Computational Complexity and Blockchain Scalability

Lulu Wang

December 6, 2021

I offer a framework to understand which blockchain scaling solutions are enough to cover a non-trivial fraction of every person's internet activity. The core of the approach combines computational complexity with economics. Computational complexity predicts how the computational requirements of different systems will change as the number of participants increase. Economics then helps us map these computational requirements into predictions for how many indivduals will be able to use the blockchain.

Computational complexity offers a more useful approach for understanding blockchain scalability than transactions per second (TPS) statistics. Using TPS to rank blockchains is similar to using clock times to rank sorting algorithms. A well implemented insertion sort outperforms merge sort on small lists, but computational complexity provides a framework for understanding why merge sort is better on large lists. This note makes precise the sense in which sharding is quadratically better than base blockchains, and how zero knowledge proofs are exponentially better. It also predicts that if zero knowledge proofs can achieve further speed improvements and achieve linear time proof generation, the vision of an infinitely scalable (as opposed to merely exponentially scalable) blockchain can be achieved.

Economics shows us that an infinitely scalable blockchain can only consume computational resources that grow linearly with the number of participants. In many computer science applications, an additional logarithmic complexity factor is of negligible importance. But the key insight is that the number of participants who can join a blockchain is determined by the per capita cost of maintaining the blockchain. A blockchain whose costs grow linearly with the number of participants has the potential to cost a constant share of income as the system grows. A blockchain whose costs grow even a little faster, say $N \log N$, means that each additional participant raises the cost of maintaining the blockchain for every existing participant. In the language of economics, a blockchain whose costs grow faster than linearly in the number of participants suffers from decreasing returns.

Understanding how blockchains scale is also crucial for understanding whether the future will be single or multi chain. On one hand the desire to maintain composability of different decentralized finance protocols may push for a single chain future. But if blockchains are not infinitely scalable applications may eventually be pushed into separate chains due to congestion and high gas fees.

I focus on scaling computation, while assuming storage and networking are solved. I do this for two reasons. First, many of the insights on difficulties in scaling computation can be extrapolated to the storage problem. Second, many of the recently developed rollups focus first on scaling computation, and so a framework to understand developments in that space has high value. Ignoring storage and network means this framework is not appropriate for understanding how data sharding affects rollups, or how new off chain data storage solutions such as zk Porter can improve scalability. Rather, this note offers a look at the very long run properties of solutions for scaling computation.

1 Model

The key starting thought experiment is to imagine a community of N individuals, each with some large k units of computing power, who each need to conduct 1 transaction. One can think of these calculations as financial transfers of wealth from one individual to another. There is a need for these transfers to be socially recognized, thereby creating a natural demand for a shared ledger. For each blockchain scaling solution, I ask how large of a community can be supported. I operate under the assumption that each transaction costs one unit of computing power to run. Relaxing this assumption merely alters the proportionality constants, and does not change the "big-oh" of how many community members can be supported.

1.1 A Blockchain Without Scaling Solutions

A simple blockchain can only support a community of $N = k$ members. Thus, the size of the community is constrained by the average compute capacity of a member. The reason for this constraint is because security in a simple blockchain is guaranteed by every participant validates every transaction on the network. Since each individual can only validate k transactions, and each individual wants to conduct one transaction, then at most $N = k$ individuals can participate in this blockchain.

1.2 Quadratic Sharding

Quadratic sharding can support a community of $N = k^2$ participants. In such a protocol, a set of coordinators split individuals into committees and then have each committee agree on a set of transactions in a format similar to a standard blockchain without scaling solutions. Within each committee, the members run a standard blockchain consensus algorithm as above, in which each participant verifies every transaction assigned to them.

Quadratic sharding is limited because the number of shards that can be managed is limited by the compute power of the coordinator. Assume that it takes one unit of computation power to assign people into committees and to verify that the committee correctly processed the transactions

they were assigned. This is reasonable because verifying the signatures attesting to the validity of k transactions takes a small amount of time relative to actually verifying the k transactions.

Since the coordinator can execute at most k calculations, then at most there can be k shards. Each shard operates as a regular blockchain, and thus each shard can only have k participants who process k transactions. Hence the total amount of transactions processed in this world is at most k^2. If each participant wants to compute one transaction, this means that the maximal N is k^2. The power of 2 is also the reason why sharding of this form is known as quadratic sharding.

1.3 Quasilinear Zero Knowledge Proofs

Scaling with zero knowledge proofs allow a community that is asymptotically exponential in the compute power of the median participant, i.e. $N = O(\exp(k/c))$ for some natural number $c \in \mathbb{N}$. Zero knowledge proofs are a new technology that enables one party to compute a large number of transactions and release a succinct proof to the world that the result of the transactions is correct. Then others can verify that the calculation was done correctly without having to do the full calculation. In general, the current state of the art systems (such as those pursued by Starkware) enable the prover to generate a proof of N transactions in with $O(N \log^c N)$ calculations and allow the verifier to check the proof in $O(\log^c N)$ time for some natural number $c \geq 1$[1].

Zero knowledge proofs enable exponential scaling. Here we resort to asymptotic notation. Suppose we had a system where one very large computer generated the proof but every other computer validated it. The total amount of computation required to handle N transactions will then be

$$\underbrace{N \log^c N}_{\text{Prover Time}} + \underbrace{N \log^c N}_{\text{Verifier Time}} \sim O(N \log^c N)$$

However, the total amount of compute power in a community of N individuals is Nk. Therefore the maximal N that can be sustained if the average computational power is k is only

$$N \sim O(\exp(k/c))$$

In moving to zero knowledge proofs we have changed both the technology and how computing power is allocated in the community. While in both the standard blockchain and the sharding example we required that the median participant with k units of computational power is able to serve all the roles, we drop that assumption for the world of zero knowledge proofs. This is a reasonable assumption because the ability to cheaply verify that calculations were done correctly means it is no longer necessary that all individuals conduct all the calculations themselves.

While zero knowledge proofs represent a dramatic step forward in scalability, they are not enough to cover the entire world. With quasilinear zero knowledge proofs, there are two con-

[1] In complexity theory these are known as polylog. These complexity measures are taken from Matter Labs' GitHub page on zero knowledge proofs: `https://github.com/matter-labs/awesome-zero-knowledge-proofs`

straints. First, because generating a proof takes an additional factor of $\log^c N$ calculations, as N gets large the number of calculations to generate a proof of all the transactions will exceed total compute capacity in the economy. After $\exp(k/c)$ even if all the compute power were aggregated into one entity, such an entity would not be able to compute a proof. Second, even though verifying a proof is fast, we still require that *everybody* verify the proof, and verification time grows with the size of the community.

1.4 Linear Zero Knowledge Proofs

If a faster method of generating zero knowledge proofs can be found so that proofs can be generated in $O(N)$ time, then zero knowledge proofs can form the foundation for a blockchain that can scale infinitely. Formally, if a proof verifying N transactions can be generated in $O(N)$ time and verification can occur in $O(\log^c N)$ for some $c \in \mathbb{N}$, the chain can scale to

$$N = O\left(\underbrace{\exp\left(\exp\left(\ldots\exp\left(\frac{k}{c}\right)\right)\right)}_{l \text{ times}}\right)$$

participants for every $l \in \mathbb{N}$. This is effectively infinite scaling.

Linear time provers enable infinite scaling by allowing for a recursive zero knowledge proof, where a tiny fraction of participants actually generate a proof for the transactions, a smaller fraction validate that proof, and then the vast majority of participants simply validate that the validators did so correctly. In this world, the total amount of calculations is

$$\underbrace{N}_{\text{Original Proof}} + \underbrace{\log^c N}_{\text{Proof of Validation}} + \underbrace{N \log^c \log^c N}_{\text{Validation of Validation}} \sim O(N \log^c \log N)$$

If the vast majority of individuals can compute k transactions, then we have that the maximal community is now

$$N \sim O\left(\exp\left(\exp\left(\frac{k}{c}\right)\right)\right)$$

This construction can of course scale to l layers. This means that as more participants join, we can scale to any number of layers to accommodate as many calculations as are required.

2 Discussion

The table below summarizes the results of the model

Setup	Community Size
Standard Blockchain	$O(k)$
Quadratic Sharding	$O\left(k^2\right)$
Quasilinear Zero Knowledge Proofs (available now)	$O\left(\exp\left({}^{k}/_{c}\right)\right), c \in \mathbb{N}$
Linear Time ZKP (not yet feasible)	∞