# Stellar Transactions

Saravanan Vijayakumaran
sarva@ee.iitb.ac.in

Department of Electrical Engineering
Indian Institute of Technology Bombay

September 26, 2019

# Stellar Transactions

- Commands to modify ledger state
- Contain upto 100 operations
- List of possible operations
  - Create Account
  - Payment
  - Path Payment
  - Manage Offer
  - Create Passive Offer
  - Set Options
  - Change Trust
  - Allow Trust
  - Account Merge
  - Inflation
  - Manage Data
  - Bump Sequence
- Transaction fees = Number of operations $\times$ Base fee
  - Current base fee = 100 stroops = $10^{-5}$ XLM
- Fees added to fee pool and distributed via inflation voting

# Transaction Fields and Sets

- Transaction fields
  - **Source account** = Account ID transaction source
  - **Fee** = Transaction fees
  - **Sequence number**: Must be 1 greater than source account sequence number
  - **List of operations**
  - **List of signatures**: Upto 20 signatures can be included
  - **Memo** = Optional data field (upto 32 bytes long)
  - **Time bounds** = Optional lower and upper UNIX times specifying transaction validity
- Transaction sets
  - Collections of transactions proposed for inclusion in next ledger closing
  - Stellar consensus protocol (SCP) is used to achieve consensus
  - The transaction set picked by SCP is applied to current ledger state

# Inflation

- New lumens added to the network at the rate of 1% per year
- Each week these lumens are distributed via the Inflation operation
- Distribution algorithm
  1. Calculate inflation pool as

$$\text{Total lumens in existence} \times \text{Weekly inflation rate} + \text{Fee pool}$$

  2. Calculate vote threshold as

$$\text{Total lumens in existence} \times 0.0005$$

  3. Determine the accounts which receive more votes than the threshold
  4. Allocate lumens to winners proportional to the votes they received
  5. Return unallocated lumens to the fee pool

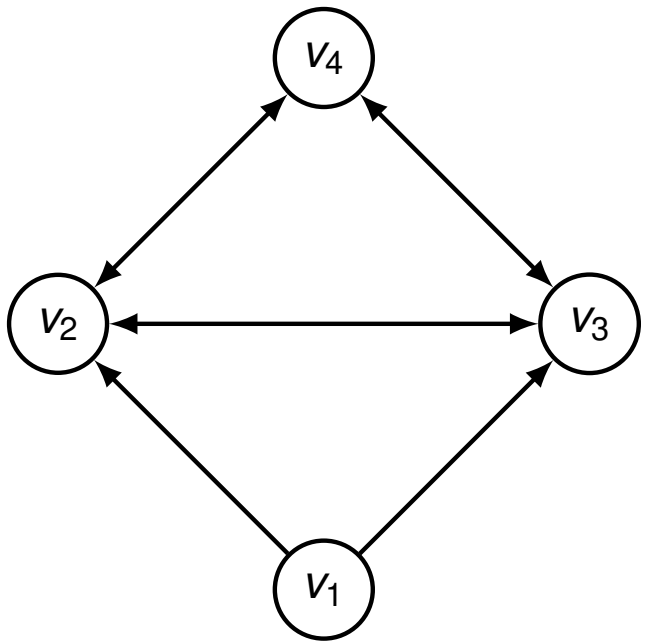
# Multisignature in Stellar

- Each account specifies upto 20 signers (in addition to owner)
  - Each signer is a public key and a weight
  - Account owner also has a weight called master key weight
  - Example: Master key weight = 1, Alice's key weight = 1, Bob's key weight = 1
- Thresholds for account operations
  - Operations have three possible categories: low, medium, high
    - **Low security**: Inflation, Allow Trust, Bump Sequence
    - **High security**: Updating signers and thresholds, Account Merge
    - **Medium security**: Payment, Create Account, Everything else
  - Thresholds for each category are an integer from 0 to 255
  - Example: low thres = 1, medium thres = 1, high thres = 3
- For each operation, sum of weights of signatories should exceed threshold
- Anchor setup example
  - Master key weight = 2, Additional key weight = 1
  - low thres = 0, medium thres = 2, high thres = 2
  - Master key is kept offline and additional key is kept online

# Stellar Consensus Protocol

# Federated Byzantine Agreement

- **Definition**: An **federated Byzantine agreement system (FBAS)** is a pair $\langle \mathbf{V}, \mathbf{Q} \rangle$ comprising of a set of nodes $\mathbf{V}$ and a quorum function $\mathbf{Q} : \mathbf{V} \mapsto 2^{2^{\mathbf{V}}} \setminus \{\emptyset\}$ specifying one or more quorum slices for each node, where a node belongs to all of its own quorum slices, i.e. $\forall v \in \mathbf{V}, \forall q \in \mathbf{Q}(v), v \in q$.
- Example

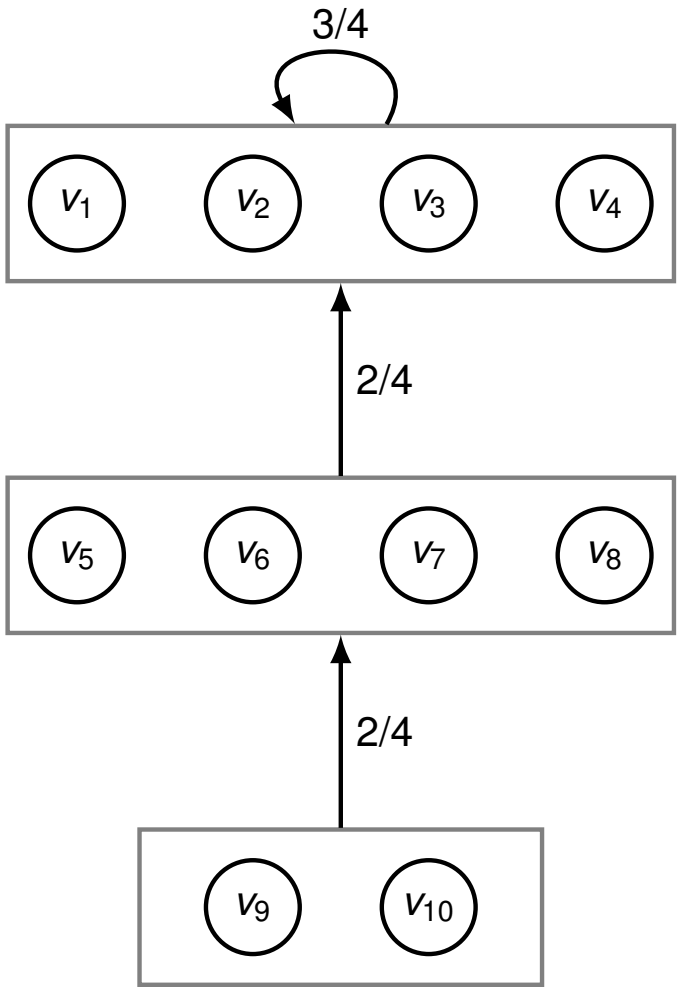



$$\mathbf{Q}(v_1) = \{\{v_1, v_2, v_3\}\}$$

$$\mathbf{Q}(v_2) = \mathbf{Q}(v_3) = \mathbf{Q}(v_4) = \{\{v_2, v_3, v_4\}\}$$

- **Definition**: A set of nodes $\mathbf{U} \subseteq \mathbf{V}$ in FBAS $\langle \mathbf{V}, \mathbf{Q} \rangle$ is a **quorum** iff $\mathbf{U} \neq \emptyset$ and $\mathbf{U}$ contains a slice for each member, i.e. $\forall v \in \mathbf{U}, \exists q \in \mathbf{Q}(v)$ such that $q \subseteq \mathbf{U}$.
- A quorum of nodes is sufficient to reach agreement

# Tiered FBAS Example




Slices are self + any other two out of $\{v_1, v_2, v_3, v_4\}$

Slices are self + any two top tier nodes

Slices are self + any two middle tier nodes

Possible quorums?

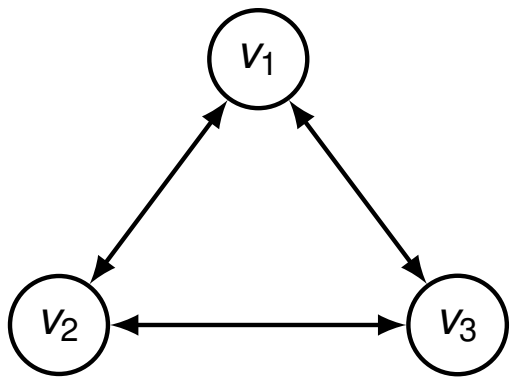
# Safety and Liveness

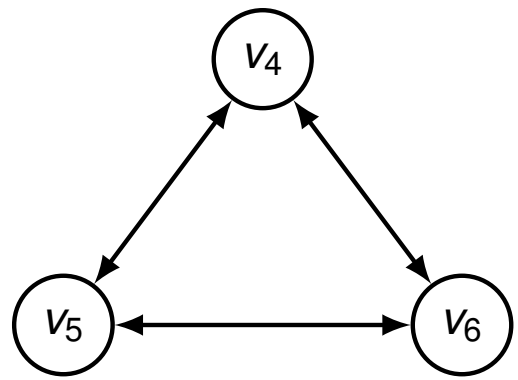
- FBA systems attempt consensus in a slot
- A node applies update $x$ in slot $i$ when
  1. it has applied updates in all previous slots and
  2. it believes all non-faulty nodes will eventually agree on $x$ for slot $i$.

  The node is said to have **externalized** $x$ in slot $i$.
- **Definition**: A set of nodes in an FBAS enjoy **safety** if no two of them ever externalize different values for the same slot
- Well-behaved nodes = obey protocol
- Ill-behaved nodes = Byzantine failures
- Well-behaved nodes can also fail (be blocked or diverge)
- **Definition**: A node in an FBAS enjoys **liveness** if it can externalize new values without the participation of any failed nodes
- Given a specific $\langle \mathbf{V}, \mathbf{Q} \rangle$ and an ill-behaved subset of $\mathbf{V}$, what is the best any FBA protocol can do?

# Quorum Intersection

- **Definition**: An FBAS enjoys **quorum intersection** if and only if any two quorums share a node.
- No protocol can guarantee safety in the absence of quorum intersection
- Example of quorum non-intersection



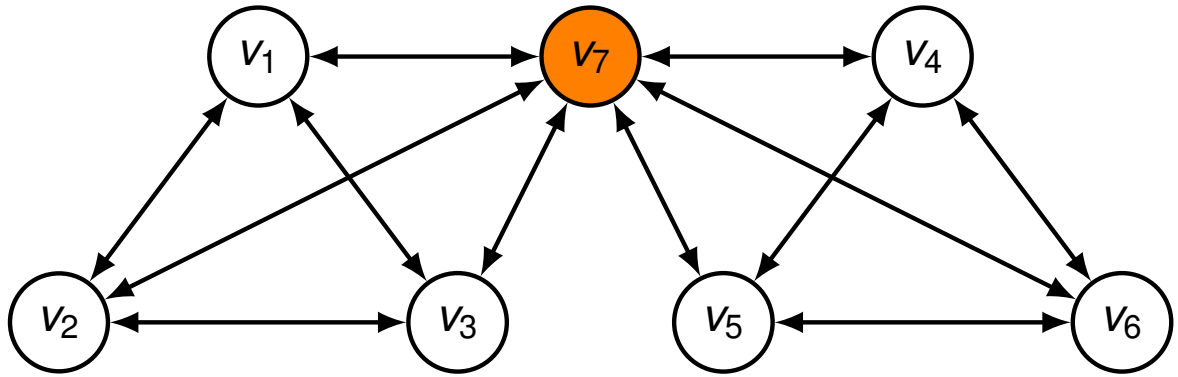




$$\mathbf{Q}(v_1) = \mathbf{Q}(v_2) = \mathbf{Q}(v_3) = \{\{v_1, v_2, v_3\}\}$$

$$\mathbf{Q}(v_4) = \mathbf{Q}(v_5) = \mathbf{Q}(v_6) = \{\{v_4, v_5, v_6\}\}$$

- $\{v_1, v_2, v_3\}$ and $\{v_4, v_5, v_6\}$ are two disjoint quorums; can approve contradictory statements

# Quorum Intersection at Ill-Behaved Nodes




$$\mathbf{Q}(v_1) = \mathbf{Q}(v_2) = \mathbf{Q}(v_3) = \{\{v_1, v_2, v_3, v_7\}\} \qquad \mathbf{Q}(v_4) = \mathbf{Q}(v_5) = \mathbf{Q}(v_6) = \{\{v_4, v_5, v_6, v_7\}\}$$

$$\mathbf{Q}(v_7) = \{\{v_1, v_2, v_3, v_7\}, \{v_4, v_5, v_6, v_7\}\}$$

- If $v_7$ is ill-behaved, the quorums are effectively disjoint
- **Necessary property for safety**: Well-behaved nodes enjoy quorum intersection after deleting ill-behaved nodes

# Stellar Consensus Protocol

- Based on the observation that we care only about well-behaved nodes (intact nodes)
- An optimal FBAS consensus protocol should guarantee safety/liveness for every intact node
- **Theorem**: If the FBAS of intact nodes enjoys quorum intersection, then the SCP guarantees safety.
- **Theorem**: Given long enough timeout and periods in which ill-behaved nodes do not send new messages, intact nodes running SCP will terminate.

# References


- Transactions https://www.stellar.org/developers/guides/concepts/transactions.html
- List of operations https://www.stellar.org/developers/guides/concepts/list-of-operations.html
- Inflation https://www.stellar.org/developers/guides/concepts/inflation.html
- Multisignature https://www.stellar.org/developers/guides/concepts/multi-sig.html
- Ledger https://www.stellar.org/developers/guides/concepts/ledger.html
- SCP talk https://www.youtube.com/watch?v=vmwnhZmEZjc
- SCP white paper https://www.stellar.org/papers/stellar-consensus-protocol.pdf