

# Mechanics of Bitcoin

Amin Sadeghi

# Bitcoin Mining

# Mining Bitcoins in 6 easy steps

1. Join the network, listen for transactions

a. Validate all proposed transactions

2. Listen for new blocks, maintain block chain

a. When a new block is proposed, validate it

3. Assemble a new valid block

4. Find the nonce to make your block valid

5. Hope everybody accepts your new block

6. Profit!

Useful  
to  
Bitcoin  
networ

Incentivi  
ze  
miners  
to do  
above

# Mining difficulty “target” end of 2017

Hash of any valid block should be less than the following:

256 bit hash output

0000000000000000000a95500

64+ leading zeroes required

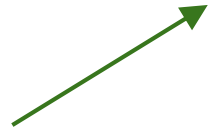
Current difficulty =  $2^{68} = 84,758,978,290,086,040,000$

Less than 1 in about  $2^{68}$  nonces will work

# Setting the mining difficulty

**Every two weeks (or approx. 2,016 blocks), compute:**

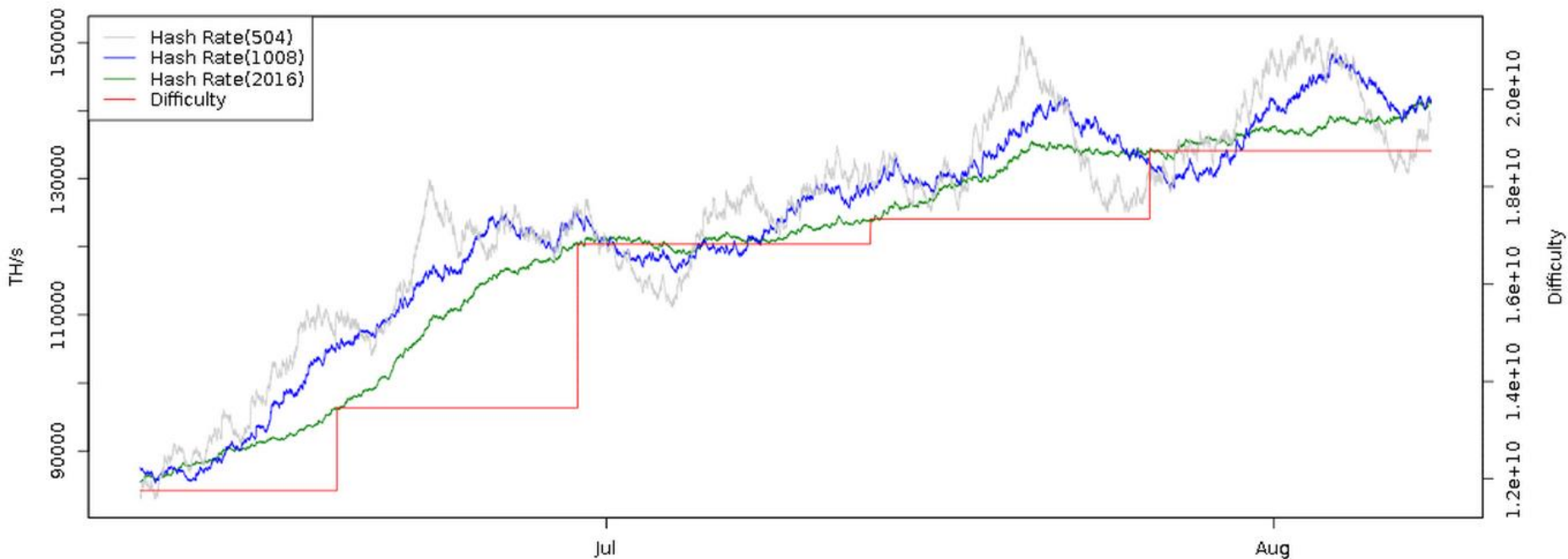
```
next_difficulty= previous_difficulty *  
                  (2 weeks) / (time to mine last 2016 blocks)
```



Expected number of blocks in 2 weeks at 10  
minutes/block

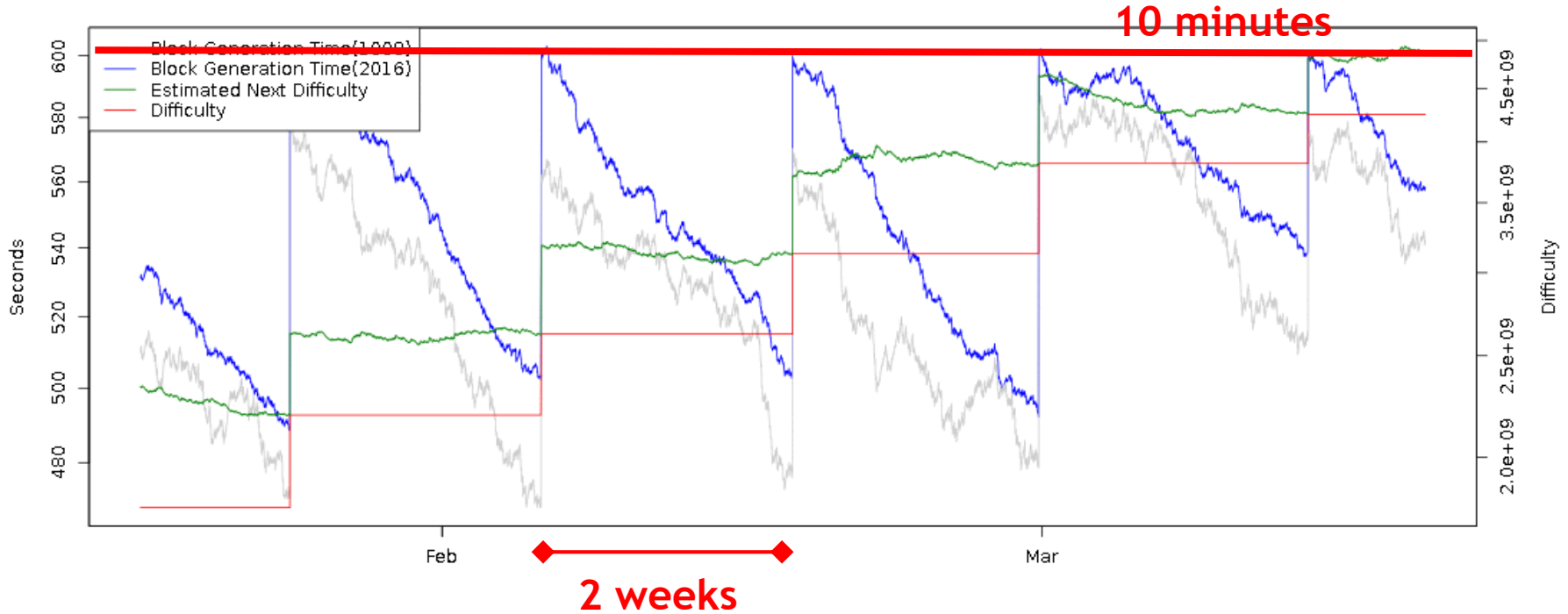
# Mining difficulty over time

Bitcoin Hash Rate vs Difficulty (2 Months)



# Time to find a block

Bitcoin Block Generation Time vs Difficulty



# Mining hardware

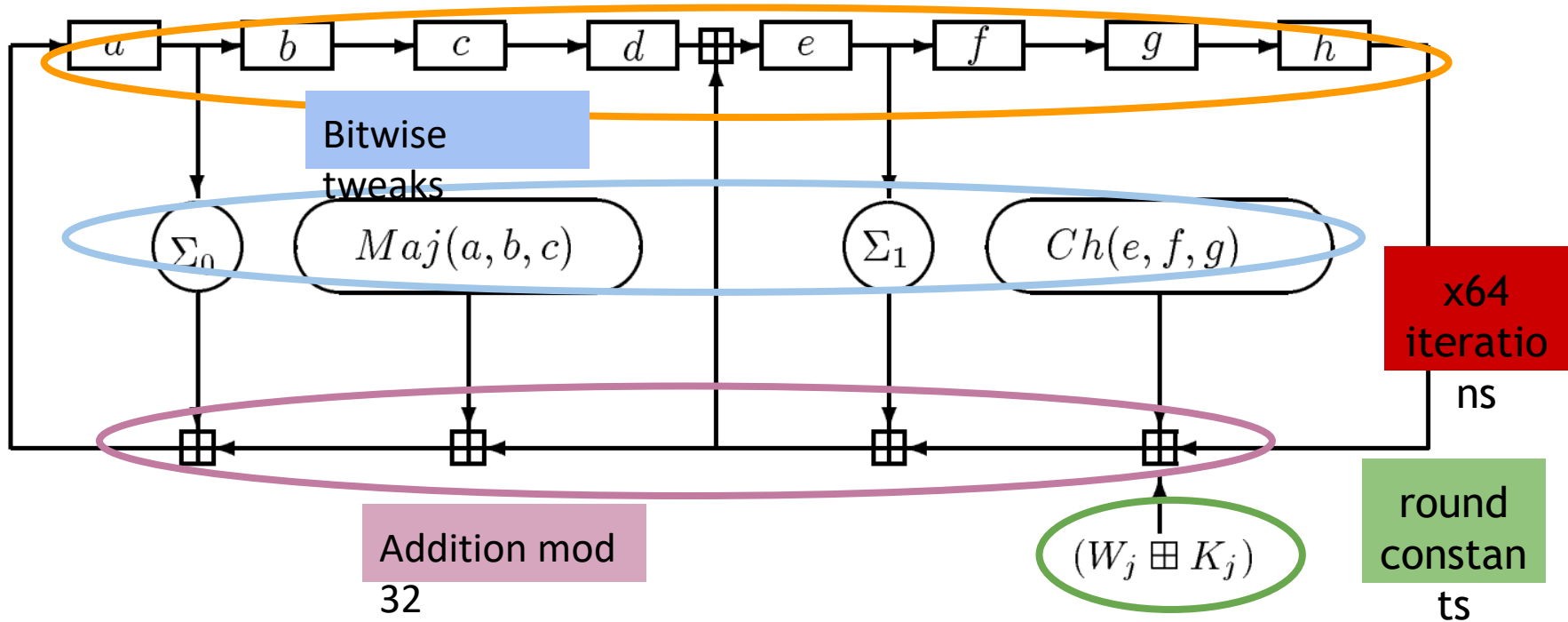


# SHA-256

- General purpose hash function
  - Part of SHA-2 family: SHA-224,SHA-384,SHA-512
- Published in 2001
- Designed by the NSA
- Remains unbroken cryptographically
  - Weaknesses known
- SHA-3 (replacement) under standardization

# SHA-256 in more depth

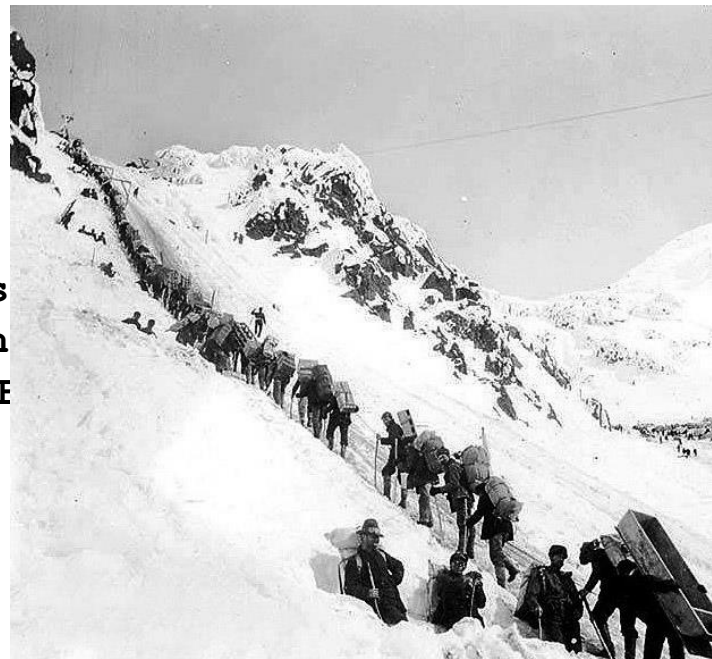
256-bit state



# CPU mining

```
TARGET = (65535 << 208)/DIFFICULTY;  
coinbase_nonce = 0;  
while (1){  
    header = makeBlockHeader(trans  
for (header_nonce=0;header_non  
    if (SHA256(SHA256(makeF  
TARGET)  
        break; //block  
    }  
    coinbase_nonce++;  
}
```

↑  
two hashes



Throughput on a high-end PC  $\approx$  **20 million hashes/sec**  
 $\approx$  **139,461** years to find a block today!

# GPU mining

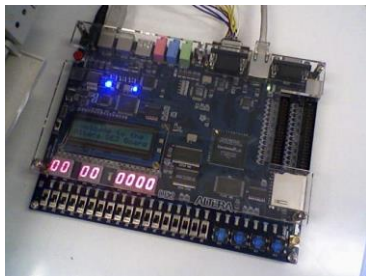


- GPUs designed for high-performance graphics
  - High parallelism
  - High throughput
- First used for Bitcoin mining in October 2010
- Implemented in OpenCL
  - OpenCL: General purpose high level language for implementing non-graphic applications on GPUs



Source:  
LeonardH,  
cryptocurren  
ciestalk.com

# FPGA mining



- **Field Programmable Gate Area**
- First used for Bitcoin mining approx. June 2011
- Implemented in Verilog





Bob Buskirk, thinkcomputers.org

# Bitcoin Application-specific Integrated Circuits (ASICs)

## TerraMiner™ IV – 2TH/s Networked ASIC Miner

\$5,999

Shipping June 2014



## 300 GH Bitcoin Mining Card

The Monarch BPU 300 C

\$1,497.00

Qty:

[ADD TO CART](#)



### DETAILS :

- 2,5 TH/s
- Dimensions:  
15" x 13.3" x 13.7"  
(38cm x 34cm x 35cm)
- 28nm ASIC technology
- Silent Cooling
- In-built WiFi Connection  
(without Antenna)
- Less than 750 watt (0.3 per GH)
- 1 Year Guarantee
  
- \$ 5.800

### COMES WITH :

1. Power Supply
2. Free Remote Power Outlet & Smartphone App
3. Free User Guide
4. Free Personal Assistance for Setup

### SHIPPING :

- Worldwide, Express
- Included in the price
- Available:  
100 Units: Shipping April  
(Week 3)



**Pre-Order Terms:** This is a pre-order. 28nm ASIC bitcoin mining hardware products are shipped according to placement in the order queue, and delivery may take 3 months or more after order. All sales are final.



# Professional mining centers

## Needs:

- Cheap power
- Good network
- Cool climate
  - (Georgia and Iceland are popular destinations for bitcoin mining)



BitFury mining center, Republic of

# Evolution of mining



CPU



GPU



FPGA



ASIC



Gold pan



Sluice box



Placer mining



Pit mining

# Energy consumption & ecology

# Energy aspects of Bitcoin mining

- **Embodied energy:** used to manufacture mining chips & other equipment
  - Should decrease over time
  - Returns to scale
- **Electricity:** used to perform computation
  - Should increase over time
  - Returns to scale
- **Cooling:** required to protect equipment
  - Costs more with increased scale!

# Is Bitcoin mining wasteful?

- All payment systems require energy!



- Although, it may be nice to have a currency with a less energy-intensive puzzle, but the same level of security!

# Data furnaces

- **Observation:** in the limit, computing devices produce heat almost as well as electric heaters!
- Why not install mining rigs as home heaters?
- Challenges:
  - Ownership/maintenance model
  - Gas heaters still at least 10x more efficient
  - What happens in summer?



# Open questions

- Will Bitcoin drive out electricity subsidies?
- Will Bitcoin require guarding power outlets?
- Can we make a currency with no proof-of-work?



Stay tuned for our lecture on alt-mining!

# Mining pools



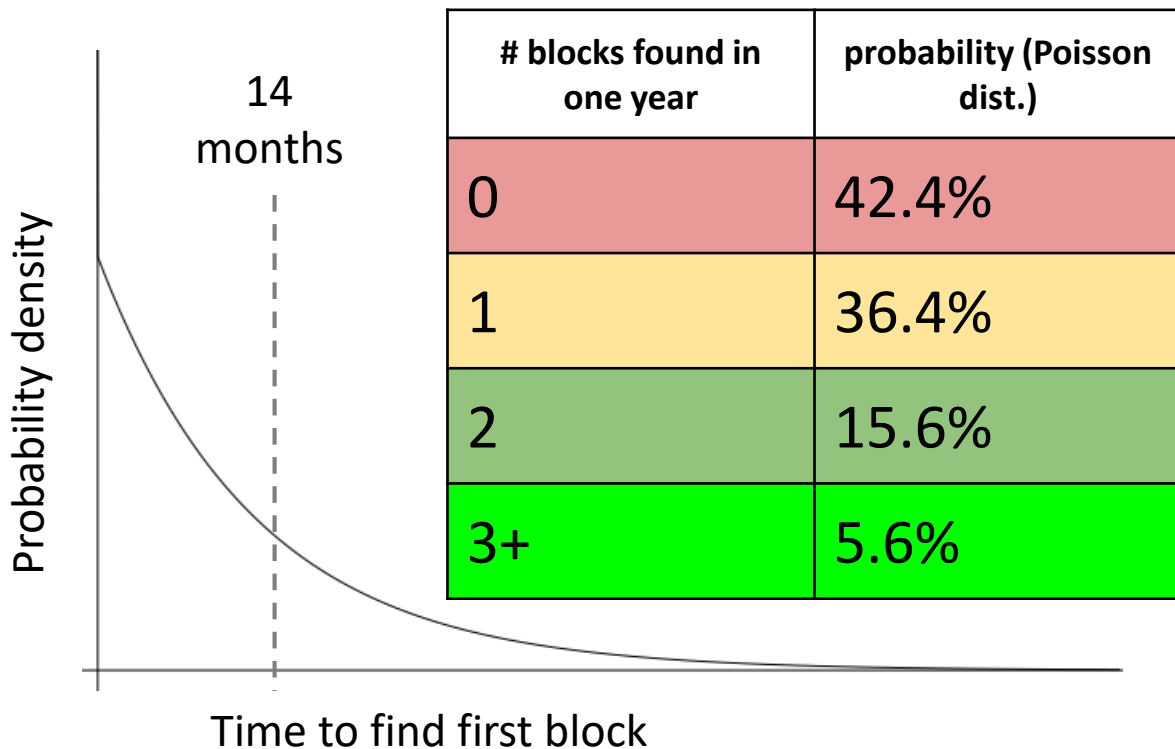
# Economics of being a small miner



- Cost: ≈US\$6,000
- Expected time to find a block: ≈14 months
- Expected revenue: ≈\$1,000/month

**TerraMiner IV**

# Mining uncertainty



# Mining pools

- **Goal:** pool participants all attempt to mine a block with the same coinbase recipient
  - Send money to key owned by pool manager
- Distribute revenues to members based on how much work they have performed
  - Minus a cut for pool manager

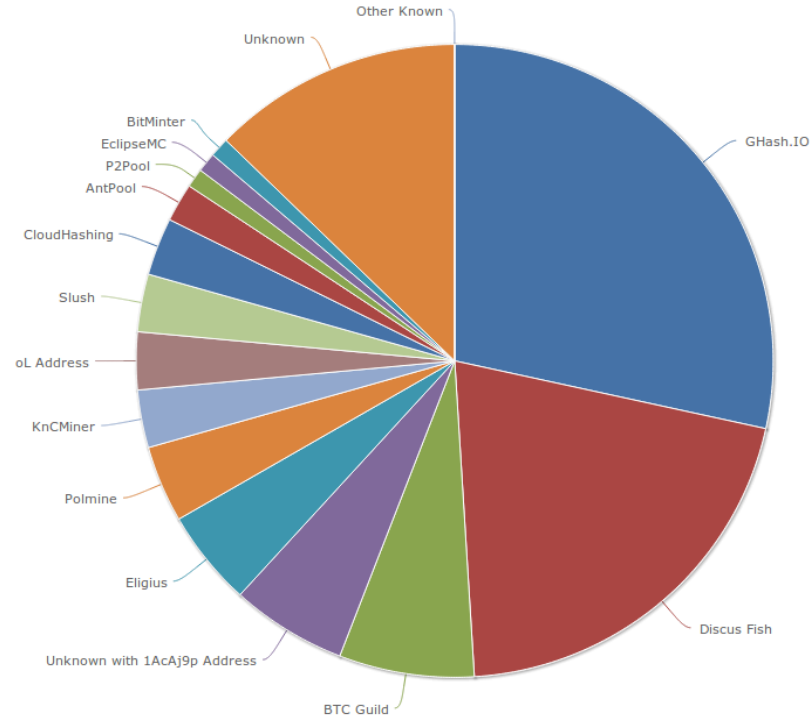
**How do we know how much work members perform?**

# Mining shares

**Idea:** prove work with “near-valid blocks” (shares)

```
4AA087F0A52ED2093FA816E53B9B6317F9B8C1227A61F9481AFED67301F2E3FB
D3E51477DCAB108750A5BC9093F6510759CC880BB171A5B77FB4A34ACA27DEDD
00000000008534FF68B98935D090DF5669E3403BD16F1CDFD41CF17D6B474255
BB34ECA3DBB52EFF4B104EBBC0974841EF2F3A59EBBC4474A12F9F595EB81F4B
00000000002F891C1E232F687E41515637F7699EA0F462C2564233FE082BB0AF
0090488133779E7E98177AF1C765CF02D01AB4848DF555533B6C4CFCA201CBA1
460BEFA43B7083E502D36D9D08D64AFB99A100B3B80D4EA4F7B38E18174A0BFB
000000000000000078FB7E1F7E2E4854B8BC71412197EB1448911FA77BAE808A
652F374601D149AC47E01E7776138456181FA4F9D0EEDD8C4FDE3BEF6B1B7ECE
785526402143A291CFD60DA09CC80DD066BC723FD5FD20F9B50D614313529AF3
000000000041EE593434686000AF77F54CDE839A6CE30957B14EDEC10B15C9E5
9C20B06B01A0136F192BD48E0F372A4B9E6BA6ABC36F02FCED22FD9780026A8F
```


# Mining pools (as of August 2014)



# Are mining pools a good thing?

- Pros
  - Make mining more predictable
  - Allow small miners to participate
  - More miners using updated validation software
- Cons
  - Lead to centralization
  - Discourage miners from running full nodes

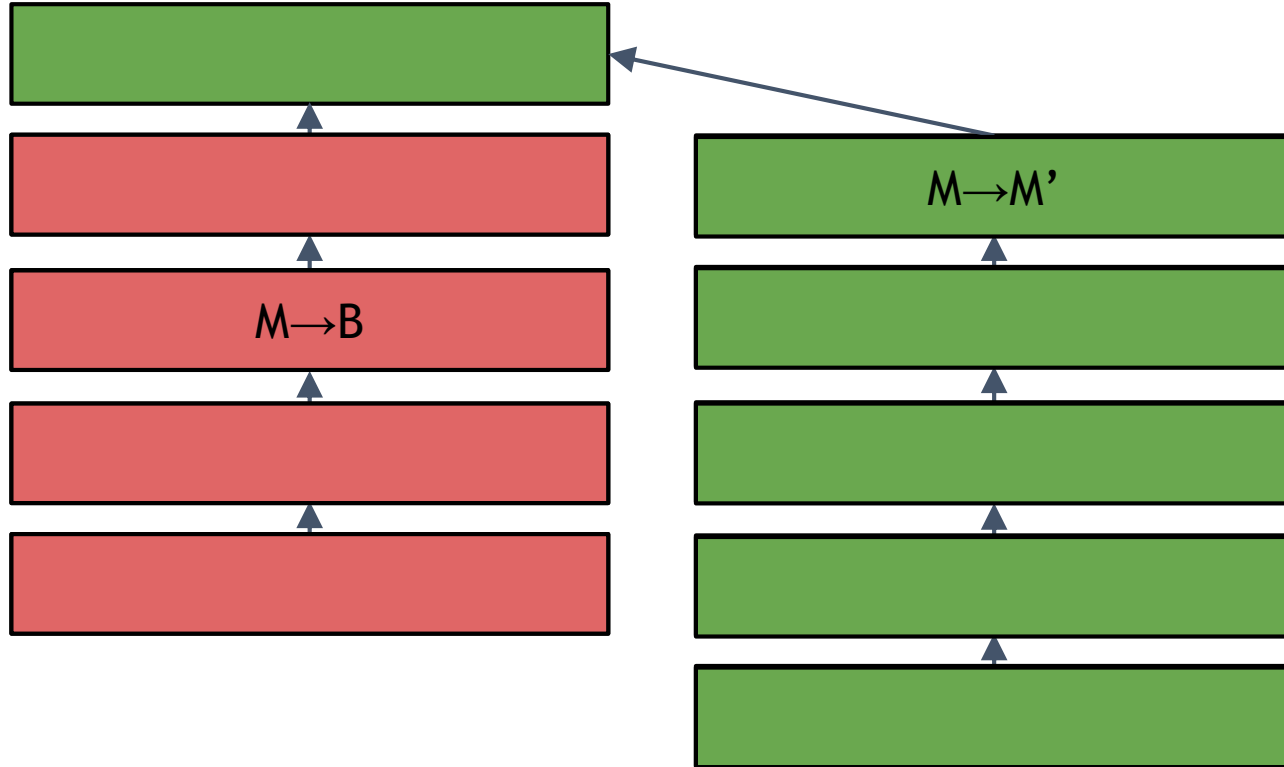
Can we prevent pools?



Stay tuned for our lecture on alt-mining!

# Mining incentives and strategies

# Forking attacks





# Forking attacks

- Certainly possible if  $\alpha > 0.5$ 
  - may be possible with less
  - avoid block collisions
- Attack is detectable
- Might be reversed
- Might crash exchange rate



*Goldfinger  
Attack?*

# Forking attacks via bribery

- **Idea:** building  $\alpha > 0.5$  is expensive. Why not rent it instead?
- Payment techniques:
  - Out-of-band bribery
  - Run a mining pool at a loss
  - Insert large “tips” in the block chain

This is an open problem!

# Punitive forking

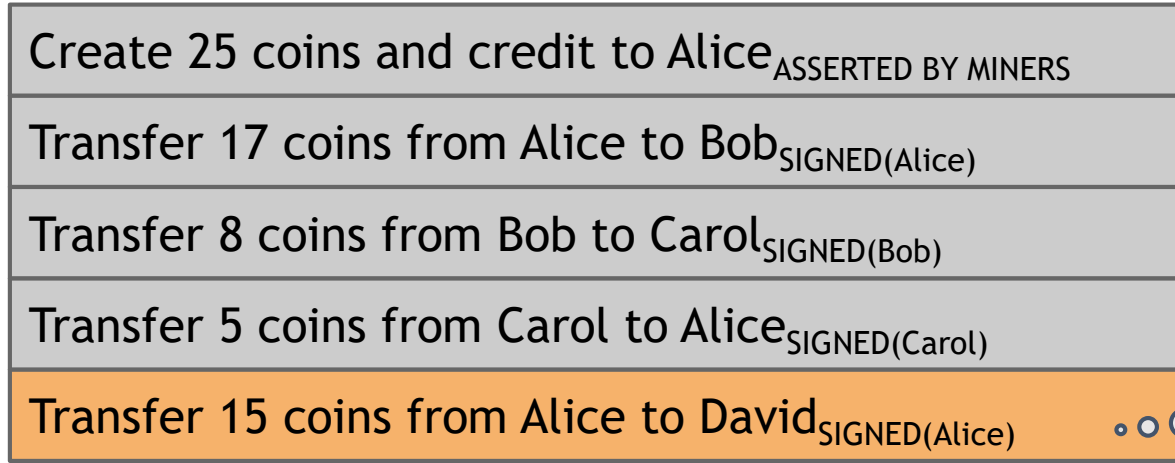
- Suppose you want to blacklist transactions from address  $X$ 
  - Freeze an individual's money forever
- **Extreme strategy:** announce that you will refuse to mine on any chain with a transaction from  $X$

With  $\alpha < 0.5$ , you'll soon fall behind the network

# Bitcoin Ledger

# An account-based ledger (*not* Bitcoin)

time

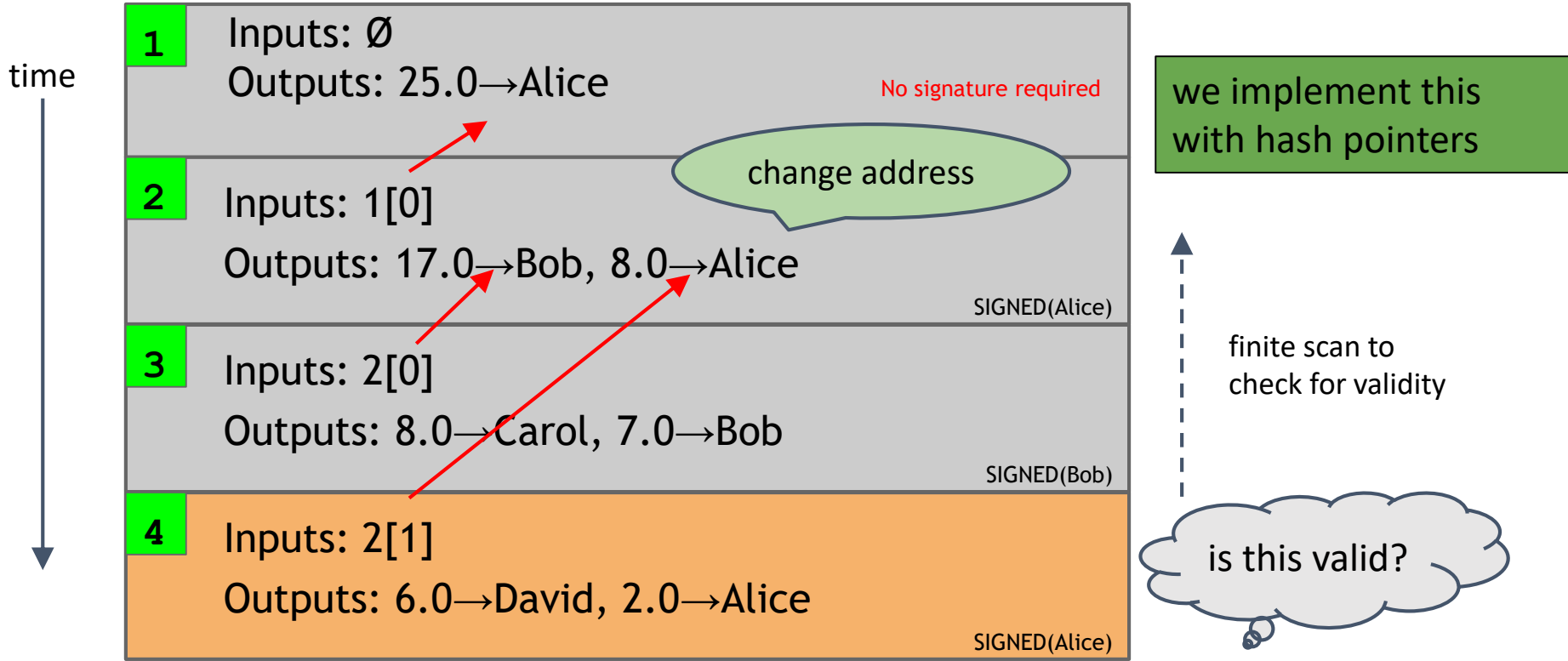


might need to scan backwards until genesis!

is this valid?

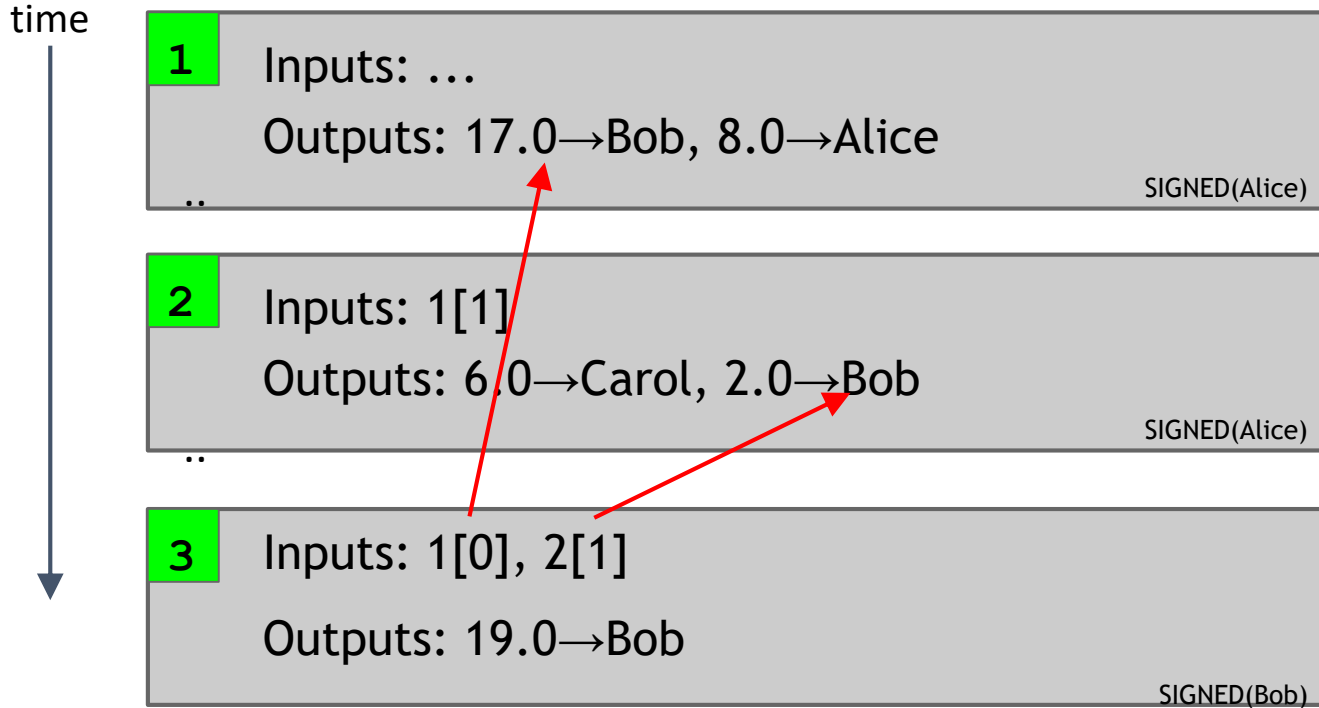
SIMPLIFICATION: only one transaction per block

# A transaction-based ledger (Bitcoin)



SIMPLIFICATION: only one transaction per block

# Merging value



SIMPLIFICATION: only one transaction per block

# Joint payments

time



**1** Inputs: ...  
Outputs: 17.0→Bob, 8.0→Alice  
.. SIGNED(Alice)

**2** Inputs: 1[1]  
Outputs: 6.0→Carol, 2.0→Bob  
.. SIGNED(Alice)

**3** Inputs: 2[0], 2[1]  
Outputs: 8.0→David  
SIGNED(Carol), SIGNED(Bob)

two signatures!

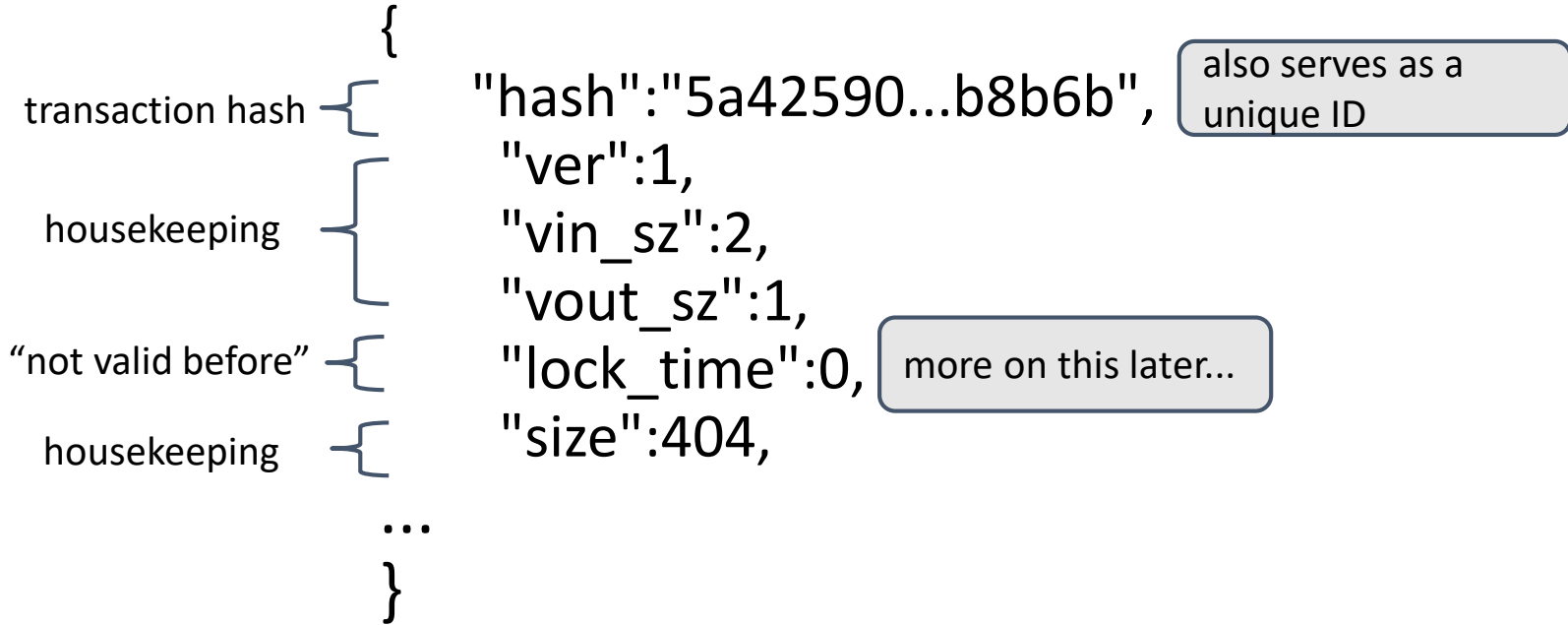
SIMPLIFICATION: only one transaction per block



# The real deal: a Bitcoin transaction



# The real deal: transaction metadata



# The real deal: transaction inputs

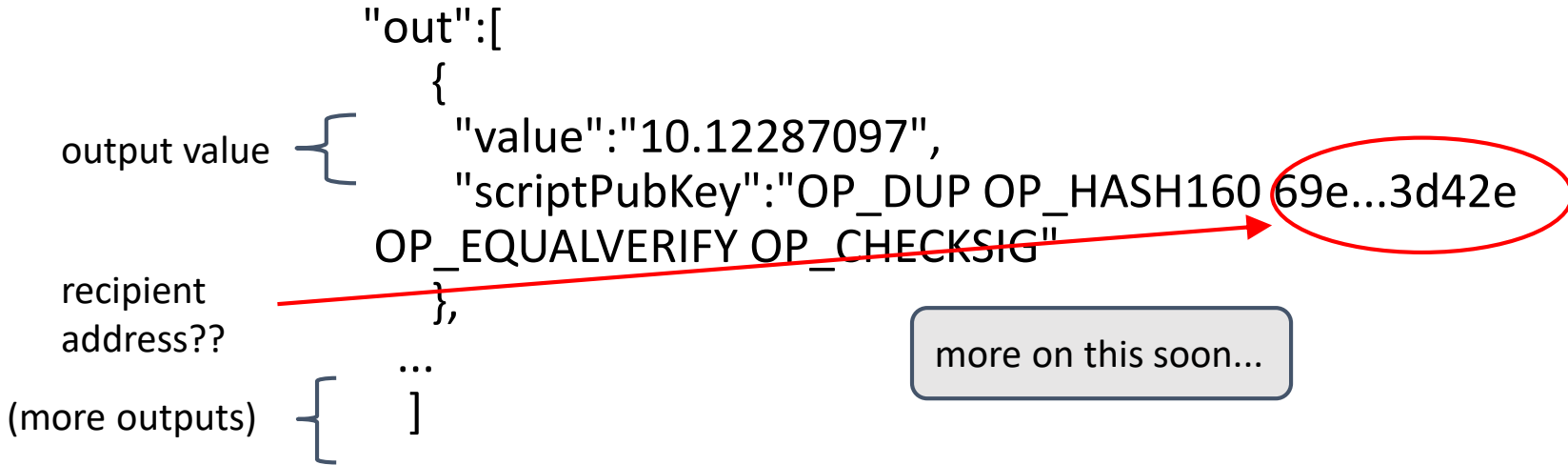
```
    "in":[
      {
        "prev_out":{
          "hash":"3be4...80260",
          "n":0
        },
        "scriptSig":"30440....3f3a4ce81"
      },
      ...
    ],
```

previous transaction {

signature {

(more inputs) {

# The real deal: transaction outputs



**Sum of all output values less than or equal to sum of all input values!**  
If sum of all output values less than sum of all input values, then difference goes to miner as a transaction fee

# Bitcoin scripts

Output “addresses” are really *scripts*

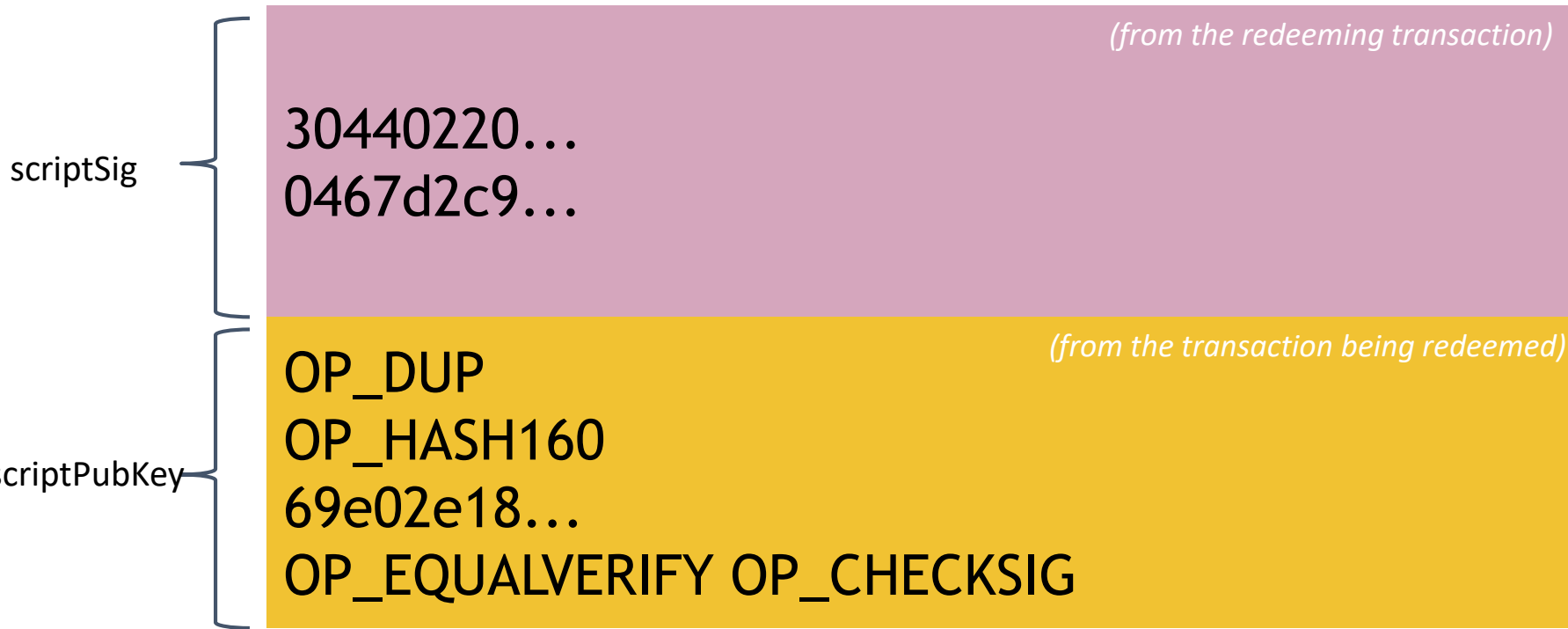
OP\_DUP

OP\_HASH160

69e02e18...

OP\_EQUALVERIFY OP\_CHECKSIG

# Input “addresses” are *also* scripts



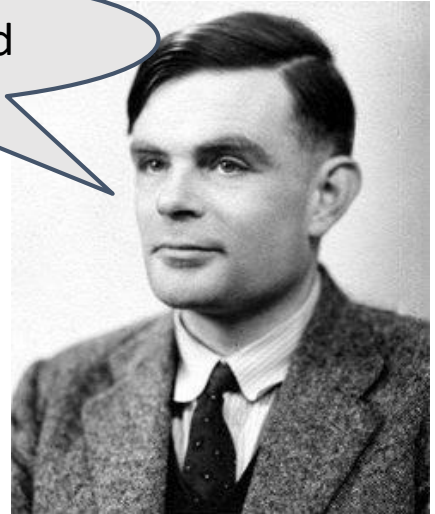
TO VERIFY: Concatenated script must execute completely with no errors

# Bitcoin scripting language (“Script”)

## Design goals

- Built for Bitcoin (inspired by Forth)
- Simple, compact
- Support for cryptography
- Stack-based (linear)
- Limits on time/memory
- No looping
  - **Result:** Bitcoin script is not Turing Complete!  
i.e, cannot compute arbitrarily powerful functions
  - **Advantage:** No infinite looping problem!

I am not impressed





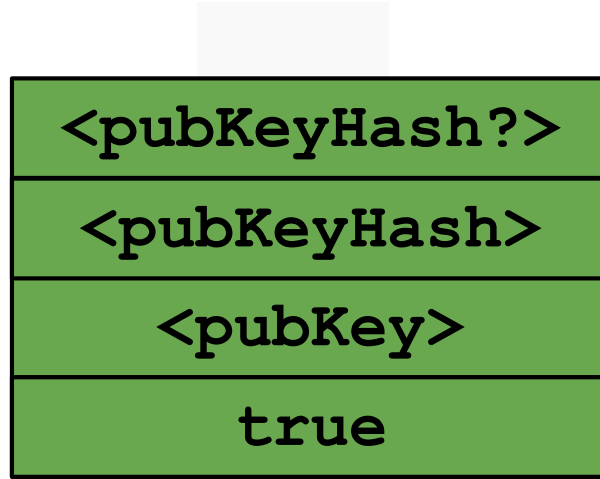
# Bitcoin scripting language (“Script”)

- 256 instructions (each represented by 1 byte)
  - 75 reserved, 15 disabled
  - Basic arithmetic, basic logic (“if” → “then”), throwing errors, returning early, crypto instructions (hash computations, signature verifications), etc.
- Only two possible outcomes of a Bitcoin script
  - Executes successfully with no errors → transaction is valid OR
  - Error while execution → transaction invalid and should not be accepted in the block chain

# Common script instructions

Name	Functions
<b>OP_DUP</b>	Duplicates top item on the stack
<b>OP_HASH160</b>	Hashes twice: first using SHA-256, then using RIPEMD-160
<b>OP_EQUALVERIFY</b>	Returns true if inputs are equal, false (marks transaction invalid) otherwise
<b>OP_CHECKSIG</b>	Checks that the input signature is valid using input public key for the hash of the current transaction
<b>OP_CHECKMULTISIG</b>	Checks that $t$ signatures on the transaction are valid from $t$ ( <i>out of <math>n</math></i> ) of the specified public keys

# Bitcoin script execution example

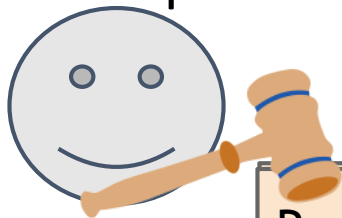


```
<sig> <pubKey> OP_DUP OP_HASH160 <pubKeyHash?> OP_EQUALVERIFY OP_CHECKSIG
```

# Applications of Bitcoin scripts

# Example: Escrow transactions

(disputed case)  
(normal case)



Judy

Pay x to Alice

SIGNED(ALICE, JUDY)



Alice



Bob

Pay x to 2-of-3 of Alice, Bob, Judy (MULTISIG)

SIGNED(ALICE)

Bob doesn't want to ship until after Alice pays.

```
lock_time
{
  "hash":"5a42590...b8b6b",
  "ver":1,
  "vin_sz":2,
  "vout_sz":1,
  "lock_time":315415,
  "size":404,
  ...
}
```

Block index or real-world timestamp before  
which this transaction can't be published

# Bitcoin blocks

# Bitcoin blocks

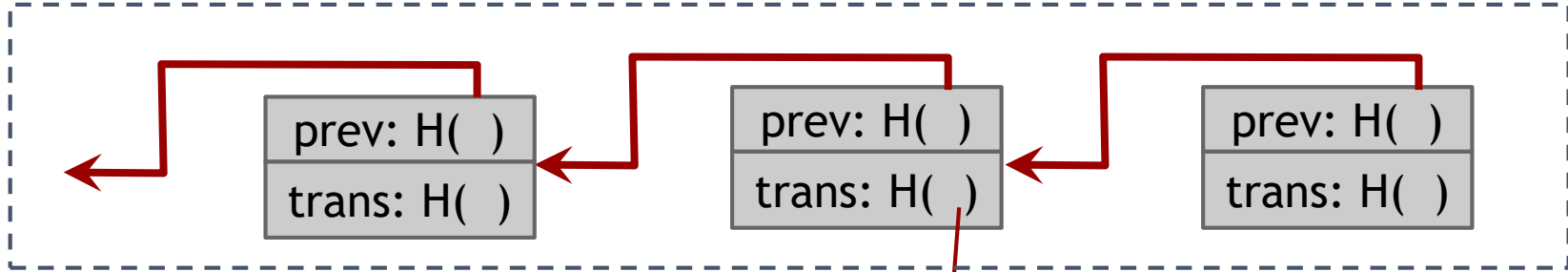
Why bundle transactions together?

- Single unit of work for miners
- Limit length of hash-chain of blocks
  - Faster to verify history

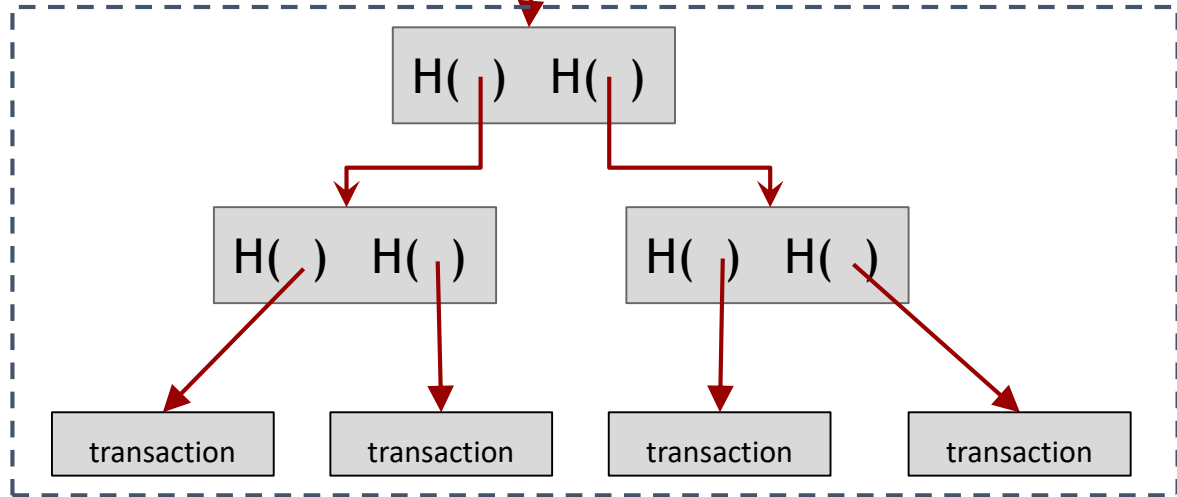


# Bitcoin block structure

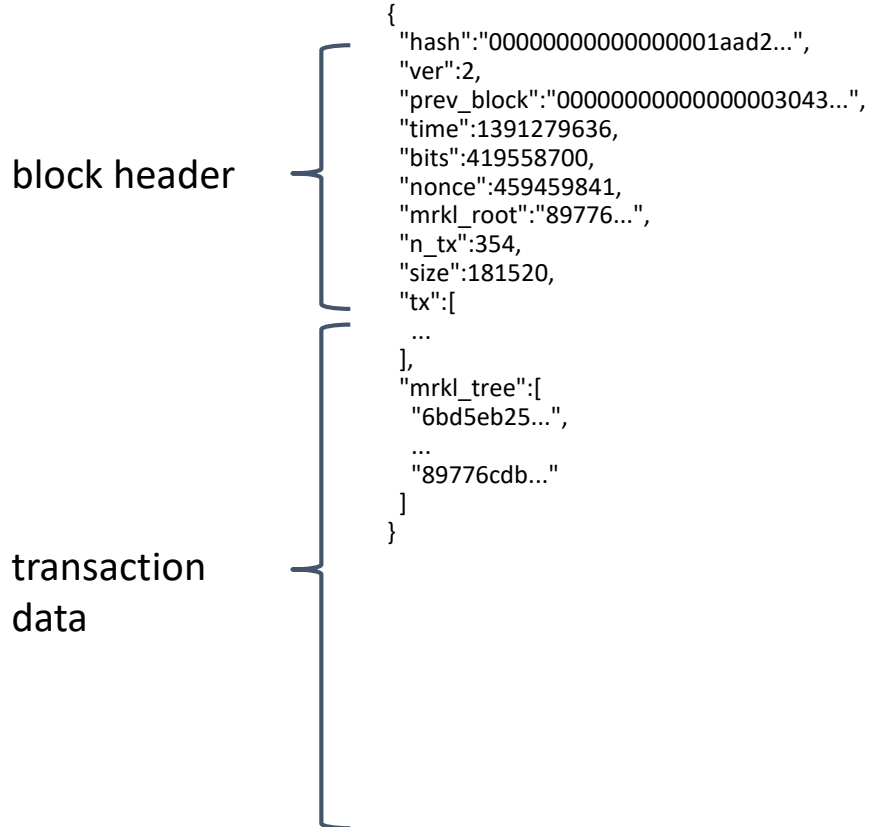
Hash chain of blocks



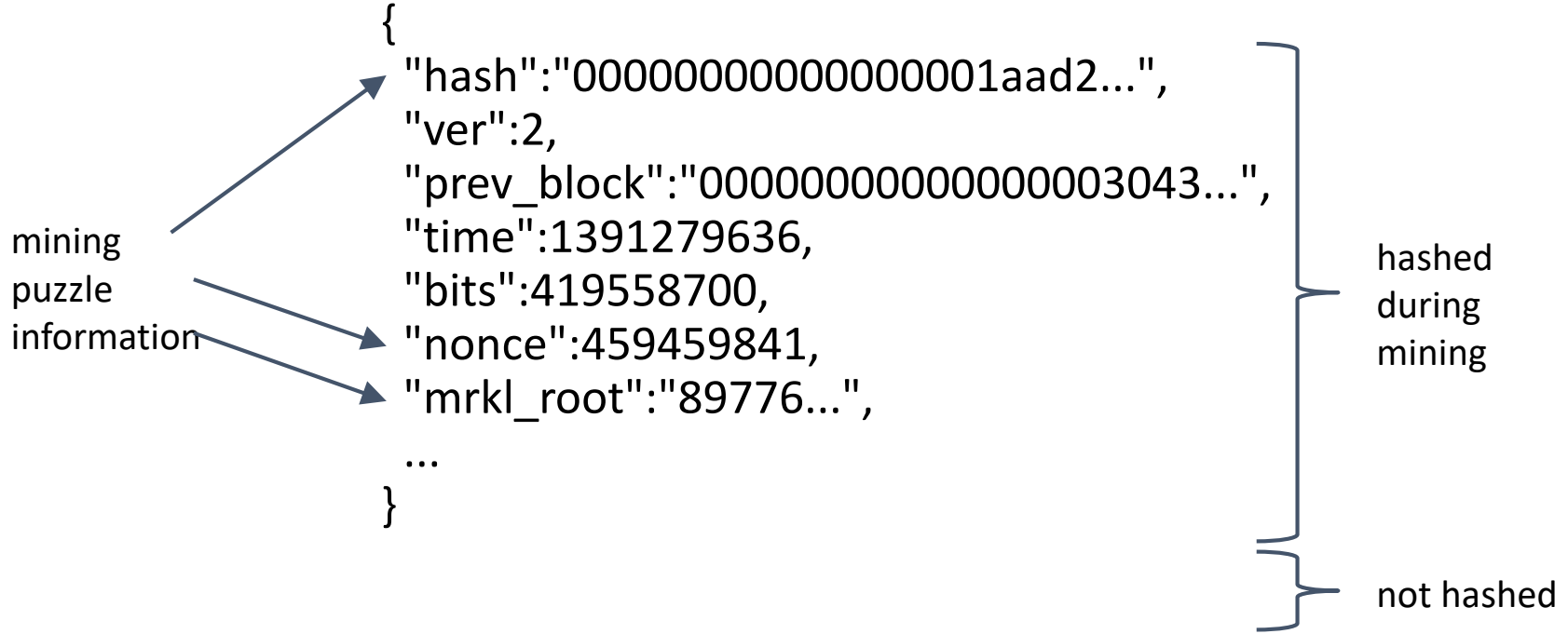
Hash tree (Merkle tree) of transactions in each block



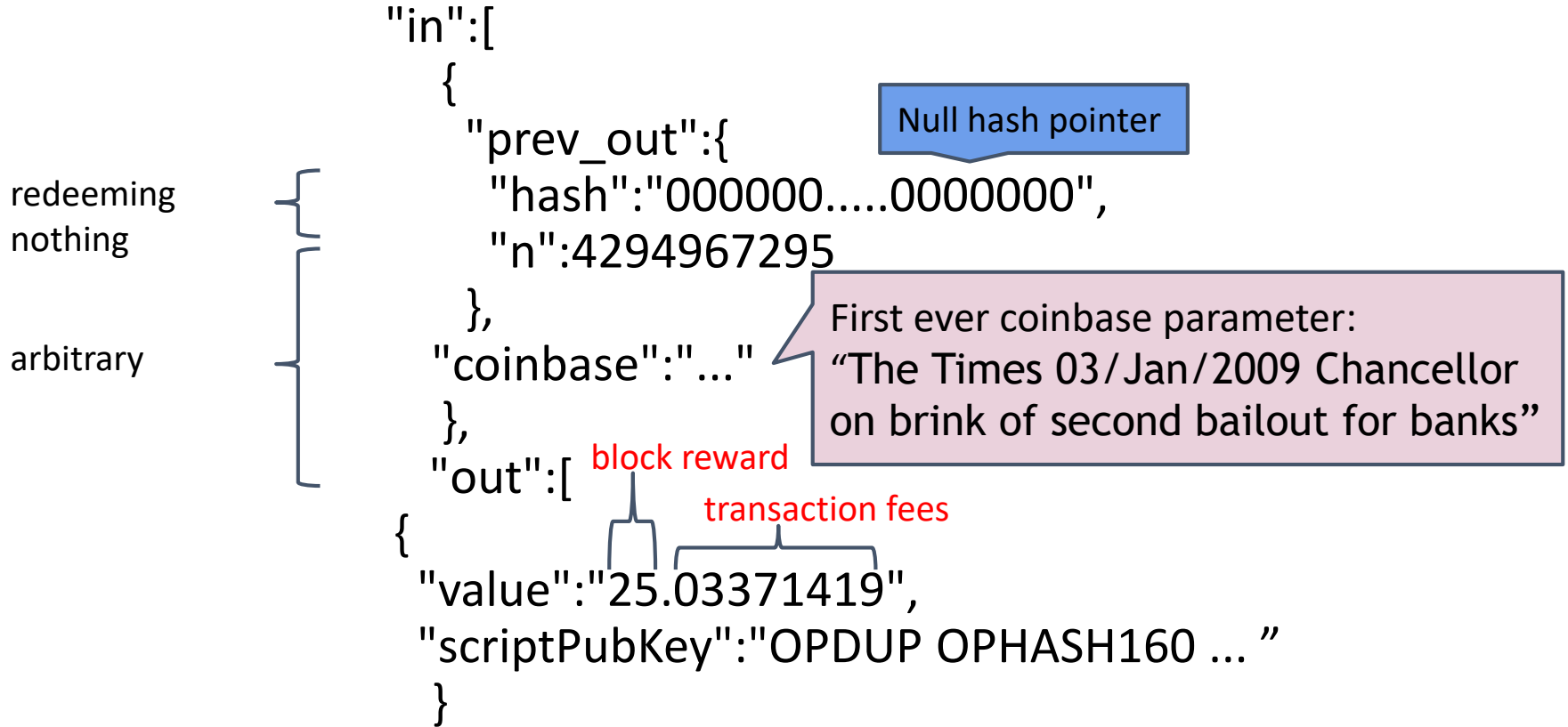
# The real deal: a Bitcoin block



# The real deal: a Bitcoin block header



# The real deal: coinbase transaction



# See for yourself!

## Transaction View information about a bitcoin transaction

151b750d1f13e76d84e82b34b12688811b23a8e3119a1cba4b4810f9b0ef408d

1KryFUt9tXHvaoCYTNPbqpWPJKQ717YmL5




1KvrdrQ3oGqMAiDTMEYcCdDSnVaGNW2YZh  
1KryFUt9tXHvaoCYTNPbqpWPJKQ717YmL5

1.0194 BTC  
3.458 BTC

9 Confirmations

4.4774 BTC

### Summary

Size	257 (bytes)
Received Time	2014-08-05 01:55:25
Included In Blocks	<a href="#">314018</a> (2014-08-05 02:00:40 +5 minutes)
Confirmations	9 Confirmations
Relayed by IP 	<a href="#">Blockchain.info</a>
Visualize	<a href="#">View Tree Chart</a>

### Inputs and Outputs

Total Input	4.4775 BTC
Total Output	4.4774 BTC
Fees	0.0001 BTC
Estimated BTC Transacted	1.0194 BTC
Scripts	<a href="#">Show scripts &amp; coinbase</a>

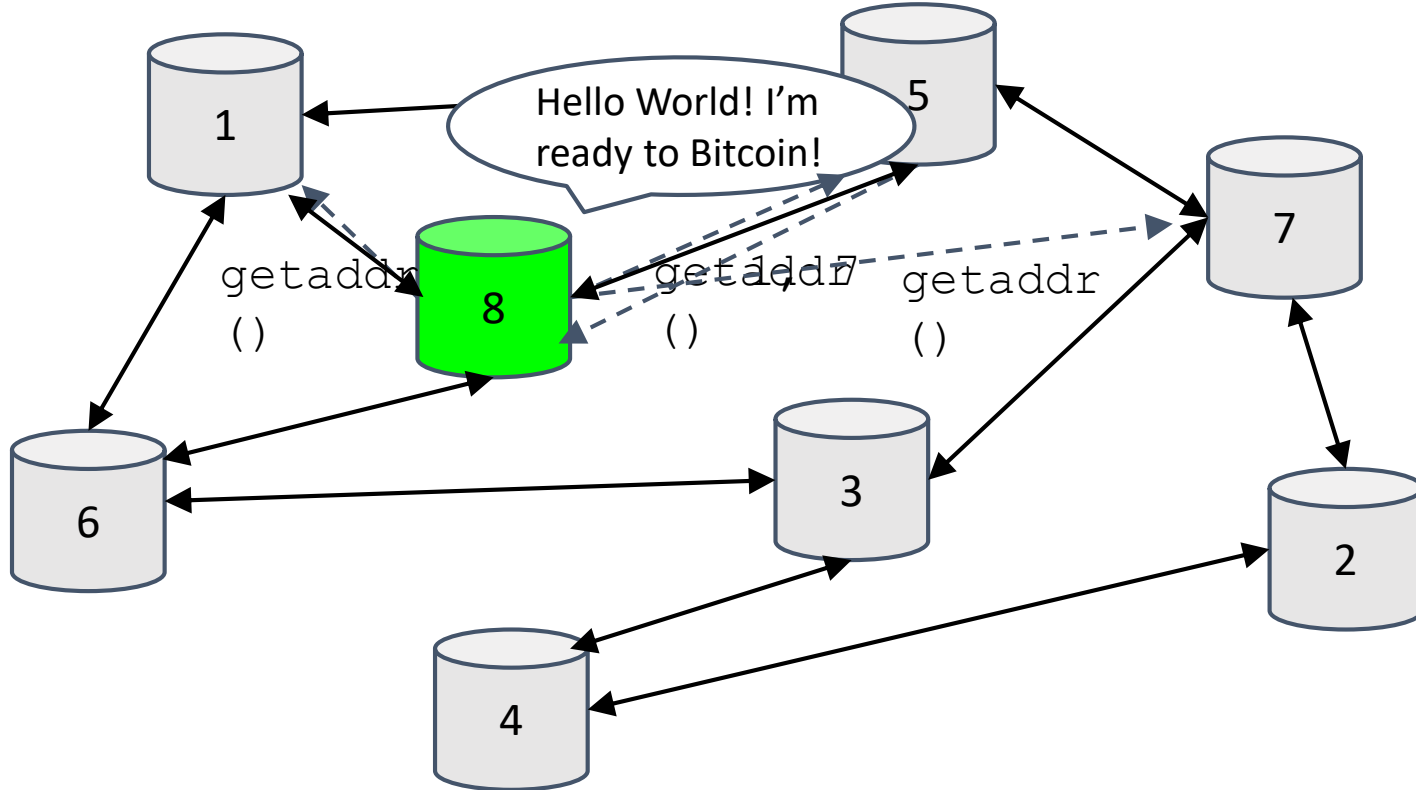
blockchain.info (and many other sites)

# The Bitcoin network

# Bitcoin P2P network

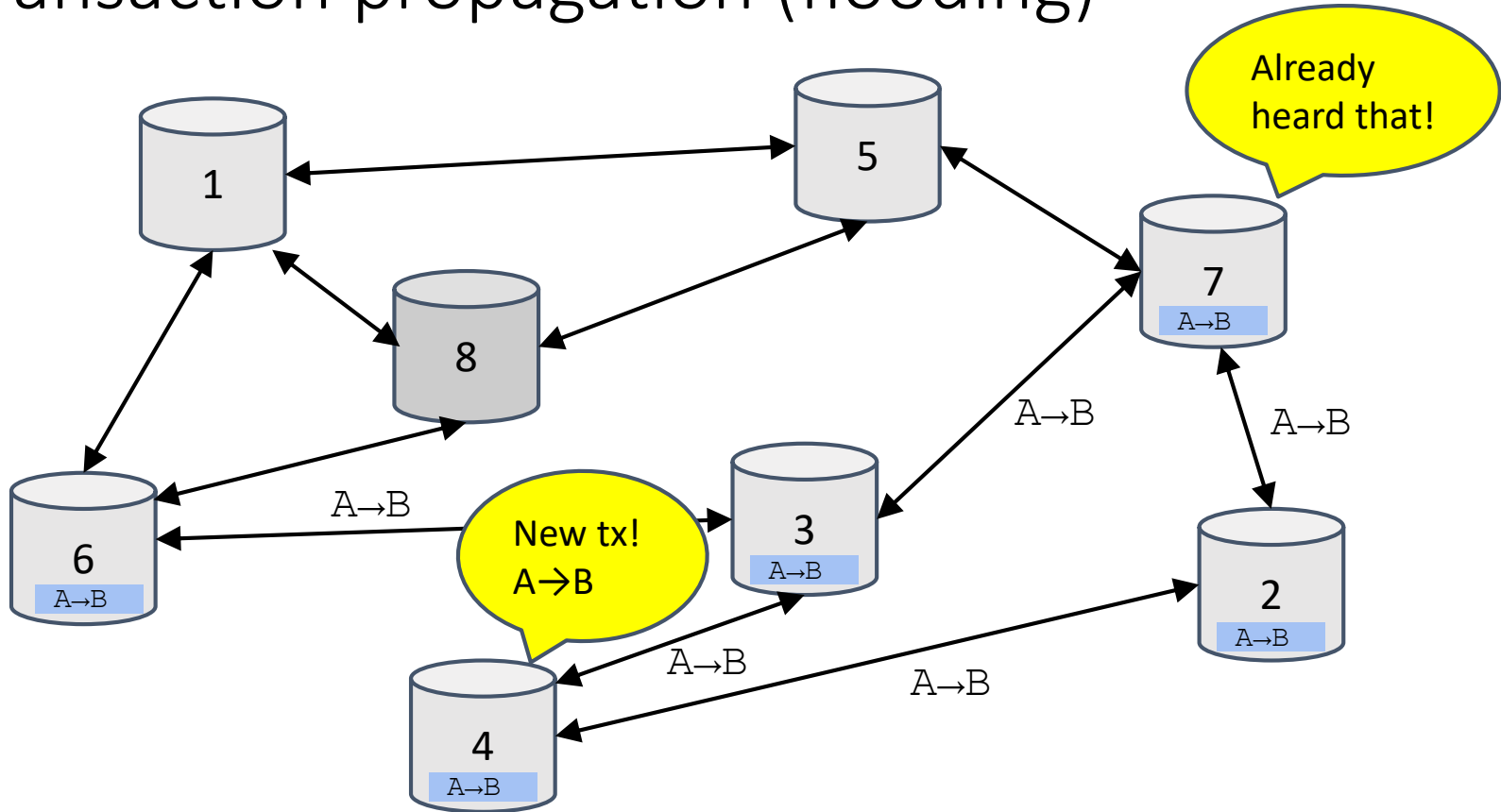
- Ad-hoc protocol (runs on TCP port 8333)
- Ad-hoc network with random topology
- All nodes are equal
- New nodes can join at any time
- Forget non-responding nodes after 3 hr

# Joining the Bitcoin P2P network





# Transaction propagation (flooding)



# Should I relay a proposed transaction?

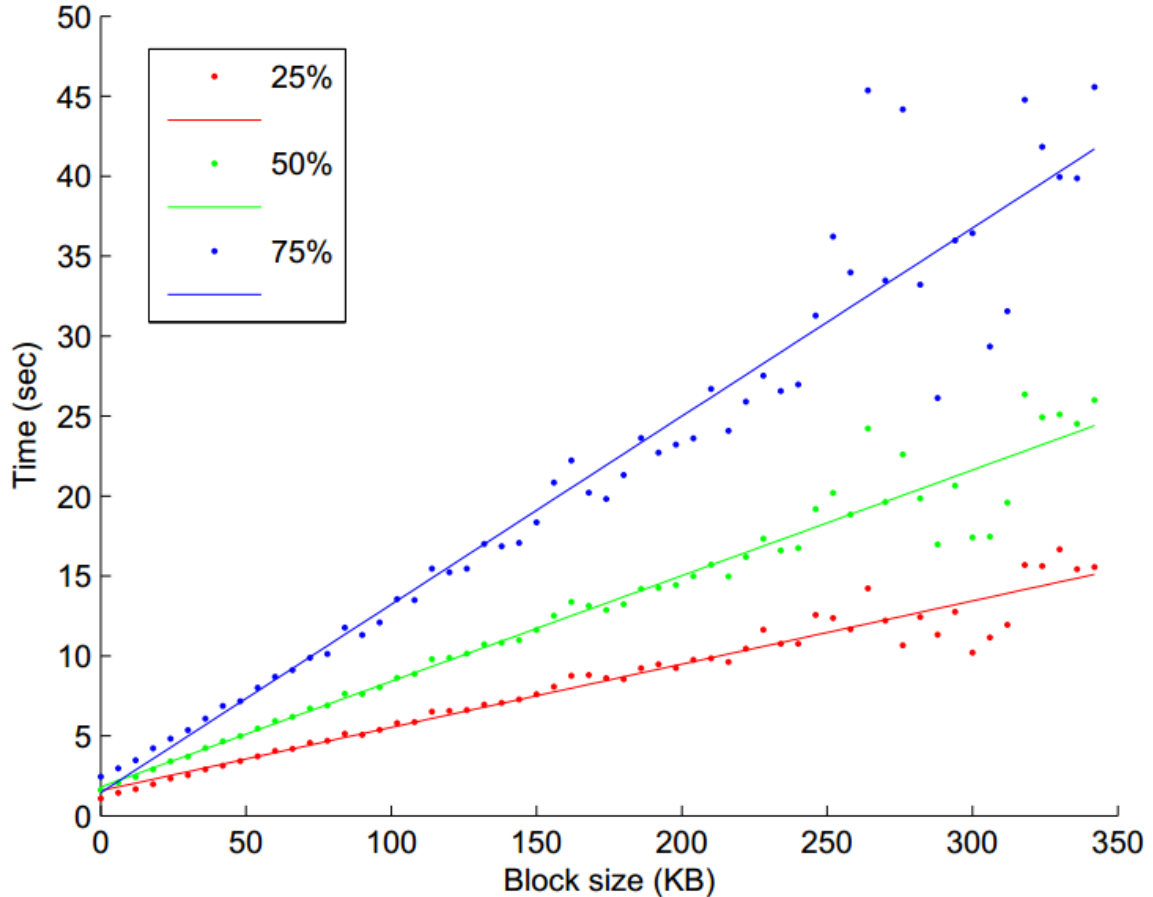
- Transaction valid with current block chain(default)
  - Run script for each previous output being redeemed and ensure that script returns true!
- Script matches a whitelist
  - Avoid unusual scripts
- Haven't seen before
  - Avoid infinite loops
- Doesn't conflict with others I've relayed
  - Avoid double-spends

Sanity checks only...

Well-behaving nodes implement them!

Some nodes may ignore them!

### Block Propagation Times



Source: Yonatan Sompolinsky and Aviv Zohar: "Accelerating Bitcoin's Transaction Processing" 2014

# How big is the network?

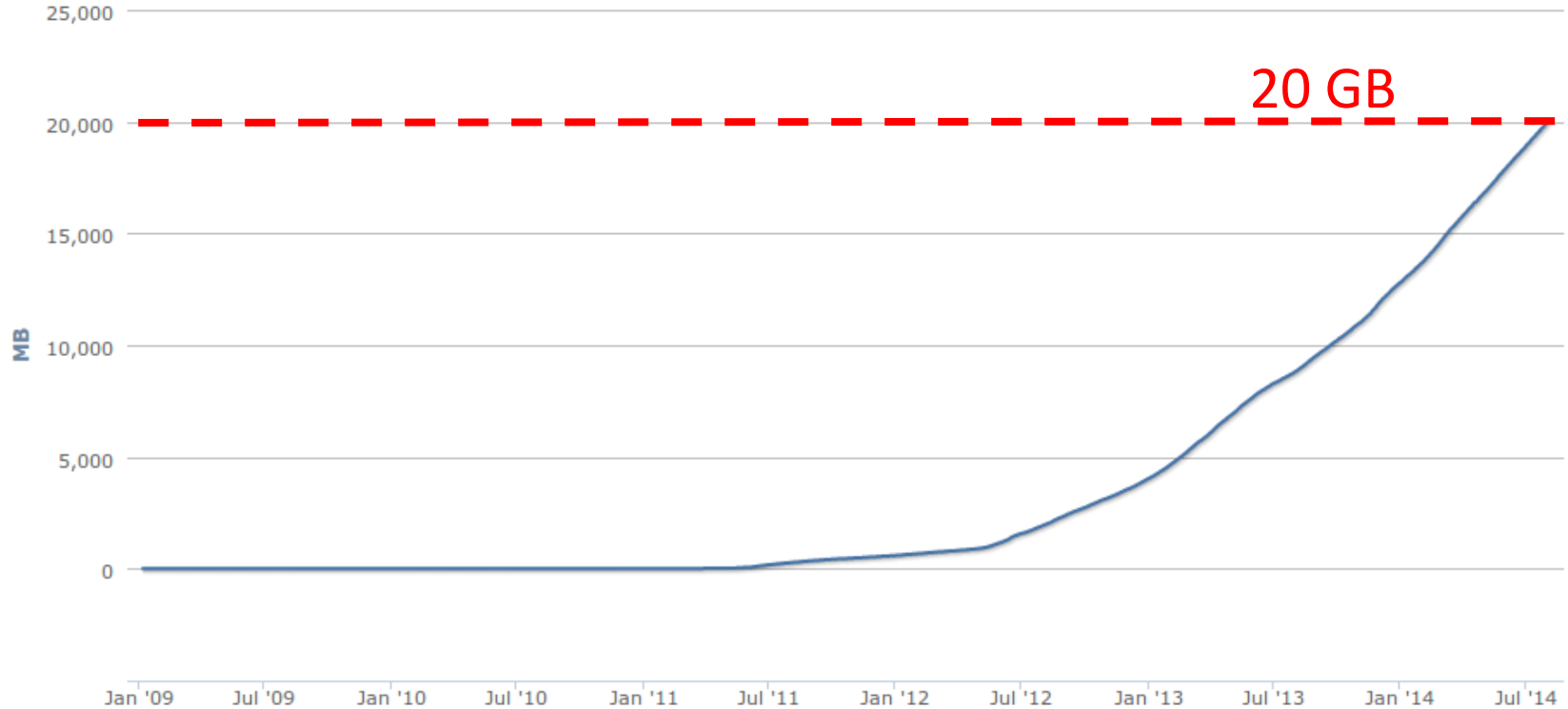
- Impossible to measure exactly
- Estimates-up to 1M IP addresses/month
- Only about 5-10k “full nodes”
  - Permanently connected
  - Fully-validate
- This number may be dropping!

# Fully-validating nodes

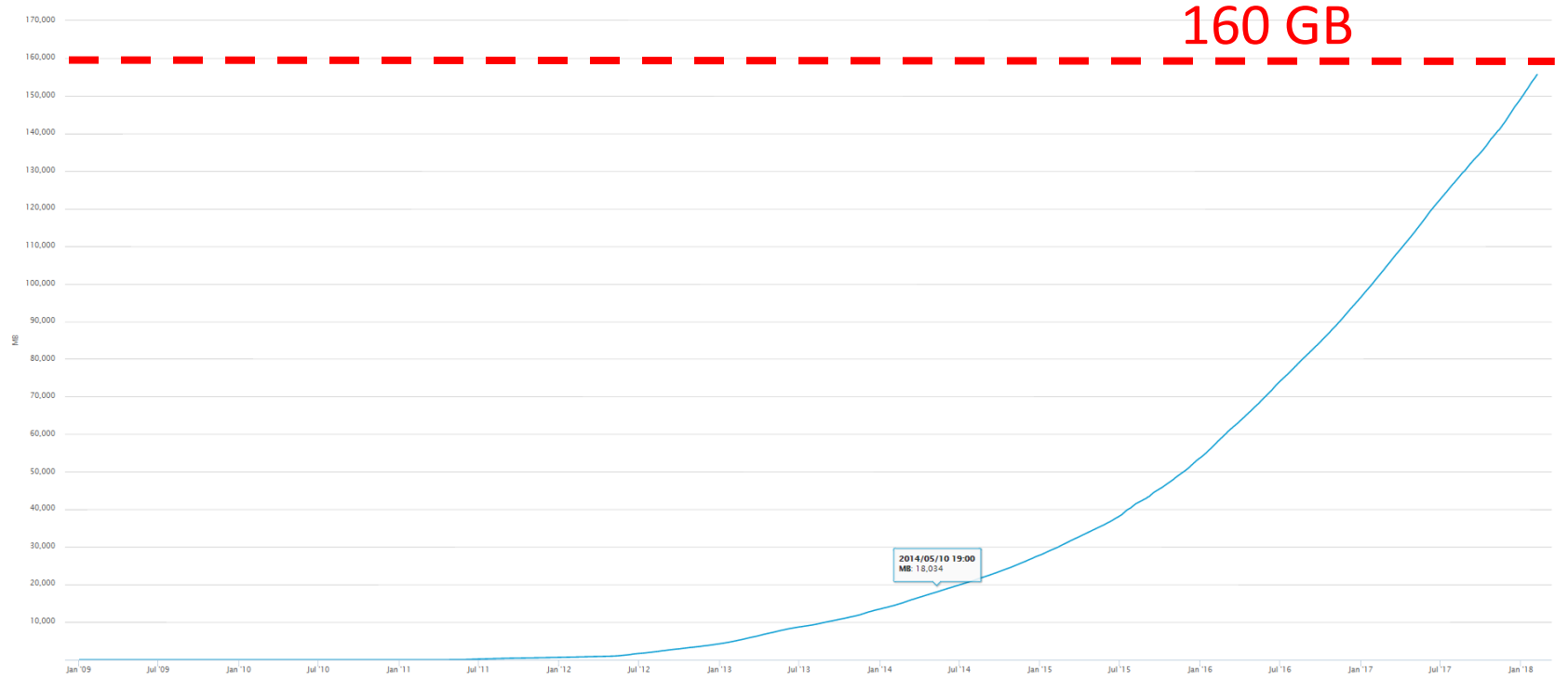
- Permanently connected
- Store entire block chain
- Hear and forward every node/transaction

# Storage costs (in 2014)

Blockchain Size  
Source: blockchain.info

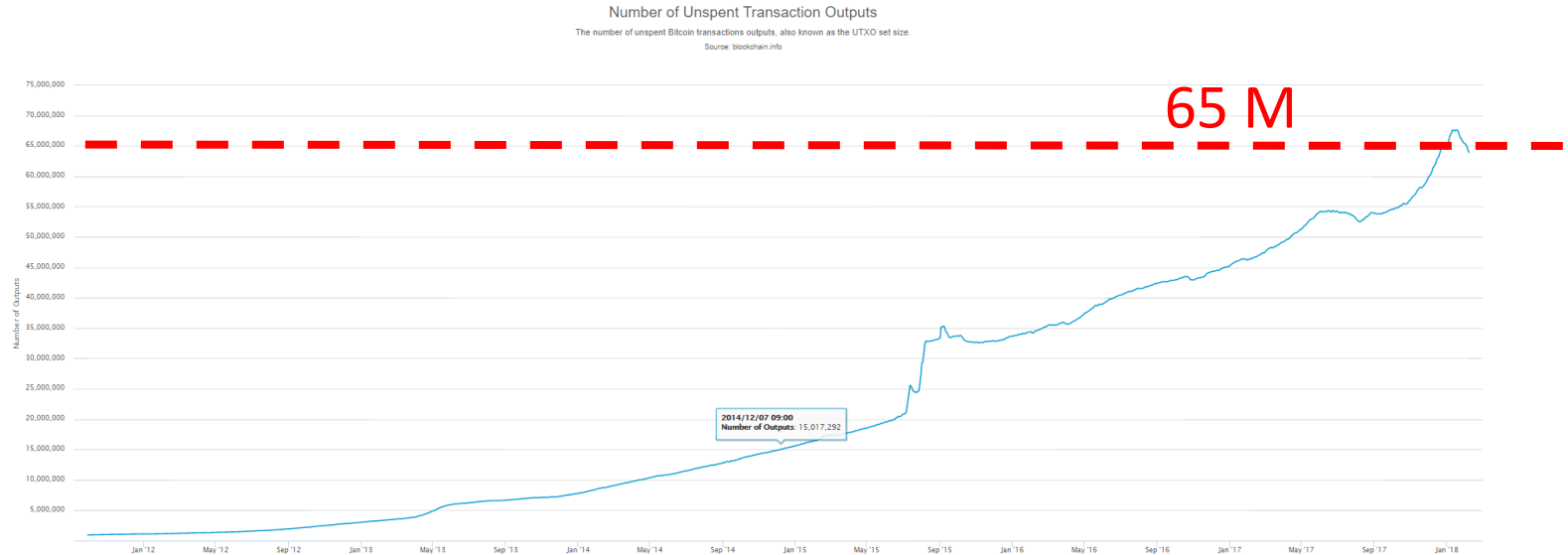


# Storage costs (in 2018)



# Tracking the UTXO set

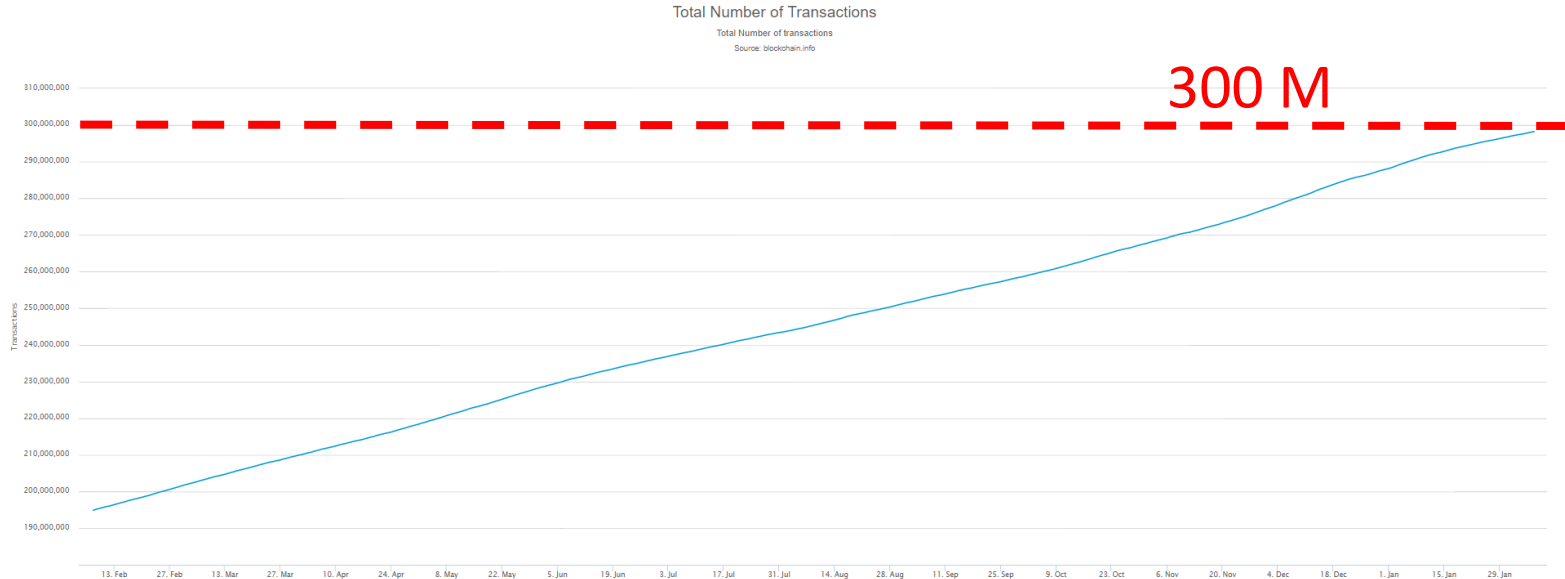
- **U**nspent **T**ransaction **O**utput
  - Should be stored in memory - everything else can be stored on disk, why?





# Tracking the UTXO set

- Currently ~65 M UTXOs
  - Out of 300 M transactions
- Can require several Gigabytes to store – can it fit in the RAM of a standard computer?



# Software diversity

- About 90% of nodes run “Core Bitcoin” (C++)
  - Some are out of date versions
- Other implementations running successfully
  - BitcoinJ (Java)
  - Libbitcoin (C++)
  - btcd (Go)
- “Original Satoshi client”

# Limitations & Improvements

# Hard-coded limits in Bitcoin

- 10 min. average creation time per block
- 1 M bytes in a block
- 20,000 signature operations per block
- 100 M *satoshis* per bitcoin
- 23M total bitcoins maximum
- 50,25,12.5... bitcoin mining reward

These affect  
economic  
balance of  
power too  
much to change  
now

# Throughput limits in Bitcoin

- 1 M bytes/block (10 min)
- >250 bytes/transaction
- 7 transactions/sec 😞

Compare to:

- VISA: 2,000-10,000 transactions/sec
- PayPal: 50-100 transaction/sec

# Cryptographic limits in Bitcoin

- Only 1 signature algorithm (ECDSA/P256)
- Hard-coded hash functions

Crypto primitives might break by 2040...

# Discussion