Validation-cost Metric

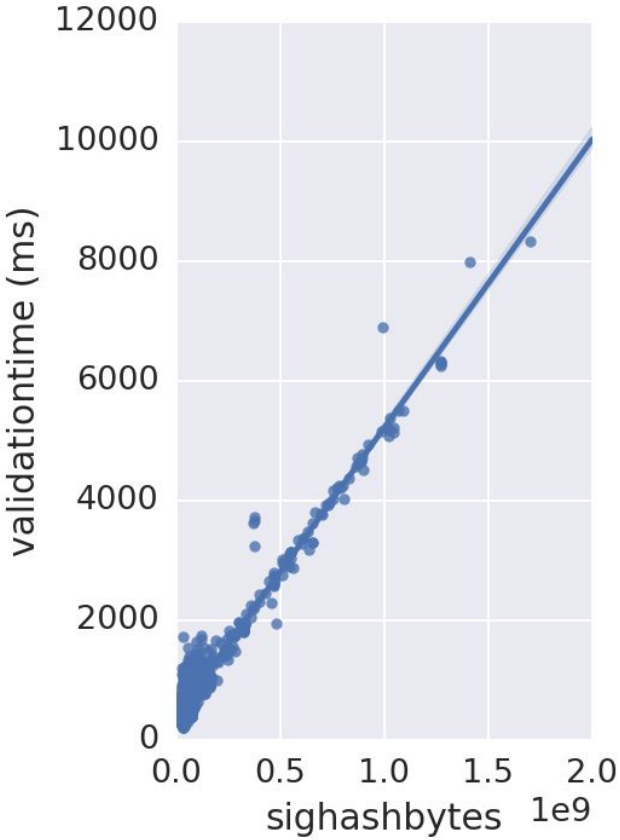
Jonas Nick, Greg Sanders, Mark Friedenbach

Motivation

- Block size correlates with resource usage only in the typical case
- OP_CHECKSIG example
  - involves transaction hashing and signature verification
- Hard limits
  - how to choose them?
  - other factors influence validation cost as well
  - ignores relationship between factors
- Instead: Use metric
  - function from block features to cost
  - example
    - `cost(block) = c_0*size + c_1*validation_cost + c_3*utxo_growth`
  - new consensus rule: `cost(block) < t`

Validation-cost Metric

- Validation-cost: how long it takes to validate a block on a reference machine
- Estimate `c_i` with linear regression
  - `validation_cost(block)`
    `= connect_duration(block)`
    `= c_0*h + c_1*v + …`
  - `h: bytes hashed,v: verifications`



1-dimensional example

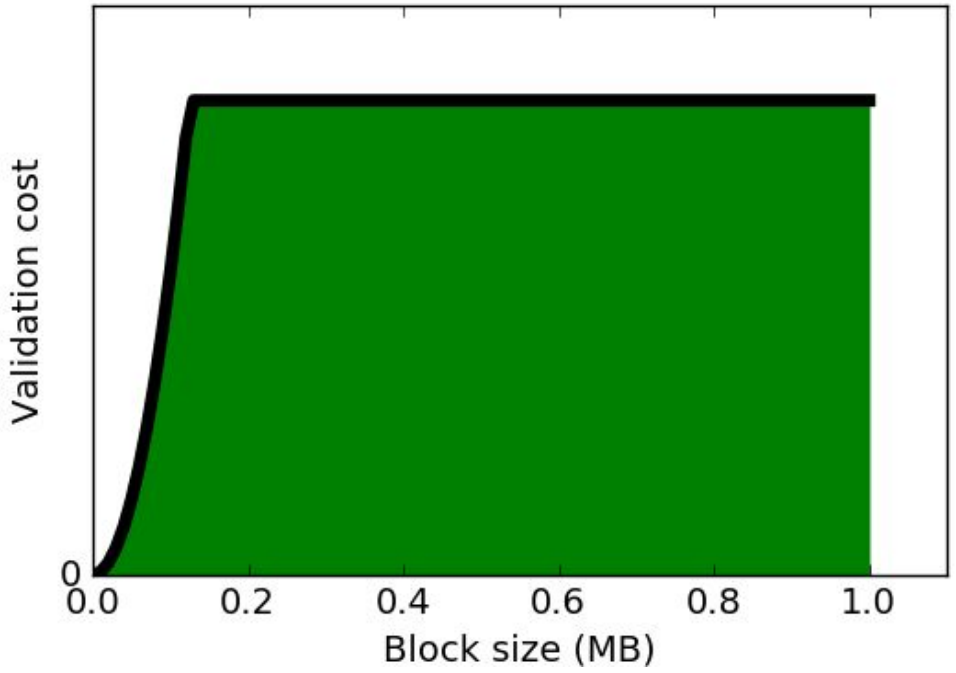
Validation-cost Metric: Experiment

- Record
  - OP_CHECKSIG bytes hashed and number of verifications
  - OP_HASH bytes hashed
  - number of bytes written and removed from the stack
  - number of inputs
  - ConnectBlock duration on reference machine
- Reference machine: laptop from 2014, 2*3GHz i7, 8GB RAM
- data: mainchain, testchain, custom chains
- v0.11.2 with libsecp validation
  - -dbcache=3000

Validation-cost Metric: Results

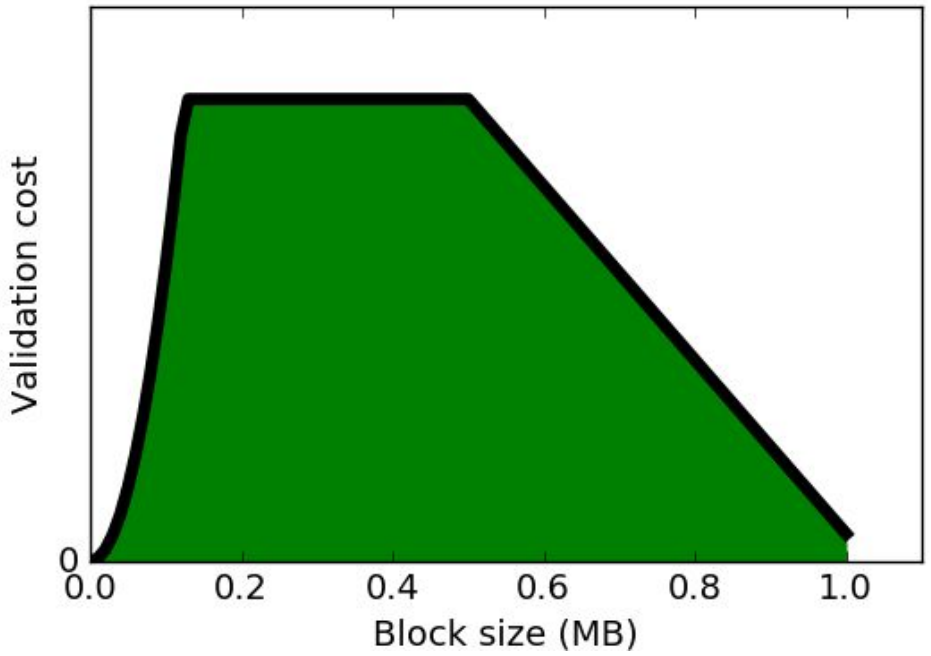
- Each kbyte of hashing adds 0.005ms, each signature verification 0.1 ms
- Other factors do not have a big impact at the moment
- Absolute average error on test and mainnet: less than 4ms
- Example of hard-to-validate block predicted accurately
  - 130.4 vs. 131.7 seconds

Cost Metric



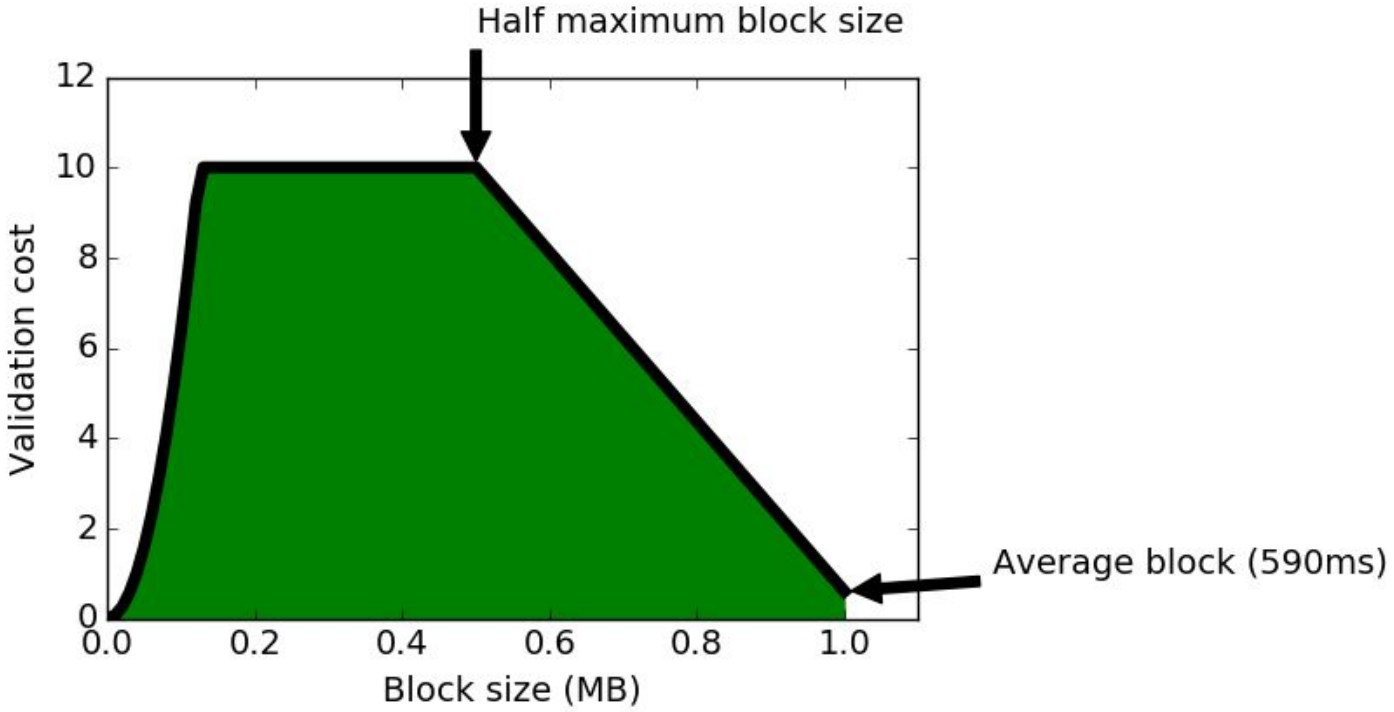
- Can pick another threshold for validation-cost
  - difficult: do not want to constrain use-cases, do not want worst-cases to sum up

Cost Metric



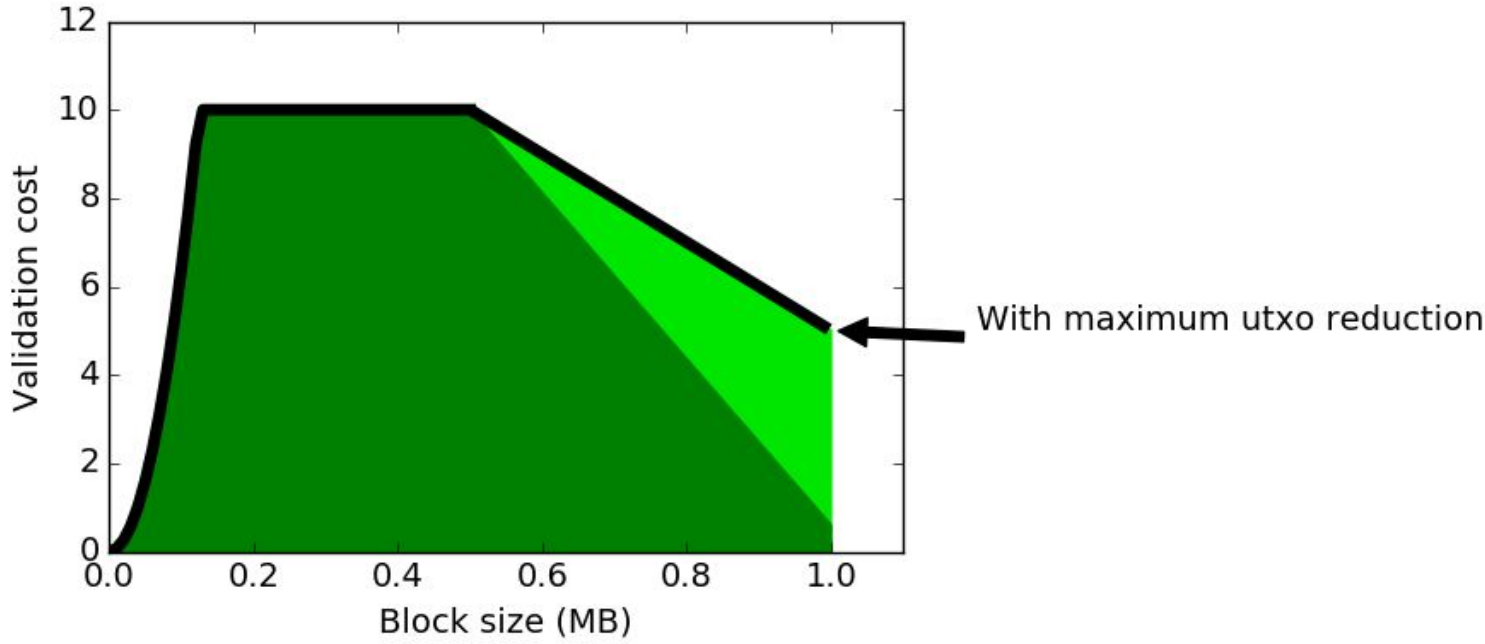
- Relate bandwidth requirements and validation cost then pick threshold
  - -> cost metric
- How exactly do you convert bandwidth requirements and validation-cost to total cost?

Cost Metric



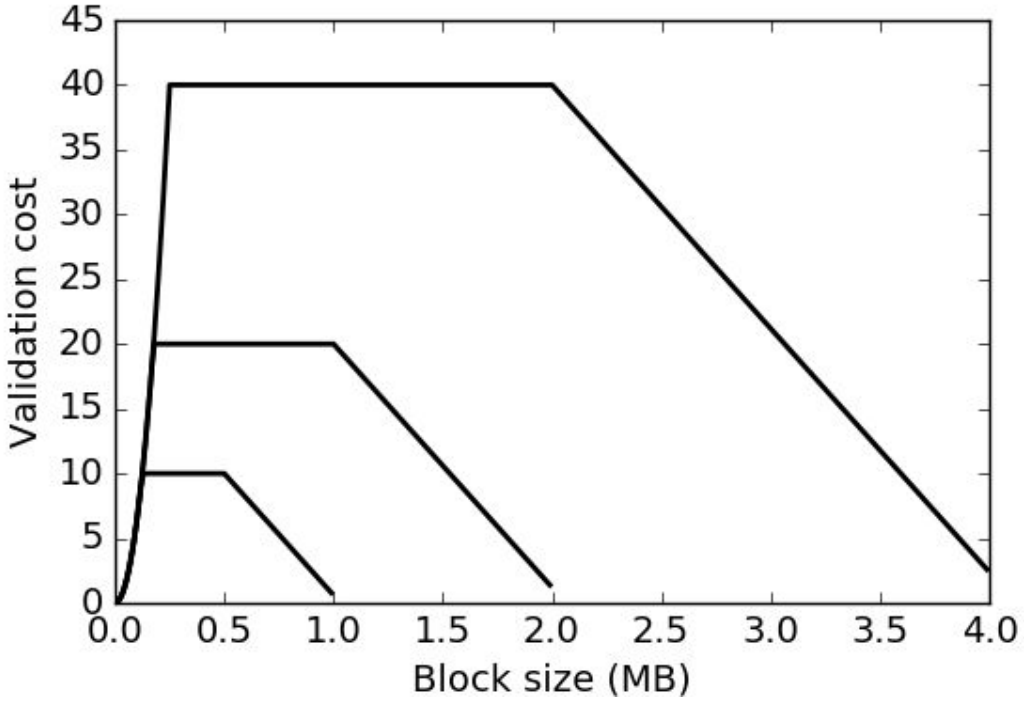
- Instead: fix max block size (given by various proposals)
  - and always allow “average blocks” for any block size
  - while enabling to trade-off block size with validation time
  - up to a hard limit

Cost Metric



- Maintaining a utxo set is a significant cost
- Current situation: no incentive for positive effect on utxos
- With cost metric: Allow a greater validation costs when the block reduces the utxo set size.

Cost Metric



- Increase maximum and average block validation cost at same rate as block size.

Conclusion

- There are various resource requirements
  - block size proposals should consider
- Cost metric helps by exploring relationships
- Estimating validation cost function straightforward
- More complete cost function difficult to derive bottom-up
  - but can build on existing blocksize proposals
  - while still getting some of the advantages of a cost metric
    - confining worst-case without restricting current average case
    - allowing to trade-off individual block aspects
    - enables to set slight incentives

Links

- https://scalingbitcoin.org/montreal2015/presentations/Day2/11-Friedenbach-scaling-bitcoin.pdf
- Benchmark: https://github.com/instagibbs/bitcoin/tree/rt
- Discuss: http://lists.linuxfoundation.org/pipermail/bitcoin-dev/2015-November/011662.html