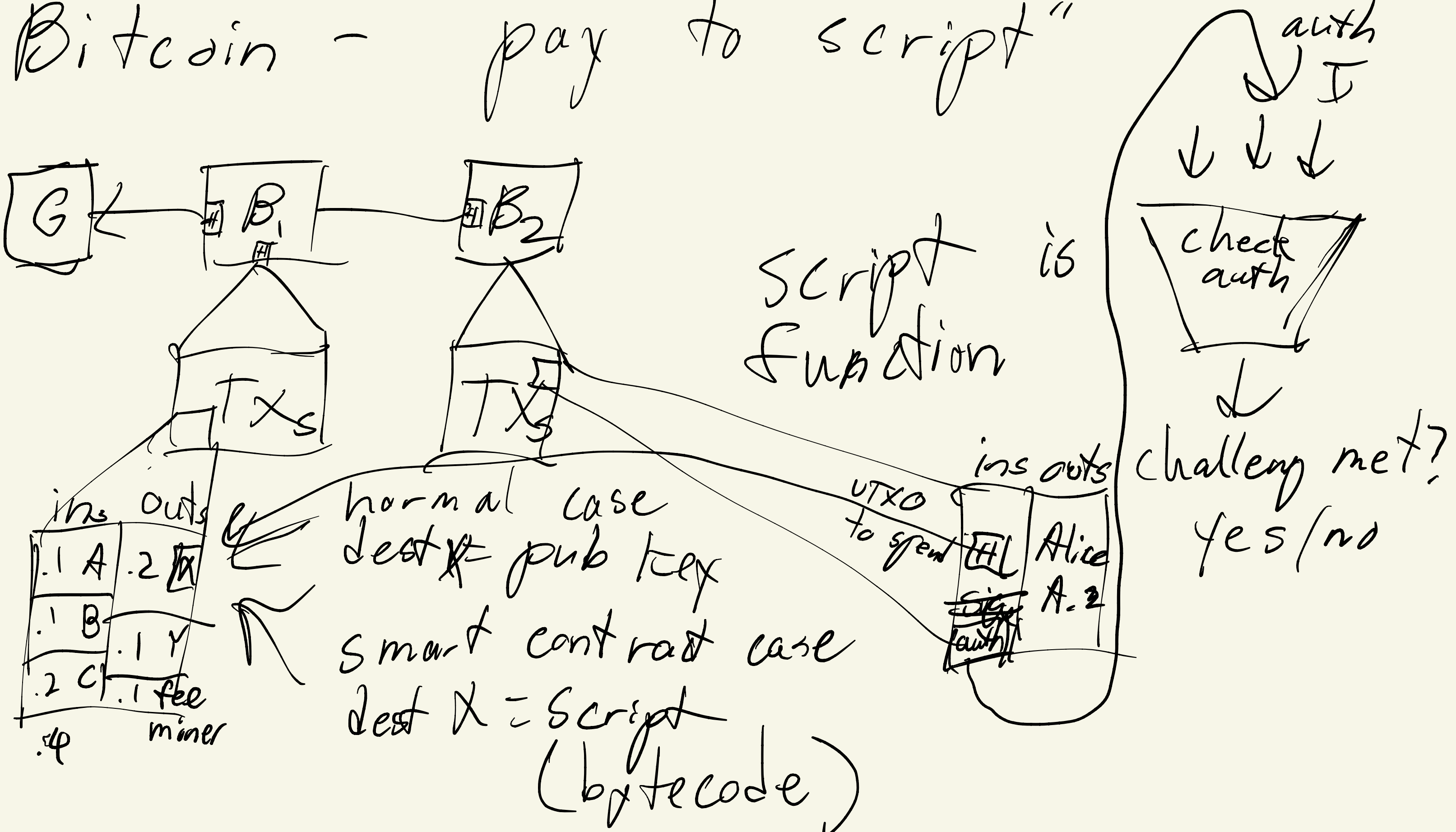
CS-438
Decentralized Systems Engineering

Fall 2022

Week 12

Smart contracts

- Bitcoin - "pay to script"



Script is function

normal case
dest X = pub key

smart contract case
dest X = Script
(bytecode)

Bitcoin - scripting language

- can: multi-signer authorization "multisig"

e.g. any t of n co-signers authorize

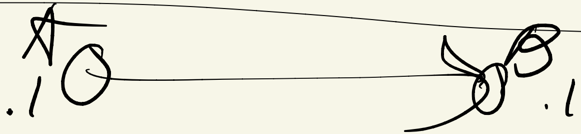
Ex: $t=2$ of 3

script:
```
$a \leftarrow 0$
$a \leftarrow a + \mathrm{check}(K_1, T, I[0 \ldots 63])$
$a \leftarrow a + \mathrm{check}(K_2, T, I[64 \ldots 127])$
$a \leftarrow a + \mathrm{check}(K_3, T, I[128 \ldots 191])$
return $(a \geq 2)$
```

Bitcoin scripts - limitations

- only a few bytecodes
- completely deterministic
- bytecode limited (1 block ≤ 1MB)
- no backward branches

Used for:



- multisig (t-of-n)
- time lock vaults / contracts
- payment channels (Lightning net)
- notaries, side-chains

Ethereum - generalized smart contracts

Differences:

- richer bytecode language (still limited)
- account-based (not UTXO-based)
  accounts persist across transactions
- Turing-complete scripts - w/ loops

Problem: infinite/unbounded execution

Ethereum - gas

- Deterministic arbitrary virtual (execution) time
  - ~ more-or-less instruction count
- Each script execution has a <u>gas limit</u>
  - must pay up-front (invoker or script)

  If script succeeds within gas limit -
  any state changes take effect, gas fee charged

  If script exhausts gas limit
  no state change, but gas limit charged

Ethereum - common uses

- Virtual coins (ICOs)
- Automated market makers (AMMs)
  Uniswap - trade between 2 coins
- Games (Cryptokitties, ...)
- Insurance (AXA Fizzy)
  (needs oracle)