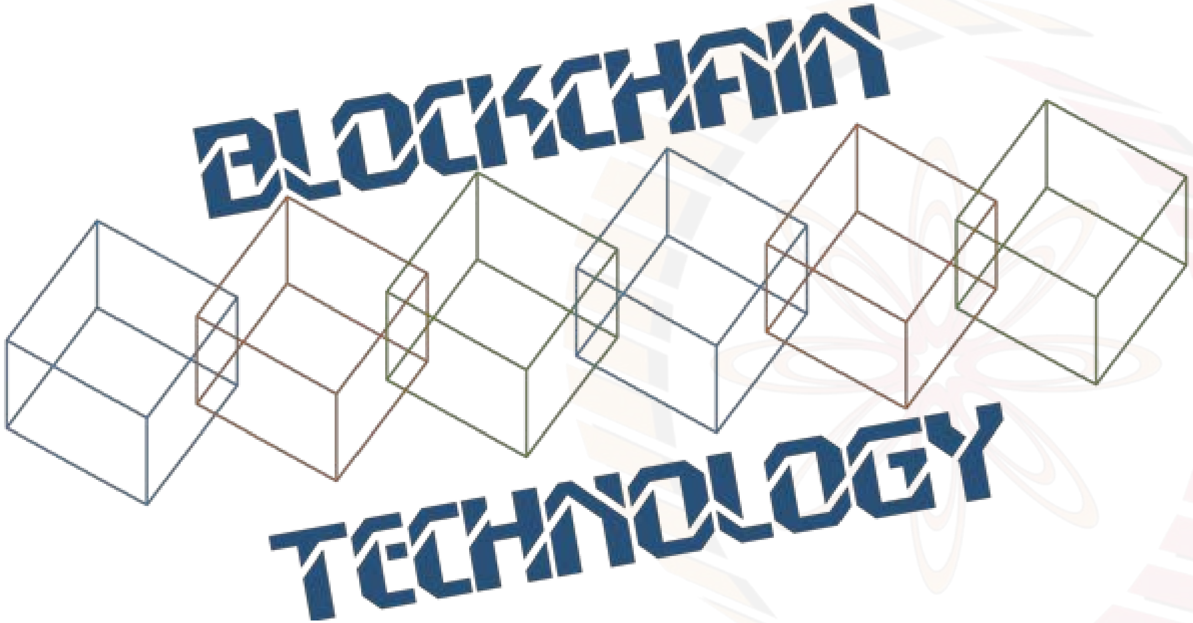
BLOCKCHAINS

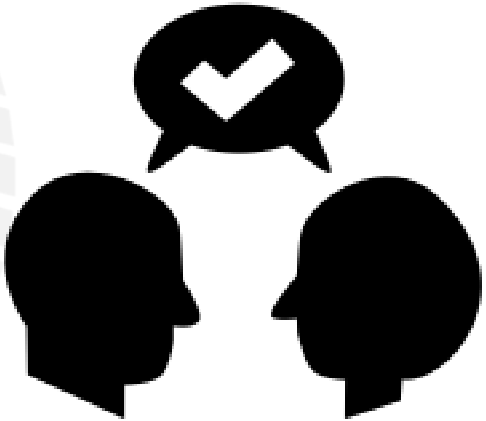
ARCHITECTURE, DESIGN AND USE CASES

SANDIP CHAKRABORTY
COMPUTER SCIENCE AND ENGINEERING,
IIT KHARAGPUR

PRAVEEN JAYACHANDRAN
IBM RESEARCH,
INDIA

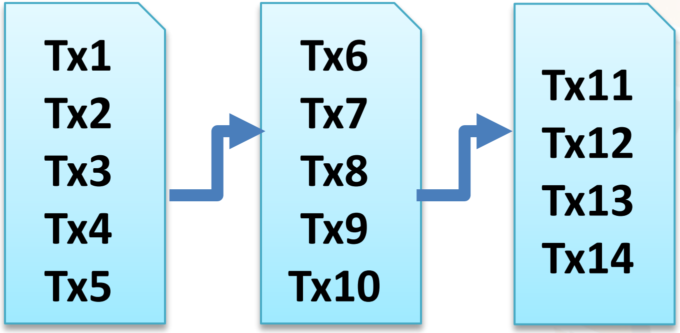
Image courtesy: http://beetfusion.com/





CONSENSUS IN BITCOIN

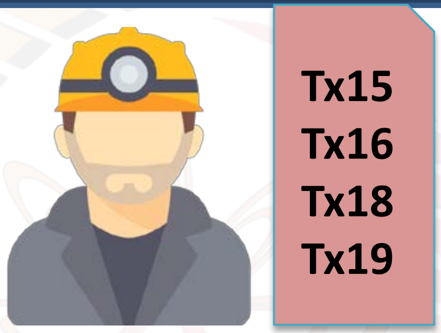
Consensus in Bitcoin



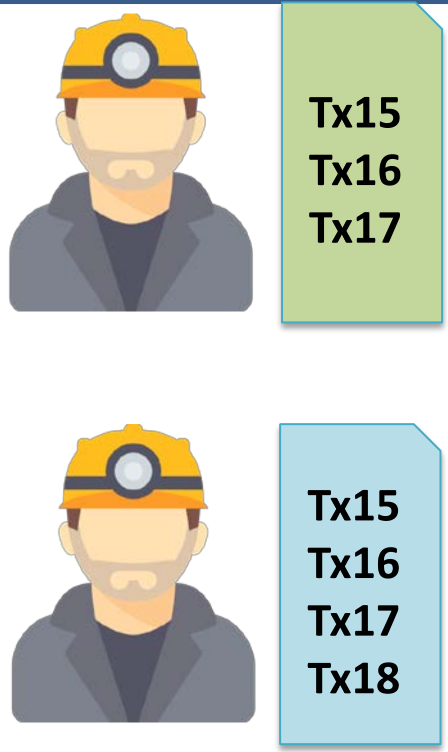
Tx1
Tx2
Tx3
Tx4
Tx5

Tx6
Tx7
Tx8
Tx9
Tx10

Tx11
Tx12
Tx13
Tx14



Tx15
Tx16
Tx18
Tx19



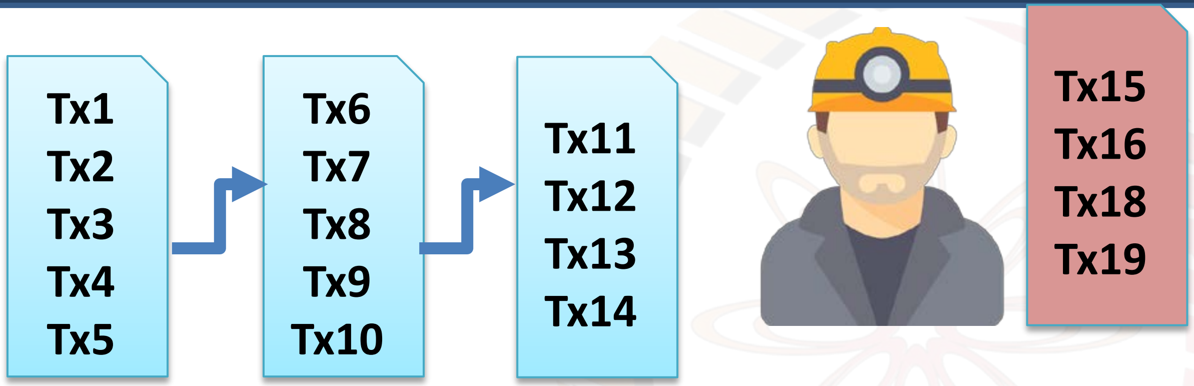
Tx15
Tx16
Tx17

Tx15
Tx16
Tx17
Tx18

Bitcoin Consensus Objective:

Which block do we add next?

Consensus in Bitcoin

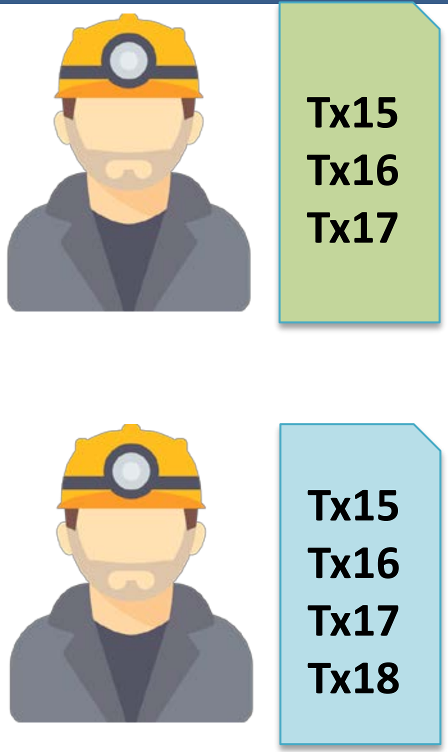


Tx1
Tx2
Tx3
Tx4
Tx5

Tx6
Tx7
Tx8
Tx9
Tx10

Tx11
Tx12
Tx13
Tx14

Tx15
Tx16
Tx18
Tx19



Tx15
Tx16
Tx17

Tx15
Tx16
Tx17
Tx18

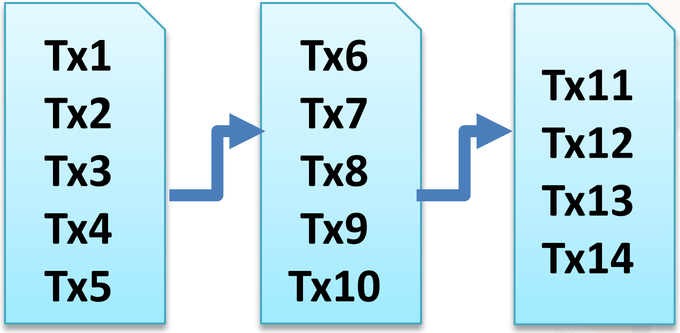
Bitcoin Consensus Objective:

Which block do we add next?

Challenge:

The miners do not know each other

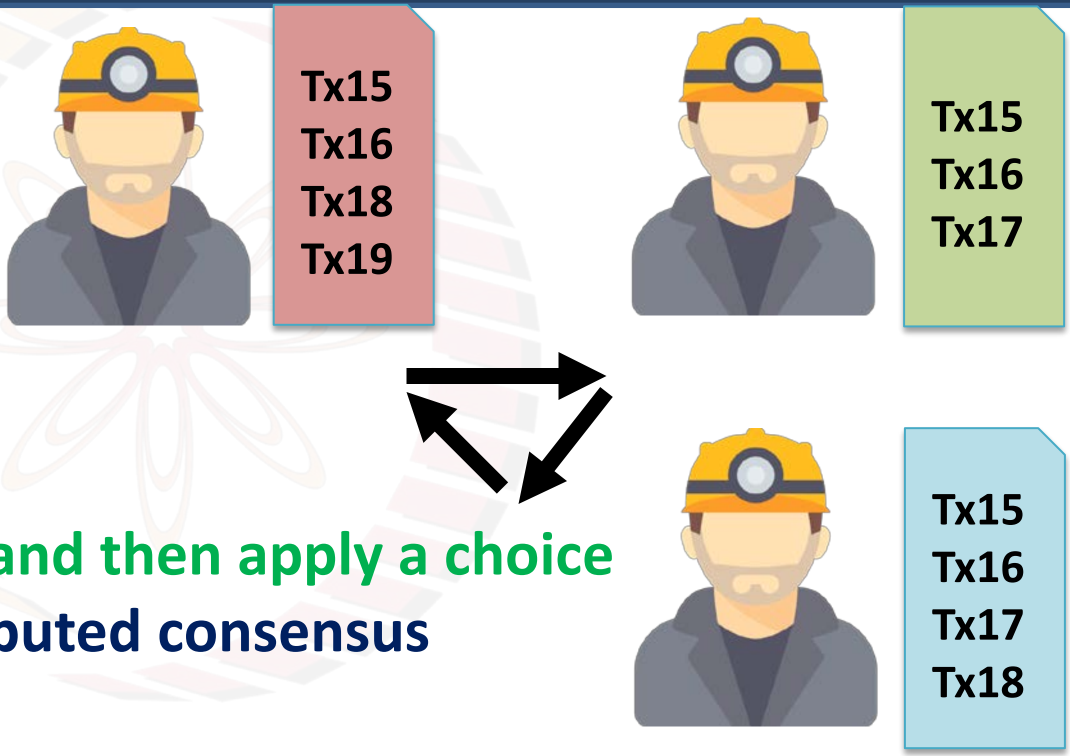
Consensus in Bitcoin



Tx1
Tx2
Tx3
Tx4
Tx5

Tx6
Tx7
Tx8
Tx9
Tx10

Tx11
Tx12
Tx13
Tx14



Tx15
Tx16
Tx18
Tx19

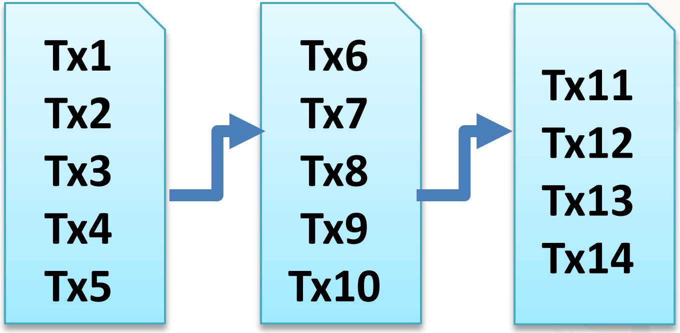
Tx15
Tx16
Tx17

Tx15
Tx16
Tx17
Tx18

Possible Solution:

Broadcast the information and then apply a choice function – traditional distributed consensus algorithms

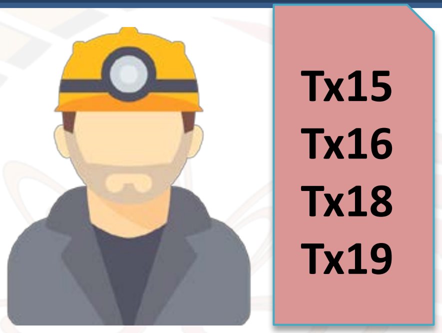
Consensus in Bitcoin



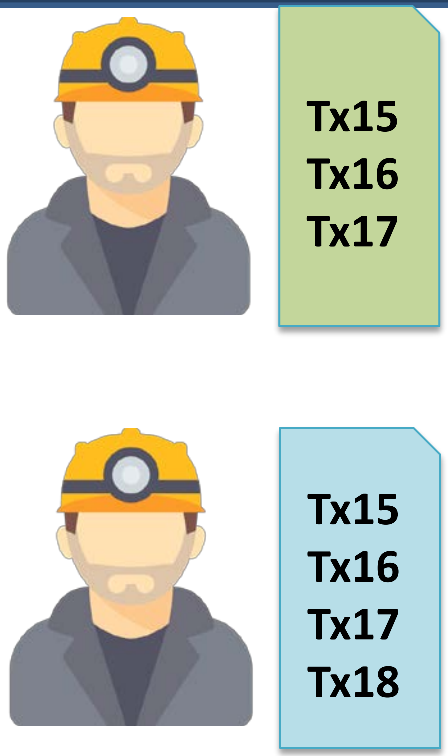
Tx1
Tx2
Tx3
Tx4
Tx5

Tx6
Tx7
Tx8
Tx9
Tx10

Tx11
Tx12
Tx13
Tx14



Tx15
Tx16
Tx18
Tx19



Tx15
Tx16
Tx17

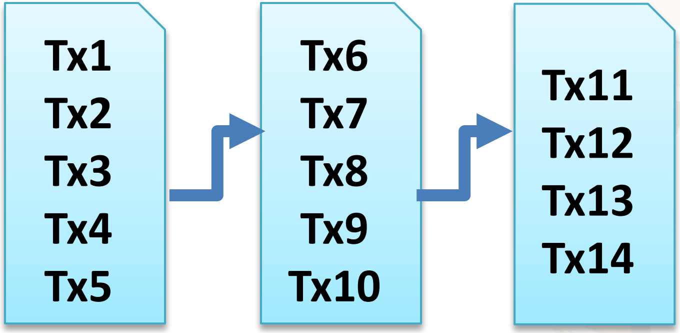
Tx15
Tx16
Tx17
Tx18

May not be Feasible:

You do not have a global clock! How much time will you wait to hear the transactions

Remember the impossibility result

Consensus in Bitcoin



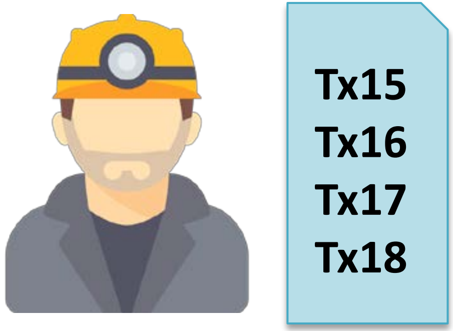
Tx1
Tx2
Tx3
Tx4
Tx5

Tx6
Tx7
Tx8
Tx9
Tx10

Tx11
Tx12
Tx13
Tx14

Tx15
Tx16
Tx18
Tx19

Tx15
Tx16
Tx17

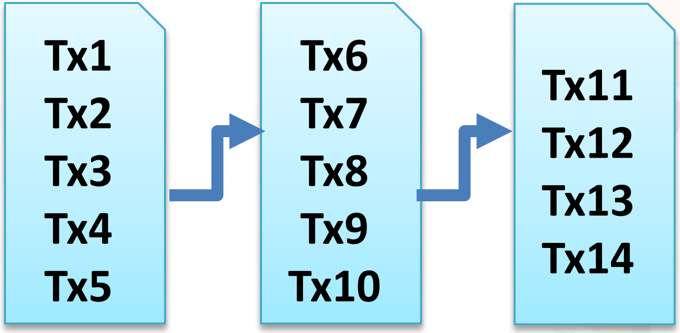


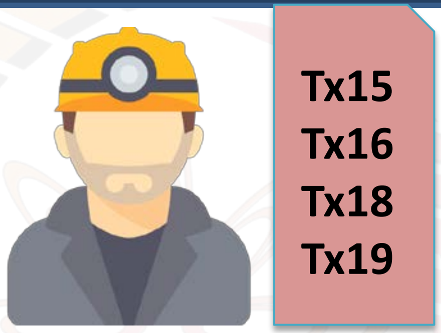
Tx15
Tx16
Tx17
Tx18

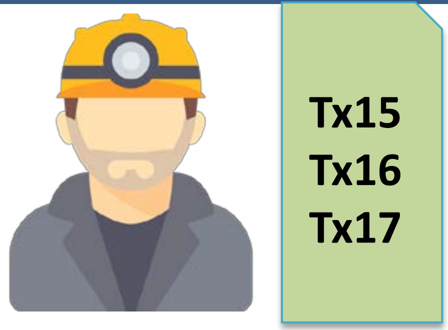
Observation - 1:

- Any valid block (a block with all valid transactions) can be accepted, even if it is proposed by only one miner

Consensus in Bitcoin

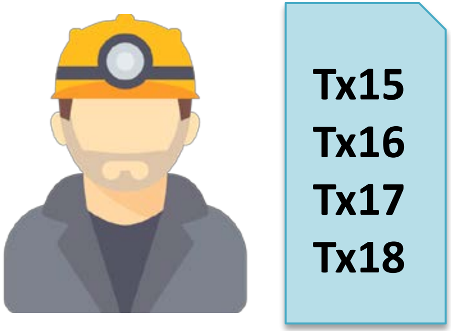




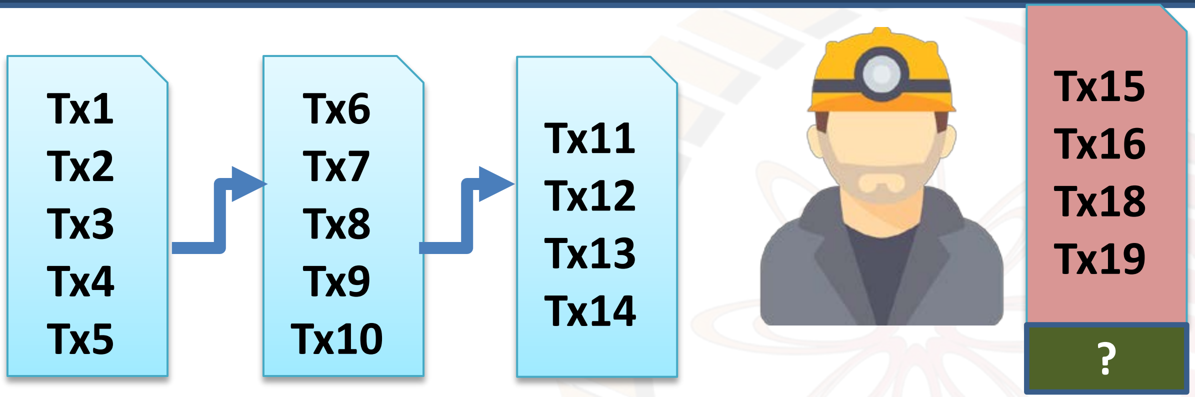


Observation - 2:

- The protocol can work in rounds
  - Broadcast the accepted block to the peers
  - Collect the next set of transactions



Consensus in Bitcoin



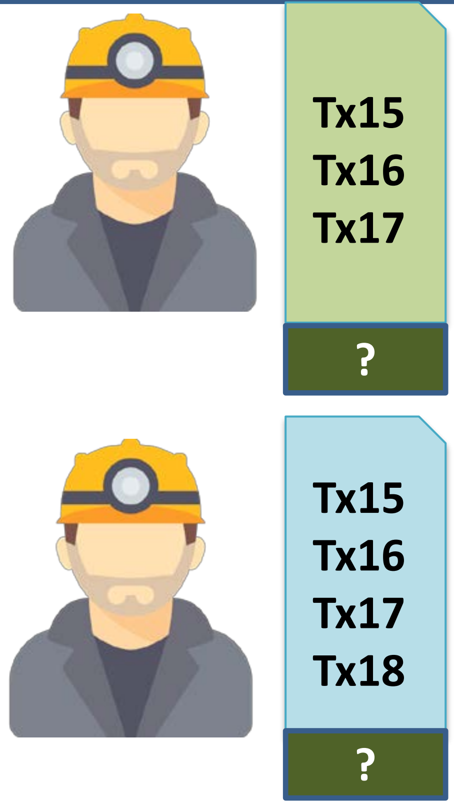
Tx1 Tx2 Tx3 Tx4 Tx5

Tx6 Tx7 Tx8 Tx9 Tx10

Tx11 Tx12 Tx13 Tx14

Tx15 Tx16 Tx18 Tx19

?



Tx15 Tx16 Tx17

?

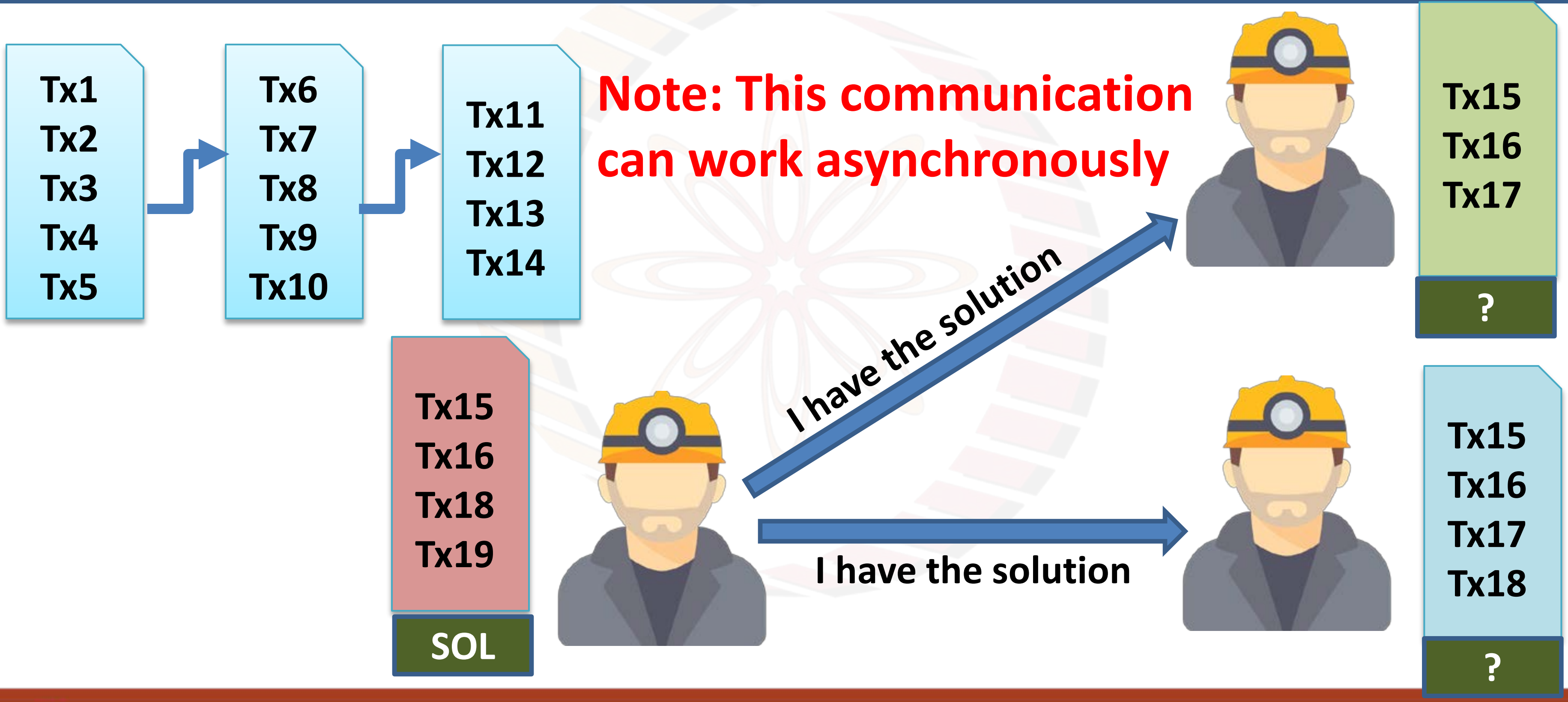
Tx15 Tx16 Tx17 Tx18

?

Solution:

- Every miner independently tries to solve a challenge
- The block is accepted for the miner who can **prove first** that the challenge has been solved

Consensus in Bitcoin



Tx1
Tx2
Tx3
Tx4
Tx5

Tx6
Tx7
Tx8
Tx9
Tx10

Tx11
Tx12
Tx13
Tx14

Note: This communication can work asynchronously

Tx15
Tx16
Tx18
Tx19

SOL

I have the solution

I have the solution

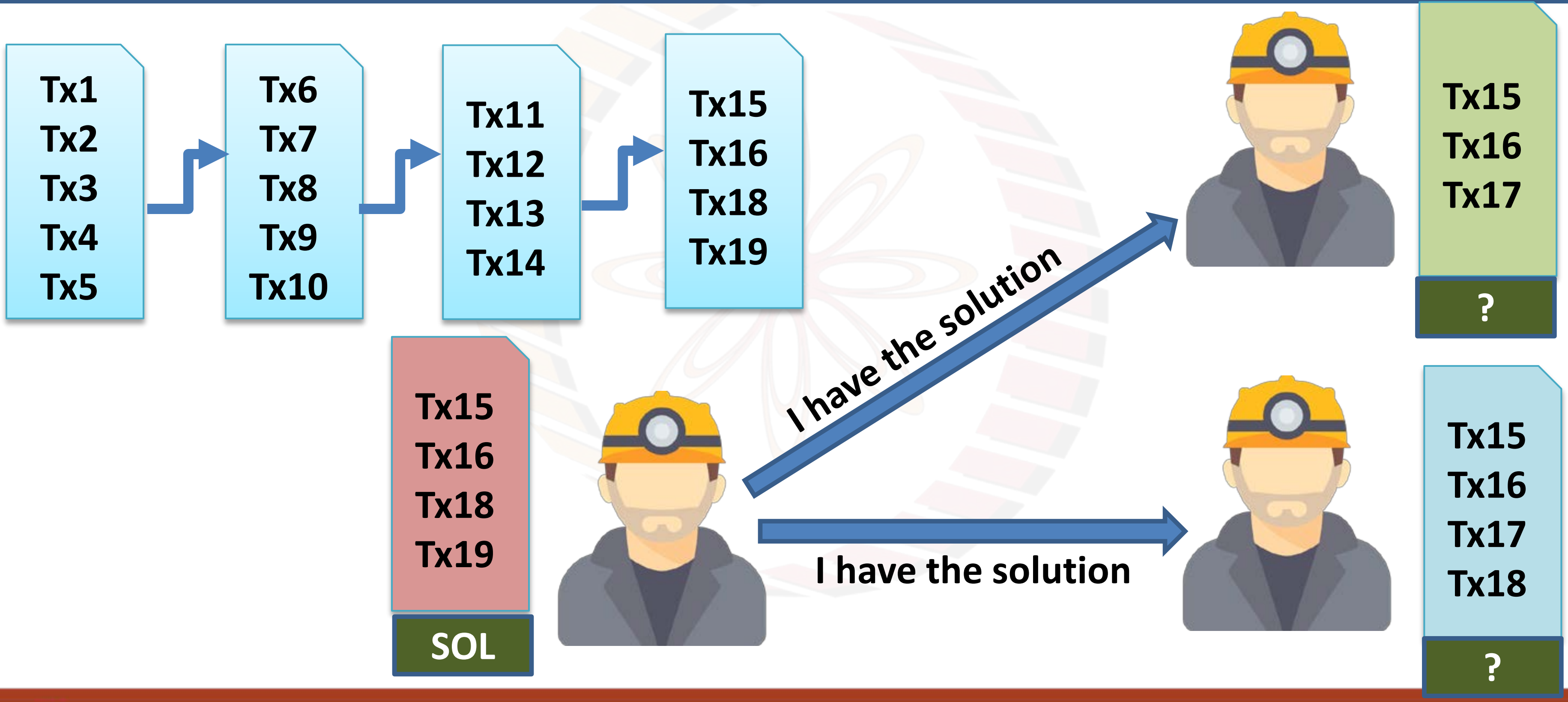
Tx15
Tx16
Tx17

?

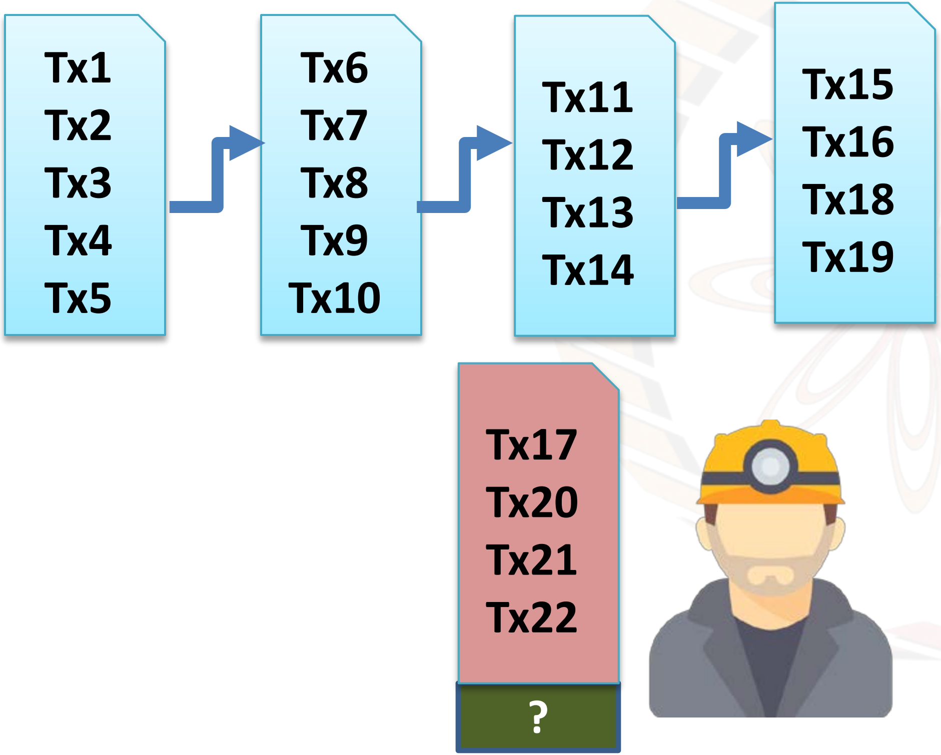
Tx15
Tx16
Tx17
Tx18

?

Consensus in Bitcoin

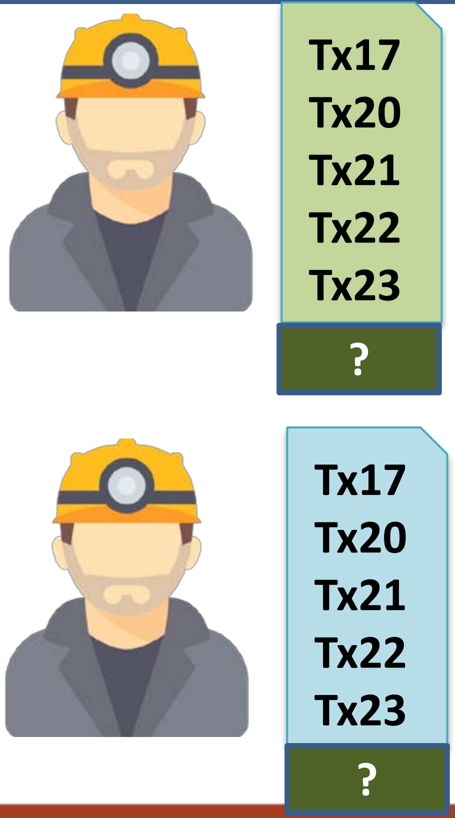


Consensus in Bitcoin



Note: Everyone can see that Tx18 and Tx19 have been committed, but Tx17 has not been committed.

Include that in the next round



Proof of Work (Pow)

- An economic measure to deter service abuses by requiring some work from the service requester (usually processing time by a computer)

- The idea came from Dwork and Naor (1992), to combat junk emails
  - You have to do some work to send a valid email
  - The attacker would be discouraged to send junk emails

Dwork, Cynthia; Naor, Moni (1993). "Pricing via Processing, Or, Combatting Junk Mail, Advances in Cryptology". ***CRYPTO'92: Lecture Notes in Computer Science No. 740*****. Springer: 139–147.**

Proof of Work (PoW) Features

- Asymmetry
  - The work must be moderately hard, but feasible for the service requester
  - The work must be easy check for the service provider

- Service requesters will get discouraged to forge the work, but service providers can easily check the validity of the work

Cryptographic Hash as the PoW

- Use the **puzzle friendliness** property of cryptographic hash function as the work
  - Given X and Y, find out k, such that $Y = Hash(X||k)$
  - It is difficult (but not infeasible) to find such k
  - However, once you have a k, you can easily verify the challenge

- Used in **Hashcash**, a proof of work that can be added with an email as a "*good-will*" token

Adam Back, "Hashcash - A Denial of Service Counter-Measure", technical report, August 2002

Hashcash PoW

- A textual encoding of a hashcash stamp is included in an email header
  - Proof that the sender has expended a modest amount of CPU time calculating the stamp before sending the email
  - It is unlikely that the sender is a spammer

- The receiver can verify the hashcash stamp very easily

- Any change in the header requires a change in the hashcash
  - Brute force is the only way to find a hashcash

Hashcash PoW

- The hashcash is included in the email header, looks like this

```
X-Hashcash:
1:20:180401:sandipc@cse.iitkgp.ac.in::0000000267674
b591257b87:6078
```

- Version: number of zero bits in the hashed code: date: resource: optional extension: string of random characters: counter

Hashcash PoW – Sender Side

- Construct the header

```
1:20:180401:sandipc@cse.iitkgp.ac.in::<hash>:<counter>
```

- The sender initializes the counter value to a random number

- Compute 160 bit SHA-1 hash of the header.
  - If the first 20 bit of the hash are all zeros, then it is accepted
  - Else try with a different counter

Hashcash PoW – Recipient Side

- Recipient checks
  - The date – should be within two days
  - Email address
  - The random string – should not be used repeatedly within a certain duration (prevent replay)

- Compute the 160 bit SHA-1 hash of the entire received string

```
1:20:180401:sandipc@cse.iitkgp.ac.in::0000000267674b591257b87:6078
```

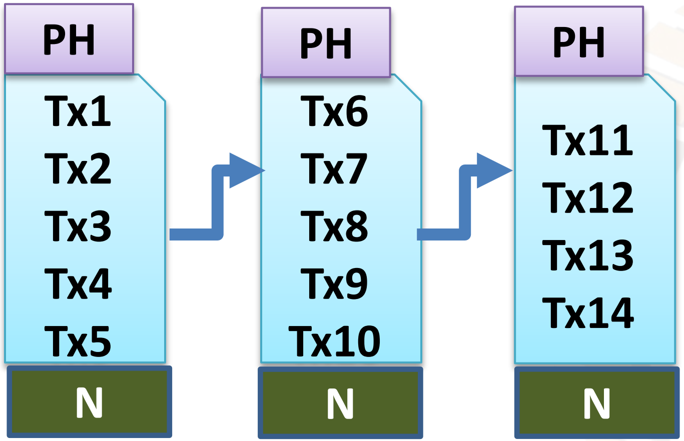
  - If the first 20 bits are not zero then it is invalid

Hashcash PoW

- On average, the sender will have to try 2^{20} hash values to find a valid header (takes about a few seconds in a general purpose computer)
  - There are 2^{160} possible hash values
  - 20 zero bits at the beginning – 2^{140} possible hash values that satisfy this criteria
  - Chance of randomly selecting a header with 20 zero bits at the prefix is 1 in 2^{20}

- The recipient requires around 2 microsecond to validate

Bitcoin Proof of Work System

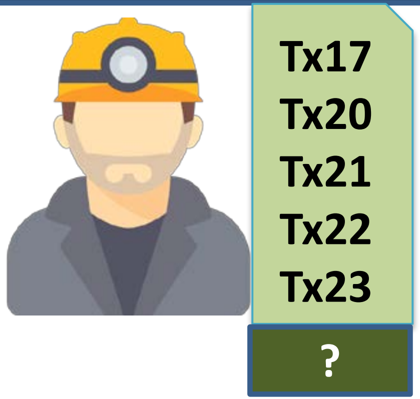


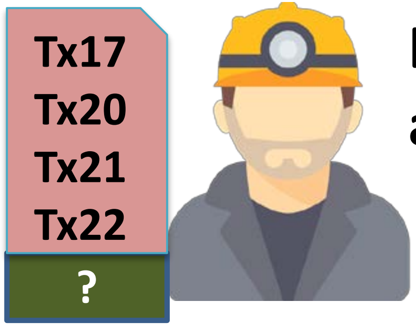
BH: Block Hash

PH: Previous Block Hash

MR: Merkle Root

N: Nonce

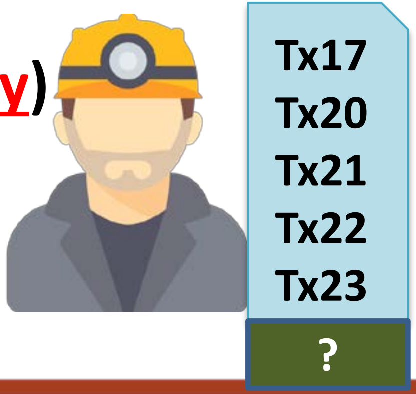




BH should have certain zeros (<u>difficulty</u>) at the beginning

BH = Hash(PH:MR:N)

N = ?



Bitcoin Proof of Work (PoW) System

- Most implementations of Bitcoin PoW use double SHA256 hash function
- The miners collect the transactions for 10 minutes (default setup) and starts mining the PoW
- The probability of getting a PoW is low – it is difficult to say which miner will be able to generate the block
  - No miner will be able to control the bitcoin network single handedly

Explore ...

- http://www.hashcash.org/
  - Download the source and try with different numbers of zero bit targets
  - Increase the number of targeted zero bits at the hash prefix, say from 20 to 2020, at a step of 100, and observe the time to compute the hashcash
  - Use **`sha1sum`** (in Linux) to compute the SHA-1 checksum of the obtained hashcash values from the above experiment. How much time do you require to validate a hashcash?

thank you!