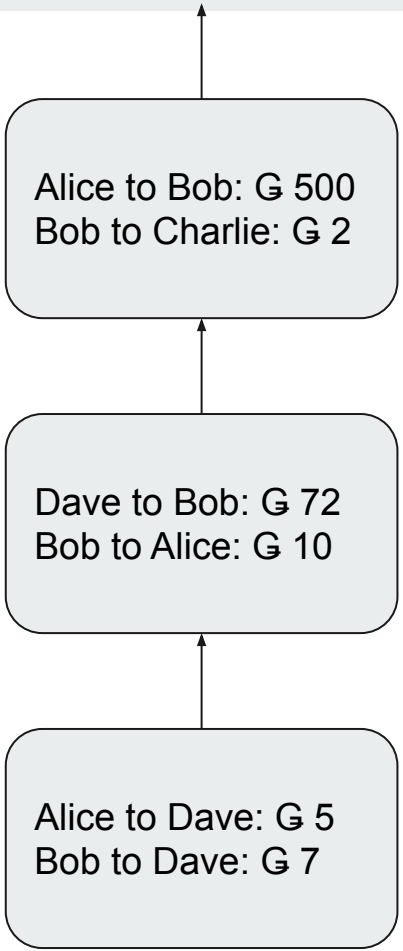
Rust at Concordium

Thomas Dinsdale-Young
ty@concordium.com

What's a blockchain?

- A ledger
  - a growable list of transactions
  - grouped into blocks
- Distributed
  - no central authority
  - consensus algorithm

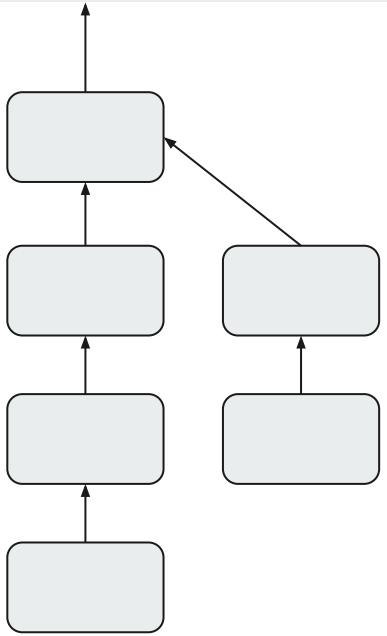


Block production on Concordium

- Bakers collect users' transactions
- A lottery determines when a baker can produce a block
  - Chance of winning is based on the baker's stake ("proof of stake")
- Baker produces a new block with ***valid*** transactions, extending their (best) chain
- The block includes a proof of winning the lottery, and is signed by the baker
- The new block is broadcast, and other nodes verify it and add it to their copies of the chain

Forking/branching

- Forks can happen for several reasons:
  - Multiple bakers can win the lottery at the same time
  - Blocks take time to fully propagate
  - Dishonest bakers can ignore blocks
- Honest participants follow a “best chain” rule, which favours the longest valid chain



Finalization

- The longest chain might not be stable over time (in the short term)
- A second level of consensus, called finalization, periodically marks blocks as ***finalized***
- The best chain rule favours the longest finalized chain
- Finalization cannot* fork
- This gives faster transaction confirmation and greater resiliency

*assuming ⅔ of stakeholders are honest

Transactions

- Define a (local) change to the state
  - e.g. transfer X from A to B
- Have limited criteria for validity:
  - Must be correctly signed by the originating account
  - The sender account must be able to pay the transaction fee
  - Must have the next sequence number for the sender account
- Valid transactions can still fail; invalid transactions are never included in blocks
- The effect of a failed transaction is only to pay the transaction fee

Smart Contracts

- Smart contracts can define custom transactions on the blockchain
- A smart contract consists of
  - State (possibly including funds)
  - Code (WebAssembly) defining operations on the state
- Smart contracts can interact with accounts and other contracts
- Use cases: escrow, custom tokens, voting, games
- We support Rust as a source language for smart contracts

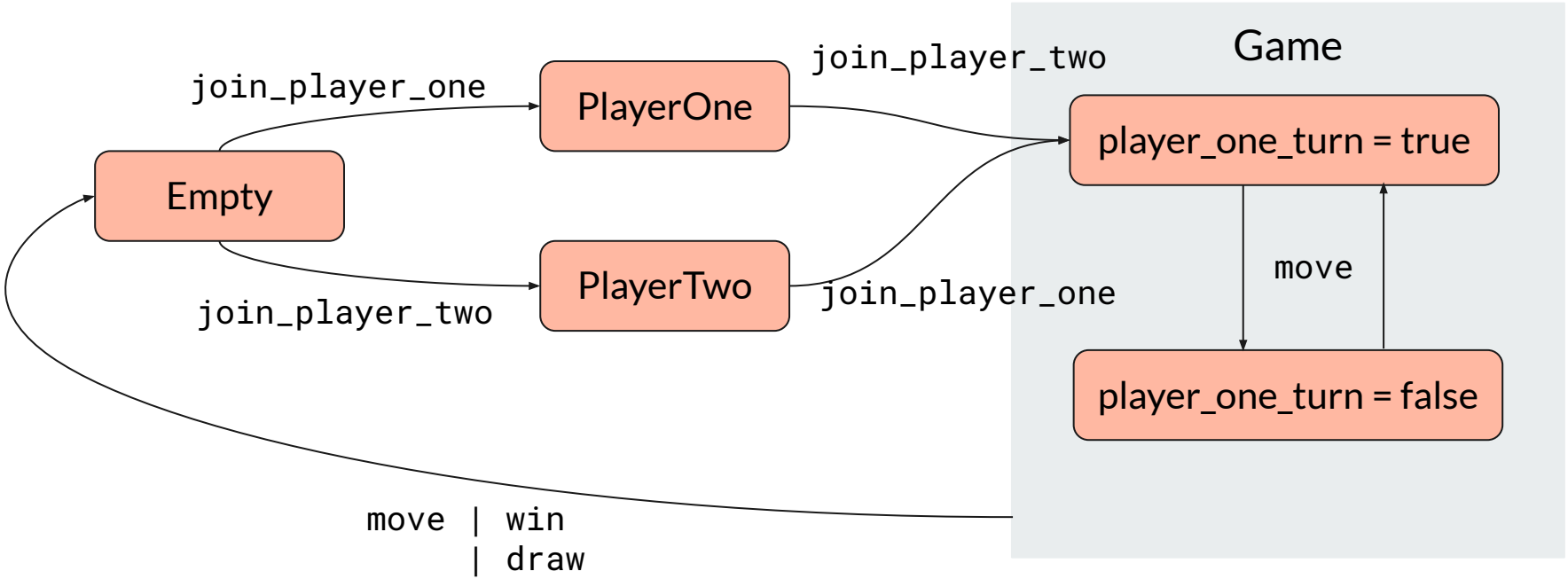
Smart Contracts: Limitations

- Smart contracts are resource limited
  - Every executed instruction and byte of storage has to be paid for
- Smart contracts are not automatic and do not interact with the outside world by themselves
  - Execution is always triggered by a transaction, originating from an account
  - Any off chain events must be triggered by monitoring the chain
- Smart contracts are binding
  - You can’t change the behaviour of an existing smart contract instance
  - (Except by the operations provided by the contract itself)

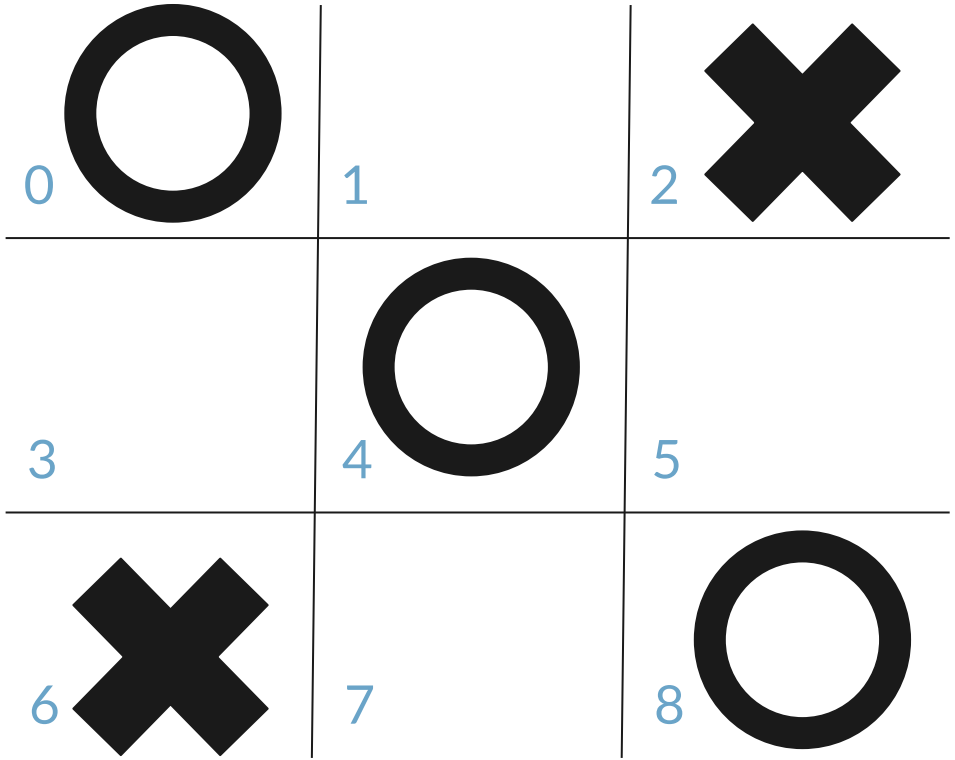
Identity Layer

- Users must verify their identity with an Identity Provider to open an account on the Concordium blockchain
- When creating an account, zero-knowledge proofs establish that your identity has been verified, without revealing it
- The identity on an account can be revealed by the Identity Provider only with the cooperation of Anonymity Revokers
- Zero-knowledge proofs can be used to verify attributes (e.g. nationality) without revealing identity

Smart Contract Demo: Noughts & Crosses



Demo


0
1
2
3
4
5
6
7
8

Find out more...

Concordium:	https://concordium.com
Code:	https://github.com/Concordium
Documentation:	https://developer.concordium.software
Support:	https://support.concordium.software

Node architecture