TreeBlock.io

Version 1.0

Time Sharding Blockchain Architecture

Apply on any existing blockchain & any consensus algorithm.
The easy way to divide workload and network power.

Combining with existing sharding solutions

&

parallely programming to improve the performance

&

the scalability of blockchains

Standardize Tx's Timing

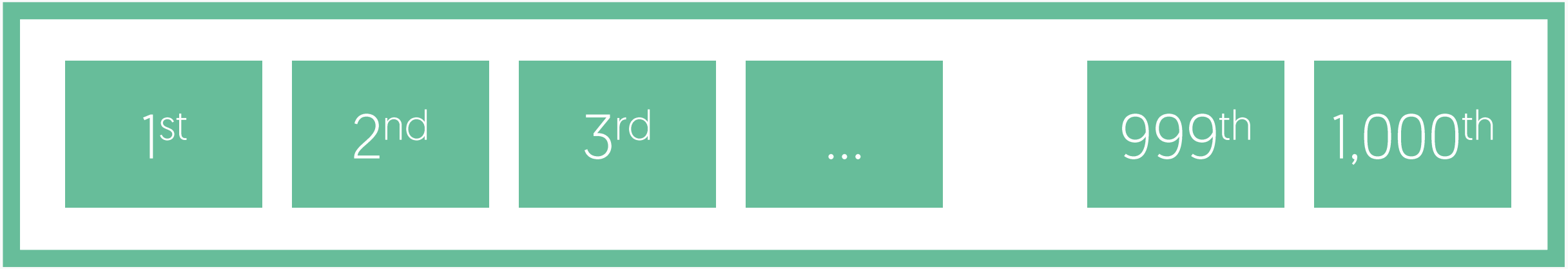
Transaction **sent** at time t:
Tx^t with TTL

Case 1

1 Second with **4 Time Segments**
1 Time Segment = **250,000 mili-sec**

Case 2

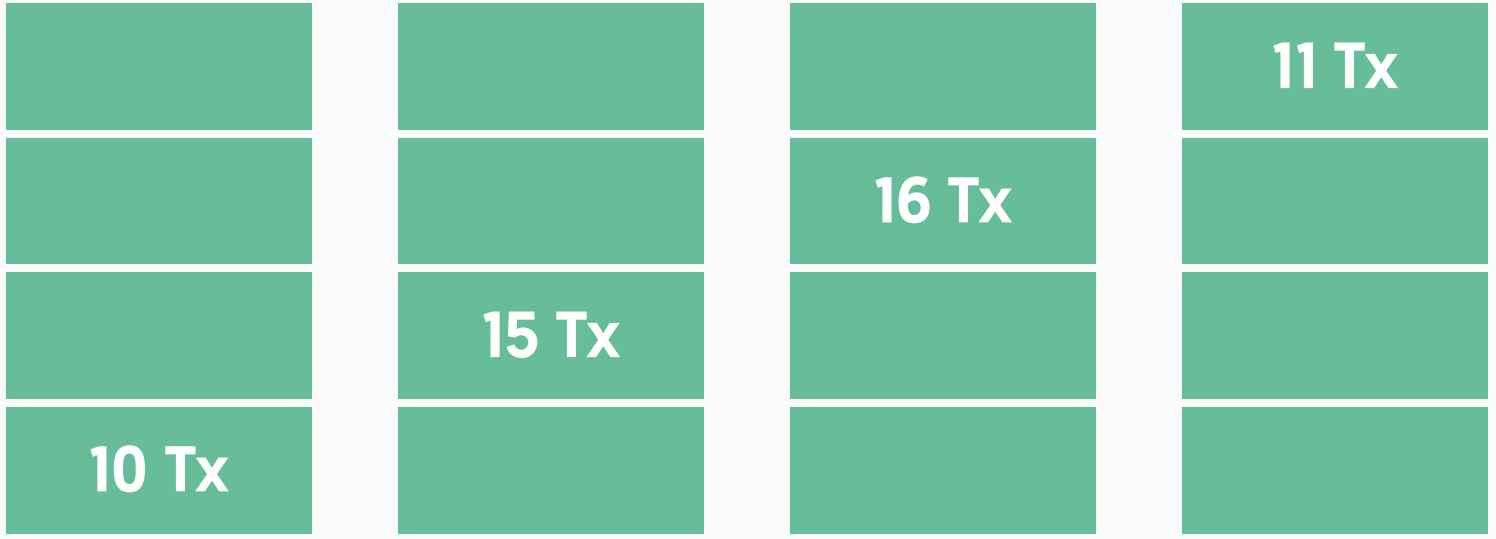
1 Second with **1,000 Time Segments**
1 Time Segment = **1,000 mili-sec**



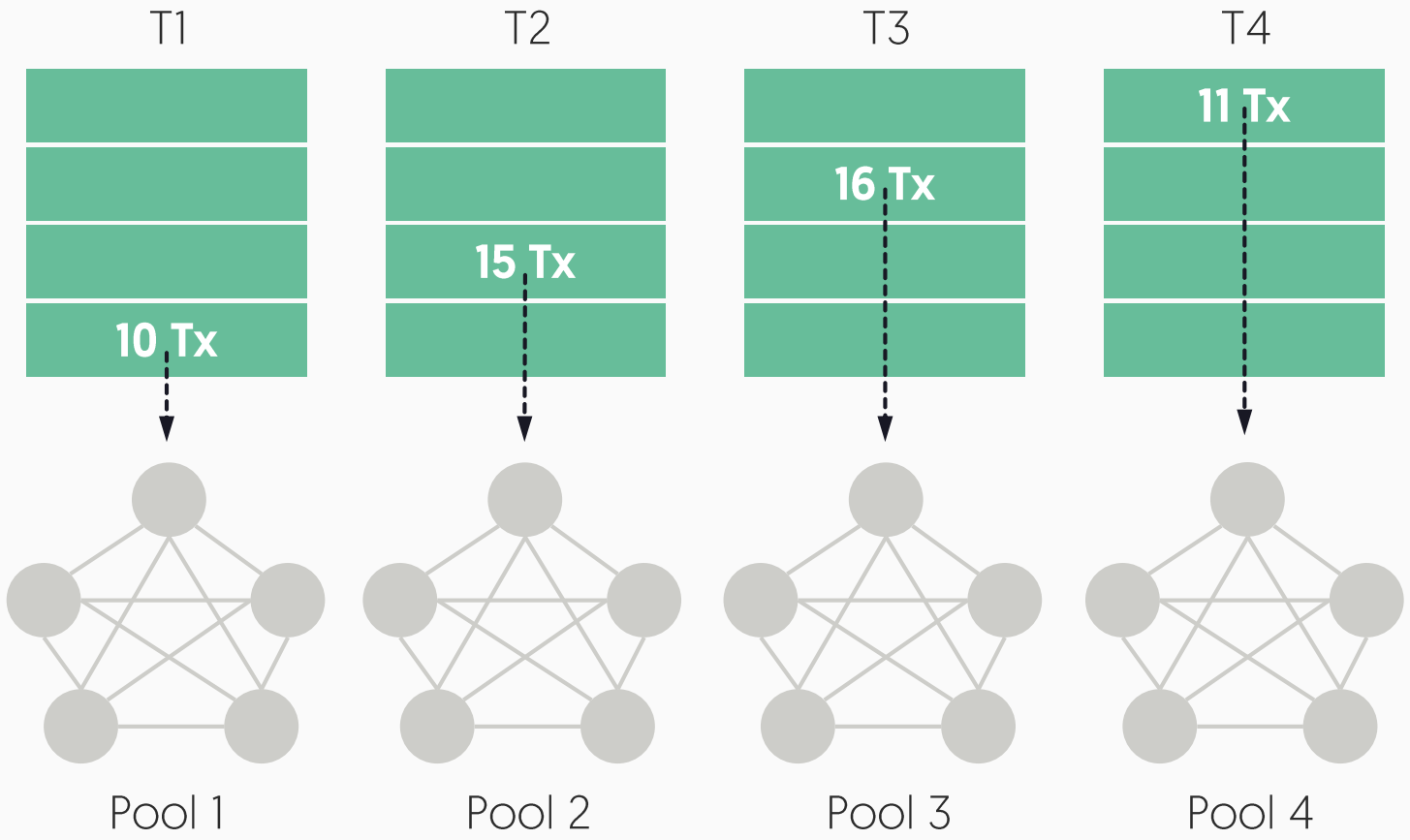
Example

Case 1 with 52 Tx in 1 Sec

Time Segment = 1/4 Sec

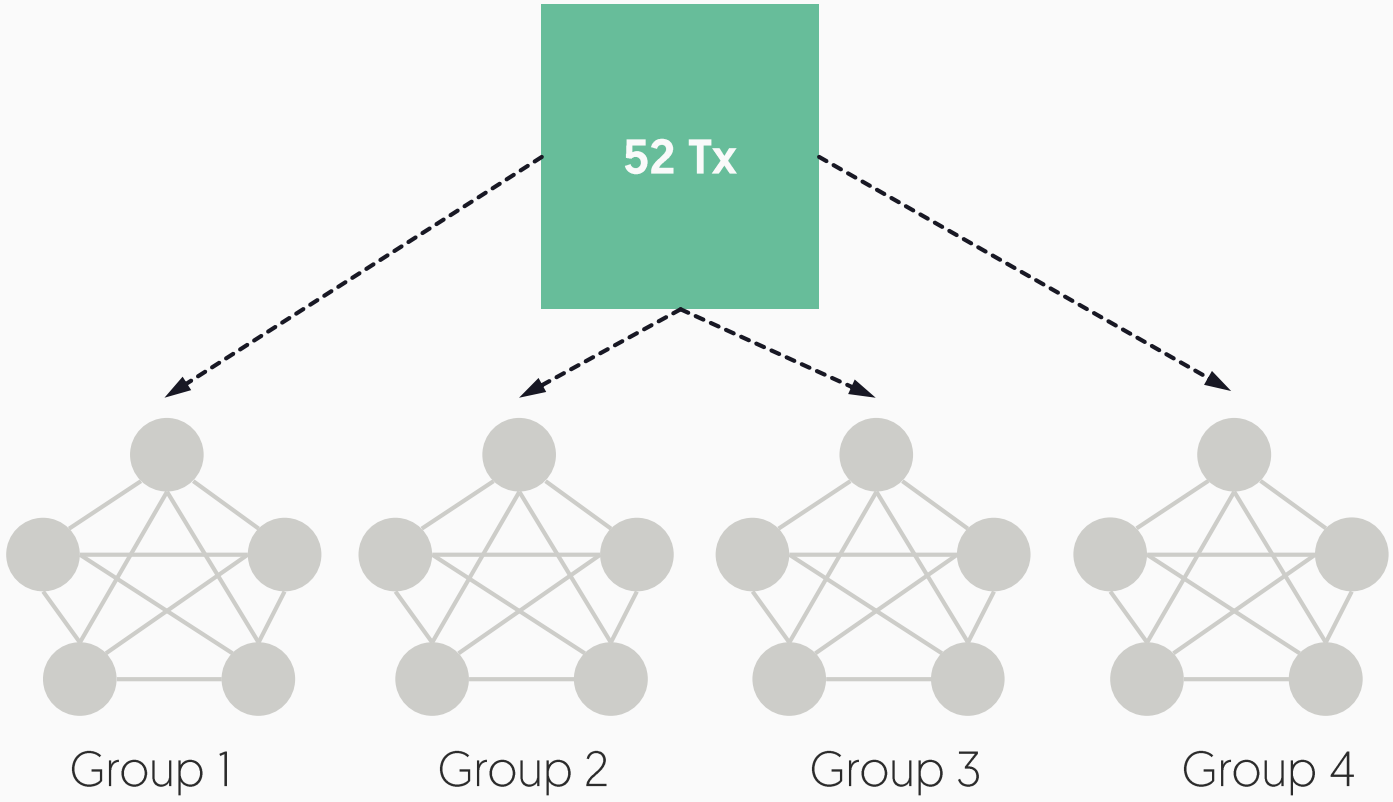


Processing Tx in 1 Second with Time Sharding



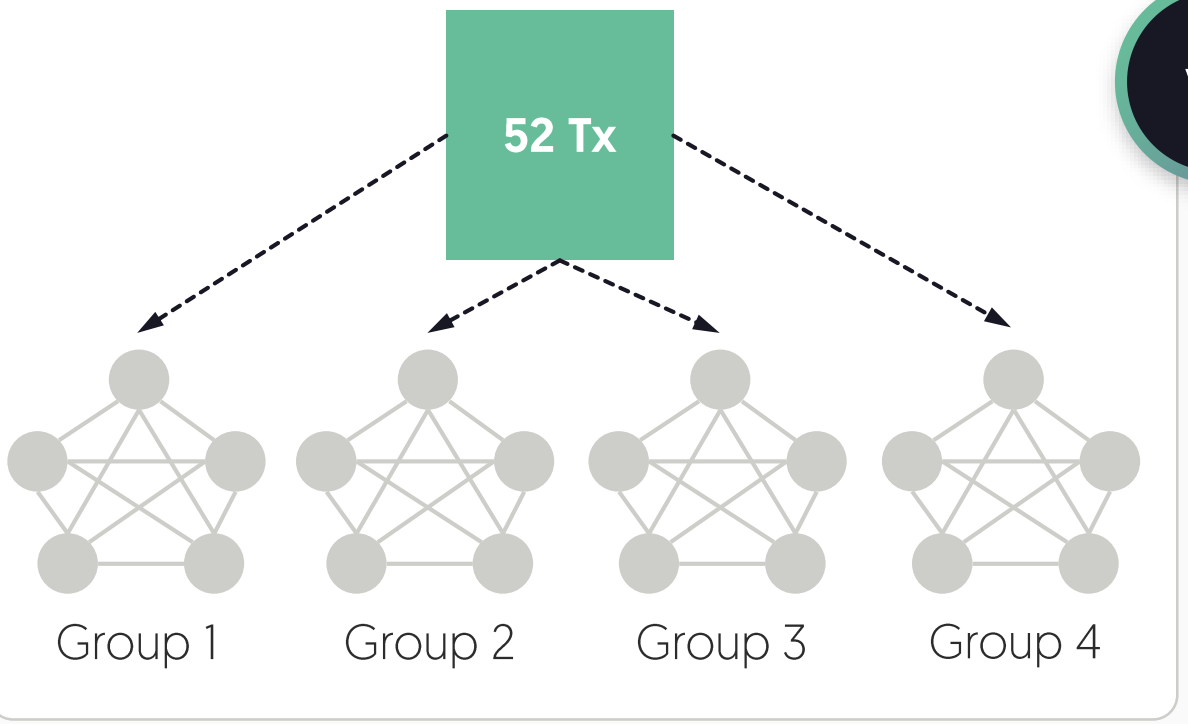
1 Pool = groups of devices

Processing Tx in 1 Second without Time Sharding



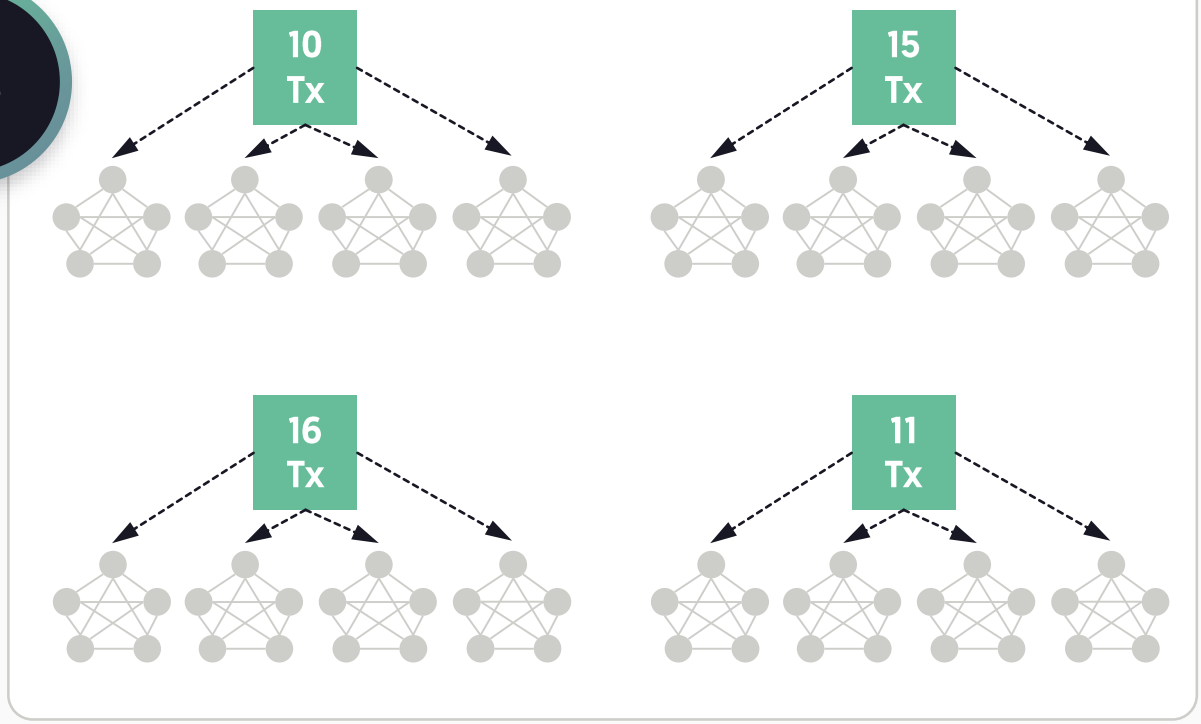
Comparison

Current



VS

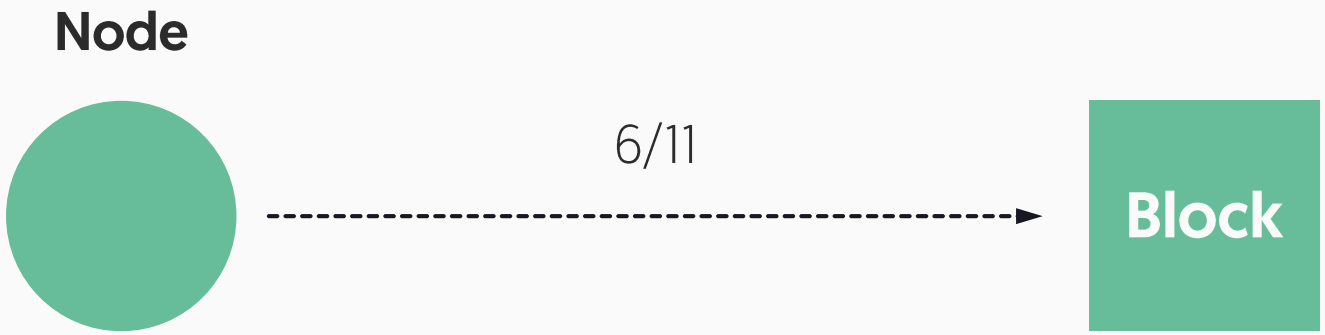
Improvement



Preferred Consensus Algorithm
“DPoS”

One Delegated Node produces Block
at a time slot

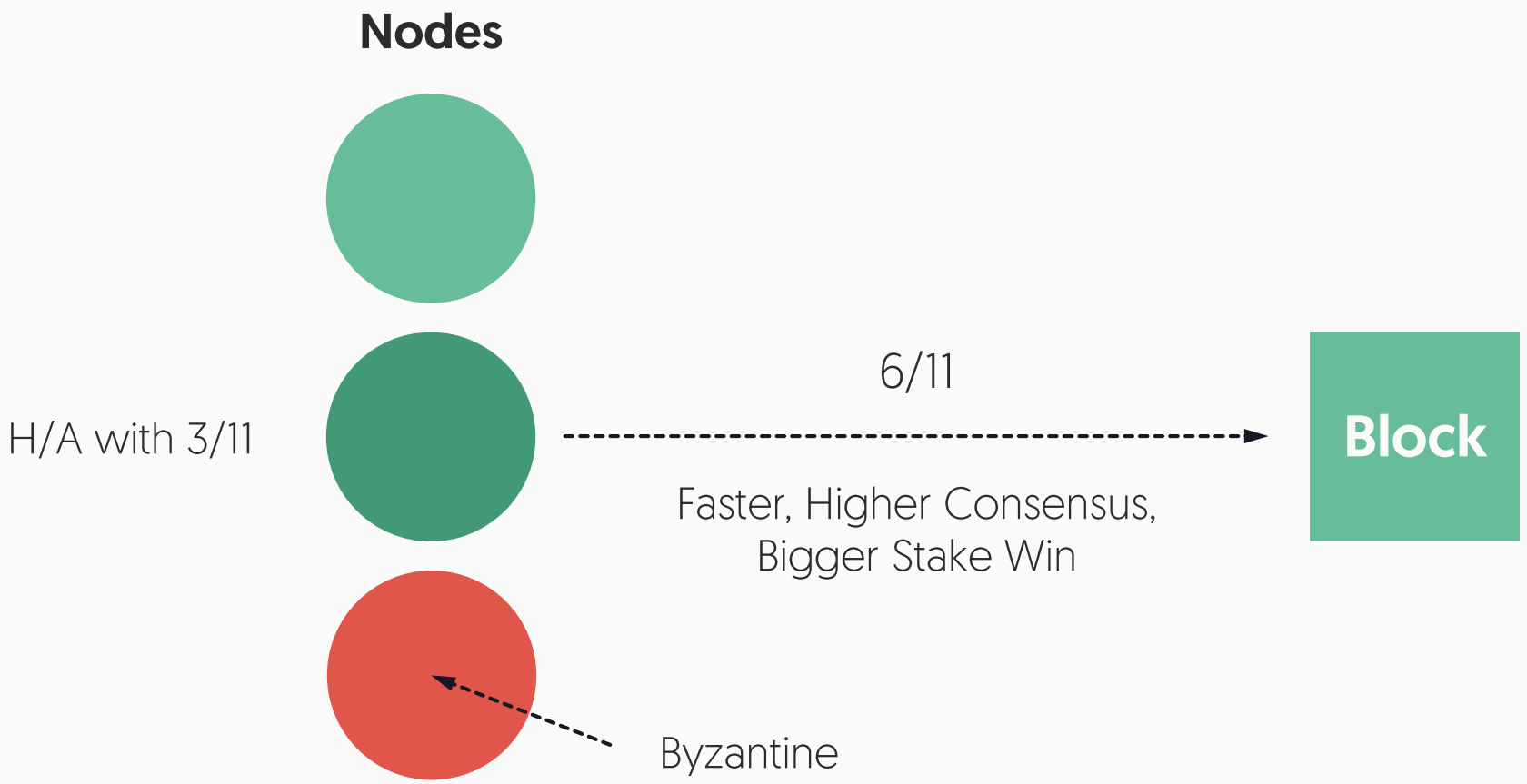
Ex: DPoS = 11



Introduce & Suggest
m-DPoS = multiple DPoS

Many Delegated Nodes compete to produce Block
at a same time slot

Ex: m-DPoS = 3-11


Nodes
H/A with 3/11
6/11
Faster, Higher Consensus,
Bigger Stake Win
Block
Byzantine

Producing Block Paradigm

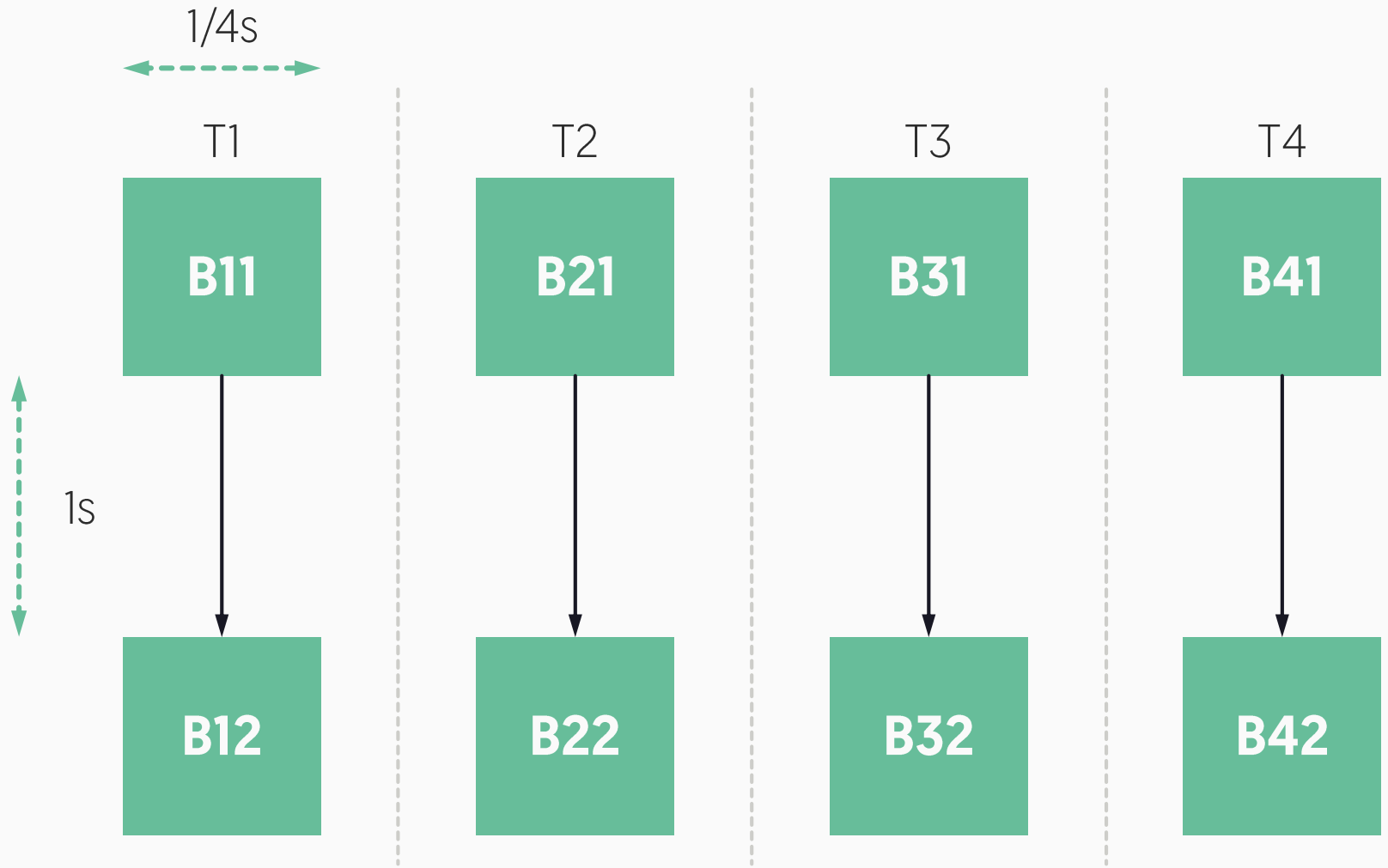
2 ways to produce block with this architecture

Parallelly vs Sequentially

Parallelly Producing Block

Example: **Every 1 Second**

Paralelly Abstract Time-Chains



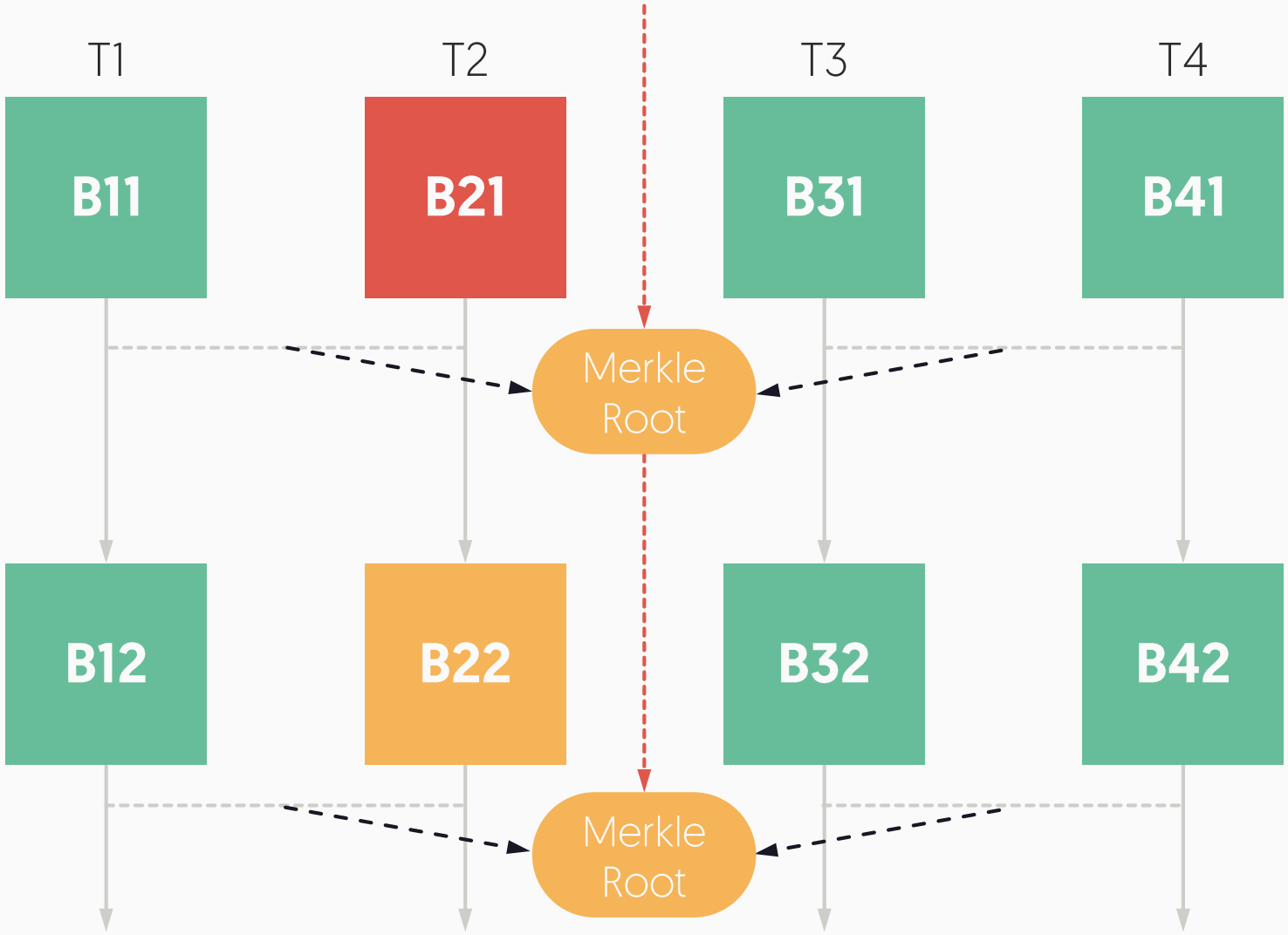
Producing Merkle Root with m-DPoS cross Time-Chains

"m" Delegated Nodes of Time-Chains will cross check

&

compete to produce Merkle Root of previous Blocks

Block Attack on Parallel Time-chains



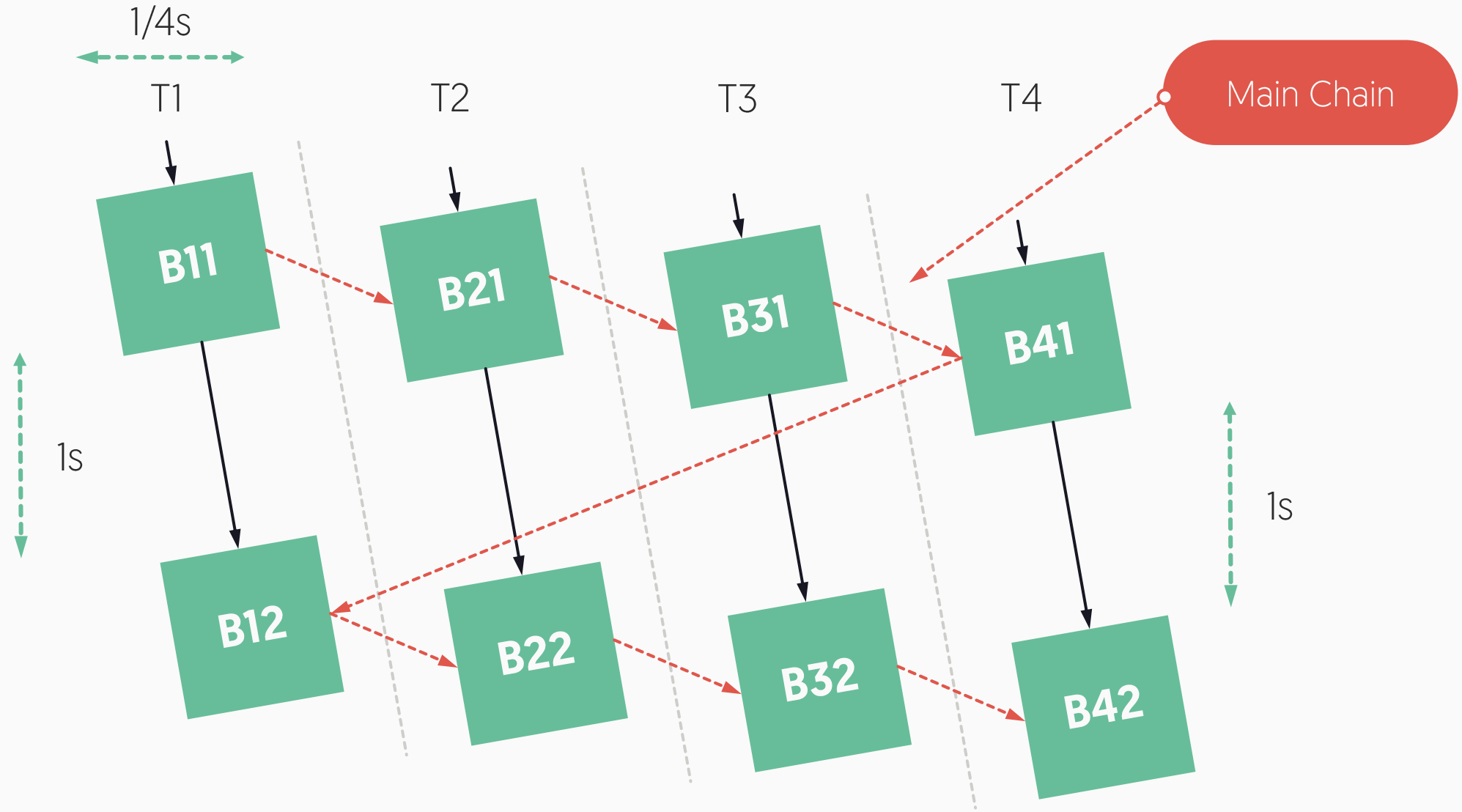
Sequentially Producing Block

Every 1 Second per **Time-chain**

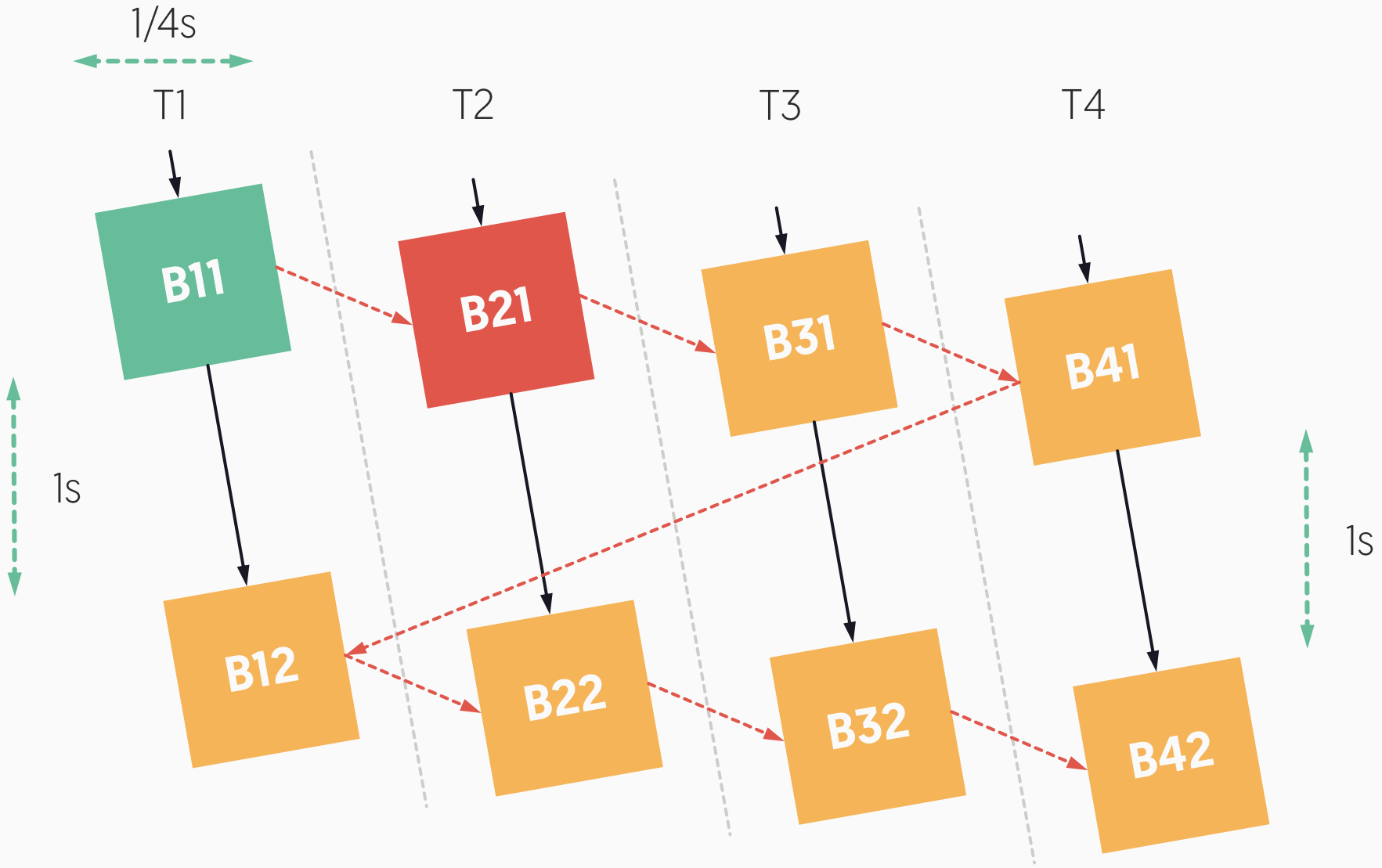
or

Every Time Step based on **Time Segment** overall

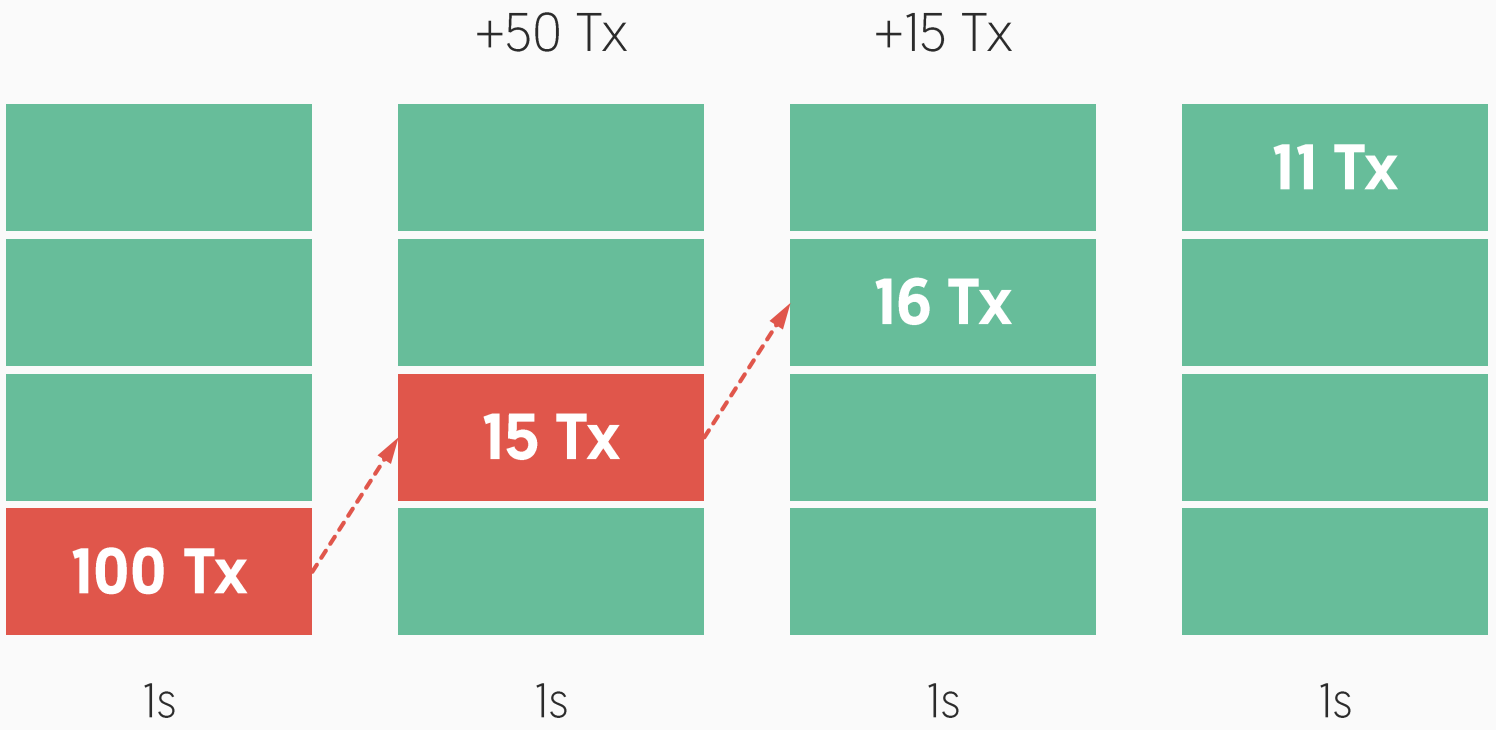
Sequential Abstract Time-Chains



Block Attack on Sequential Time-Chains



Case: Overload & Load Balance
Ex: Maximum 50 Tps per Time-Chain



Shift and Process overloaded Tx on other Time-Chains after cross-check

Parallelly

1. Heavy traffic when parallelly cross-check between Time-Chains per cycle
2. Better if extend block producing time, 2 s or more
3. Better with high internet speed and Node power
4. Scale unlimited with smaller Time Segments
5. Main-chain: Complicated

vs

Sequentially

1. Sacrifice some times to check cross Time-Chains per cycle
2. Able to work with shorter cycle for block producing
3. Better with high internet speed and Node power
4. Scale unlimited with smaller Time Segments
5. Main-chain: Zigzag

YouRoot.io

Open Discussion

Anh Le
Founder & CTO
anh@youroot.io