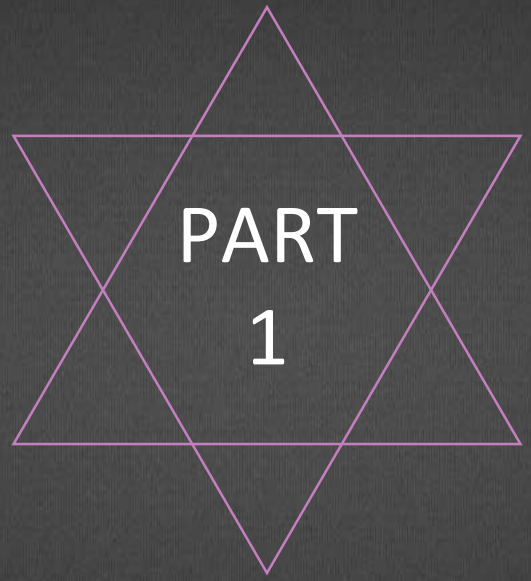
Hashgraph

Yu Xia

Junlin Yin

Zhaojin Li



Introduction

Hedera Project introduction

Hedera is designed for fast, fair, and secure applications to take advantage of the efficiency of hashgraph on a decentralized, public network you can trust.

FAST FAIR SECURE STABLE

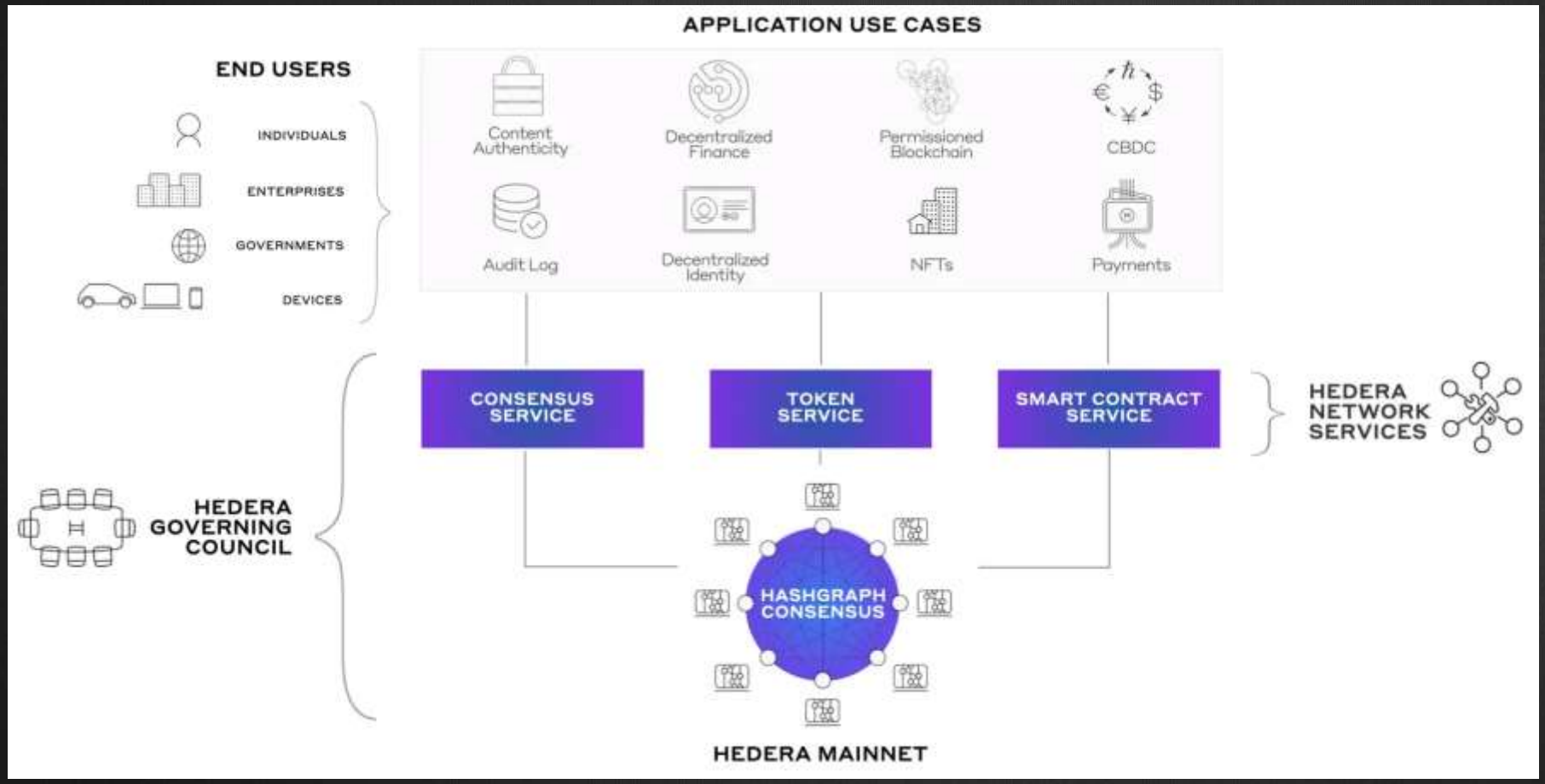
finality in 3~5 sec

FAIR ACCESS
TIMESTAMPS
TRANSACTION ORDER

asynchronous
Byzantine Fault
Tolerant
(aBFT)

no fork

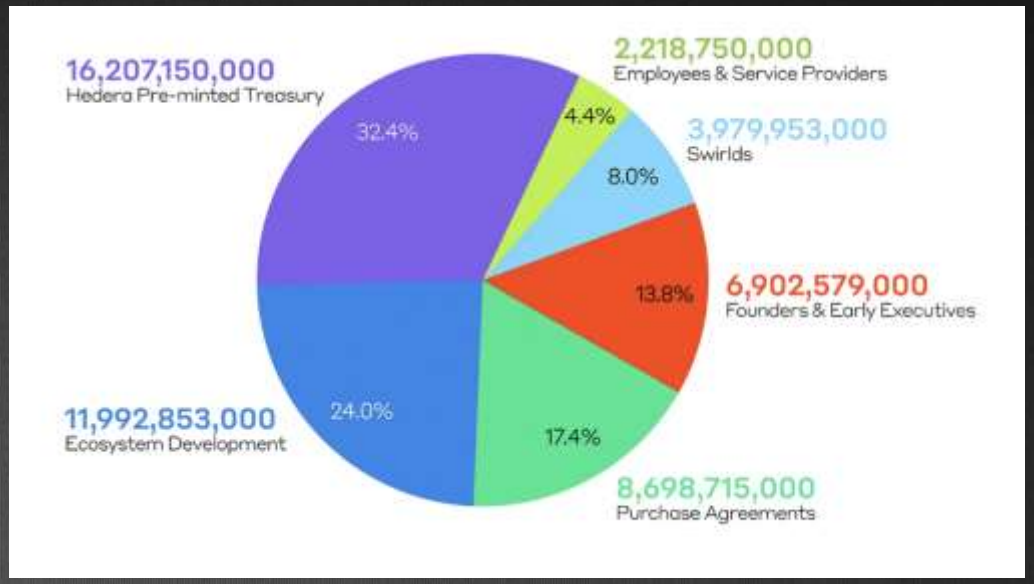
Hedera Project introduction

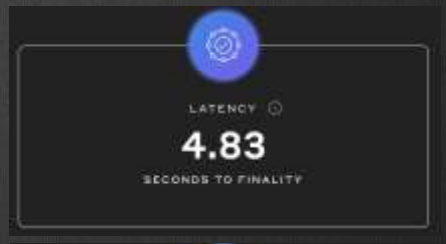


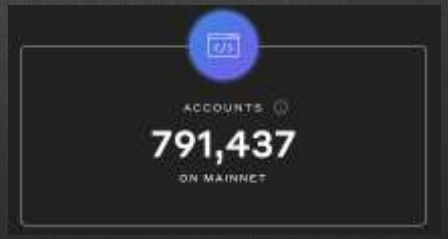
ALLOCATIONS

performance

	1ST GENERATION BITCOIN BTC	2ND GENERATION ETHEREUM ETH	3RD GENERATION HEDERA HBAR
TRANSACTIONS PER SECOND	3+ TPS	12+ TPS	10,000+ TPS*
AVERAGE FEE	$22.57 USD**	$19.55 USD^	$0.0001 USD
TRANSACTION CONFIRMATION	10-60 MINUTES	10-20 SECONDS	3-5 SECONDS (w/ finality)
ENERGY CONSUMPTION PER TRANSACTION	885+ KWH^^	102+ KWH^^	0.00017 KWH^^







Scale

Transaction volume by network service

Understand Hedera network services usage on a per-second basis.

Month | Week | Day | Hour

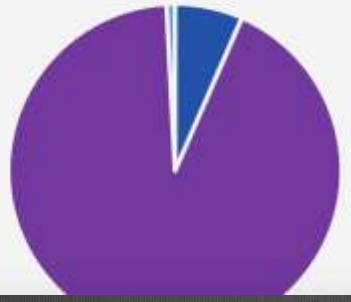
Avg. Daily TPS - Last Month: **24.0**

Max Daily TPS - Last Month: **33.1**

Total Transactions: **66.3 Mil**

Transactions per Second - Last Month

Share of Transactions by Service



Total number of accounts created

Research the rate at which accounts on the network are growing.

Total Accounts (All-Time): **800 K**

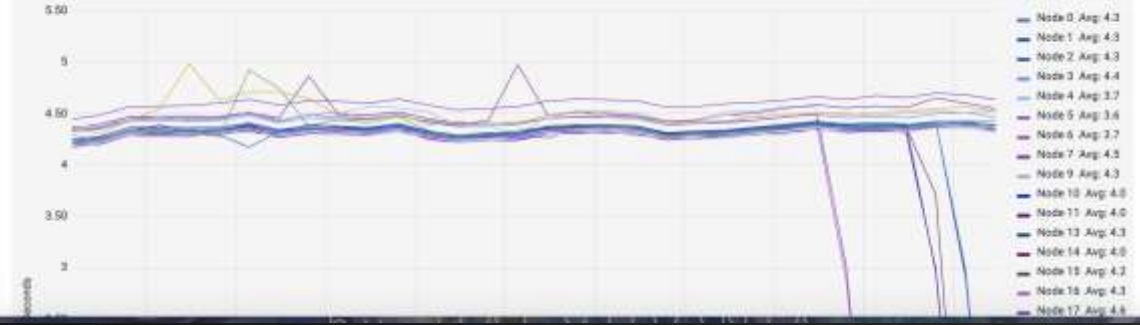
Created Accounts (Last Month): **62.7 K**

Time to consensus finality

See how efficiently hashgraph is functioning to deliver low-latency transaction finality.

Last Month | Last Week | Last Day | Last Hour

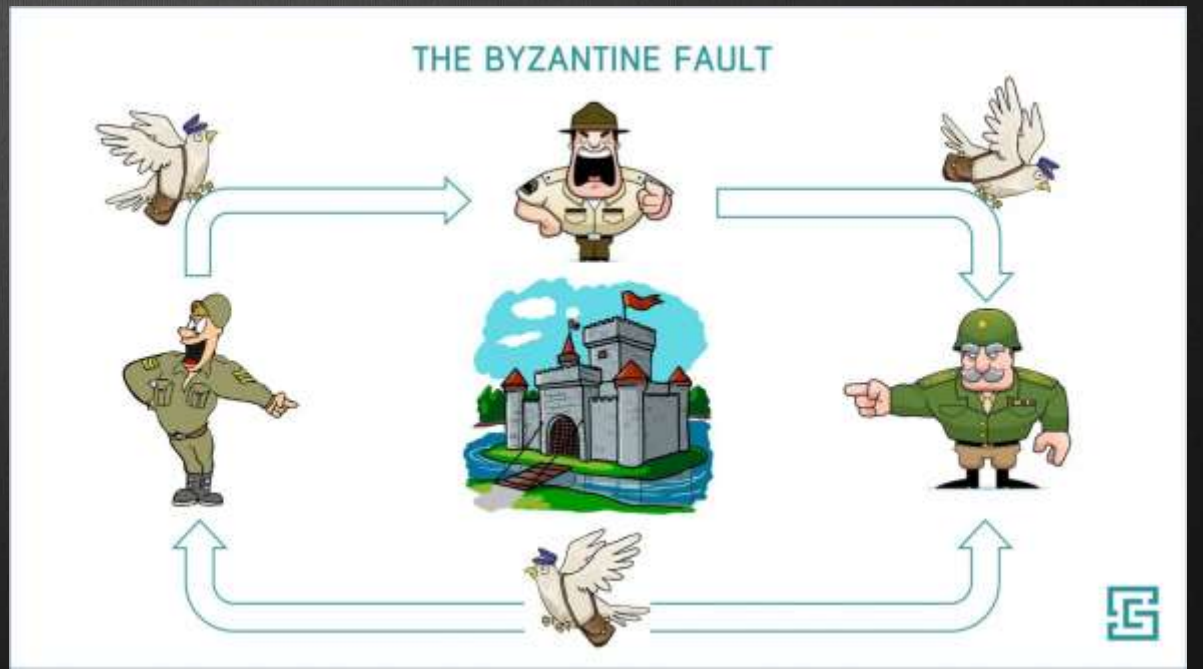
Avg. Create-to-Consensus Latency (1 Day Intervals)



Hedera consensus: asynchronous Byzantine Fault Tolerance

Hedera consensus	Byzantine Fault Tolerance(BFT)	Asynchronous Byzantine Fault Tolerance(ABFT)
assumption	maximum threshold of message latency	no maximum threshold of message latency

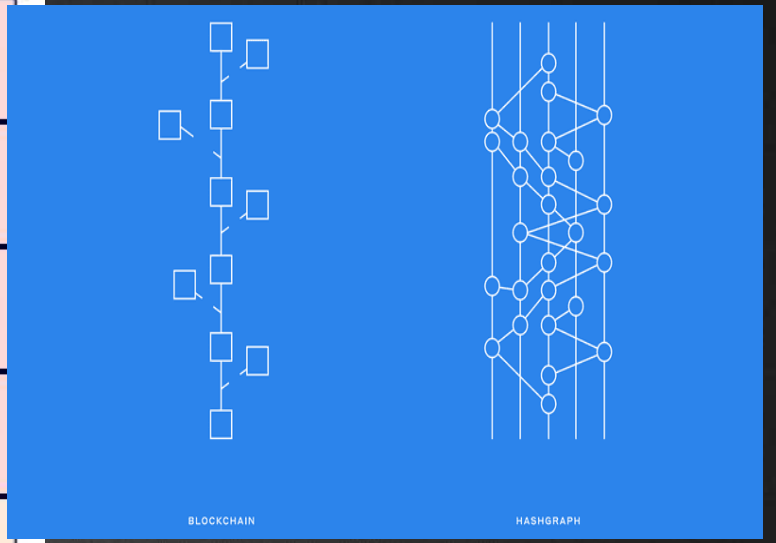
- Derived from Byzantine General Problems



Source: What is asynchronous byzantine fault tolerance (ABFT)? | Hedera
https://medium.com/coinmonks/a-note-from-anthony-if-you-havent-already-please-read-the-article-gaining-clarity-on-key-787989107969

Consensus Comparison

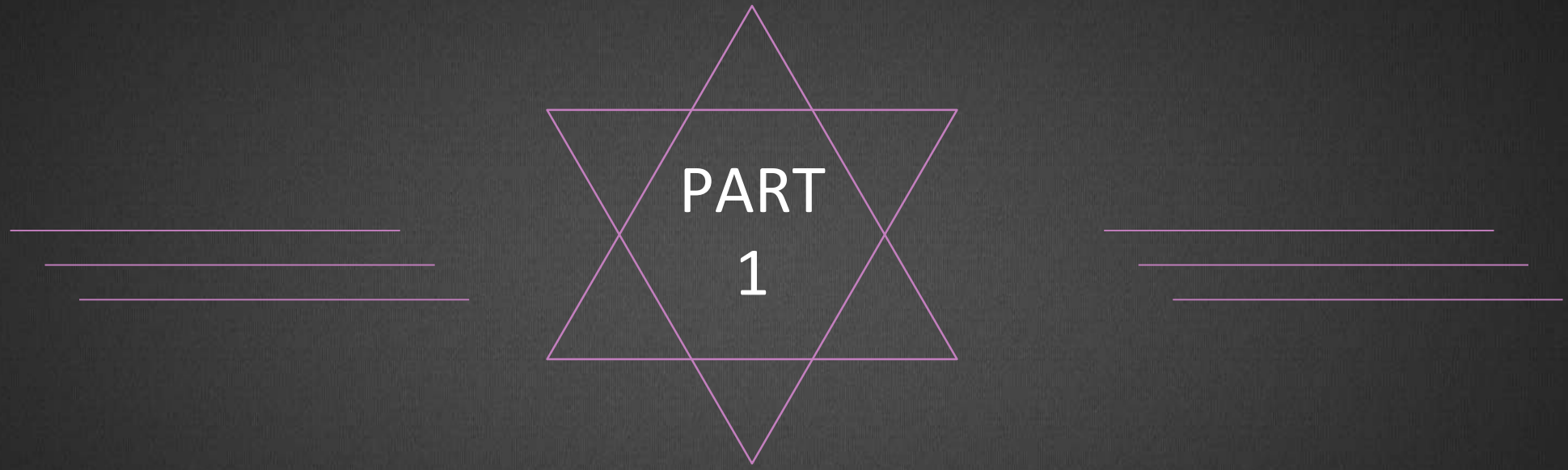
Block chain based Application: Bitcoin, Ethereum	Proof of Work	Transaction speed	Bitcoin 7/s	
			Ethereum 15/s	
		Transaction fee	0.03-10.00$	
		Security	51 % attack Ξ	
Hedera Hashgraph: HBAR	Asynchronous Byzantine Fault Tolerance	Transaction speed	hedera 10K/s	
		Transaction fee	0.0001$ Ξ	
		Security	At most 1/3 bad nodes, BUT still works	Achieve consensus with 100% finality



how hash graph work ?

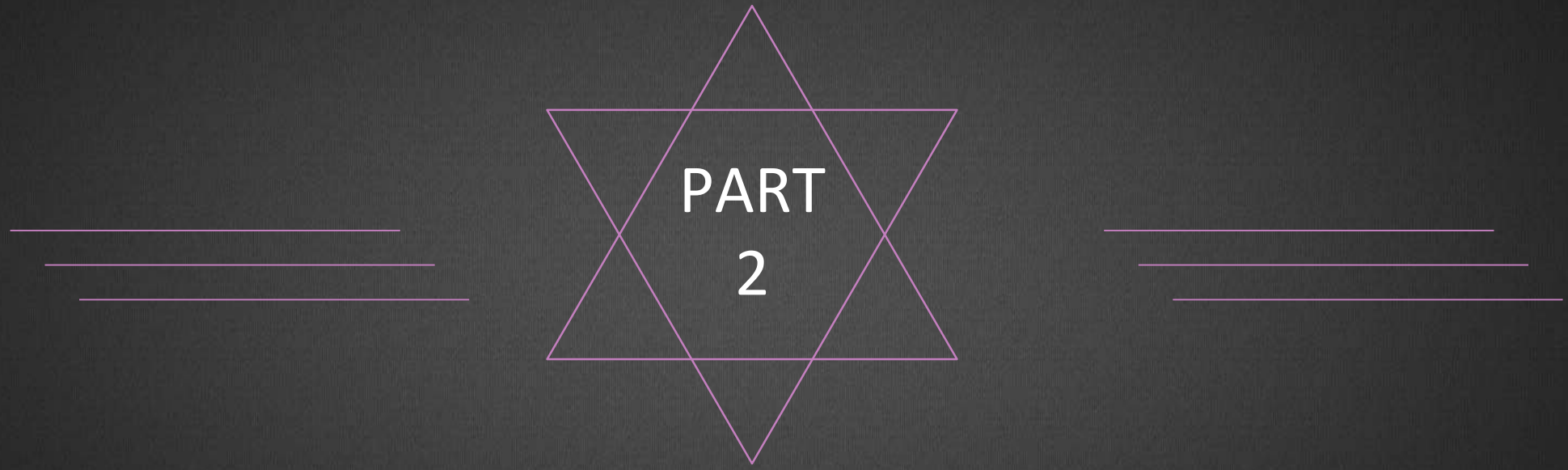
Hashgraph

content



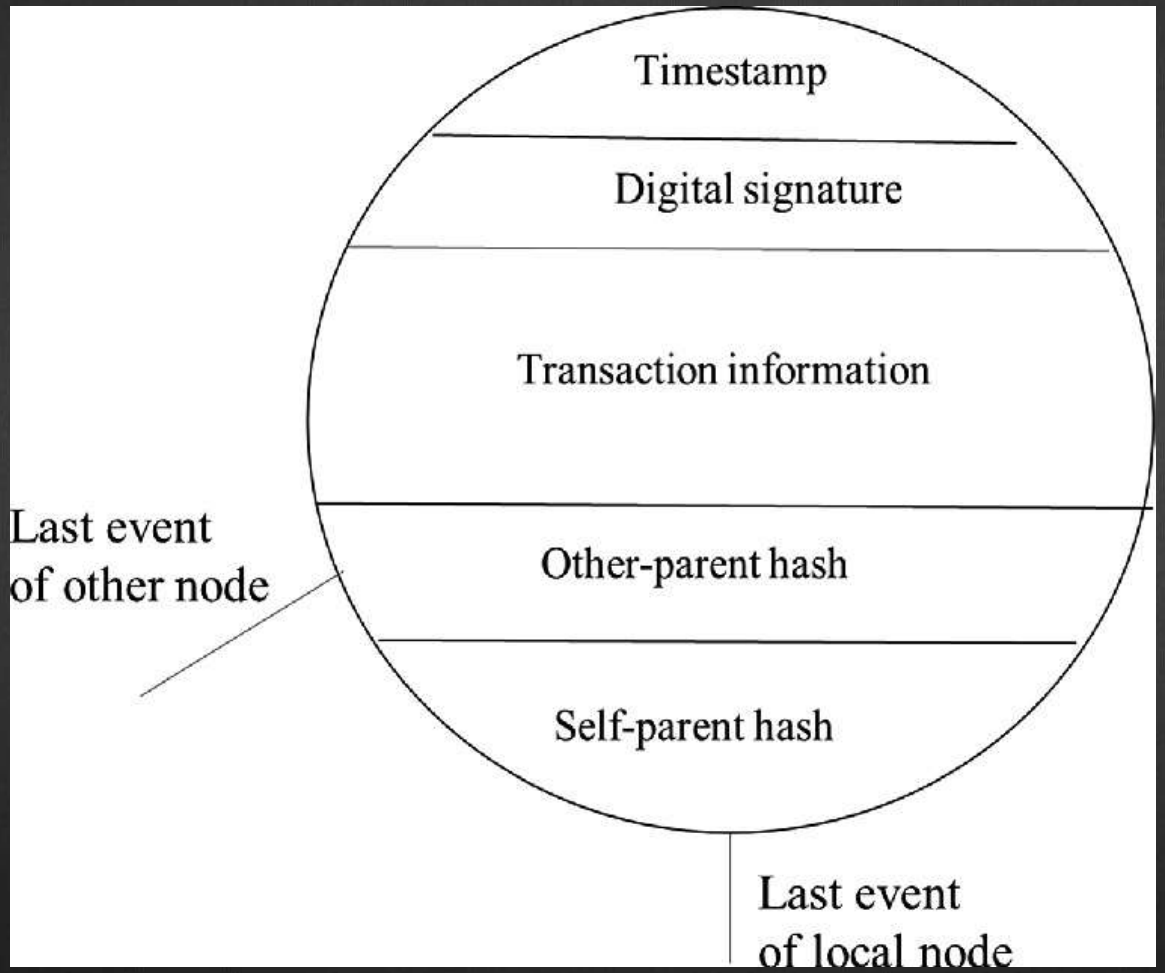
Introduction

Hashgraph uses DAG (directed acyclic graph) structure
and BFT model,
and assume the network is **completely asynchronous.**

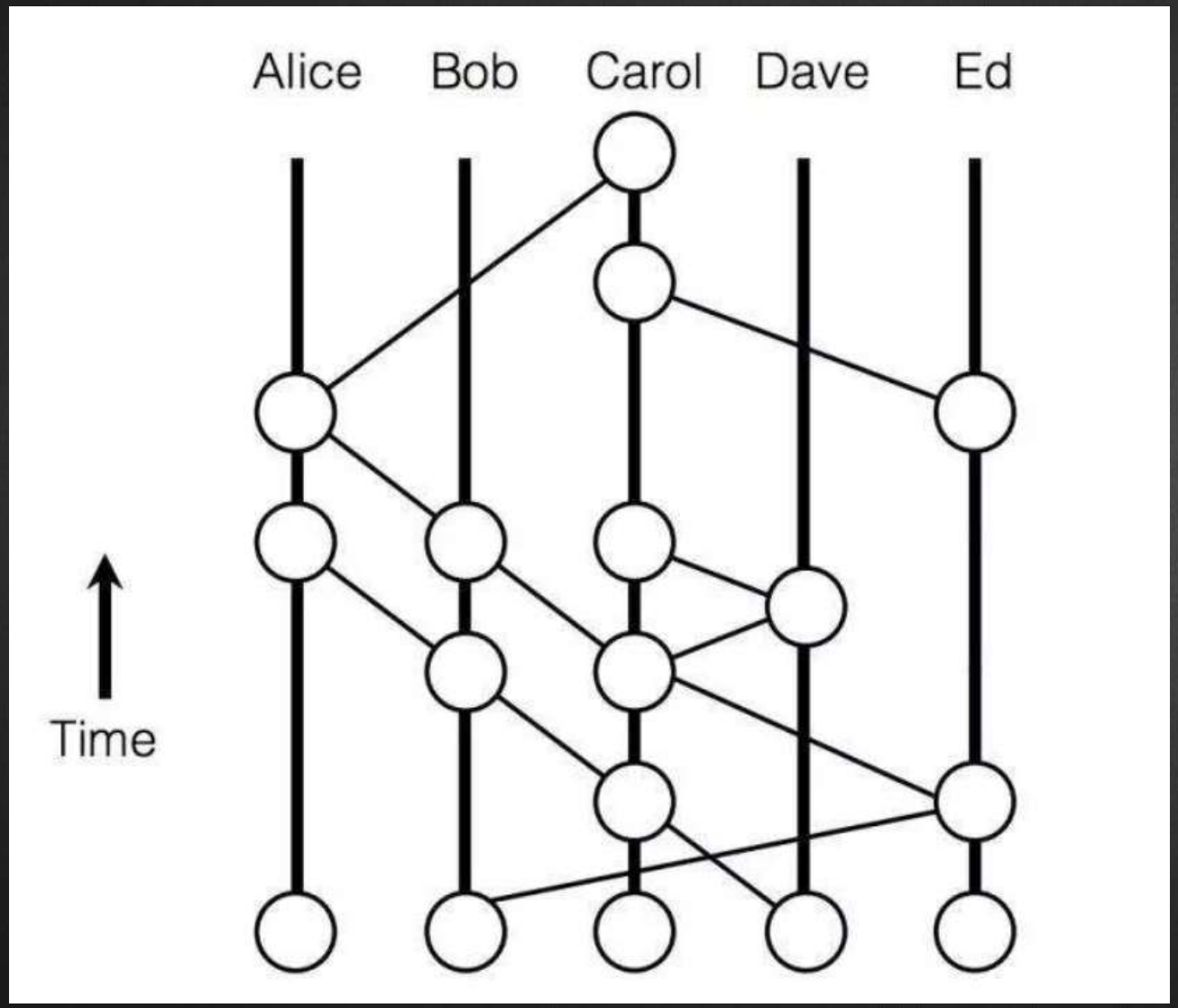


Data Structure

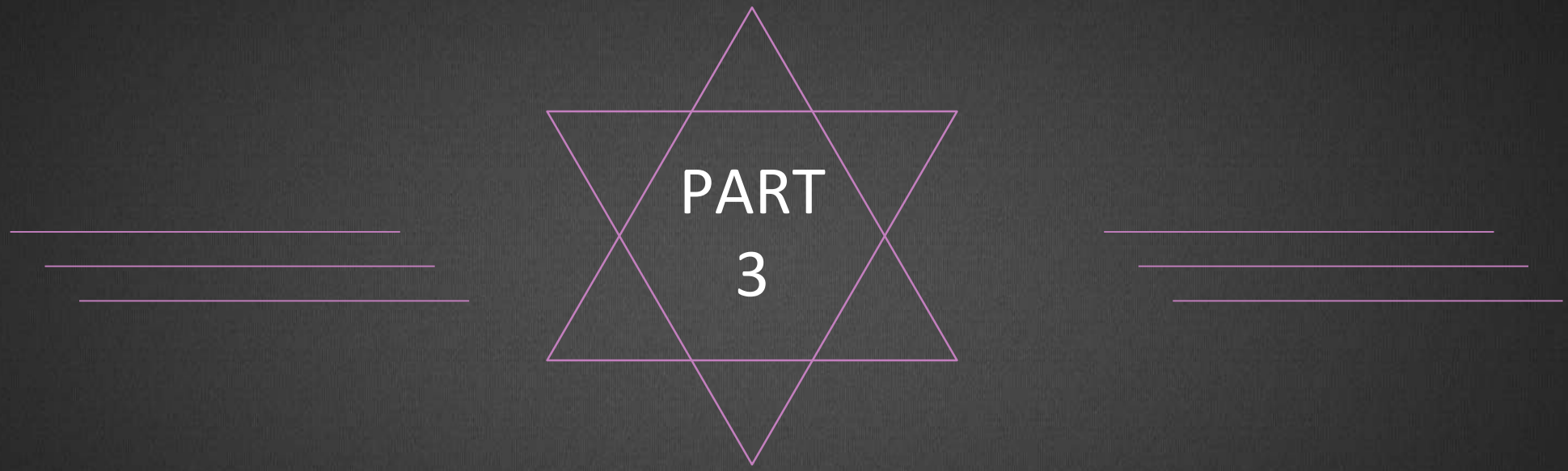
Event


Timestamp
Digital signature
Transaction information
Other-parent hash
Self-parent hash
Last event
of other node
Last event
of local node

Hashgraph



Transactions are grouped into event.
Each column represent a node.
All the nodes hold the nearly same hashgraph.
Each node once receive an event from others, a new event will be created and spread out.



How hashgraph works

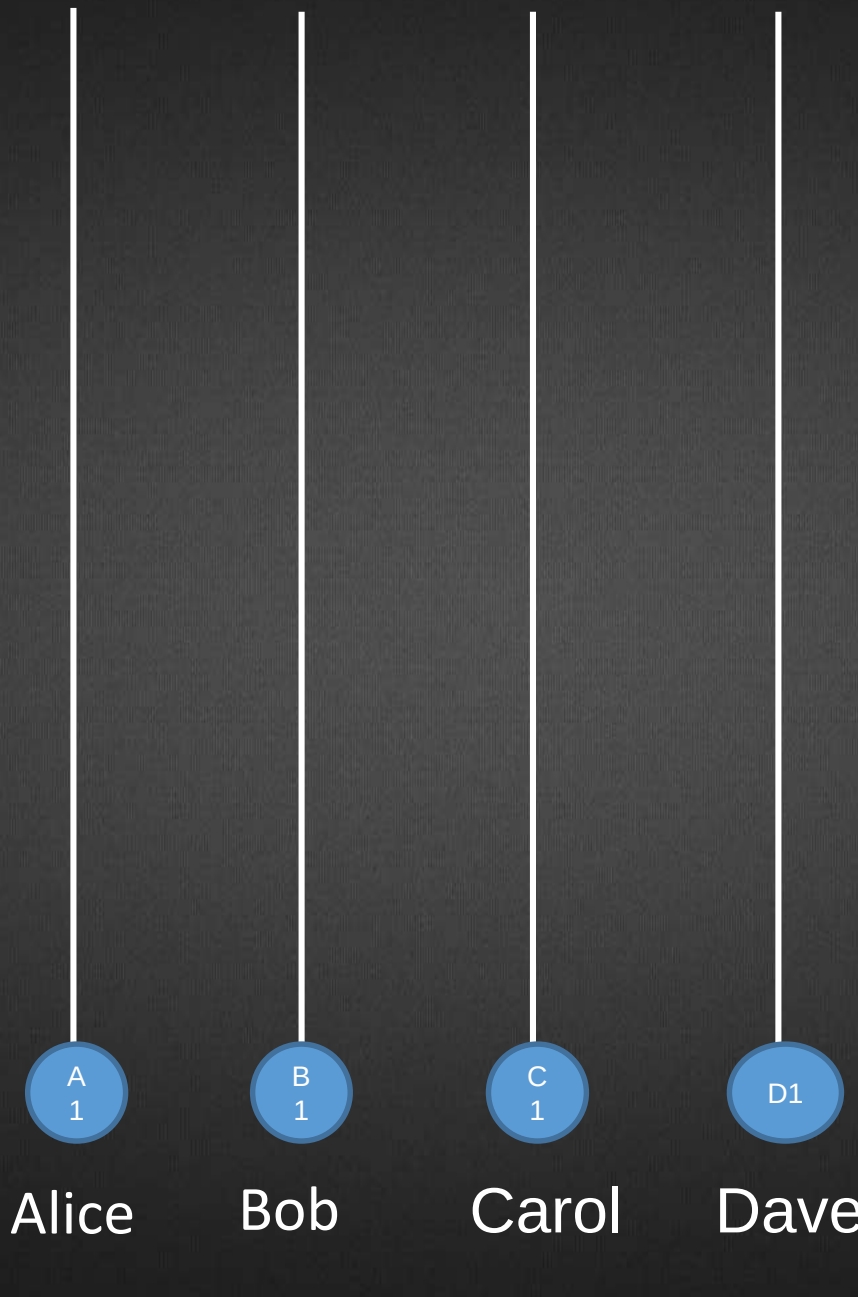
Gossip about Gossip

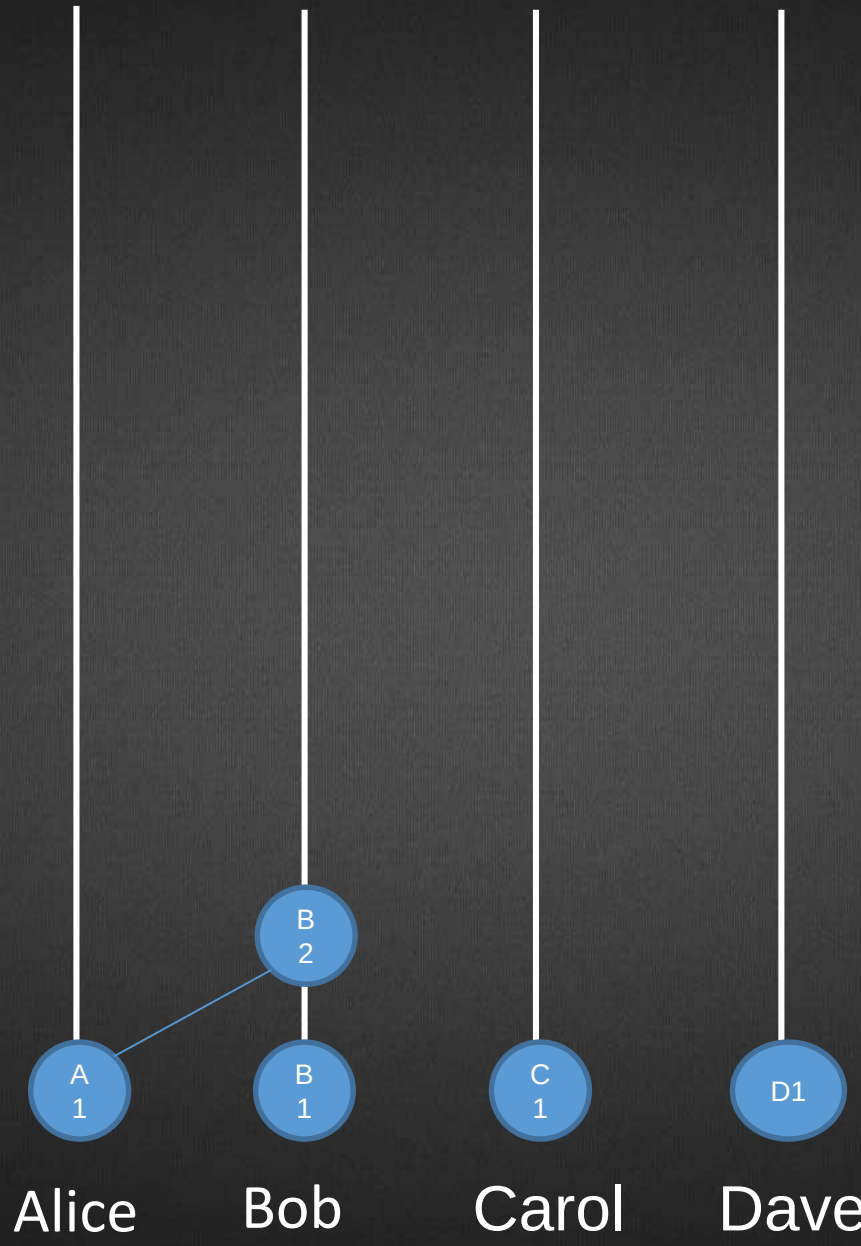
What it gossips about is gossip itself, which contains not only the transactions, but also the route about how transactions are transferred.

Talk about the history how nodes talk to each other, and enable each node to build up a complete graph.

Every node always gossips out its newest event, to tell everything it knows to others.

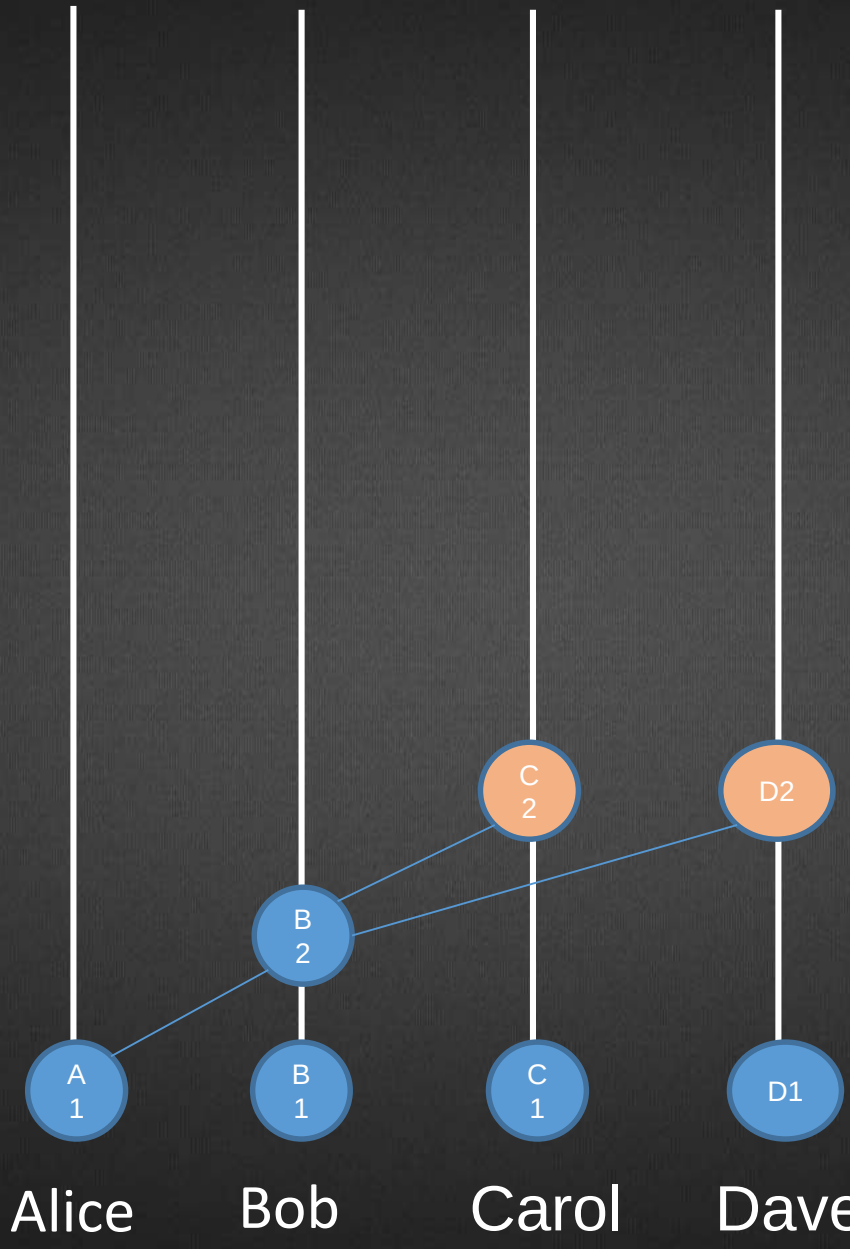
What is virtual-voting?


A
1
B
1
C
1
D1
Alice
Bob
Carol
Dave

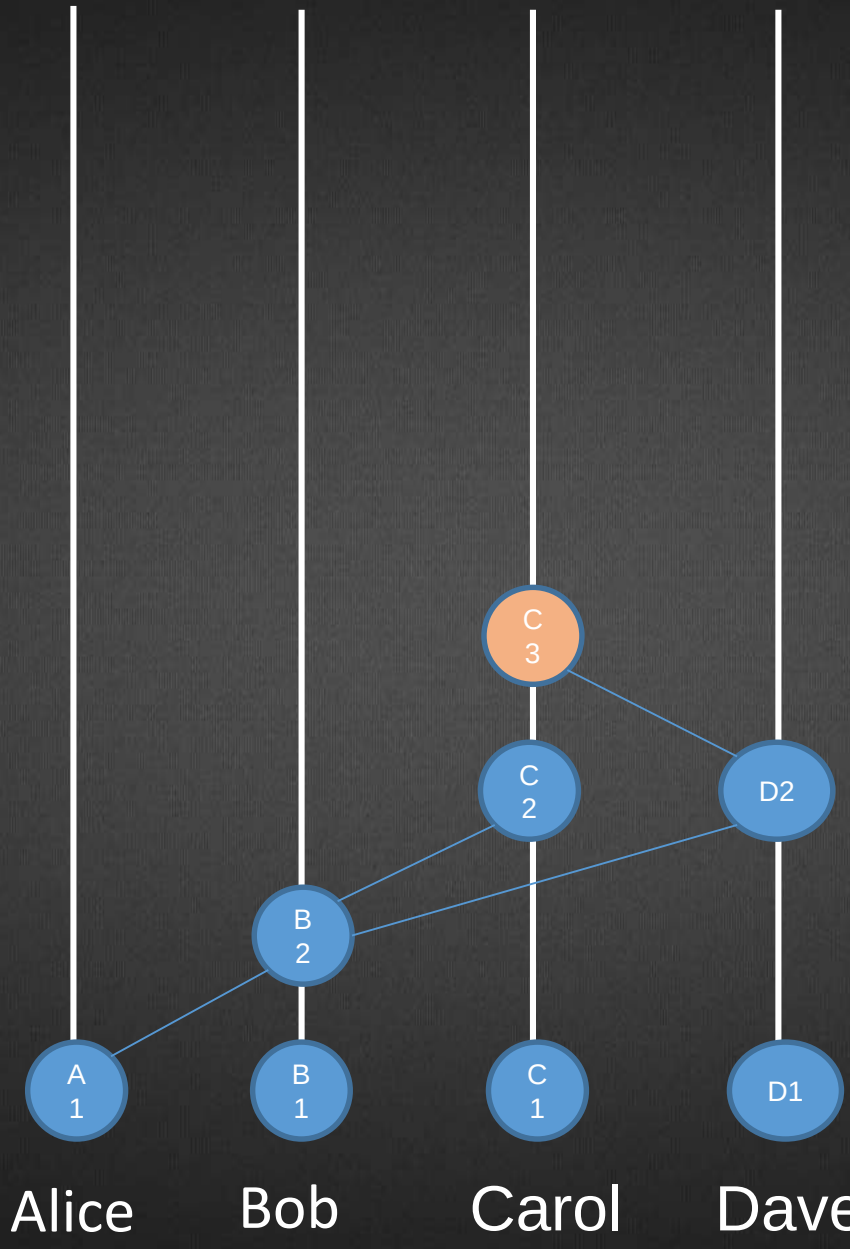


B receives A1 from A, B2 knows everything about A1 and also B1.

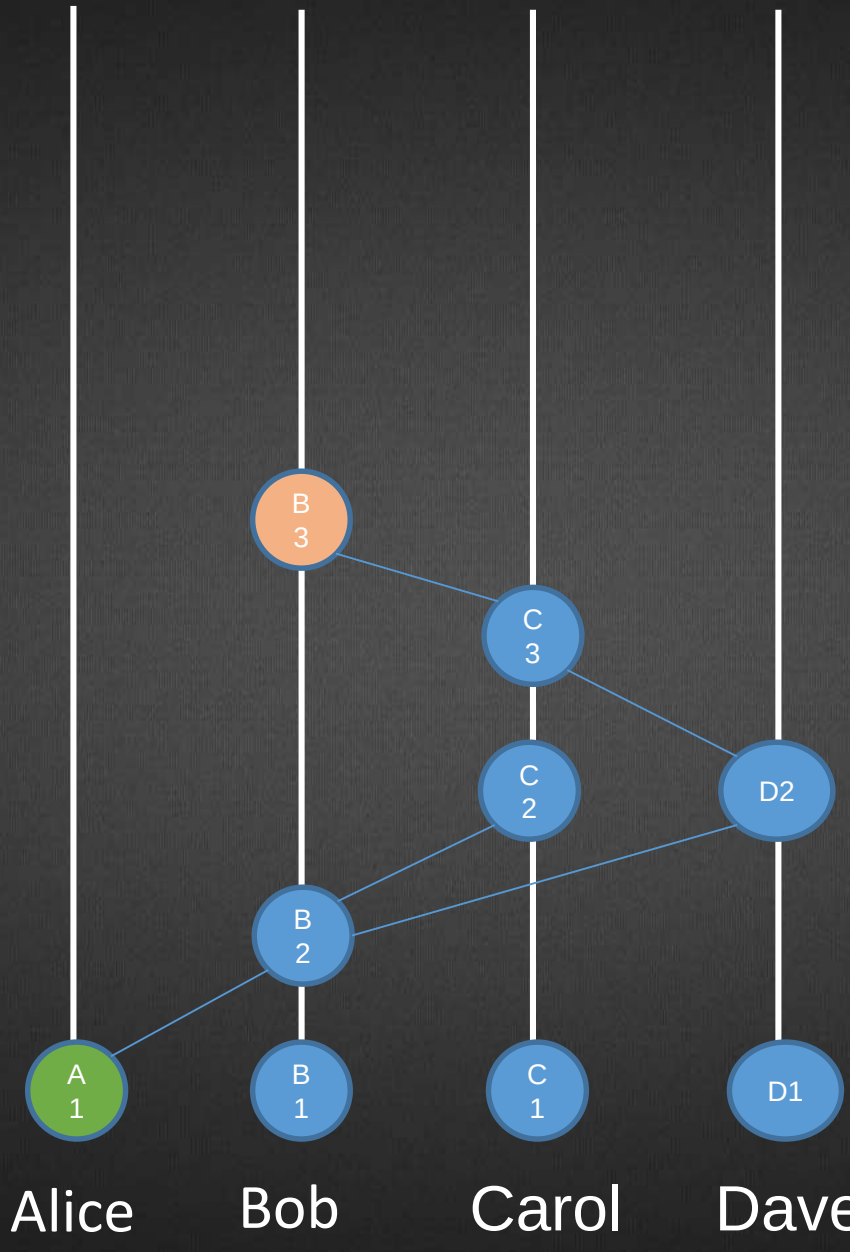
We can think it as a tree and B2 as the root.



Strongly see (virtual vote)

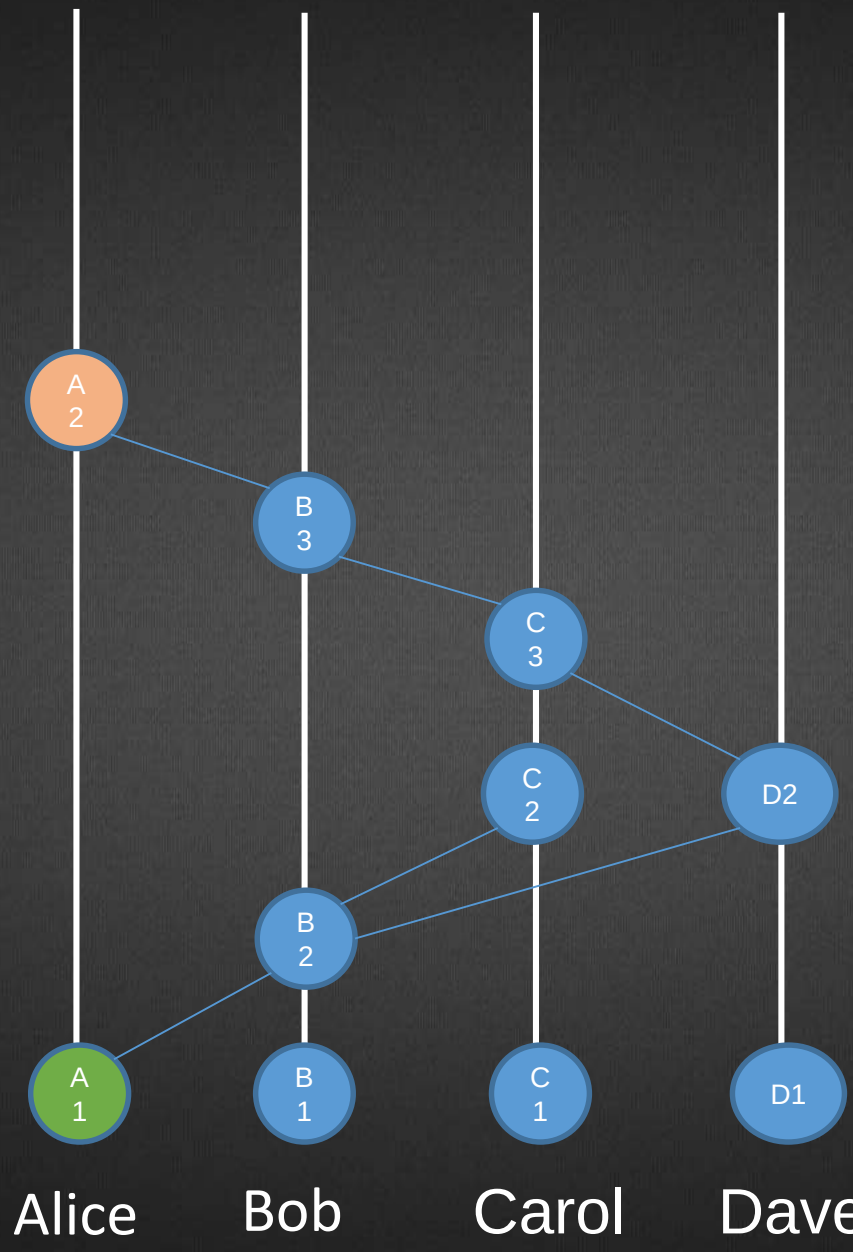


C3 knows everything that C2 and D2 know, so C3 knows that C2 and D2 will vote to A1.

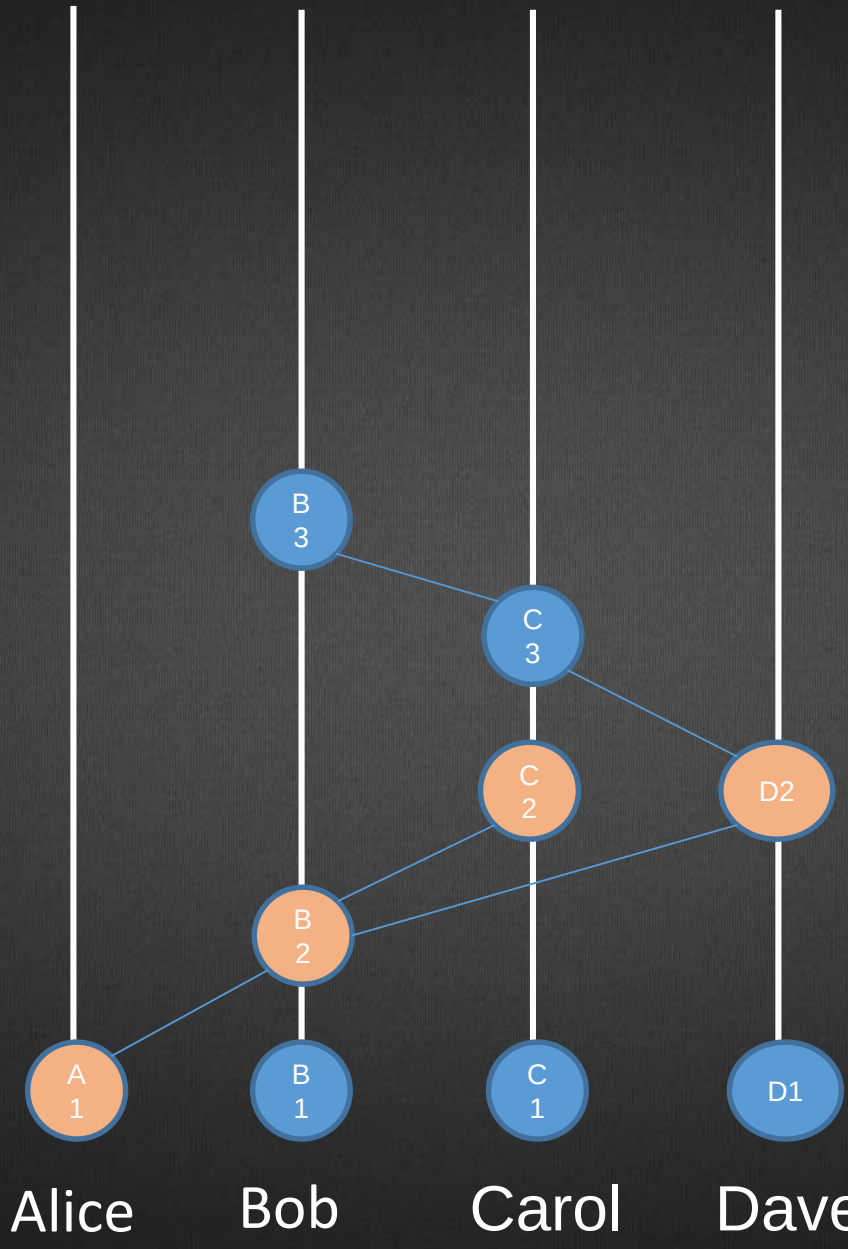


B3 knows that C3 and D2 will vote to A1

3 nodes (more than 2/3 of 4 nodes) Have virtual-voted to A1, it's committed from now on

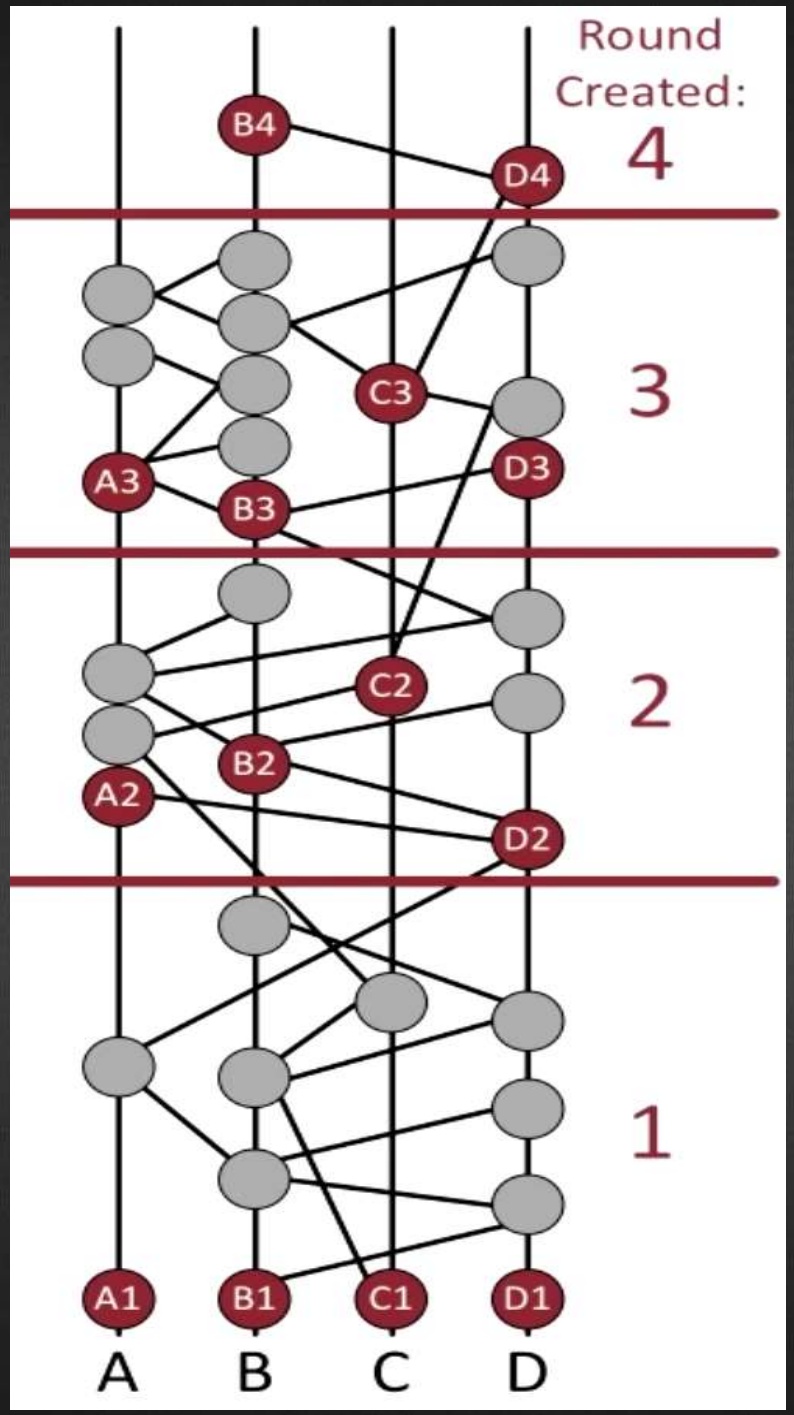


Now A2 knows that B3, C3 and D2 have voted to A1, consensus achieves

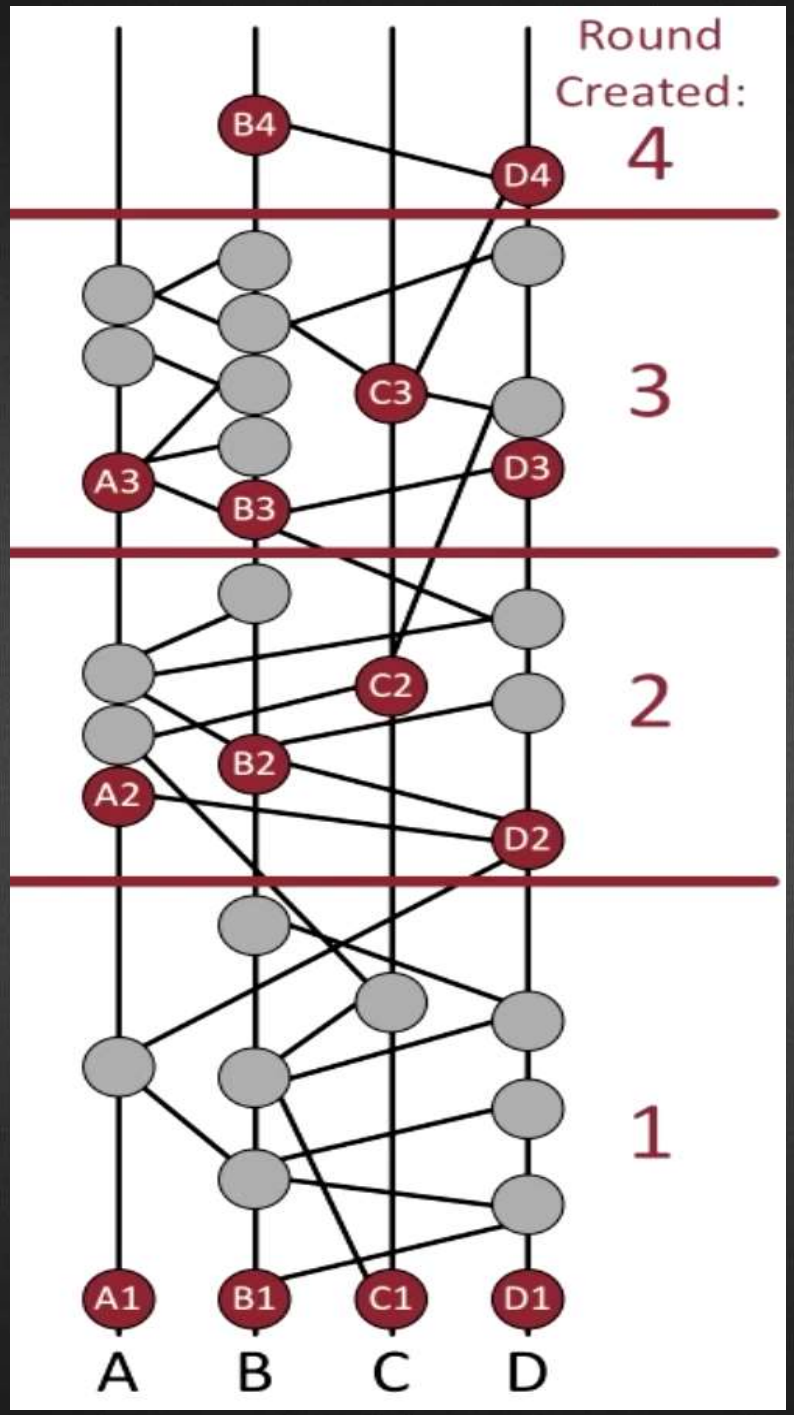


Confirm time is when B3 is created.

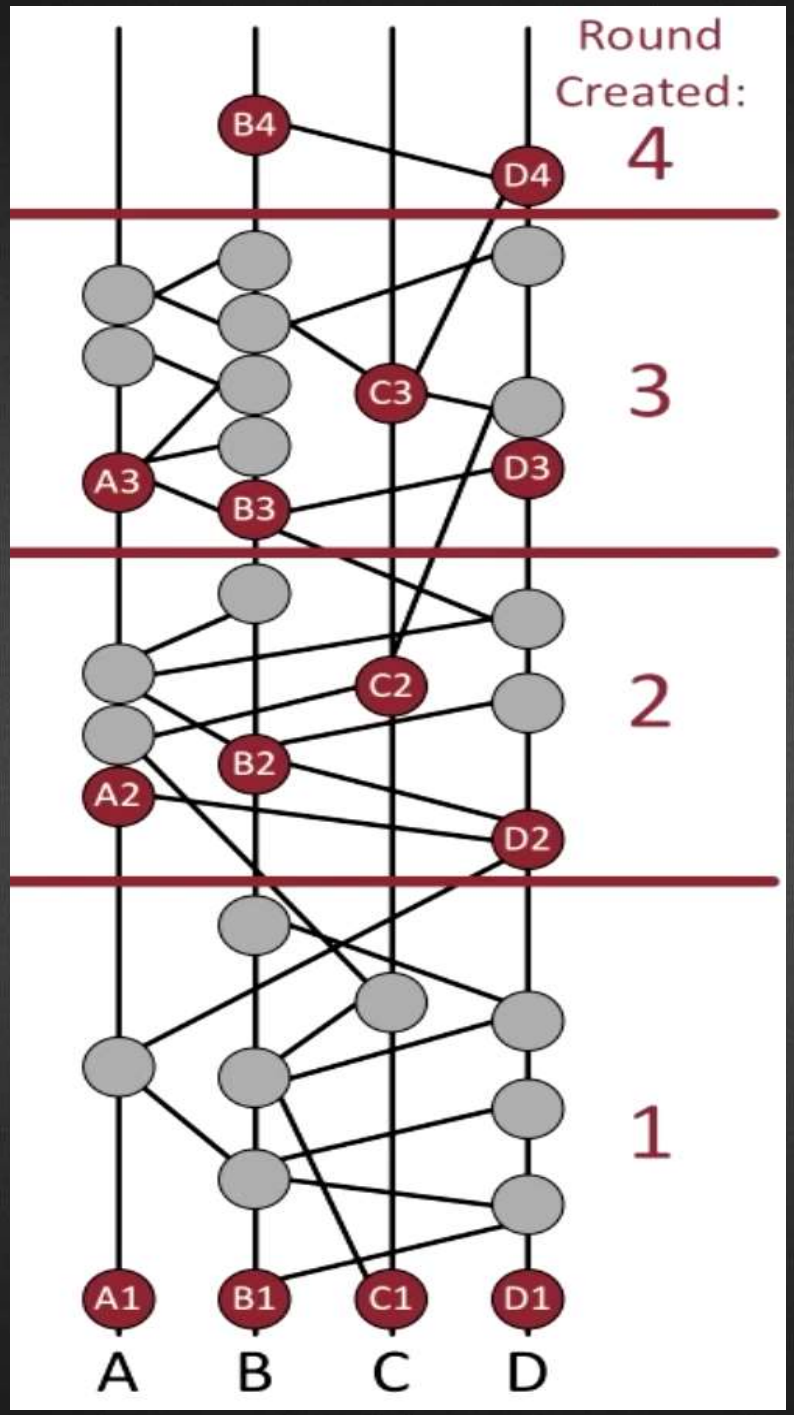
The final timestamp of A1 is the median of A1, B2 and either of (C2 and D2) , and C3 will decide which one received A1 earlier.



However, hashgraph doesn't directly use those kind of concept, but introduces 'round'.
And only the witnesses of next round can virtual-vote to the witnesses of current round.



Trigger : When an event can strongly see the supermajority of witness, it starts a new round.
And here, we say D2 can strongly see B1, C1 and D1.



Why it introduces ‘round’?
To reduce the calculation.

Why it works in asynchronous network?

Because even the messages can be unbounded latency or dropped, the gossip histories keep unchanged.
Sooner or later other nodes will receive it, and it doesn’t require a time limit.

Pros and cons

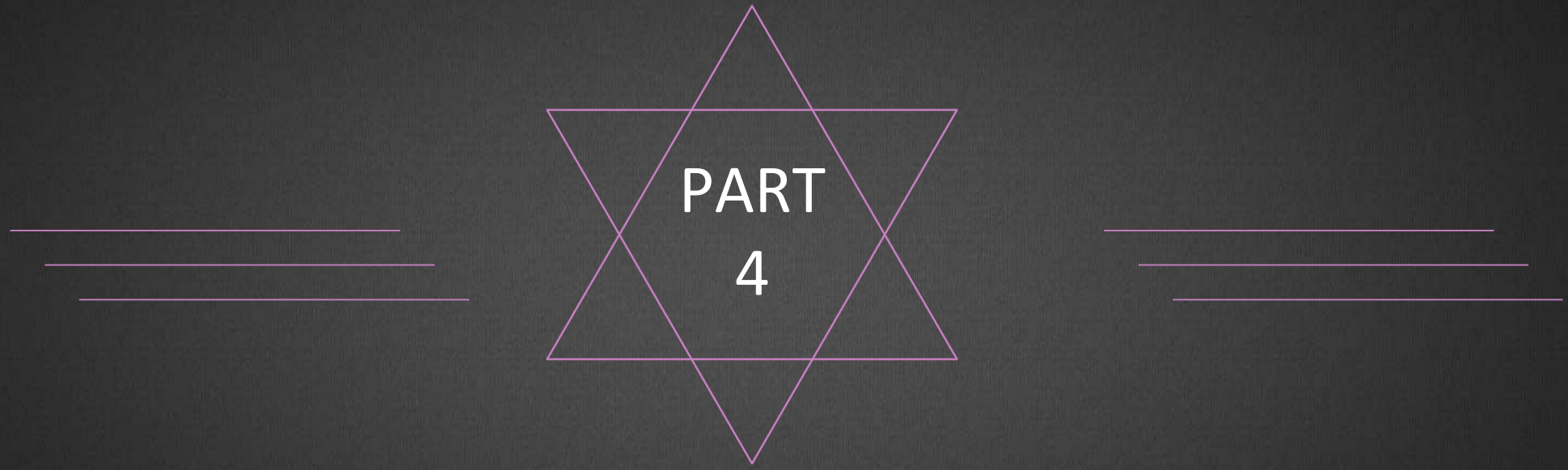
Core Q : comparing with other consensus, what pros and cons hashgraph has?

Pros:

Hashgraph is a math-proved BFT model, which ensures its safety and feasibility.
No needs for voting messages and reception, nearly all the transferring messages are valuable transactions, low time complexity.
fair and robust transaction orders and timestamps, no special nodes, and resist DDoS.

Cons:

The ledger structure isn’t as obvious and easy-understanding as Bitcoin, and it needs to trace back the hashgraph to find the information and transaction orders. (mirror nodes)

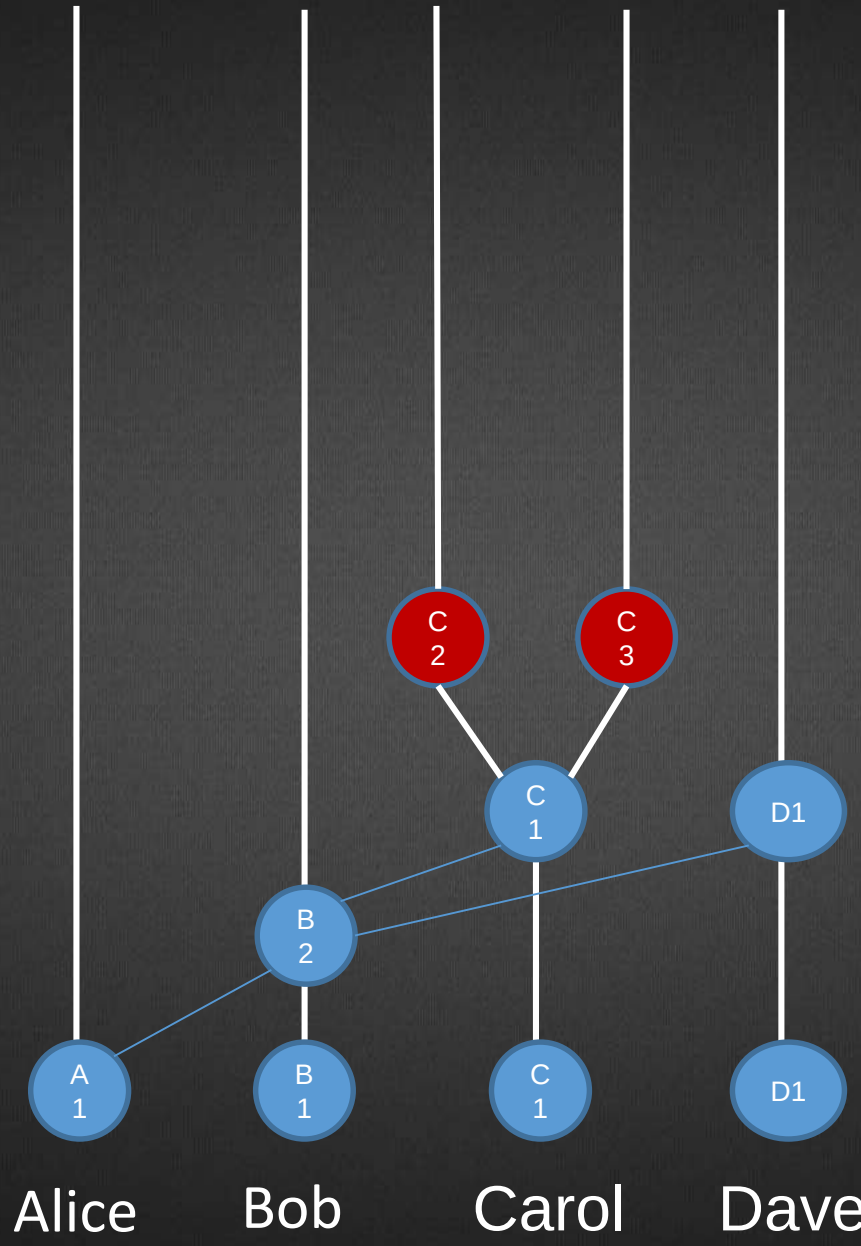


Handle issues

3 example issues

1. fork
2. Double spending
3. Partition

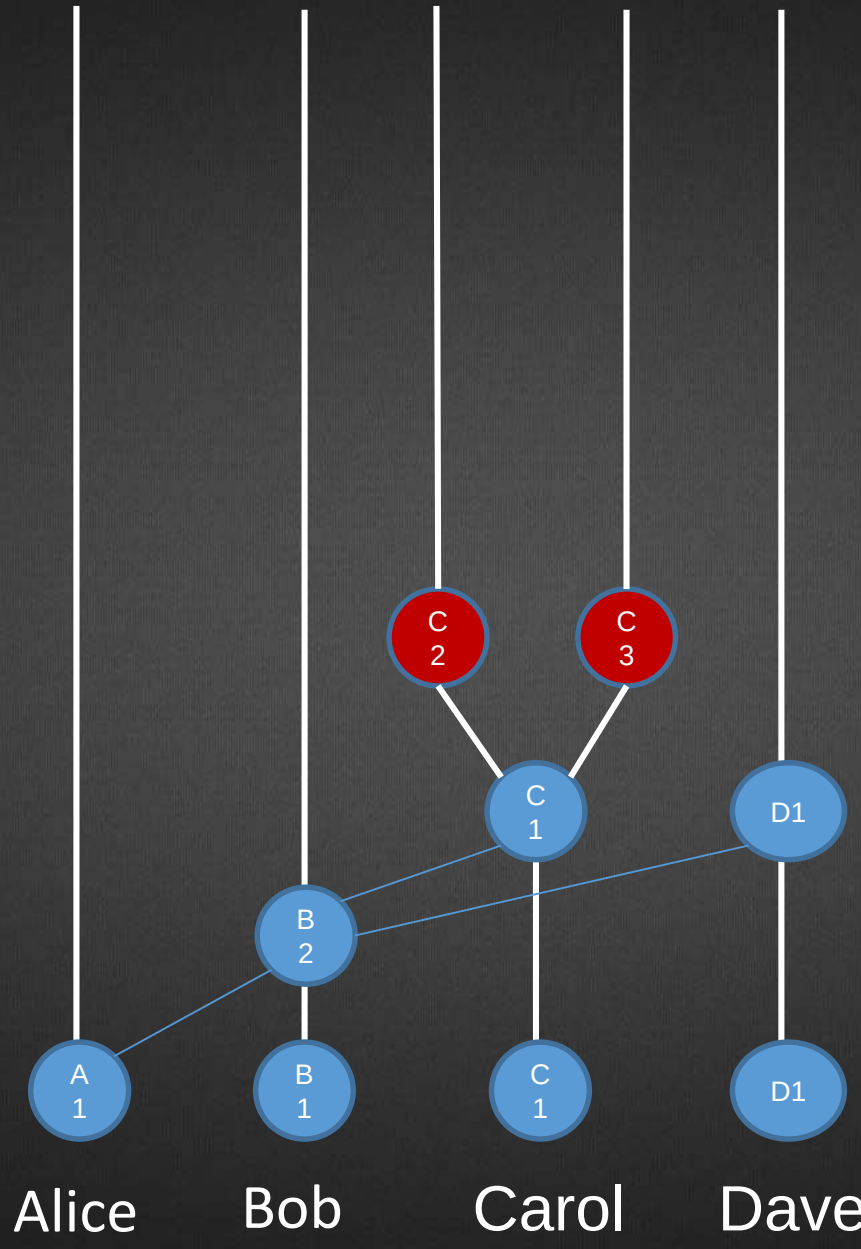
1. fork



If C wants to fork…

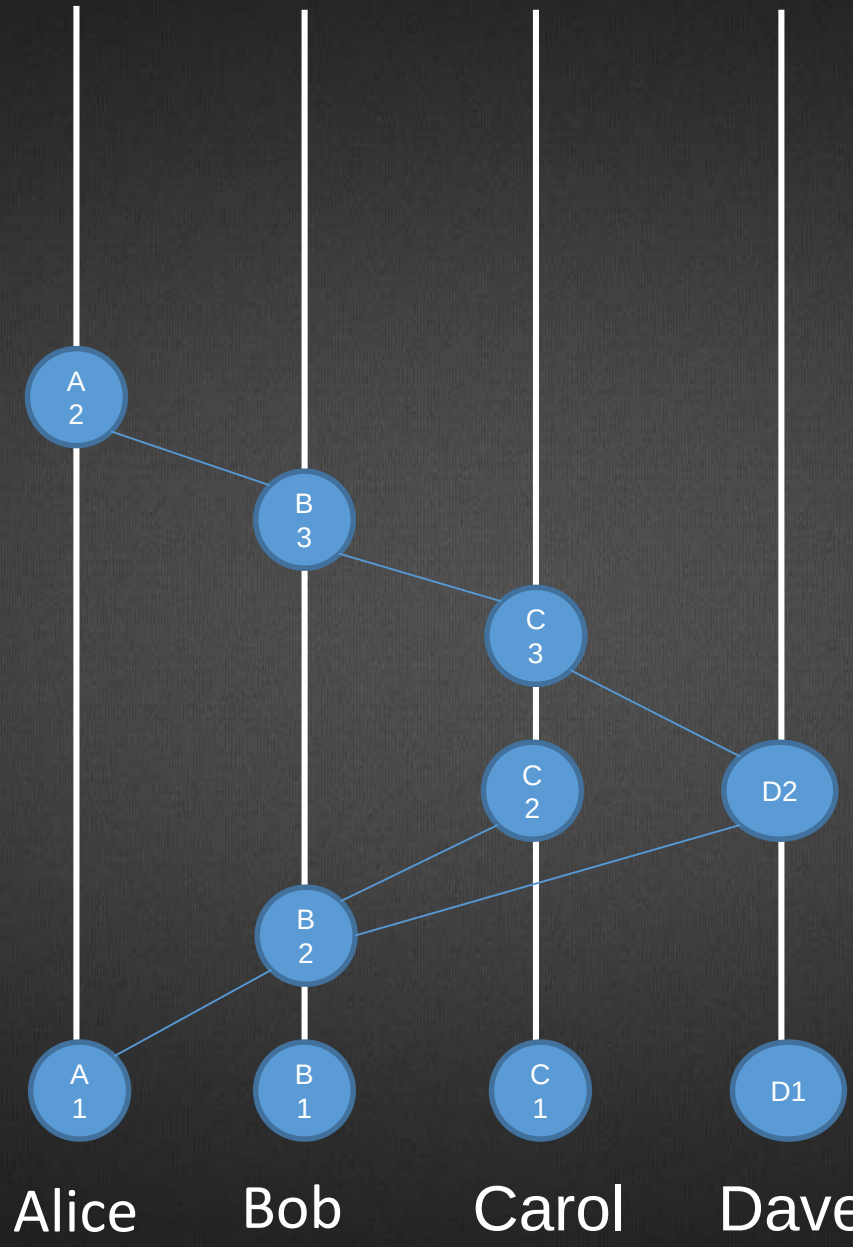
But it will fail to commit because 2 forks can’t be verified by supermajority at the same time

2. Double spending



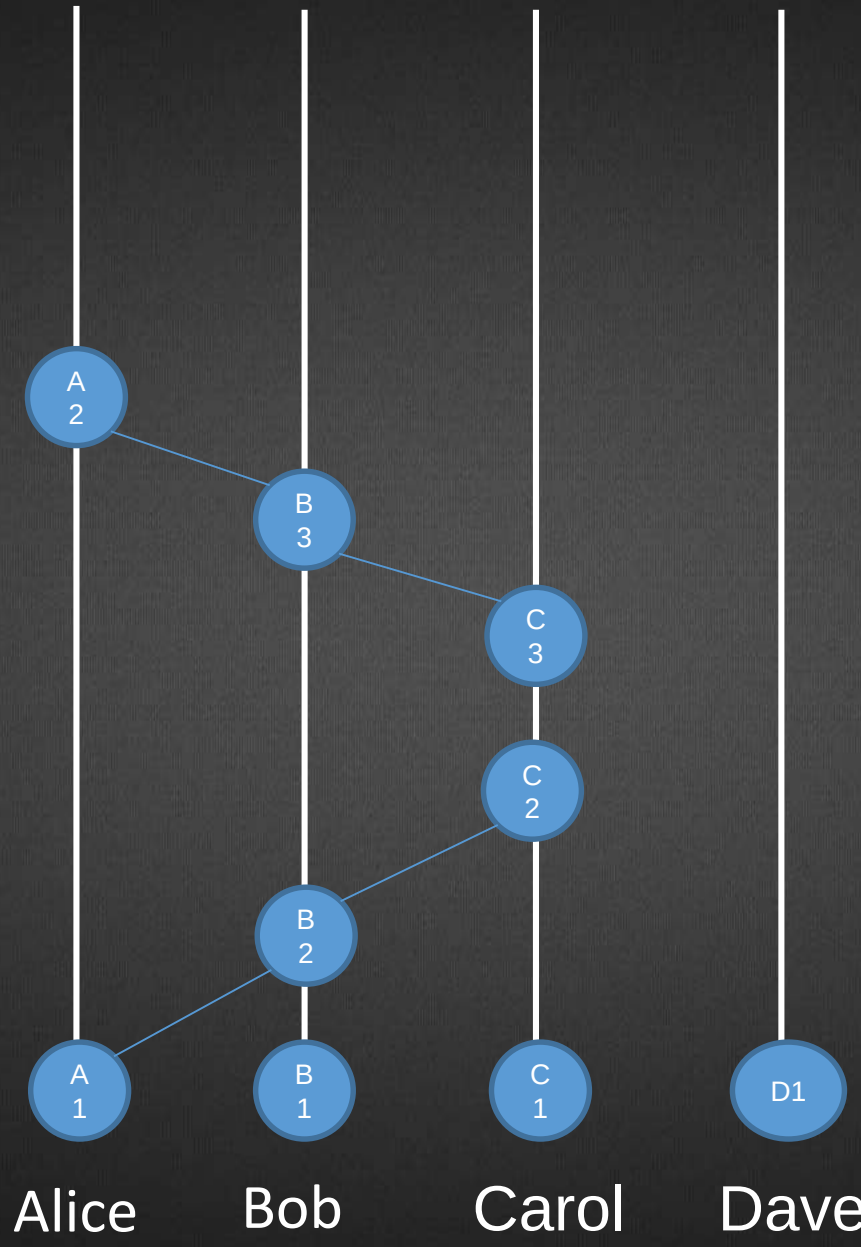
Double spending is the same

3. Partition



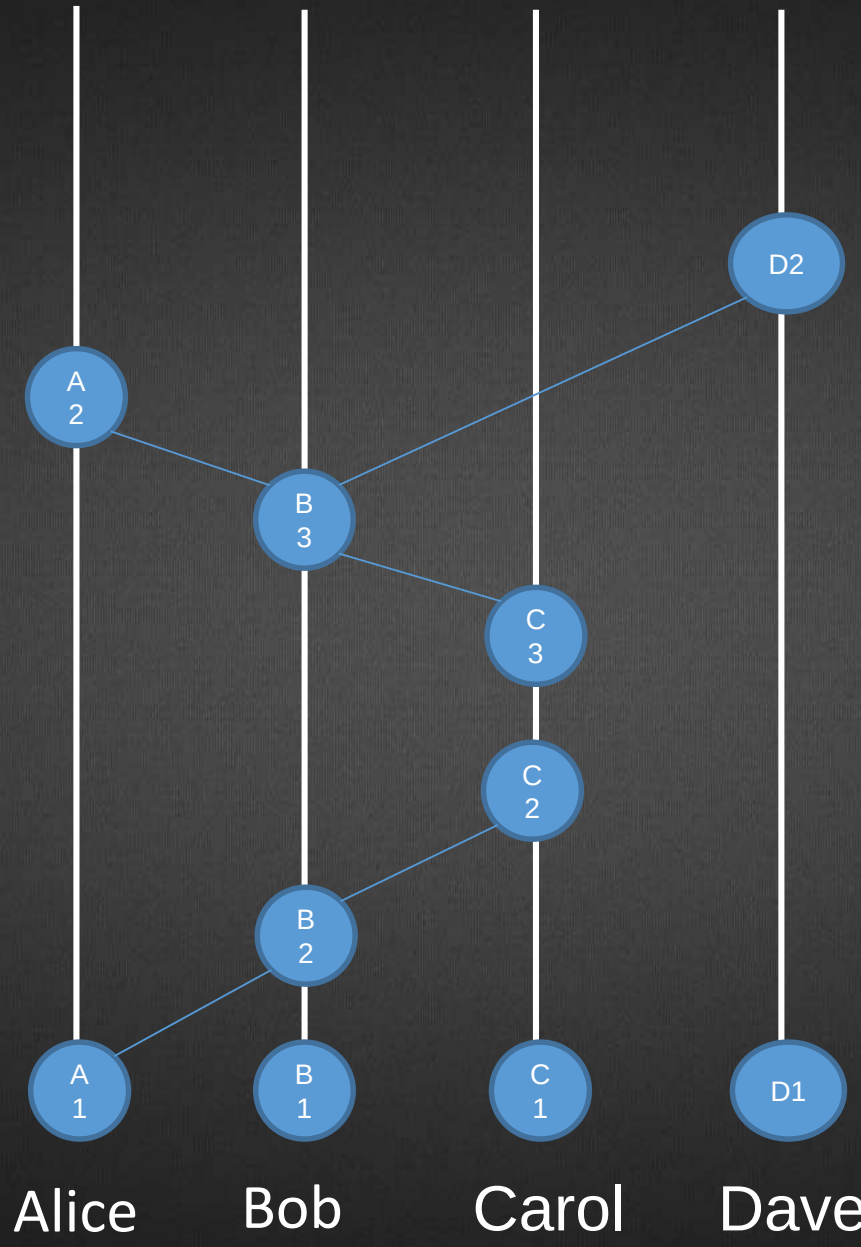
How about D is in partition?

3. Partition

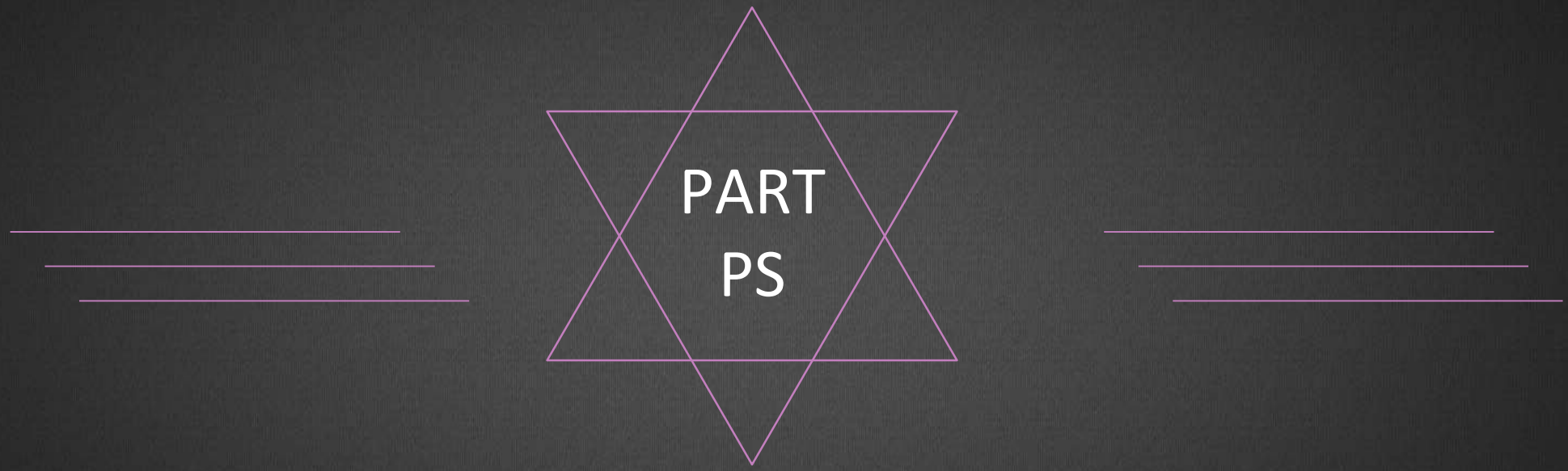


How about D is in partition?

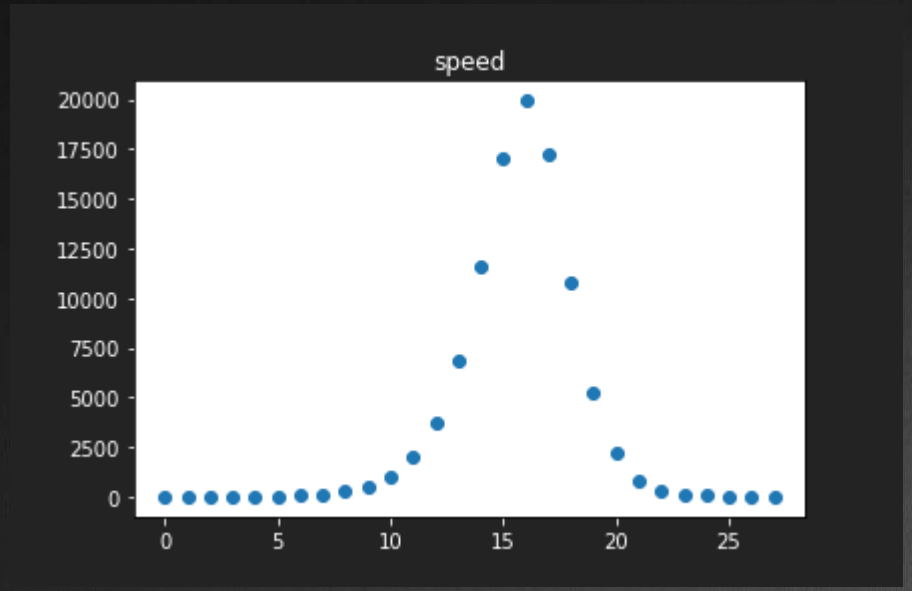
3. Partition

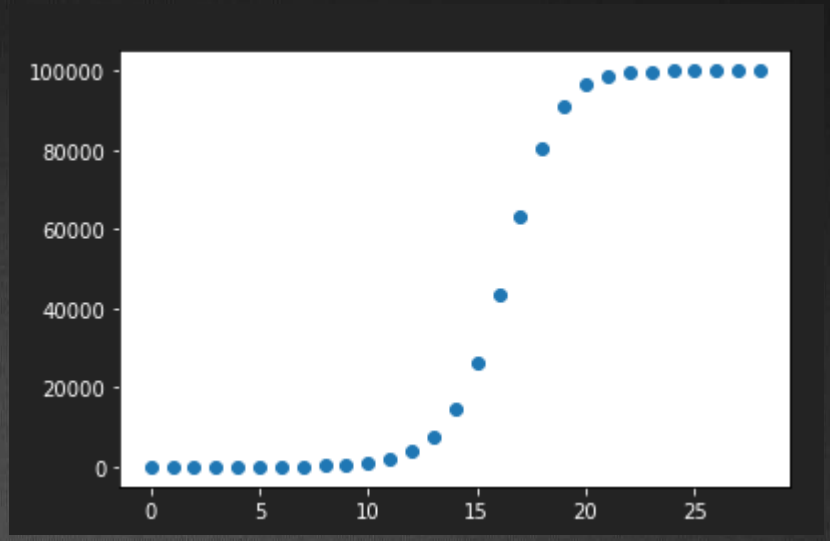


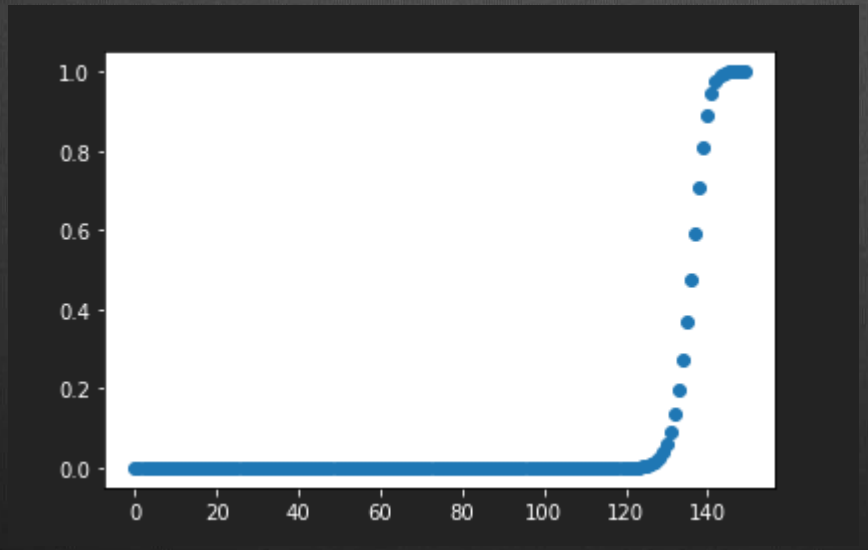
A, B and C will continue to create, gossip and virtual-vote, once D is connected, they can merge easily as every gossip contains the history about how nodes talk to each other.



Performance evaluation


speed
20000
17500
15000
12500
10000
7500
5000
2500
0
0
5
10
15
20
25


100000
80000
60000
40000
20000
0
0
5
10
15
20
25


1.0
0.8
0.6
0.4
0.2
0.0
0
20
40
60
80
100
120
140

THANKS