the consensus mechanism of Conflux

a bit about me...

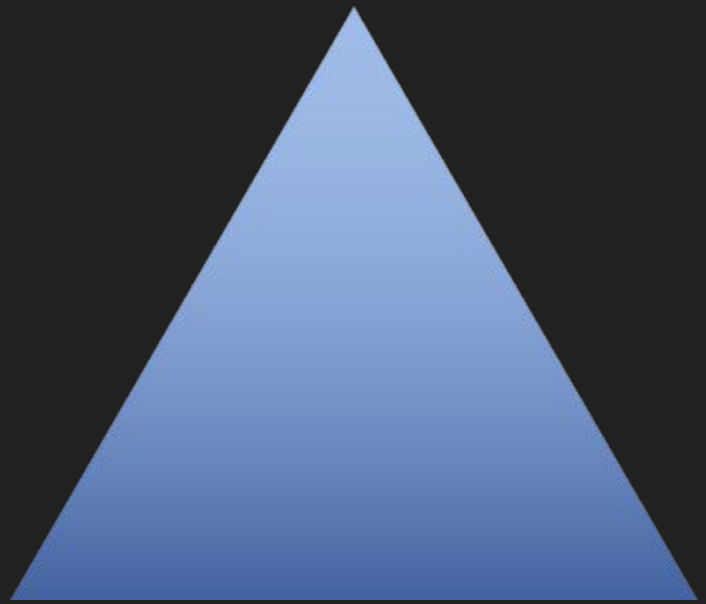
Péter Garamvölgyi

I did my Masters in CS at Tsinghua (2018-20)

I served as the previous president of TIBA

I joined Conflux in May 2019

scalability



decentralization

security

P2P Network

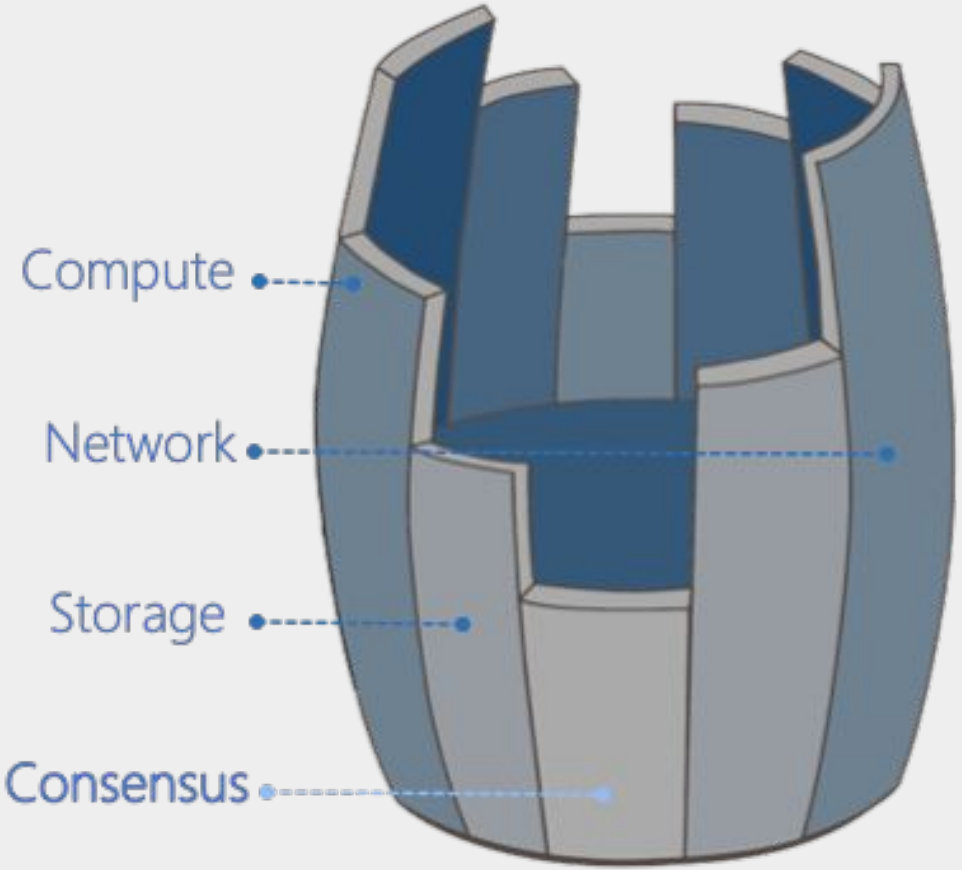
~10Mbps
=> 3000-6000 TPS

Storage I/O

20K-40K SSD I/O Operations
=> 2000-10000 TPS

Compute

4000~8000
Transaction Signature
Verification
per Second


Compute
Network
Storage
Consensus

consensus in Bitcoin

everyone agrees which transactions happened in which order

components: **block tree**

longest chain rule

proof-of-work

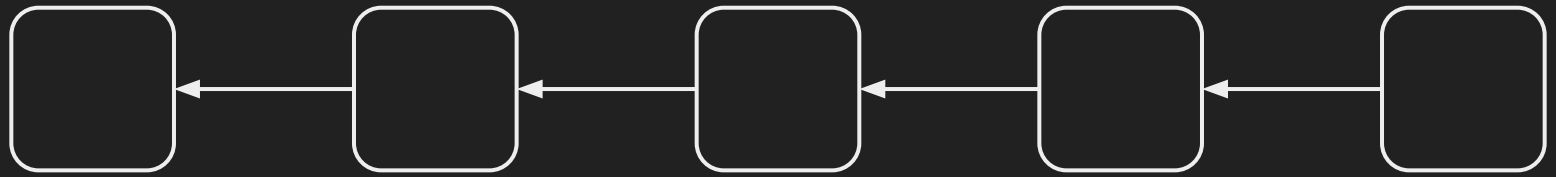
block tree in Bitcoin

transactions are packed into blocks

block tree in Bitcoin

transactions are packed into blocks

each block references exactly one previous block

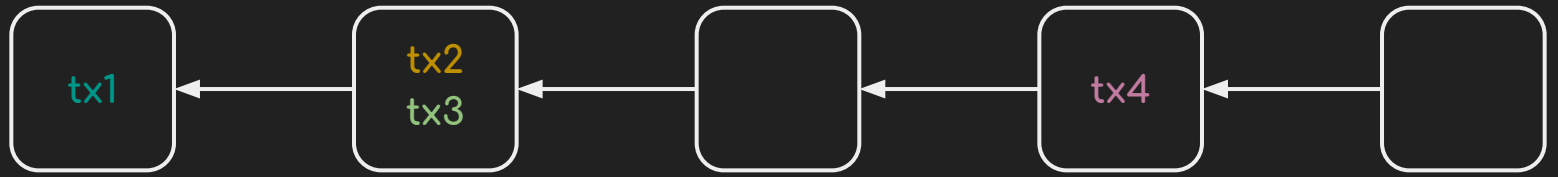


block tree in Bitcoin

transactions are packed into blocks

each block references exactly one previous block

the chain of blocks defines the canonical transaction order



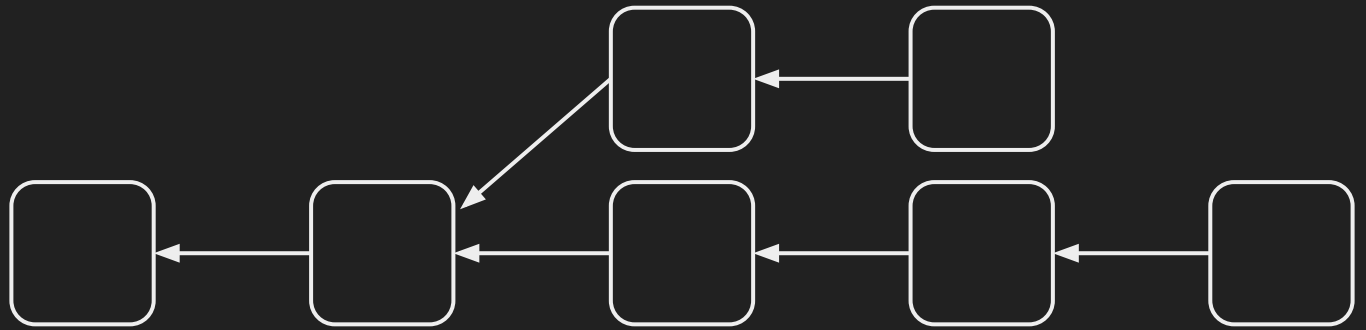
block tree in Bitcoin

transactions are packed into blocks

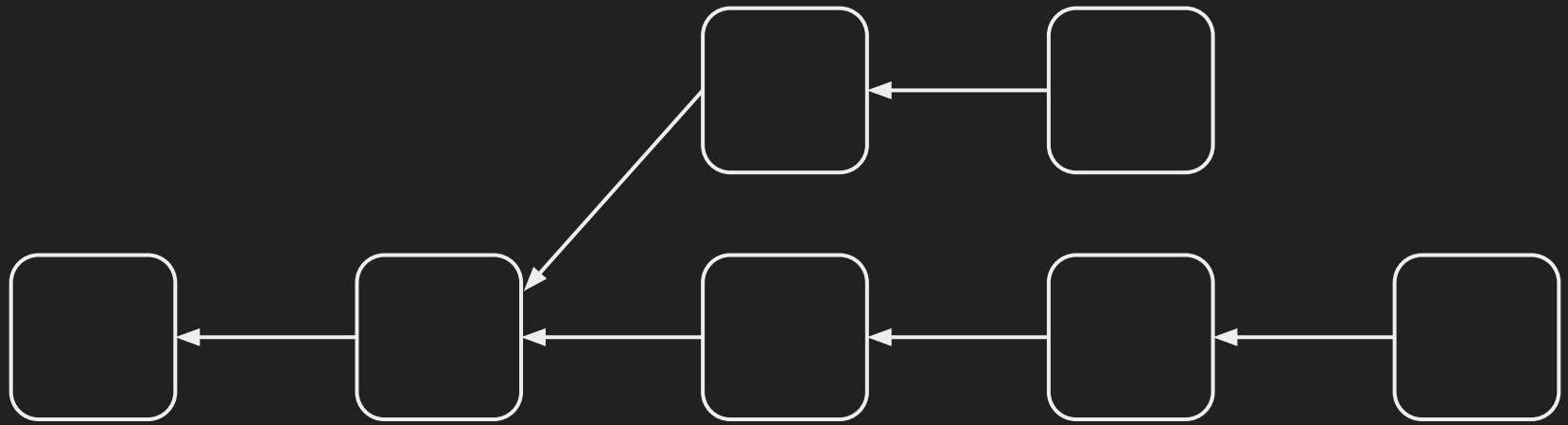
each block references exactly one previous block

the chain of blocks defines the canonical transaction order

two blocks can reference the same parent block



block tree in Bitcoin



consensus in Bitcoin

everyone agrees which transactions happened in which order

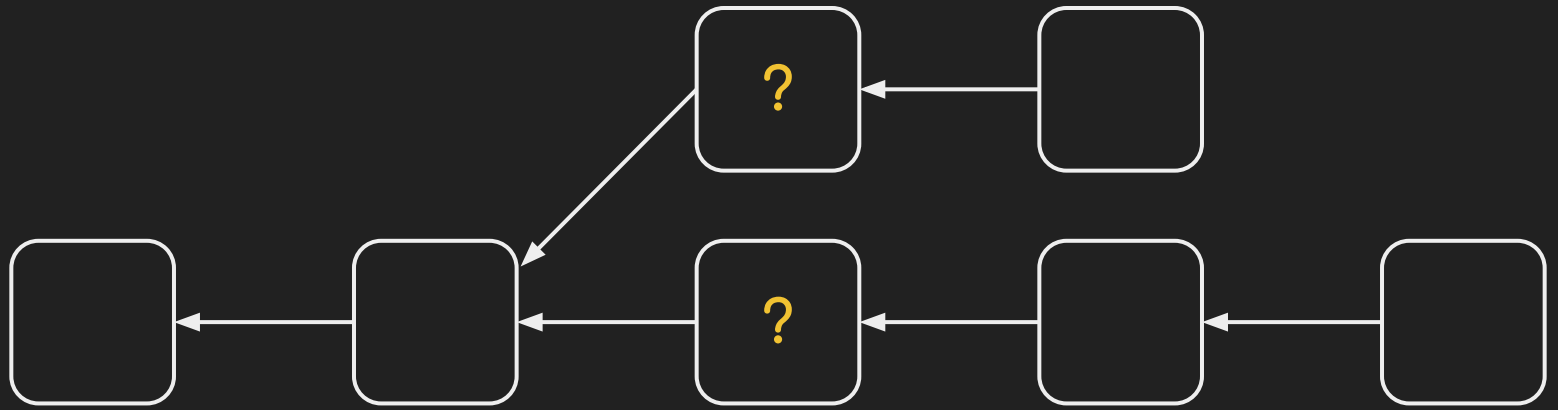
components: block tree

longest chain rule

proof-of-work

longest chain rule in Bitcoin

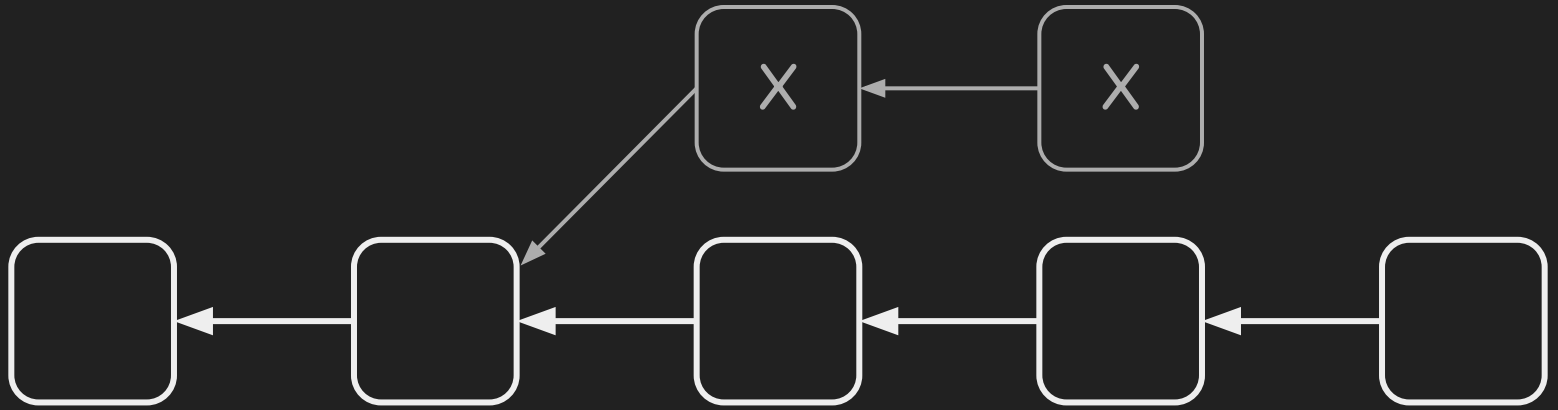
how do we handle concurrent blocks?



longest chain rule in Bitcoin

how do we handle concurrent blocks?

always select the longest chain, discard the rest

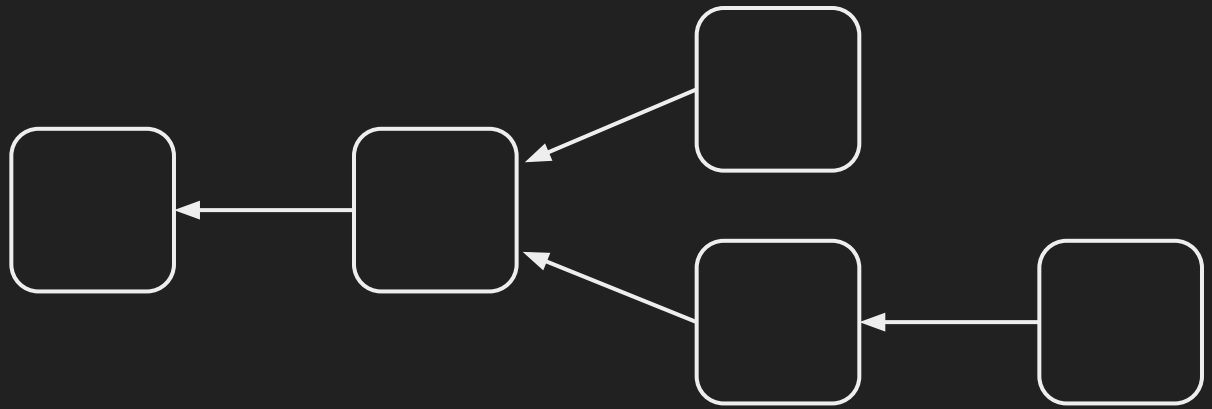


longest chain rule in Bitcoin

how do we handle concurrent blocks?

always select the longest chain, discard the rest

newly mined blocks will reference the longest chain

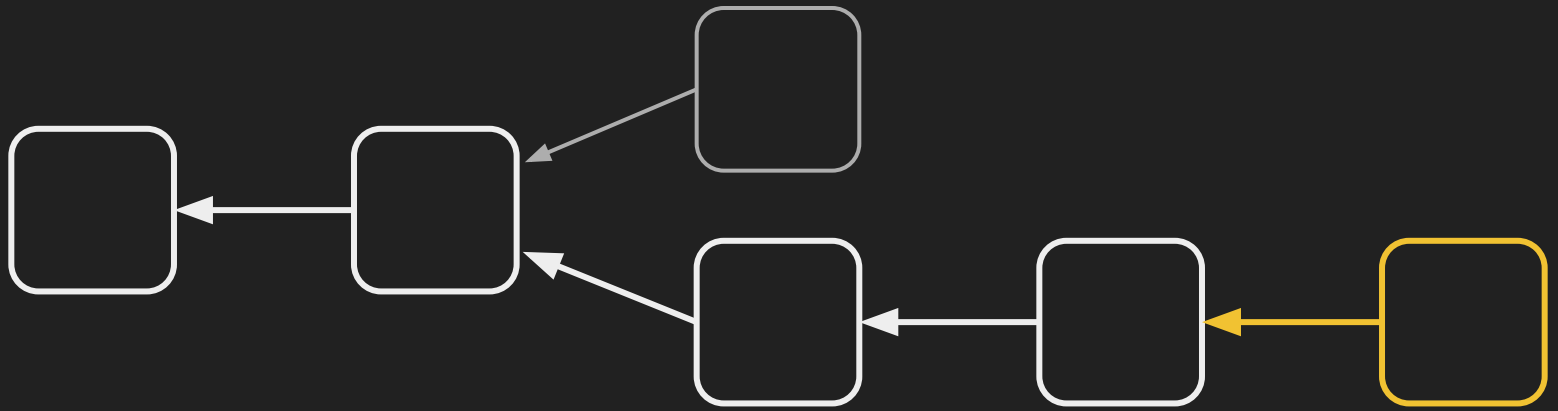


longest chain rule in Bitcoin

how do we handle concurrent blocks?

always select the longest chain, discard the rest

newly mined blocks will reference the longest chain



consensus in Bitcoin

everyone agrees which transactions happened in which order

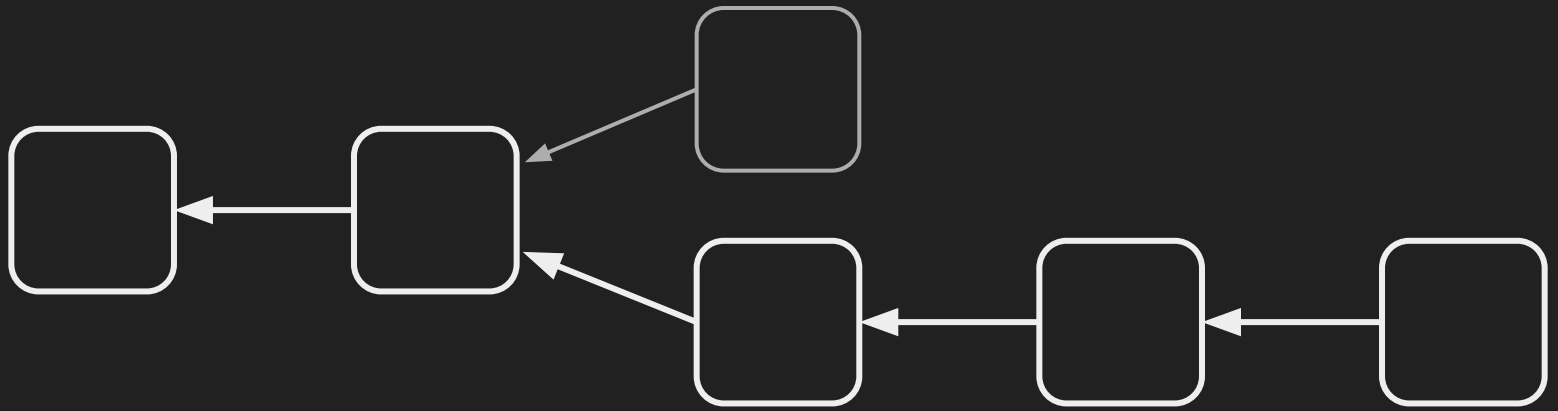
components: block tree

longest chain rule

proof-of-work

proof-of-work

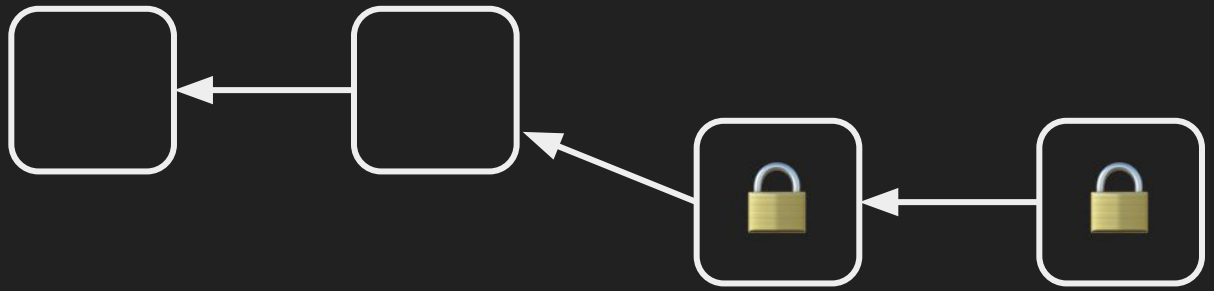
it is trivial to make a competing long chain of blocks



proof-of-work

it is trivial to make a competing long chain of blocks

we artificially make it hard to create blocks

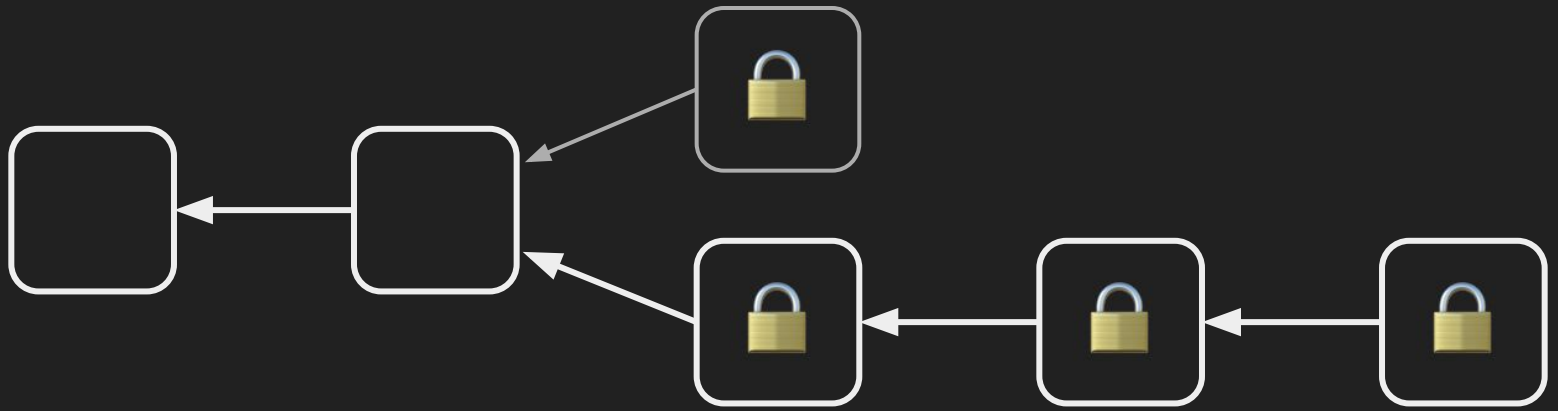


proof-of-work

it is trivial to make a competing long chain of blocks

we artificially make it hard to create blocks

an attacker with <50% hashpower cannot keep up



consensus in Bitcoin

everyone agrees which transactions happened in which order

components: block tree

longest chain rule

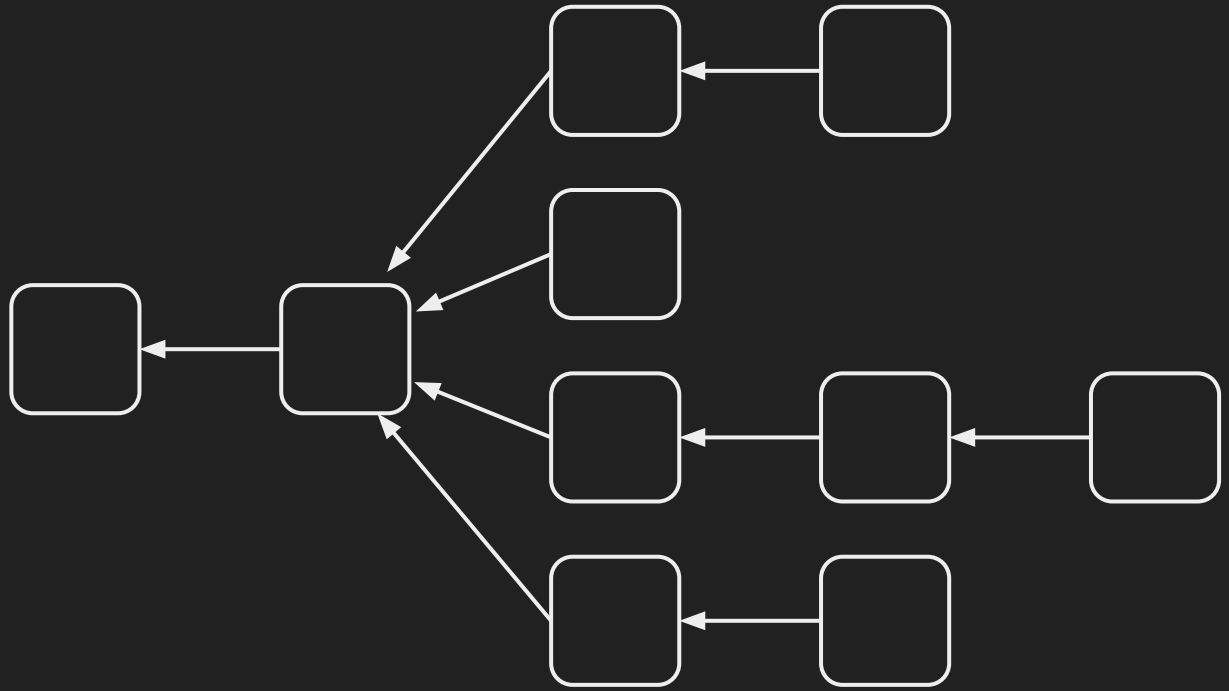
proof-of-work

problem: discarded blocks do not contribute to the security or throughput of the system.

can we do better?

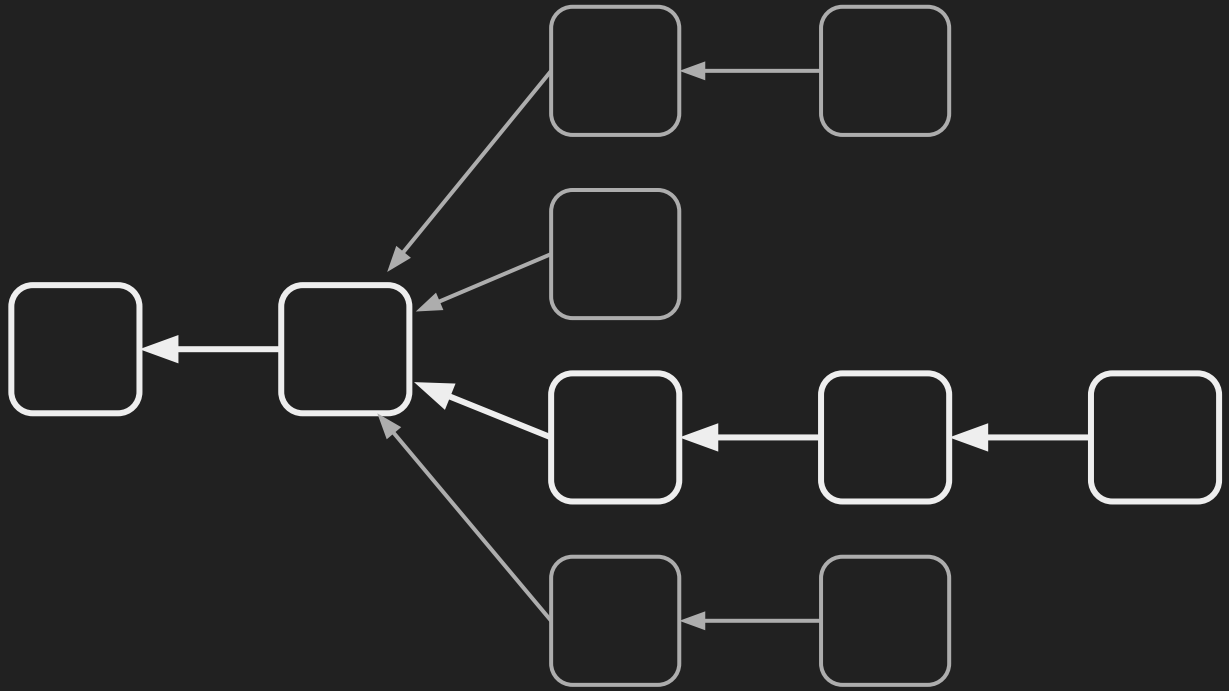
GHOST chain selection

what happens if we increase the block creation rate?



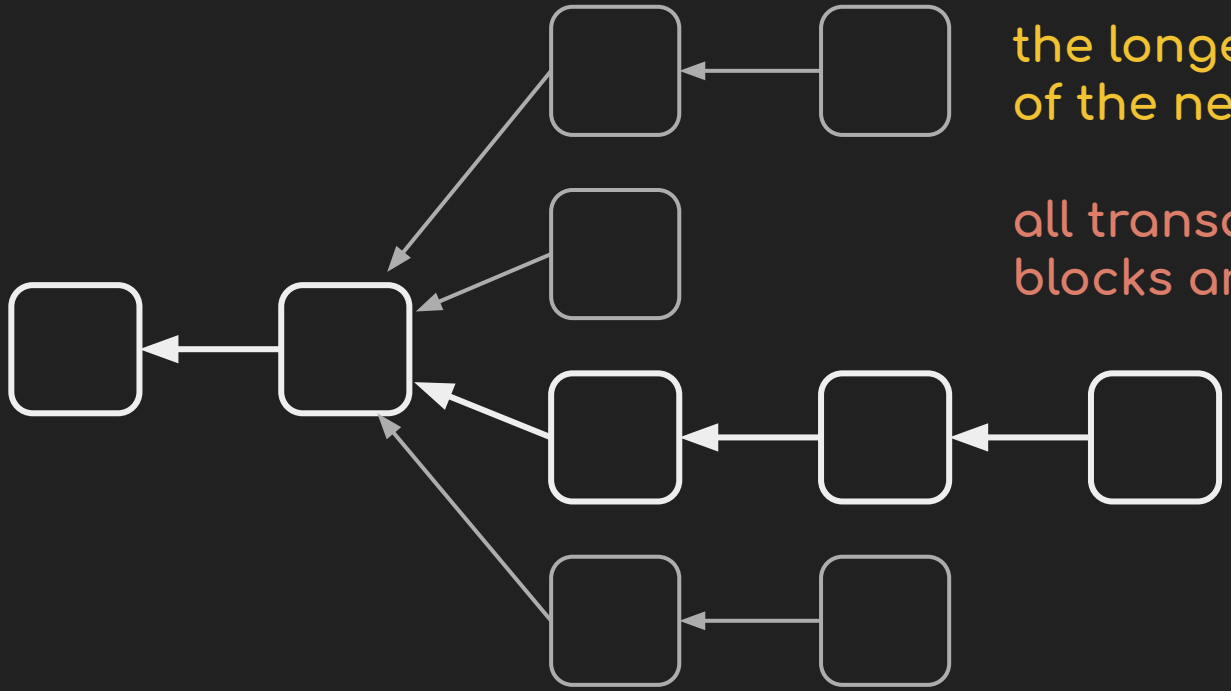
GHOST chain selection

what happens if we increase the block creation rate?



GHOST chain selection

what happens if we increase the block creation rate?

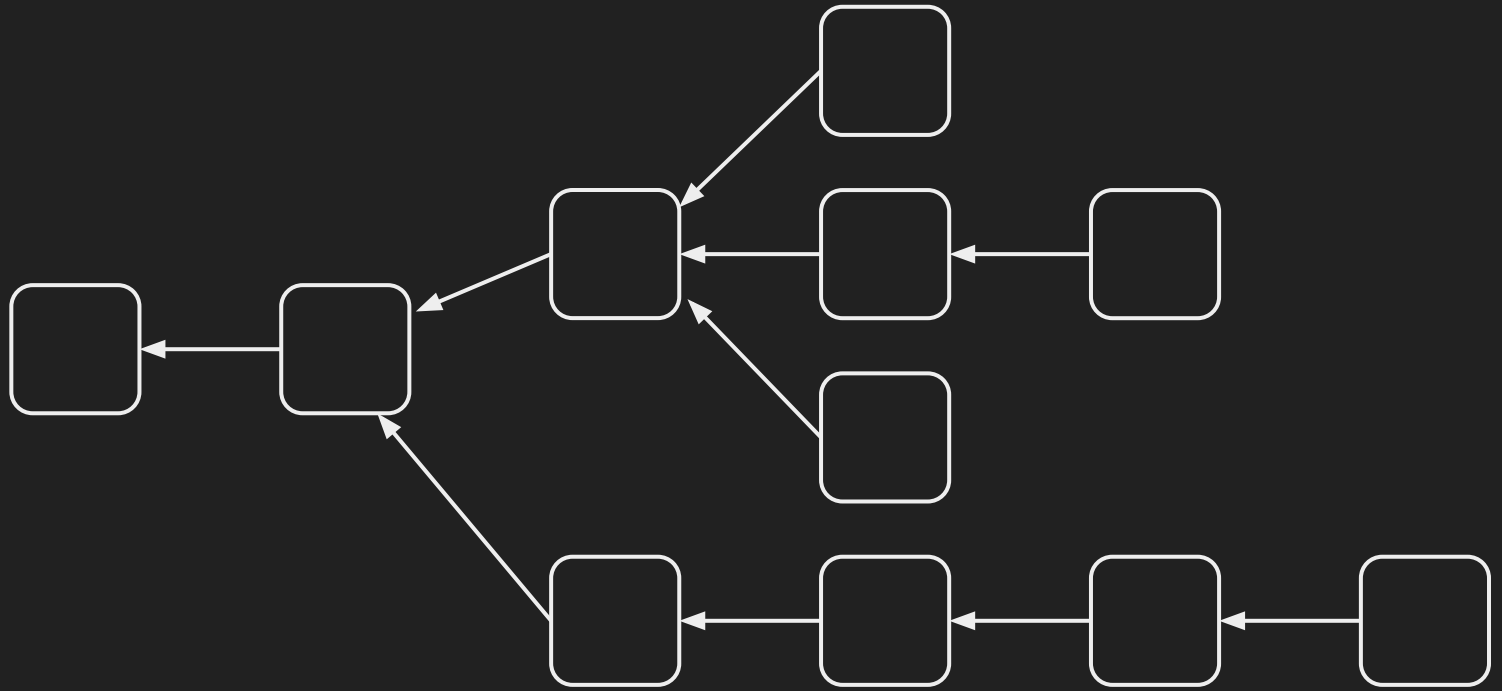


the longest chain has 37.5% of the network's hashpower

all transactions from 5 blocks are discarded

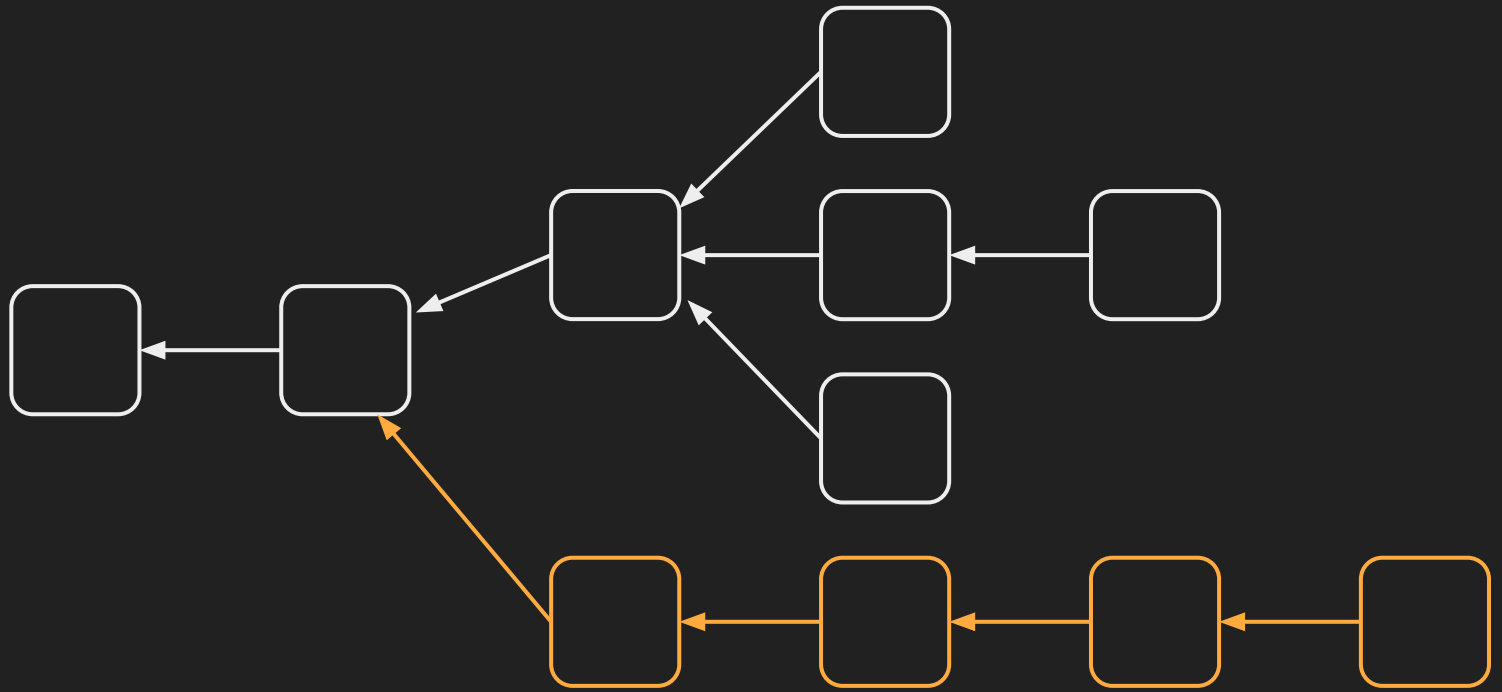
GHOST chain selection

instead of the longest chain rule, we will use the GHOST rule



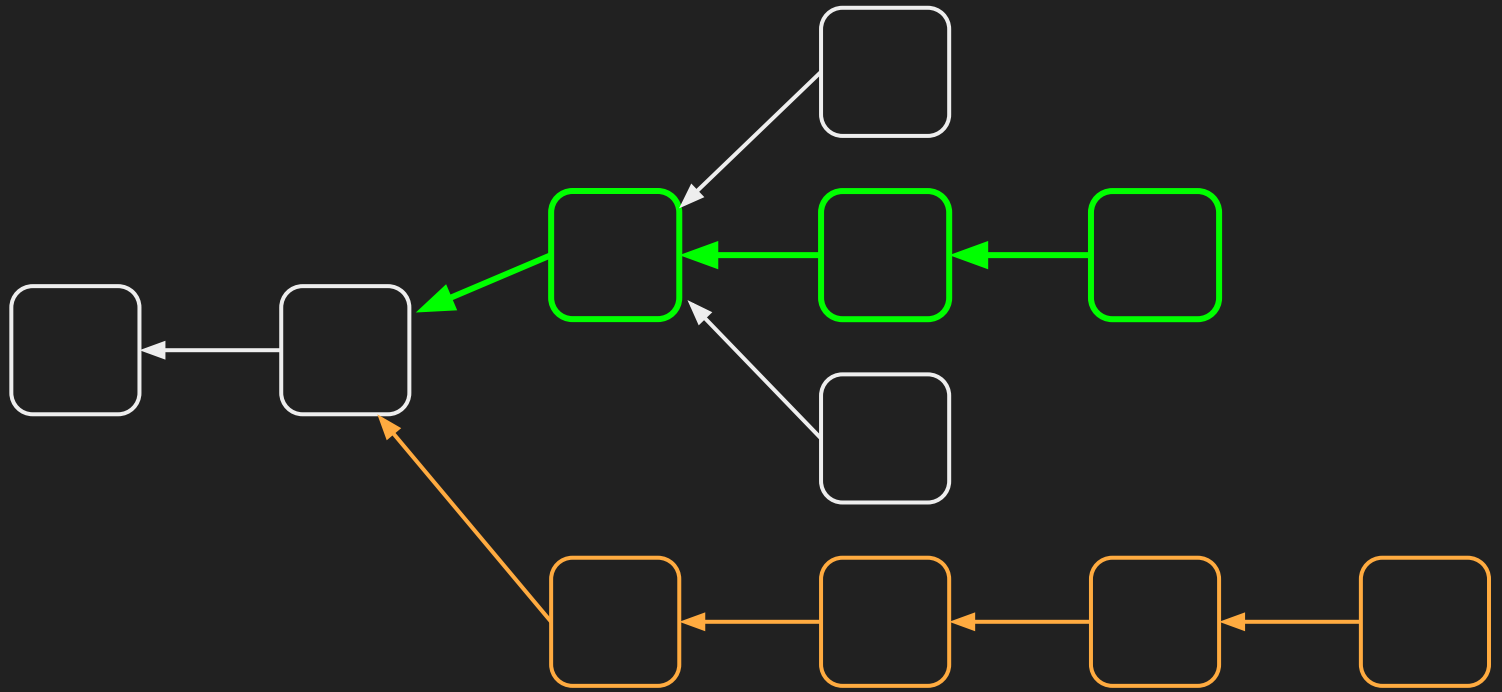
GHOST chain selection

instead of the longest chain rule, we will use the GHOST rule



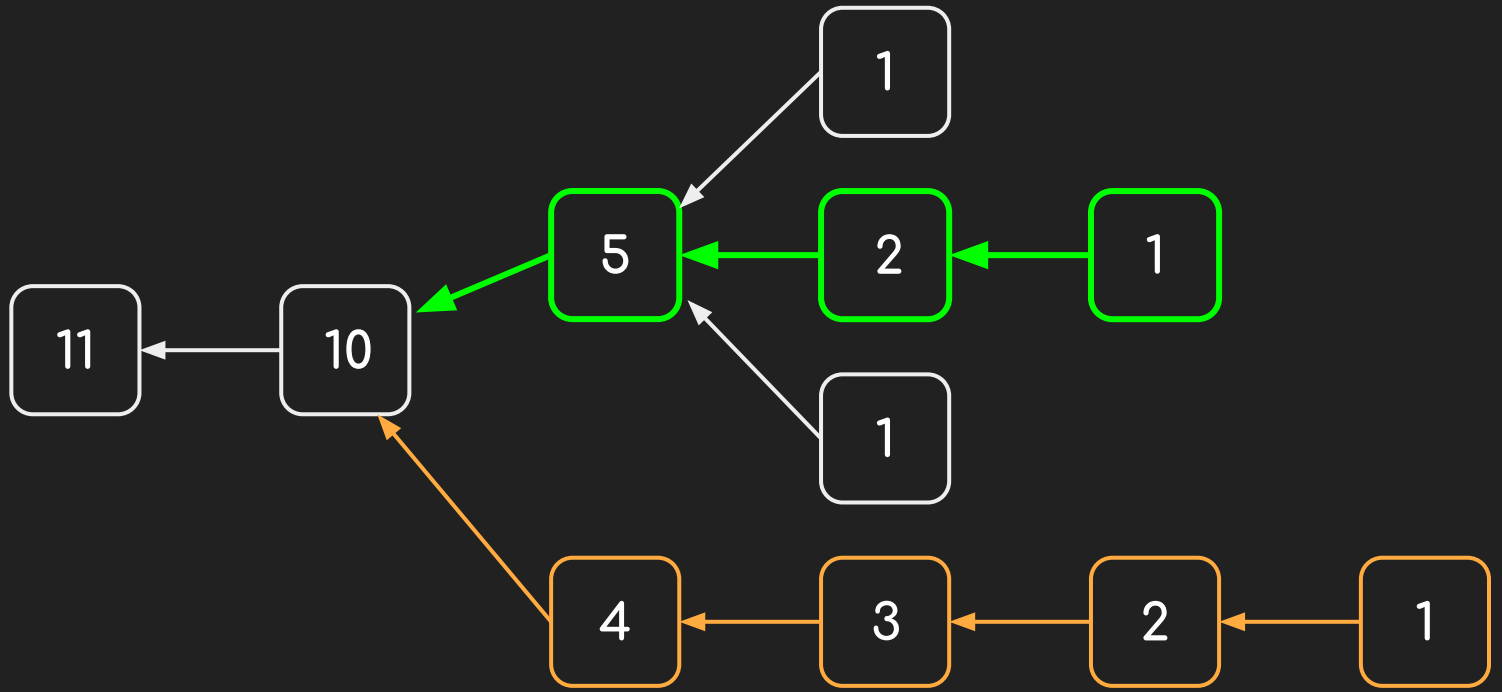
GHOST chain selection

instead of the longest chain rule, we will use the GHOST rule



GHOST chain selection

instead of the longest chain rule, we will use the GHOST rule



GHOST chain selection

instead of the longest chain rule, we will use the GHOST rule

with GHOST, we always choose the subtree of the majority

this makes GHOST more resilient, event under high block rates

Sompolinsky, Yonatan, and Aviv Zohar. "Secure high-rate transaction processing in bitcoin." 2015.

GHOST chain selection

instead of the longest chain rule, we will use the GHOST rule

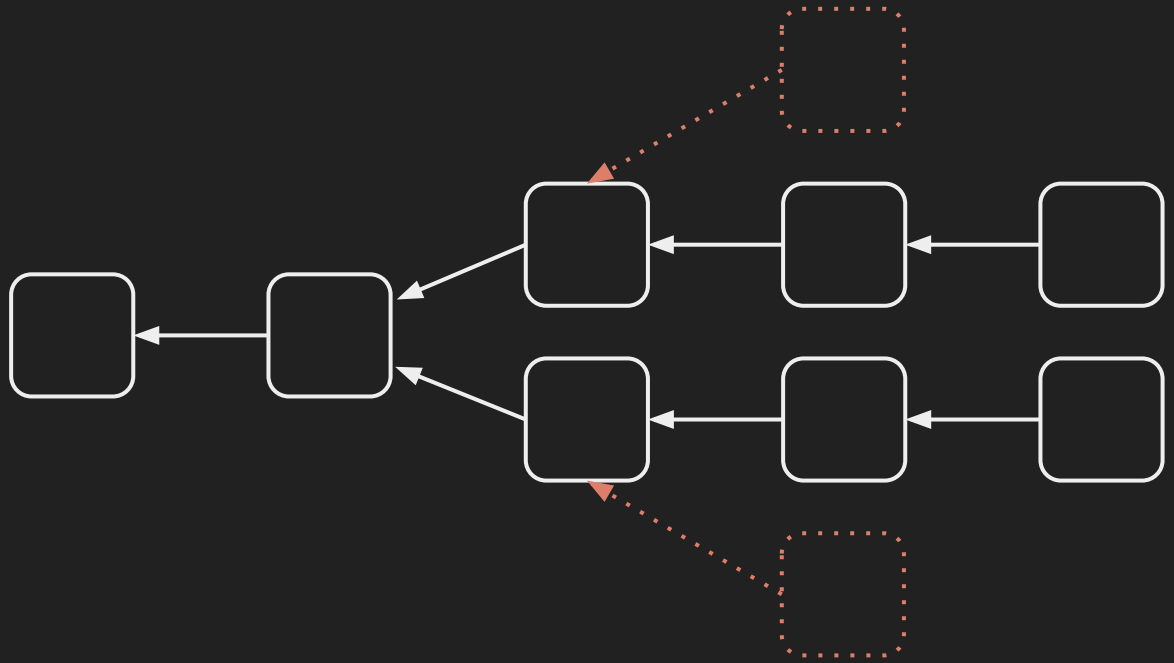
with GHOST, we always choose the subtree of the majority

this makes GHOST more resilient, event under high block rates

can GHOST still be attacked?

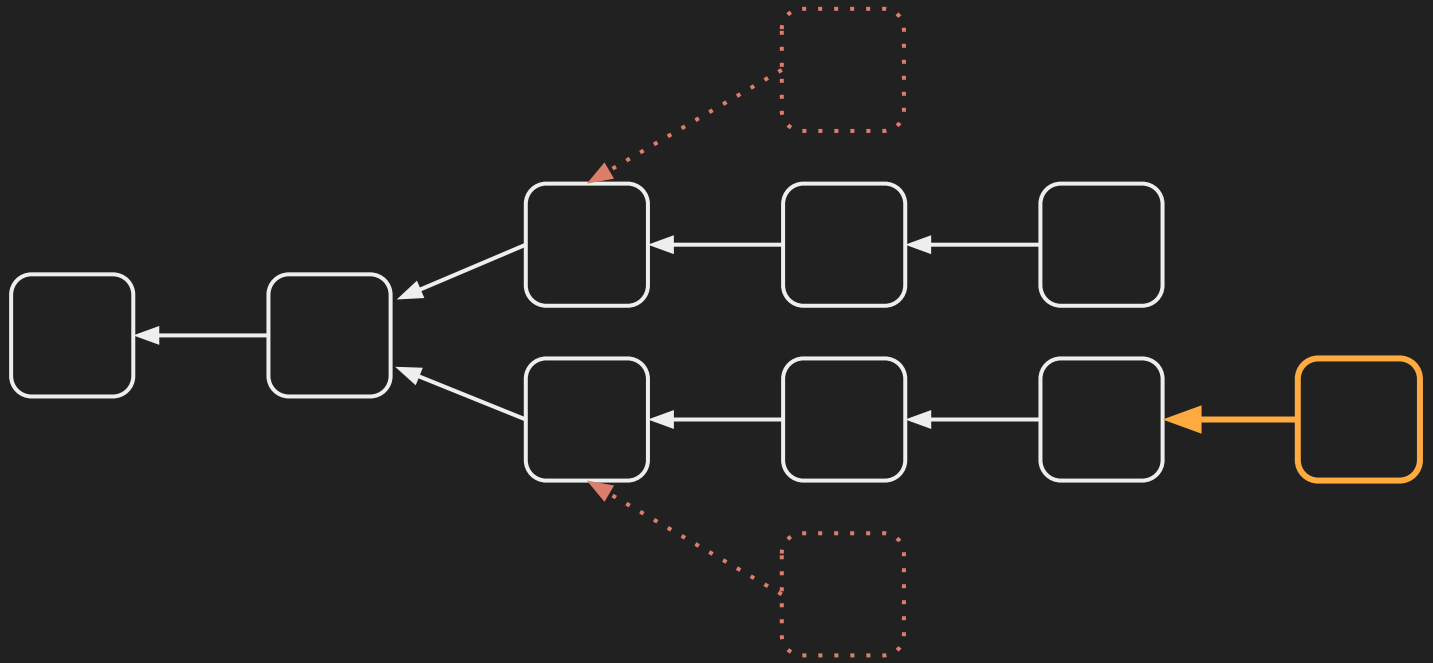
liveness attack on GHOST

an attacker pre-mine blocks and maintain long forks



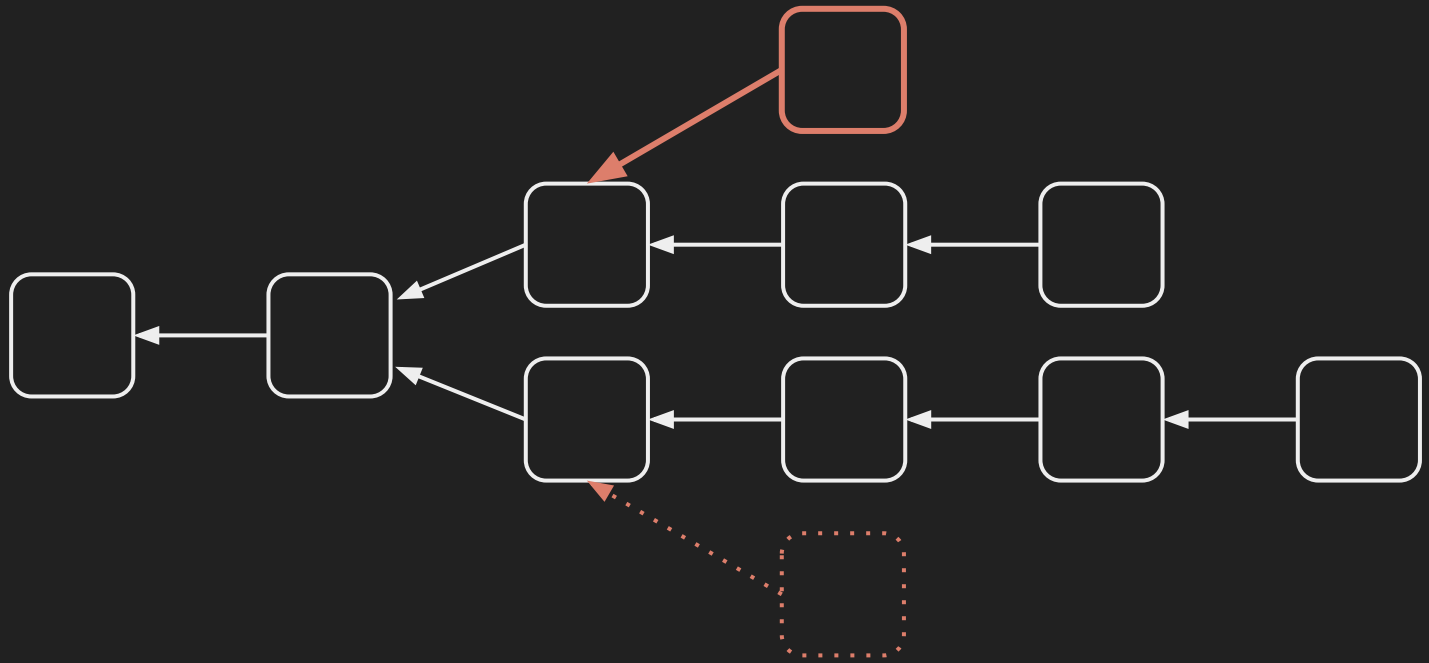
liveness attack on GHOST

an attacker pre-mine blocks and maintain long forks



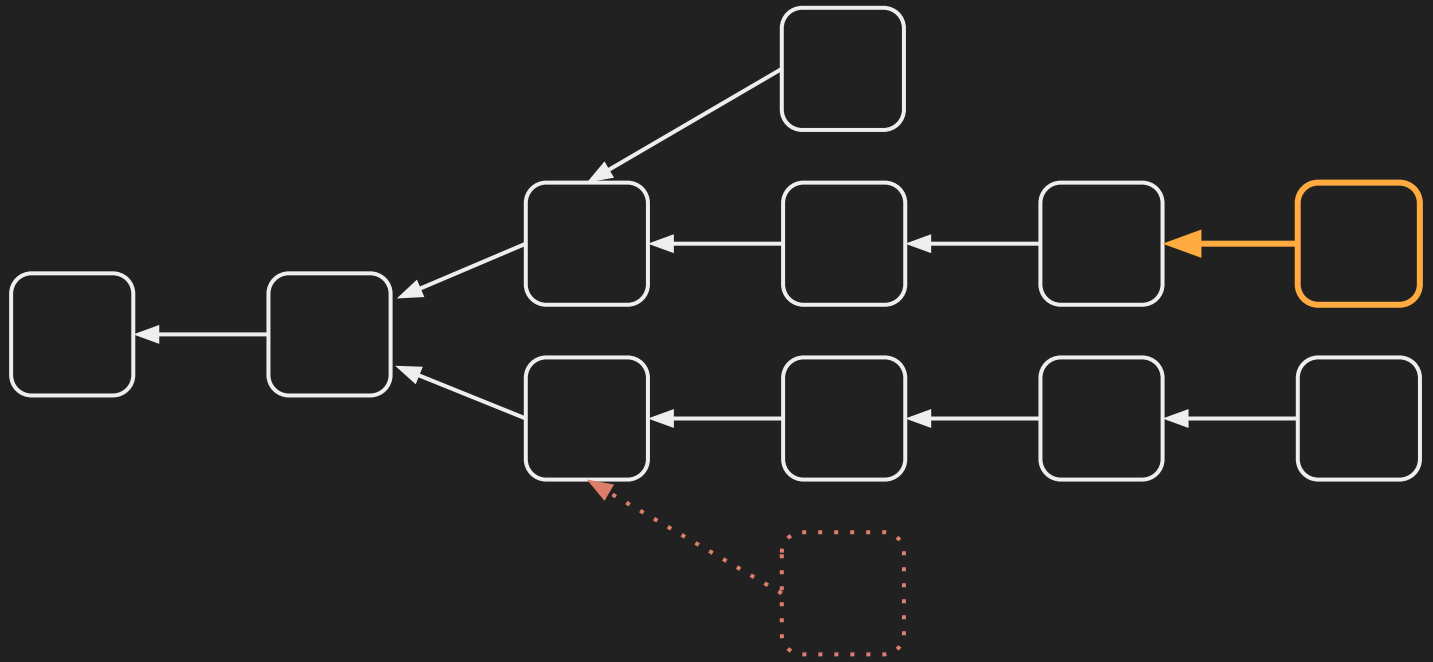
liveness attack on GHOST

an attacker pre-mine blocks and maintain long forks



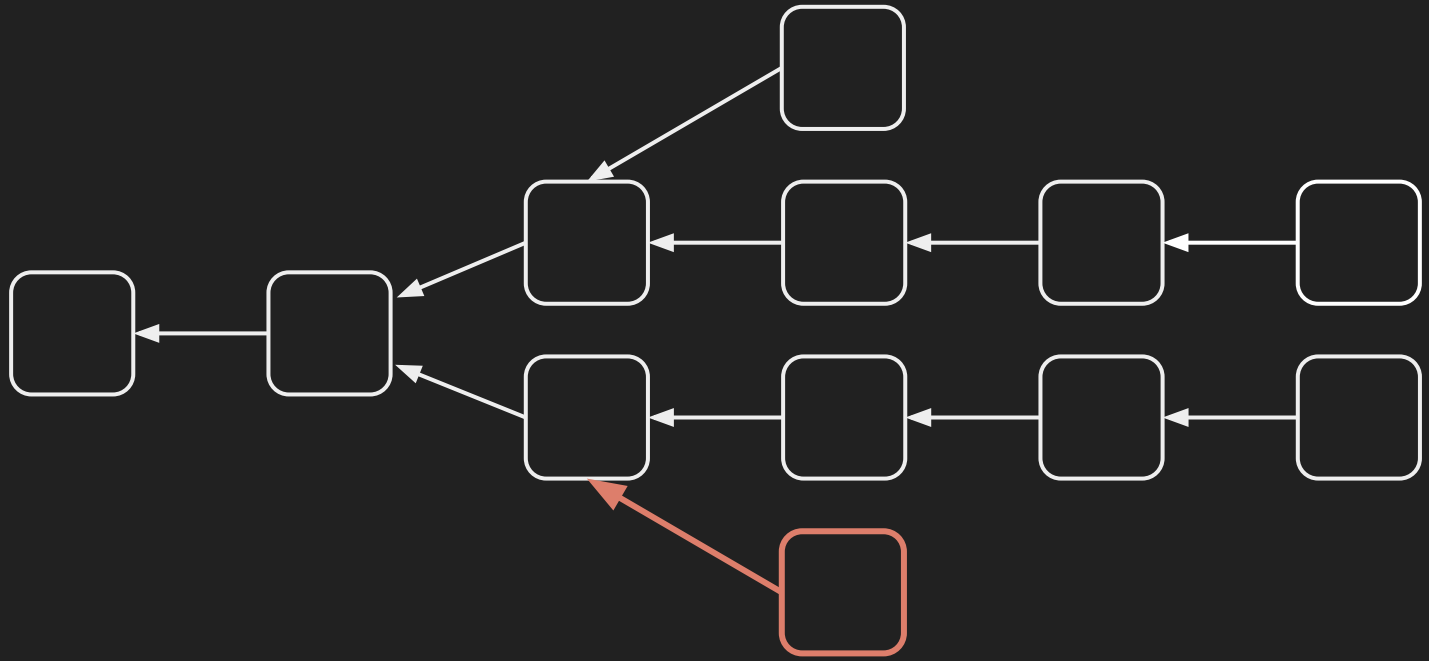
liveness attack on GHOST

an attacker pre-mine blocks and maintain long forks



liveness attack on GHOST

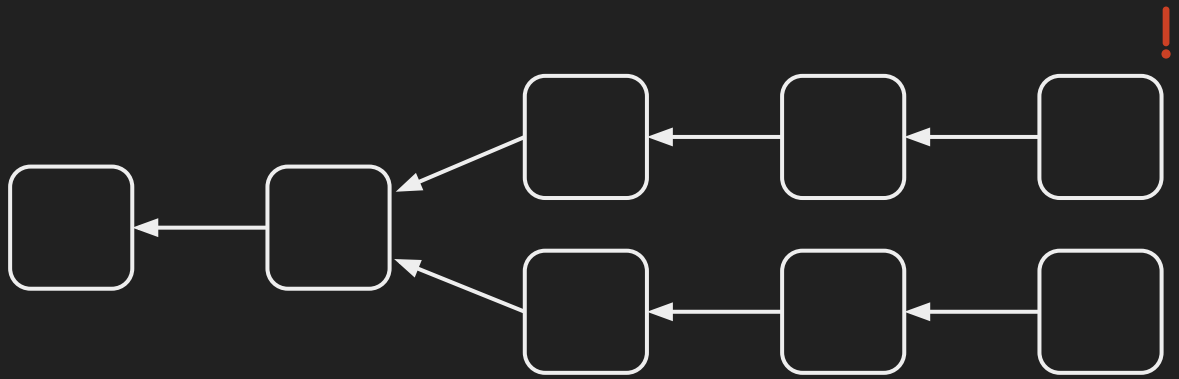
an attacker pre-mine blocks and maintain long forks



GHAST chain selection

an attacker pre-mine blocks and maintain long forks

when we detect a liveness attack, we switch to adaptive mode

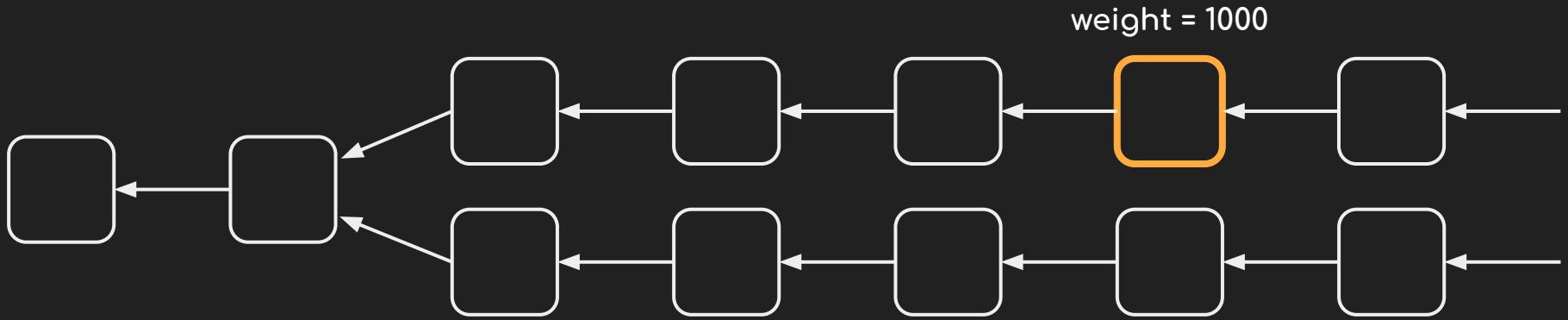


GHAST chain selection

an attacker pre-mine blocks and maintain long forks

when we detect a liveness attack, we switch to adaptive mode

occasional rare blocks will break the balance



GHAST chain selection

an attacker pre-mine blocks and maintain long forks

when we detect a liveness attack, we switch to adaptive mode

occasional rare blocks will break the balance

GHAST: Greedy Heaviest Adaptive Sub-Tree

so where are we now?

with GHOST, the chain is stable even under high block rates

with GHAST, the chain is resilient against liveness attacks

so where are we now?

with GHOST, the chain is stable even under high block rates

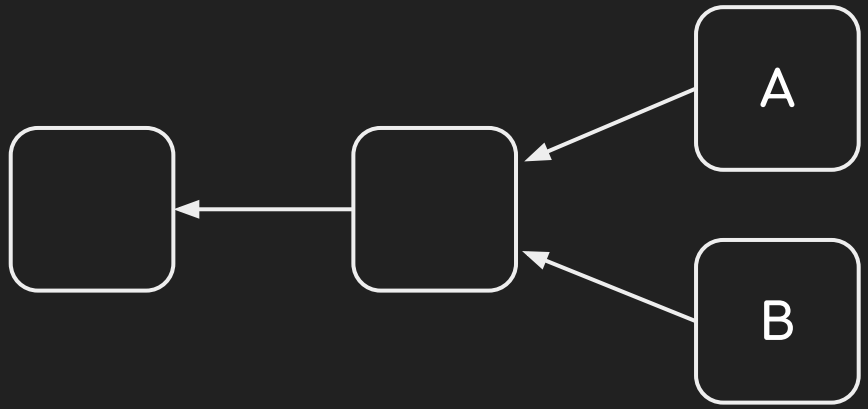
with GHAST, the chain is resilient against liveness attacks

but discarded blocks still do not contribute to throughput

is there a way we could use them?

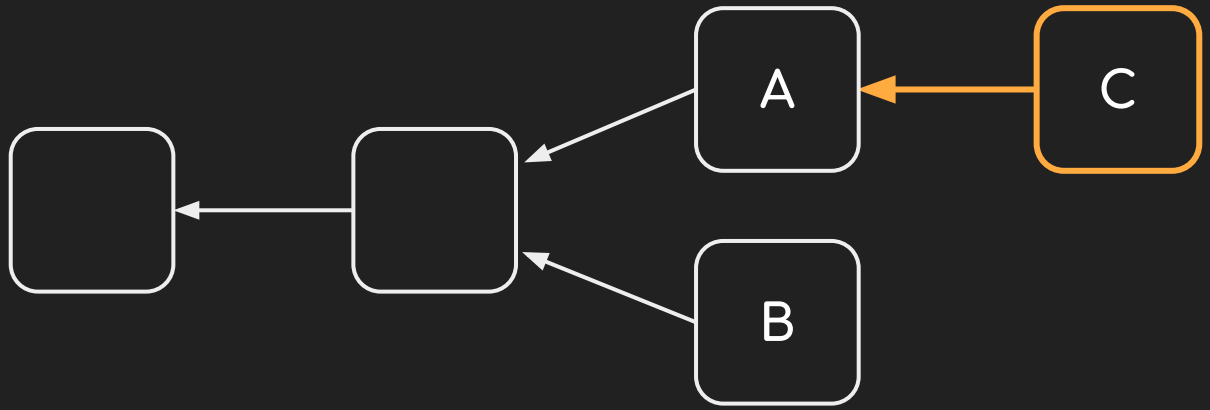
tree-graph

let's look at a simple fork



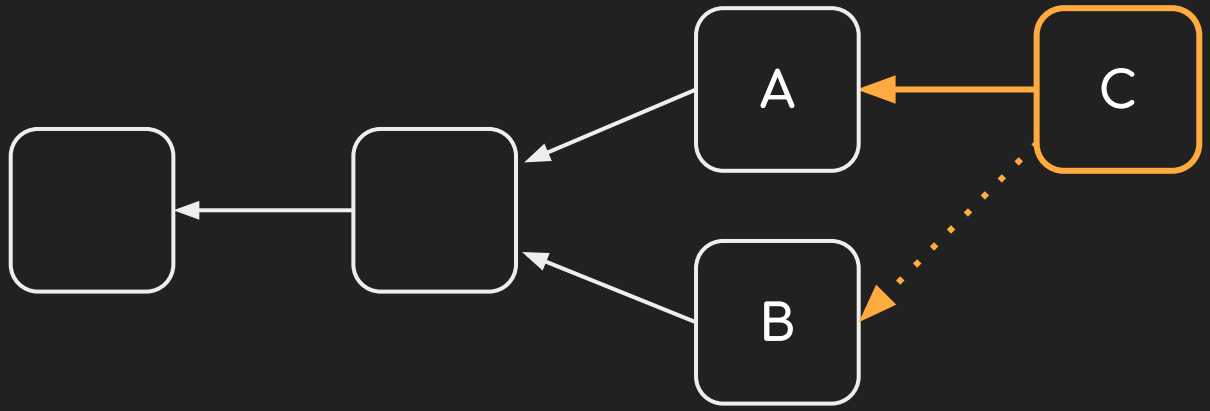
tree-graph

we know the A comes before C, but B and C are concurrent



tree-graph

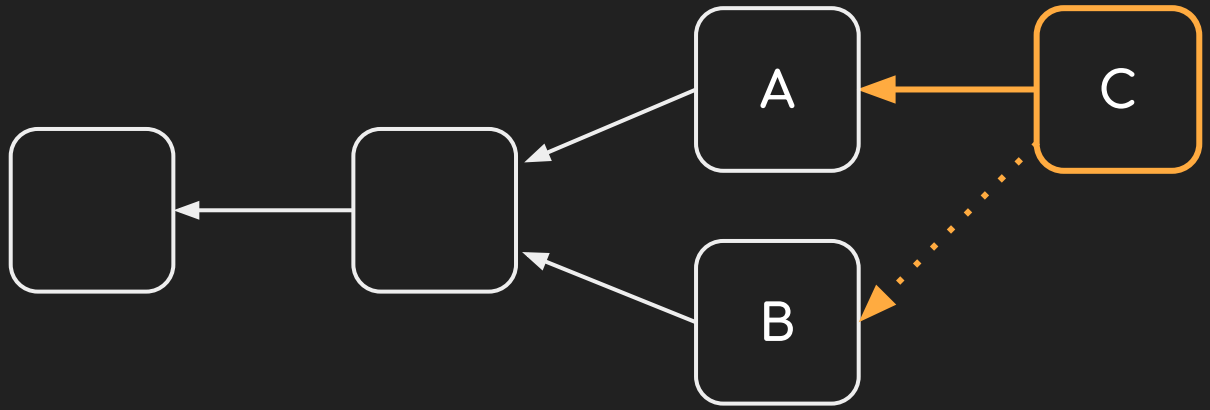
let's introduce reference edges that capture the order: B-C



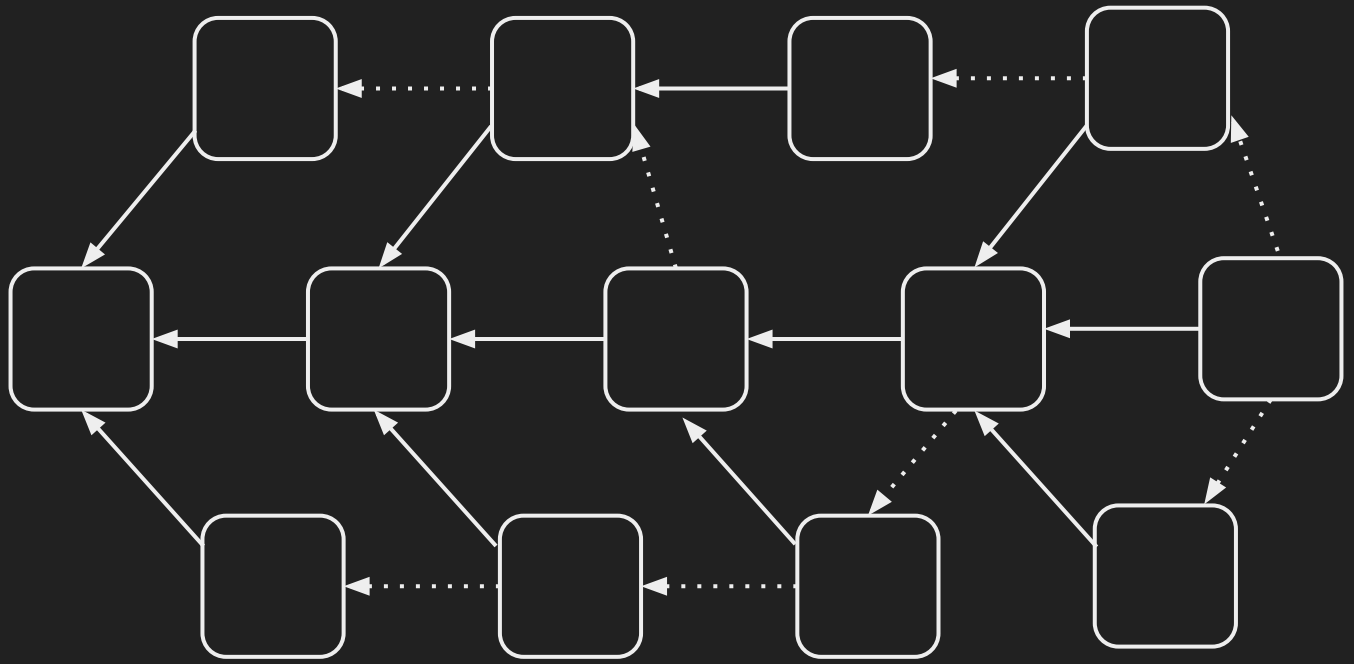
tree-graph

let's introduce reference edges that capture the order: B-C

this additional information will help us derive an order

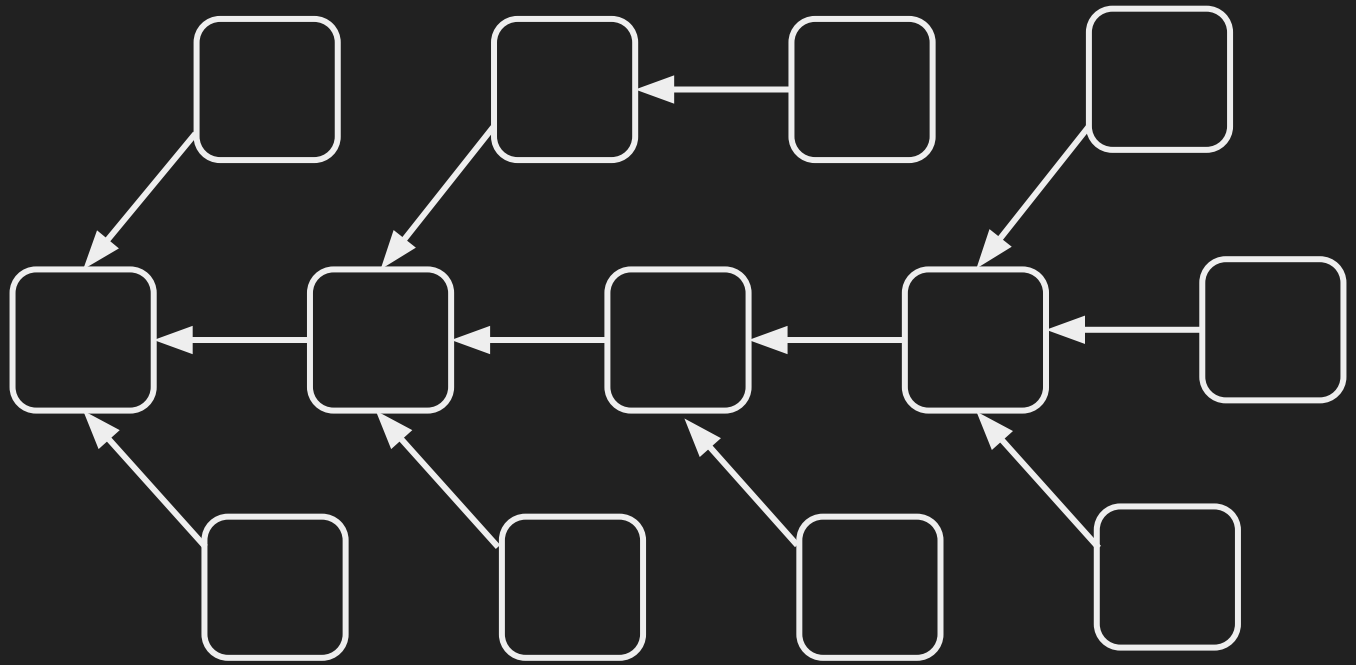


tree-graph



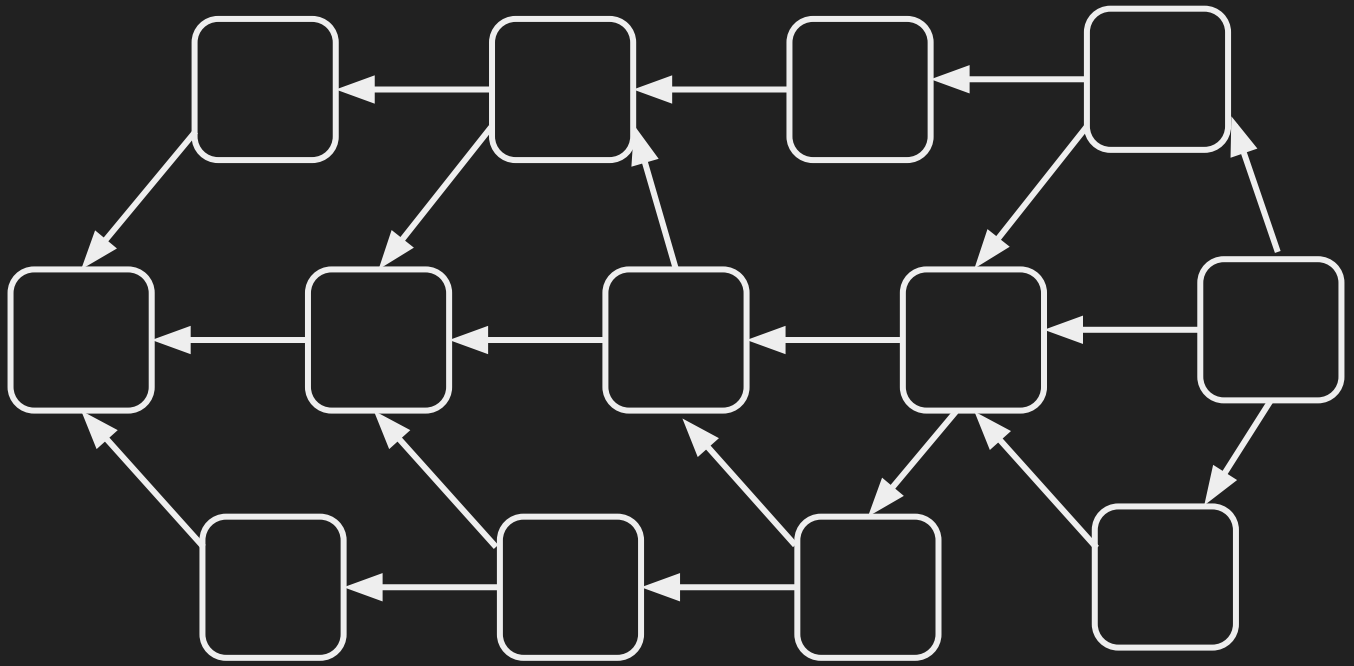
tree-graph

parent edges form a tree



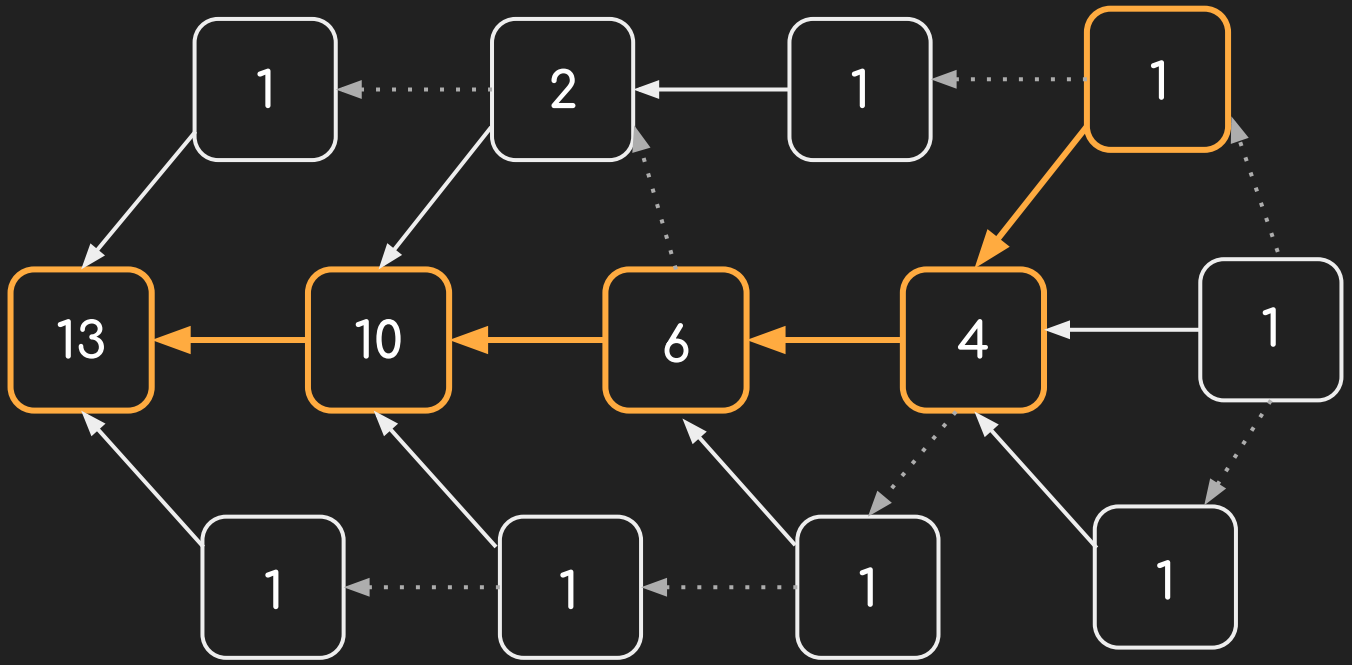
tree-graph

all edges form a DAG



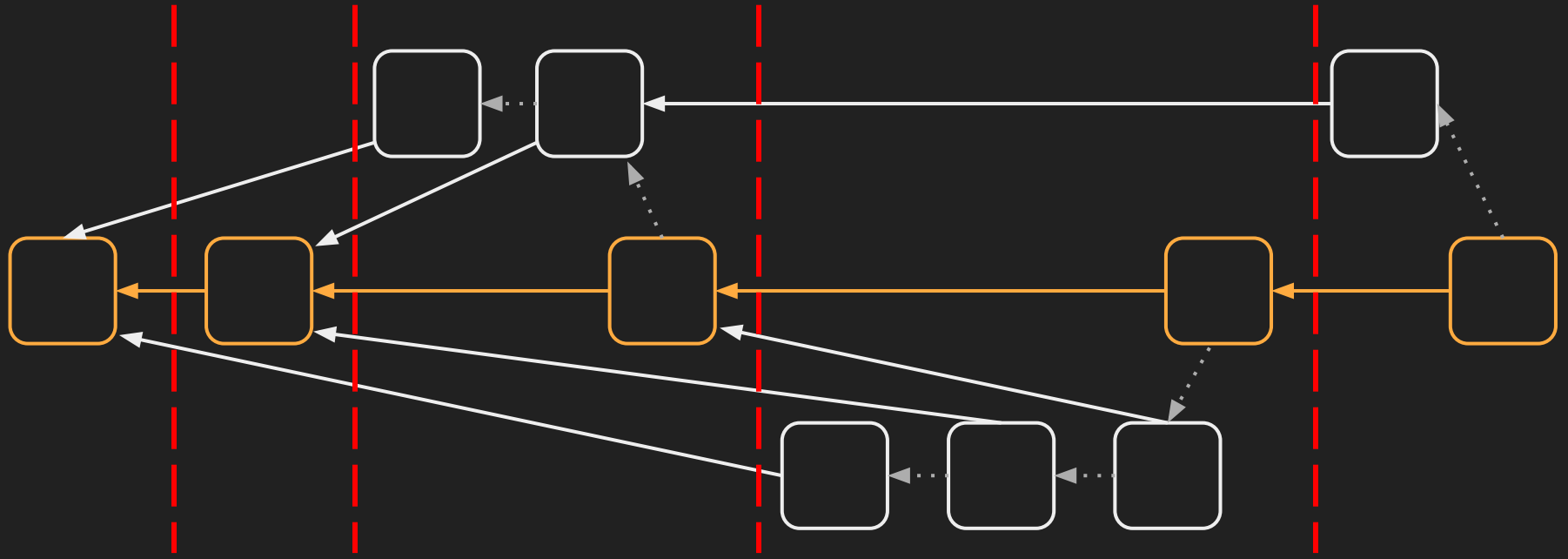
tree-graph

we perform GHAST on the parent-tree to get the pivot chain



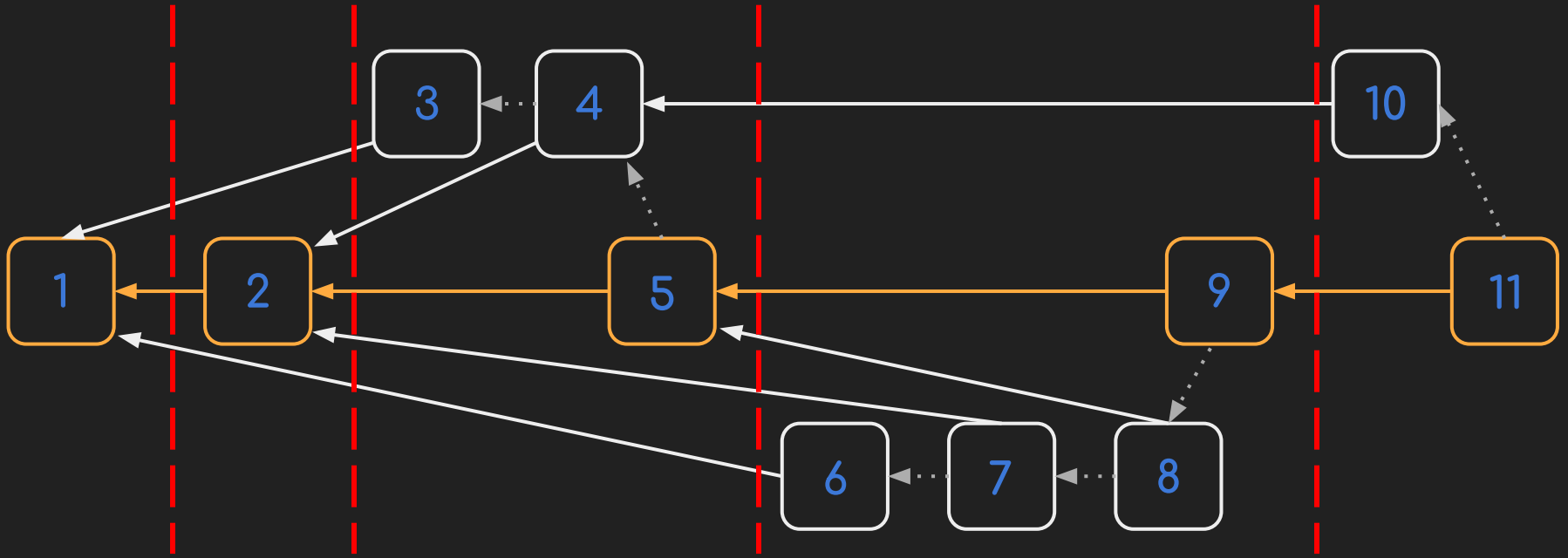
block order in the tree-graph

then, we divide the graph into epochs



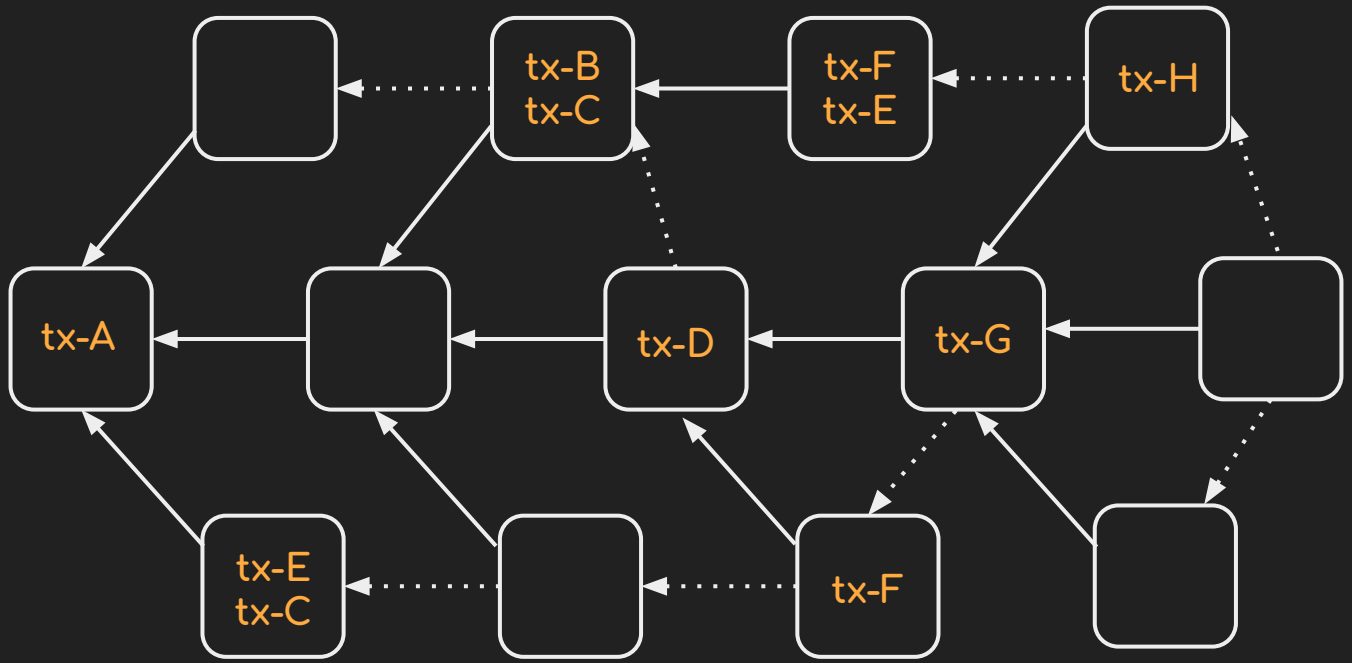
block order in the tree-graph

we can derive a linear order of blocks from the epochs

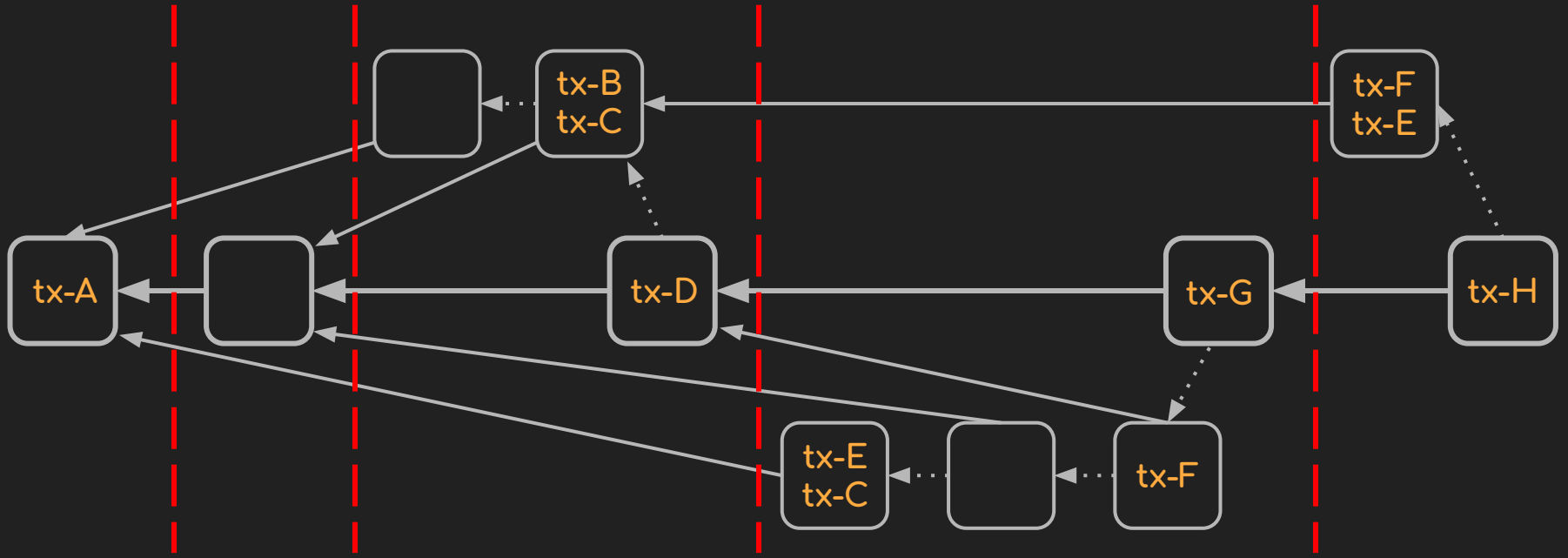


transaction order in the tree-graph

from a tree graph, we can also derive the transaction order



transaction order in the tree-graph



tx-A tx-B tx-C tx-D tx-E tx-C tx-F tx-G tx-F tx-E tx-H

things we learned today

Bitcoin uses Proof-of-Work and the longest chain rule to achieve consensus on a linear sequence of transactions

things we learned today

Bitcoin uses Proof-of-Work and the longest chain rule to achieve consensus on a linear sequence of transactions

with GHAST, the main chain remains stable, even with many forks, and it is resilient against liveness attacks

things we learned today

Bitcoin uses Proof-of-Work and the longest chain rule to achieve consensus on a linear sequence of transactions

with GHAST, the main chain remains stable, even with many forks, and it is resilient against liveness attacks

with the Tree-Graph ledger structure, we can keep transactions from all blocks, thus increasing the overall throughput

thank you!

resources

Sompolinsky, Yonatan, and Aviv Zohar. "Secure high-rate transaction processing in bitcoin." 2015.
https://eprint.iacr.org/2013/881.pdf

Li, Chenxin, Peilun Li, Dong Zhou, Zhe Yang, Ming Wu, Guang Yang, Wei Xu, Fan Long, and Andrew Chi-Chih Yao. "A decentralized blockchain with high throughput and fast confirmation." 2020.
https://www.usenix.org/conference/atc20/presentation/li-chenxing

Conflux technical presentation
https://confluxnetwork.org/files/Conflux_Technical_Presentation_20200309.pdf