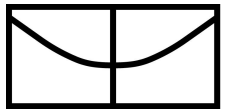


MIT Digital Currency Initiative and the University of Brasilia presents

Cryptocurrency Design and Engineering

Lectures 7 and 8: Mining, Forks, and Governance
Taught by: Neha Narula
September 30, 2025
MAS.S62

Recap

- Hash functions (like SHA256)
- Proof-of-work
- Consensus for building a log of transactions
- Transaction formats and state

Mining

- How does Bitcoin securely build a log in practice? Everyone tries:
  - Gathering up a bunch of transactions
  - Putting them into a block
  - Publishing the block to the network
- Main goal: make it really hard for anyone to keep you from getting your transaction confirmed (**censorship resistance**)

Strawman

Pick a bunch of companies; assign them the role of including transactions in the log

- I need to convince at least one company to take my transaction
- What could go wrong?

Open questions

- How many companies?
- How do we motivate them to keep doing the work?
- What if they all disappear?
- Could they coordinate?

Goal: I can become a block producer

Idea: include my own transaction!

- Proof-of-stake: I need to acquire stake (**in protocol**) to start producing blocks
- Proof-of-work: I need to acquire computational power (**extra-protocol**) to start producing blocks

Origins of proof-of-work

- Hashcash (Back, 1997)
- Pricing via Processing or Combatting Junk Mail (Dwork and Naor, 1992)

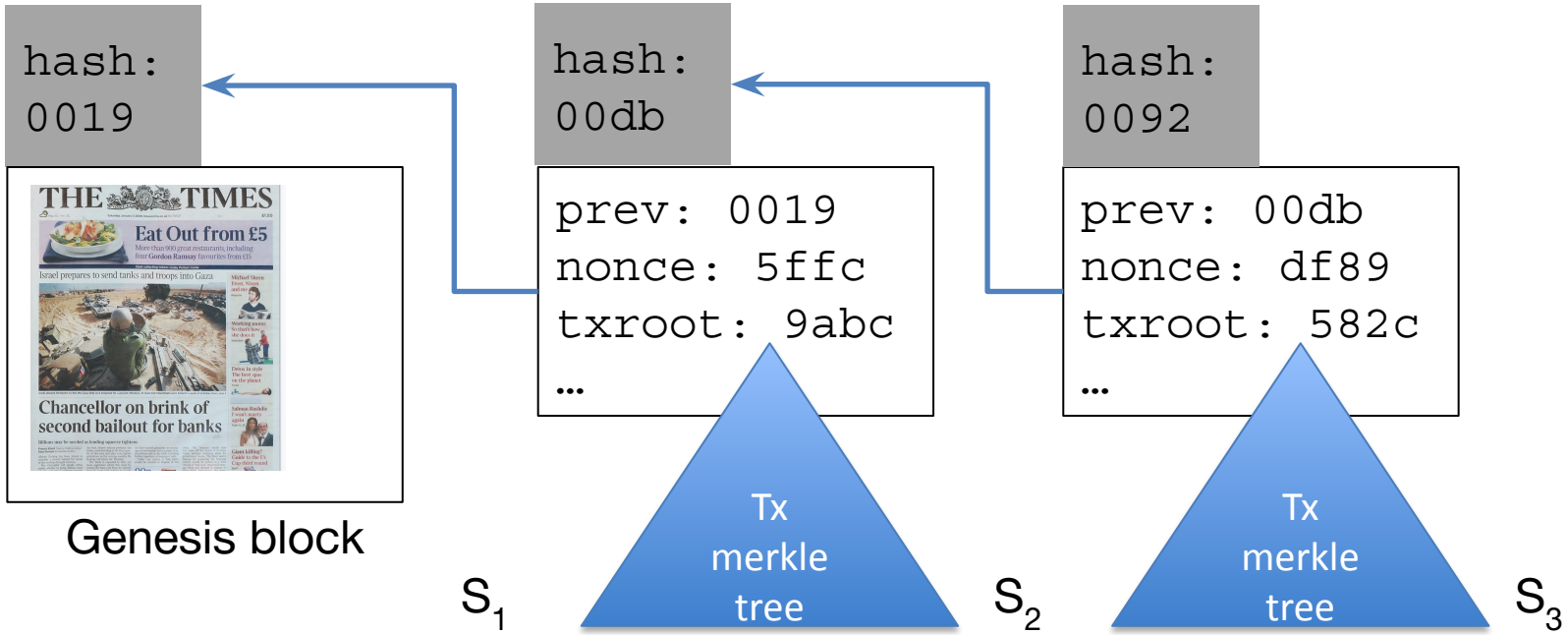
Proof-of-work in Bitcoin

- Message m, nonce n, target t
- Choose n such that SHA256(m, n) = h, h < t

Properties of proof-of-work

- Permissionless (just show up with a solution!)
- Deterministic, non-interactive verification
- Cheap to verify – O(1)
- Memoryless

Blockchain


hash:
0019
THE TIMES
Eat Out from £5
Israel prepares to send tanks and troops into Gaza
Chancellor on brink of second bailout for banks
Genesis block
hash:
00db
prev: 0019
nonce: 5ffc
txroot: 9abc
…
Tx merkle tree
hash:
0092
prev: 00db
nonce: df89
txroot: 582c
…
Tx merkle tree
S_0
S_1
S_2
S_3

```
00000000   01 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ................
00000010   00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ................
00000020   00 00 00 00 3B A3 ED FD  7A 7B 12 B2 7A C7 2C 3E   ....;£íýz{.²zÇ,>
00000030   67 76 8F 61 7F C8 1B C3  88 8A 51 32 3A 9F B8 AA   gv.a.È.Ã^ŠQ2:Ÿ¸ª
00000040   4B 1E 5E 4A 29 AB 5F 49  FF FF 00 1D 1D AC 2B 7C   K.^J)«_Iÿÿ...¬+|
00000050   01 01 00 00 00 01 00 00  00 00 00 00 00 00 00 00   ................
00000060   00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ................
00000070   00 00 00 00 00 00 FF FF  FF FF 4D 04 FF FF 00 1D   ......ÿÿÿÿM.ÿÿ..
00000080   01 04 45 54 68 65 20 54  69 6D 65 73 20 30 33 2F   ..EThe Times 03/
00000090   4A 61 6E 2F 32 30 30 39  20 43 68 61 6E 63 65 6C   Jan/2009 Chancel
000000A0   6C 6F 72 20 6F 6E 20 62  72 69 6E 6B 20 6F 66 20   lor on brink of
000000B0   73 65 63 6F 6E 64 20 62  61 69 6C 6F 75 74 20 66   second bailout f
000000C0   6F 72 20 62 61 6E 6B 73  FF FF FF FF 01 00 F2 05   or banksÿÿÿÿ..ò.
000000D0   2A 01 00 00 00 43 41 04  67 8A FD B0 FE 55 48 27   *....CA.gŠý°þUH'
000000E0   19 67 F1 A6 71 30 B7 10  5C D6 A8 28 E0 39 09 A6   .gñ¦q0·.\Ö¨(à9.¦
000000F0   79 62 E0 EA 1F 61 DE B6  49 F6 BC 3F 4C EF 38 C4   ybàê.aÞ¶Iö¼?Lï8Ä
00000100   F3 55 04 E5 1E C1 12 DE  5C 38 4D F7 BA 0B 8D 57   óU.å.Á.Þ\8M÷º..W
00000110   8A 4C 70 2B 6B F1 1D 5F  AC 00 00 00 00            ŠLp+kñ._¬....
```

Tick tock, next block…

Block headers

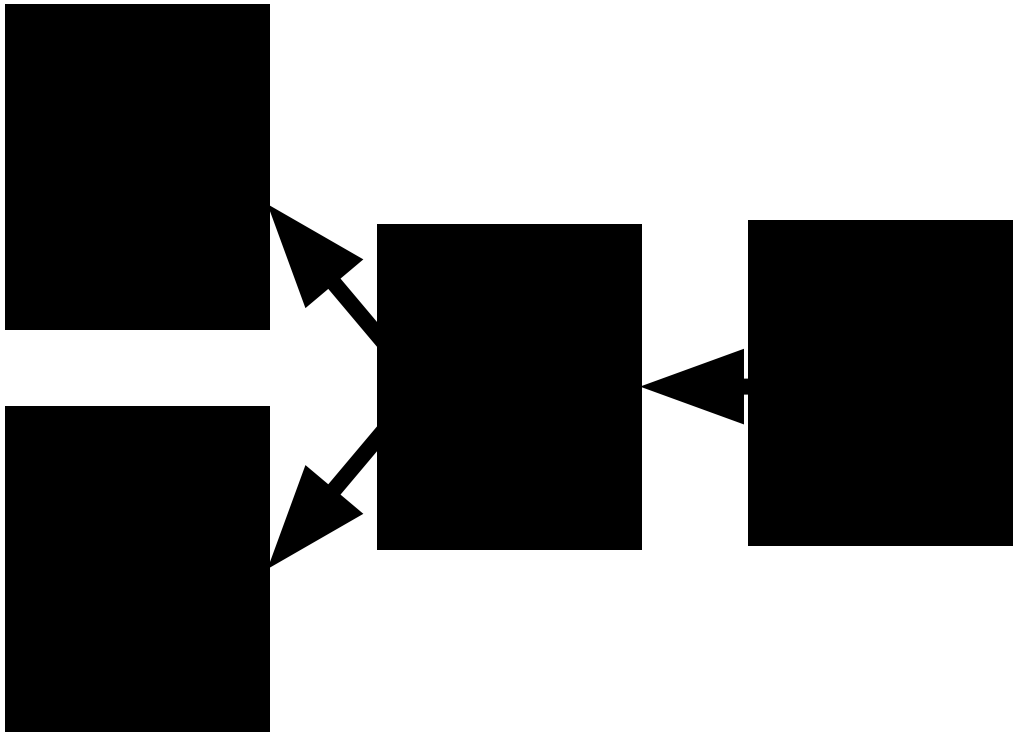
field	size	purpose
version	4B	Block version
prev hash	32B	Hash of previous block
Merkle root	32B	Root of merkle tree of all transactions
time	4B	Unix timestamp
difficulty	4B	Proof-of-work target
nonce	4B	To calculate proof-of-work

Validation Rules

- Block size
- Enough proof of work
- Prev block hash pointers
- Block timestamps
- Only valid transactions

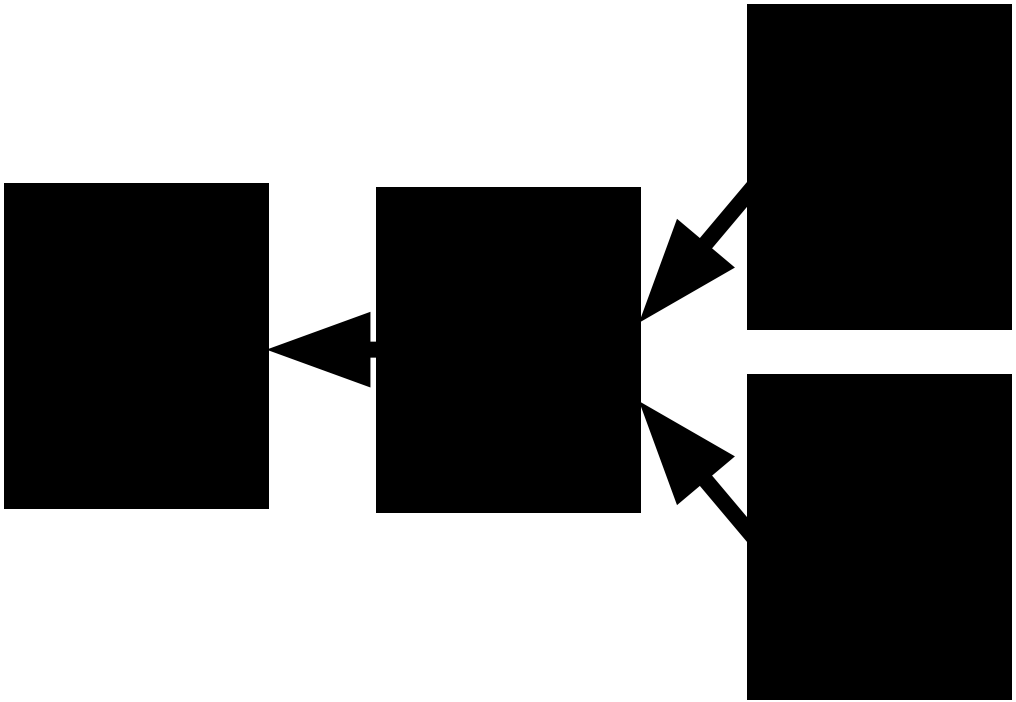
"Consensus critical"

Can a block point to two prev blocks?



No! Only one spot for prev hash

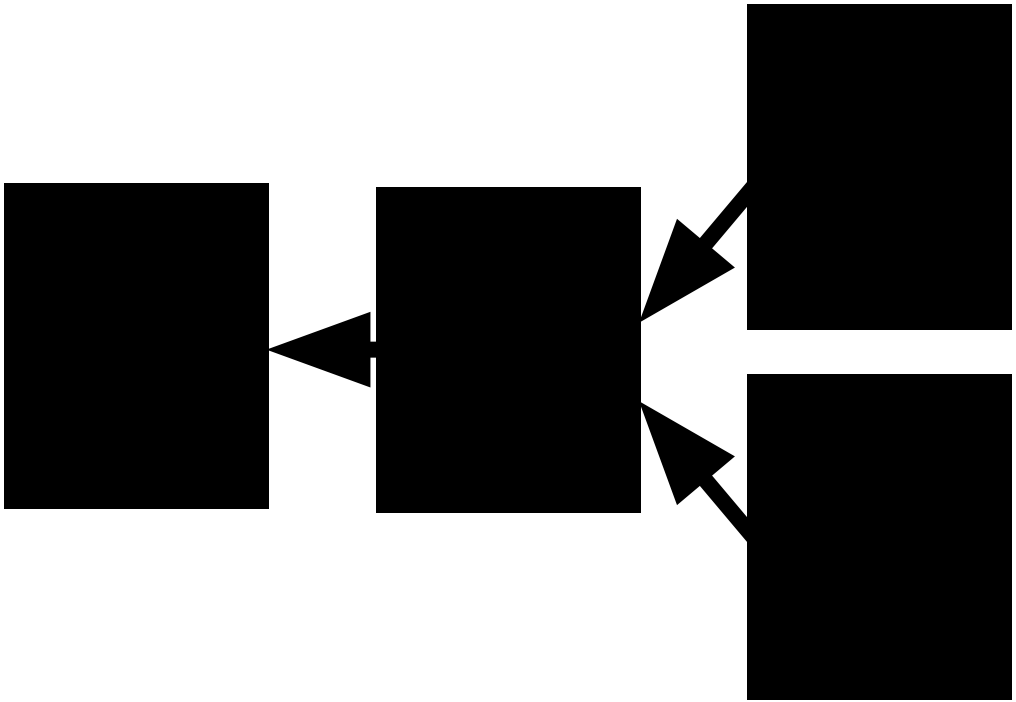
Can two blocks point to one?



Yes! Known as a FORK.

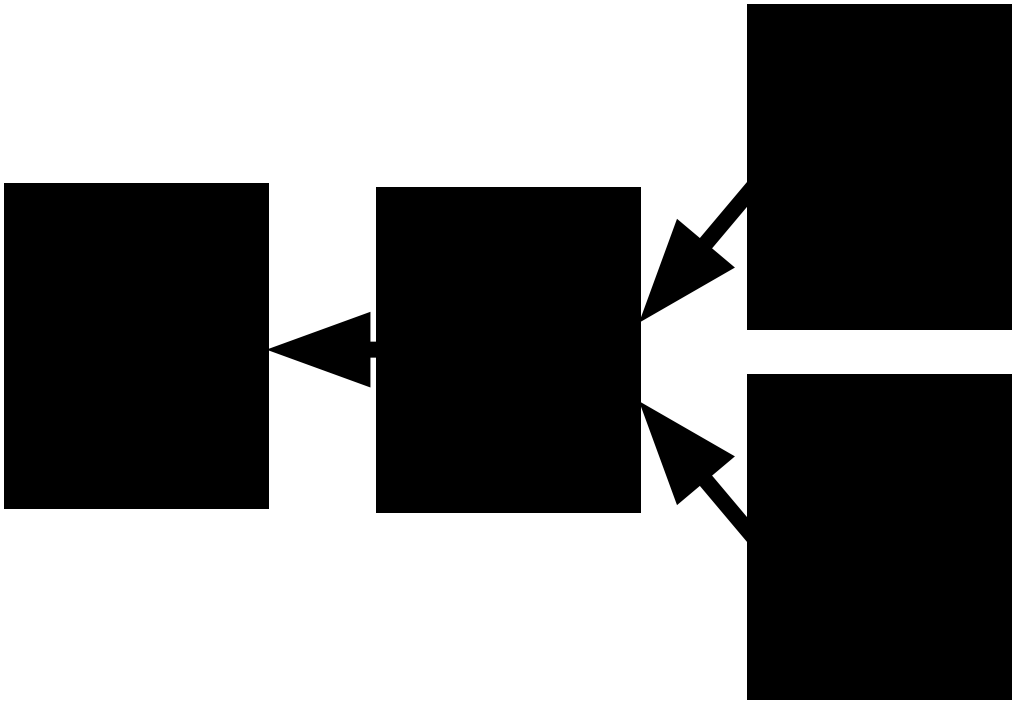
What does this mean?

What does a fork mean?



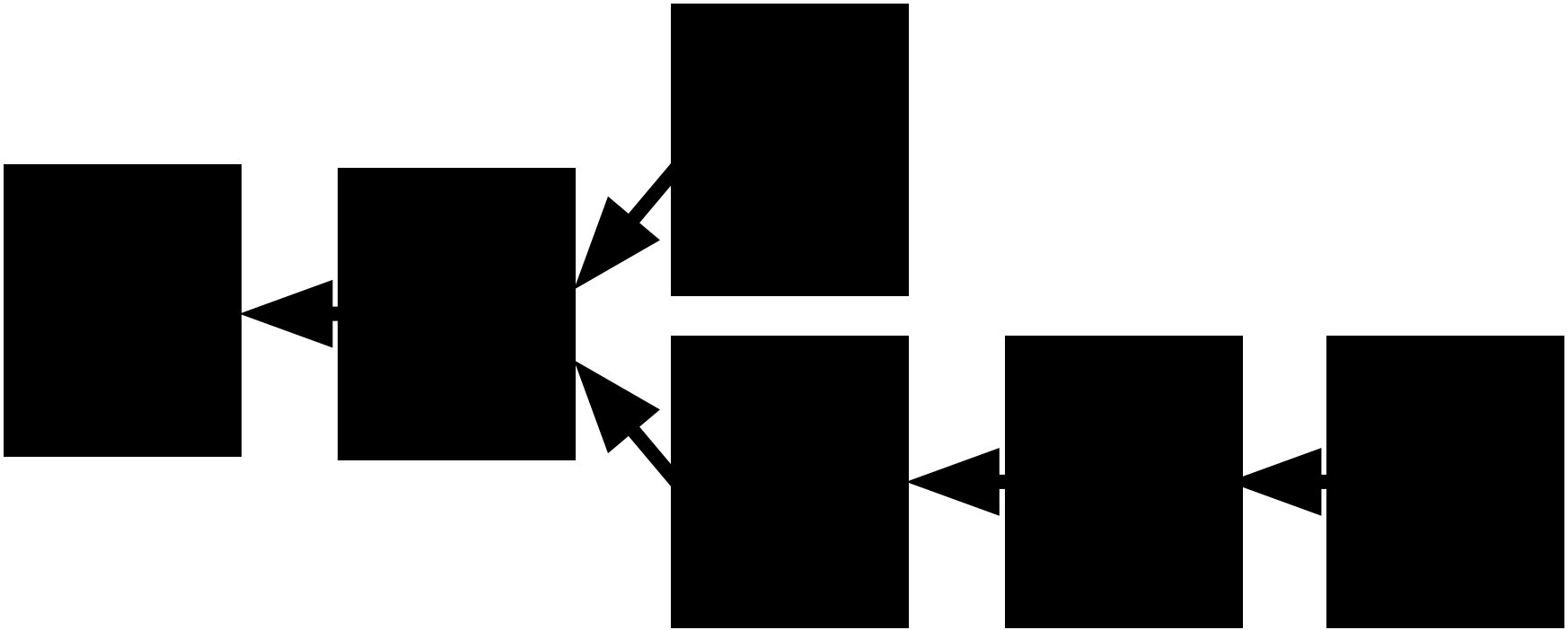
- Two versions of history
- Possible double spends
- Two currencies!

How do we fix it?

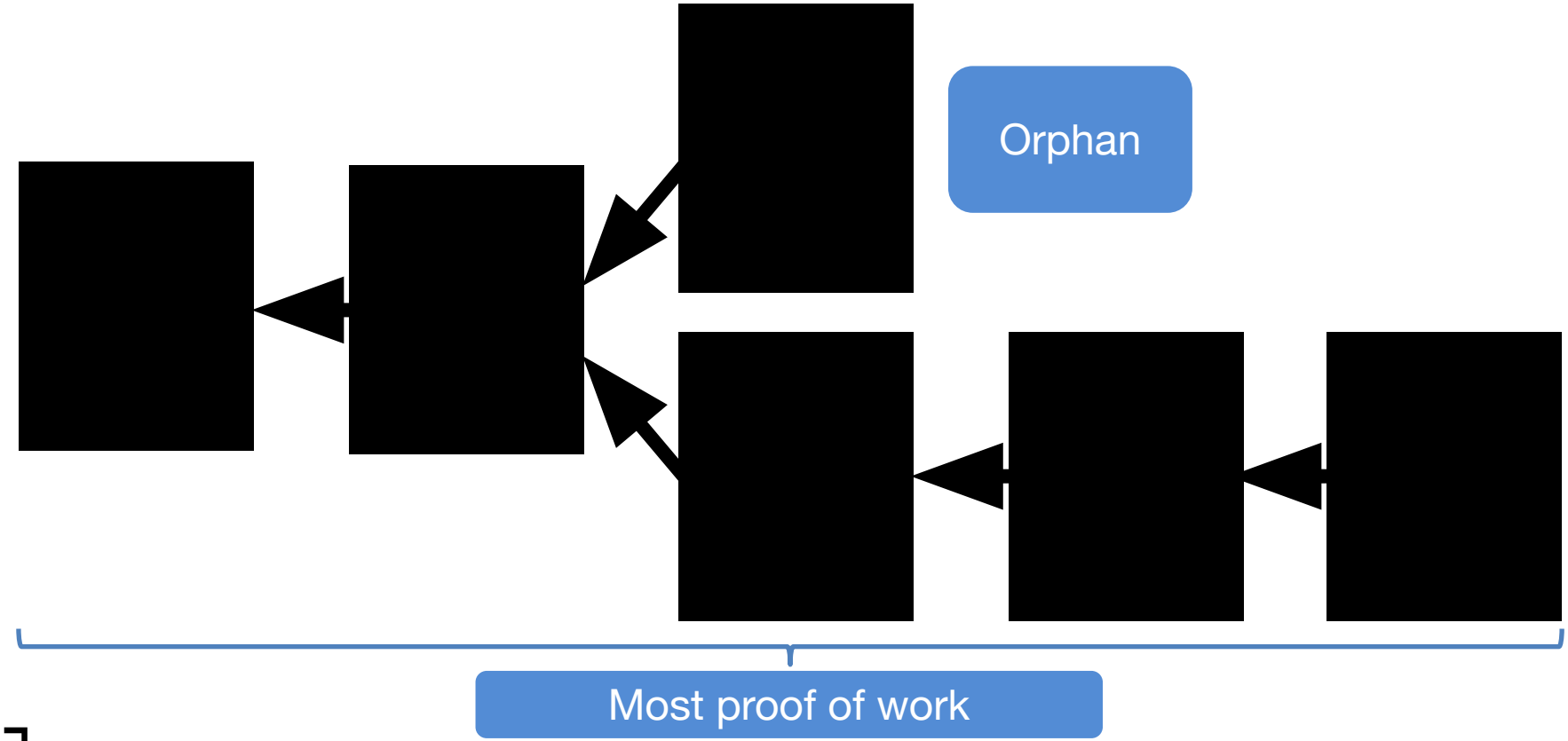


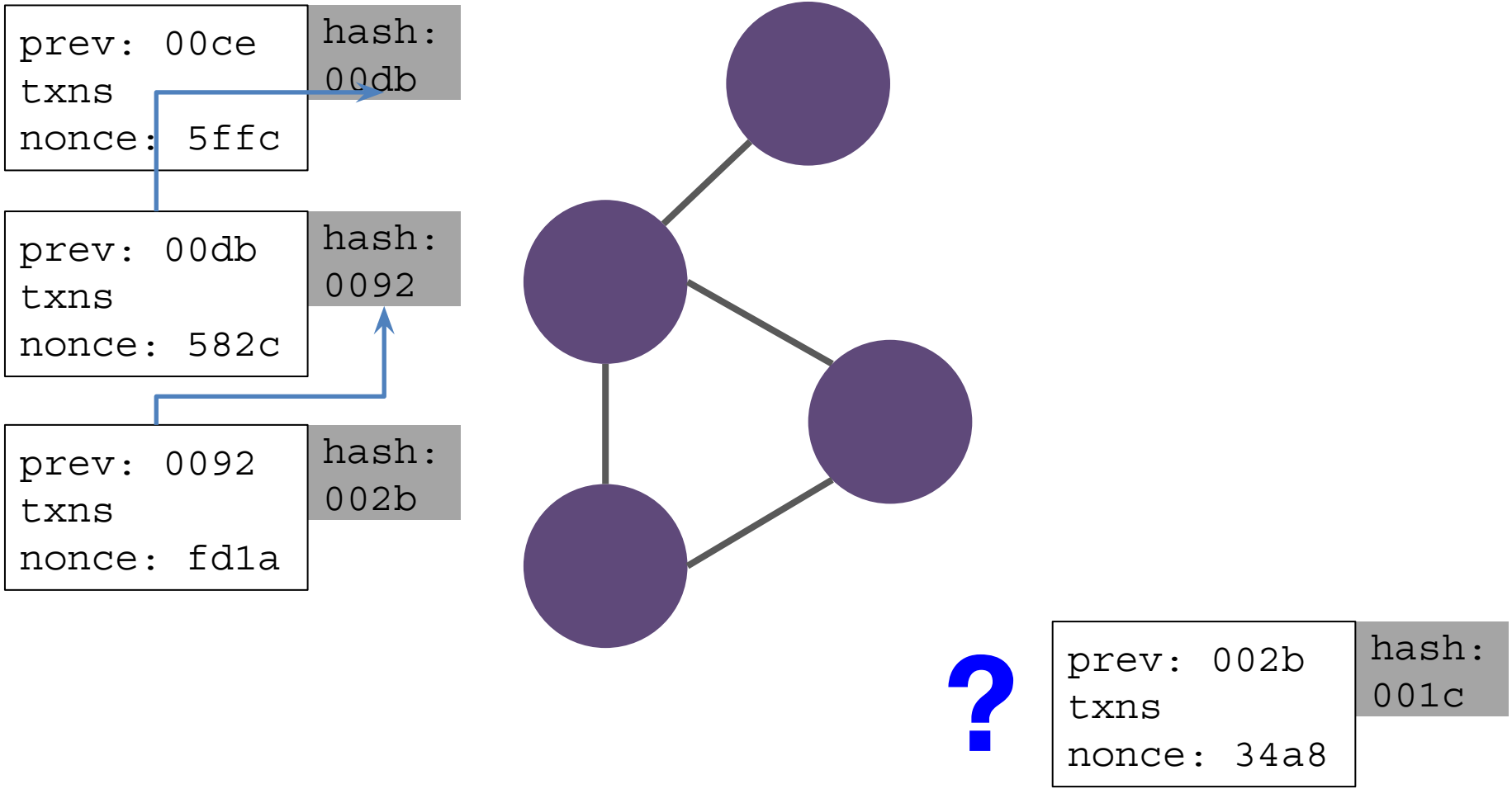
Which is the “right” one?

Over time, one will win



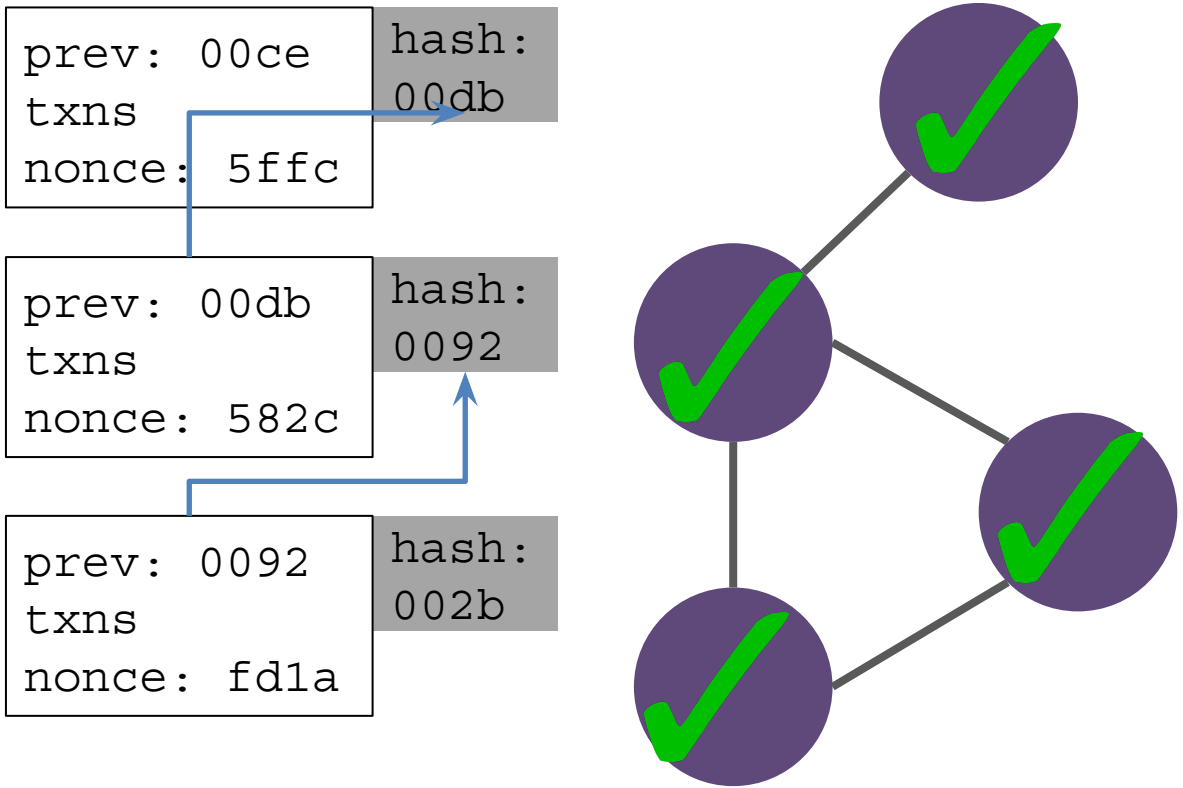
Over time, one will win




prev: 00ce
txns
nonce: 5ffc
hash: 00db
prev: 00db
txns
nonce: 582c
hash: 0092
prev: 0092
txns
nonce: fd1a
hash: 002b
?
prev: 002b
txns
nonce: 34a8
hash: 001c

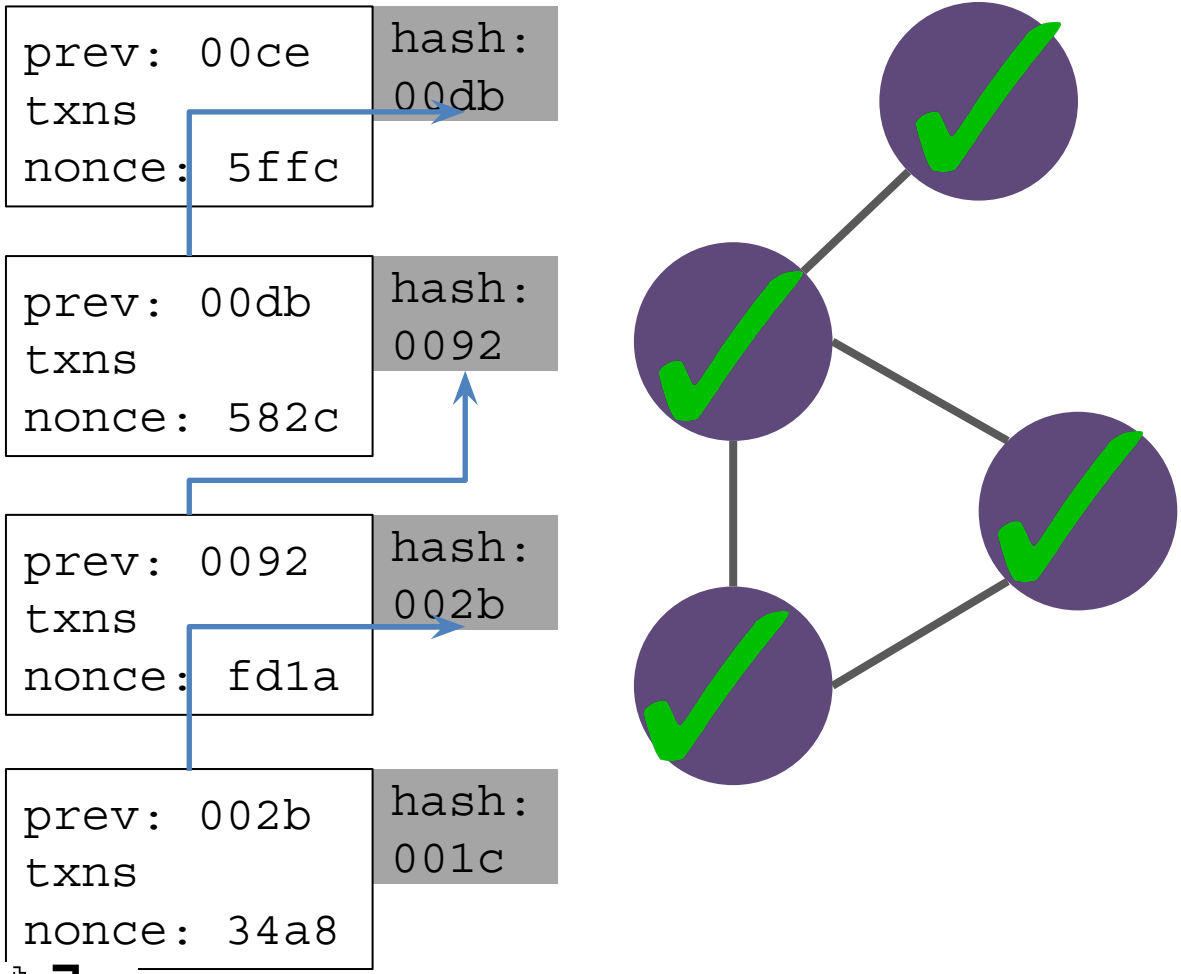
Validation Rules

- Block size
- Enough proof of work
- Prev block hash pointers
- Block timestamps
- Only valid transactions



prev: 002b
txns
nonce: 34a8

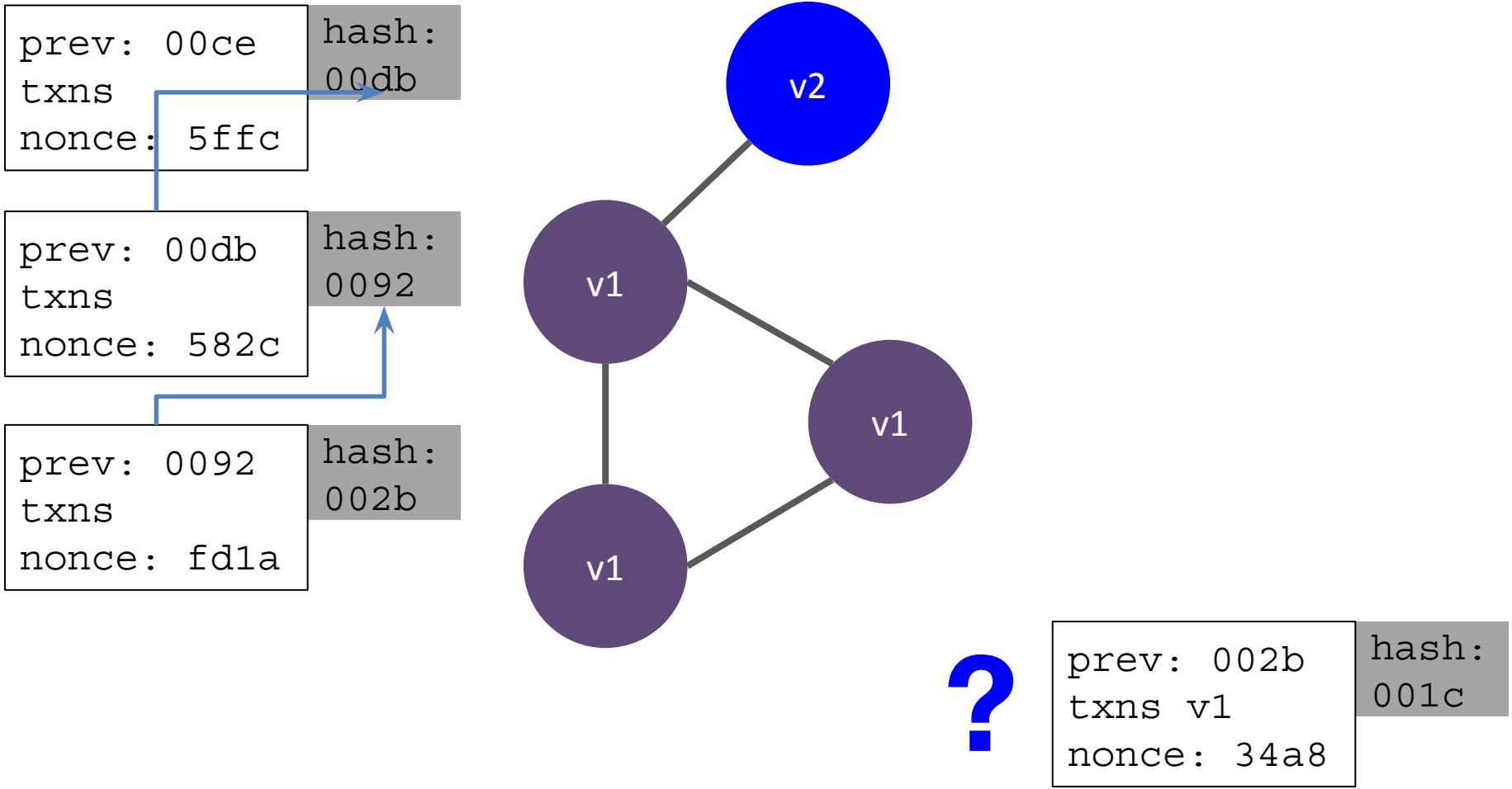
hash: 001c


prev: 00ce
txns
nonce: 5ffc
hash: 00db
prev: 00db
txns
nonce: 582c
hash: 0092
prev: 0092
txns
nonce: fd1a
hash: 002b
prev: 002b
txns
nonce: 34a8
hash: 001c

Changing the validation rules

- Fix bugs
- Major security issues
- New features

Can’t get everyone to upgrade at the same time!


prev: 00ce
txns
nonce: 5ffc
hash: 00db
prev: 00db
txns
nonce: 582c
hash: 0092
prev: 0092
txns
nonce: fd1a
hash: 002b
v2
v1
v1
v1
?
prev: 002b
txns v1
nonce: 34a8
hash: 001c

prev: 00ce
txns
nonce: 5ffc

hash: 00db

prev: 00db
txns
nonce: 582c

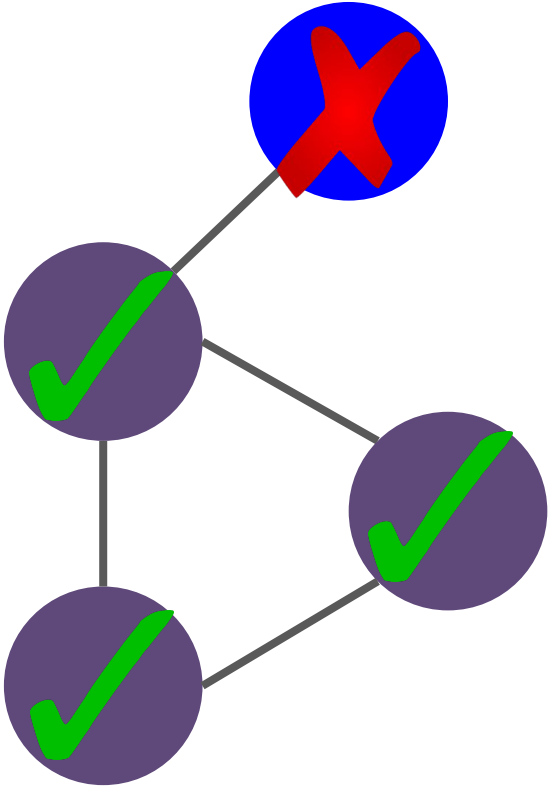
hash: 0092

prev: 0092
txns
nonce: fd1a

hash: 002b

prev: 002b
txns v1
nonce: 34a8

hash: 001c



prev: 00ce
txns
nonce: 5ffc

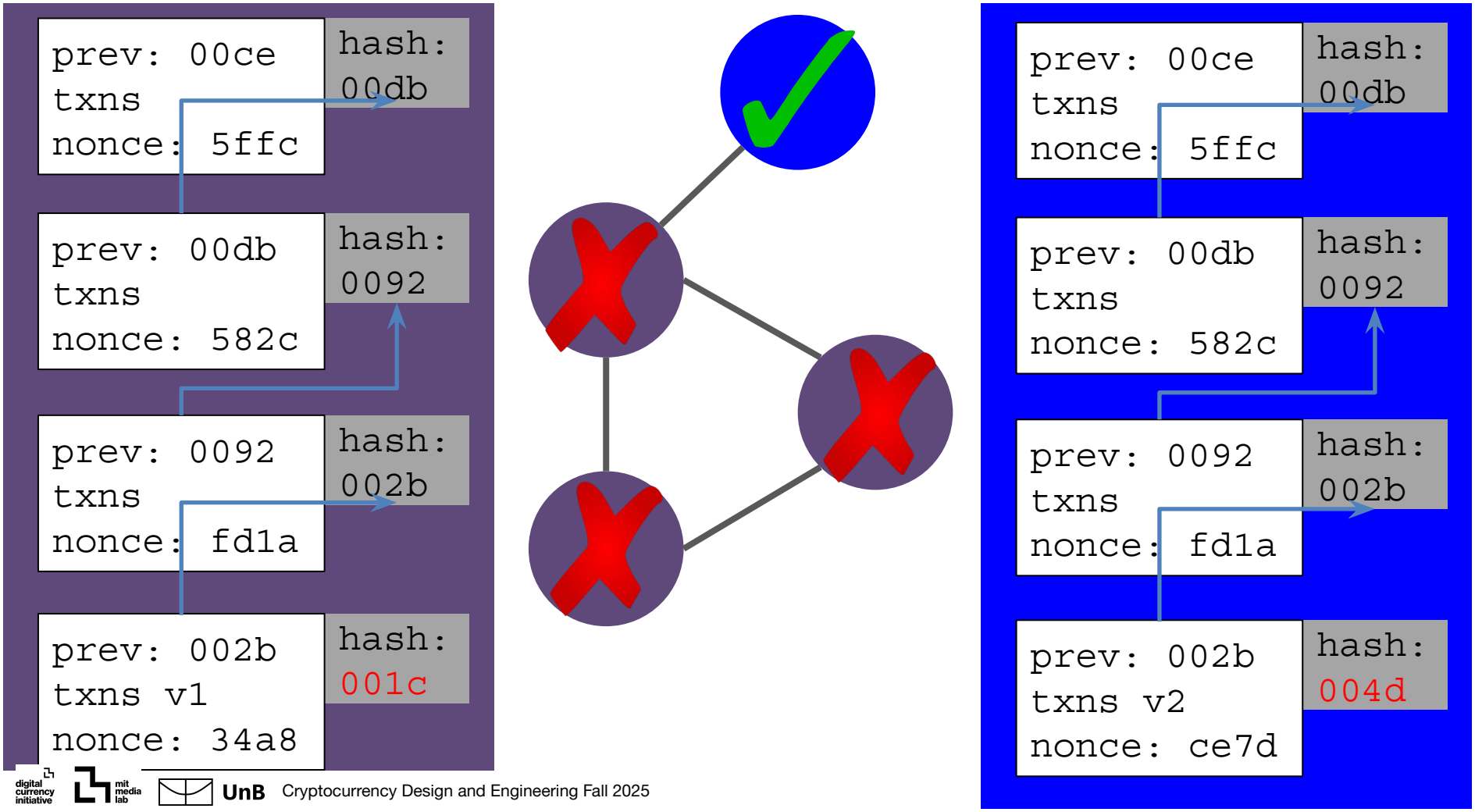
hash: 00db

prev: 00db
txns
nonce: 582c

hash: 0092

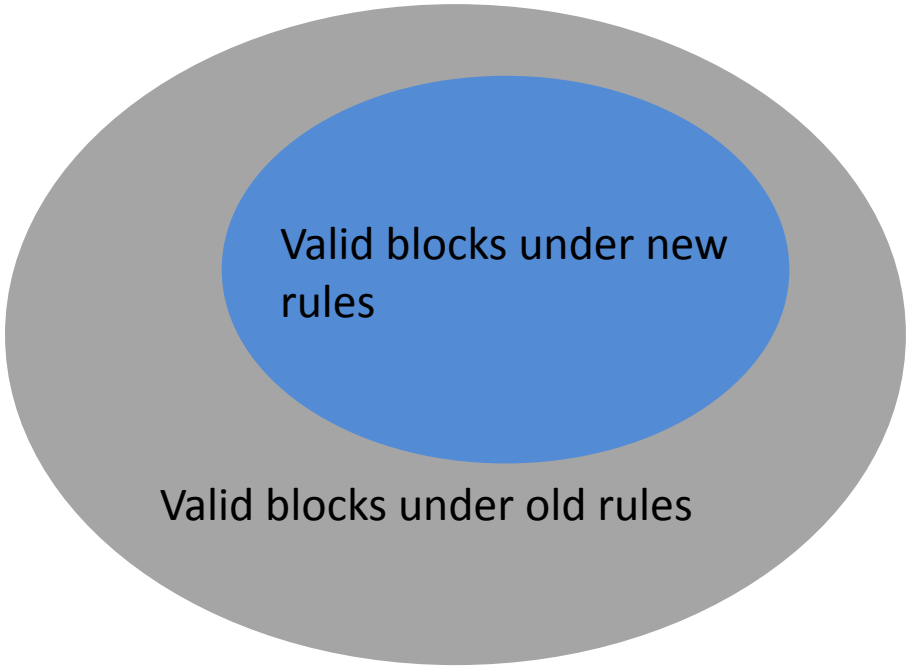
prev: 0092
txns
nonce: fd1a

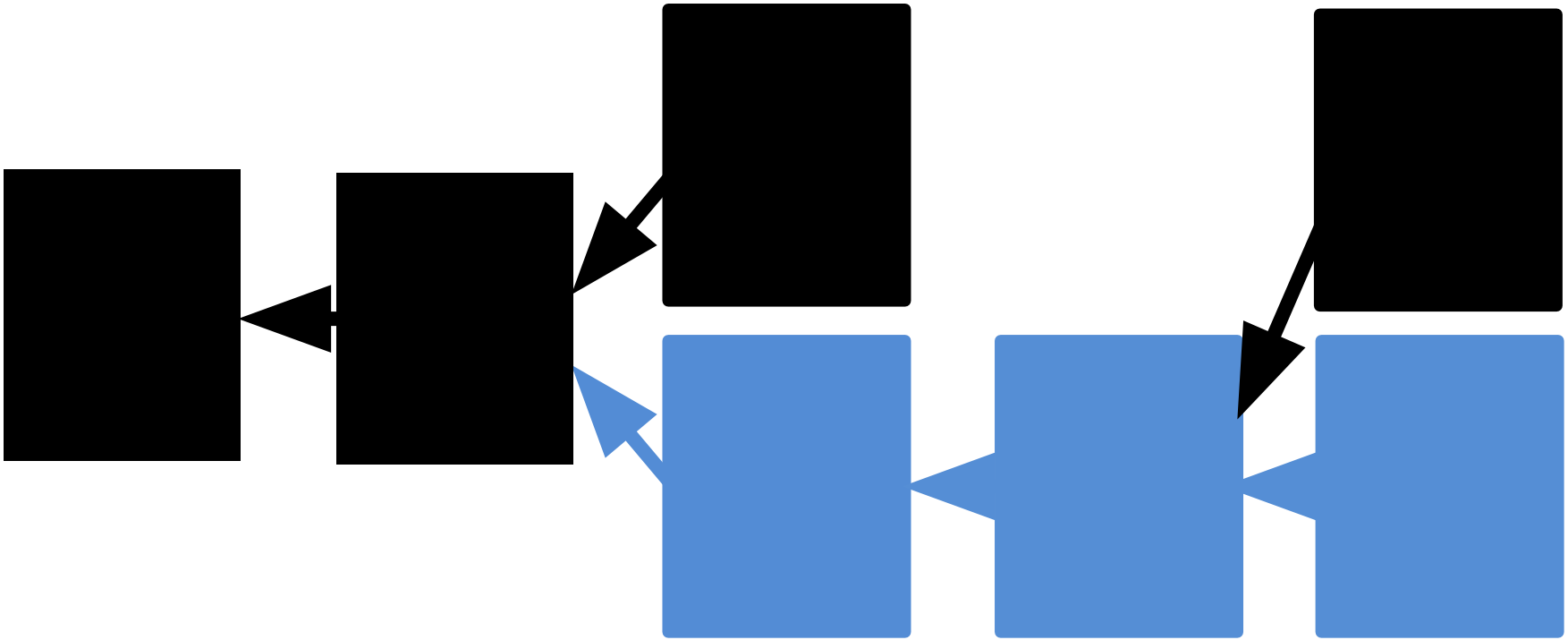
hash: 002b


prev: 00ce
txns
nonce: 5ffc
hash: 00db
prev: 00db
txns
nonce: 582c
hash: 0092
prev: 0092
txns
nonce: fd1a
hash: 002b
prev: 002b
txns v1
nonce: 34a8
hash: 001c
prev: 00ce
txns
nonce: 5ffc
hash: 00db
prev: 00db
txns
nonce: 582c
hash: 0092
prev: 0092
txns
nonce: fd1a
hash: 002b
prev: 002b
txns v2
nonce: ce7d
hash: 004d

Soft forks

- Backwards compatible
- Only adding new rules: Old-rule nodes will see new-rule blocks as valid

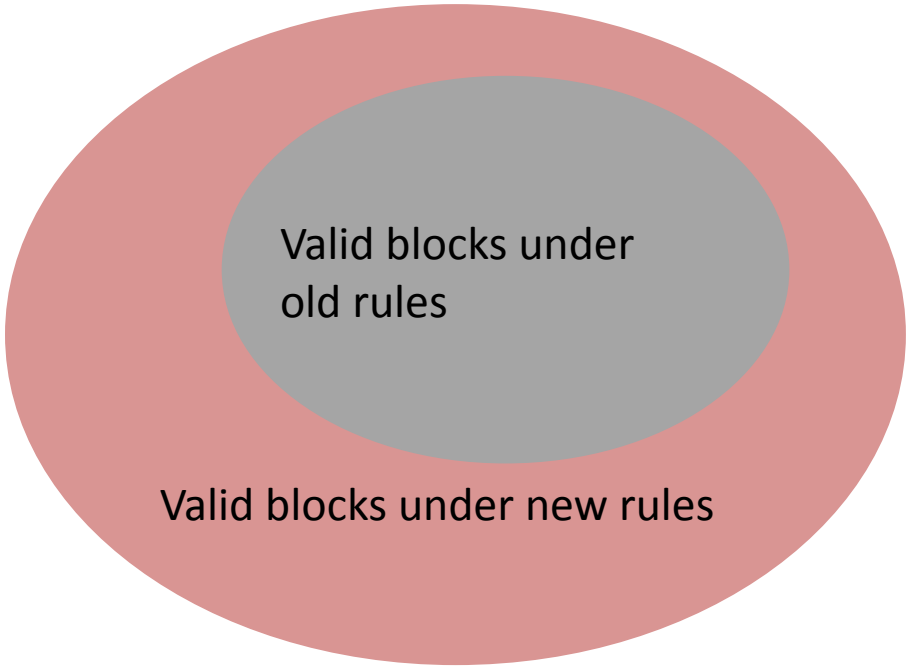


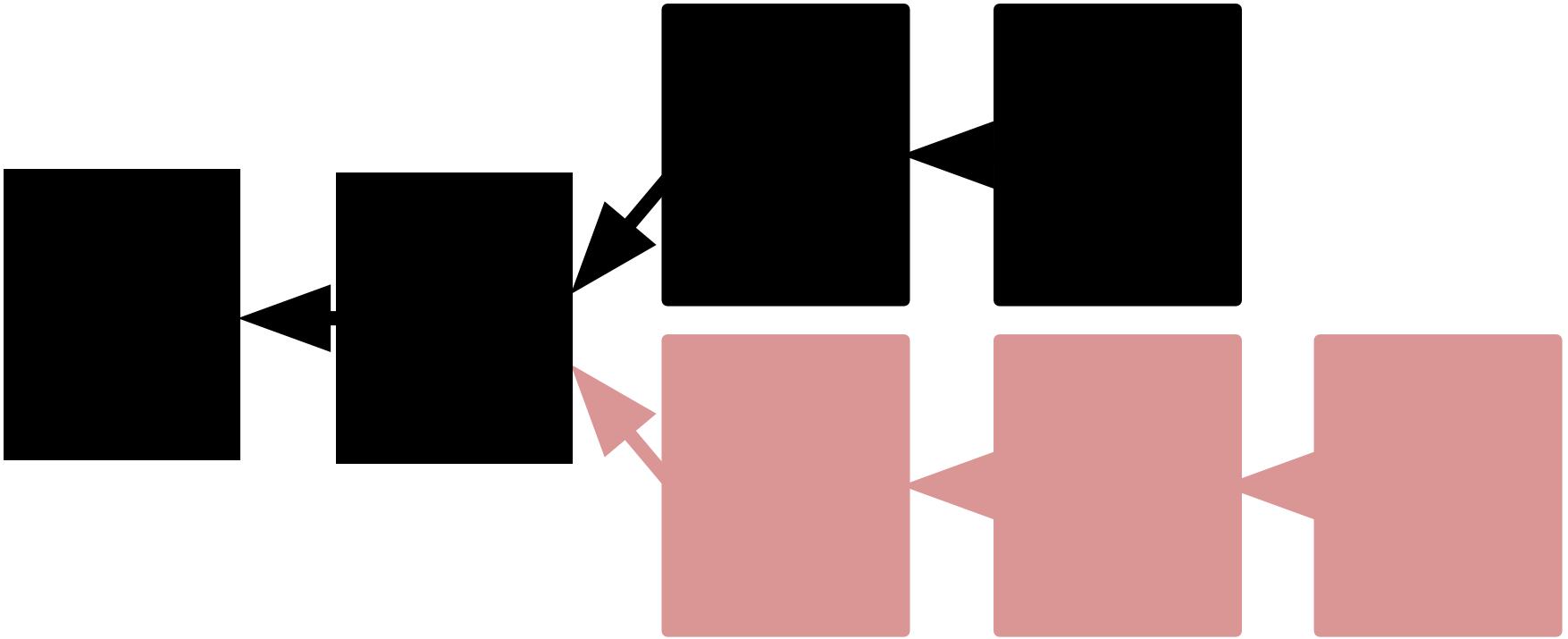


Miners who don’t upgrade might produce invalid blocks, but they will be orphaned

Hard forks

- Not backwards compatible
- Removing rules: Old-rule nodes will NOT see new-rule blocks as valid





Two chains, possibly forever.

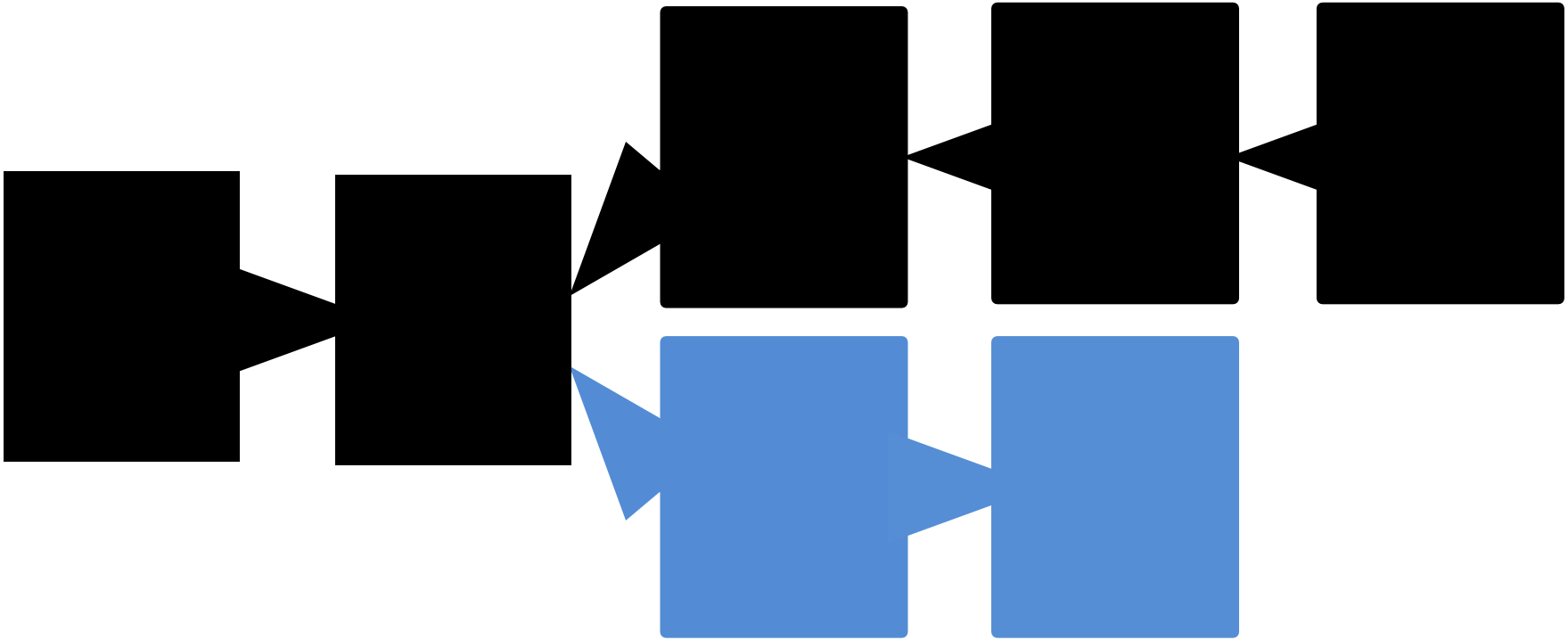
Hard fork vs. Soft fork

- Hard forks are NOT backwards compatible
- Some prefer only soft forks so as not to “lose” nodes that don’t upgrade
- Others see soft forks as coercive

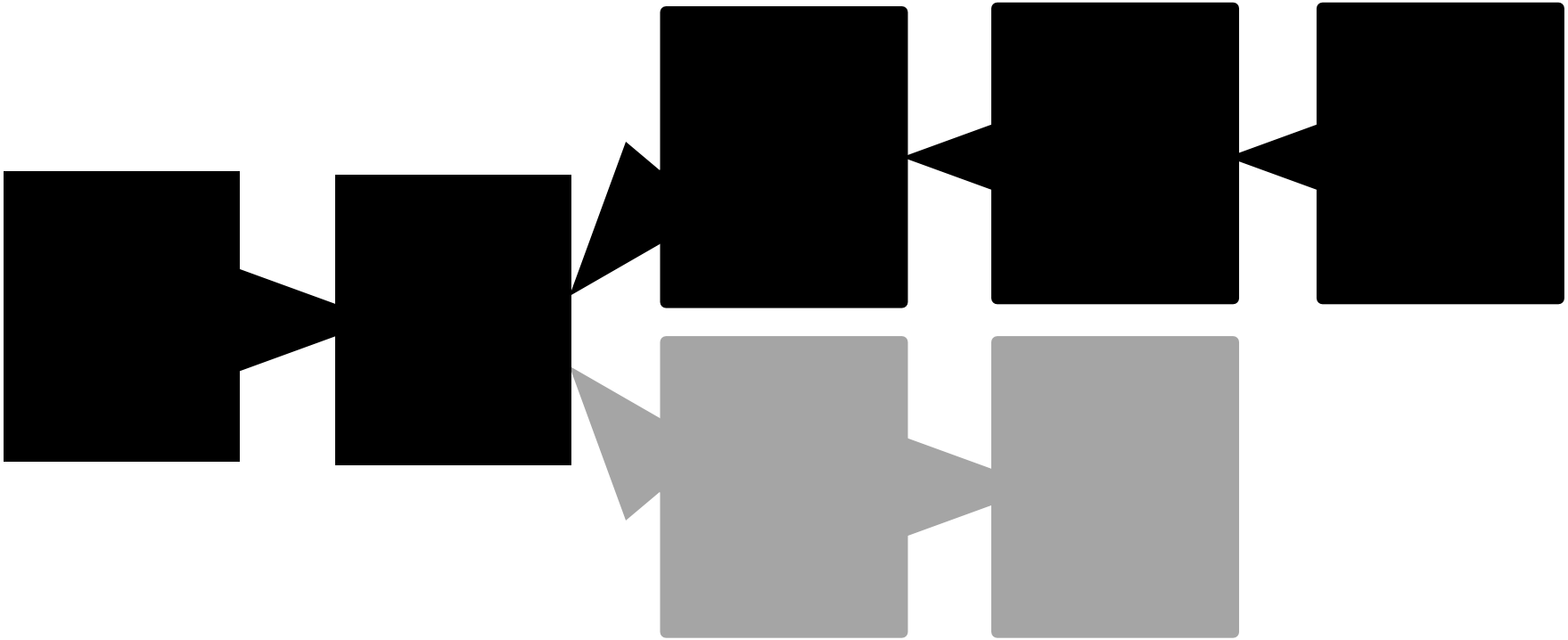
Who controls forks?

Developers | Miners | Nodes

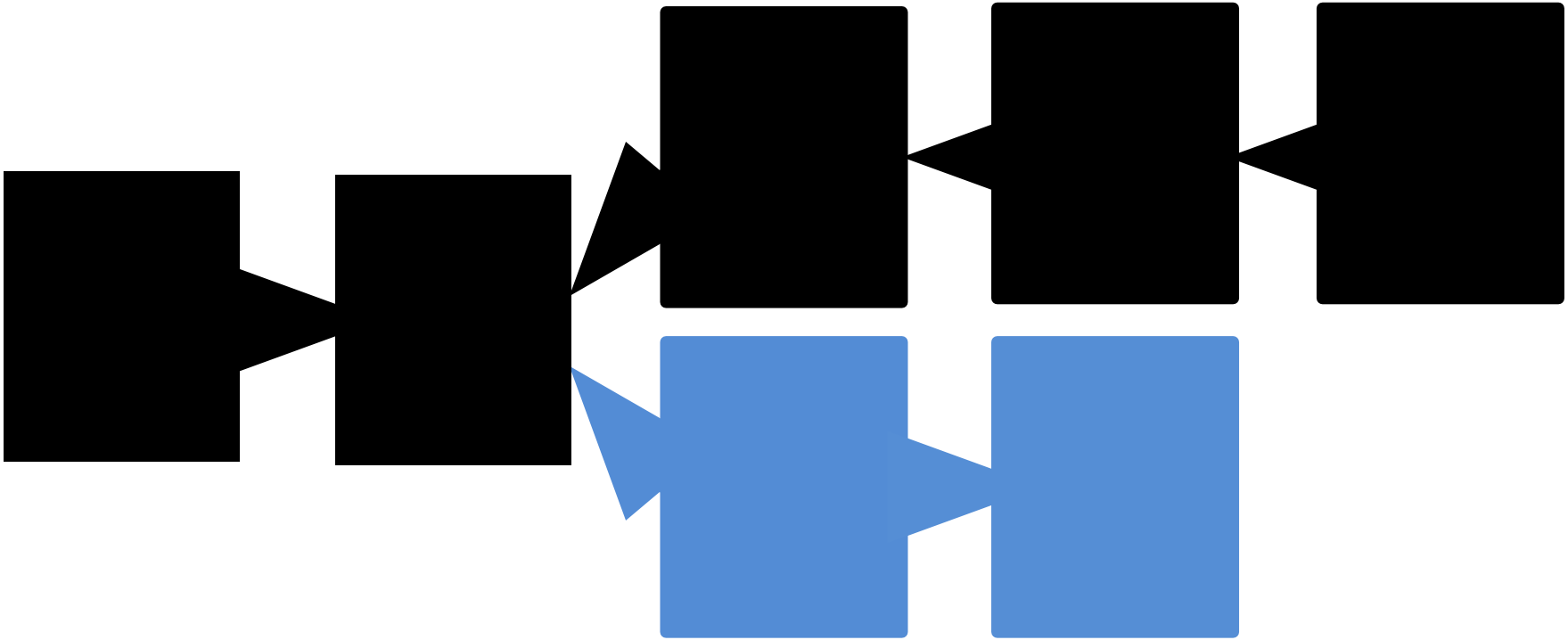
- Developers write new software
- Miners install the software (or not) and create (and validate) blocks
- Nodes install the software (or not) and validate blocks



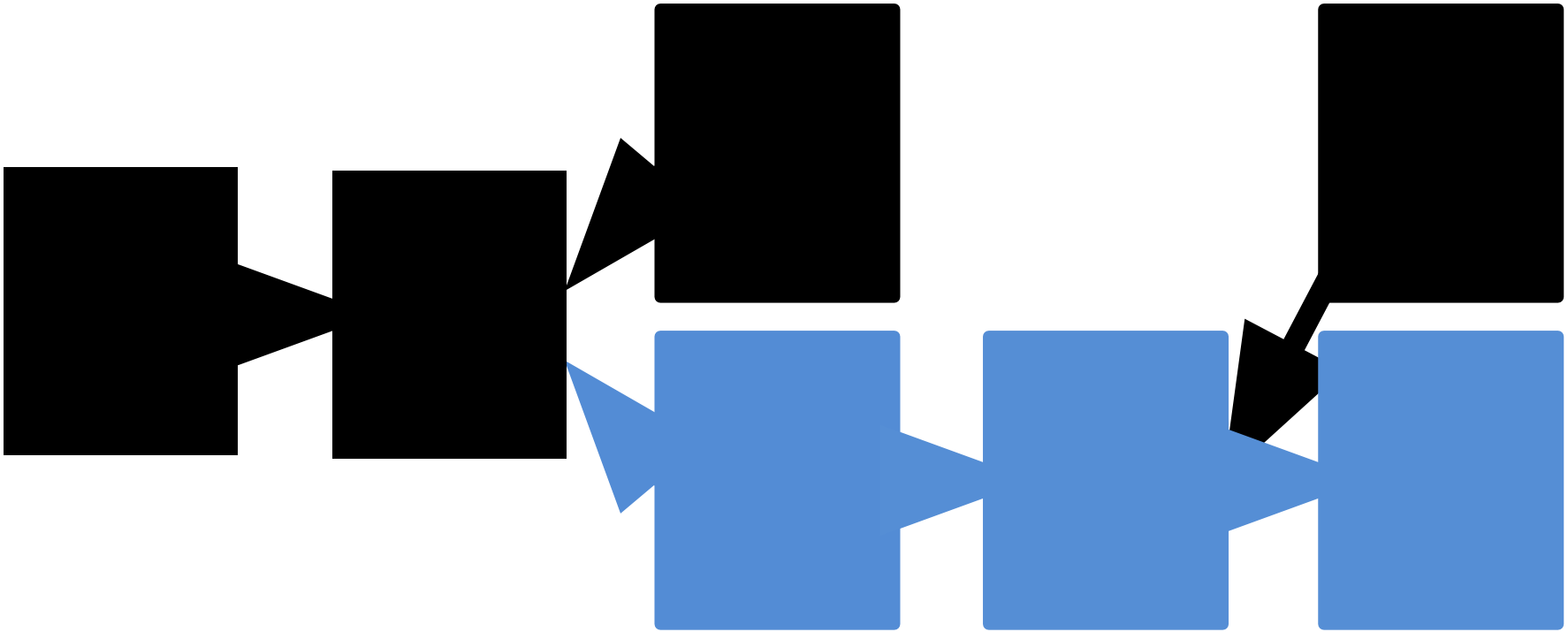
What happens if a soft fork doesn’t obtain > 50% of hash rate?



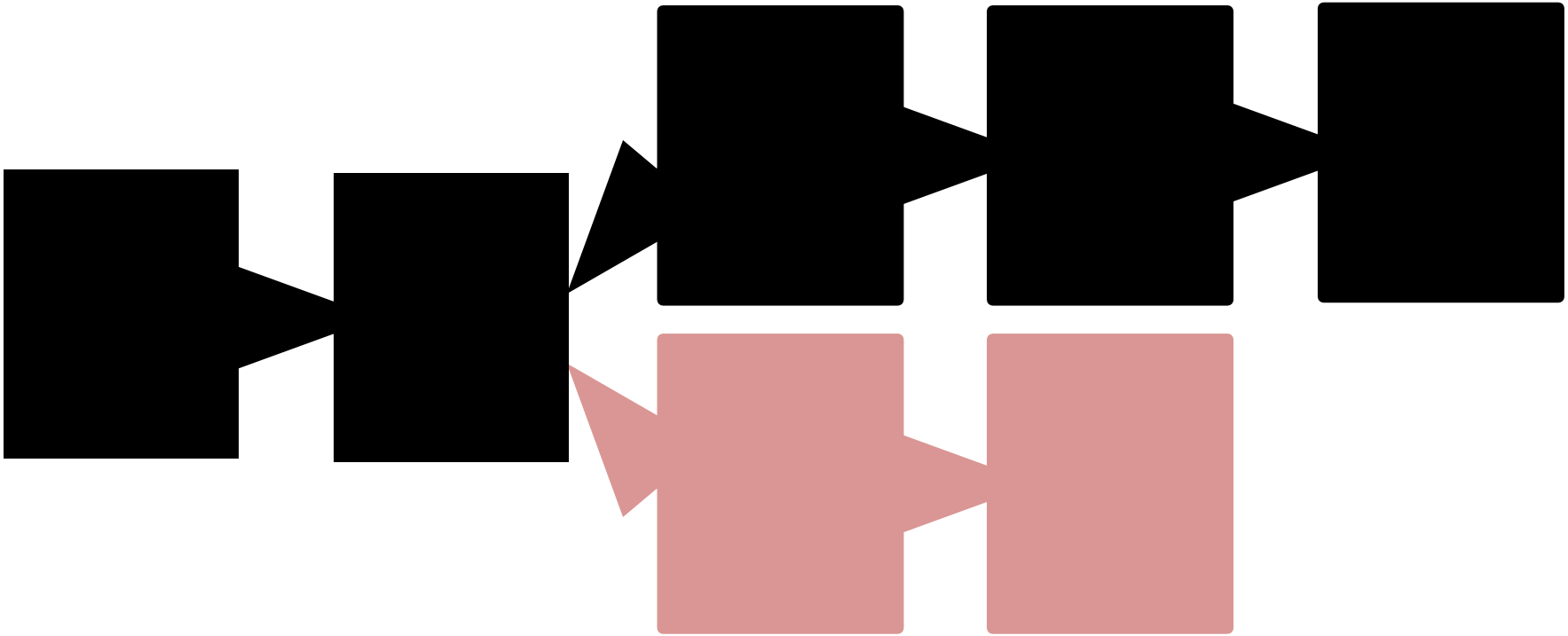
Depends on the soft fork! If old-rule blocks are still valid, soft fork gets reorg’d out



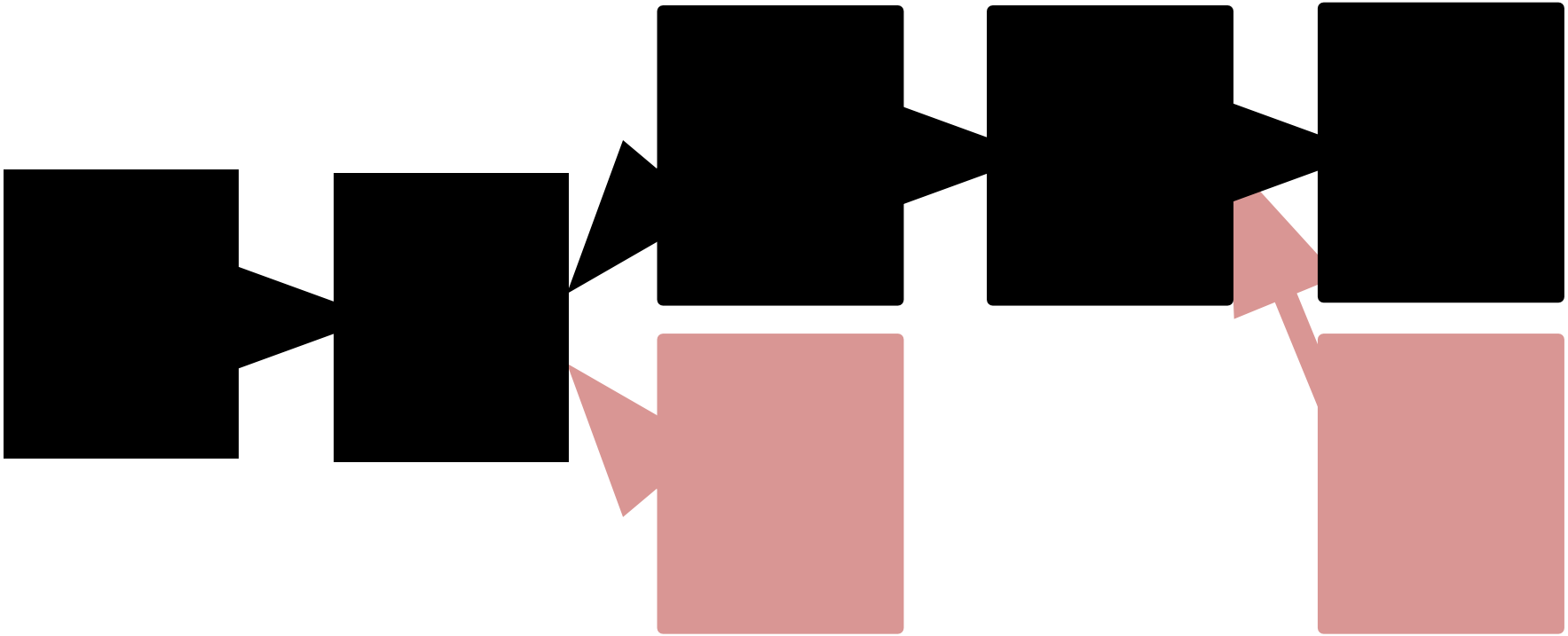
If old-rule blocks are now invalid, fork will persist



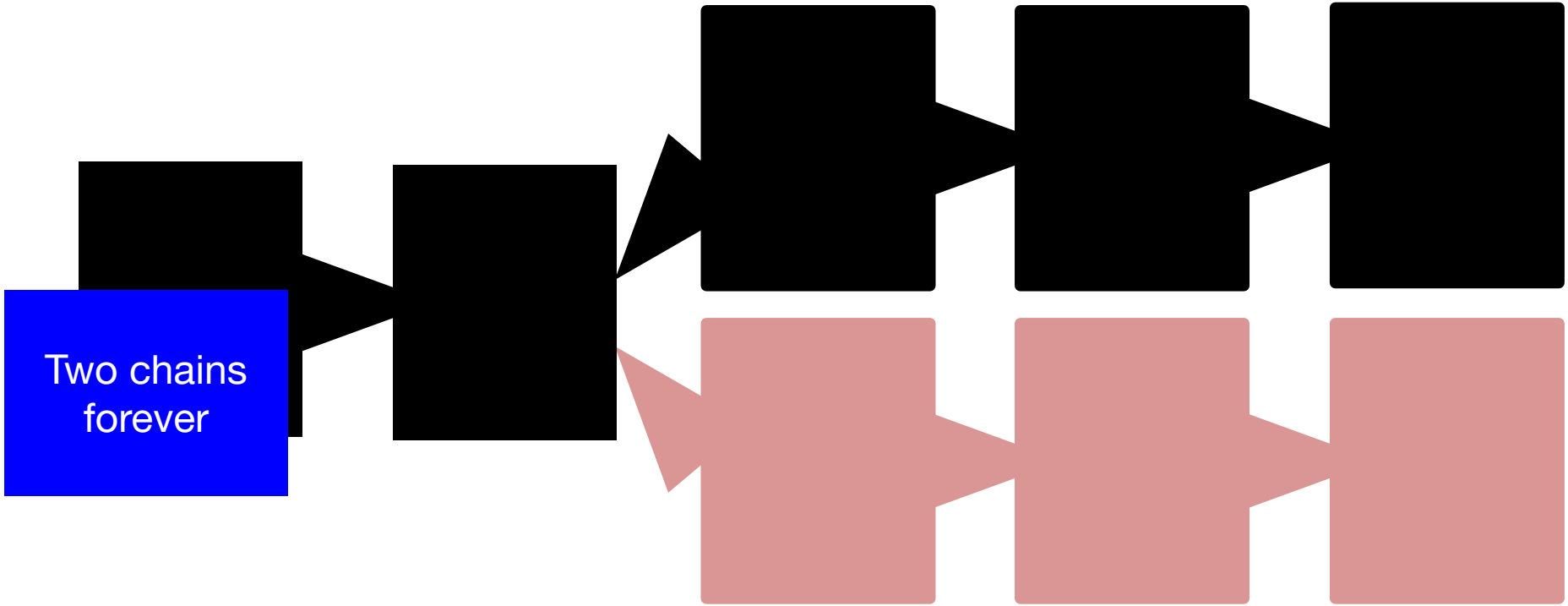
If soft fork > 50%, old-rule blocks will follow new fork automatically



What happens if a hard fork doesn’t obtain > 50% of the hash rate?



Again depends, but if old-rule blocks are still valid, new-rule nodes will follow along



What happens if a hard fork does obtain > 50% of the hash rate?

Soft forks in practice

- Lots! P2SH, Segwit, Taproot
- Big conversation right now about next soft fork in Bitcoin (and if there should even be any more non-bug-fixing soft forks)

Hard forks in practice

- New Bitcoins (Bitcoin Cash, Bitcoin Gold, Bitcoin Diamond)
- Ethereum DAO hard fork (unplanned)
- Ethereum upgrades (planned)
- Some cryptocurrencies hard fork frequently

Ethereum DAO hard fork

- Block 1920000 transferred ~12M ETH from one set of accounts to another for reclamation
- 85% of mining power went along with it
- Two currencies: ETH and ETC (~226:1 today)

Downsides of proof-of-work

- Proof-of-work is very costly
  - High energy usage
- Probabilistic finality
- High latency
- Attacks: Selfish mining

Summary

- Forks are extremely challenging
- Blockchains have different challenges than traditional consensus
- Assignment #1 due tonight
- Next class: Programmability!