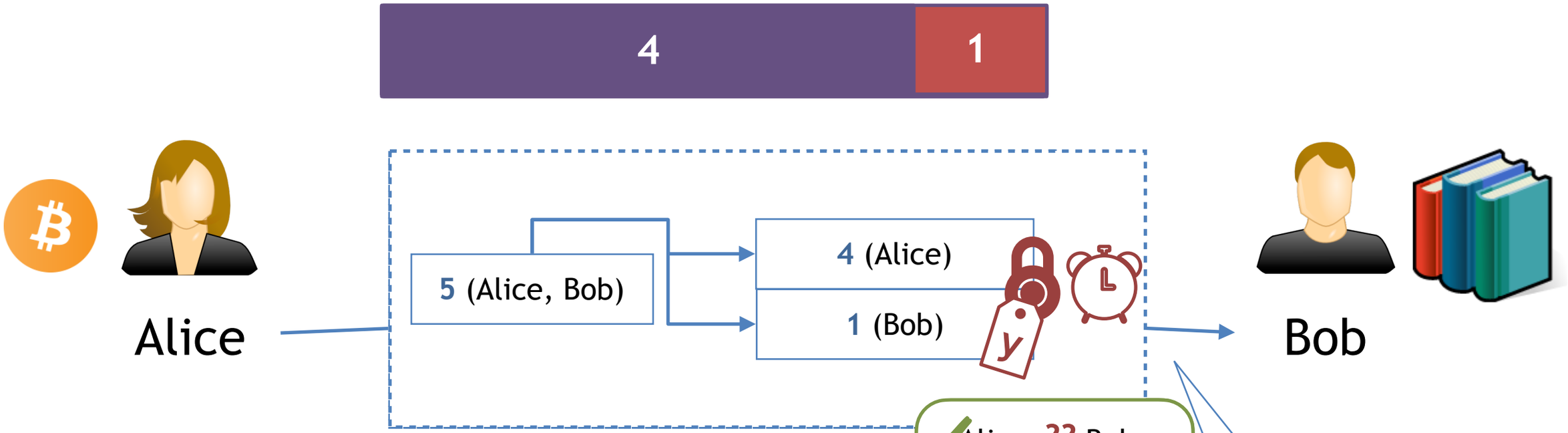
Hashtime Lock Contract (HTLC)

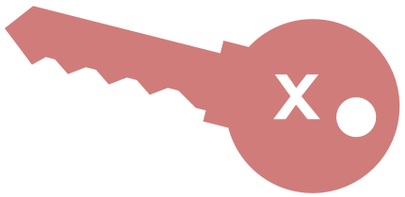




With knowledge of **x**, Bob can "open" + publish the transaction on the blockchain for enforcing the payment

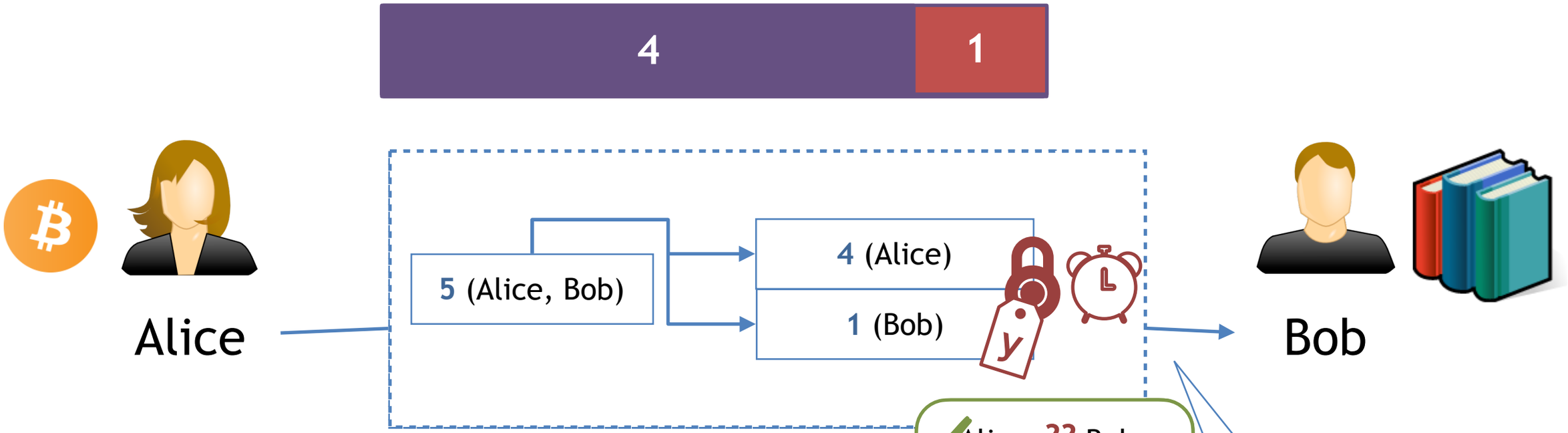
Hashtime Lock Contract (HTLC)





With knowledge of **x**, Bob can "open" + publish the transaction on the blockchain for enforcing the payment

Hashtime Lock Contract (HTLC)

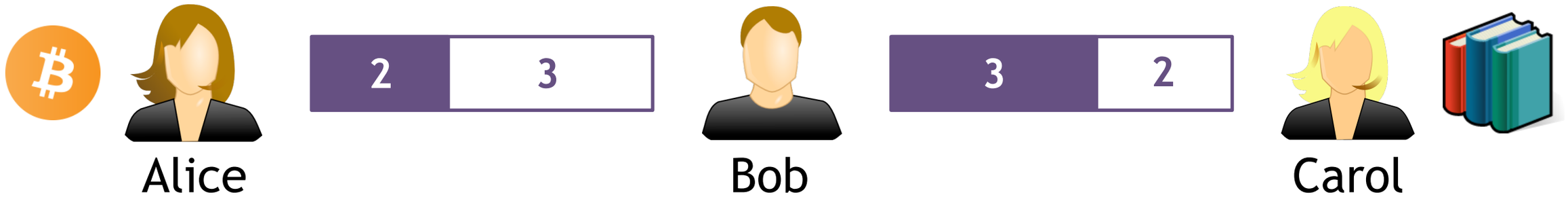


After time the transaction cannot be published anymore on the blockchain

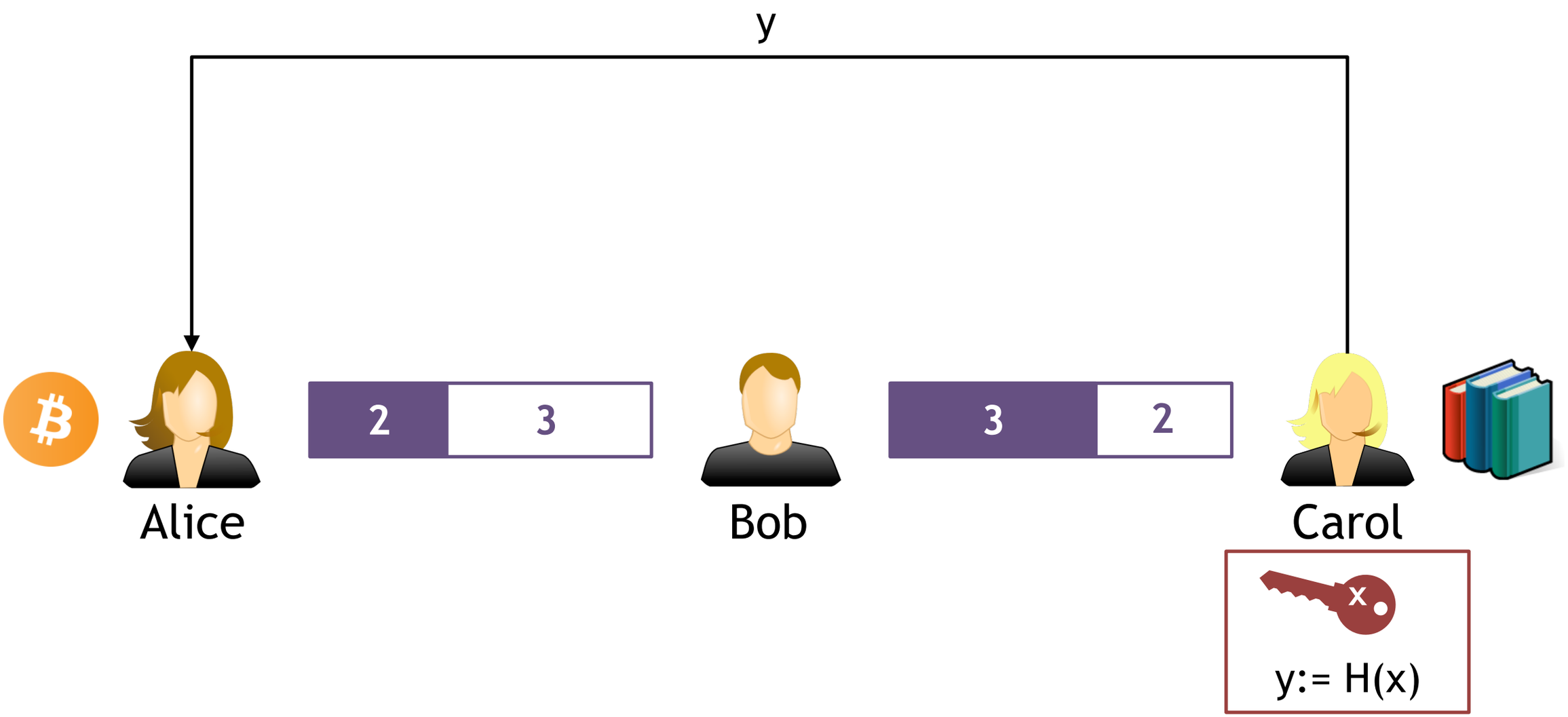
With knowledge of **x**, Bob can "open" + publish the transaction on the blockchain for enforcing the payment

HTLC (Alice, Bob, 1, y,):
Alice pays Bob 1 BTC iff Bob shows some x such that H(x) = **y** before

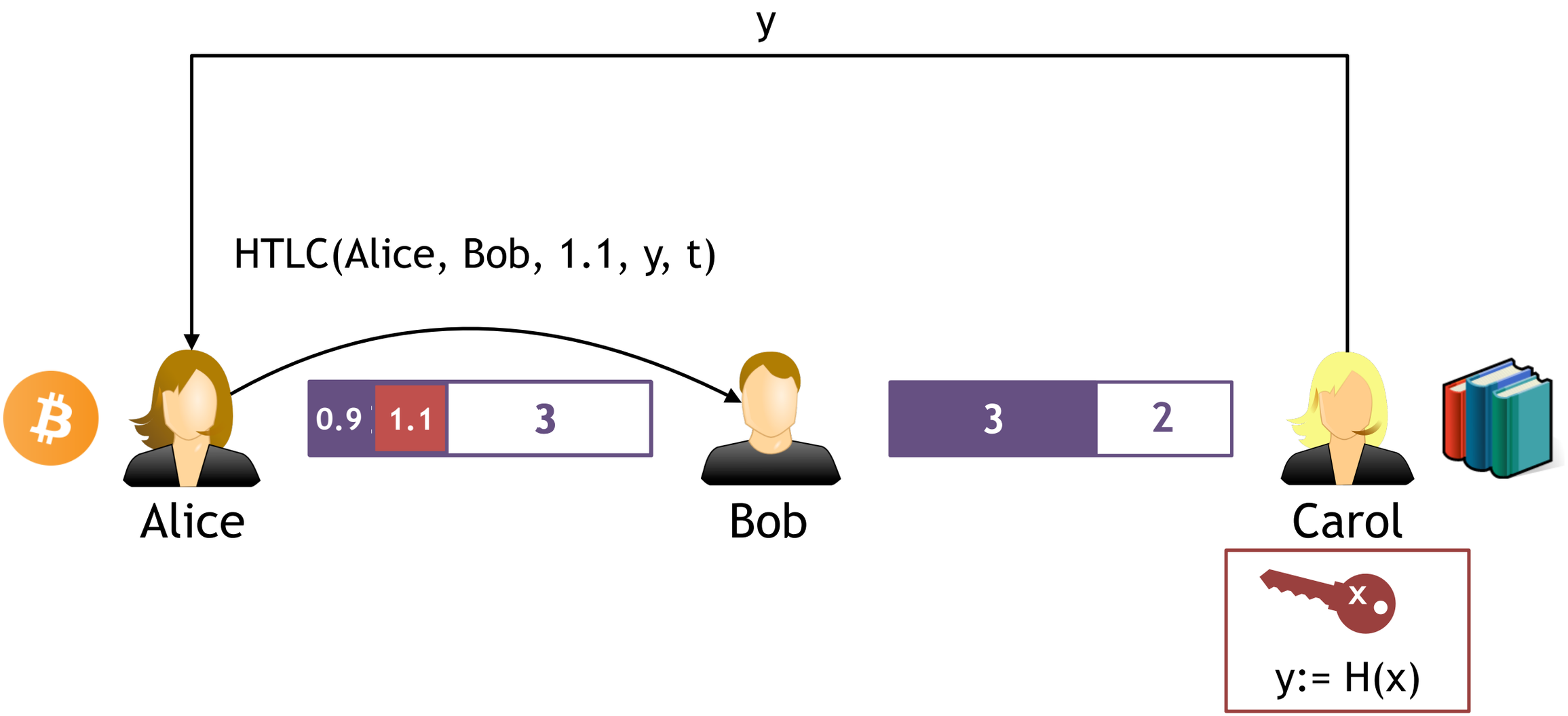
HTLC for Multi-hop Payments



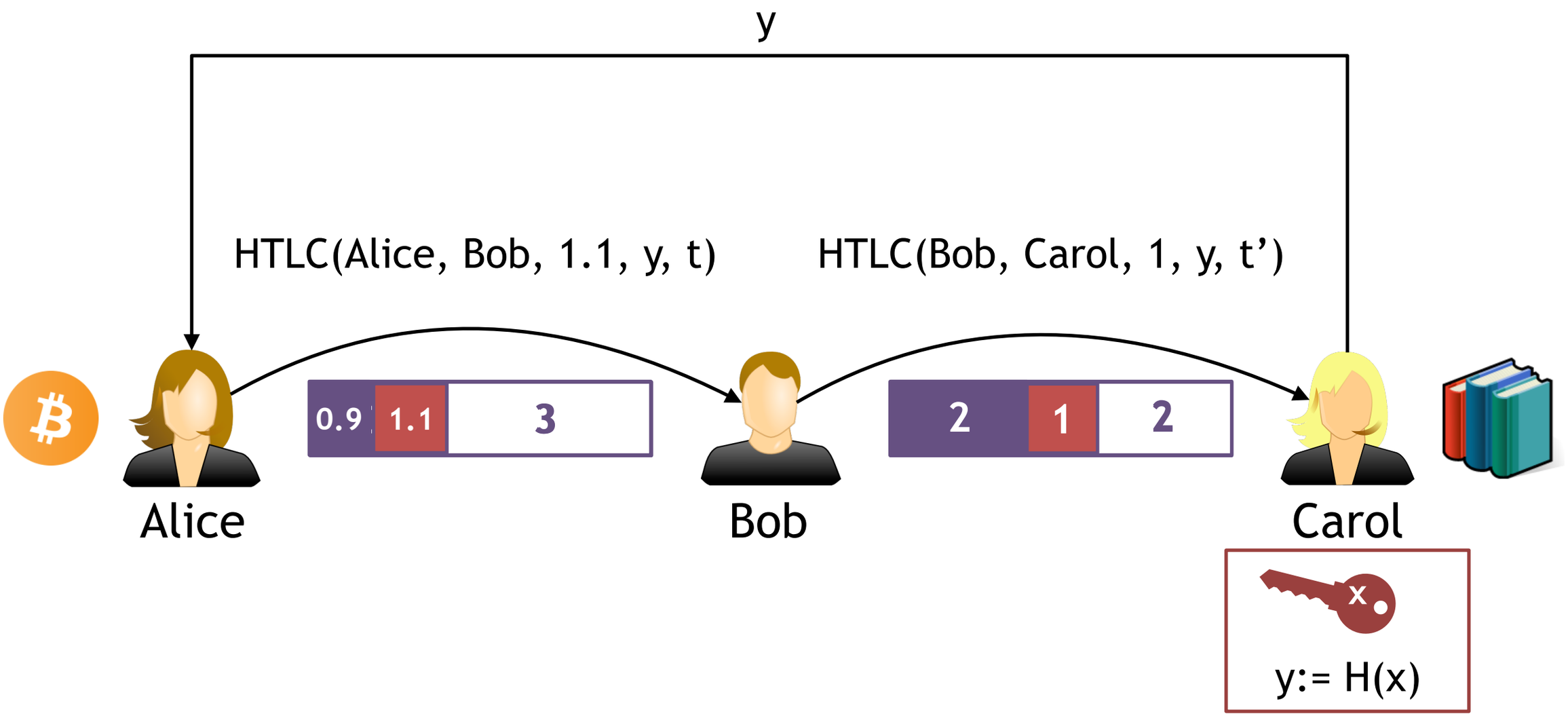
HTLC for Multi-hop Payments



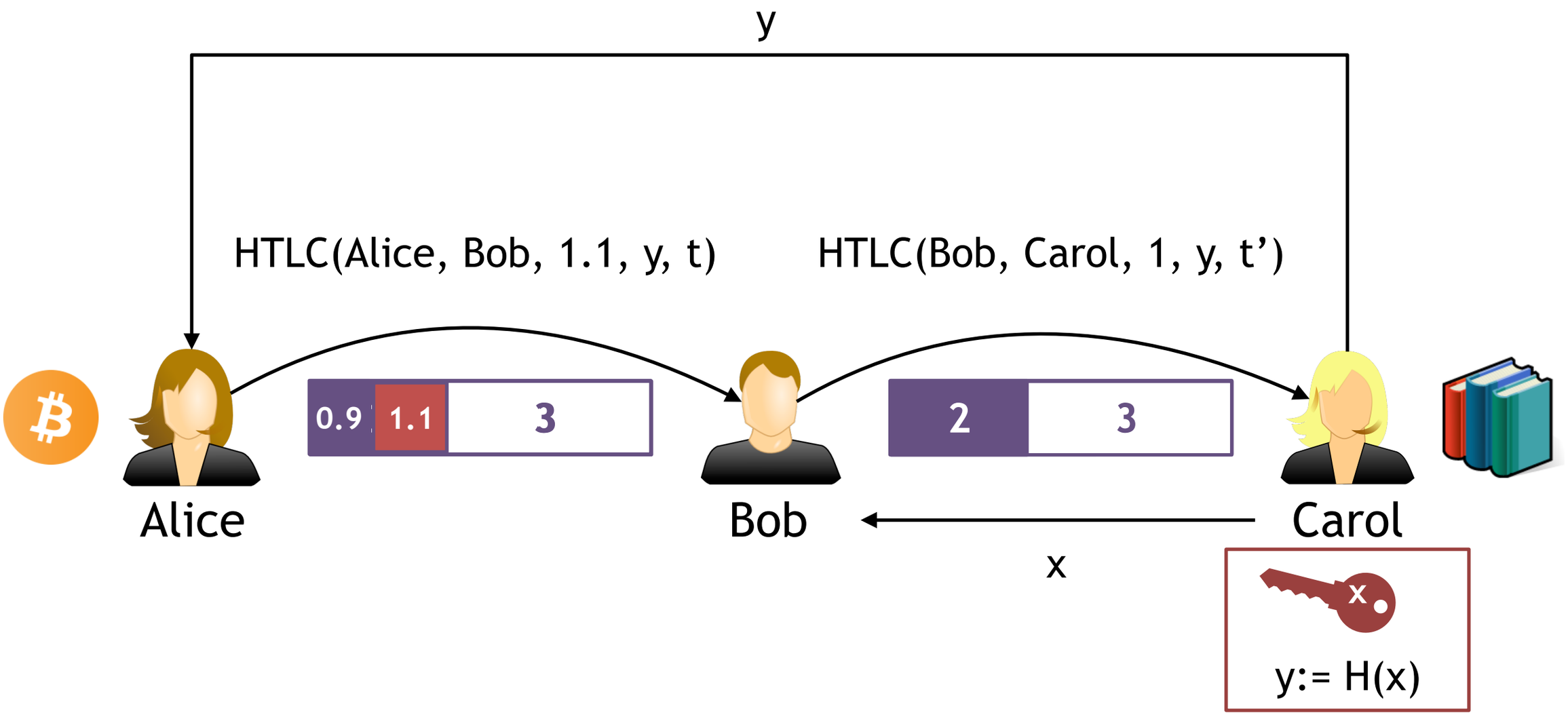
HTLC for Multi-hop Payments



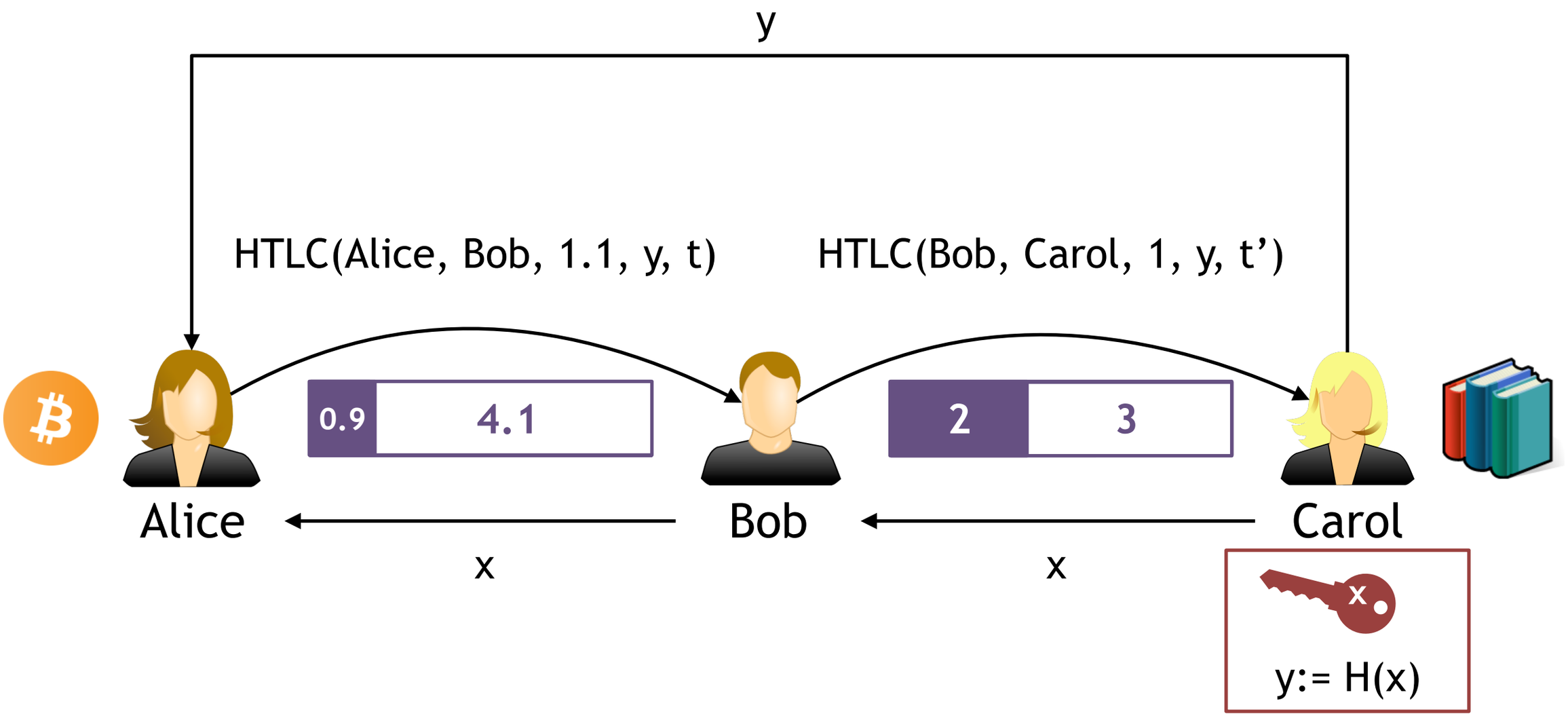
HTLC for Multi-hop Payments



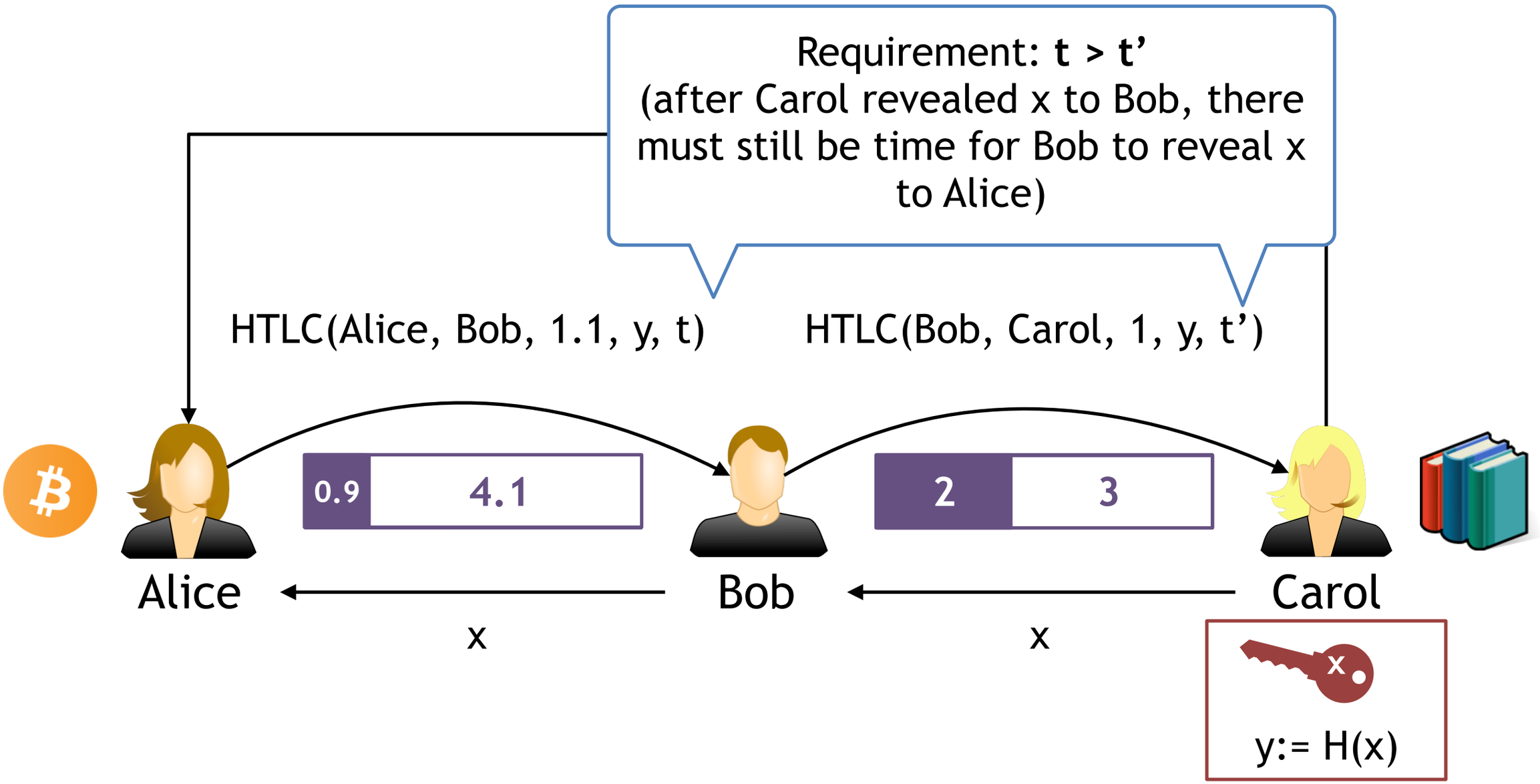
HTLC for Multi-hop Payments



HTLC for Multi-hop Payments



HTLC for Multi-hop Payments



Security and Privacy Issues in Existing PCNs

ACM CCS 2017

Concurrency and Privacy with Payment-Channel Networks*

Giulio Malavolta†
Friedrich-Alexander-University Erlangen-Nürnberg
malavolta@cs.fau.de

Pedro Moreno-Sanchez†
Purdue University
pmorenos@purdue.edu

Aniket Kate
Purdue University
aniket@purdue.edu

Matteo Maffei
TU Wien
matteo.maffei@tuwien.ac.at

Srivatsan Ravi
University of Southern California
srivatsr@usc.edu

Abstract

Permissionless blockchains protocols such as Bitcoin are inherently limited in transaction throughput and latency. Current efforts to address this key issue focus on off-chain payment channels that can be combined in a Payment-Channel Network (PCN) to enable an unlimited number of payments without requiring to access the blockchain other than to register the initial and final capacity of each channel. While this approach paves the way for low latency and high throughput of payments, its deployment in practice raises several privacy concerns as well as technical challenges related to the inherently concurrent nature of payments that have not been sufficiently studied so far.

In this work, we lay the foundations for privacy and concurrency in PCNs, presenting a formal definition in the Universal Composability framework as well as practical and provably secure solutions. In particular, we present Fulgor and Rayo. Fulgor is the first payment protocol for PCNs that provides provable privacy guarantees for PCNs and is fully compatible with the Bitcoin scripting system. However, Fulgor is a blocking protocol and therefore prone to deadlocks of concurrent payments as in currently available PCNs. Instead, Rayo is the first protocol for PCNs that enforces *non-blocking progress* (i.e., at least one of the concurrent payments terminates). We show through a new impossibility result that non-blocking

NDSS 2019

Anonymous Multi-Hop Locks for Blockchain Scalability and Interoperability

Giulio Malavolta*§, Pedro Moreno-Sanchez*¶†, Clara Schneidewind†, Aniket Kate‡, Matteo Maffei†
§Friedrich-Alexander-University Erlangen-Nürnberg, †TU Wien, ‡ Purdue University

***Abstract*—Tremendous growth in cryptocurrency usage is exposing the inherent scalability issues with permissionless blockchain technology. *Payment-channel networks* (PCNs) have emerged as the most widely deployed solution to mitigate the scalability issues, allowing the bulk of payments between two users to be carried out off-chain. Unfortunately, as reported in the literature and further demonstrated in this paper, current PCNs do not provide meaningful security and privacy guarantees [32], [42].**

In this work, we study and design secure and privacy-preserving PCNs. We start with a security analysis of existing PCNs, reporting a new attack that applies to all major PCNs, including the Lightning Network, and allows an attacker to steal the fees from honest intermediaries in the same payment path. We then formally define anonymous multi-hop locks (AMHLs), a novel cryptographic primitive that serves as a cornerstone for the design of secure and privacy-preserving PCNs. We present several provably secure cryptographic instantiations that make AMHLs compatible with the vast majority of cryptocurrencies. In particular, we show that (linear) homomorphic one-way functions suffice to construct AMHLs for PCNs supporting

I. Introduction

Cryptocurrencies are growing in popularity and are playing an increasing role in the worldwide financial ecosystem. In fact, the number of Bitcoin transactions grew by approximately 30% in 2017, reaching a peak of more than 420,000 transactions per day in December 2017 [2]. This striking increase in demand has given rise to scalability issues [20], which go well beyond the rapidly increasing size of the blockchain. For instance, the permissionless nature of the consensus algorithm used in Bitcoin today limits the transaction rate to tens of transactions per second, whereas other payment networks such as Visa support peaks of up to 47,000 transactions per second [9].

Among the various proposals to solve the scalability issue [22], [23], [40], [50], *payment-channels* have emerged as the most widely deployed solution in practice. In a nutshell, two users open a payment channel by committing a single transaction to the blockchain, which locks their bitcoins in a deposit secured by a

Expressiveness and Collateral in Payment-Channel Networks

ACM CCS 2019

Atomic Multi-Channel Updates with Constant Collateral in Bitcoin-Compatible Payment-Channel Networks

Christoph Egger
Friedrich-Alexander University Erlangen-Nuremberg

Pedro Moreno-Sanchez
TU Wien

Matteo Maffei
TU Wien

ABSTRACT

Current cryptocurrencies provide a heavily limited transaction throughput that is clearly insufficient to cater their growing adoption. Payment-channel networks (PCNs) have emerged as an interesting solution to the scalability issue and are currently deployed by popular cryptocurrencies such as Bitcoin and Ethereum. While PCNs do increase the transaction throughput by processing payments off-chain and using the blockchain only as a dispute arbitrator, they unfortunately require high collateral (i.e., they lock coins for a non-constant time along the payment path) and are restricted to payments in a path from sender to receiver. These issues have severe consequences in practice. The high collateral enables denial-of-service attacks that hamper the throughput and utility of the PCN. Moreover, the limited functionality hinders the applicability of current PCNs in many important application scenarios. Unfortunately, current proposals do not solve either of these issues, or they require Turing-complete language support, which severely limit their applicability.

1 INTRODUCTION

The permissionless nature of major cryptocurrencies such as Bitcoin [30] largely hinders their transaction throughput, limiting it to tens of transactions per second [11]. In contrast, other (centralized) payment networks such as Visa caters to a vast mass of users and payments by supporting a transaction throughput of up to tens of thousands of transactions per second [34]. Thus, permissionless cryptocurrencies suffer from a severe scalability issue preventing them from serving a growing base of payments.

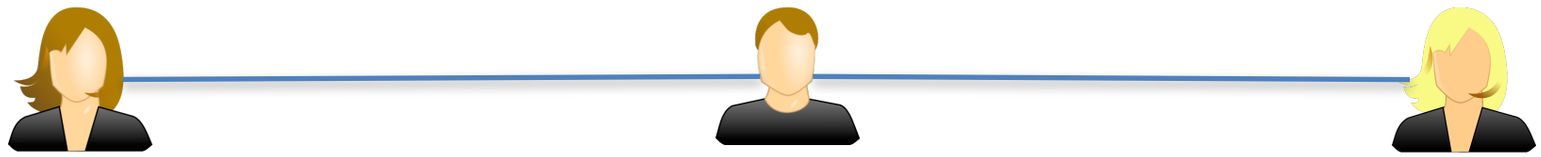
In this state of affairs, payment channels have emerged as an interesting mitigation technique for the scalability issue and is currently deployed in popular cryptocurrencies such as Bitcoin or Ethereum [12, 24, 31]. In a nutshell, payment channels aim at establishing a two-party ledger that two users can privately maintain without resorting to the blockchain for every payment and yet ensuring that they can claim their rightful funds in the blockchain at any given time. For that, users first create a deposit transaction that establishes on-chain the initial balances for their two-party ledger.

Open Challenges

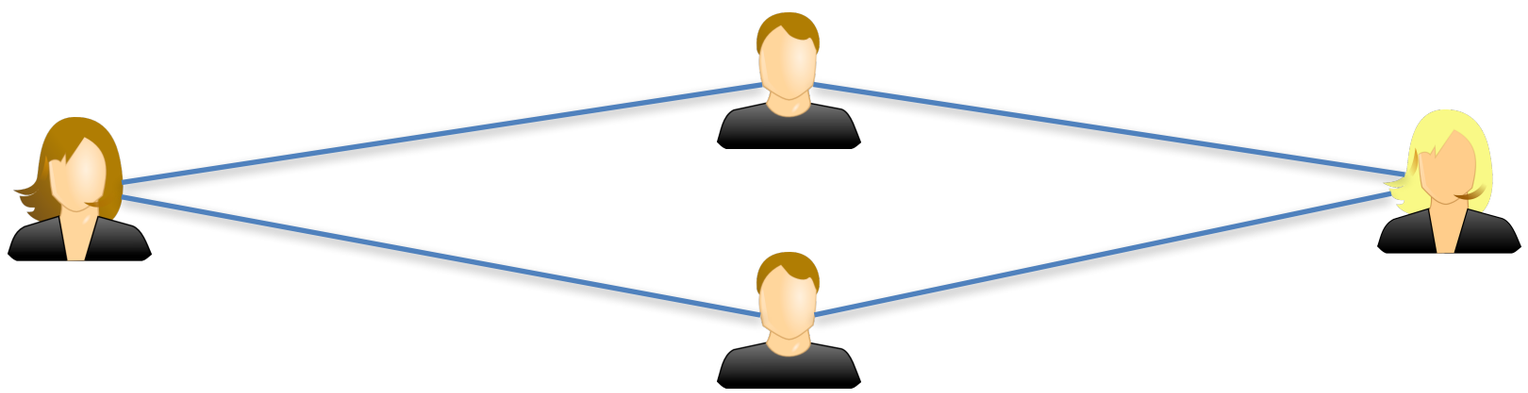
- In this work, we identify two open challenges:
  - Restricted expressiveness (and functionality)
    - Current Bitcoin-compatible PCNs restricted to single path-based payments
  - High collateral
    - A payment requires to put aside coins for a very long time

Improve Expressiveness Beyond Paths…

- So far we focused on single path-based payments



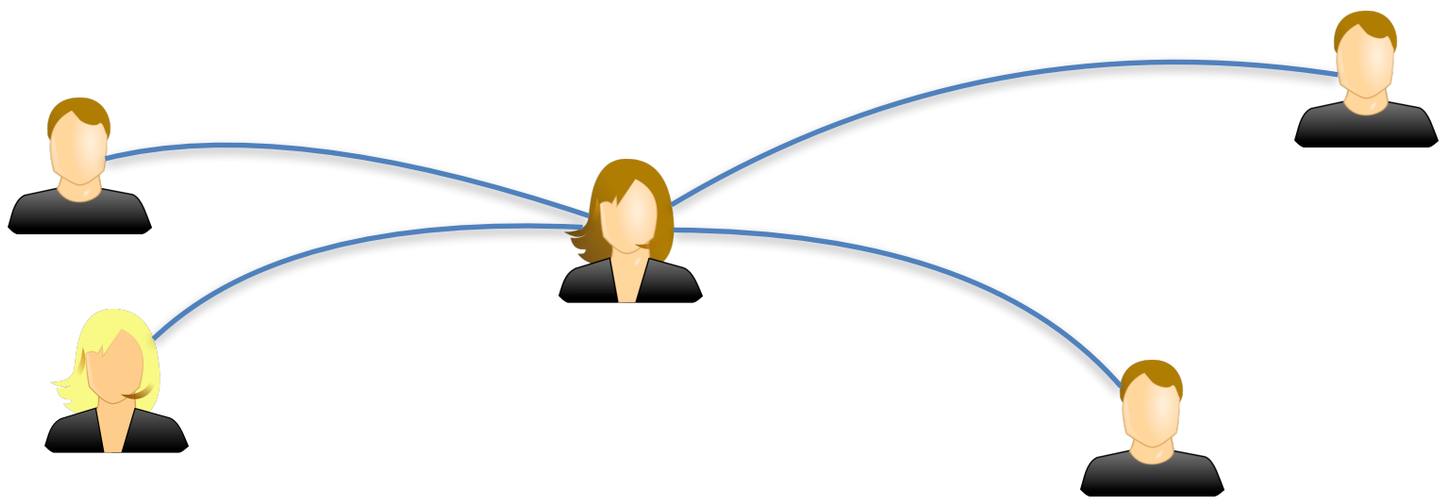
- Atomic Multi-Path (AMP)[1] payments: First step towards expressiveness



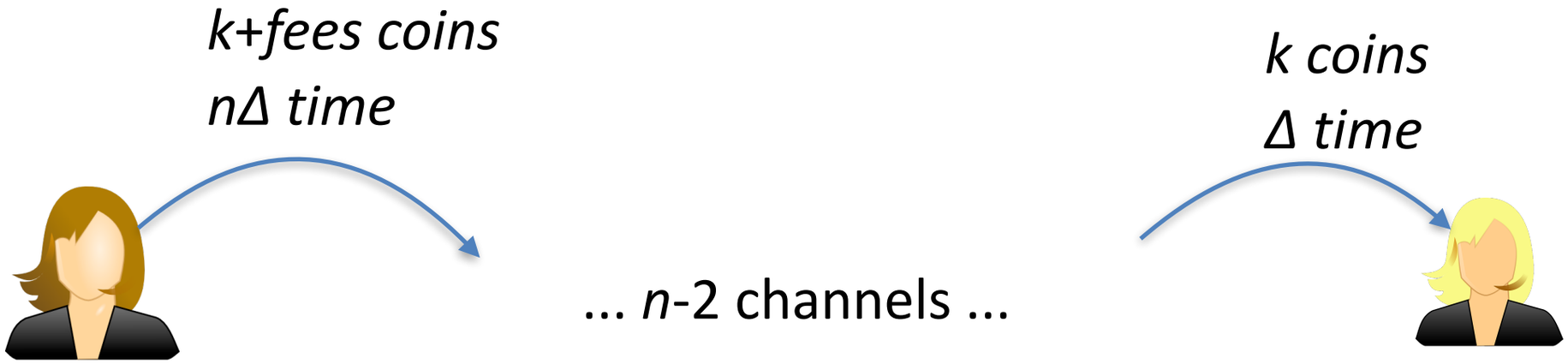
[1] https://lists.linuxfoundation.org/pipermail/lightning-dev/2018-February/000993.html

Our Goal: Full Expressiveness

- Support for arbitrary graph topology
- Enable new applications:
  - Crowd funding
  - Channel rebalancing
  - Netting
  - Your own application?

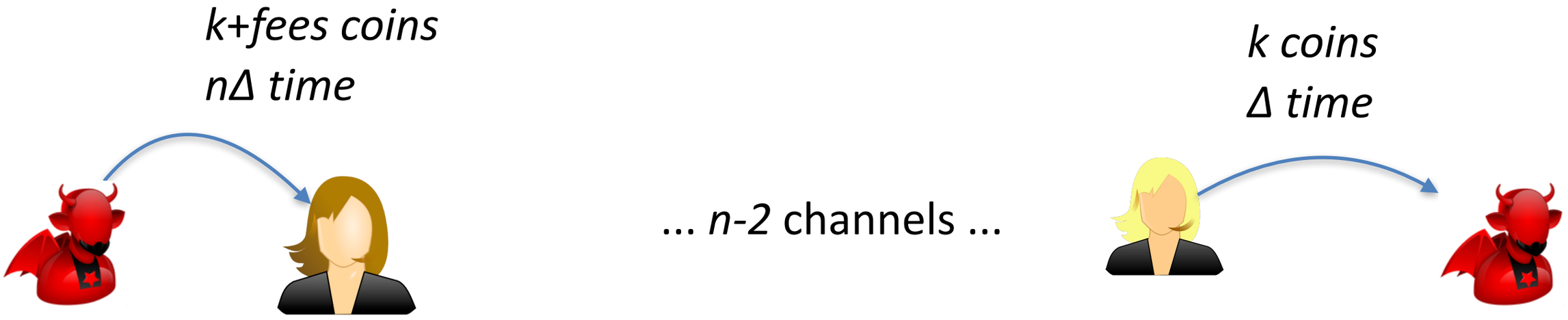


Collateral



- Each payment of *k* coins along an *n*-channel path requires to put aside at least *kn* coins
- Also, each user *i* has to lock her coins for a time Δ(*n*-*i*) where Δ is the time to safely close a channel
- Coins locked too long!

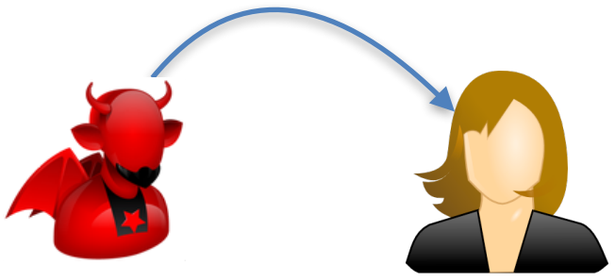
Griefing attack



- The adversary has a time amplification factor of *n*-1
- Δ is 1 day in the Lightning network!
- The attacker can use several paths

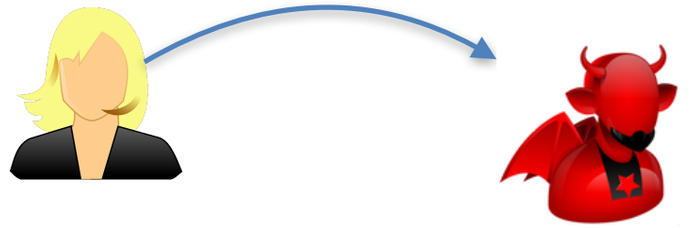
Our Goal: Constant Collateral

k+fees coins
nΔ time

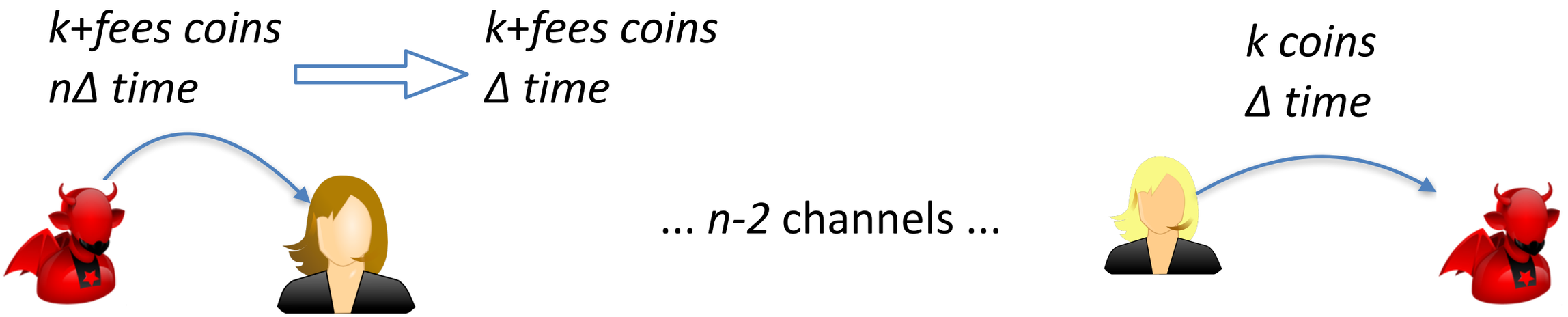


... *n-2* channels ...

k coins
Δ time

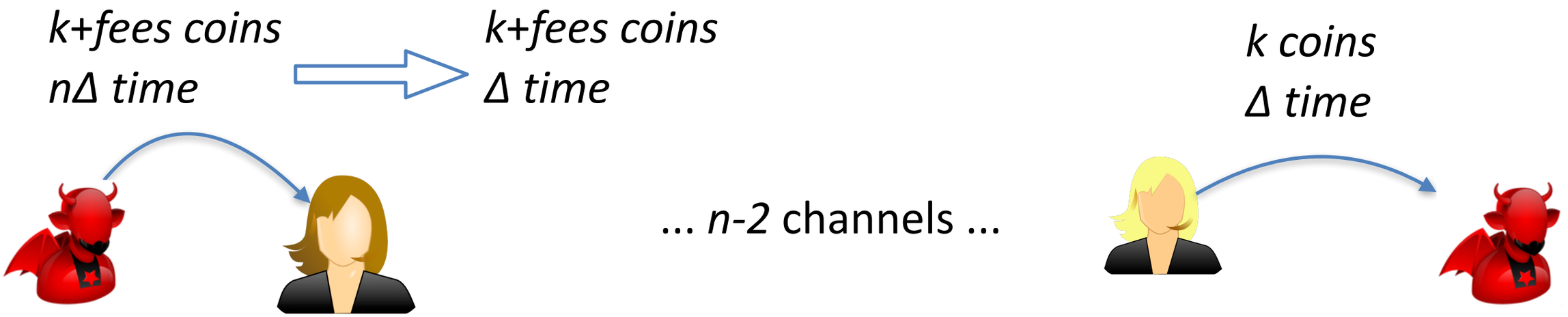


Our Goal: Constant Collateral



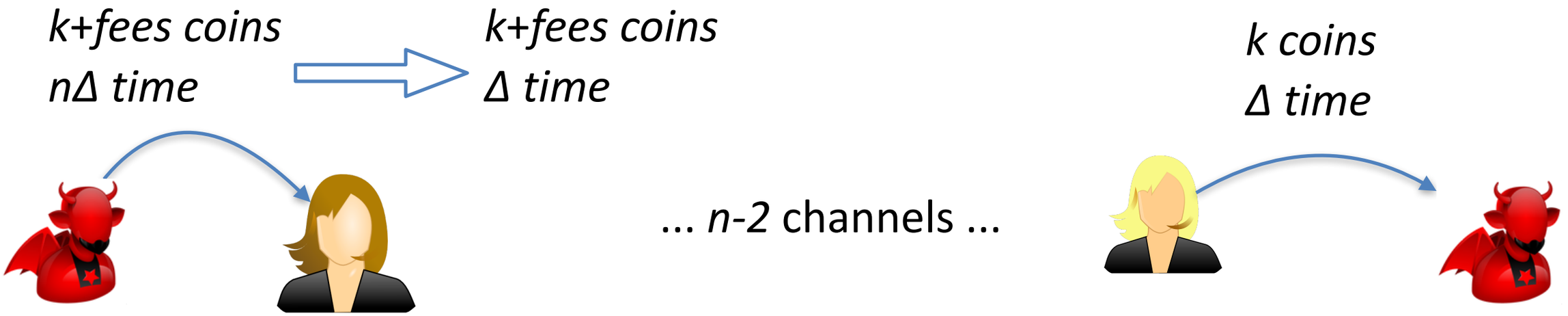
- **Constant collateral: Coins are locked only for Δ time, independently of the number of channels**

Our Goal: Constant Collateral



- Constant collateral: Coins are locked only for Δ time, independently of the number of channels
- Reduces the amplification factor

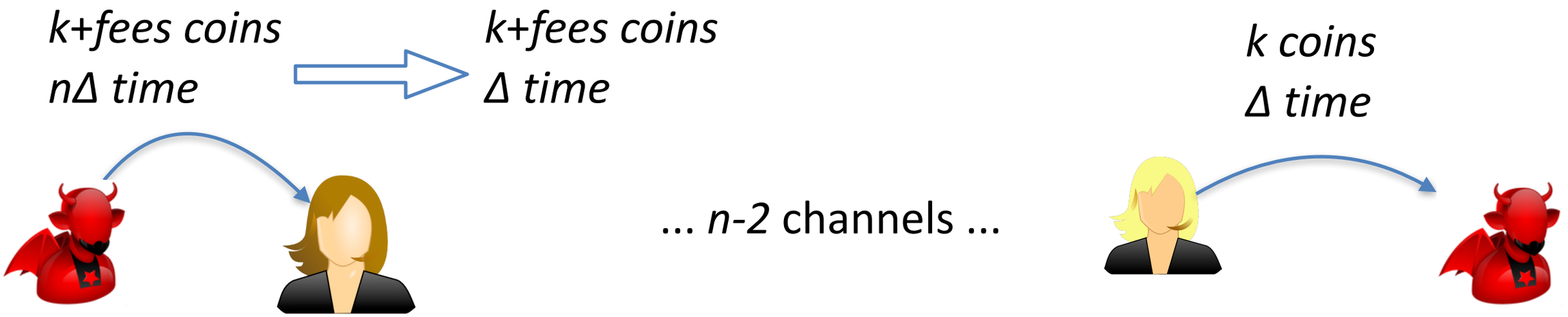
Our Goal: Constant Collateral



- Constant collateral: Coins are locked only for Δ time, independently of the number of channels
- Reduces the amplification factor
- Feasible in Ethereum-based PCNs: Sprites[1]

[1] A. Miller et al. Sprites and State Channels: Payment Networks that Go Faster than Lightning.

Our Goal: Constant Collateral

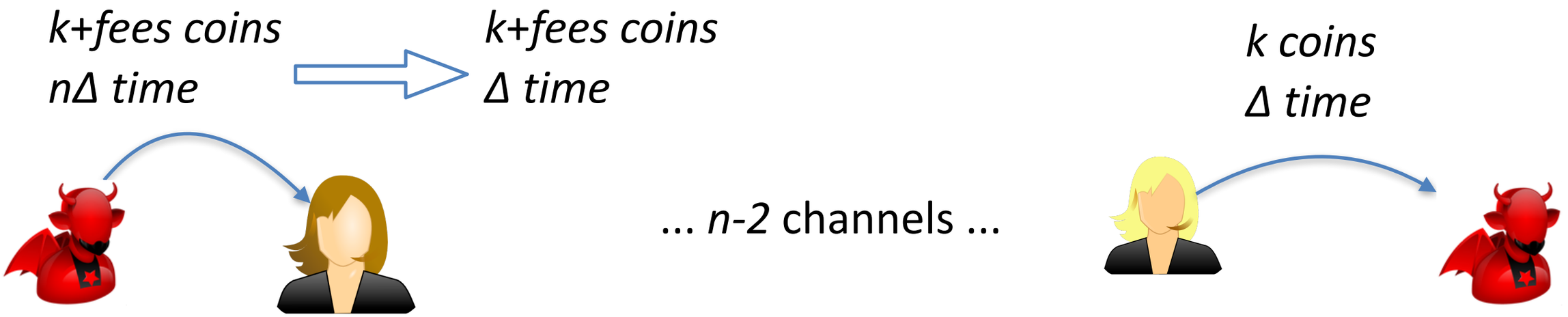


- Constant collateral: Coins are locked only for Δ time, independently of the number of channels
- Reduces the amplification factor
- Feasible in Ethereum-based PCNs: Sprites[1]

> *a) Feasibility of constant locktimes in Bitcoin:* Our constant locktimes construction relies on a global contract mechanism, which is easily expressed in Ethereum, but cannot (we conjecture) be emulated in Bitcoin without some modification to its scripting system. Are there minimal modifications to Bitcoin script that would enable constant locktimes?

[1] A. Miller et al. Sprites and State Channels: Payment Networks that Go Faster than Lightning.

Our Goal: Constant Collateral



- Constant collateral: Coins are locked only for Δ time, independently of the number of channels
- Reduces the amplification factor
- Feasible in Ethereum-based PCNs: Sprites[1]

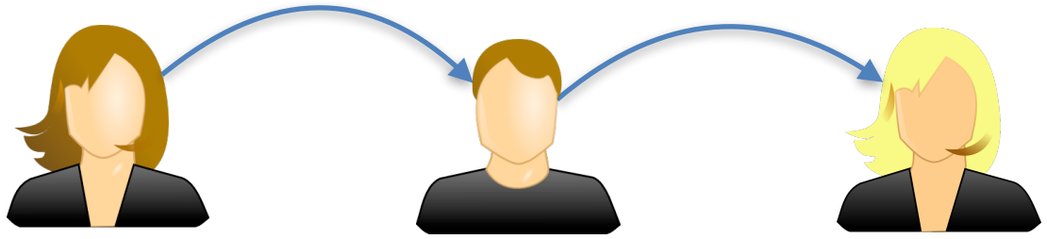
> *a) Feasibility of constant locktimes in Bitcoin:* Our constant locktimes construction relies on a global contract mechanism, which is easily expressed in Ethereum, but cannot (we conjecture) be emulated in Bitcoin without some modification to its scripting system. Are there minimal modifications to Bitcoin script that would enable constant locktimes?

AMCU: Constant collateral and backwards compatible with Bitcoin script

[1] A. Miller et al. Sprites and State Channels: Payment Networks that Go Faster than Lightning.

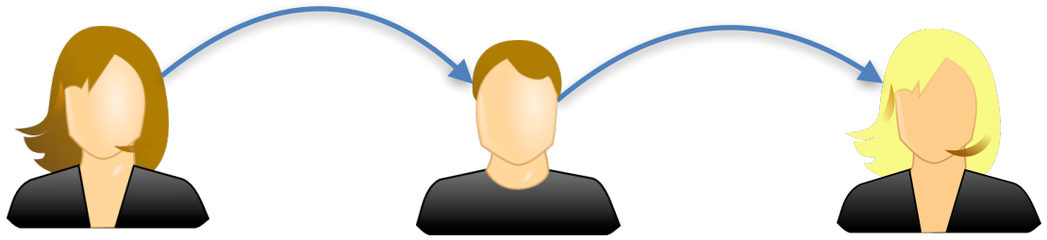
Atomic Multi-Channel Updates (ACMU)

8 (out of 10) 7 (out of 30)

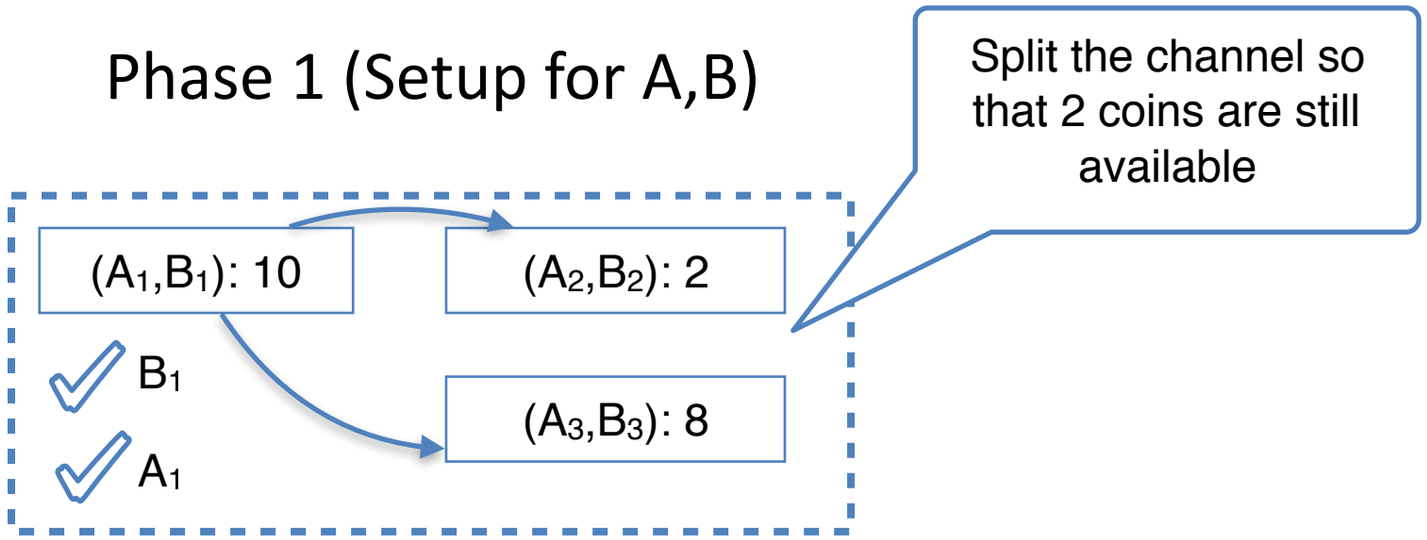


Atomic Multi-Channel Updates (ACMU)

8 (out of 10) 7 (out of 30)

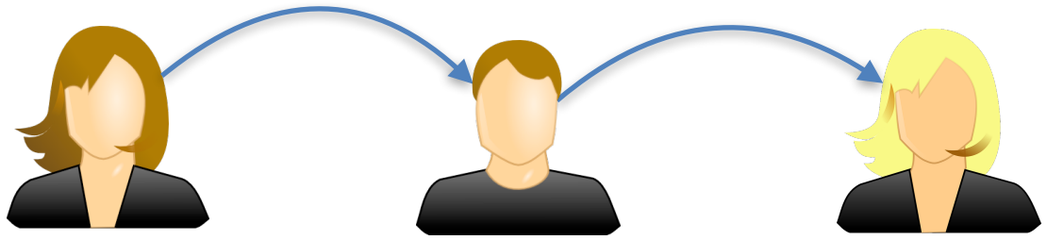


Phase 1 (Setup for A,B)

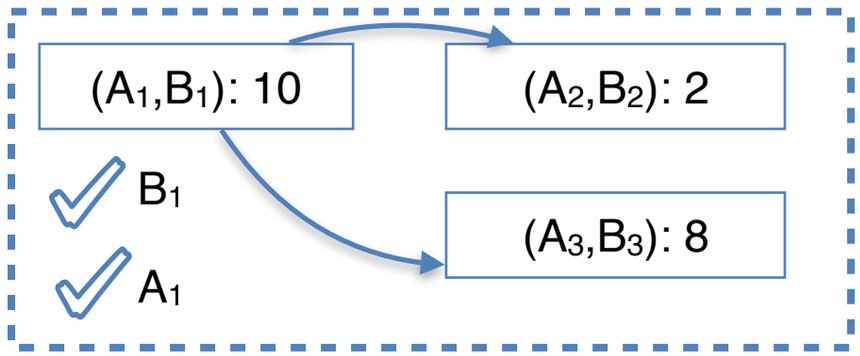


Atomic Multi-Channel Updates (ACMU)

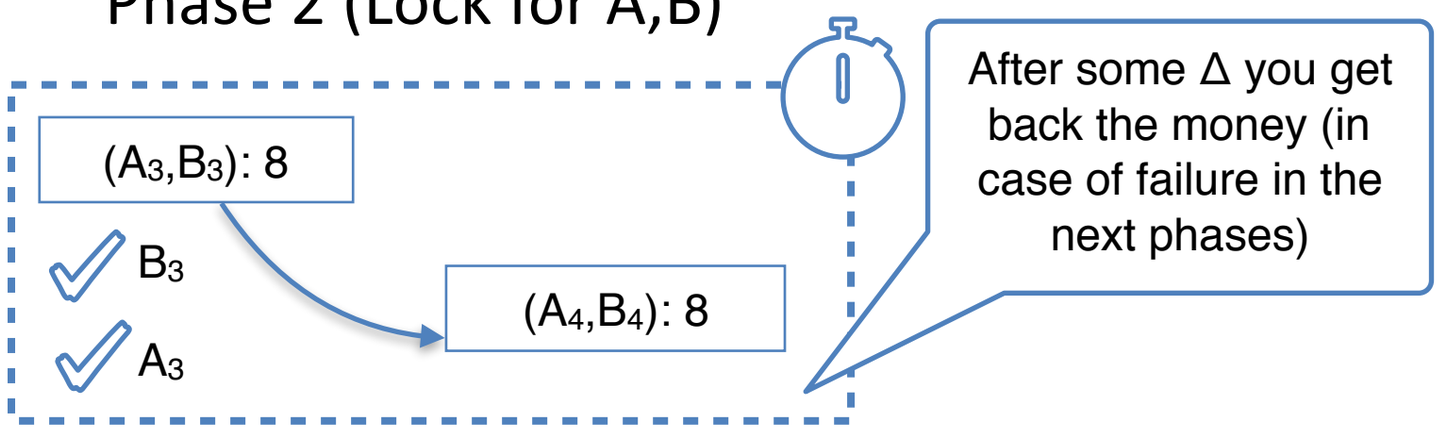
8 (out of 10) 7 (out of 30)



Phase 1 (Setup for A,B)

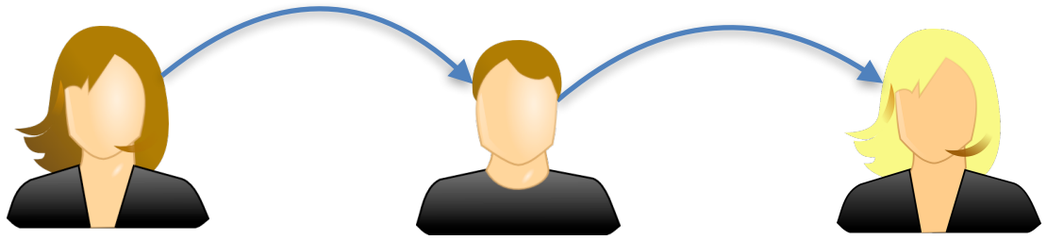


Phase 2 (Lock for A,B)

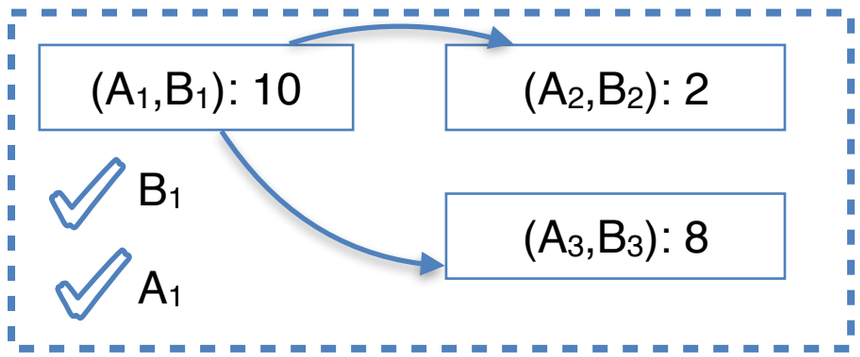


Atomic Multi-Channel Updates (ACMU)

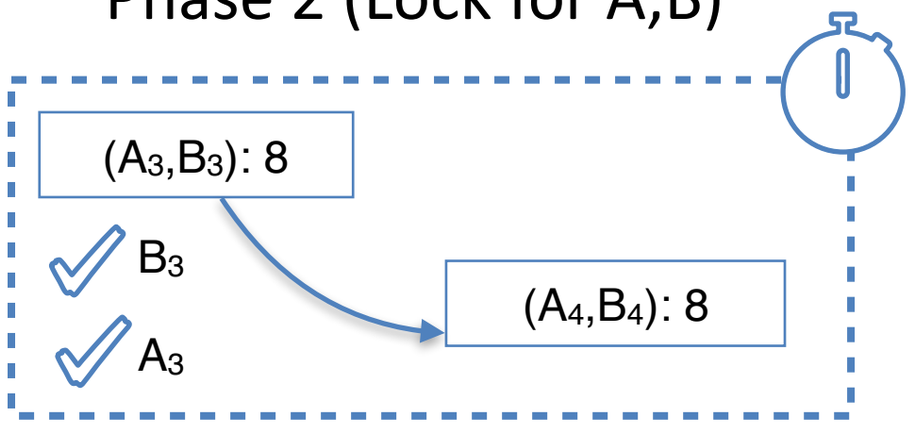
8 (out of 10) 7 (out of 30)



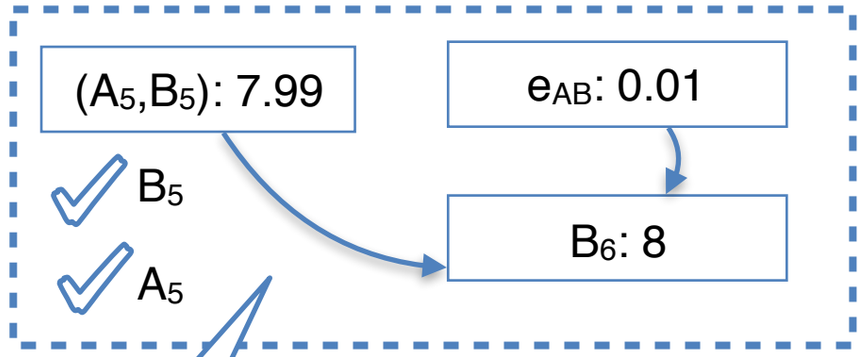
Phase 1 (Setup for A,B)



Phase 2 (Lock for A,B)



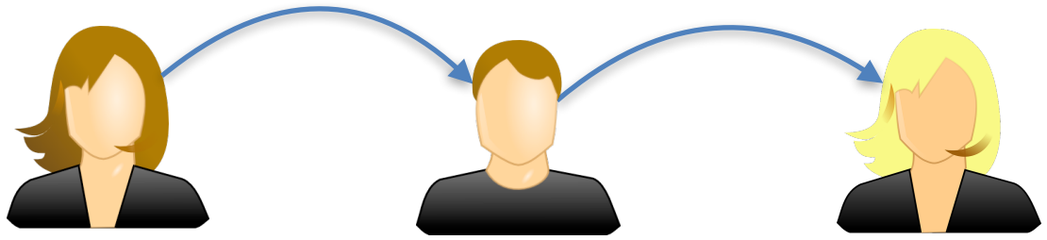
Phase 3 (Consume for A,B)



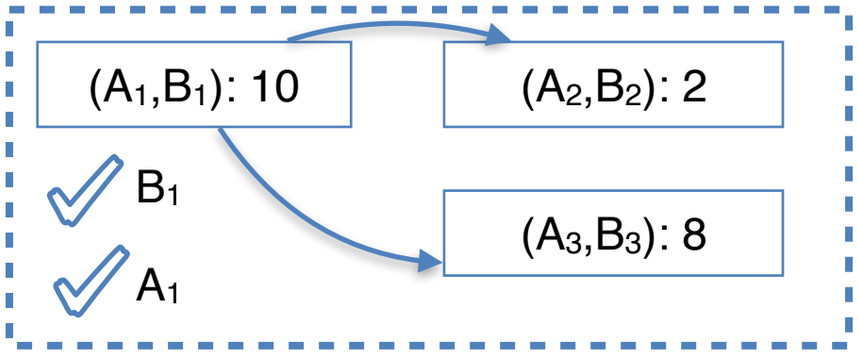
To spend you need money in a fresh account, which does not have money yet, key towards atomicity

Atomic Multi-Channel Updates (ACMU)

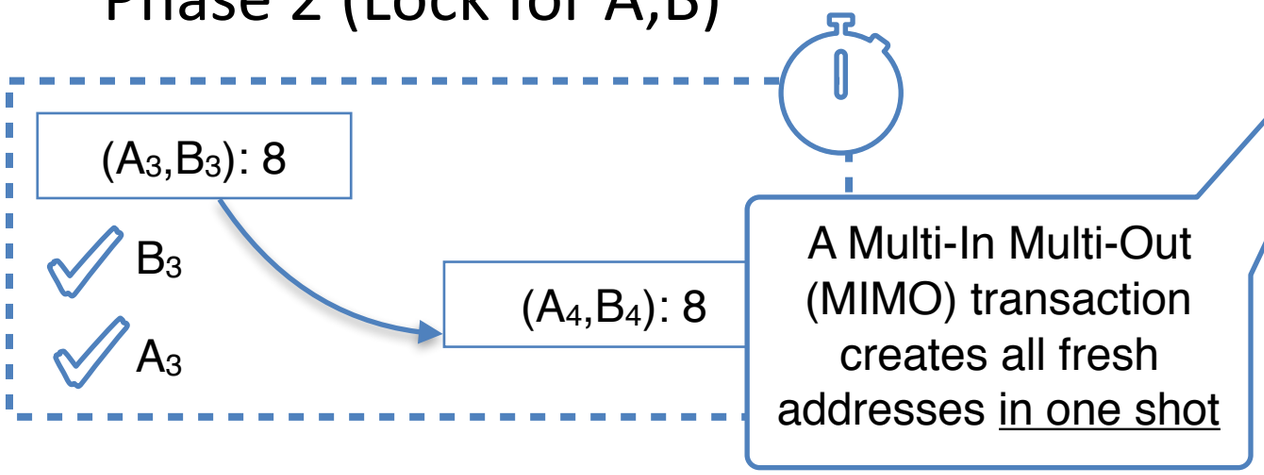
8 (out of 10) 7 (out of 30)



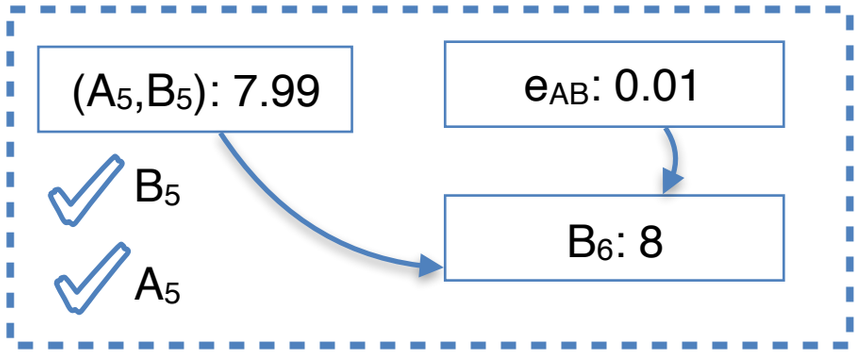
Phase 1 (Setup for A,B)



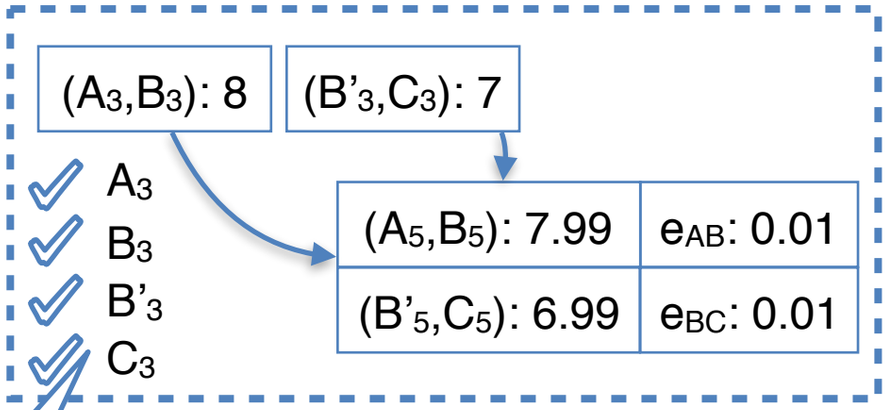
Phase 2 (Lock for A,B)



Phase 3 (Consume for A,B)

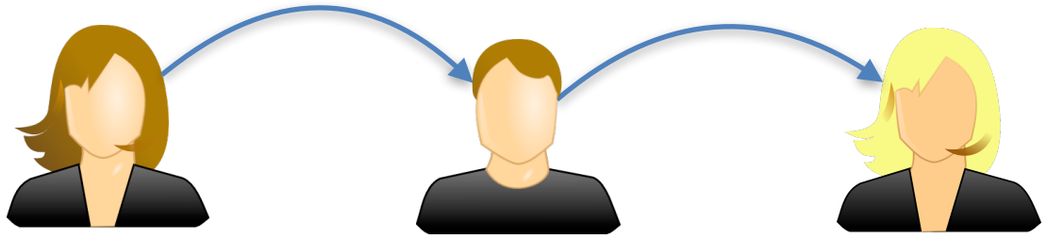


Phase 4 (Enable)

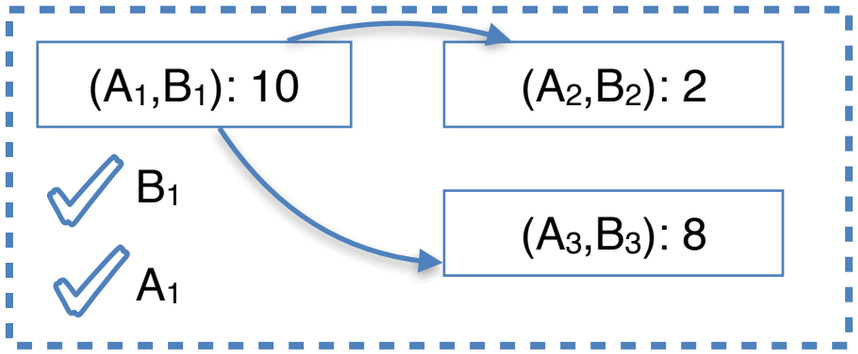


Atomic Multi-Channel Updates (ACMU)

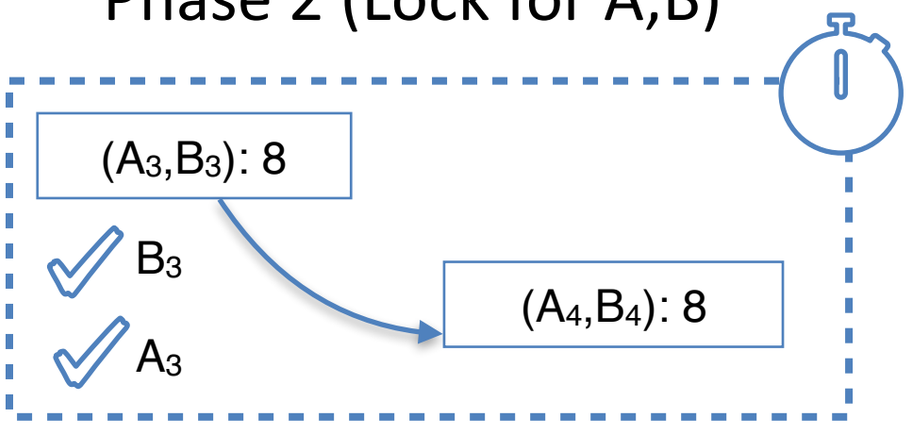
8 (out of 10) 7 (out of 30)



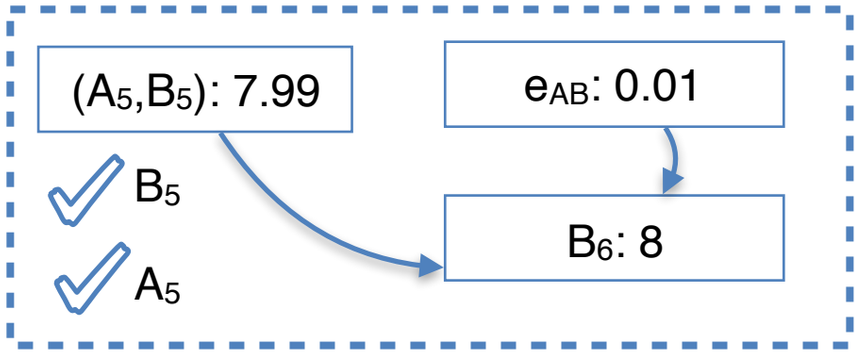
Phase 1 (Setup for A,B)



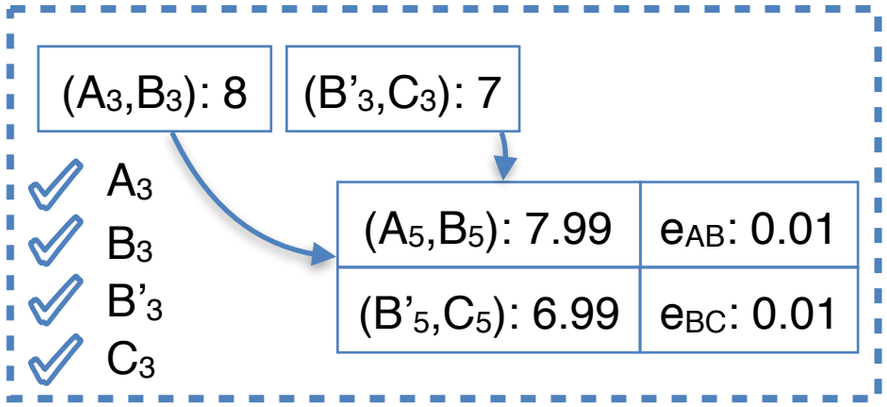
Phase 2 (Lock for A,B)



Phase 3 (Consume for A,B)



Phase 4 (Enable)

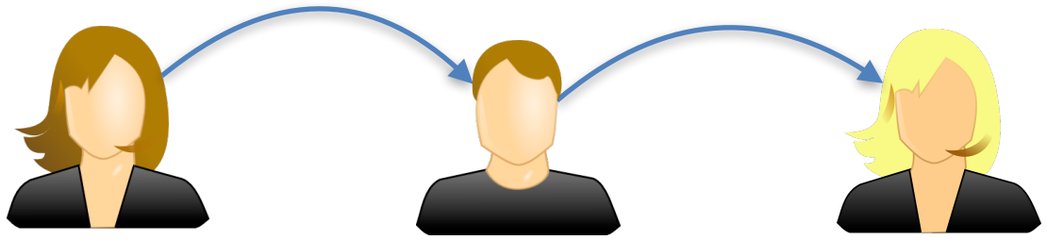


Setup —> Enable —> Consume

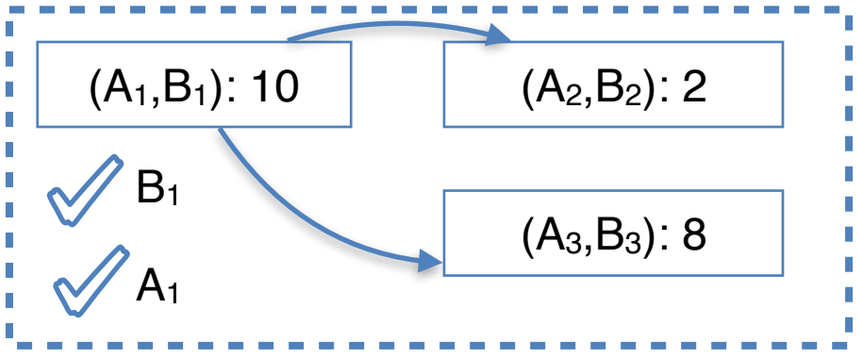
Setup —> Lock

Atomic Multi-Channel Updates (ACMU)

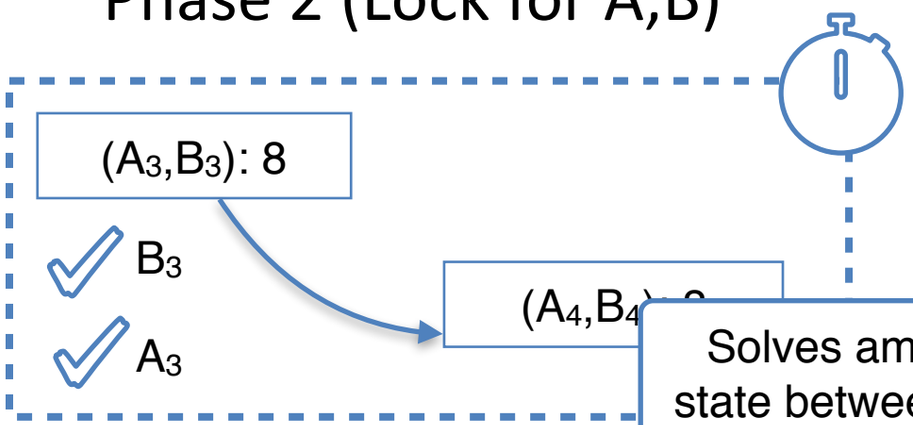
8 (out of 10) 7 (out of 30)



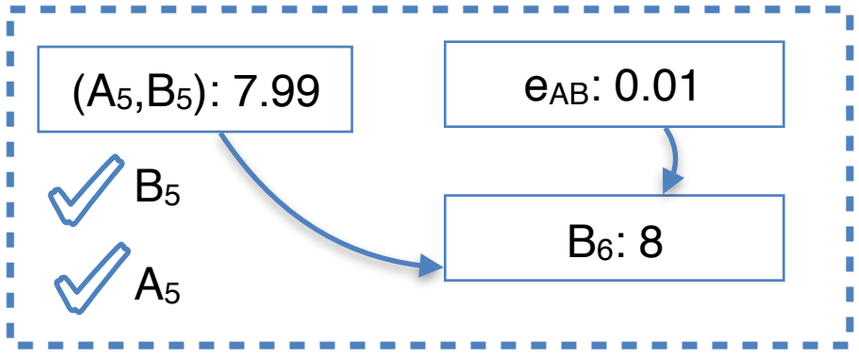
Phase 1 (Setup for A,B)



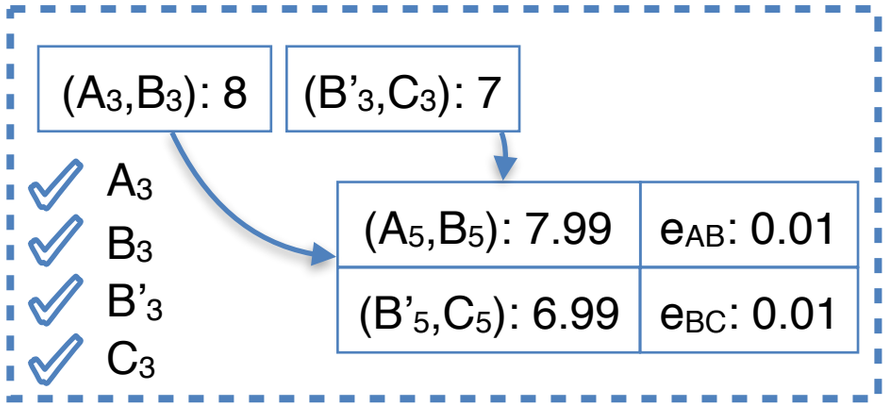
Phase 2 (Lock for A,B)



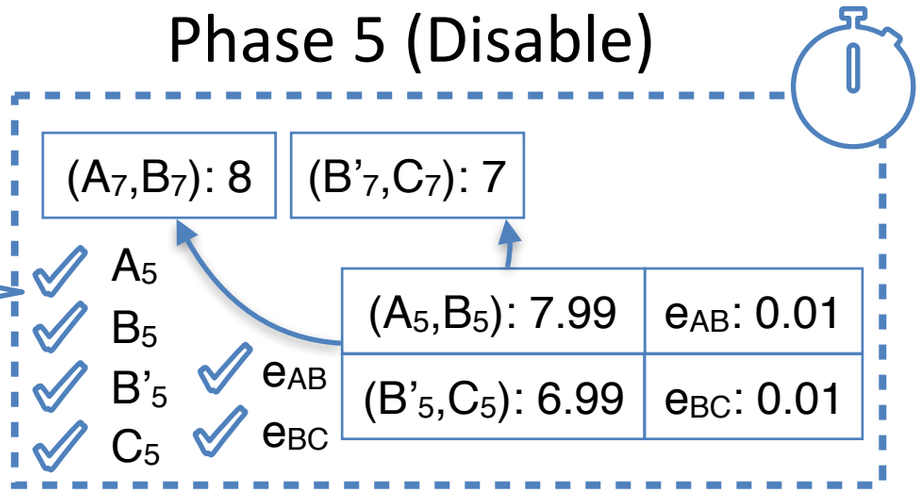
Phase 3 (Consume for A,B)



Phase 4 (Enable)



Phase 5 (Disable)



Security and Privacy Analysis

- AMCU achieves atomicity. In particular:
  - If the coins at one channel are ready to be sent to expected receiver, then all channels ready to forward payment
  - Otherwise, coins remain at a channel owned by original owners
- AMCU does not achieve relationship anonymity
  - Every user in the path collaborates with each other
- Instead,
  - Constant collateral (coins locked constant time)
  - Backwards compatible with current Bitcoin scripting language
  - Accountability: Possible to show a proof of misbehavior

Take Home...

- We can reduce the collateral to a constant and synchronize multiple transaction atomically
- Backwards compatible with Bitcoin script

- Formally specified and proven secure in the UC Framework
- Advantages:
  - Makes the collateral constant
  - Enables new classes of off-chain applications (e.g., crowd funding, channel rebalancing and more?)

- To be presented at ACM CCS 2019
- Paper available at https://eprint.iacr.org/2019/583.pdf

Take Home...

- We can reduce the collateral to a constant and synchronize multiple transaction atomically
- Backwards compatible with Bitcoin script

- Formally specified and proven secure in the UC Framework
- Advantages:
  - Makes the collateral constant
  - Enables new classes of off-chain applications (e.g., crowd funding, channel rebalancing and more?)

- To be presented at ACM CCS 2019
- Paper available at https://eprint.iacr.org/2019/583.pdf

THANKS!

@pedrorechez