COMP4137 Blockchain Technology and Applications
COMP7200 Blockchain Technology

Lecturer: Dr. Hong-Ning Dai (Henry)

Lecture 10

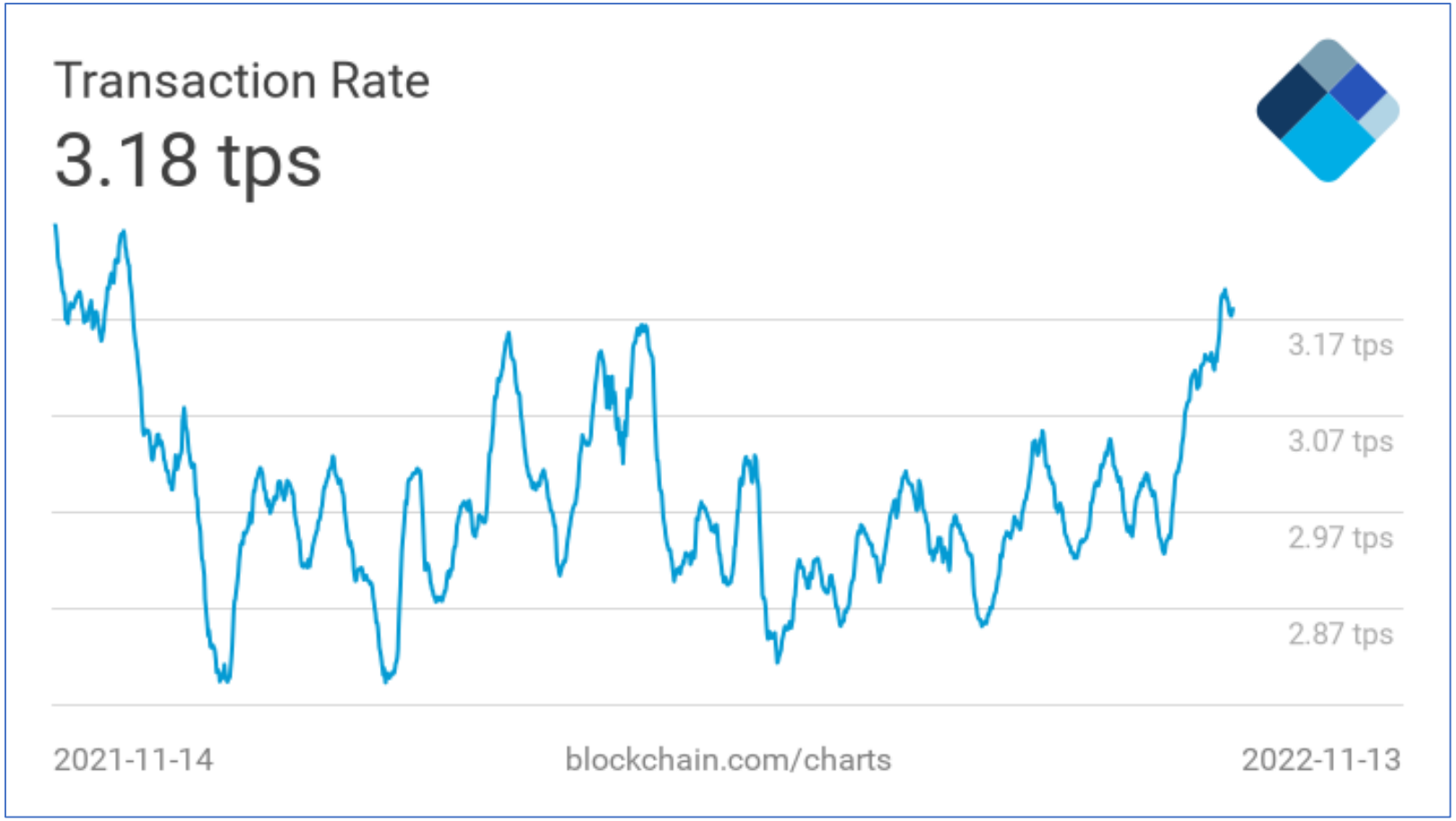
Advanced Topics

Outline

- Scaling blockchain by payment channel

- Blockchain applications

Bitcoin Tx per second



≈4200 Tx/block
1 block / 10 mins

⇒ max: 7 Tx/sec

Ethereum Tx per second

Ethereum avg Tx per second:

≈ 15 Tx/sec

Simple Tx: 21k Gas
max 30M Gas per block
⇒ max 1428 tx/block

1 Block/12s
⇒ max 119 tx/s

In comparison …

- Visa: up to 24,000 Tx/sec (regularly 2,000 Tx/sec)
- PayPal: 200 Tx/sec

- Ethereum: 15 Tx/sec
- Bitcoin: 7 Tx/sec

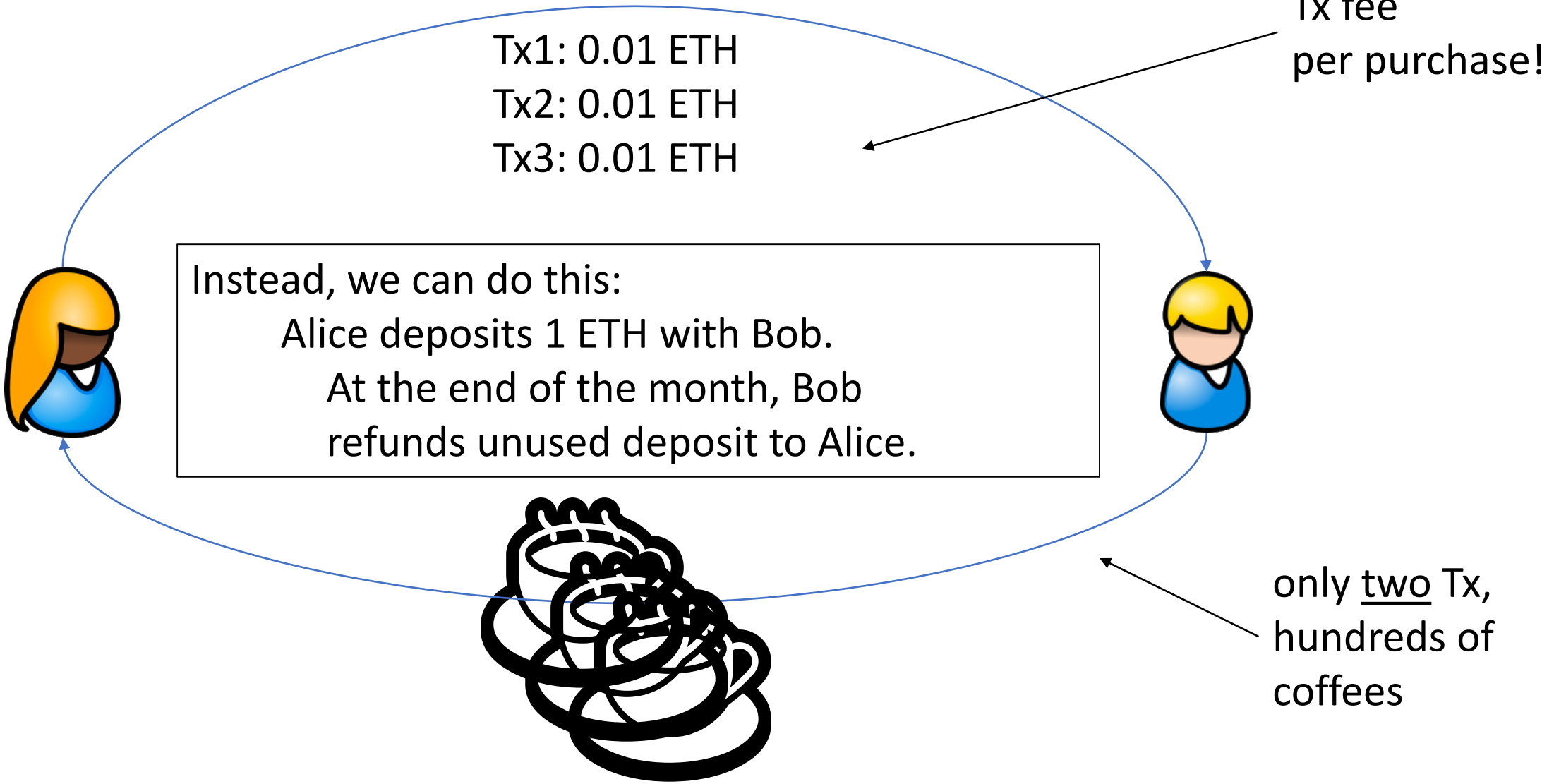
Goal: scale up blockchain Tx speed

How to process more Tx per second

Many approaches:

- Use a faster consensus protocol
- Payment channels, reduce the need to touch the chain

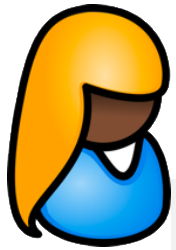
Payment Channels: the basic idea



Unidirectional Payment Channel

Alice creates:

Contract A:
1 ETH



Bob does not post on chain

Post Tx3 on Blockchain (close channel)

Tx1: send 0.99 to Alice / 0.01 to Bob from Contract A signed by Alice

Tx2: send 0.98 to Alice / 0.02 to Bob from Contract A signed by Alice

Tx3: send 0.97 to Alice / 0.03 to Bob from Contract A signed by Alice

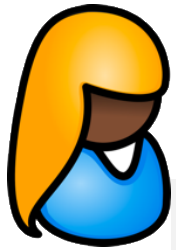
Problem: Alice could post Tx1 before Bob even though she bought three coffees.

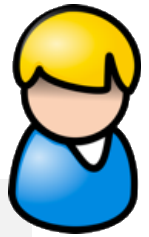
A solution?

Alice creates:

Contract A:
1 ETH

Only Bob can close the channel





Post Tx3 on Blockchain (close channel)

Tx1: send 0.99 to Alice / 0.01 to Bob from Contract A signed by Alice

Tx2: send 0.98 to Alice / 0.02 to Bob from Contract A signed by Alice

Tx3: send 0.97 to Alice / 0.03 to Bob from Contract A signed by Alice

Problem: What if Bob never publishes Tx3?
⇒ Alice never gets her 0.97 ETH back !!

Unidirectional Payment Channel

Alice needs a way to ensure refund if Bob disappears

Solution: create a channel that can be closed in one of two ways

- **Normal close Tx**: Sends 0.97 to Alice / 0.03 to Bob

  … requires signatures by both Alice and Bob.

- **Timelock Tx**: Sends 1 ETH to Alice

  … requires signature by Alice,
  but is accepted 7 days after channel is created

Unidirectional Payment Channel

After 6 days:

- Bob can close channel by signing and posting Tx3.

After 7 days:

- Alice can close channel using timelock Tx, gets back her 1 ETH.

- Timelock period determines the lifespan of channel

- Once Alice sends the full 1 ETH to Bob, the Channel is “exhausted”

Payment Channel in Solidity

```
contract SimplePaymentChannel {
    address payable public sender;      // The account sending payments.
    address payable public recipient;   // The account receiving the payments.
    uint256 public expiration;  // Timeout in case the recipient never closes.

    constructor (address payable _recipient, uint256 duration)
        public
        payable
    {
        sender = msg.sender;
        recipient = _recipient;
        expiration = now + duration;
    }


    /// the recipient can close the channel at any time by presenting a
    /// signed amount from the sender. the recipient will be sent that amount,
    /// and the remainder will go back to the sender
    function close(uint256 amount, bytes memory signature) public {
        require(msg.sender == recipient);
        require(isValidSignature(amount, signature));

        recipient.transfer(amount);
        selfdestruct(sender);
    }

    /// if the timeout is reached without the recipient closing the channel,
    /// then the Ether is released back to the sender.
    function claimTimeout() public {
        require(now >= expiration);
        selfdestruct(sender);
    }
}
```

Alice creates contract with funds, specifies timelock and recipient

verify Alice's signature on final amount.
Only Bob can call close() !!

send all funds to sender after timelock

Bidirectional Payment Channel

Alice and Bob want to move funds back and forth

Two Unidirectional Channels?

Not as useful because Channels get exhausted

Bidirectional Payment Channel

On Ethereum: create a shared contract, each contributes 0.5 ETH:

channel state: A: 0.5 ETH, B: 0.5 ETH, Nonce=0

Off chain: Bob sends 0.1 ETH to Alice by both signing new state:

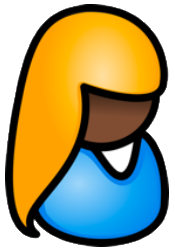
new state: A: 0.6, Bob: 0.4, Nonce=1

Alice sig, Bob sig

Bidirectional Payment Channel

On chain contract does not change:

balance: 1 ETH, Nonce=0



Off chain: Alice and Bob can move funds back and forth
by sending updated state sigs to each other:

A: 0.3, Bob: 0.7, Nonce=7

Alice sig, Bob sig

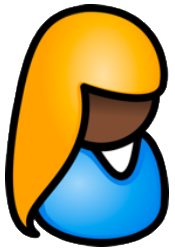
(7^{th} transfer)

Eventually: Alice wants to close payment channel

Alice does: sends latest balances and signatures to contract

⇒ starts challenge period (say, 3 days)

on chain: A: 0.3 ETH, B: 0.7 ETH, Nonce=7 (pending close)



if Bob does nothing for 3 days:

⇒ funds disbursed according to Alice's submitted state

if Bob submits signed state with a higher nonce (e.g., nonce=9)

⇒ funds disbursed according to Bob's submitted state

Watchtowers

- A lightning wallet must be online on a regular basis to track it's payment channels for cheating attempts.
- A daily spending wallet, in particular, is offline whenever the user stops using the app. This means there is a greater risk of cheating attempts.
- A watchtower service can fix this.
  - Watchtowers monitor the payment channels of offline users. If a counterparty attempts to steal a user's funds, the watchtower can step in to help.
  - The watchtower can prevent the theft by submitting a justice transaction.

Watchtowers

Bidirectional channel requires Bob to constantly check that Alice did not try to close the channel with an old stale state

⇒ post latest state if she did

Watchtowers outsource this task

Trusted for availability

Bob sends latest state to watchtower.

Main points: summary

Payment channel between Alice and Bob:

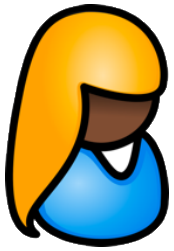
- **One on-chain** Tx to create channel (deposit funds);
- Alice & Bob can send funds to each other **off-chain**
  … as many Tx as they want;
- **One on-chain** Tx to close channel and disburse funds

⇒ only two on-chain Txs

A more general concept: State Channels

Smart contract that implements a game between Alice and Bob.

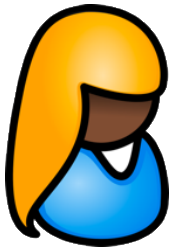
Begin game & end game: on chain. **All moves are done off-chain**.



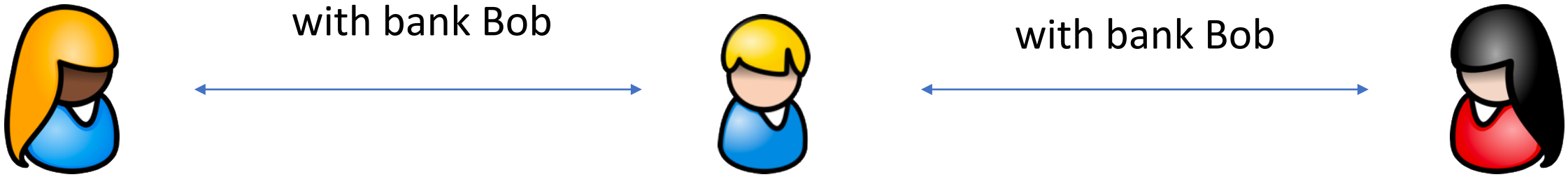
State Channels

Can be used to implement any 2-party contract off chain!

two Txs on-chain: 1) contract creation and 2) termination



Multi-hop payments (Bitcoin's lighting)

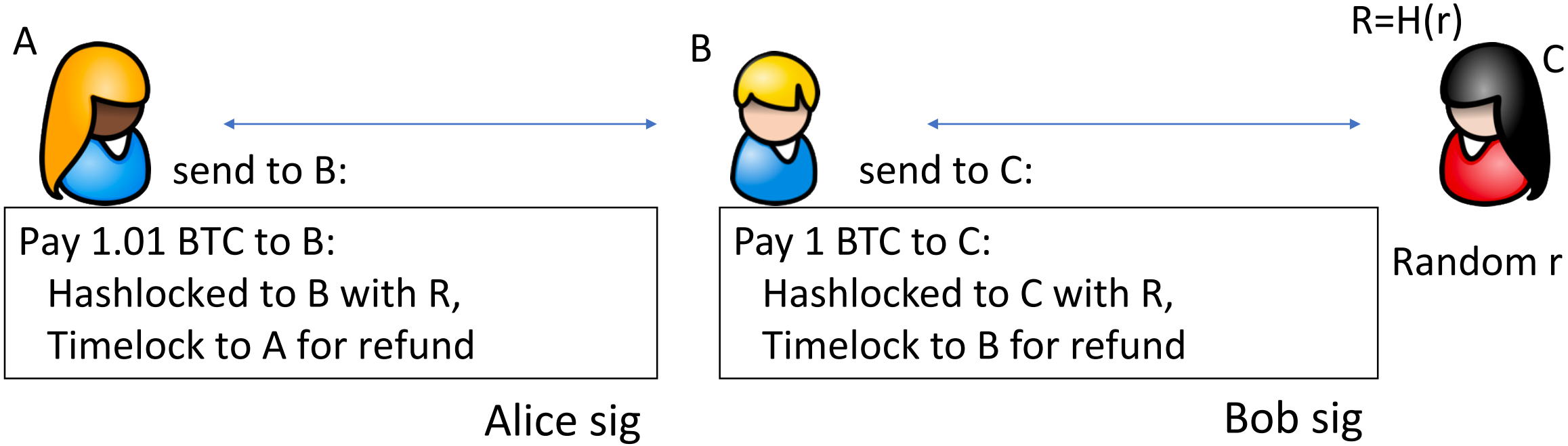


Alice wants to pay Carol 1 BTC through *untrusted* intermediary Bob

How: (i) Alice pays Bob 1.01 BTC, (ii) Bob pays Carol 1 BTC

The challenge: steps (i) and (ii) need to be **atomic**

Multi-hop payments (briefly)



Then B can claim 1.01 BTC with r

C can claim 1 BTC on-chain with r
⇒ r is publicly known

if Carol never claims, Alice & Bob get funds back after timelock

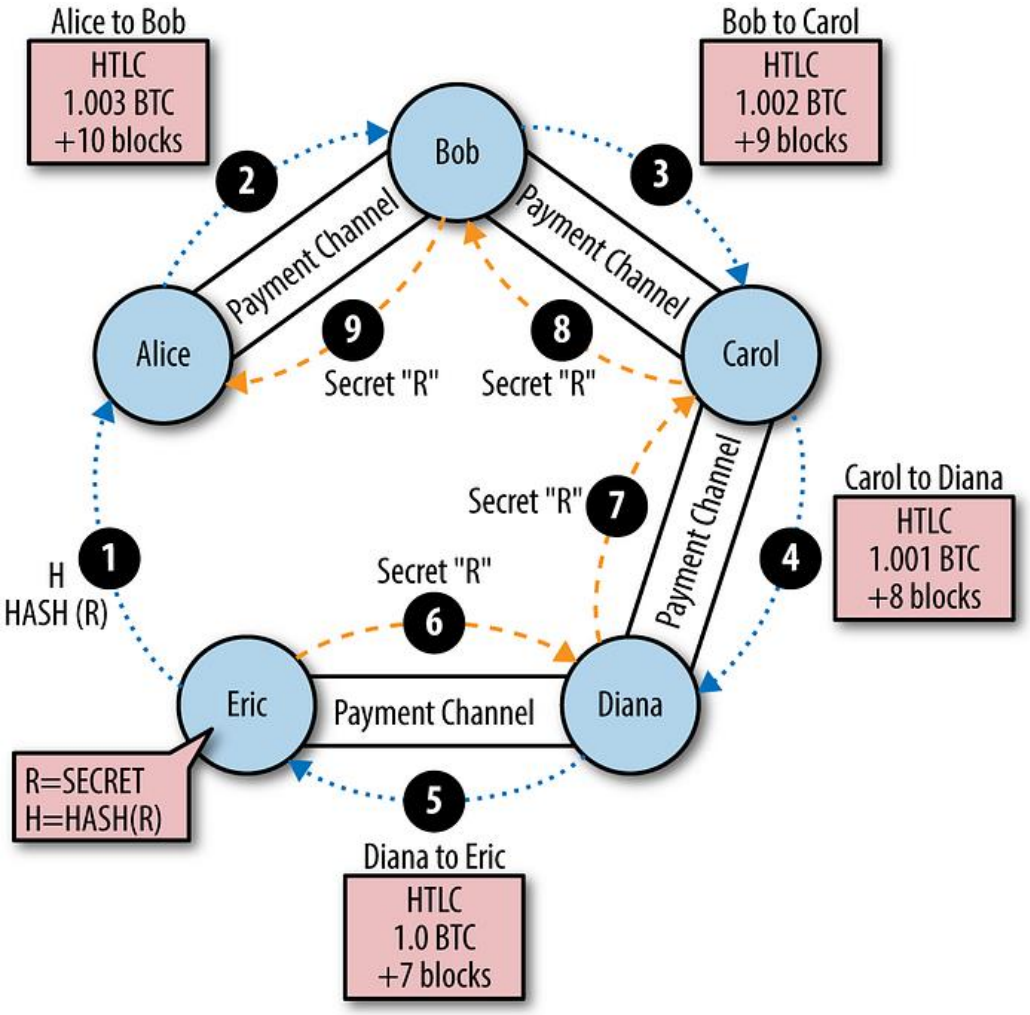
Atomicity of Payment Channel

- The atomicity in the payment channel means that the first payment and the second payment in a multi-hop payment channel network must be done as a whole or none.

- The atomicity can be achieved by hashed time-lock contract (HTLC) in the lightning network (Bitcoin)

```
OP IF
   OP HASH160 <Hash160 (R)> OP EQUALVERIFY
   2 <Alice2> <Bob2> OP CHECKMULTISIG
OP ELSE
   2 <Alice1> <Bob1> OP CHECKMULTISIG
OP ENDIF 1> <Bob1> OP CHECKMULTISIG
OP ENDIF
```

Multi-hop example

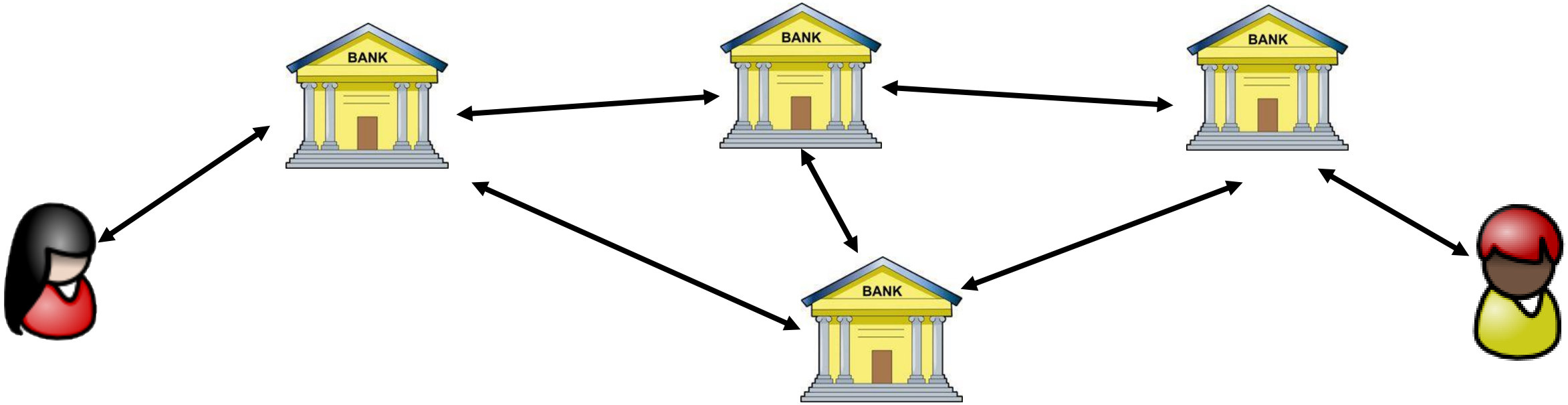


1. Eric generates a secret('R'), hashes that secret, and sends it to Alice. The secret is only known by Eric.
2. Alice's node computes Eric's payment (1BTC), and creates an HTLC to pay Bob 1.03BTC if he can provide the secret within the next 10 blocks, otherwise, the funds will be refunded to Alice.
3. …
4. Until reaching Eric, he then provides the secret('R') that he generated and unlocks the HTLC to get the 1BTC payment
5. Other parties, Diana, Carol, Bob, and Alice will eventually retrieve the funds locked in

The lightning network

The network: lots of open bi-directional payment channels.

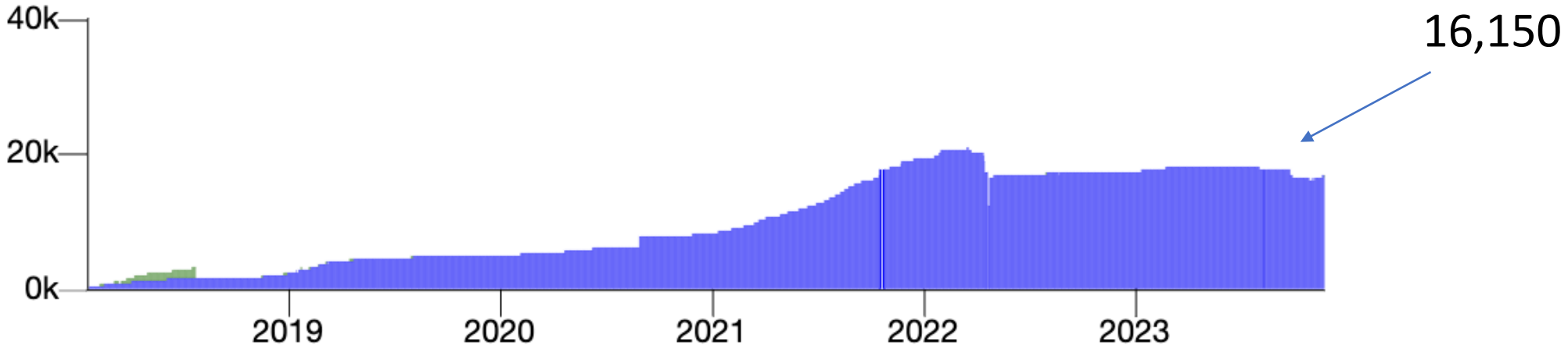
Alice wants to pay Bob: she finds a route to Bob through the graph



Many extensions possible: multi currency hubs, credit hubs, …

Stats

nodes in lightning network (Nov. 2023)



Number of channels: 63K

Network capacity: ≈$205M

Outline

- Scaling blockchain by payment channel

- Blockchain applications

Blockchain applications

1. Blockchain for Internet of Things
2. Blockchain for mobile crowdsensing

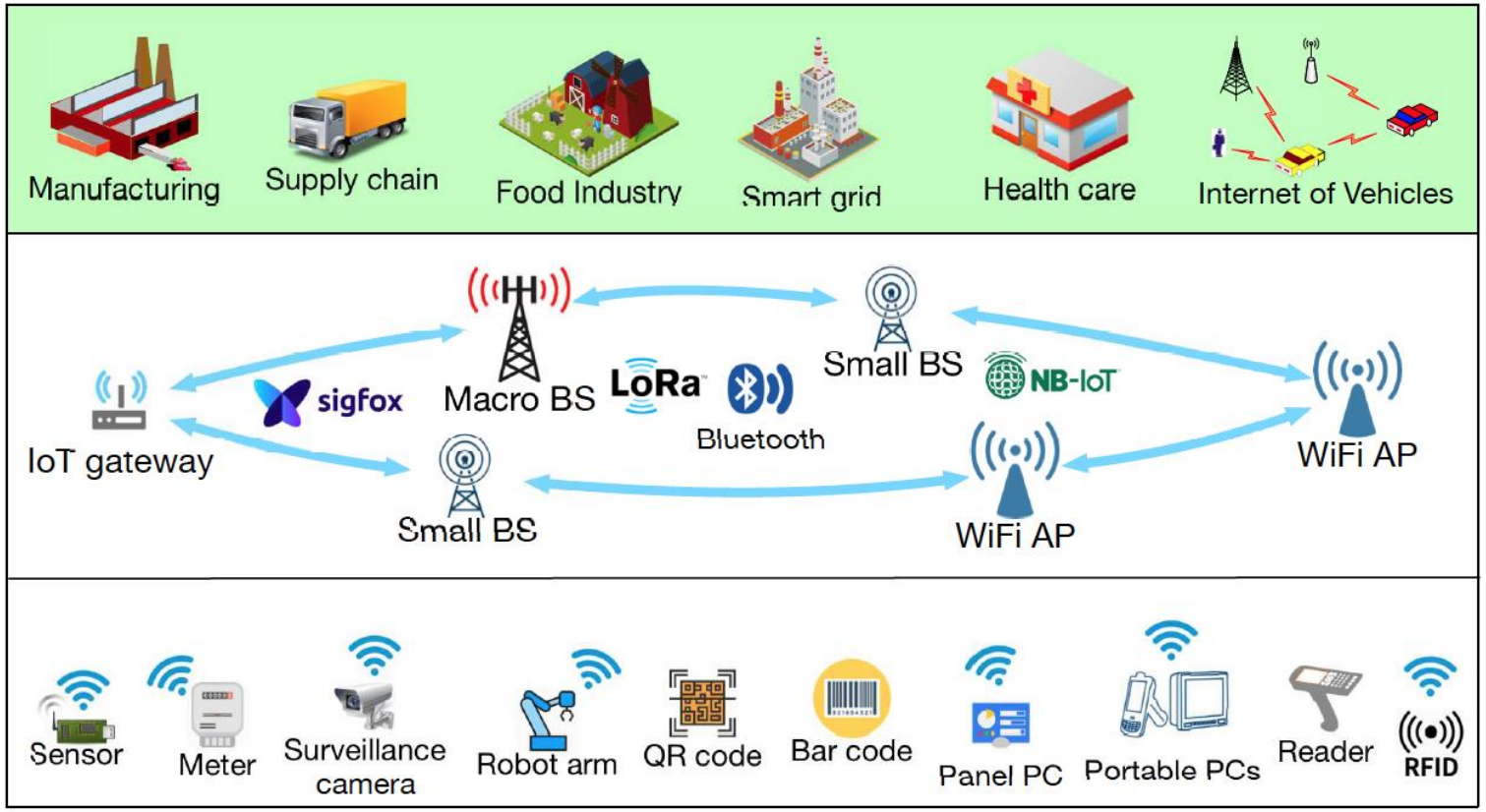
2.1 Internet of Things

- Internet of Things (IoT) is a network of intelligent computers, devices, and objects that collect and share huge amounts of data
- IoT can support a number of industrial applications

Industrial applications

Communication layer

Perception layer



2.1 Challenges of IoT

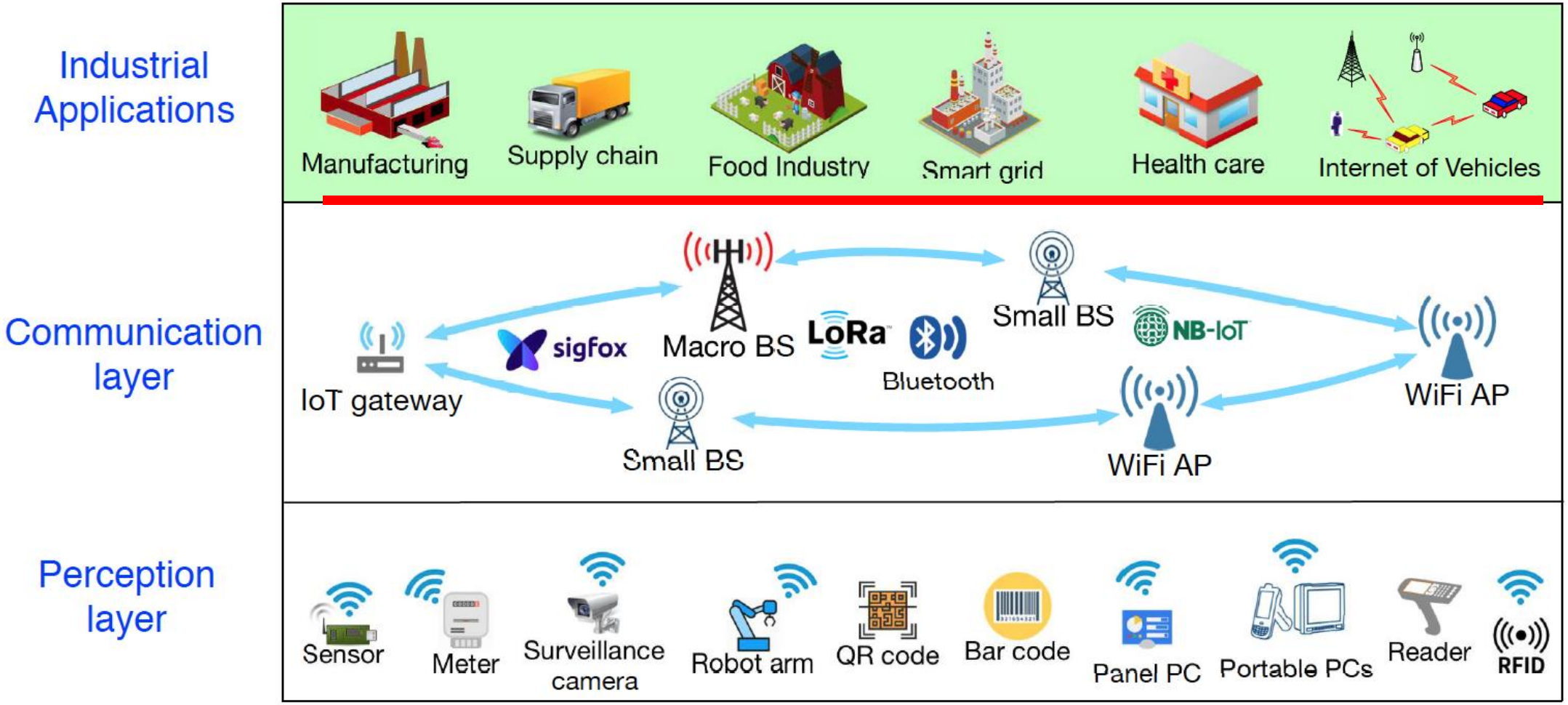
- Heterogeneity of IoT systems
  - Heterogeneous IoT devices, heterogeneous communication protocols and heterogeneous IoT data types
- Complexity of networks
  - Different types of IoT protocols (NFC, WiFi, 6LoWPAN, NB-IoT, LoRa)
- Poor interoperability
  - Due to the decentralization and the heterogeneity of IoT systems

2.1 Challenges of IoT (cont.)

- Resource constraints of IoT devices
  - IoT devices such as sensors, actuators, RFID tags and smart meters suffer from limited resources including computing resource, storage resource and battery power.
- Privacy vulnerability
  - Due to the complexity and decentralization of IoT systems
  - Uploading data to remote clouds -> risk of privacy exposure
- Security vulnerability
  - Authentication, authorization and communication encryption may not be applicable to distributed IoT

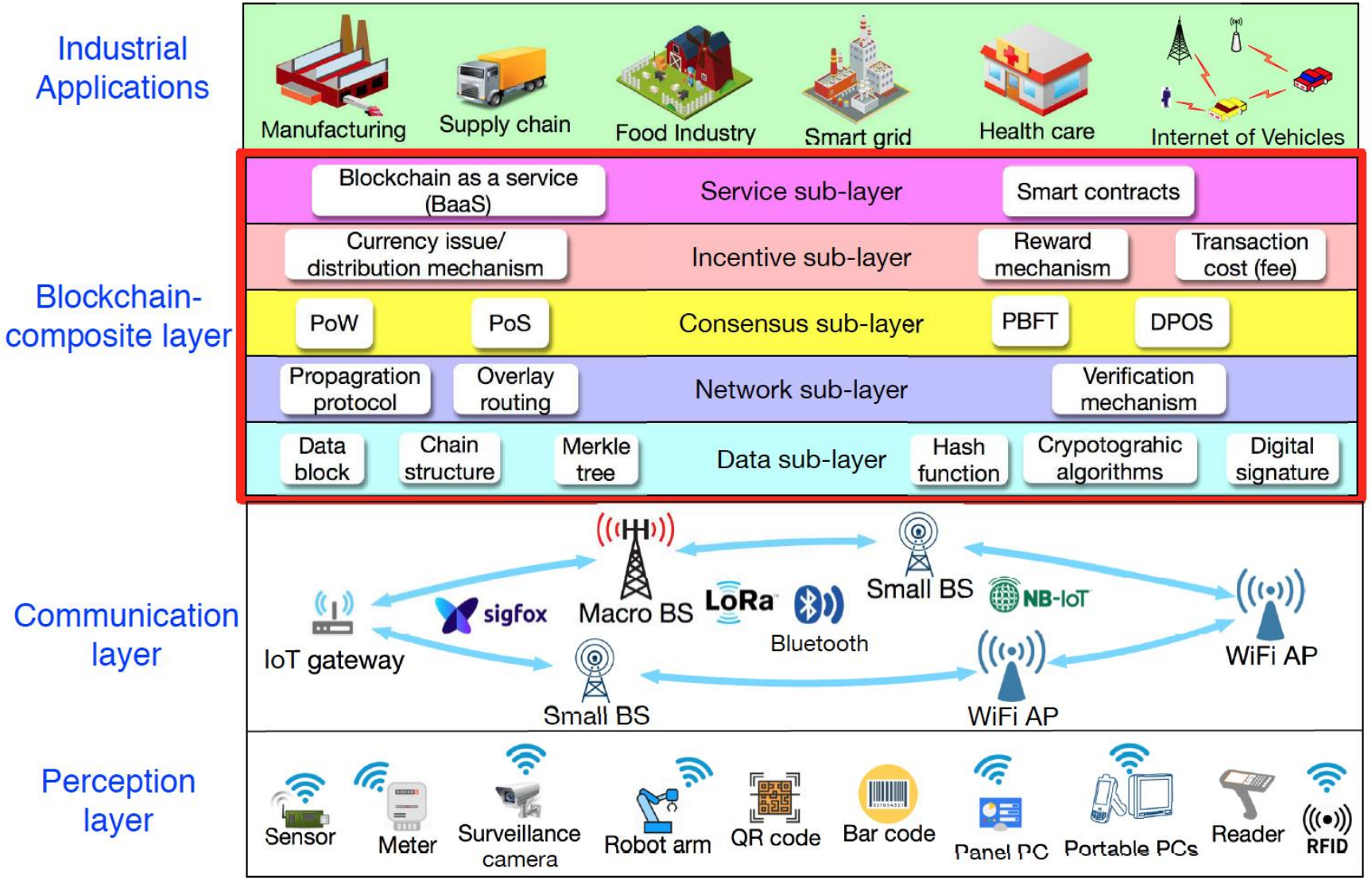
2.2 Architecture of Blockchain of Things

- We name such integration of blockchain with IoT as BCoT.



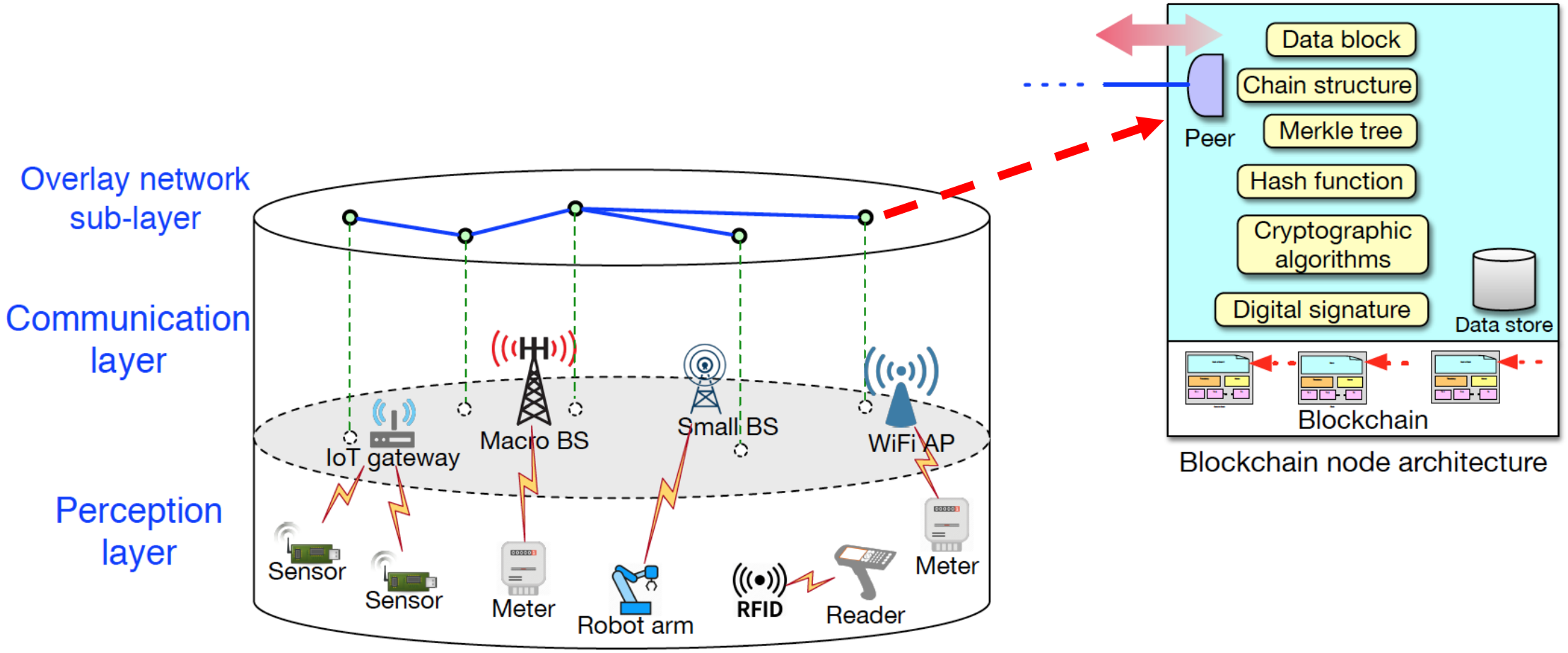
2.2 Architecture of Blockchain of Things

- We name such integration of blockchain with IoT as BCoT.



P2P overlay network and blockchain node architecture

- Each node in overlay network consists of a number of components corresponding to the composite blockchain layer

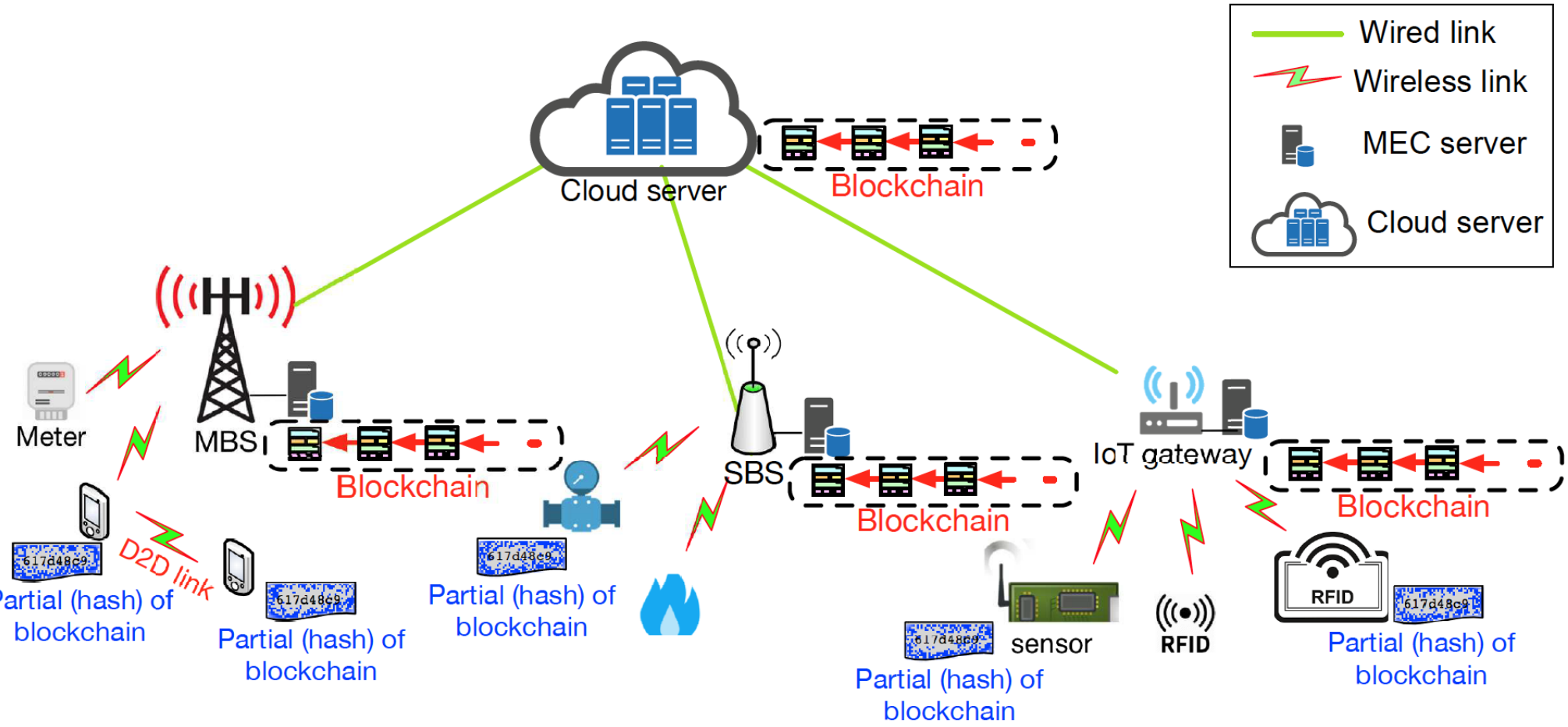


Features of BCoT

- Blockchain serves as a middleware between IoT and applications
  - offering an abstraction from the lower layers in IoT
  - providing users with blockchain-based services
- Hiding complexity of IoT systems
- Offering general interfaces to various IoT applications

Opportunities of integrating blockchain with IoT

- Enhanced interoperability of IoT systems
  - Transforming various IoT data into blockchain
  - Interoperating crossing different types of fragmented networks
- Improved security of IoT systems
  - Encryption and digital signature
- Traceability and Reliability of IoT data
  - Data in blockchain is traceable and reliable
- Autonomic interactions of IoT systems
  - Distributed autonomous Corporations (DACs) to automate transactions



- Cloud servers and edge servers may store the *whole blockchain* (or partial blockchain) data and IoT devices only store the *partial blockchain*

Orchestration of Cloud and edge computing is a necessity!

Acknowledgement

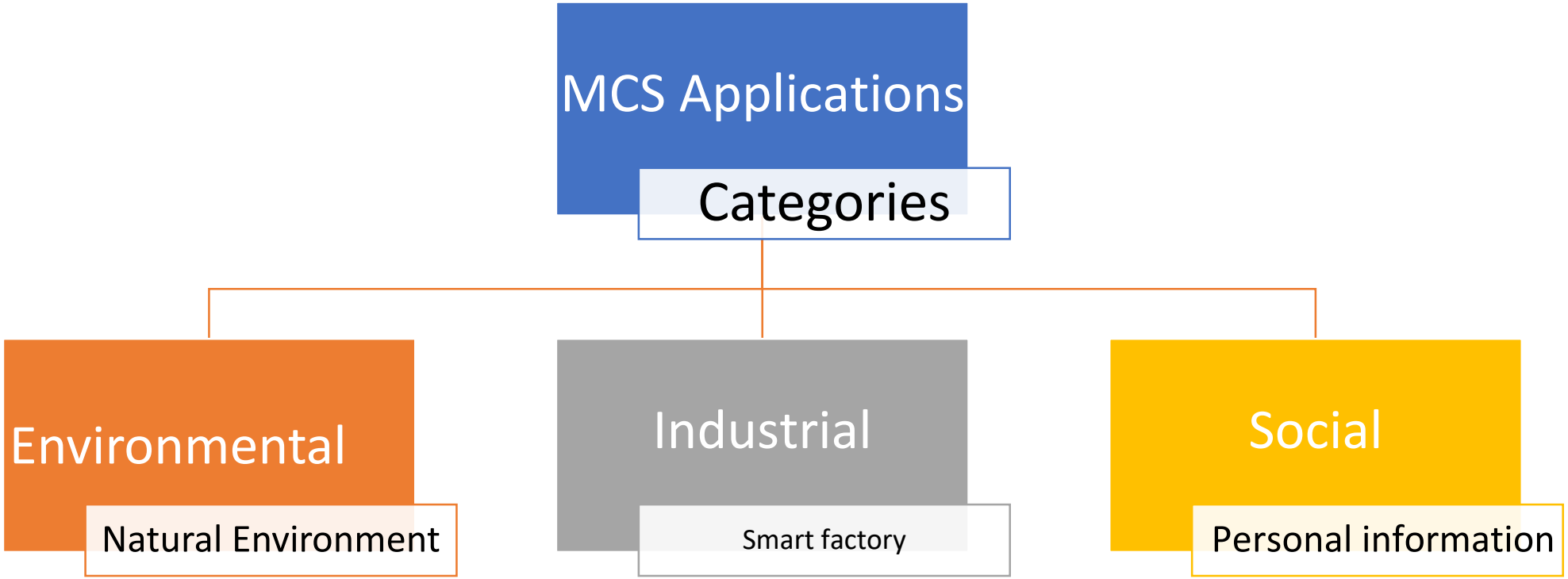
- This is a joint work with Zibin Zheng (Sun Yat-sen University, China) and Yan Zhang (Oslo University, Norway)
- For more details, please refer to our paper entitled "*Blockchain for Internet of Things: A Survey*" in (ESI highly-cited paper) IEEE Internet of Things Journal, 2019

Blockchain applications

1. Blockchain for Internet of Things
2. Blockchain for mobile crowdsensing

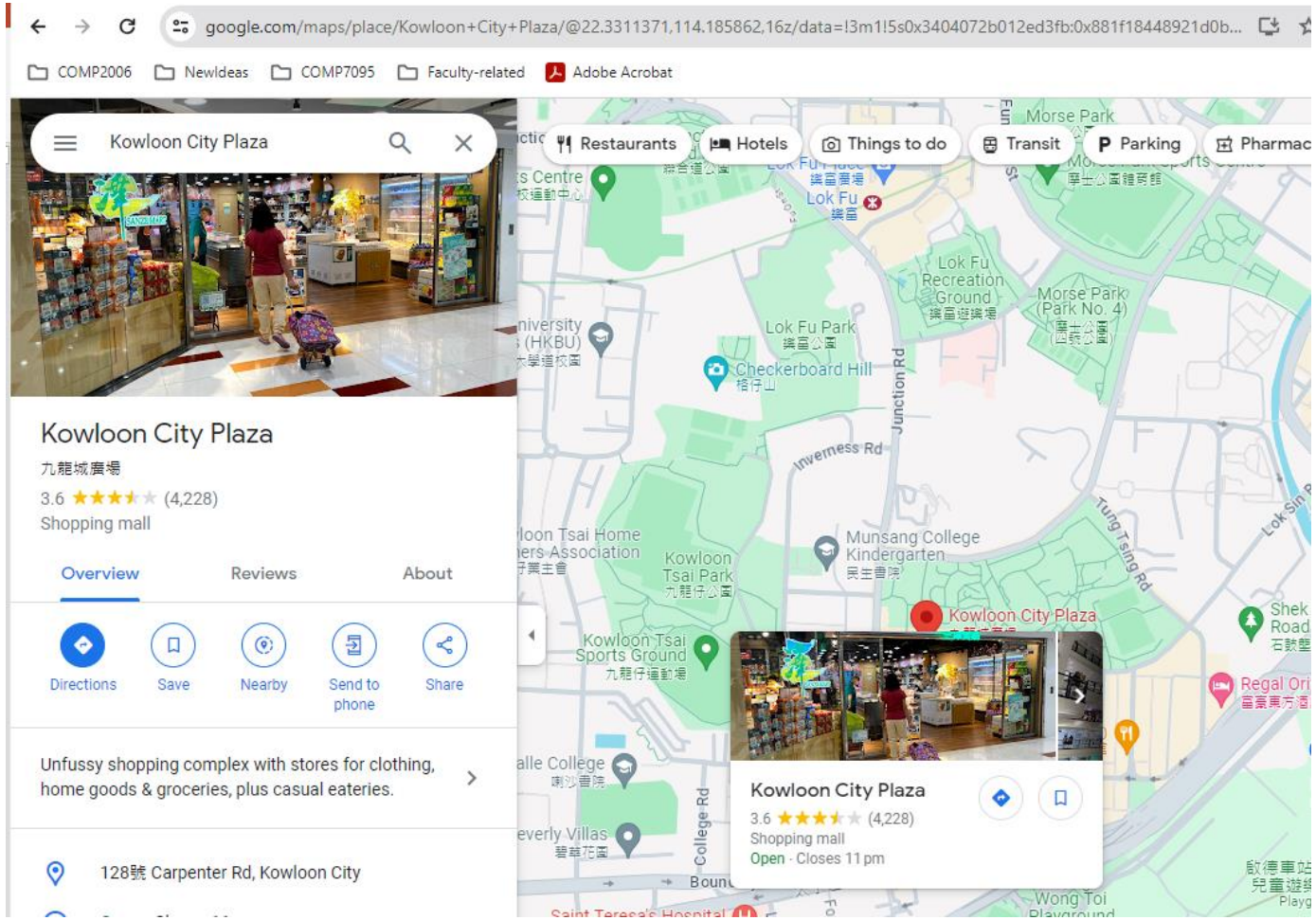
3.1 Mobile Crowdsensing

- Mobile crowdsensing (MCS) has become a new paradigm which takes advantages of pervasive mobile devices and sensors to collect data efficiently, thereby enabling numerous applications.



Examples of MCS

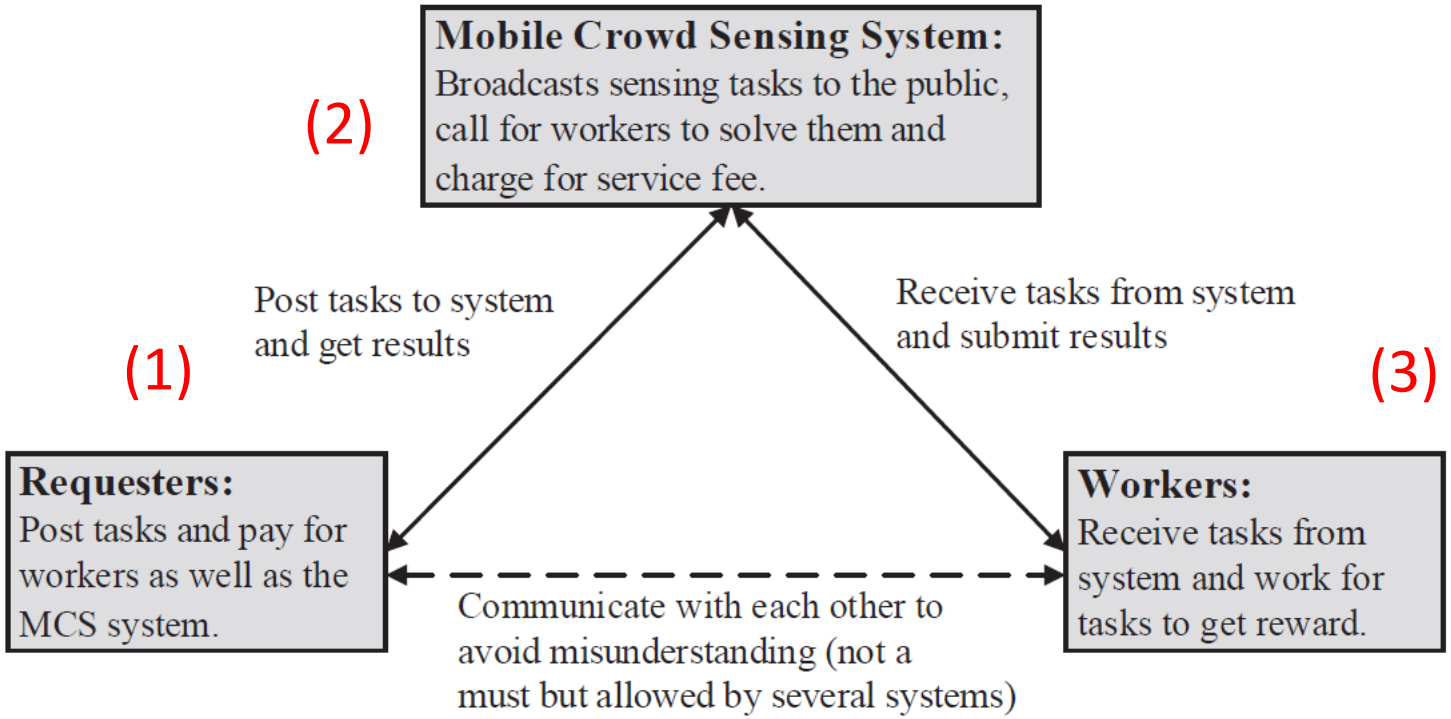
Google street map



HealthMap

https://www.healthmap.org/en/

Traditional mobile crowdsensing triangular architecture



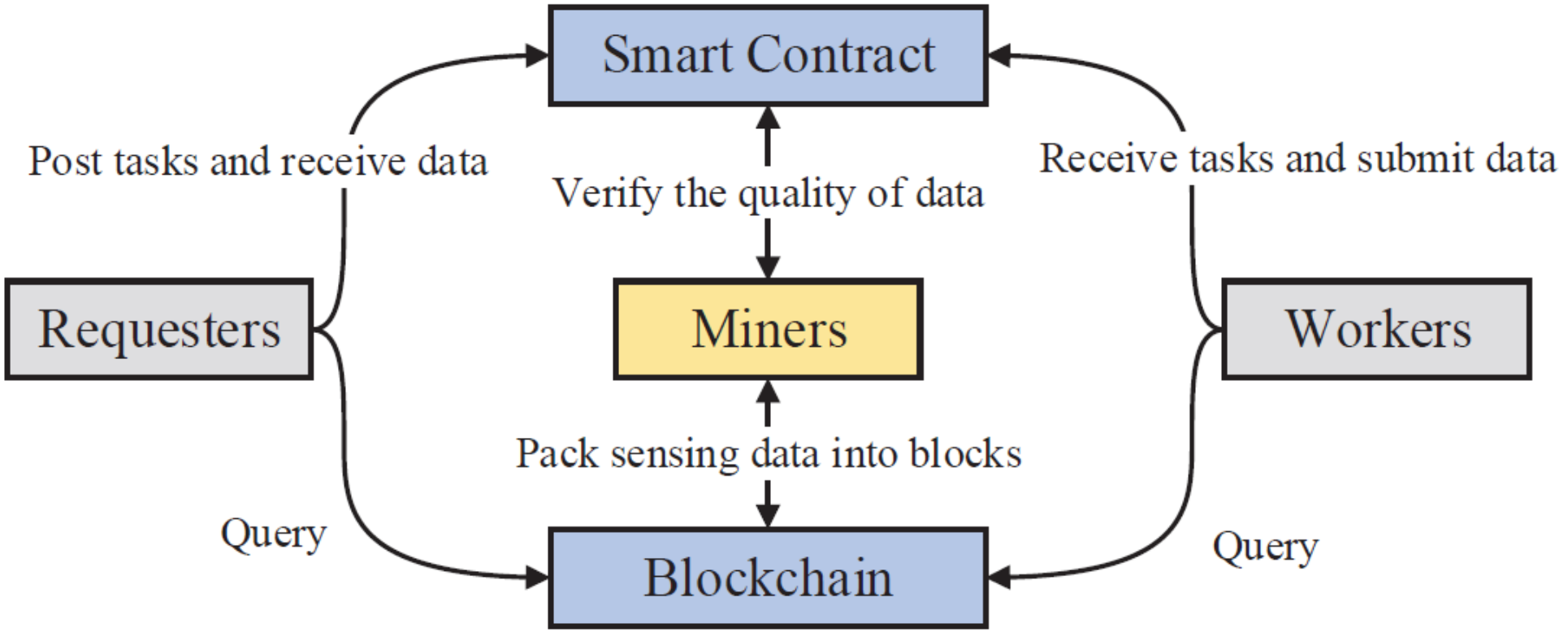
Challenges of MCS

- Reliability of MCS
  - Centralized architecture requires mobile devices to communicate with MCS system -> susceptible to single-point-failure
- Security of MCS
  - MCS is exposed to the risk of DDoS and man-in-the-middle attacks
  - Data at MCS server to be leaked or falsified
- Data quality of MCS
  - Data collected by human being (i.e., behave maliciously or unreliably-> poor quality of sensing data)

3.2 Why blockchain can be used for MCS?

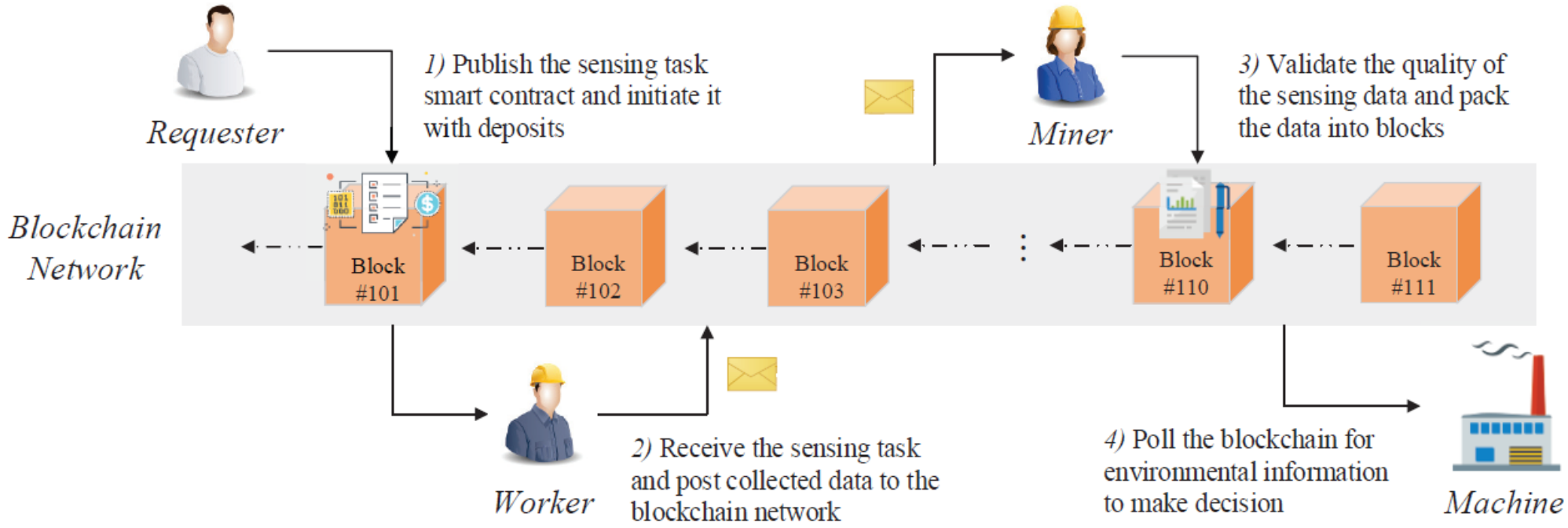
- Sensory data at centralized server of MCS cannot be fully trusted while blockchain can guarantee decentralized trust (non-tamperable data)
- Decentralized architecture of blockchain can also improve the system reliability
- Smart contracts on top of blockchain can prevent malicious behaviors such as free-riding and false-reporting in MCS

Blockchain for MCS (BMCS)



Decentralize conventional centralized MCS architecture!

Working procedure of BMCS

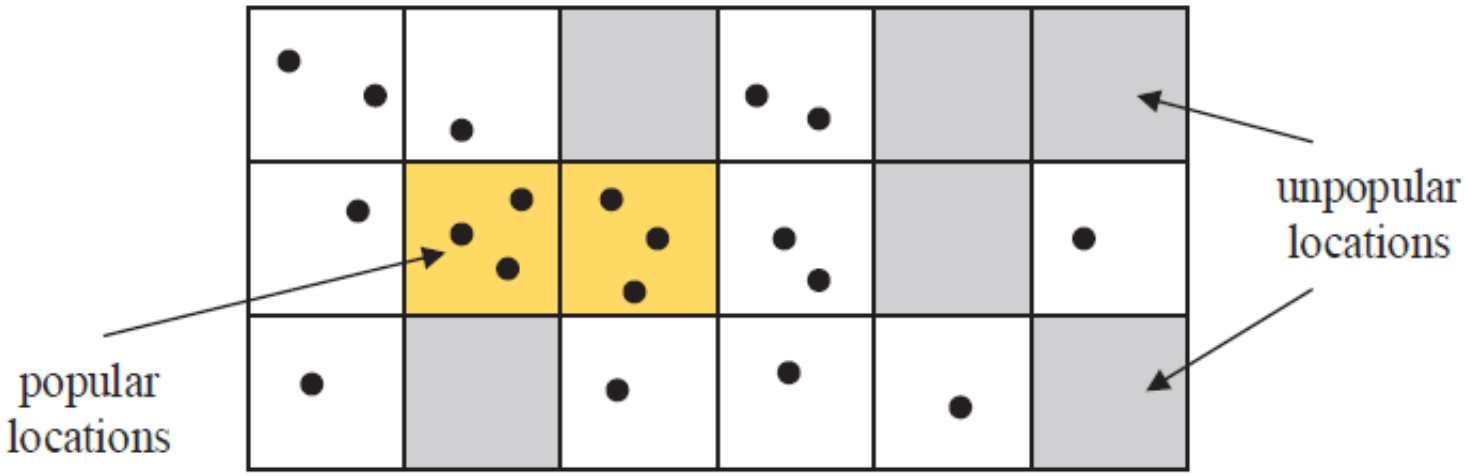


Two issues to be solved:

1. Incentive mechanism to motivate workers to contribute to sensing tasks (especially on some unpopular regions)
2. Data quality assurance mechanism

Incentive mechanism

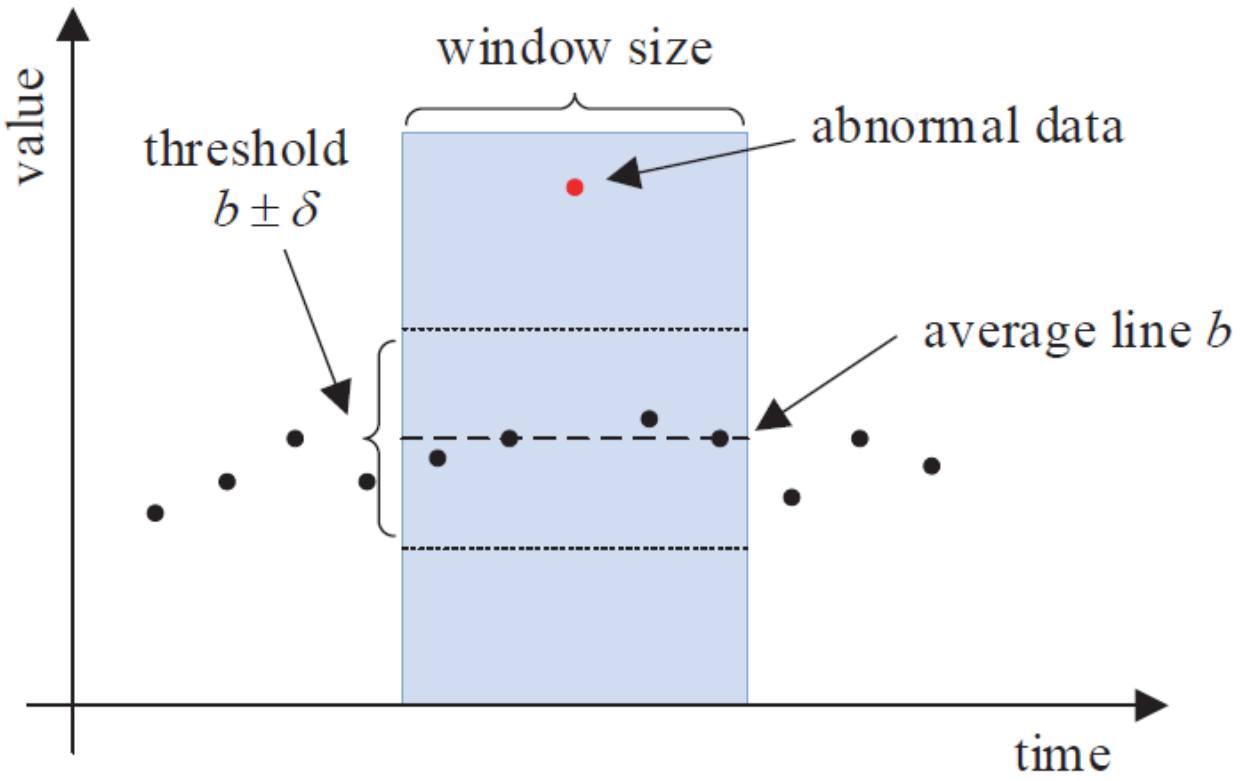
- Dynamic reward ranking (DR2) incentive mechanism



$$r_i = \begin{cases} w \times (d - k_i) / \sum_{i=1}^{n} \max(d - k_i, 0) & if\ k_i < d; \\ 0 & if\ k_i \geq d. \end{cases} \quad (1)$$

Sensory Data Quality Detection Scheme

- Normal data contributed by users has the temporal stability and spatial correlation
- Abnormal Data due to hardware faults or fake data submitted by dishonest users;

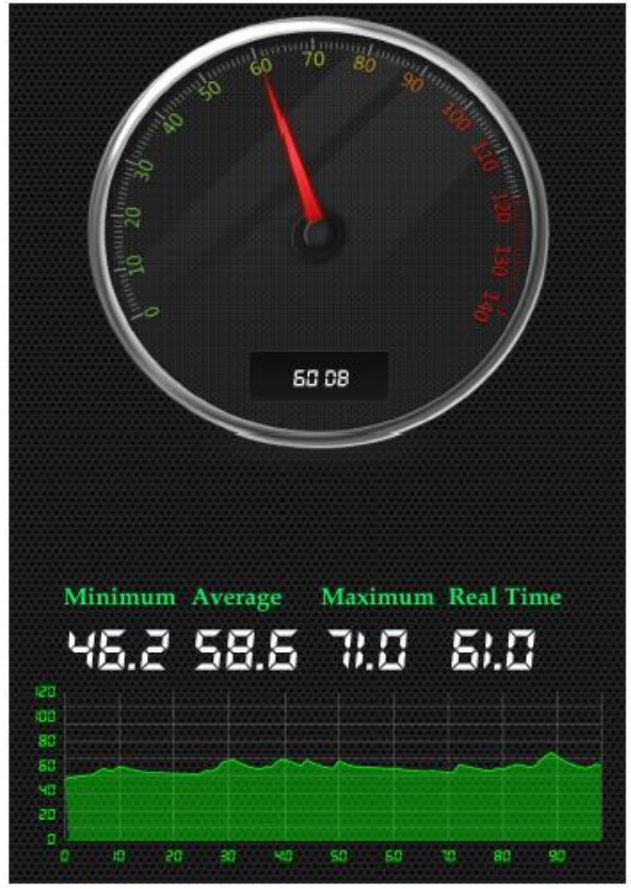


If the data point is in the range of $a \pm \beta$, it is normal data; it is abnormal otherwise.

Experiments

- We implemented a prototype of BMCS on Ethereum and conducted sound sensing tasks in the public blockchain test network.
- We used 6 MI-4 mobile phones for workers

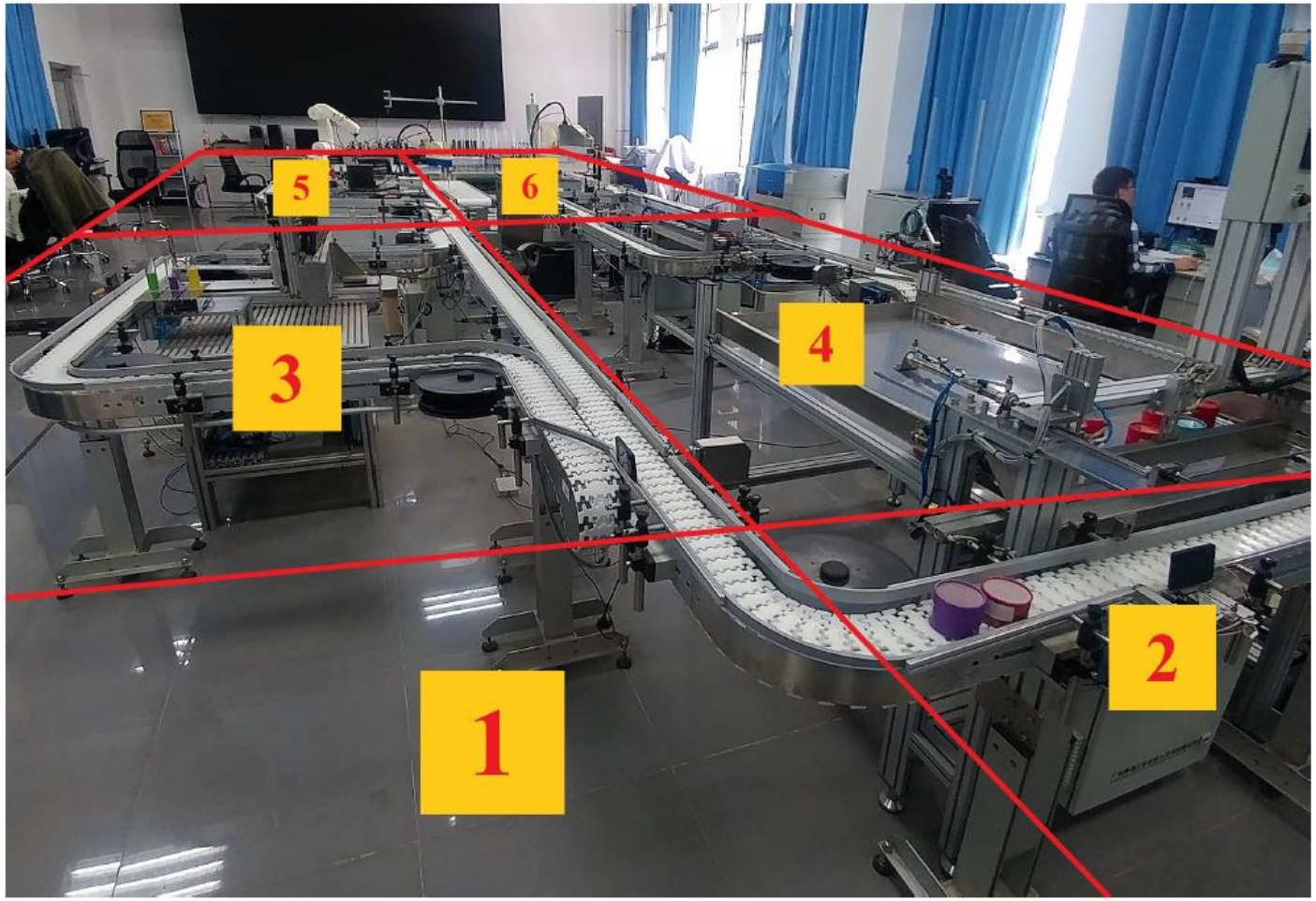
(a) MI-4 mobile phones



(b) Sound sensing screenshot

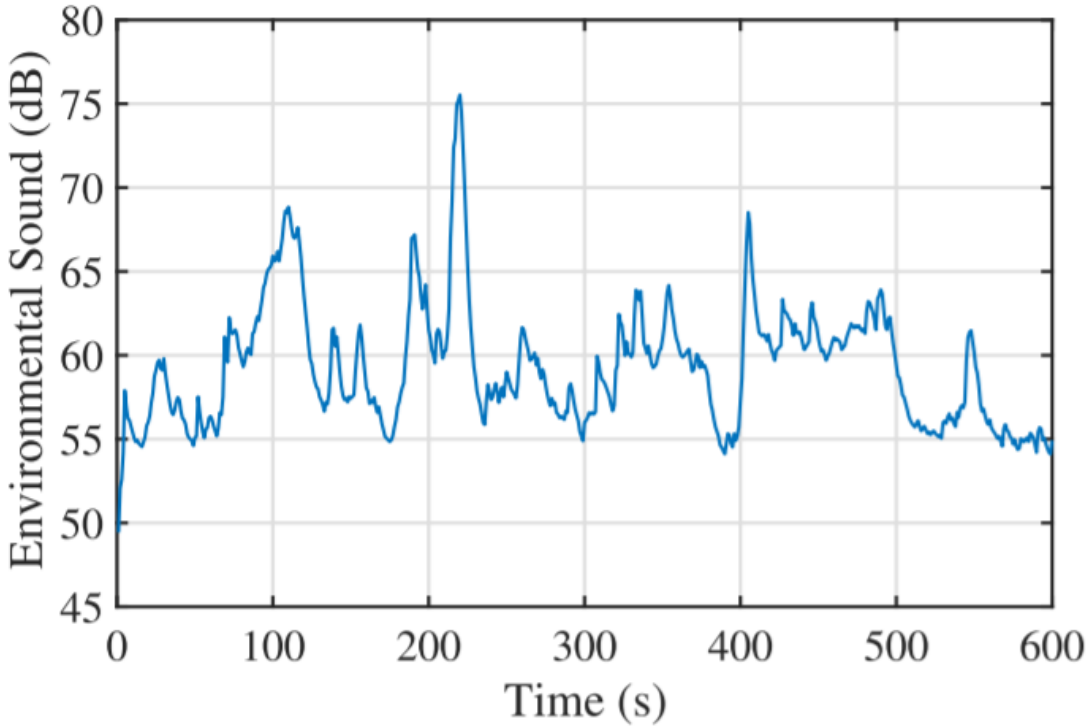
Real testbed

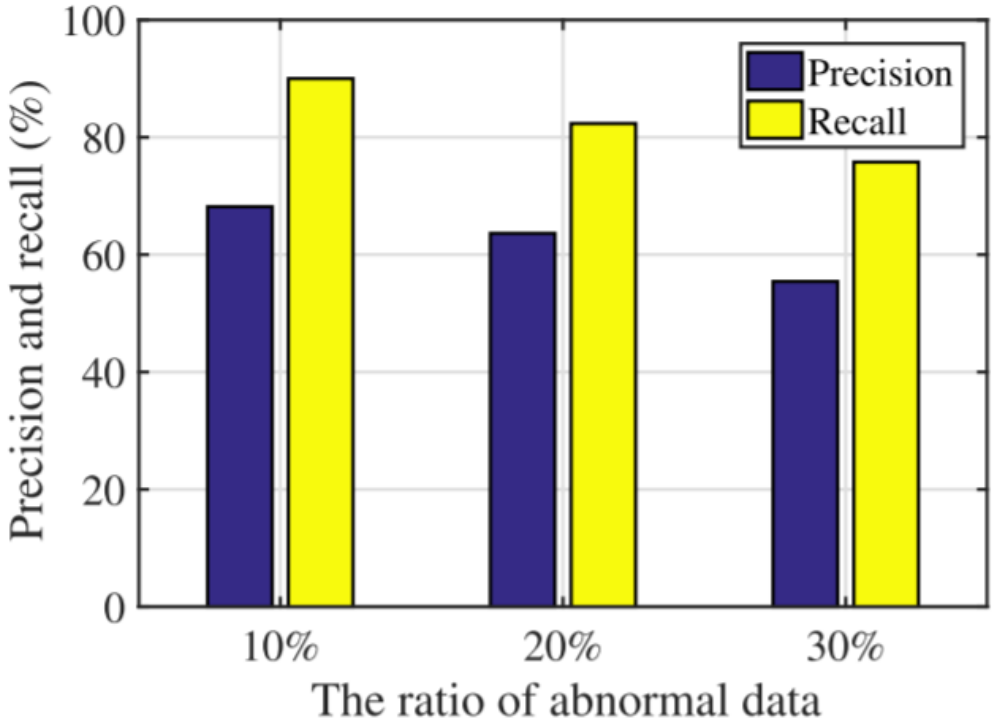
- We conducted experiments in a factory workroom
- Test scenario of sound sensing tasks.



Sensory Data Quality Detection Scheme

Proposed data quality detection scheme achieves an excellent performance





Acknowledgement

- This is a joint work with Junqin Huang, Linghe Kong (Shangjiao Jiao Tong University, China), Steve Xue Liu (McGill University, Canada) and Long Cheng (Clemson University, USA)
- For more details, please refer to our paper entitled "BlockSense: Towards trustworthy mobile crowdsensing via proof-of-data blockchain" in IEEE Transactions on Mobile Computing, 2024

Summary

- Speed up blockchain by different strategies (e.g., Payment channel network)

- Blockchain applications
  - Blockchain for IoT
  - Blockchain for mobile crowdsensing

Thank you!