Ethereum in Numbers - Where physics meets TPS

Notes from [Peter Szilagyi's

](https://x.com/peter_szilagyi/status/1647489536153513986?s=20) stellar presentation at Eth Prague 2022. The key takeaway is that state growth is the bottleneck and it brings an existential risk along with it – see the brick wall and Binance Smart Chain discussion .

Ethereum and TPS is usually not a conversation you want to read on the internet. Usually when you read about TPS and Ethereum, it's about how Ethereum sucks and your favorite Ethereum killer rules all. How much truth is there to this?

Physics limits you to certain aspects, and then we construct many extras on top.

Ethereum Does not scale

- Expensive or too expensive? Capacity supply vs. demand?

That is a very ambiguous thing to unpack. What people want to say is that Ethereum is too expensive. Is a Tesla car too expensive? That depends on your salary and what you would like to use the car far. Too expensive is subjective, but we can agree that Ethereum is expensive. Capacity is driven by supply and demand. If you would like to make Ethereum cheaper, then there are two things we can do (i) raise capacity and (ii) lower demand.

- The EF is working on both by convincing people to use rollups which lowers demand.

Ethereum can barely do 15-25 TPS. Can capacity be increased? Capacity defined as TPS is super ambiguous; is the transaction an ETH send, a DEX swap, NFT mint, etc.? These transactions are not comparable.

But what Ethereum does under the hood is it uses gas to abstract out the disproportionately of transactions.

- Ethereum runs 1.1M gas per second

Avalanche C-chain 1.8M, Binance 7M (previously 25M), but their using Geth!?

Is there something these teams aren't telling us, they are running EVM using the same state model, how can they push it faster?

There is a really nice hint that they are not telling us. Binance Smart Chain was running at 25M gas per second. Currently, it's running at 7M gas per second. How did the super smart Binance smart chain guys optimize geth then unoptimized it a year later?

Gas is a fairly bad approximation of how much it costs to run a transaction. Usually we have 4 aspects to system load

- CPU load
- Memory
- Disc
- Network

Running a transaction this year and last year takes the same amount of cpu, memory, network traffic, however disc is interesting.

Bane of Ethereum: Merkle Patricia trie

- Merkle tree containing account data leaves, linked together via 16-child internal nodes
- Catch: The more accounts there are, the deeper the state trie becomes

This is how Ethereum stores its state. All the accounts are laid out flat, and we have a cryptographic tree structure on top to prove that everything is correct. Since this is a logarithmic structure, every node has 16 children every mathematician would say this is super efficient. The problem there is that as time passes and the state becomes bigger then this logarithm starts to appear and cause troubles.

Logarithmic depth surely doesn't matter?

- Ethereum has ~ 174M accounts => 6.86 internal depth + 1 leaf layer
- state trie of depth 8 is perfectly acceptable

- state trie of depth 8 is perfectly acceptable
- Plain transfers update 2 accounts => 15 new nodes in the account trie
- create 8 for new account of Alice and 8 for Bob
- create 8 for new account of Alice and 8 for Bob
- LevelDB stores data in 7 disk layers => amplifies at worse to 105 writes
- Amplifies 15 writes into 105 writes
- Amplifies 15 writes into 105 writes
- Need the old Merkle root path read for root hash calculation => bumps to potentially 210 IO ops per transactions
- Mined blocks need to propagate => 210 ops miner side, 210 ops full node side
- running a single transaction then requires 420 I/O operations on the network
- doesn’t sound bad, except when you start laying out the exact numbers of different technologies
- running a single transaction then requires 420 I/O operations on the network
- doesn’t sound bad, except when you start laying out the exact numbers of different technologies

HDD

capped at

80 IOPS =>

0.19 TPS

(x2/3 = 0.12 TPS with disk pruning)

SSD (SATA 6)

capped at

90,000 IOPS

214 TPS

(x2/3 = 142 TPS with disk pruning)

SSD (NVMe over PCIe 3)

capped at

360,000 IOPS

857 TPS

(x2/3 = 571 TPS with disk pruning)

SSD (NVMe over PCIe 4)

capped at

1,000,000 IOPS

2381 TPS

(x2/3 = 1587 TPS with disk pruning)

- Every piece of hardware has its limits. If we take these limits and divide by 420(210 x 2) we get some super hard limits about what is possible
- Ethereum currently aims for PCIe3 NVMe hard drive which means that technically 857 TPS is the absolute maximum the Ethereum network can do.
- We are doing more than that now - the way we are doing more is that these limits are only relevant if every transaction pushes everything to disk like an archive node

- What we try to do is keep as much data as possible in memory so we don't have to screw around so much with disk
- We are doing more than that now - the way we are doing more is that these limits are only relevant if every transaction pushes everything to disk like an archive node
- What we try to do is keep as much data as possible in memory so we don't have to screw around so much with disk

To raise TPS we must lower the disk IOPS:

- Keep things in memory and avoid hitting disk
- OS uses free memory as disk cache => db shuffling in RAM - If you have a 64 GB machine the OS will use the whole RAM to cache the disk. W/e you try to access all the data it will try to access through memory
- Geth does in-memory (state) pruning => ephemeral state never hits the db - if Alice sends one Ether to Bob, one to Charlie and one to Dave, you don't need intermediate states where Alice balance was decreased by incremental amounts. Its fine to just save the last state.
- OS uses free memory as disk cache => db shuffling in RAM - If you have a 64 GB machine the OS will use the whole RAM to cache the disk. W/e you try to access all the data it will try to access through memory
- Geth does in-memory (state) pruning => ephemeral state never hits the db - if Alice sends one Ether to Bob, one to Charlie and one to Dave, you don't need intermediate states where Alice balance was decreased by incremental amounts. Its fine to just save the last state.
- These are nice optimizations that allow us to reduce the I/O operations and thus raise the TPS capacity of the Ethereum Network

Unfortunately, system memory is still limited

- State outgrows the RAM => db writes revert to physical disk writes
- State becomes bigger => pruning needs more RAM or it flushes more

If the OS is nice at caching stuff in RAM if I have enough RAM to fit the entire state into it or we can use nice in memory pruning in GETH if we have a enough RAM to fit it into it.

Not entirely sure what the average GETH Ethereum node user has, how much RAM (32, 64GBs), but the current Ethereum's state is way over that. The more you overflow it the more often the OS produces a cache miss or page fold that results in reaching down to disk.

The same happens with pruning. I can prune and stuff a certain amount of stuff of data into the pruning algorithm but as the blocks or state grows I either need to raise that limit or keep leaking more data down to disk. This is a self-referencing cycle because the more the state grows the less RAM there is to operate in it. The faster the state grows the less RAM there is to operate in it. What happens is that we eventually revert back to the original TPS cap produced by the hard drive.

The question is; Big state is bad, how big is bad?

How fast does it grow? The faster it grows the worse it gets.

- Example June 5, 2022 - Sunday
- ~0.64 accounts/s, 7.8 storage slots/s
- ~31.7 B/s for accounts, 593 B/s for storage ~= 54MB/day, 19.7 GB/yr
- ~0.64 accounts/s, 7.8 storage slots/s
- ~31.7 B/s for accounts, 593 B/s for storage ~= 54MB/day, 19.7 GB/yr
- Catch: above growth is pure useful state data (no merkle proofs)
- Account trie weighs 155.2B/acc, storage tries weigh ~142.3B/slot
- Trie grows 99.3B/s for accounts, 1110 B/s for storage ~= 104.5MB/day, 38.2GB/yr
- Account trie weighs 155.2B/acc, storage tries weigh ~142.3B/slot
- Trie grows 99.3B/s for accounts, 1110 B/s for storage ~= 104.5MB/day, 38.2GB/yr
- Data was interpolated on a Sunday can be 2-3x more during the week. Perhaps 50-100 GBs per year is a better estimate for the raw state needed for contract execution.

- Is this a lot or acceptable? If I have 64 GBs of RAM then piling 100 GBs per yr on top of it uses it up fast.
- Is this a lot or acceptable? If I have 64 GBs of RAM then piling 100 GBs per yr on top of it uses it up fast.

What does this all mean?

Ethereum (along with all its forks) is on a potential death trajectory - if you take any of the Ethereum networks and just stick a constant transaction throughput on it the state wilkl grow. Binance smart chain was growing at 2.5 TB/yr at one point for example.

- Constant TPS => state growth => higher RAM => more IOPS => lower TPS => brick wall
- Mainnet can do a lot more TPS, it brings the brick wall closer
- This is kind of a race against time. Can we keep that brick wall far away so it doesn't impact us in a detrimental way?
- This is kind of a race against time. Can we keep that brick wall far away so it doesn't impact us in a detrimental way?

The moment we revert to hard drives is the moment we impose hard limits.

But does the brick wall need to exist?

- EIP 4444 - Bound Historical data in execution clients (doesn't help state growth)
- EIP 4844 - Shard blob transactions (Move data to L2s and delete the data at L1)
- Neither touch on raising capacity or moving brick wall forward, we are kind of running out of time
- State rent or exponential costs
- Possibly stateless clients
- State rent or exponential costs
- Possibly stateless clients
- This does mean no solution exists currently, and Ethereum ahs time. If you look at other chains who are pushing the limits super hard is they will beat Ethereum to the brick wall on chain growth.
- You can see some of this on BSC when you dig through their issue tracker
- You can see some of this on BSC when you dig through their issue tracker

Ethereum's TPS is deliberately low. We could push it significantly harder, but it would give us a significantly shorter time span to fix things. The good news is we are trying, and we have pioneers pushing it harder than us, so we can learn from them.

What if computers just get better and fast enough?

That would be nice, but they don't. Computers get better in certain areas, but current limitations are disk latency - The best in class is PCIe NVMe hard drives which can do ~ 1M IOPS per second which translates to ~2000 TPS, perhaps. There is nothing really coming that will make it significantly faster. There are always some fancy technologies you can deploy in a DC if you want to spend money, Ethereum tends to take the more public approach where anyone can run a node. Then you need to keep the hardware requirements within reasonable levels. Probably nothing coming for average users to make it 10x better. Long-story short is L2s are the way to go and L1 needs to be kept in reasonable shape.