



CS1699: Blockchain Technology and Cryptocurrency

7. Proof of Work and Mining

Bill Laboon

Recall Our Naive Consensus Protocol

- ❖ “Pick a node at random, allow them to add their transactions in their transaction pool to blockchain”
- ❖ Problems:
 - ❖ How to pick a random node?
 - ❖ Assumes honest nodes - easy to subvert
 - ❖ Nothing at stake, no reason to be honest

Incentives

- ❖ We can't punish dishonest nodes but...
- ❖ What if there was an incentive to be an honest node?
- ❖ How about... bitcoins?

Incentives in Bitcoin

- ❖ Two incentive mechanisms:
 - ❖ Block reward
 - ❖ Transaction fees

Block Reward

- ❖ The creator of a valid block gets a reward (currently 12.5 bitcoin - halves every 210,000 blocks, will drop to 6.25 around May 2020) - <http://www.thehalvening.com/>
- ❖ Block reward only valid if it's on long-term consensus branch, which means other nodes must accept it
- ❖ If longest-valid-branch, will incentivize to work off of longest branch and be honest
- ❖ Creating blocks is hard work (as we will see) - don't want to waste it!

The Future of the Block Reward

- ❖ No other way to generate bitcoins besides block reward
- ❖ Block rewards will end around the year 2140 (they keep halving approximately four years, for much longer than that they will be so small as to be insignificant)
- ❖ So... nobody will mine from 2140 on, right?

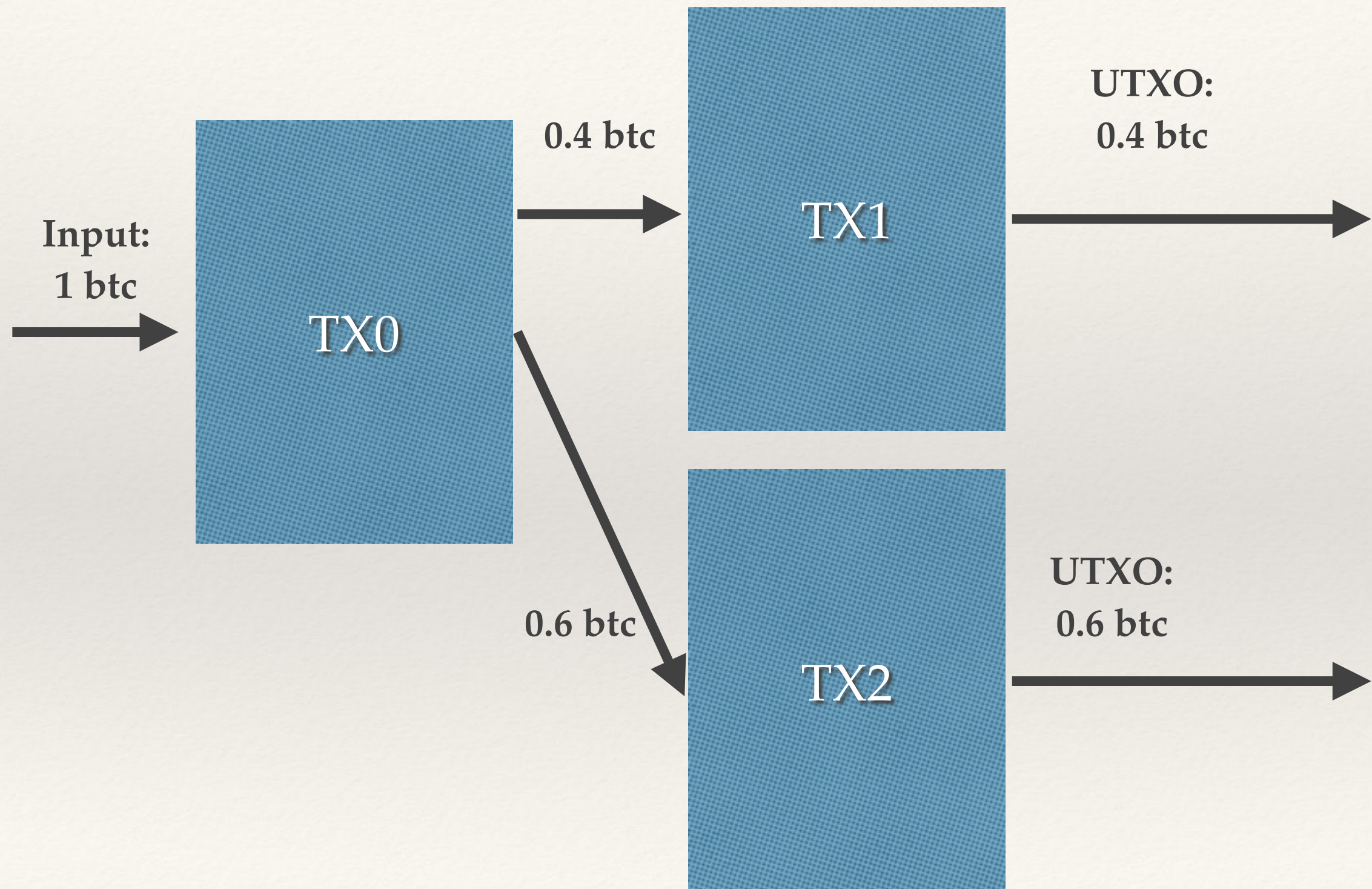
Transaction Fees

- ❖ The other incentive mechanism is transaction fees
- ❖ The creator of a transaction can decide to give themselves back less in “change” than necessary
- ❖ This “missing bitcoin” can be claimed by the miner of a block

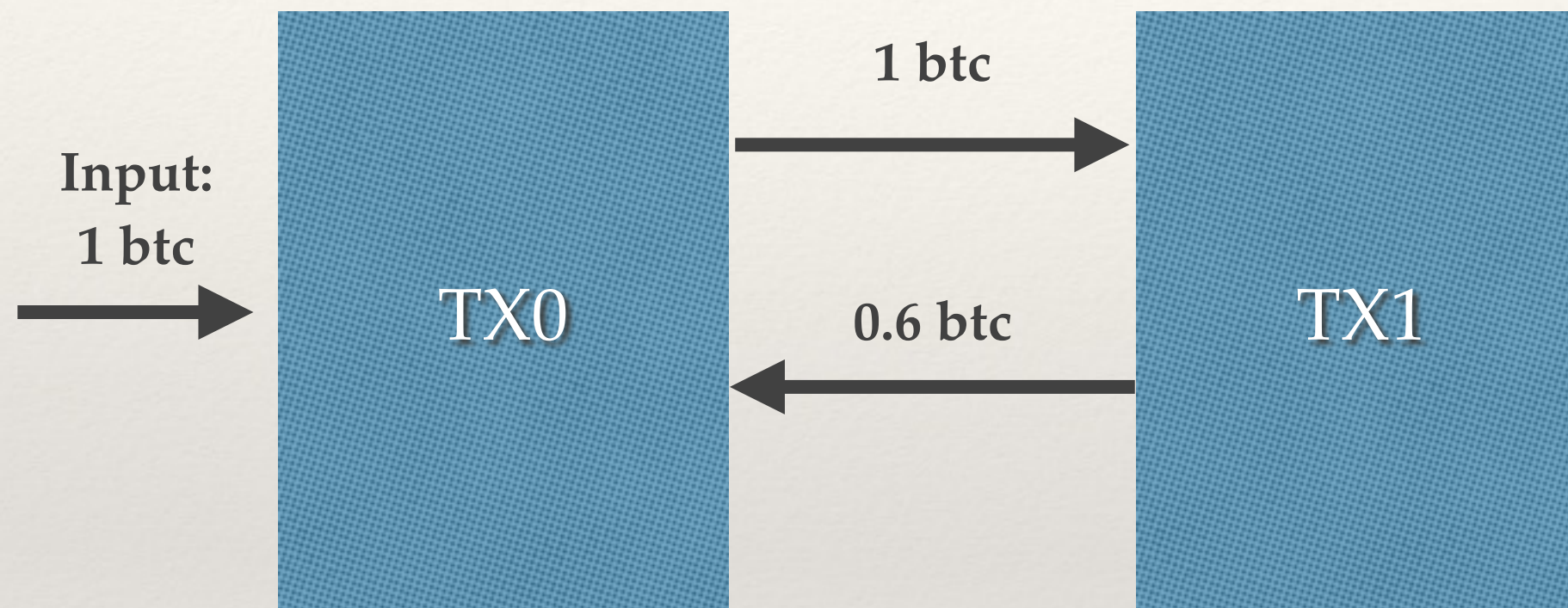
Side Note: UTXOs

- ❖ Although we think of Bitcoin as a distributed ledger where address X has 4 bitcoin and address Y has 2.5 bitcoin, really each has a collection of *unspent transaction outputs*, that is, transactions that have been input to an address but not yet output
- ❖ We can tally up these outputs and determine the full amount of bitcoin that a particular address has, but behind the scenes they are actually a collection of these outputs

Transactions Behind the Scenes

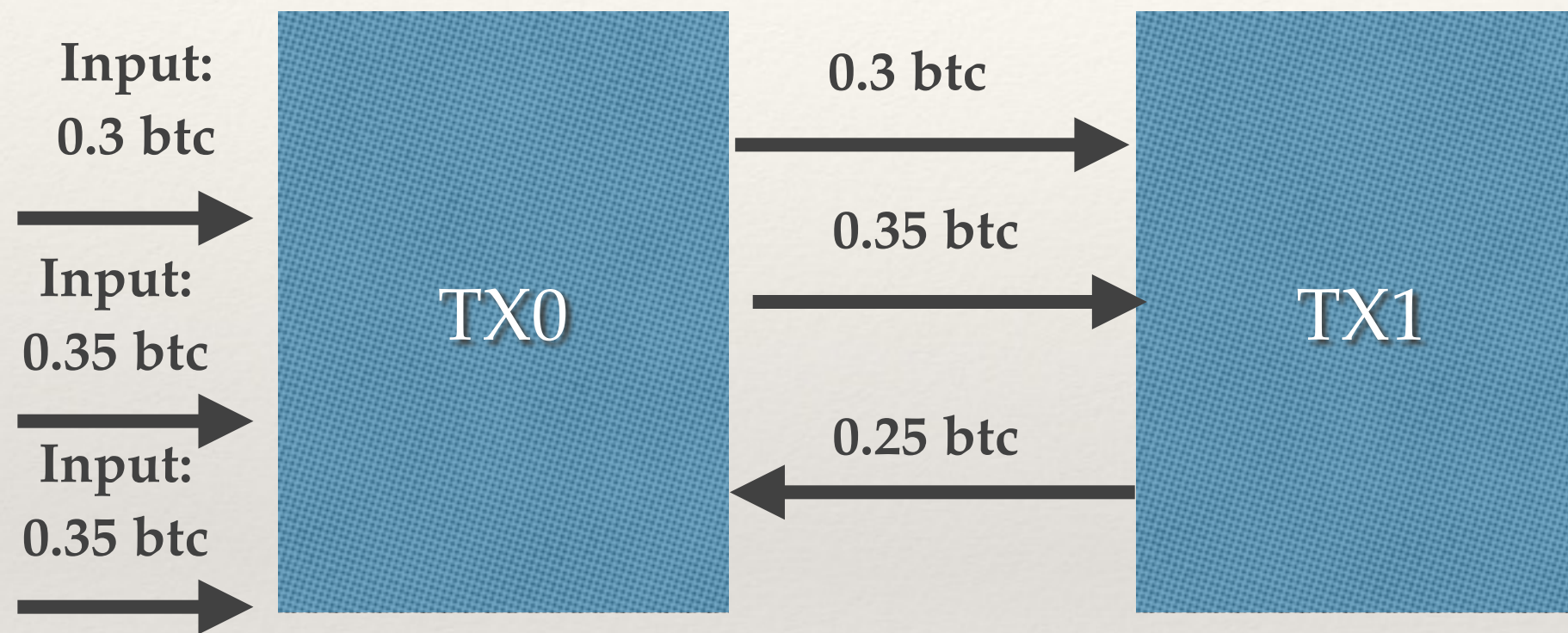


Splitting TXs - Sending 0.4 btc



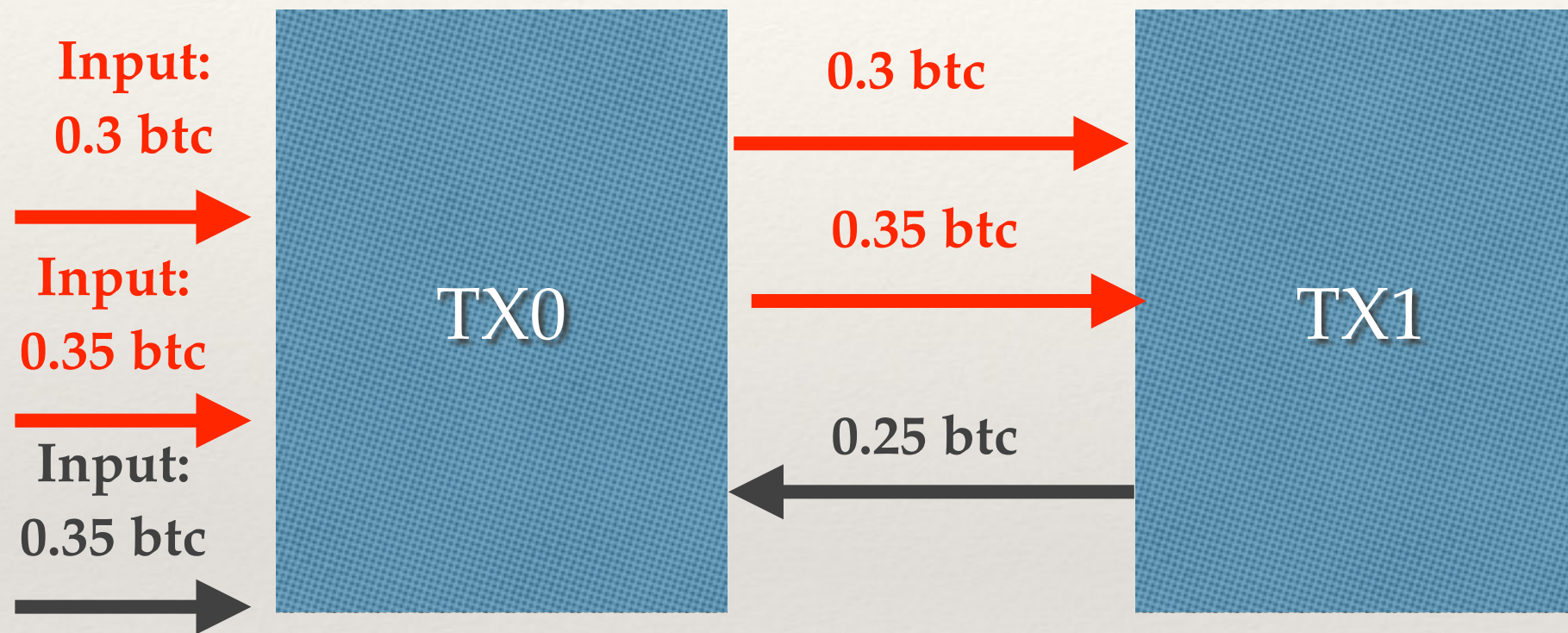
Multiple TX Input/Output

Sending 0.4 btc



Spent vs Unspent Tx's

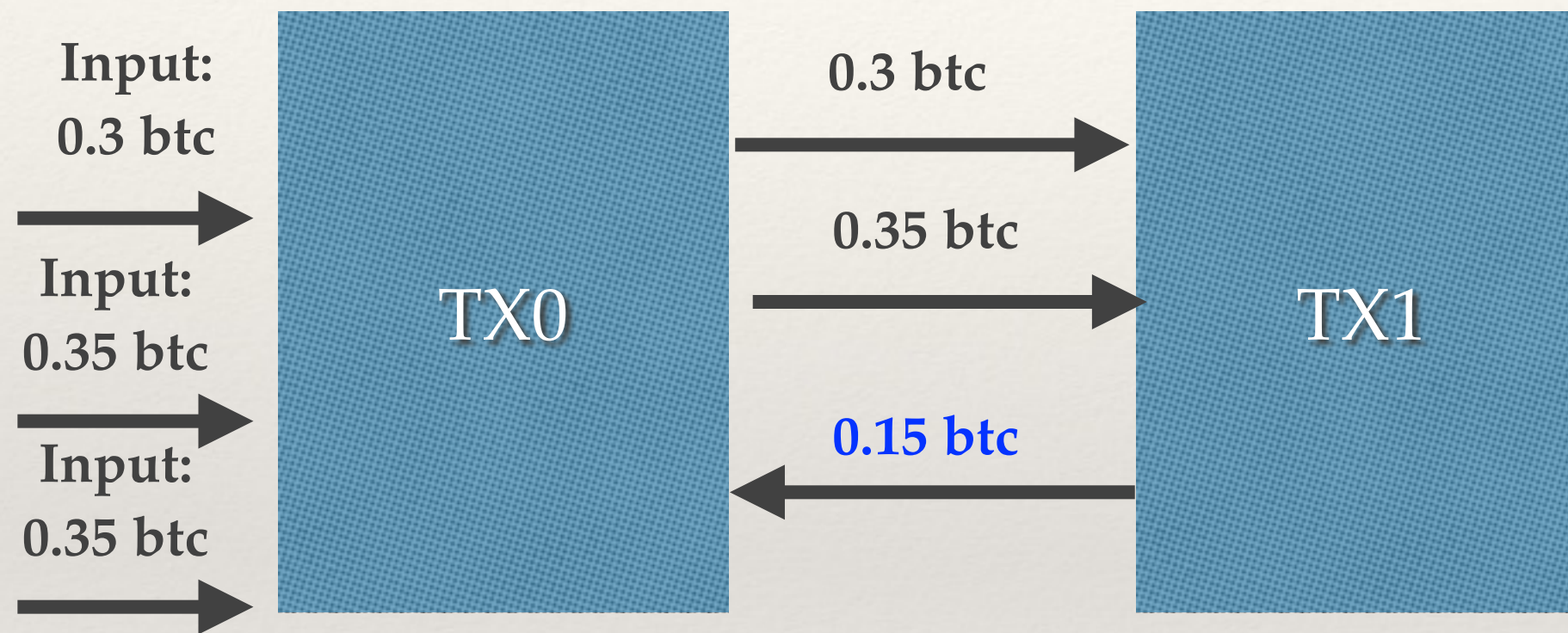
Sending 0.4 btc



Generator of TX0 now has two UTXOs: 0.35 btc and 0.25 btc
Most bitcoin wallets will just show you $0.35 + 0.25 = 0.6$ btc
 $1 \text{ btc} - 0.4 \text{ btc} = 0.6 \text{ btc}$

Multiple TX Input/Output

Sending 0.4 btc



What if we only asked for 0.15 btc back instead of 0.25?

Extra money is the *transaction fee* - whoever mines a block with this transaction can claim it!

Transaction Fees

- ❖ *Ceteris paribus*, miners prefer to include transactions with higher fees
- ❖ There are reasons not to do so, however, which we will discuss later...
- ❖ Think of it as bidding for space or express mail - if you want your transaction in a block quickly, you gotta pay
- ❖ <https://bitcoinfees.info/>

Proof of Work

- ❖ So what's to stop you from going out there and making a block yourself and gathering some transaction fees?

❖ NOTHING! *

* ... As Long You Can Provide a Valid Nonce

- ❖ To create a block is going to require work (a tax on identity - avoids Sybil attacks, or at least cheap ones)
- ❖ Bitcoin uses SHA-256 hash puzzles, i.e., find a nonce (**number used once**) such that a hash of concatenation of the nonce, the hash of the previous block, and all transactions in the block is less than some target
- ❖ $H(\text{nonce} \parallel \text{prev_hash} \parallel tx_1 \parallel tx_2 \parallel \dots tx_n) < \text{target}$

Recall Puzzle-Friendliness of SHA-256 Hash

- ❖ No solving strategy for $H(id \parallel x) \in Y$ better than trying possible values of id
- ❖ In other words, assuming a non-trivial target, we are going to have to try some very large number of nonces (with no shortcuts) before we find a nonce such that $H(nonce \parallel rest_of_block) < target$
- ❖ And trying different values is the ONLY way to do so - no shortcut, one-way function

Hash Puzzle Properties

1. Difficult to compute
2. Parameterizable (i.e. adjustable) cost
3. Trivial to verify

1. Difficult to Compute

- ❖ We want people to prove that they have worked for this (to avoid Sybil attacks / nothing-at-stake problem)
- ❖ As of the creation of this slide, miners around the world are trying ~ 50,419,619,703,000,000,000 (> 50 quintillion) possible nonces every second, and they generally don't find a solution until after about ten minutes of this!
- ❖ This number of attempts per second is called the *hash rate* or *hashpower* of the Bitcoin network

2. Paramaterizable Cost

- ❖ January 12, 2009 - hash rate was (best estimate) $\sim 4,500,000$ (orders of magnitude lower than today)
- ❖ But blocks still were generated at a rate of about one every ten minutes
- ❖ Target re-adjusts itself every 2016 blocks based on (self-reported!) block times
 - ❖ If blocks are taking longer than 10 minutes, difficulty adjusts down (target increases, making it easier to find nonce)
 - ❖ If blocks are taking less than 10 minutes, difficulty adjusts up (target decreases, making it harder to find nonce)

Impact of Puzzle-Friendliness

- ❖ Solving hash values is probabilistic - some blocks are produced within seconds (or even “negative” time - remember dealing with time in a distributed system!), others can take hours
- ❖ Average is ~ 10 minutes
- ❖ https://data.bitcoinity.org/bitcoin/block_time/all?f=m10&t=1

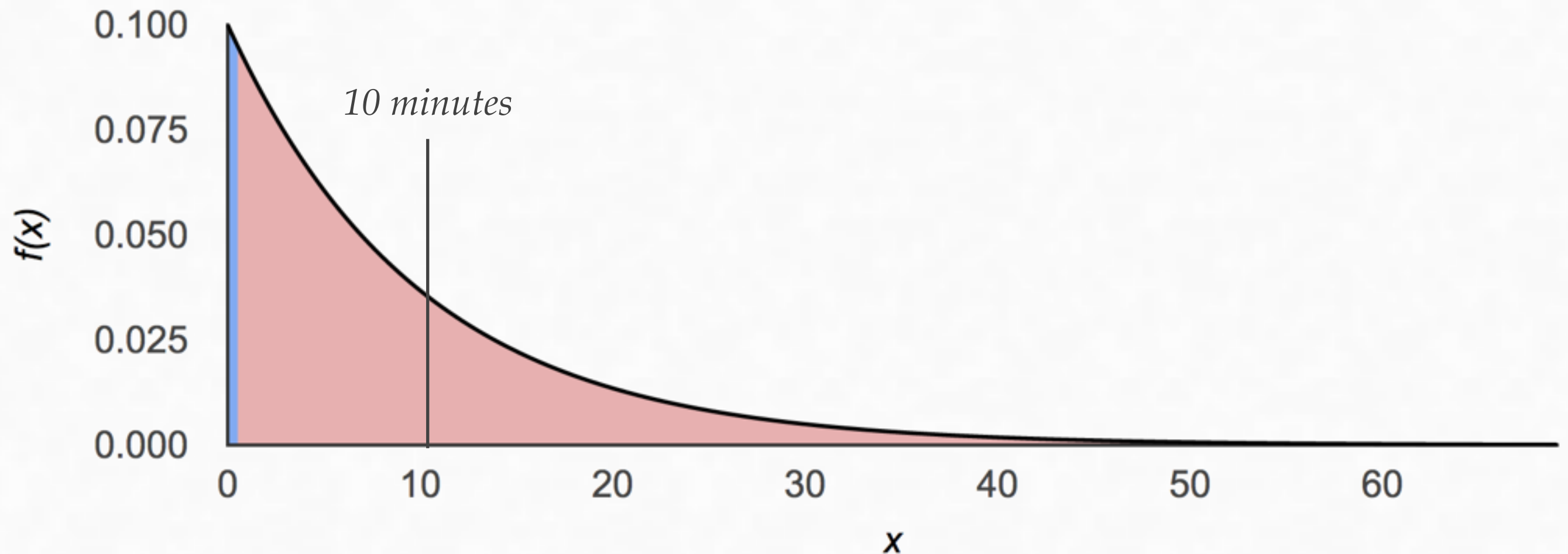
Bernoulli Trials

- ❖ A Bernoulli trial is a process with two possible outcomes (success and failure), and the probability of being successful (and by implication, failure) is fixed between trials
- ❖ Examples: coin flip, dice, roulette
- ❖ Mining: Success - $H(\text{nonce} || \text{rest_of_block}) < \text{target}$
Failure - $H(\text{nonce} || \text{rest_of_block}) \geq \text{target}$

Poisson Process

- ❖ Iterated Bernoulli trials (at large enough numbers so that we can basically act as if the result is continuous instead of discrete) represent a *Poisson process*
- ❖ A continuous probabilistic process where events occur independently at a constant *average* rate
- ❖ Likelihood of a block occurring follows an exponential distribution

Exponential Distribution



Note graph is positively skewed - median < mean.

Poisson Process - Mining Bitcoin

- ❖ No nonces have been tried - what is the mean time until $H(\text{nonce} \parallel \text{rest_of_block}) < \text{target}$?
- ❖ 500 quintillion nonces have been tried - what is the mean time until $H(\text{nonce} \parallel \text{rest_of_block}) < \text{target}$?

Poisson Process - Dice

- ❖ I just rolled 1. How many rolls on average until I roll a 6?
- ❖ I just rolled 4, 3, 1, 5, 2, 3, 3, 3, 2, 5, 4. How many rolls on average until I roll a 6?

Poisson Process - Next Successful Event?

- ❖ For dice: on average (mean), 6 rolls
- ❖ For Bitcoin blocks: on average (mean), 10 minutes
- ❖ ALWAYS, at any point in time - Bernoulli trials have no “memory”, thus they are always — “on average”, i.e. mean — 10 minutes away no matter how much time has already elapsed since the previous block.
- ❖ It's been 45 minutes since the last block? On average, you *still* have 10 more minutes to wait! Poor you.

How Long for a Given Miner To Find a Block?

- ❖ I don't care about the network as a whole, I care about me!

10 minutes

$$\text{mean time to my next block} = \frac{\text{10 minutes}}{\text{fraction of hashpower I control}}$$

3. Trivial to Verify

- ❖ It should be very difficult to MINE a block, but easy for others to verify that it is valid
- ❖ Computationally simple
 - ❖ Run SHA-256 hash function against block with miner-calculated nonce
 - ❖ If $H(\text{nonce} \parallel \text{rest_of_block}) < \text{target}$, block is valid

Mining Overview

```
// Should I mine Bitcoin?  
mining_reward = block_reward + tx_fees  
mining_cost = hardware_cost + operating_costs  
if (mining_reward > mining_cost)  
    return true  
else  
    return false
```

51% Attack

- ❖ Recall that a key tenet of Bitcoin is decentralization
- ❖ Mining allows us to perform distributed consensus and avoid a ScroogeCoin-like (or a Scrooge hiding behind an army of Sybils) centralization
- ❖ But what if more than half of the hashpower is owned by a single entity? Is this a new Scrooge?

What Can Our Pseudo-Scrooge Do?

- ❖ *Steal coins?* **No.** The ONLY way to move coins is knowing the secret key of the owner (public key).
- ❖ *Suppress transactions?* **Yes.** Just like Scrooge, they can ignore transactions and refuse to put them in blocks - but users will see transactions in mempool and realize this attack is occurring.
- ❖ *Change the block reward?* **No.** They cannot change the rules of consensus - valid nodes will ignore the blocks.
- ❖ *Double-spend?* **Yes.** They can decide to build off a blockchain which does not include the block which includes a sent transaction.
- ❖ *Hurt confidence in Bitcoin?* **Yes.** A key benefit of Bitcoin lies in its decentralized nature; destroying this may start an exodus to alternatives.

A Pale Yet Dangerous Shadow

- ❖ Scrooge could change the block reward, suppress transactions without others realizing, or even change the rules of consensus - none of which our Pseudo-Scrooge can do.
- ❖ Decentralization has benefits even when it's been centralized!