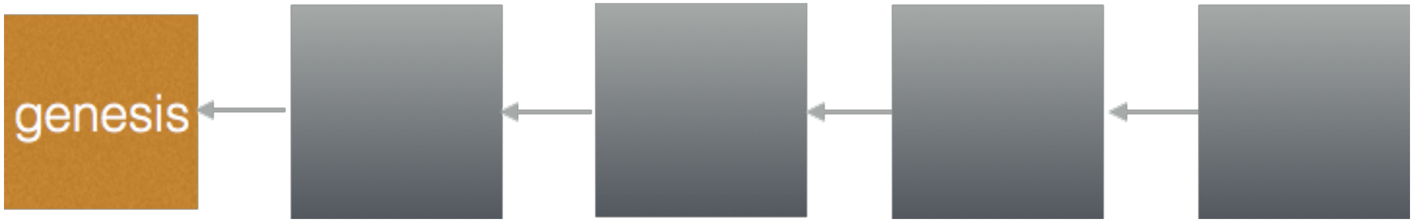
The Blockchain Abstraction

THE UNIVERSITY OF
SYDNEY

The Blockchain abstraction

Let G = ⟨B, P⟩ a directed acyclic graph (DAG) where blocks B point to each other with pointers P

⟨b0, b1⟩ ∈ P is a pointer from current block b1 to previous block b0



The *pointer* is a representation of a hash of the destination block that the source block contains

The *genesis block* is a special block known initially by all participants

Byzantine Consensus

THE UNIVERSITY OF
SYDNEY

Consensus

Consensus problem is having non-faulty nodes agree on one value:

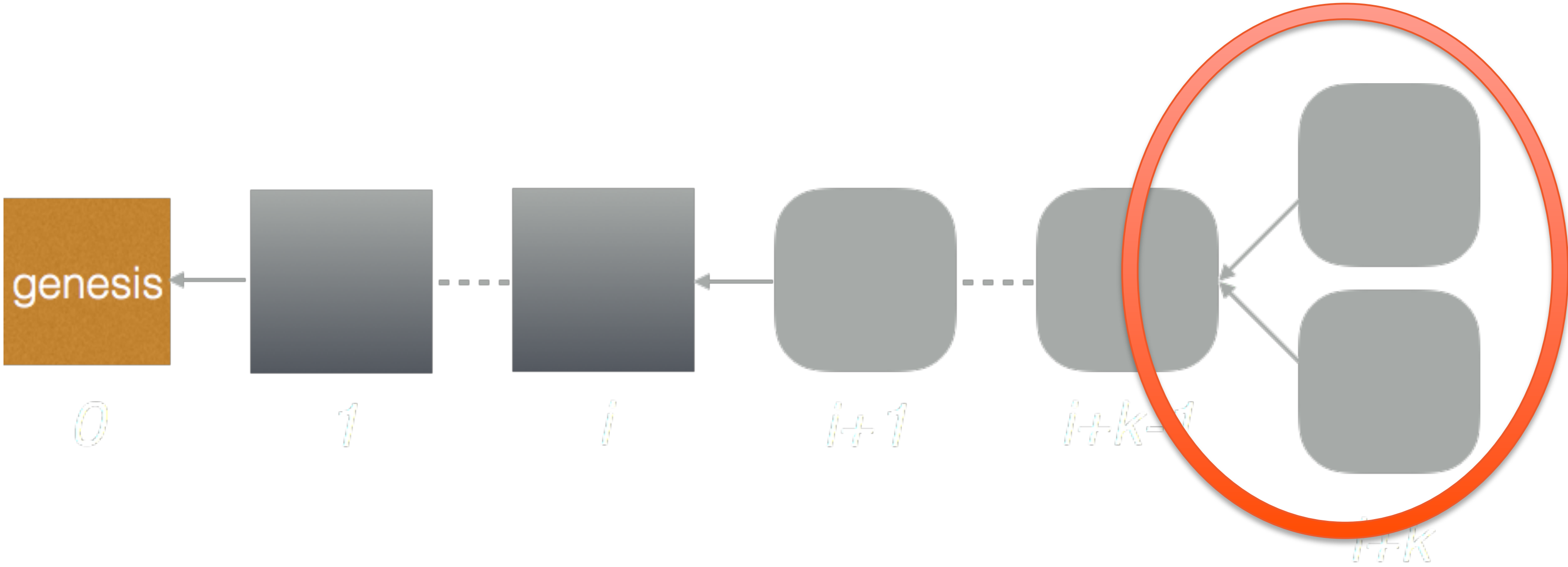
1. Agreement: all nodes that decide choose the same value;
2. Validity: the output value is an input value of a correct node;
3. Termination: all correct nodes eventually decide.

Monte Carlo Consensus:

1. Aggrement with some probability greater than a threshold
2. Validity
3. Termination

Why is consensus needed in blockchain?

Consensus is necessary to totally order the blocks, maintaining the *chain*



Disagreement ⇒ no chain but a DAG

Failure model

Model

- n nodes in the system
- f are faulty

Failure model

Model

- n nodes in the system
- f are faulty

Because blockchains protect ownership, there is an incentive for an attacker to steal the goods of others

- The fault model is Byzantine (i.e., arbitrary)

Solution for Byzantine consensus

Limiting the number f of failures is key to solving consensus

There are solutions when f < n/3 [CL02]

[CL02] M. Castro and B. Liskov. Practical byzantine fault tolerance and proactive recovery. ACM Trans. Comput. Syst., 20(4):398{461, Nov. 2002.

Proof-of-Work

THE UNIVERSITY OF
SYDNEY

Sybil attack

A *Sybil attack* is an attack where a malicious user forges identities

It is named after the subject of the book *Sybil*, a case study of a woman diagnosed with dissociative identity disorder.

Some solutions [CL02] are prone to Sybil attacks where an adversary generates fake faulty nodes to have $f \geq n/3$ ⇒consensus impossible.

Miners

Specialised peers, called *miners*, receive a reward for verifying transactions provably solving a cryptopuzzle [Bla02] to append a new transaction block to the blockchain.

Cryptopuzzle: given a block and a threshold, a miner repeatedly:

- selects a nonce and
- applies a pseudo-random function to this block and the selected nonce

…until it obtains a result lower than the threshold.

The nonce is included in the block: getting the block takes time, but validating that the nonce is correct is easy

[Bla02] A. Black, "Hashcash - a denial of service counter-measure", Cypherspace, TR 2002. http://www.hashcash.org/papers/hashcash.pdf

Proof-of-Work

The nonce is included in the block, this is the *proof-of-work* [DN93]:

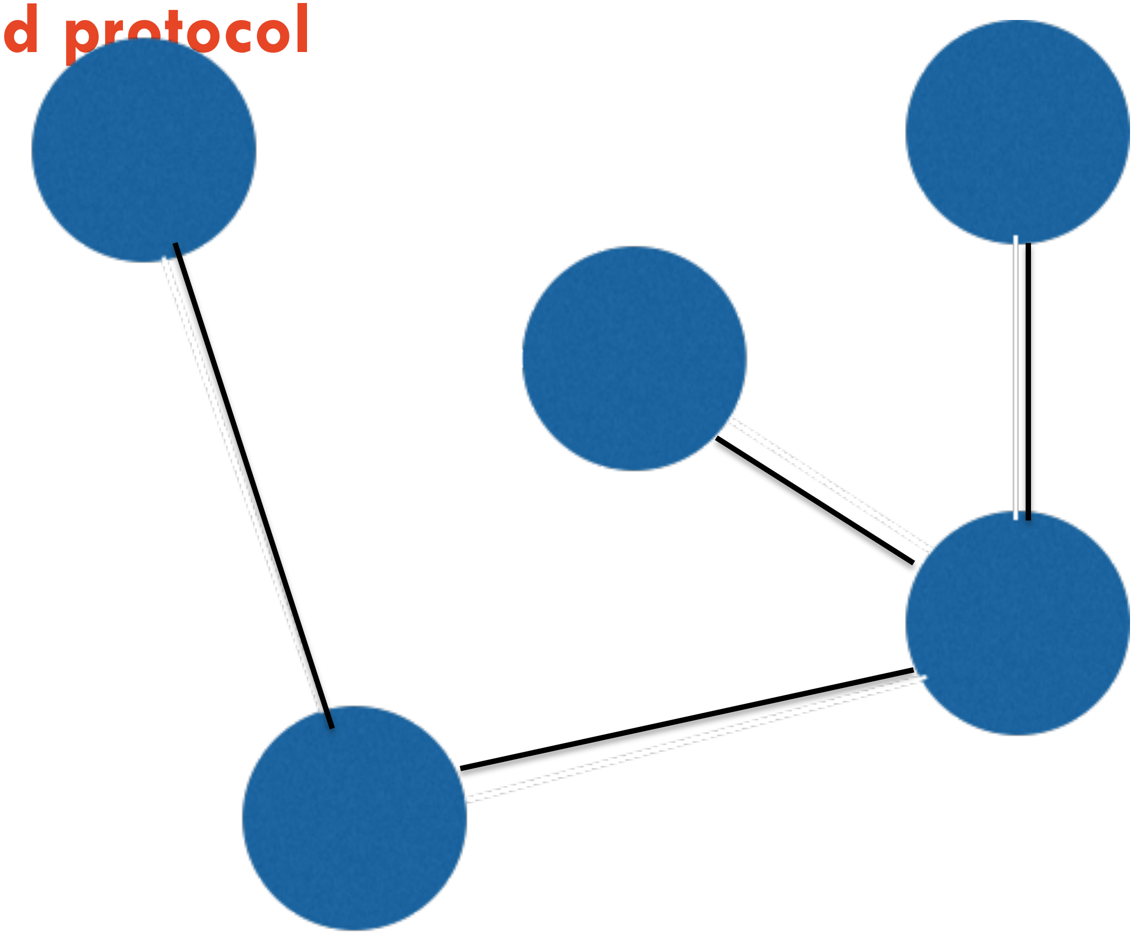
- finding the nonce takes time, but
- validating that the nonce is correct is easy.

Everyone can verify that someone lied about having solved the puzzle

[DN93] C. Dwork and M. Naor. Pricing via processing or combatting junk mail. In Proceedings of the 12th Annual International Cryptology Conference on Advances in Cryptology, CRYPTO '92, pages 139-147, 1993.

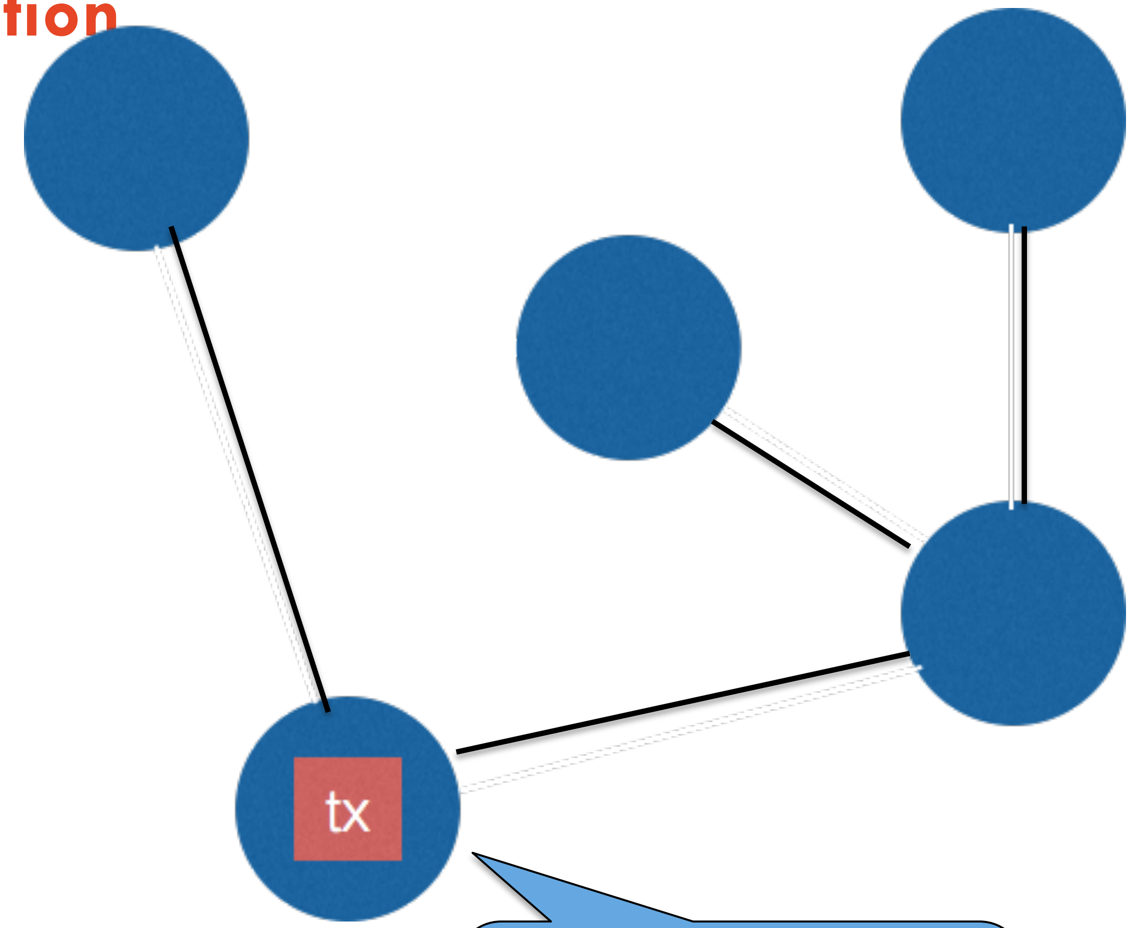
Execution

Gossip-based protocol



Current blockchain state

New transaction

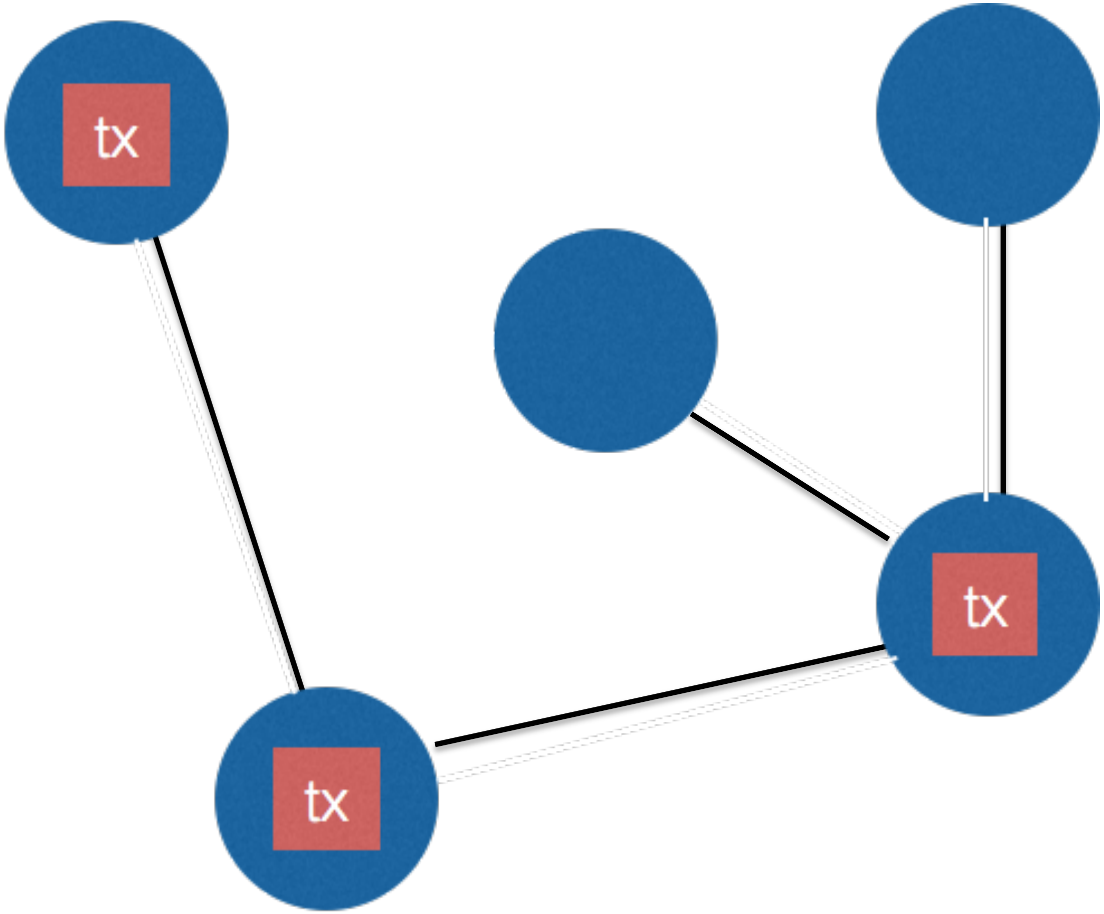


Let’s transfer 10BTC from my account to Alice’s account

genesis

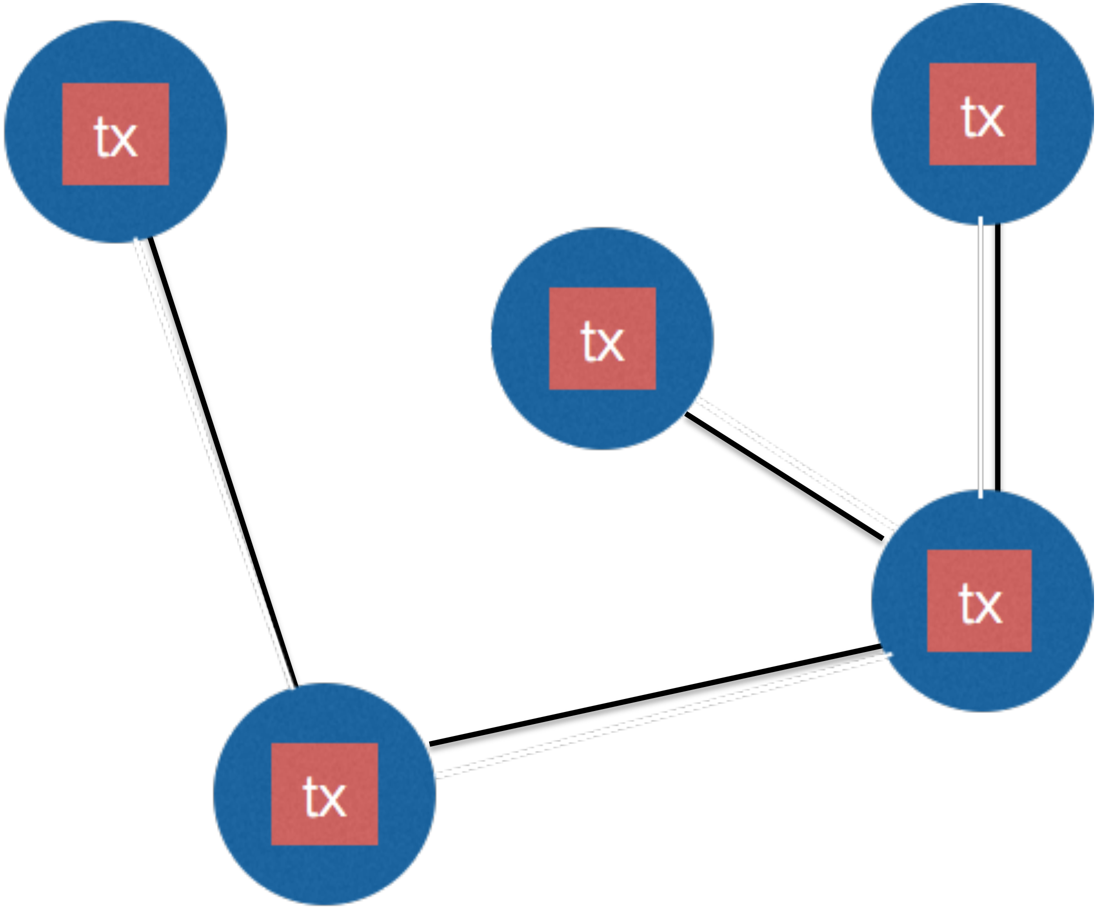
current blockchain

Broadcast



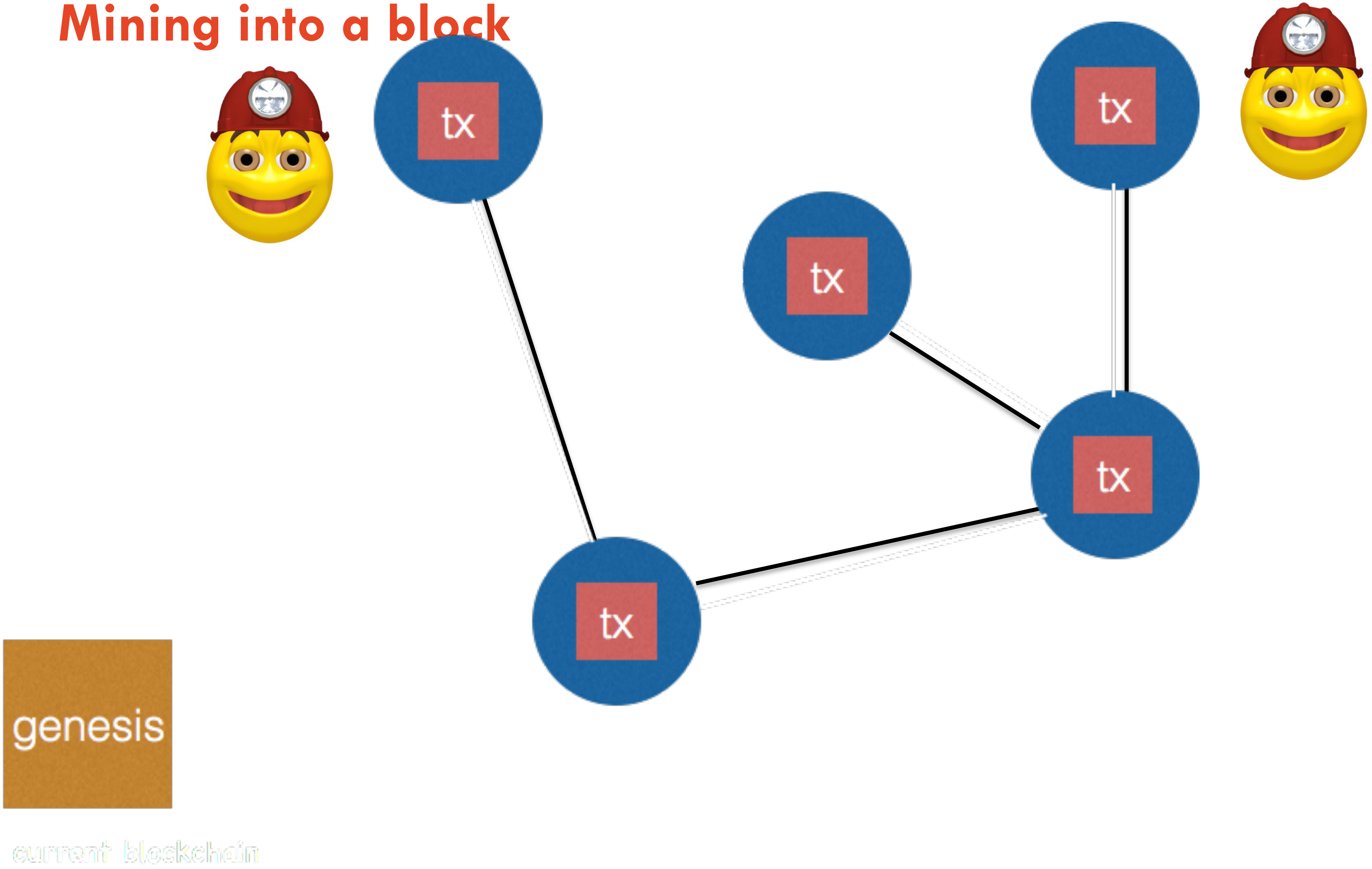
genesis
current blockchain

Broadcast

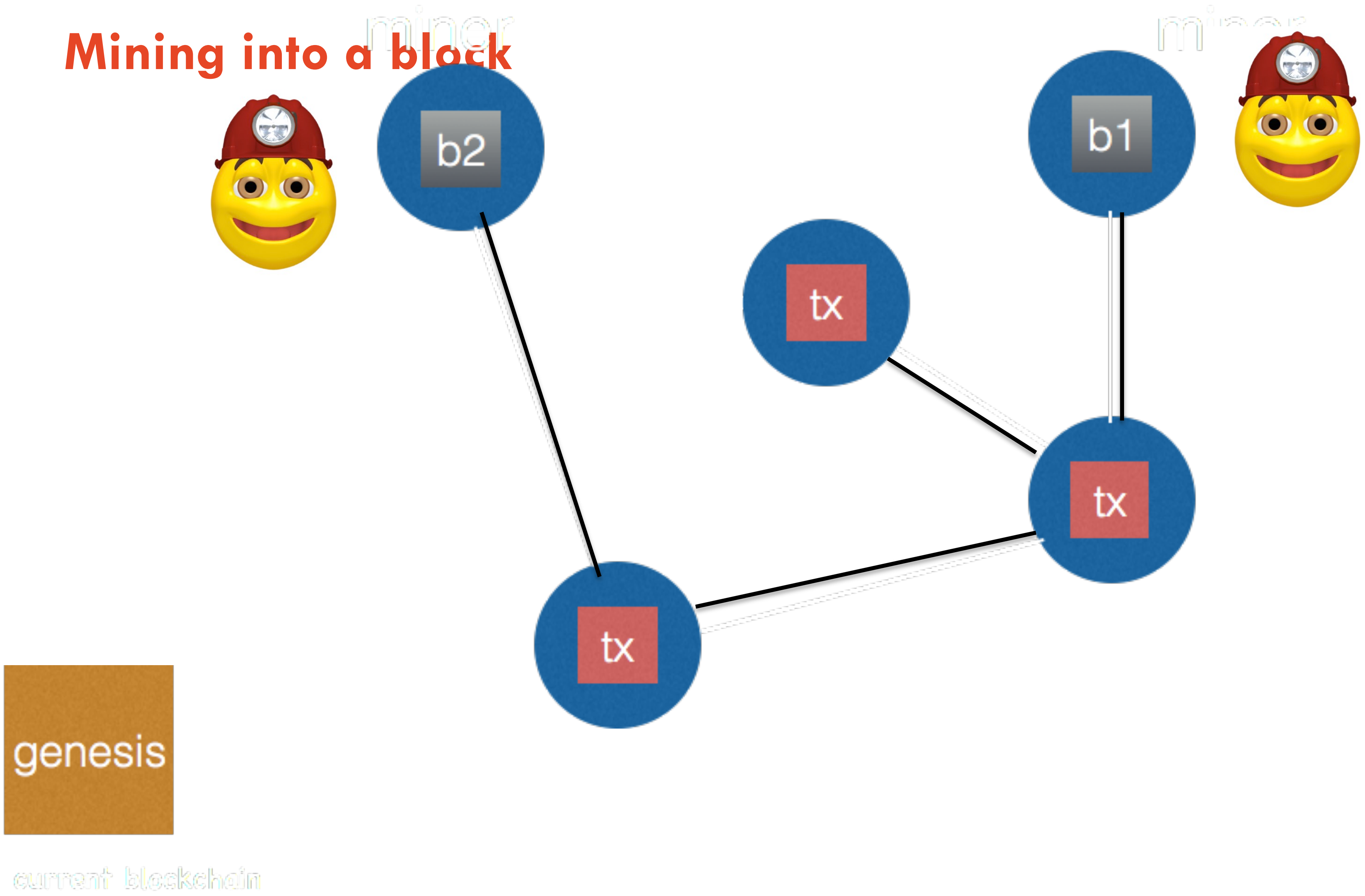


genesis
current blockchain

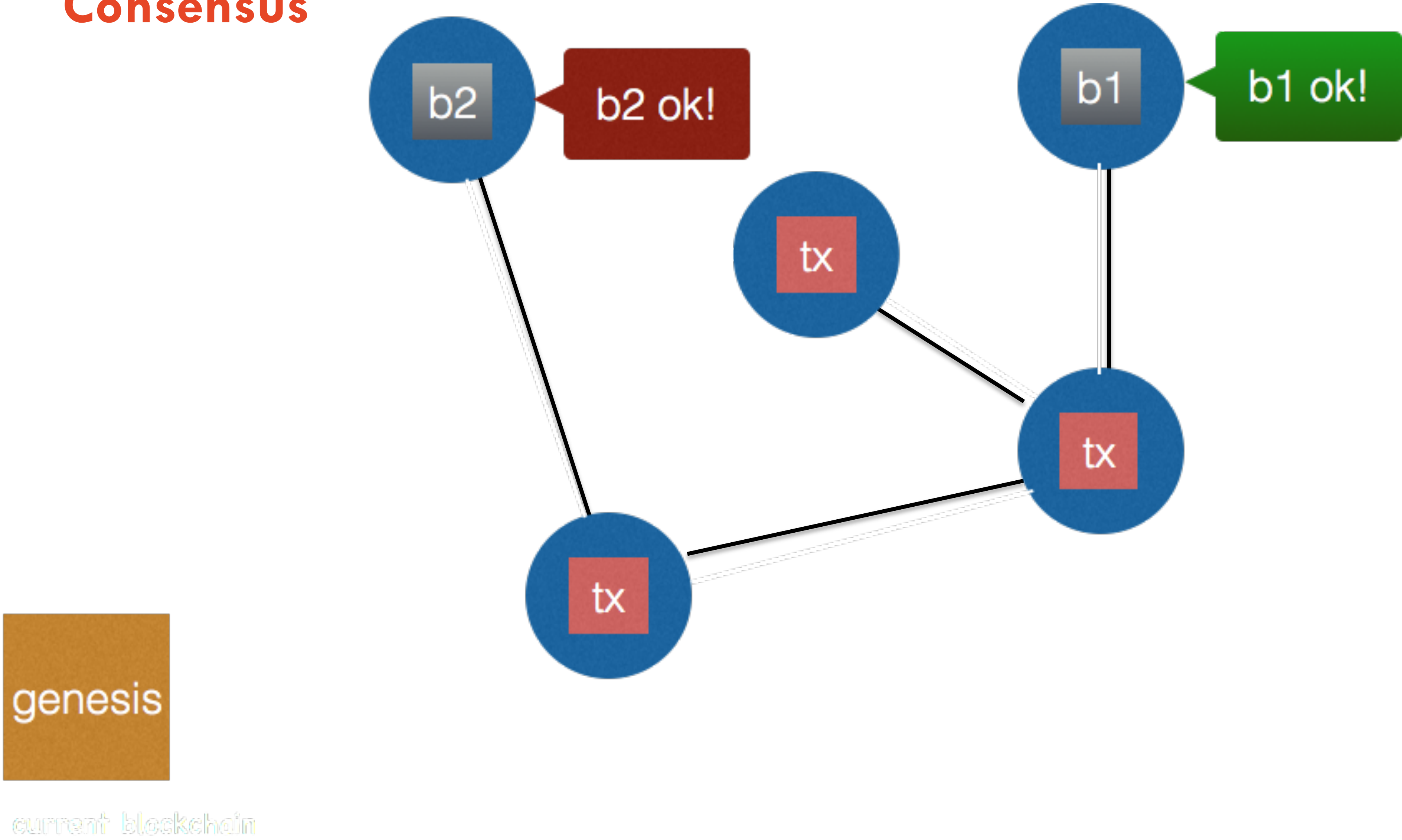
Mining into a block



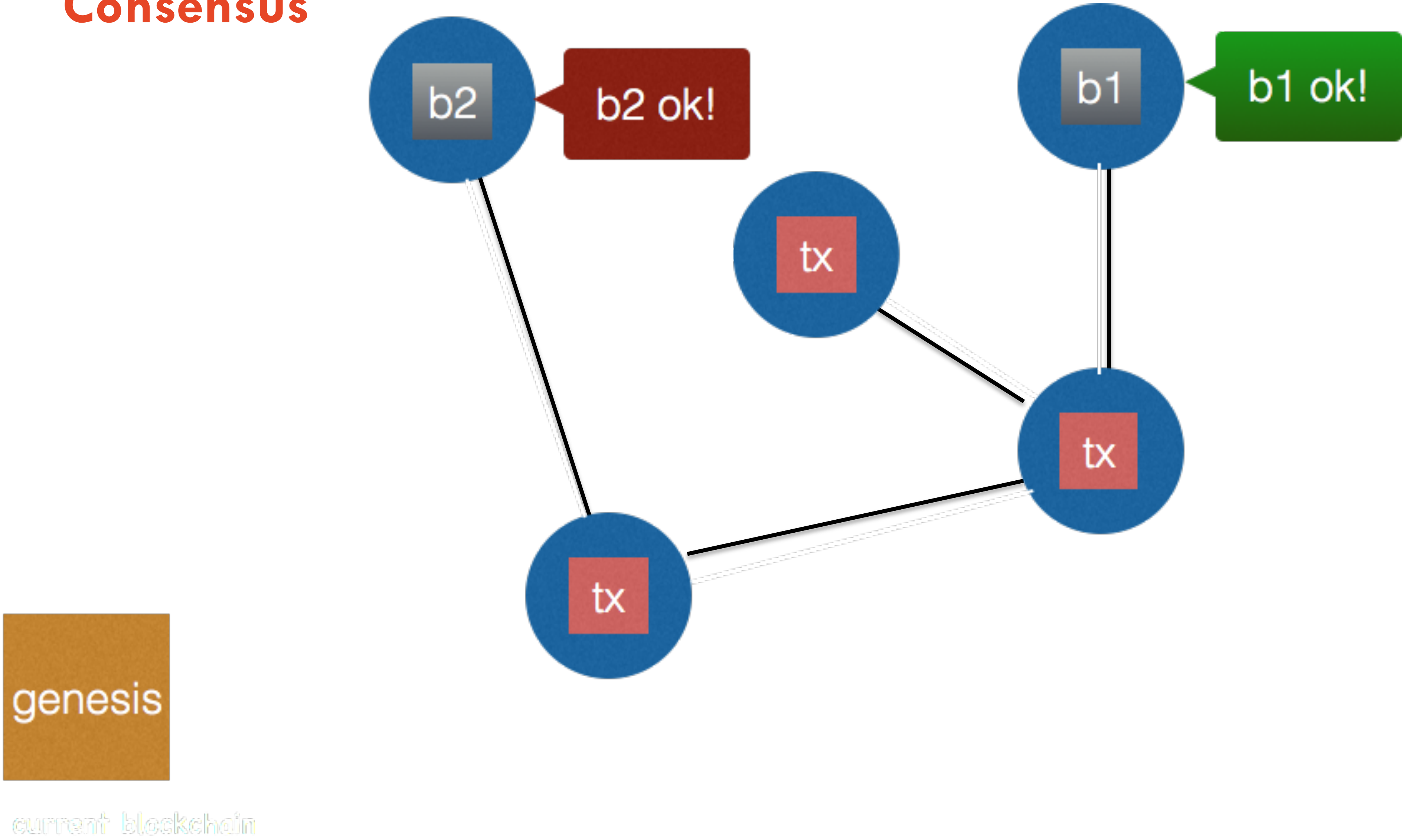
Mining into a block



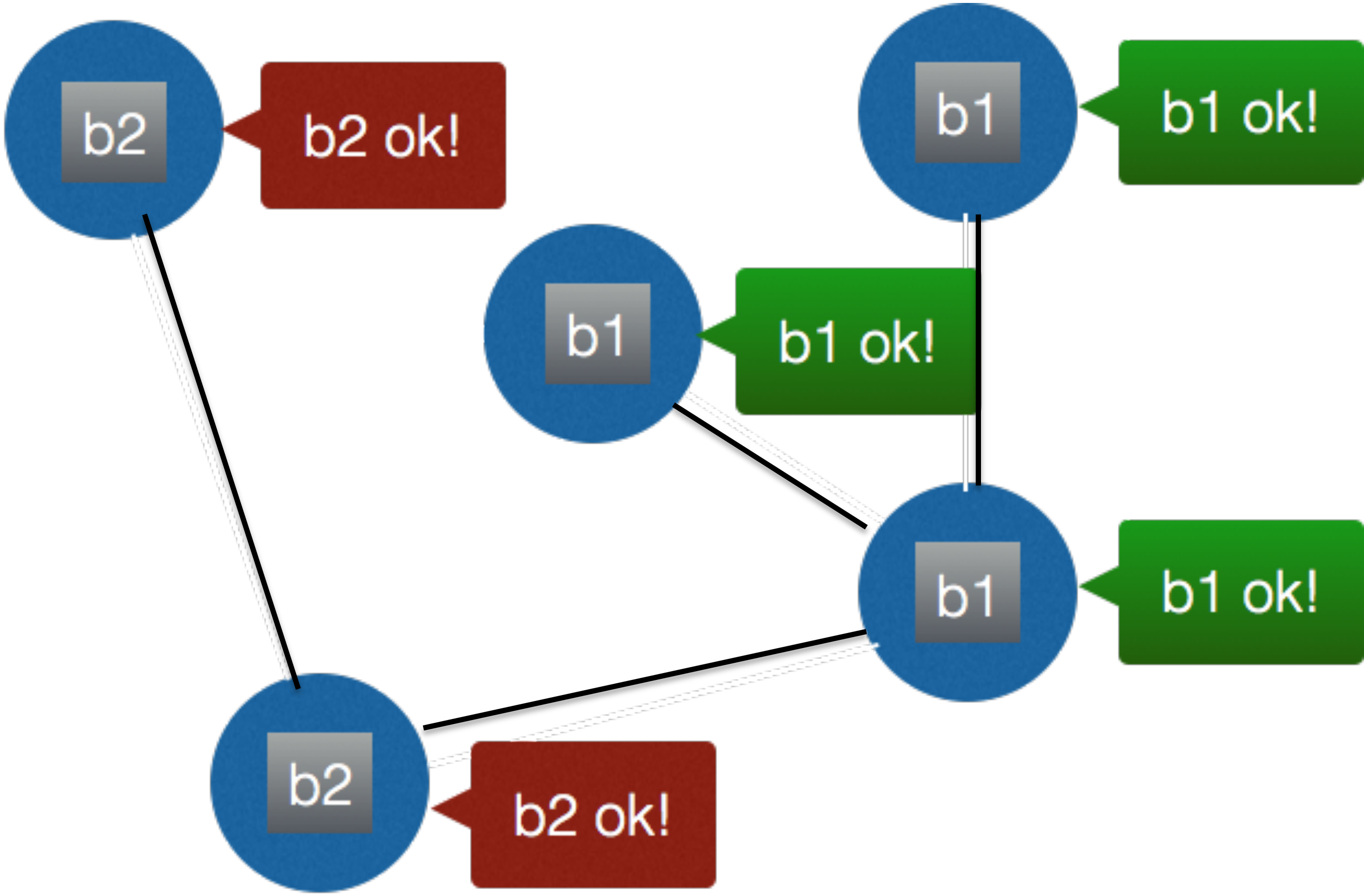
Consensus



Consensus

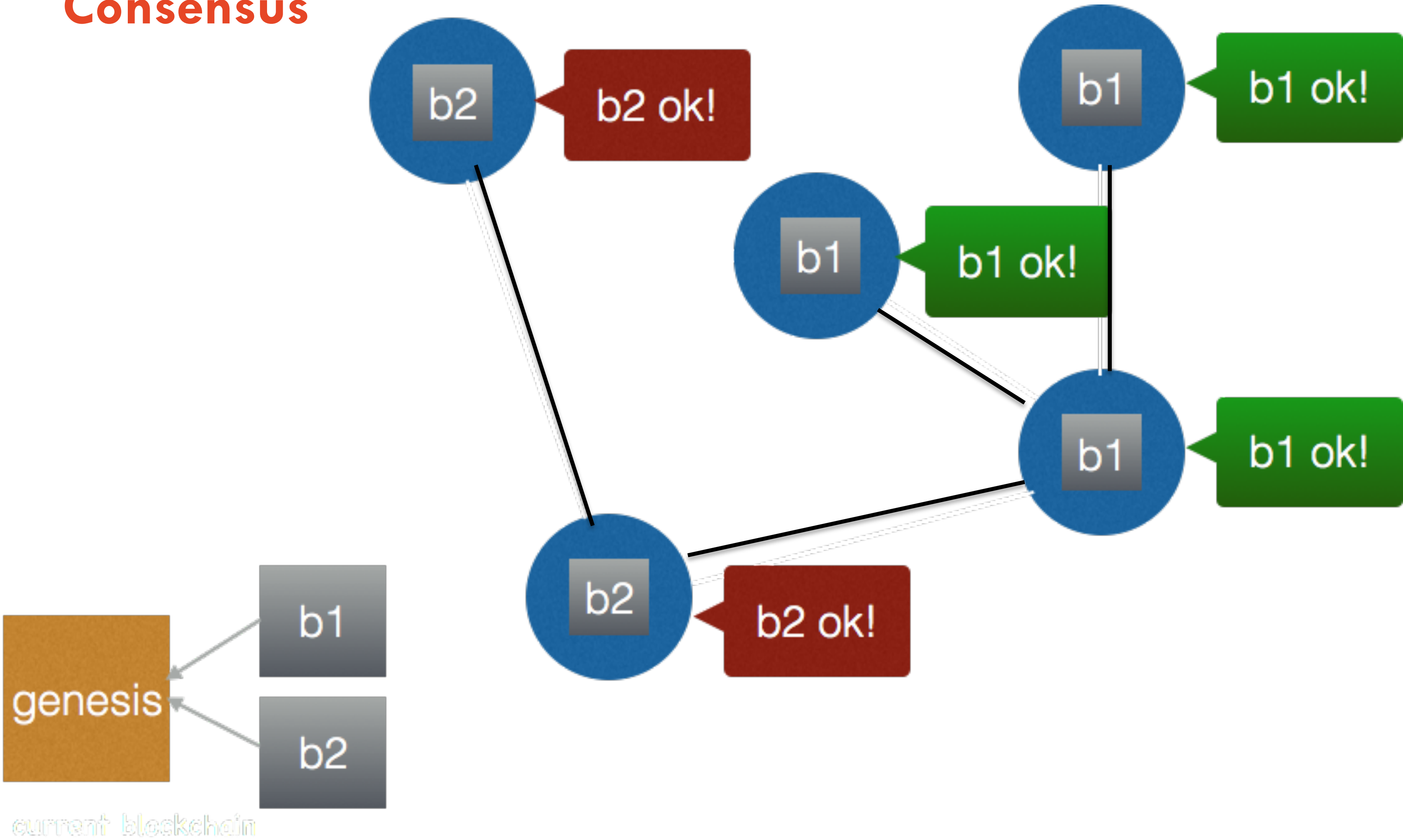


Consensus

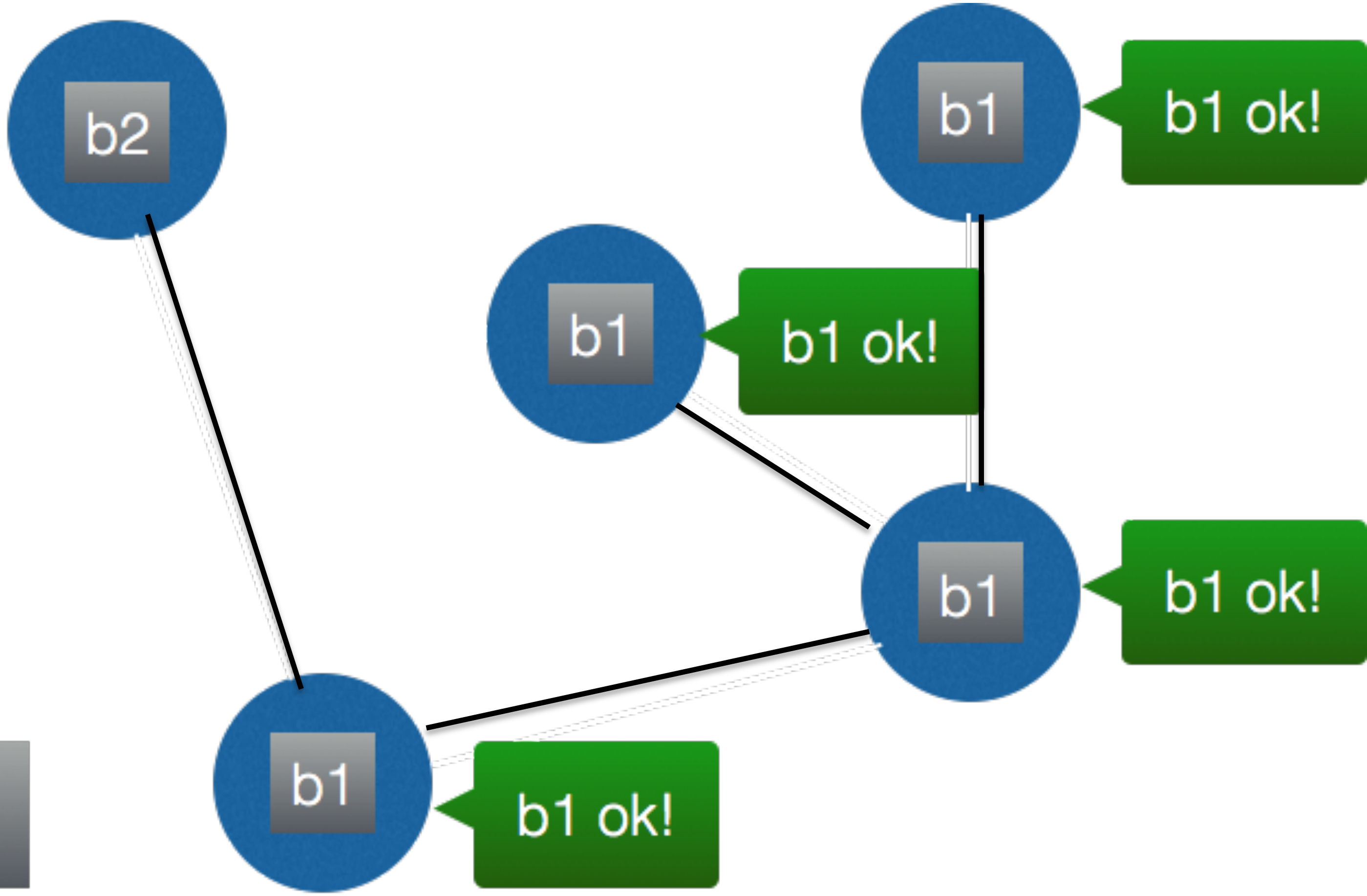

b2
b2 ok!
b1
b1 ok!
b1
b1 ok!
b1
b1 ok!
b2
b2 ok!

genesis
current blockchain

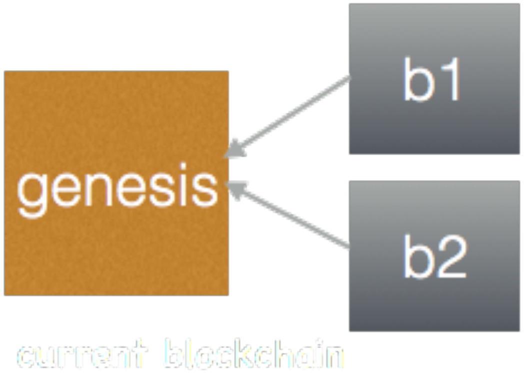
Consensus


b2
b2 ok!
b1
b1 ok!
b1
b1 ok!
b1
b1 ok!
b2
b2 ok!
genesis
b1
b2
current blockchain

Consensus

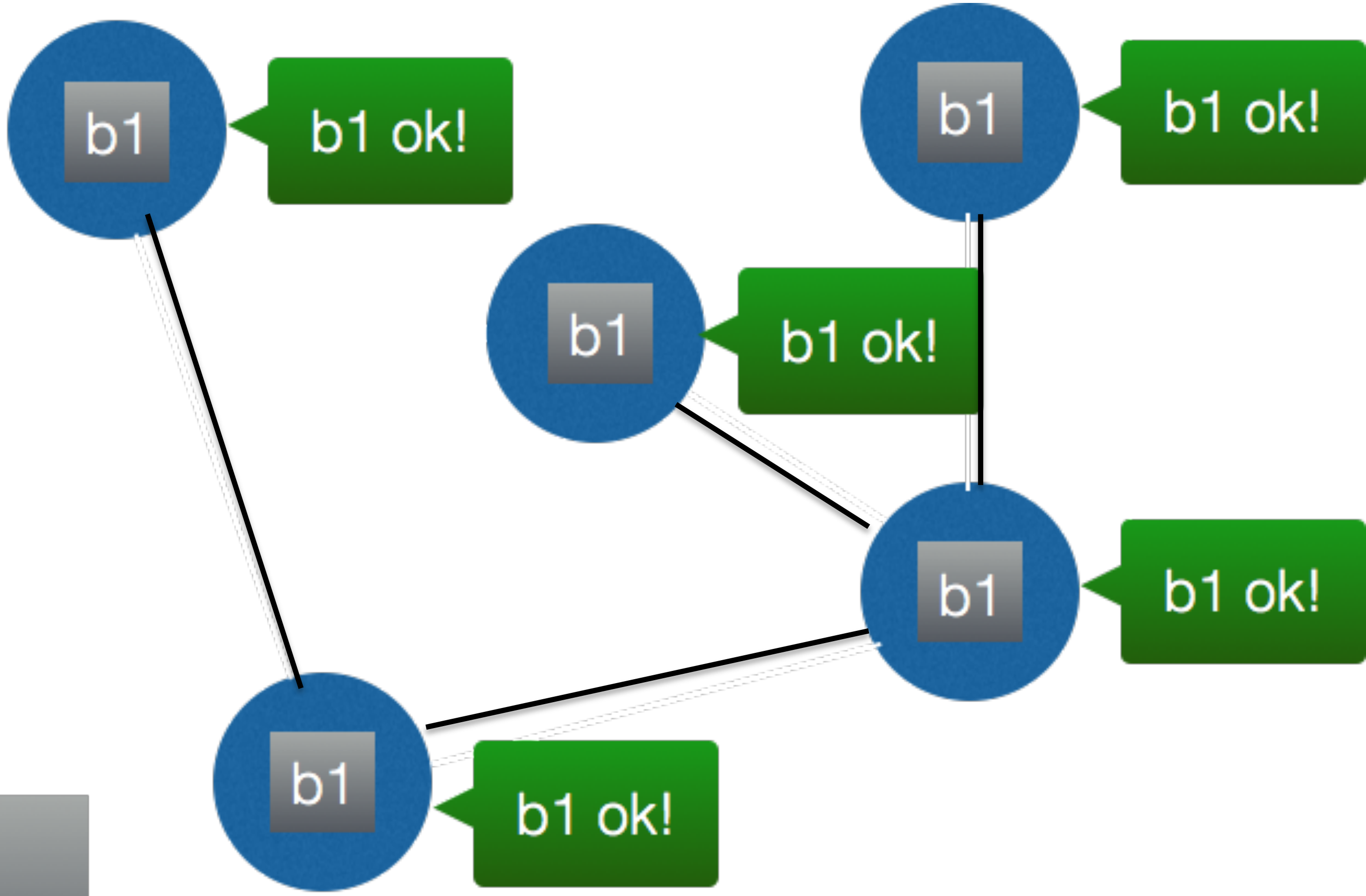


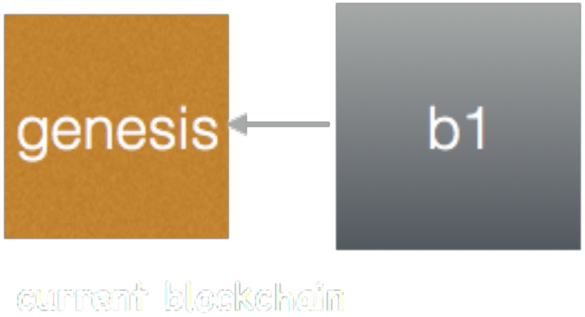
b2
b1
b1 ok!
b1
b1 ok!
b1
b1 ok!
b1
b1 ok!



genesis
b1
b2
current blockchain

Consensus

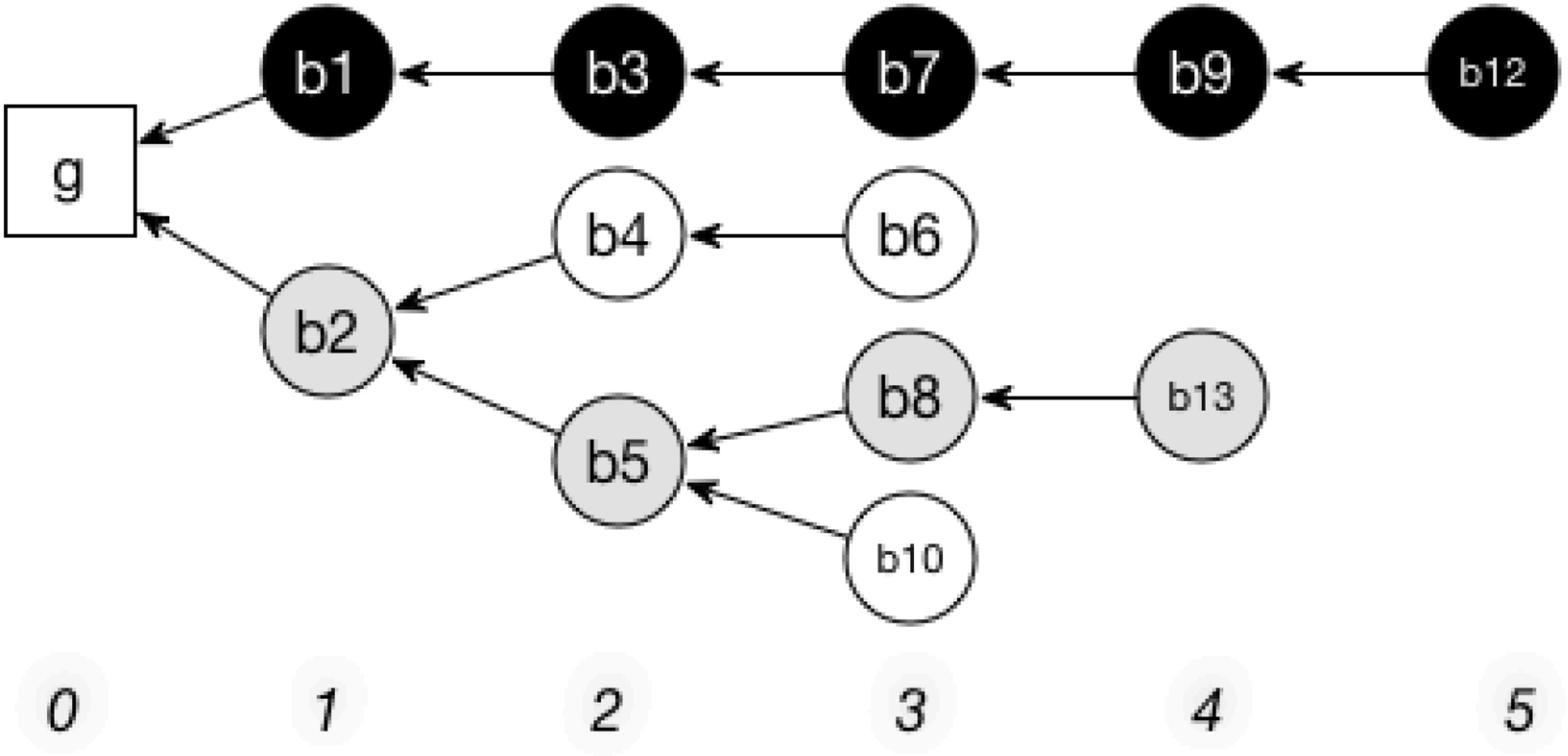

b1
b1 ok!
b1
b1 ok!
b1
b1 ok!
b1
b1 ok!
b1
b1 ok!


genesis
b1
current blockchain

Resolving a fork

THE UNIVERSITY OF SYDNEY

Assume that the current blockchain state is this DAG



Bitcoin’s consensus chooses the deepest branch

State

$\langle Bi,Pi \rangle$ the local blockchain view

Each peer of the blockchain executes:

```
Receive blocks $\langle Bj, Pj \rangle$ from j
   $Bi = Bi \cup Bj$
   $Pi = Pi \cup Pj$

depth(b):
  if $children(b) = \varnothing$ then return 1
  else return $1 + \max_{c \in children(b)} depth(c)$
```

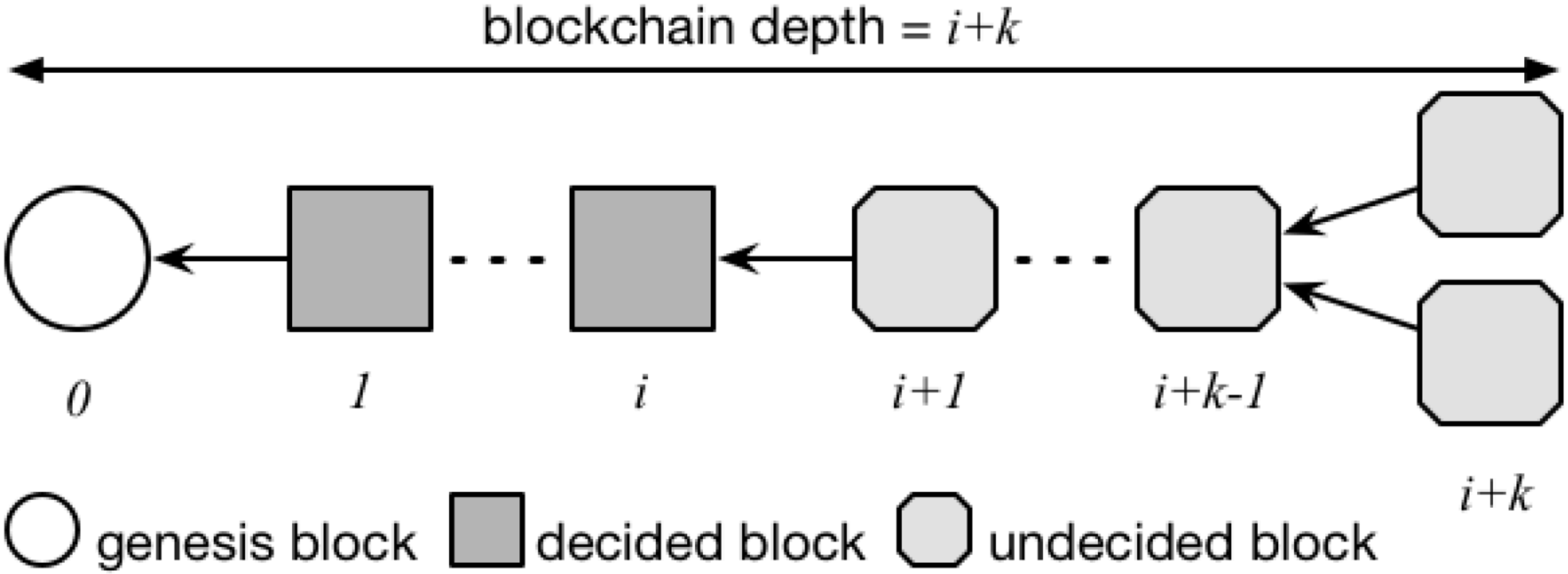
```
Prune shortest branches at i
  b = genesis-block(Bi)
  while $b.next \neq \bot$
     $block = \mathrm{argmax}_{c \in children(b)} \{ depth(c) \}$
     $B = B \cup \{block\}$
     $P = P \cup \{\langle block,b \rangle\}$
     b = block
  $\langle Bi,Pi \rangle = \langle B,P \rangle$
```

[Nak08] S. Nakamoto, “Bitcoin: a peer-to-peer electronic cash system,” 2008, http://www.bitcoin.org.

When is a transaction committed?

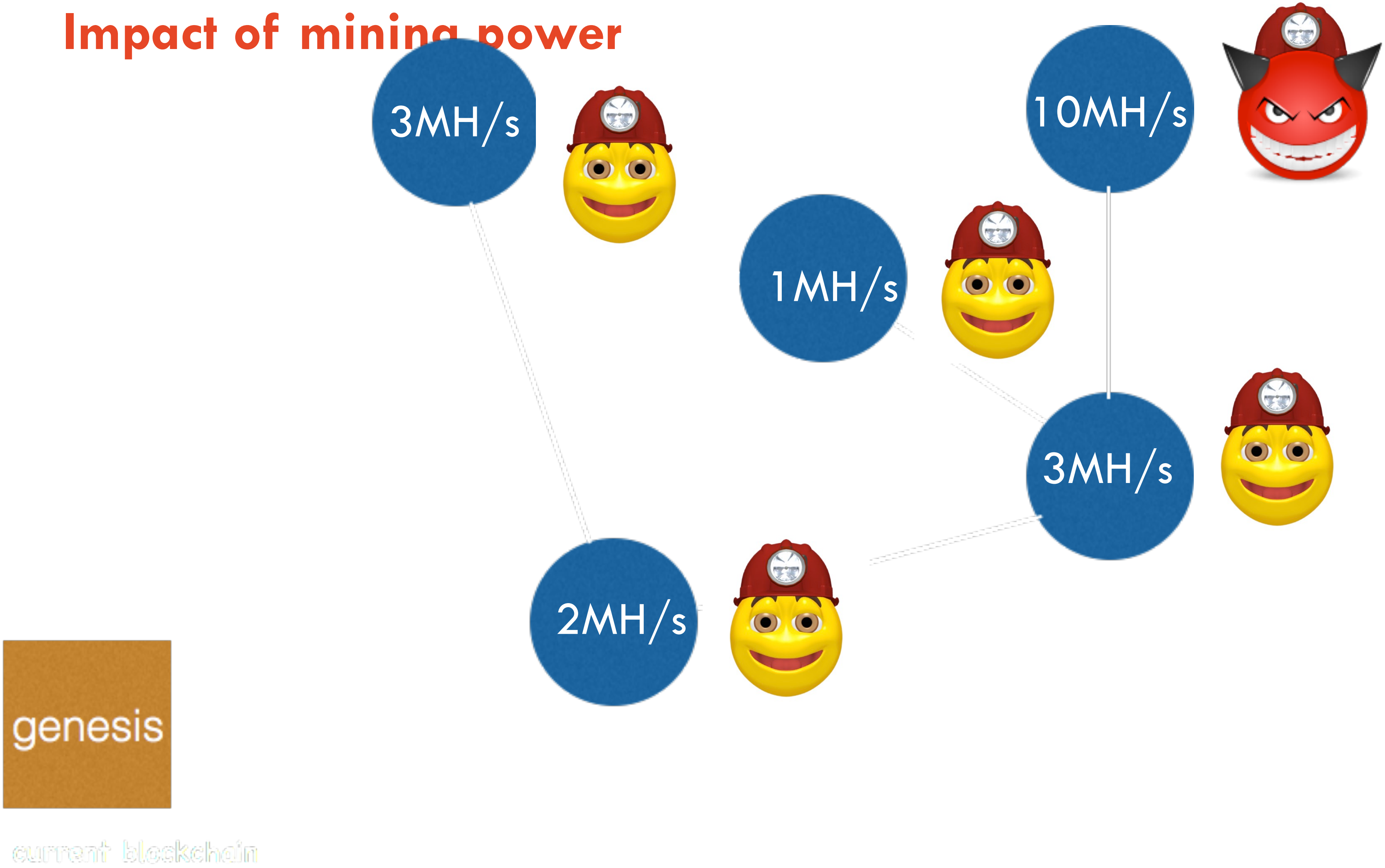
Committed transaction

- Given a blockchain with parameter k, a block at index i is *decided* when the chain depth reaches i+k
- A transaction is *committed* if it belongs to a decided block

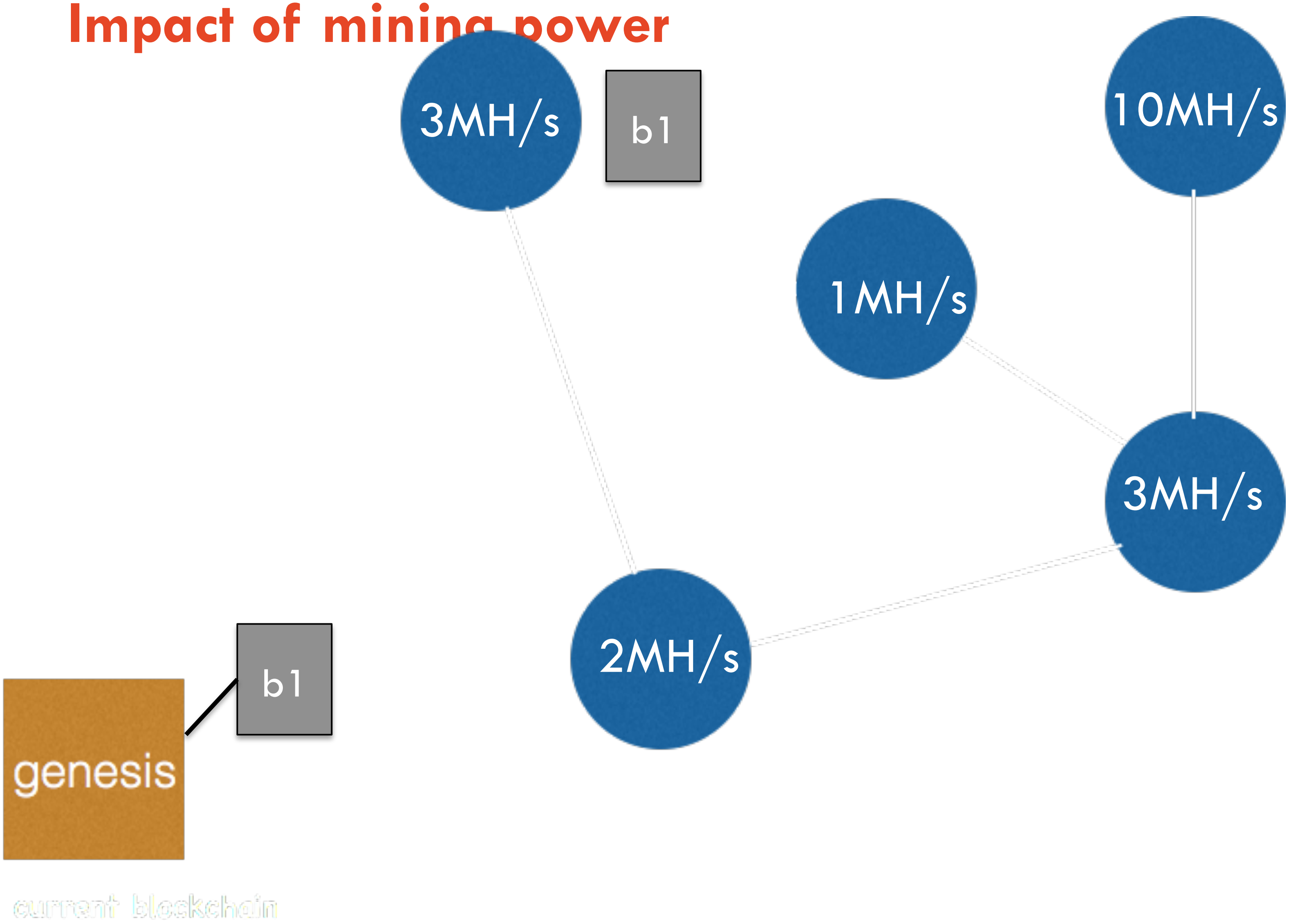


51% attack

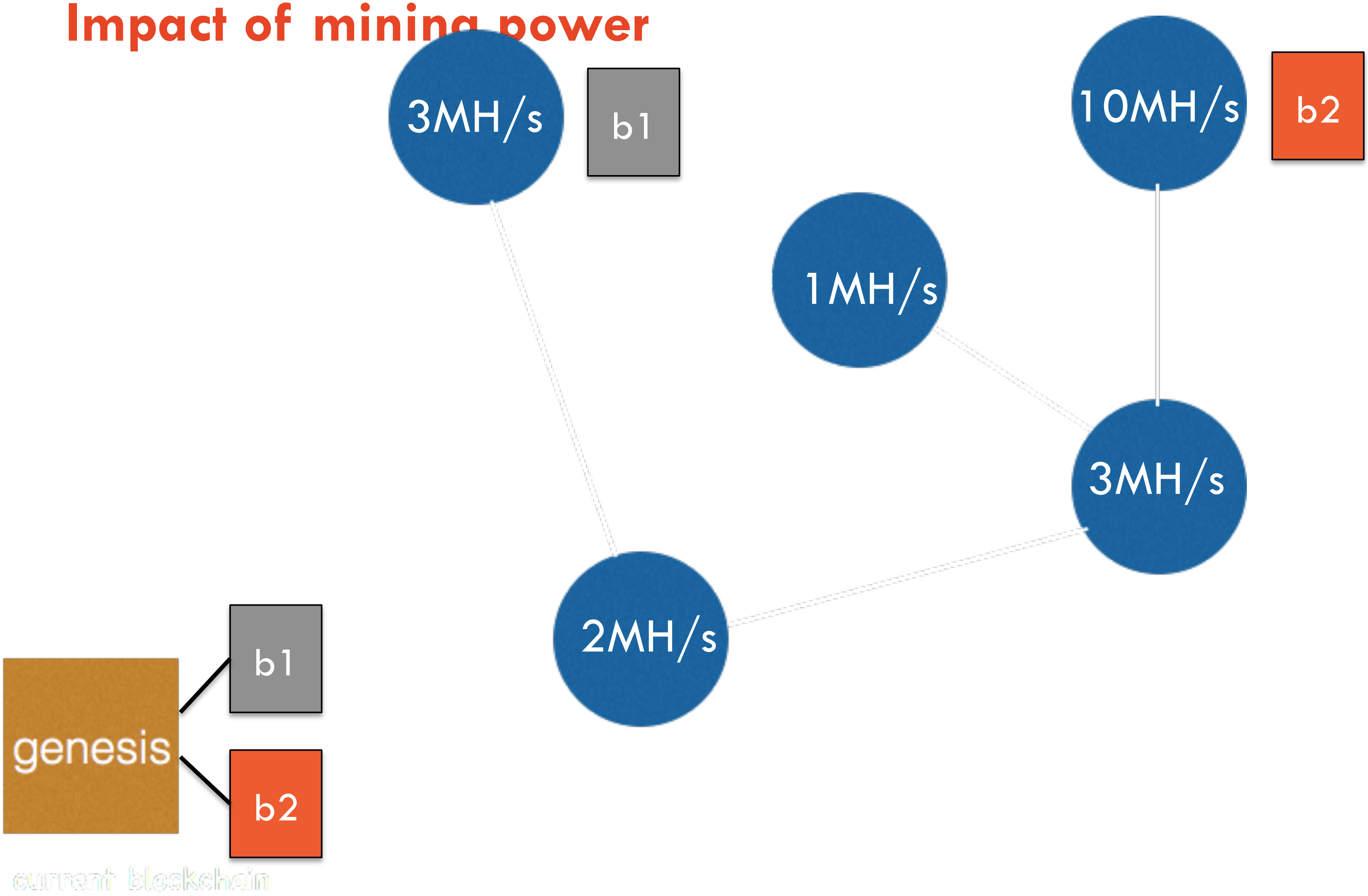
Impact of mining power



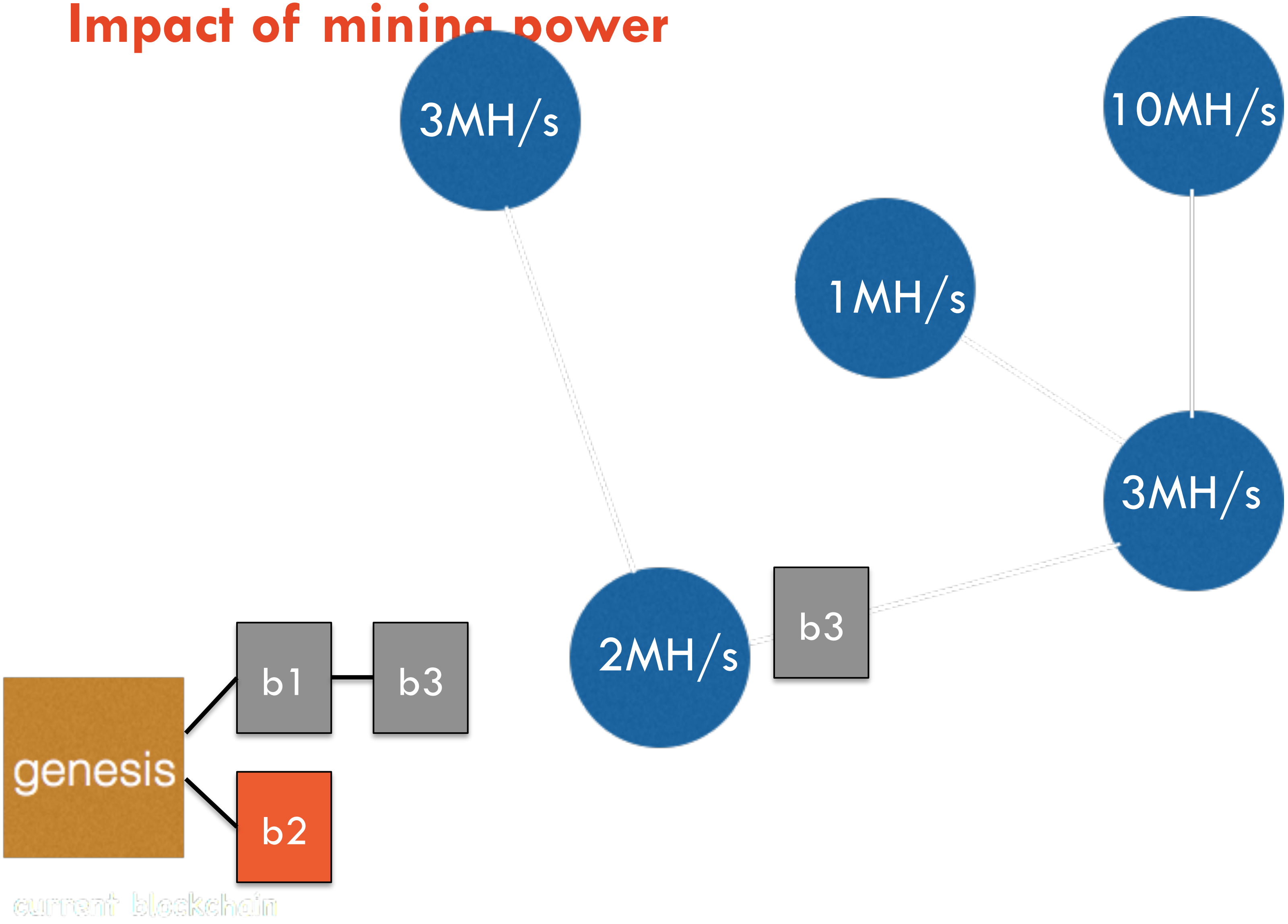
Impact of mining power



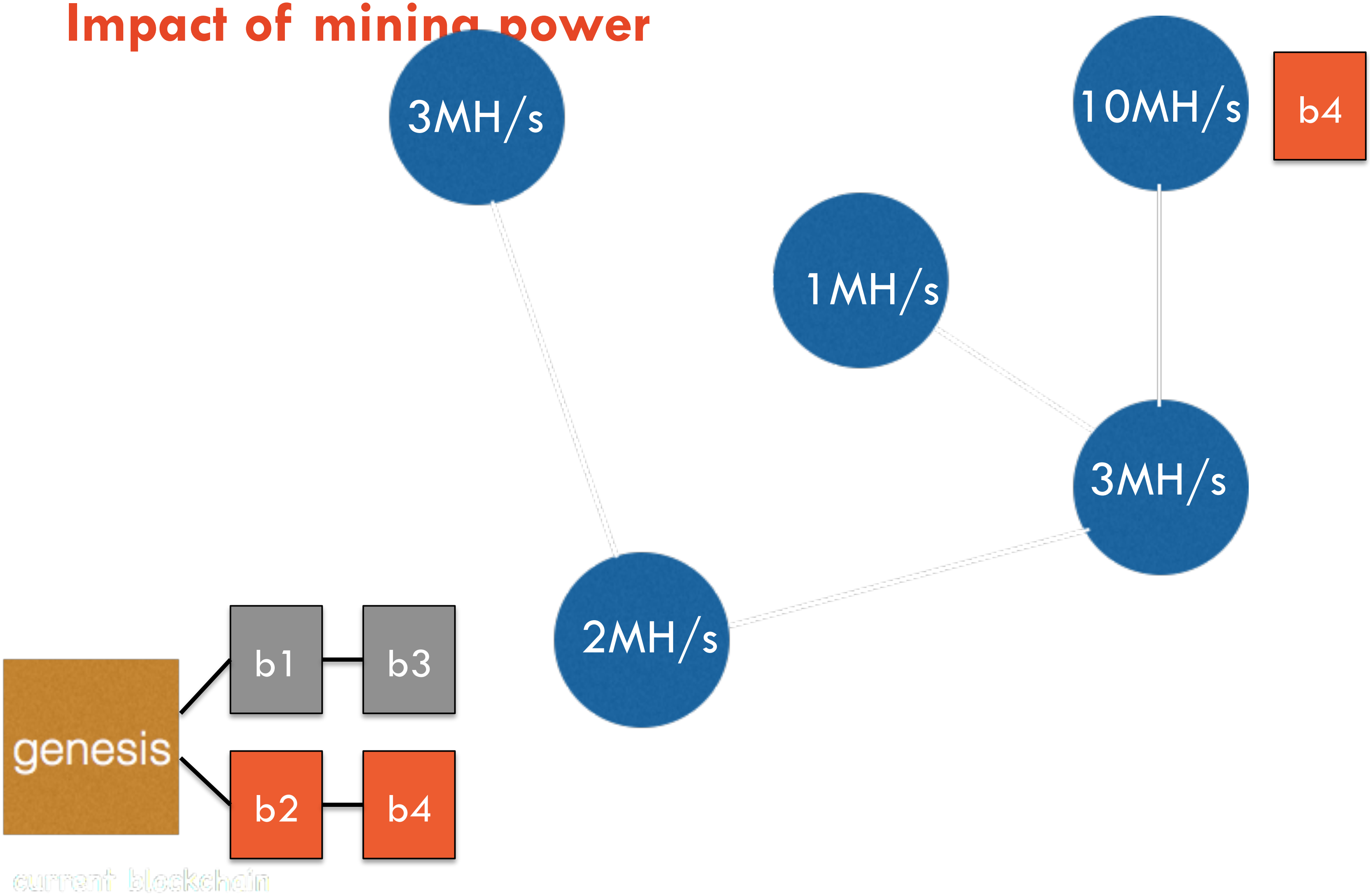
Impact of mining power



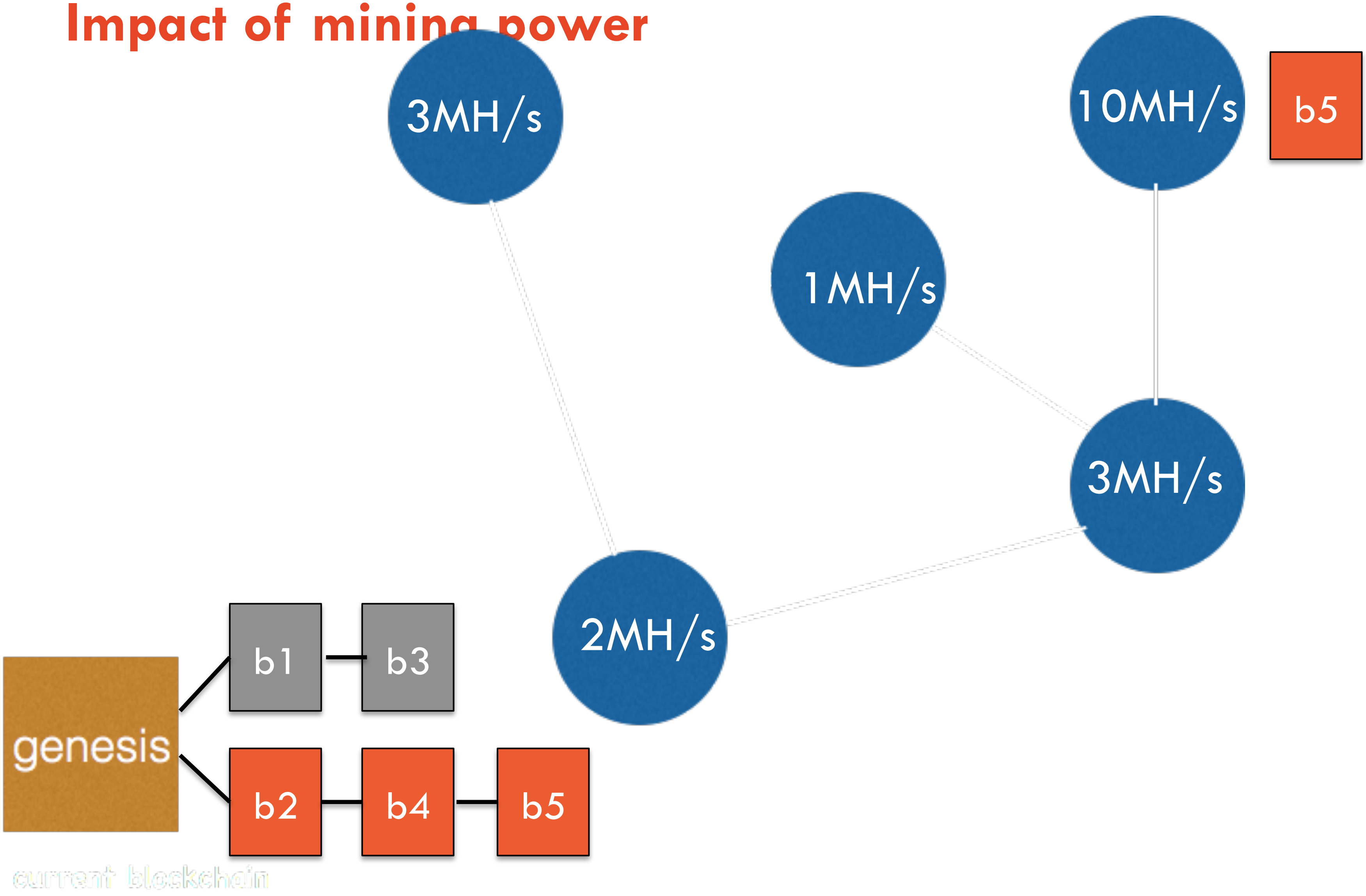
Impact of mining power



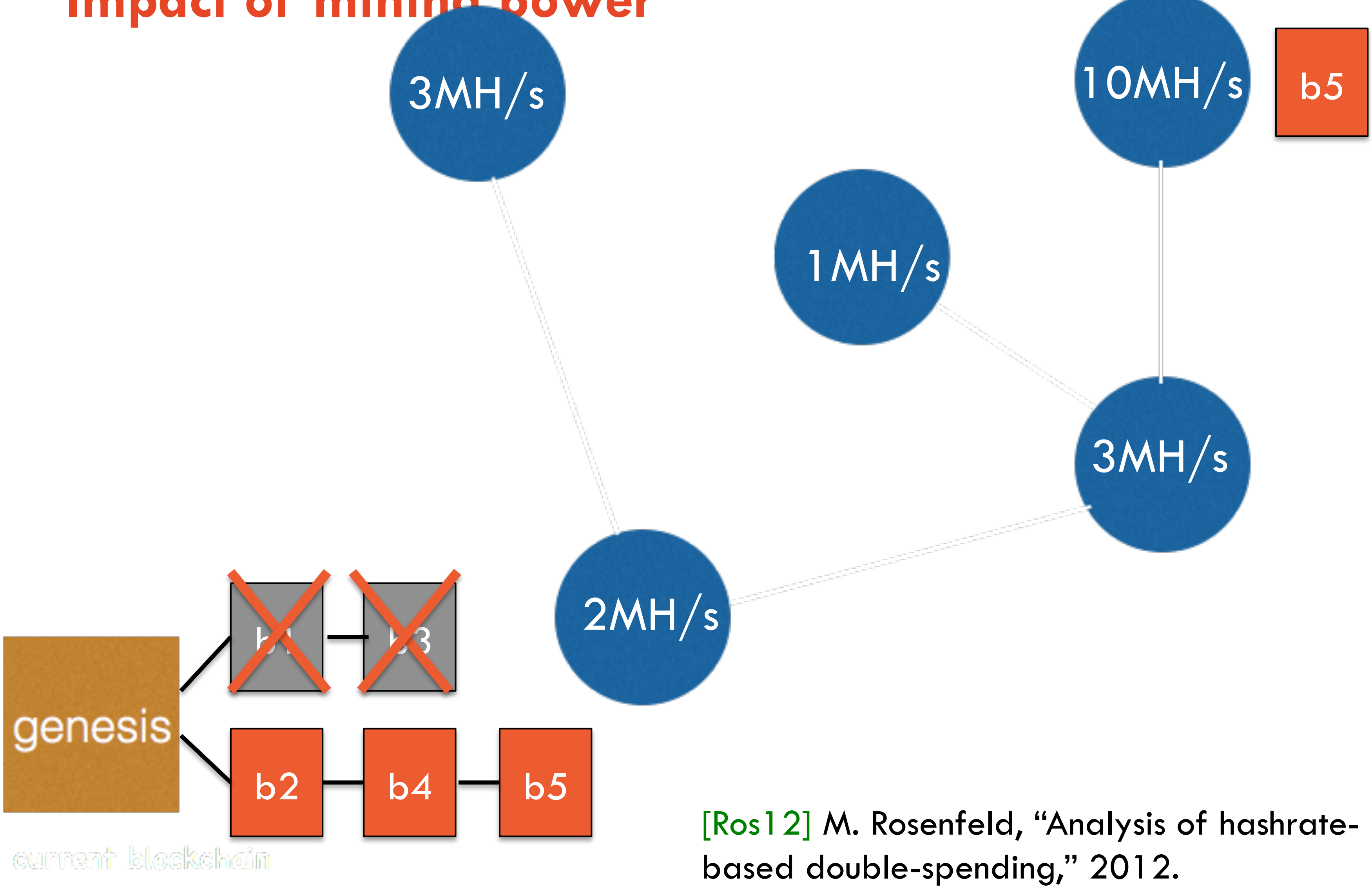
Impact of mining power



Impact of mining power



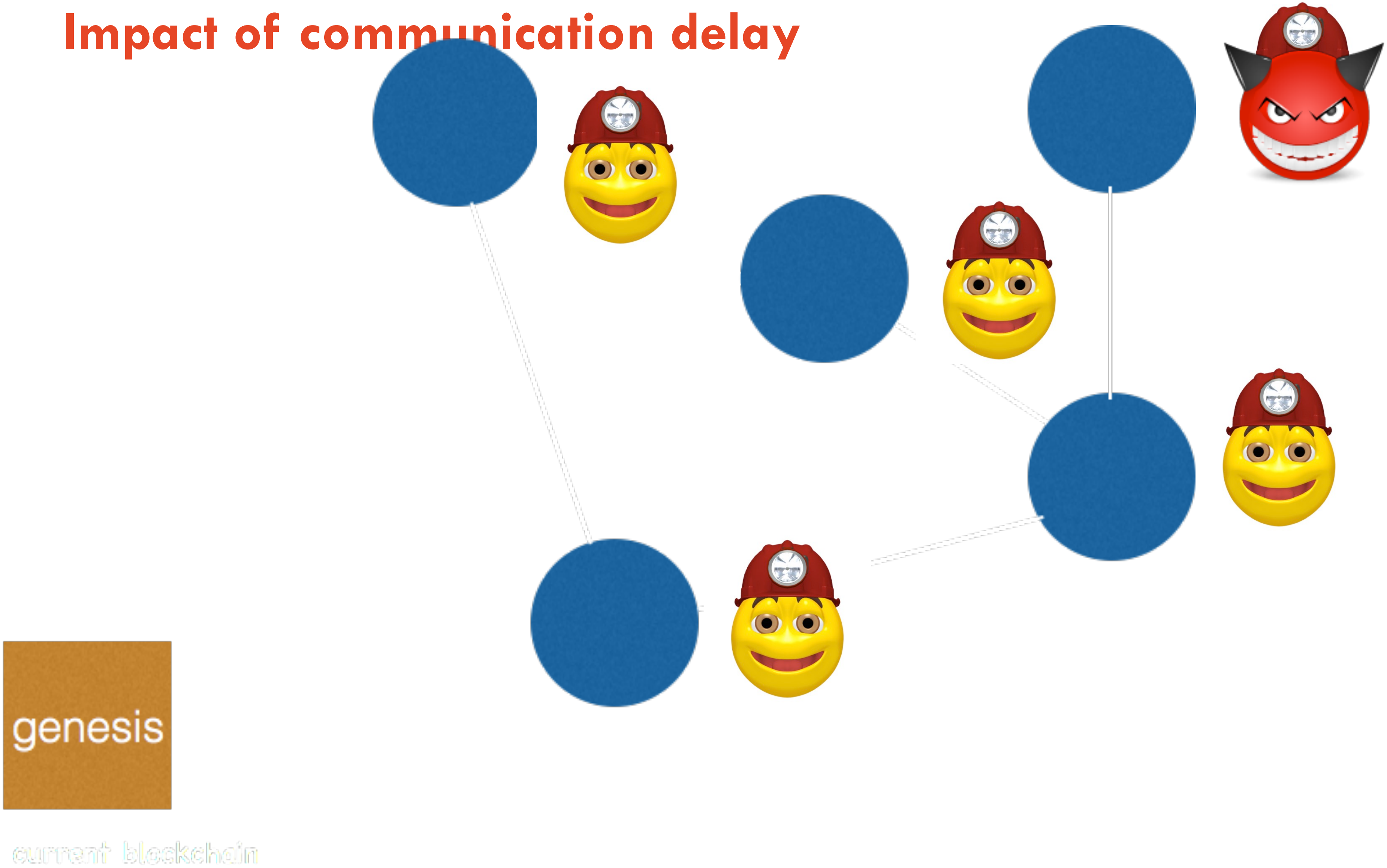
Impact of mining power



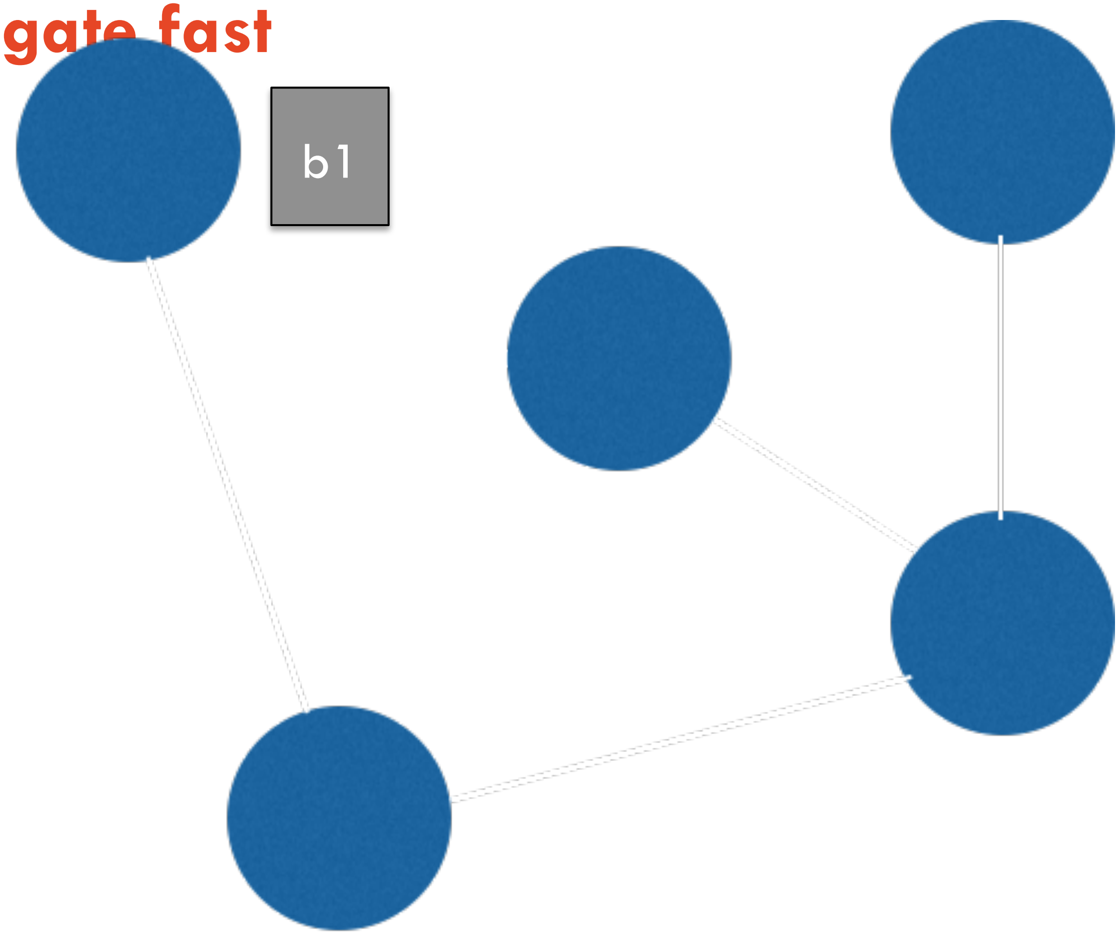
[Ros12] M. Rosenfeld, “Analysis of hashrate-based double-spending,” 2012.

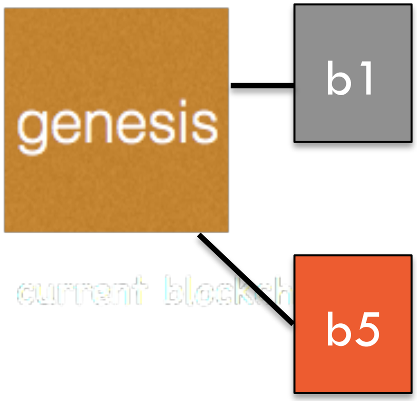
Impact of Communication Delay

Impact of communication delay

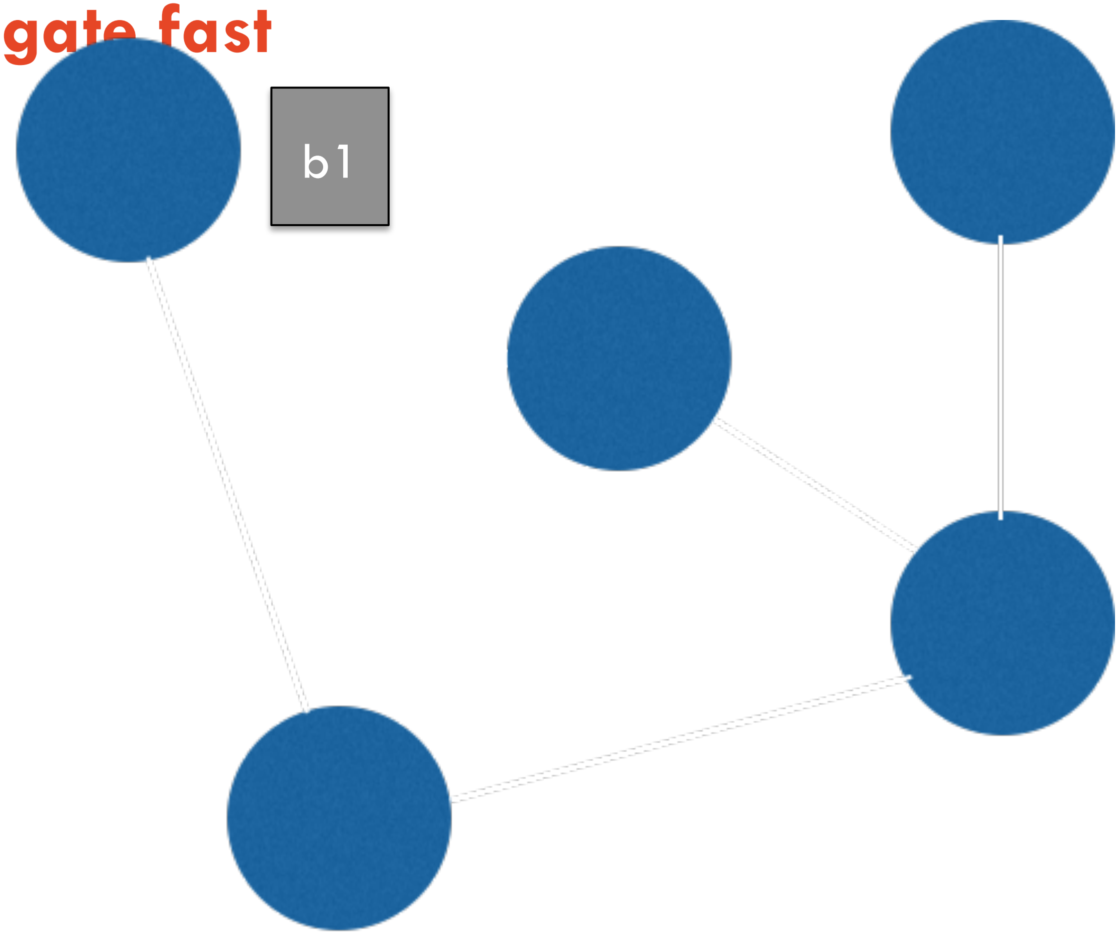


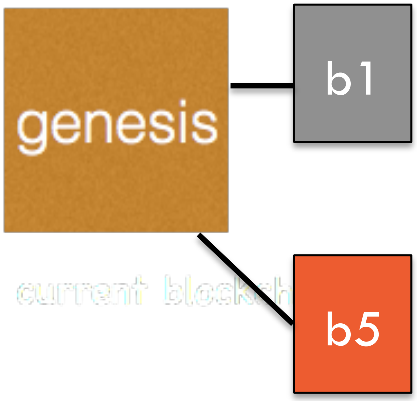
Blocks propagate fast



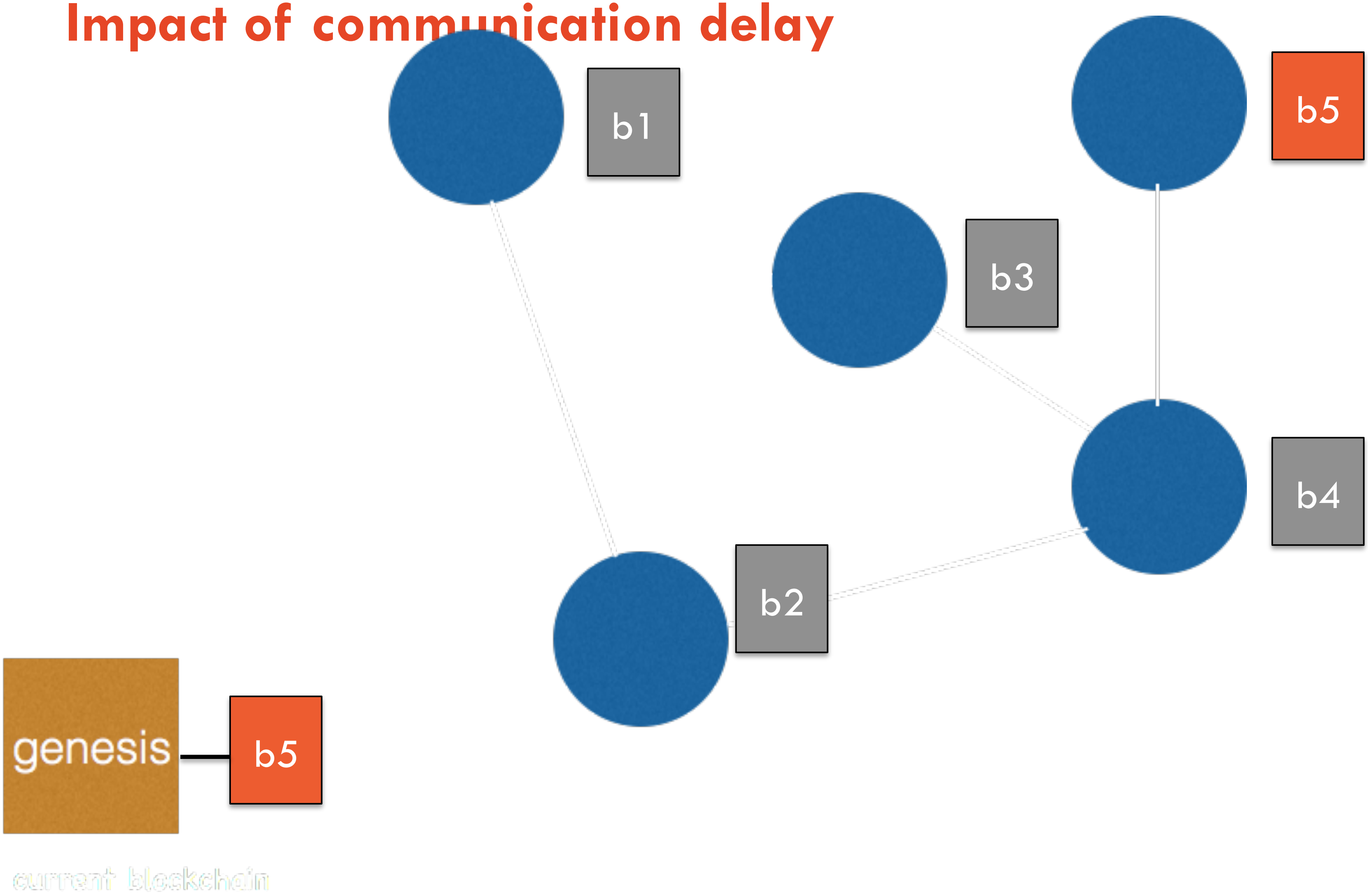


Blocks propagate fast

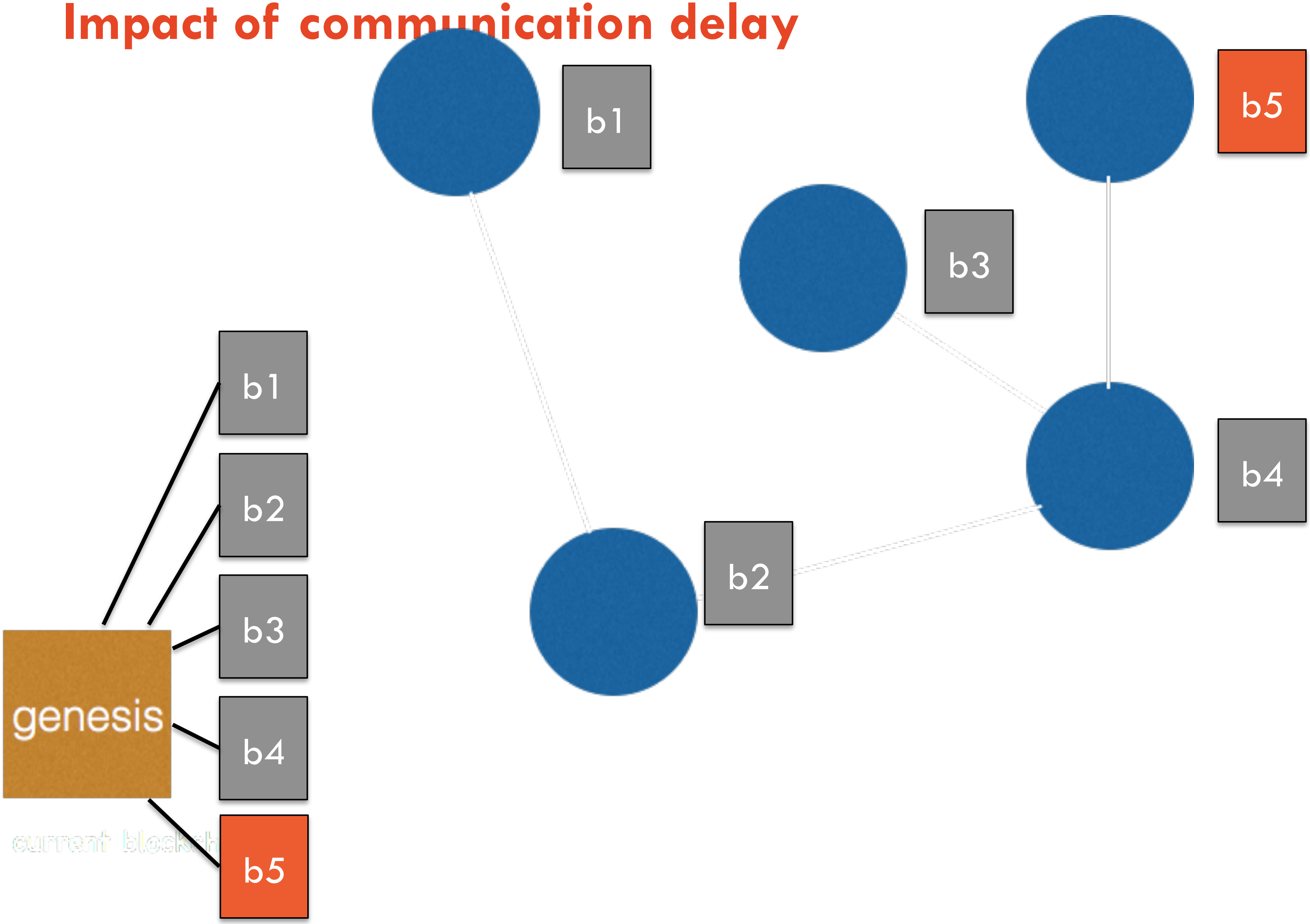




Impact of communication delay



Impact of communication delay



Ethereum and GHOST

THE UNIVERSITY OF SYDNEY

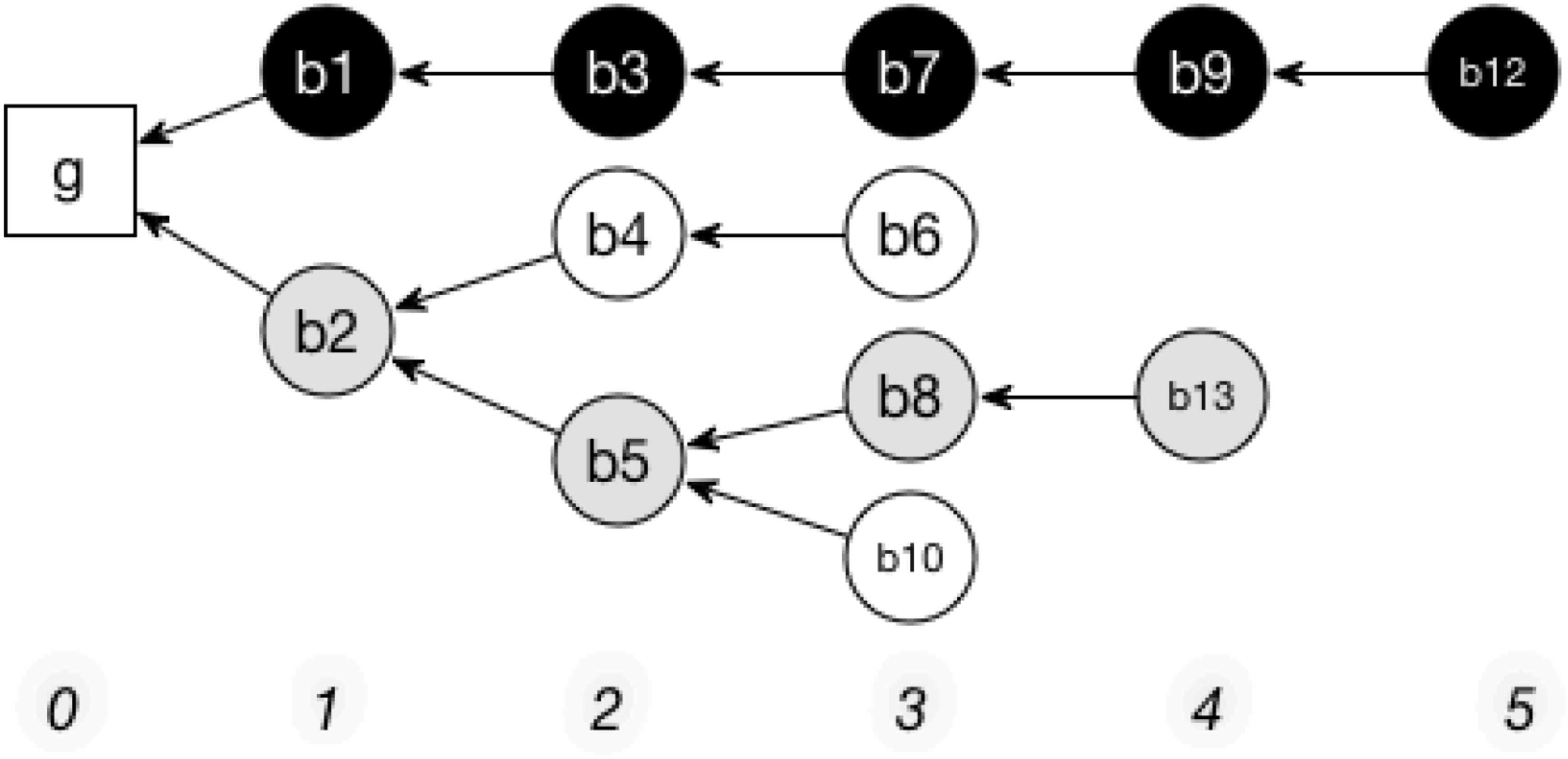
Ethereum consensus chooses greedily the heaviest subtree (GHOST)

```
State
$\langle Bi,Pi \rangle$ the local blockchain view

Each peer of the blockchain executes:

Receive blocks $\langle Bj, Pj \rangle$ from j
   $Bi = Bi \cup Bj$
   $Pi = Pi \cup Pj$

num-desc(b):
  if $children(b) = \emptyset$ then return 1
  else return $1 + \sum_{c \in children(b)}$ num-desc(c)
```

```
Prune lightest branches at i
  b = genesis-block(Bi)
  while $b.next \neq \perp$
    block = $\mathrm{argmax}_{c \in children(b)}$ {num-desc (c)}
    $B = B \cup \{block\}$
    $P = P \cup \{\langle block,b \rangle\}$
    b = block
  $\langle Bi,Pi \rangle = \langle B,P \rangle$
```

[SZ15] Y. Sompolinsky and A. Zohar, “Secure high-rate transaction processing in bitcoin,” in Financial Cryptography and Data Security FC’15, 2015, pp. 507–527.

Bitcoin vs. Ethereum


b1
b3
b7
b9
b12
g
b4
b6
b2
b8
b13
b5
b10
0
1
2
3
4
5