# Introduction to Ethereum

Topics:

- Background
- Motivations
- Monetary Supply in Ethereum
- Consensus Protocol - variant of GHOST Proof of Work
- Difficulty Adjustment

# Background

- Proposed in 2013 by Vitalik Buterin (b. 1994)
- Company formed 2014, followed by establishment of the non-profit Ethereum Foundation
- Crowd sale (Initial Coin Offering) Aug 2014
- First open source code deployment July 2015
- Mix of MIT, LGPL, GPL, Alfero Licenses
- https://www.ethereum.org
- Quickly became one of top cryptocurrencies.



Image: https://twitter.com/VitalikButerin

# Motivations

From Ethereum White Paper (Buterin, 2013):

- Better support for decentralised applications built on top of a crypto-currency platform
- More expressive scripting language
- (See Smart Contract Intro lecture for Bitcoin limitations)

# Ethereum Currency

| Unit | Alternate Name | Wei Value |
|---|---|---|
| wei | | 1 |
| babbage | Kwei | $10^3$ |
| lovelace | Mwei | $10^6$ |
| shannon | Gwei | $10^9$ |
| szabo | microether | $10^{12}$ |
| finney | milliether | $10^{15}$ |
| ether | | $10^{18}$ |

# Ethereum's Crowdsale

The Ethereum project raised funds to support development using a form of crowd-funding
"presale" called an *Initial Coin Offering*
(ICO, by analogy to IPO = Initial Public Offering of shares)

In exchange for Bitcoin, investors received a promise of the Ether currency once the platform launched. (Also called *premining of coins.)*

Price: 2,000 Ether per BTC for early buyers, 1,337 Ether per BTC for latest investors



60M Ether sold in presale, raising $US 18.4M worth of BTC



Another 12M Ether premined:

- 3M endowment for Ethereum Foundation
- 6M as payment to developers for unpaid work prior to crowdsale
- 3M to back developer options to purchase at presale prices.

# Monetary Supply in Ethereum

Initial (premine) supply of Ether: 72M

Block reward: originally 5 ETH per block ( ~ every 15 seconds ), changed by hard fork only

Total issuance per year capped at 18M Ether (25% of initial supply)

- NO pre-set halving, NO upper bound on the amount of Ether
- Constant number of new Ether per year (to cover e.g. lost coins)
- Rate of *inflation* converges to 0 over time



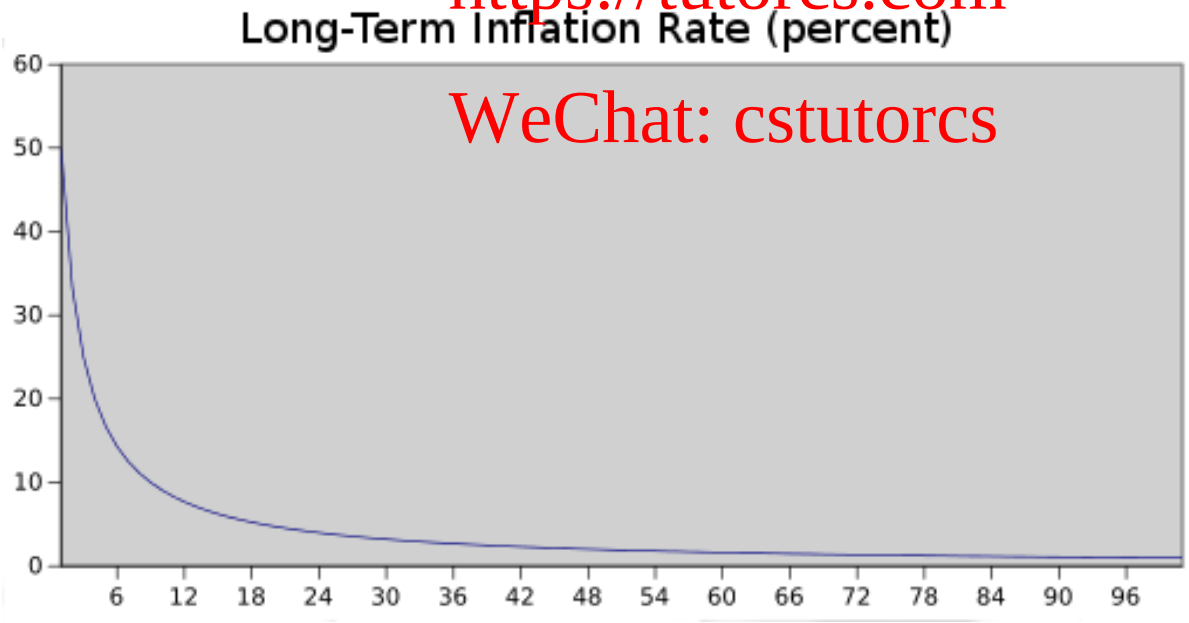
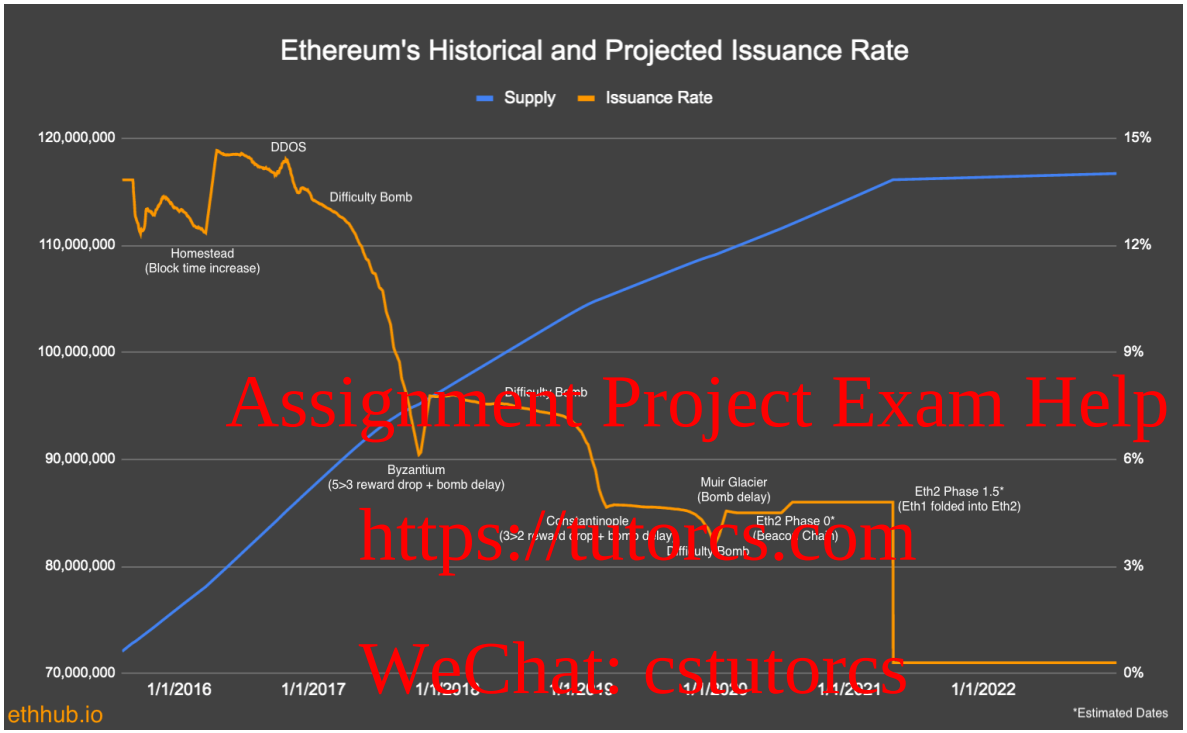



Figure from Ethereum White paper

# Monetary Supply Law Changes






See https://docs.ethhub.io/ethereum-basics/monetary-policy/

Original block reward was 5 ETH per block

2017: Change to 3 ETH per block by hard fork due to large increase in ETH value

2019: Change to 2 ETH per block

“Difficulty Bomb” changes to block production rate in 2017, 2019, 2020

Further changes expected 2021 due to switch to ``Proof of Stake” mining

# GHOST Protocol

"Secure High-rate Transaction Processing in Bitcoin", Y. Sompolinsky, A. Zohar, Financial Cryptography 2015

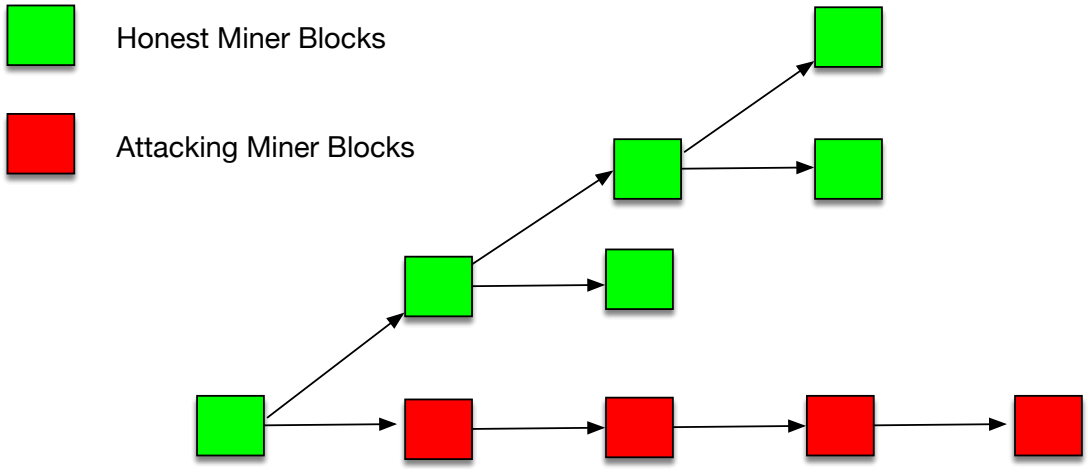
**Observation**: There is a trade-off between security and throughput in Bitcoin:

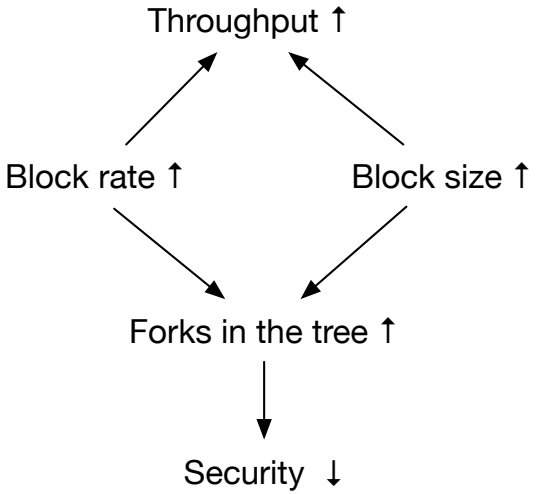
Consider an attacker attempting to construct a chain longer than the official chain. The attacker acts in coordinated way with its power, the honest nodes compete.



Define $$\text{Security} = \frac{\text{growth-rate of main branch}}{\text{growth-rate of attacker branch}}$$





# GHOST Protocol

The main idea of the GHOST (Greedy Heaviest-Observed Subtree) Protocol:

define the weight of the main chain so as to include blocks that are in other branches

**GHOST Algorithm** for selecting the leaf that defines the main chain:

*Input:* A tree of blocks, root = genesis block

B := genesis block

while children(B) non-empty

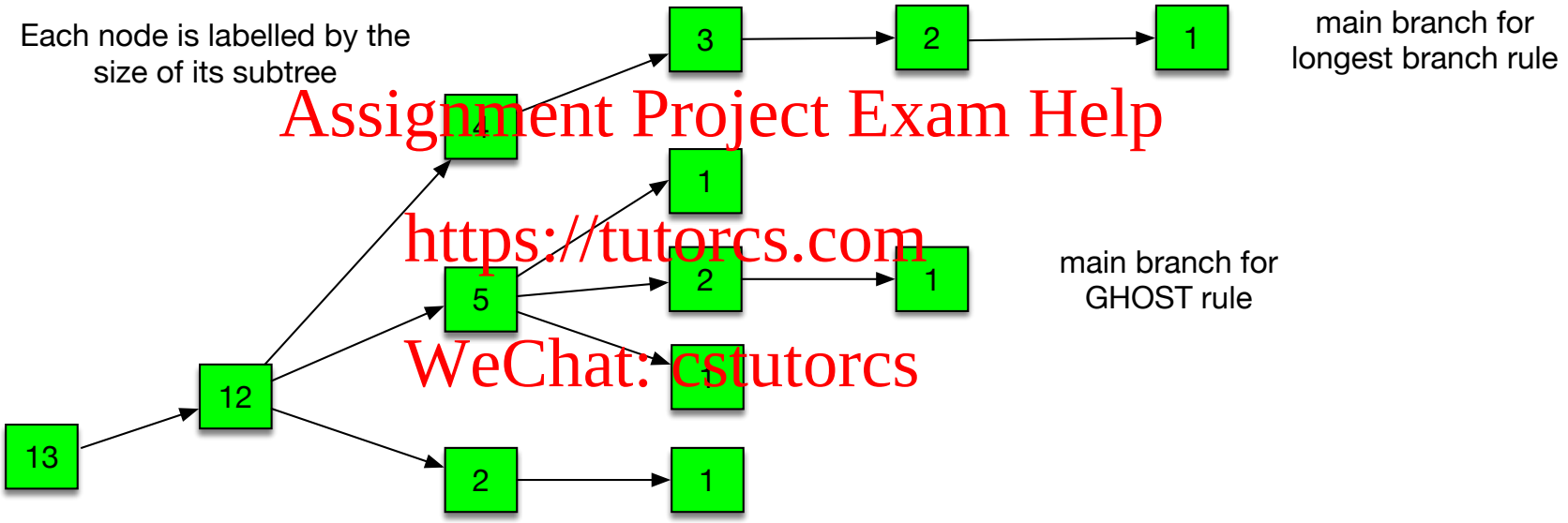
do { B := a child of B with the largest weight* subtree }

return(B)

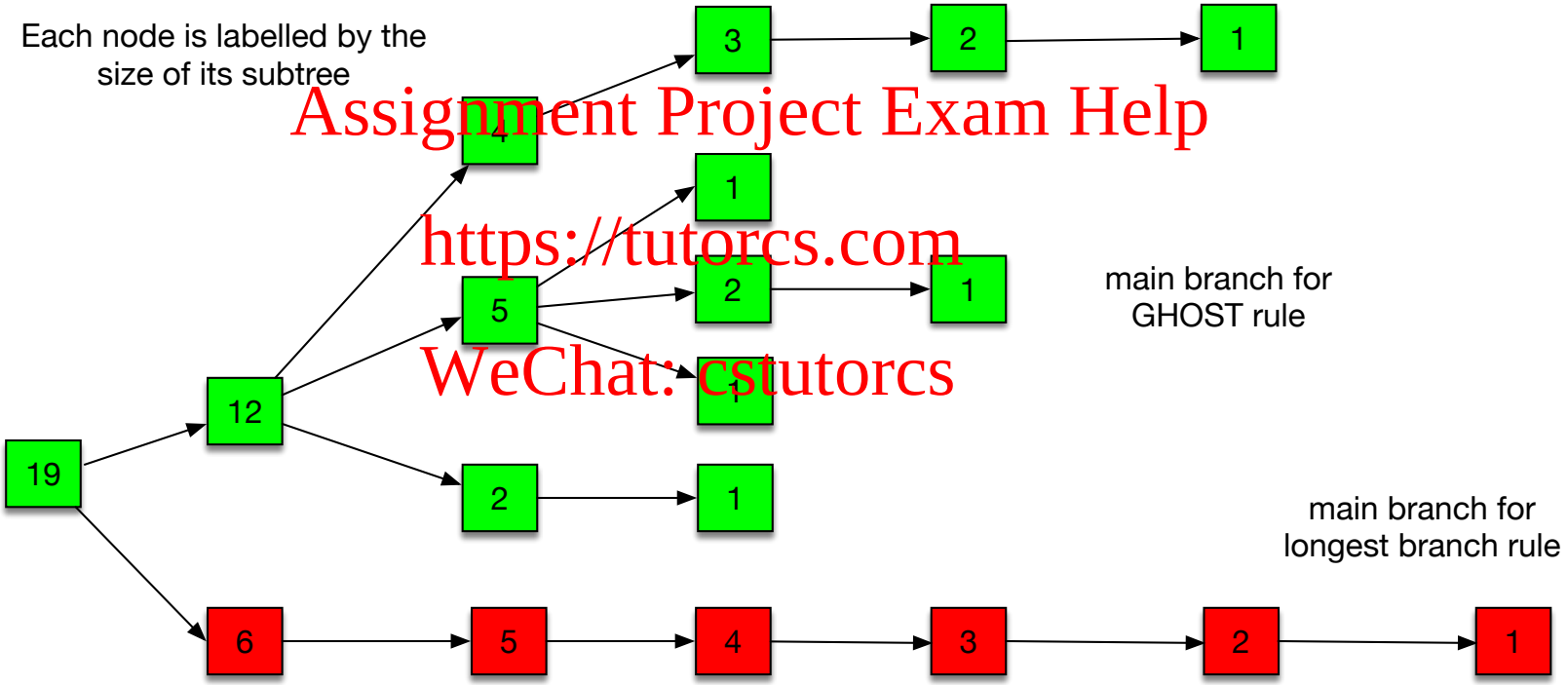


* = we illustrate with subtree size as the measure, in reality should use total difficulty

# Example



# Example - after the attacker reveals a chain



# A further issue: centralisation of mining power

Suppose the block rate is high and many forks are produced.

Say a block is *stale* if it ends up not being included in the main chain.



Consider



- miner A with 10% hash power, risks producing a stale block 90% of the time
- miner B with 30% hash power, risks producing a stale block 70% of the time



Thus - smaller miners waste more of their effort, and have an incentive to join a larger pool.

# Ethereum's variations to GHOST

Ethereum takes ideas from GHOST, but

- adds measures to counteract a tendency to miner centralisation
- simplifies GHOST for "ease of implementation"



Main idea:

- blocks B may point to some (valid) stale blocks S
- reward
  - the miner of a (valid) stale block S for their effort
  - the miner of the block B for pointing to S

Data61 visualisation of Ethereum Blockchain construction:

http://www.ethviewer.live

# More specifically …

An *uncle*, or *ommer* (gender neutral version) of a block B is a descendant of an ancestor of B that is not itself an ancestor.

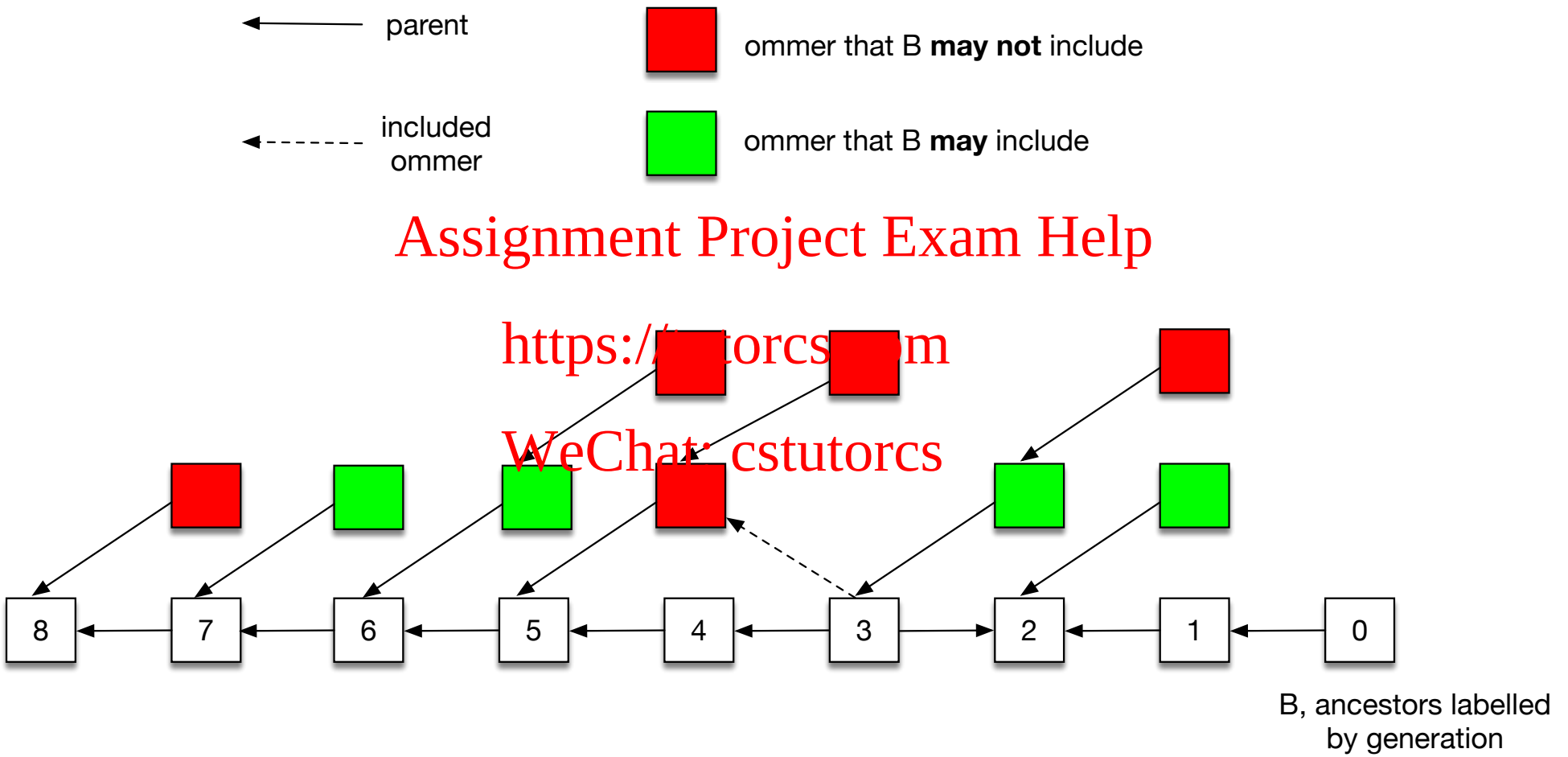
A block B specifies a parent, and 0 .. 2 ommers

- An ommer O included in B must satisfy:
  - O is a direct child of an ancestor of B from the 2..7th previous generation (1 = parent)
  - O may not be an ancestor of B
  - O must be a valid block header, but does not need to be a valid block (*)
  - O may not be included twice in the same chain:
    - O must be different from all ommers included in previous blocks
    - O must be included at most once in B



**Note:** transactions from included ommers are NOT included in the state history, they go back into the mempool. So invalidity of a block in (*) does not cause damage.

# Example



# Reward Structure

Rewards are associated with ommer inclusion as follows:

- The Block including an ommer gets an *additional* block reward of 1/32 of the standard block rewards (excluding transaction fees)
- Each ommer O included gets a reward that depends on how soon it was included:
  - most recent ommers (child of parent of parent) get 7/8 of the block reward (no fees)
  - ommers from the preceding generation get 6/8 of the block reward (no fees)
  - ….
  - ommers from 7 generations back get 2/8 of the block reward (no fees)

# Difficulty Adjustment

Ethereum originally adjusted difficulty based just on the generation rate of the most recent blocks on the main chain.

This was shown to permit an *uncle mining* strategy, whereby miners could benefit more by mining ommers rather than blocks extending the main chain!



S. D. Lerner, "Uncle mining, an ethereum consensus protocol flaw," *Wordpress Blog*, 2016. https://bitslog.wordpress.com/2016/04/28/uncle-mining-an-ethereum-consensus-protocol-flaw/





The Byzantium hard fork (Oct 2017) fixed this by counting included ommers in the generation rate

# Summary

Ethereum:

- Background
- Motivations
- Monetary Supply
- GHOST protocol
- Ethereum Consensus Protocol - variant of GHOST
- Difficulty Adjustment