Congestion, Scaling and the Market for Shards in Blockchains

NICOLA DIMITRI, Department of Economics and Statistics, University of Siena, Siena, Italy

Sharded blockchains have been introduced as a possible solution to scalability issues. In general, the platform decides how to allocate new transactions to the available shards. This is done according to the shards' capacity, their level of occupancy and possible other criteria. In this article, we take an exploration into an alternative perspective, allowing the users to choose which shard to store a transaction into. It follows that such choice would generate a *market* for shards, where users select among available shards considering their performance and transaction fees. In the article, we characterise shards' selection as a Nash Equilibrium of the relevant game. The main finding shows that, when free to choose, users would in general select different shards, therefore scaling the blockchain without main congestions. This is except for when there is a *dominating* shard, that is one which all users prefer regardless of how many users have chosen it. In the article, we discuss possible measures to prevent such dominance.

CCS Concepts: • **Applied computing → Law, social and behavioral sciences**; **Economics**;

Additional Key Words and Phrases: Blockchain, Sharding, Congestion, Nash Equilibrium

ACM Reference format:
Nicola Dimitri. 2025. Congestion, Scaling and the Market for Shards in Blockchains. *Distrib. Ledger Technol.* 5, 1, Article 4 (December 2025), 12 pages.
https://doi.org/10.1145/3716322

1 Introduction

ACM's one of the main desirable features of a blockchain is its scalability, namely the possibility to grow without congesting the platform, which might compromise its correct functioning [Di Maggio, 2024; Halaburda et al., 2022] and discourage its future adoption. Blockchain sharding, that is, the subdivision of the blockchain into multiple sub-chains, has been considered as a possible solution to scalability.

Yet, sharding is not the only solution to scaling and other solutions, such as Layer 2 chains have also been implemented. On this point, Sguanci et al. [2021] compare sharding with Layer 2 solutions, arguing that the former may be more prone to hacking and attacks.

There have been several proposals in the literature on how blockchains can accommodate shards [Alchemy Team, 2022; Dang et al., 2019; Hang et al., 2021; Kokoris-Kogias et al., 2018; Liu et al., 2022a,b; Liu et al., 2023; Luu et al., 2016; Wang and Wang, 2019; Wang et al., 2019; Yu et al., 2020; Zamani et al., 2018], suggesting that there may be a range of alternative solutions. Typically, it is the blockchain platform, which decides how to allocate transactions to shards, although recently alternative allocation criteria have been considered [Chen et al., 2023; Okanami et al., 2020]. That is, based on the shards' capacity, and how many transactions have already been stored in each sub-chain, the platform would decide where to store the next transaction.

This work was supported by NEAR Foundation.
Author's Contact Information: Nicola Dimitri (corresponding author), Department of Economics and Statistics, University of Siena, Siena, Italy; e-mail: dimitri@unisi.it.



This work is licensed under Creative Commons Attribution International 4.0.

© 2025 Copyright held by the owner/author(s).
ACM 2769-6480/2025/12-ART4
https://doi.org/10.1145/3716322

The allocation may also take account of *cross-sharding*, that is of the possibility that a transaction is initially stored in one shard but ends in a different shard. However, whenever possible, the allocation would try to prevent cross-sharding, which in general is more expensive in terms of transaction fees, featuring also longer waiting time to confirm a transaction.

When shards exhibit the same/similar *performance* and *transactions fees* users could be completely indifferent about which shard their transactions will be hosted by. However, if shards would differ in terms of performance, and/or fees, then users may not be indifferent, as they might prefer their transaction to be stored in what they consider to be the most attractive shard, plausibly the one with the best cost–benefit ratio. In this case, blockchain platforms may consider leaving users free to choose their preferred shard and, in so doing, introduce a *market for shards*.

In the article, we discuss how such market may evolve characterising, as Nash Equilibria of the relevant game, the shards chosen by the users. I think the findings could be useful in supporting future decisions that might be taken by platforms when considering changes in the current situation. Although our approach bears some analogies with Okanami [2020] and Chen et al. [2021], to our knowledge, this is the first article discussing such possibility.

To undertake the analysis, we first introduced a *performance* indicator for the shards to measure the benefit induced by a shard, which we proposed to be the number of validated blocks per unit of time in each sub-chain. There could be other indicators to consider as an alternative, or jointly, to the number of validated blocks. For example, the number of detected invalid blocks could also be an informative performance measure. Moreover, broadly speaking, trust and security are also very important when users can choose their shards. However, to keep the exposition simple, without much loss of generality, in most of the analysis, we focused on the number of confirmed blocks, deferring a discussion on other indicators of shard attractiveness in Section 4.

Finally, considering the users' preferences, over performance and transactions fees, we discuss how they could select across alternative shards. A main message of the analysis is that, in general, there would be multiple Nash Equilibria, where users choose different shards, in so doing avoiding main congestions and facilitating the blockchain scaling. For this to take place key, in our model, is that we assume transaction fees to be positively related to the number of a shard's users. That is, the larger such number the higher the fees, due to a stronger competition among users for including one's transaction in such shard. However, under a very specific condition, users may all prefer the same shard. This occurs when a shard is *dominant*, that is, when for each user it generates a utility level which is larger than in any other shard, regardless of how many users have selected it. The presence of a dominant shard may hinder non-congested scaling, even if a strong demand for it would induce higher transaction costs than in alternative shards. This suggests that for a sharded chain, when users can freely choose shards, to solve scalability issues, performance and transaction fees should *balance out* for users over alternative shards.

The article is structured as follows: In Section 2, we discuss the performance indicator. Section 3 introduces the users' preferences and investigates the market for shards. Section 4 elaborates on limitations and future directions of our analysis, while Section 5 concludes the article.

2 Measuring Shards' Performance

In this section, we introduce a basic model of a sharded blockchain and discuss how to measure its performance. Shards are separate blockchains within the same platform. However, they can communicate with each other and are *coordinated* by an additional chain, for instance, in the Near blockchain called *Beacon Chain*, which coordinates the shards' activity and determines the validators for the various blocks in the chain [NEAR, 2021].

2.1 Some Shards Fundamentals

Suppose $s = 1, .., S$ is a generic shard of a blockchain, with S being the number and, to save on notation, also the set of shards in the platform. Clearly, the number of shards could change over time, so that S should be indexed by time. However, to keep the exposition simple, with no major loss of generality, we assume that such

number is constant. Moreover, suppose b is the beacon chain. Therefore, the platform is described by the profile of blockchains (b, S).

Furthermore, let $t \geq 0$ be the time index and suppose $v = 1, .., V_t$ is a generic validator at time t, with V_t being the total number and the set of the platform validators at date t. Therefore, if at time t the number, and the set, of validators in shard s is V_{st}, it is $V_t = \sum_{s=1}^{S} V_{st}$. When validators are uniformly distributed over the S shards, then $V_{st} = \frac{V_t}{S}$ is the number of validators per single shard, as long as V_t can be divided by S. Indeed, if not, a uniform distribution of validators across the shards may not be possible.

Additionally, define as $n_t(s)$, the number of blocks validated up to time t in the sth shard. Moreover, let $t_s \leq t$ be the starting date of the sth shard; hence $n_{t_s}(s) = 0$. Now assume the time interval from t_s to t to be divided into $m \leq t$ subperiods; at the beginning of each subperiod, the shard's validators are assigned again by the platform. In general, this would not prevent some validators, or even all of them, to be re-assigned to the same shard, unless some specific restrictions on this are introduced. In any case, at the start of each subperiod, their allocation is reconsidered.

It follows that the ratio $\sigma_{t_k t_{k+1}}(s)$, defined as

$$\sigma_{t_k t_{k+1}}(s) = \frac{(n_{t_{k+1}}(s) - n_{t_k}(s))}{(t_{k+1} - t_k)} \; with\; k = 0, 1, .., m-1,\; t_0 = t_s \; and \; t_m = t \tag{1}$$

expresses the average number of blocks, over the subperiod from t_k to t_{k+1}, or also the number of blocks per unit of time, validated in the sth shard during the relevant sub-period. In case the subintervals are of the same time length, that is $(t_{k+1} - t_k) = \frac{(t-t_s)}{m}$, then Equation (1) becomes

$$\sigma_{t_k t_{k+1}}(s) = \frac{m(n_{t_{k+1}}(s) - n_{t_k}(s))}{(t - t_s)} \tag{2}$$

From Equation (2), it follows that the average $A(\sigma_{tm}(s))$, of all the $\sigma_{t_k t_{k+1}}(s)$ until t, over all the m subintervals, which is defined as

$$A(\sigma_{tm}(s)) = \left(\frac{1}{m}\right) \sum_{k=0}^{m-1} \sigma_{t_k t_{k+1}}(s) = \left(\frac{1}{m}\right) \sum_{k=0}^{m-1} \frac{m(n_{t_{k+1}}(s) - n_{t_k}(s))}{(t - t_s)} = \frac{n_t(s)}{(t - t_s)} \tag{3}$$

satisfies the following property

$$A(\sigma_{tm}(s)) = \frac{n_t(s)}{(t - t_s)} = A(\sigma_{t1}(s)) \; for\; any\; m = 1, 2, . . \tag{4}$$

That is, in case of uniform subintervals, $A(\sigma_{tm}(s))$ is independent of m and equal to $A(\sigma_{t1}(s)) = \frac{n_t(s)}{(t-t_s)}$, the number of validated blocks per time unit, in shard s, along the entire time interval of length $(t - t_s)$. In case of non-uniform time intervals, Equation (4) ceases to hold, as the following simple numerical example illustrates. Suppose $(t - t_s) = 10$, $n_t(s) = 100$, $m = 2$, $(t_1 - t_s) = 6$, $n_{t_1}(s) = 36$, $n_t(s) - n_{t_1}(s) = 64$. Hence, $A(\sigma_{t1}(s)) = \frac{100}{10} = 10$ while $A(\sigma_{t2}(s)) = \frac{6+16}{2} = 11 > 10$, which may raise the question on what is a proper indicator for the number of blocks validated per unit of time over the period $(t - t_s)$.

!Therefore, to prevent such issue, it is advisable to consider uniform time intervals, at which the set of validators are reconsidered. This would certainly not imply the same value for $\sigma_{t_k t_{k+1}}(s)$ but as in Equation (4), their average, with respect to the number of intervals, will coincide with the number of blocks validated per unit of time over the entire time period.

The ratio $\sigma_{t_k t_{k+1}}(s)$ is an indicator representing the speed with which blocks have been validated in shard s, and it provides a measure of performance, hence desirability, of that shard. Indeed, as we discuss in the next section, if users *would be allowed* to choose across different shards, we think it reasonable to assume that the faster is block validation, the more attractive is a shard for a user.

Based on Equation (1), it is also possible to introduce a related, additional, indicator as the *productivity* of validators V_{st_k} in shard s, between dates t_k and t_{k+1}, as follows

$$\lambda_{t_k t_{k+1}}(s) = \frac{\sigma_{t_k t_{k+1}}(s)}{V_{st_k}} = \frac{(n_{t_{k+1}}(s) - n_{t_k}(s))}{V_{st_k}(t_{k+1} - t_k)} \tag{5}$$

which provides the average number of blocks that, per time unit, each validator included in shard s. Clearly, the larger is Equation (5), the more *productive* have been validators over the relevant time interval.

If Equation (5) could be considered a *local*, between t_k and t_{k+1}, indicator of validators' performance in analogy with Equation (3), it is possible to obtain a global, from t_s to t, indicator of shard s's validators performance as follows

$$A(\lambda_{tm}(s)) = \left(\frac{1}{m}\right) \sum_{k=0}^{m-1} \lambda_{t_k t_{k+1}}(s) = \left(\frac{1}{m}\right) \sum_{k=0}^{m-1} \frac{(n_{t_{k+1}}(s) - n_{t_k}(s))}{V_{st_k}(t_{k+1} - t_k)} \tag{6}$$

The above expression is a global measure of performance for shard s's validators over the time subintervals. However, when validators rotate across such intervals, the performance indicator refers to a "generic" shard's validator, rather than to a "specific" one. Indeed, this could only be possible if $V_{st_k} = V_s$, that is, if shard s had the same set of validators until date t.

In fact, assuming uniform intervals $(t_{k+1} - t_k) = \frac{(t - t_s)}{m}$, and the same set of validators over the entire period, namely $V_{st_k} = V_s$, Equation (6) becomes

$$A(\lambda_{tm}(s)) = \frac{n_t(s)}{V_s(t - t_s)} = A(\lambda_{t1}(s)) \; for \; any \; m = 1, 2, .. \tag{7}$$

which represents the number of blocks confirmed, on average, per unit of time by each shard s's validator until date t.

Unless shard s exhibits some specific features distinguishing s from the other shards, assuming that validators are chosen in a random manner, at each subperiod, as t and m grow, we expect Equation (6) to be about the same across different shards. However, if the committee of validators remains the same over time in each shard, $V_{st} = V_s$, then possible differences in Equation (7) could be imputed to the *ability* of the specific shard's validators. This point will be taken up later when introducing the validators' *reputation*.

Yet, prior to discussing reputation formation, in what follows for completeness, we discuss how validators can be randomly allocated across different shards.

2.2 Validators' Assignment

We envisage the assignment of validators to the shards, in each time sub-interval, as typically done according to a random sampling, without replacement, procedure. Consider the single interval and start taking the simplest case of $V_{st} = V_s = \frac{V}{S}$, with $\frac{V}{S}$ being a positive integer, that is, each shard contains the same number of validators. It follows that there would be $\binom{V}{\frac{V}{S}}$ possible groups of validators, each containing $\frac{V}{S}$ members.

Hence, assume that in each subperiod, the assignment of validators to the S shards is sequential. That is, the first group of $\frac{V}{S}$ selected members is assigned to shard $s = 1$, the second group of $\frac{V}{S}$ selected members to shard $s = 2$ and so on until the Sth group drawn which is assigned to shard $s = S$. It follows that each sequence of S validators' committees, one for shard, can be selected with the following probability

$$\prod_{i=0}^{S-1} \frac{1}{\binom{V - \frac{iV}{S}}{\frac{V}{S}}} \tag{8}$$

For example, with $V = 10$ and $S = 5$, then $\frac{V}{S} = 2$. Therefore, committees will be composed by two members, there will be 45 possible different committee pairs and, according to Equation (8), the probability of selecting a particular sequence of validators groups is

$$\frac{1}{\binom{10}{2}}\frac{1}{\binom{8}{2}}\frac{1}{\binom{6}{2}}\frac{1}{\binom{4}{2}} \sim 0.000009$$

which is a remarkably small value. Of course, this depends on the fact that S is relatively large as compared to V. Indeed, with $S = 2$, the smallest *interesting* number of shards, then the probability of a specific sequence of 2 committees, each with 5 members, would be

$$\frac{1}{\binom{10}{5}} \sim 0.004$$

which is still low, yet much larger than with $S = 5$. At the other extreme, if $S = 10$, that is, one validator per shard, then each validator assignment will have a probability of $\frac{1}{10!} \sim 0.0000003$, which is the smallest possible one.

The above discussion could be immediately extended to committees with a generic number of members V_s. In this case, defining $V_0 = 0$, Equation (8) would become

$$\prod_{i=0}^{S-1} \frac{1}{\binom{V-\sum_{s=0}^{i} V_s}{V_{i+1}}} \tag{9}$$

It follows that in m independent random samples, to define the set of validators across the shards in m subperiods, the probability of drawing exactly the same validators, in each shard, is given by

$$\left(\prod_{i=0}^{S-1} \frac{1}{\binom{V-\sum_{s=0}^{i} V_s}{V_{i+1}}}\right)^m \tag{10}$$

which is clearly smaller than Equation (9) and decreasing with m.

To conclude, it is also interesting to compute the probability that a specified, *single, validator* will be included in the 1st, 2nd, ..., Sth shard. It is immediate to see that the probability to be selected for the first shard is $\frac{V_1}{V}$, for the second shard is $\frac{(V-V_1)}{V}\frac{V_2}{(V-V_1)} = \frac{V_2}{V}$ and so on, with the probability to be part of the Sth shard being $\frac{V_S}{V}$. Such probabilities will be used below when discussing the validators' reputation.

Indeed, in the next paragraph, we are going to investigate how the indicators proposed in Section 2.1, taken jointly with Section 2.2, could be used to build up a validator's reputation index.

2.3 Validators' Reputation

The above indicators could be taken to construct a reputation index, for the generic validator v, in the following way. Let $s_{t_k v}$ be the shard which validator v was assigned to during the subperiod between t_k and t_{k+1}. Then, we can define her reputation, performance, indicator at date t as follows:

$$A(r_t(v)) = \frac{1}{m}\sum_{k=0}^{m-1} \lambda_{t_k,t_{k+1}}(s_{t_k v}) \tag{11}$$

that is, the average of all the $\lambda_{t_k,t_{k+1}}(s_{t_k v})$ indicators, representing the global performance that, on average, validator v has had until time t. Hence, for example, if $m = 2$, $\lambda_{t_0,t_1}(s_{t_0 v}) = 0.3$ and $\lambda_{t_1,t}(s_{t v}) = 0.2$, then this means that validator v has confirmed, on the overall, 0.3 blocks per unit of time in the first subperiod and 0.2 in the second subperiod. Therefore, it seems natural to consider the number $A(r_t(v)) = \frac{0.3+0.2}{2} = 0.25$, as representing a reasonable, overall, performance index for v.

Clearly, Equations (6) and (11) are connected since the validators' performance are based on the shards' performance, and the contrary. For this reason, as t and m get sufficiently large, Equation (11) tends to take the following approximate equation

$$A(r_t(v)) \sim \sum_{s=1}^{S} A(\lambda_{tm}(s)) \left(\frac{V_s}{V}\right) \tag{12}$$

that is, validator $v's$ reputation index tends to become a weighted average of the shards' performance indexes, where the weights are given by the relative frequency with which a validator is assigned to the different shards. In the *long run*, such frequencies tend to correspond to the probabilities $\frac{V_s}{V}$ with which a validator is assigned to a shard in a single draw.

Finally, notice that if the number of shards and validators would change over time, Equation (12) may cease to hold. In the following section, we investigate the market for shards, where their attractiveness will be measured by their performance indicator only.

3 The Market for Shards

The above discussion laid the ground for considering the possibility that users may choose across different shards based on their performance indicators. As said, in practice, the possibility for a user to choose a shard seems to be limited. Yet, from an economic perspective, we find it interesting to investigate how possible differences across shards' performance may affect users' demand for implementing transactions in some specific shards. Based on this, we will also discuss how such possible *market for shards* may develop over time and, moreover, if scaling could evolve without congestions.

We begin the analysis introducing the shards' performance and transaction fees, paid by the users to implement transactions and smart contracts, as an argument of the users' preferences. Consider a generic user and assume, insofar as the shard's choice is concerned, that her utility function is defined as

$$U(A(\lambda_{tm}(s)), c_s(u_s)) \tag{13}$$

that is depends on two arguments. The first one is $A(\lambda_{tm}(s))$, while the other is the unitary transaction fee, cost, function $c_s(u_s)$ that users have to pay to implement a transaction in shard s, where the number of users is u_s. We explicitly define the cost expression $c_s(u_s)$ as a function, typically increasing, of u_s. In general, we assume Equation (13) to be increasing in $A(\lambda_{tm}(s))$ and decreasing in $c_s(u_s)$, hence also in u_s.

There could be several utility functions satisfying the above assumptions; one possible simple form is the quasi-linear one

$$U(A(\lambda_{tm}(s)), c_s(u_s)) = f(A(\lambda_{tm}(s))) - c_s(u_s) \tag{14}$$

where f is an increasing, and weakly concave, function of $A(\lambda_{tm}(s))$. Therefore, should this be possible, a user would choose the shard s, which maximises Equation (14).

However, the above considerations immediately suggest that if users have similar preferences, then most, if not all, of them might try to select the same shard. Such behaviour, of course, may cause congestion problems on that shard, possibly raising the scalability issue again. This unless, as to be expected, the performance indicator and/or the costs would adjust to changes in demand for shards, in such a way to induce users to select different shards. Below, we shall see that since, in Equation (14), the cost function depends on the number of adopters, it is the level $c_s(u_s)$ that will change according to the number of users.

Indeed, broadly speaking, in a blockchain where users are free to choose their preferred shards, congestions may be prevented, and scalability facilitated, if performance and usage costs are such that users would find it convenient to spread their transactions across different shards. A sufficient, although not necessary, condition for this is when all shards exhibit the same performance and costs. However, in the next section, we are going to see

why performance and transaction fees may not necessarily be the same across shards, discussing the simple case of homogeneous preferences.

3.1 Homogeneous Preferences

Consider the following very simple case. Suppose Equation (14) is the same for all the users, with $f(A(\lambda_{tm}(s))) = A(\lambda_{tm}(s))$. Hence,

$$U(A(\lambda_{tm}(s)), c_s(u_s)) = A(\lambda_{tm}(s)) - c_s(u_s) \tag{15}$$

From Equation (15), it follows that the same utility level, say $k \geq 0$, could be obtained by any pair $A(\lambda_{tm}(s))$, $c_s(u_s)$ satisfying the equation $A(\lambda_{tm}(s)) = k + c_s(u_s)$. Therefore, since in principle there is an infinite number of such pairs, we deduce that there may be several alternative shards, with different performances and transaction fees, inducing the same level of utility, satisfaction, across the users. This could make users *indifferent* over alternative shards and willing to be randomly assigned to any one of them. However, whether or not such indifference would take place depends on several elements.

The following, simple, example illustrates how this might happen. Consider a generic date t and that, for simplicity, each user wants to introduce a single transaction in the next block, of the most preferred shard. Moreover, suppose $S = 2$, $A_1 = A(\lambda_{tm}(s = 1)) > A(\lambda_{tm}(s = 2)) = A_2$, $c_1(u_1) = a_1u_1$, $c_2(u_2) = a_2u_2$ where $a_s > 0$, u is the total number of users in the platform, with $i = 1, .., u$ being the generic user, u_1 the number of users adopting shard $s = 1$ and $u_2 = u - u_1$ the number of users choosing shard $s = 2$. Therefore, under our assumption, the number of users corresponds to the number of transactions to be introduced into the blockchain.

The definition of $c_1(u_1)$ and $c_2(u_2)$ reflects the intuitive idea that as the number of a shard's users increases, the unit cost of a transaction fee increases proportionally, due to the increased level of competition to prioritise transactions in blocks with limited space. In fact, if the block size would adjust to the demand size, the unit cost for storing transactions may not change. Finally, observe also that alternative costs functions could depend on the number of bytes required to store a transaction in a block, rather than simply on the number of users demanding that shard.

Assume that all the individuals have the same preferences, as in Equation (15). In what follows we are going to characterise the number of users, adopting the two shards, as a pure strategy Nash Equilibrium of the underlying game. Indeed, for a single individual, choosing a shard is a strategic decision because the final payoff, utility level, that she is going to obtain from her choice depends not only on the shard that she selects but also on the shards chosen by the other users. To keep the example simple, in what follows we assume that each shard has sufficient capacity to contain all the users and the costs are mostly due to waiting time, which may increase when a shard gets congested.

Hence, suppose $\rho = (s_1, s_2, ., s_u)$, with $s_i \in \{1, 2\}$ is the profile of shards chosen by the u users at a given date. Then, the following result holds

Proposition 1. *Suppose that*:

(i) $A_1 - a_1u > A_2 - a_2$; *then there exists a unique pure strategy Nash Equilibrium in which all users choose* $s = 1$.

(ii) $A_1 - a_1(u+1-j) < A_2 - a_2 j$ *for* $j = 1, 2, .., u_2$, *with* $u_2 = 1, .., u-1$, *and* $A_1 - a_1(u+1-j) > A_2 - a_2 j$, *for* $j = (u_2 + 1)$. *Then, there exist* $\frac{u!}{u_1!u_2!}$ *pure strategy Nash Equilibria profiles* ρ *in which* u_1 *users choose shard* $s = 1$ *and* u_2 *users choose shard* $s = 2$.

(iii) $A_1 - a_1 < A_2 - a_2u$; *then there exists a unique pure strategy Nash Equilibrium in which all users choose* $s = 2$.

Proof.

(i) Suppose $A_1 - a_1u > A_2 - a_2$. Then, it follows immediately that $A_1 - a_1(u - j) > A_2 - a_2(j + 1)$, with $j = 1, ., (u - 1)$. Now consider user $i = 1$; against all the other users choosing $s = 1$ her best reply would

be to select $s = 1$ too. The same reasoning holds also for any $i > 1$. Additionally, it is simple to verify uniqueness. Indeed, for each user, it can be checked that strategy $s = 1$, in this case, is strictly dominant which implies that no strategy profile, where at least one user selects $s = 2$, could be a Nash Equilibrium.

(ii) Suppose $u_2 = 1$; then $A_1 - a_1 u < A_2 - a_2$ and $A_1 - a_1(u-1) > A_2 - a_2 2$. We now show that the profile $\rho = (s_1 = 2, s_2 = 1, .., s_u = 1)$ is a Nash Equilibrium. Consider user $i = 1$ who expects all the other users to choose $s = 1$; then, if $i = 1$ chooses $s = 1$ her payoff will be $A_1 - a_1 u$ while choosing $s = 2$ her payoff will be $A_2 - a_2$ and since $A_1 - a_1 u < A_2 - a_2$ her best reply is to select $s = 2$. Consider now user $i = 2$. If she expects user $i = 1$ to choose $s = 2$, while all the others choose $s = 1$, in selecting $s = 1$ her payoff would be $A_1 - a_1(u-1)$ while selecting shard $s = 2$ she would obtain a payoff of $A_2 - a_2 2$. However, since $A_1 - a_1(u-1) > A_2 - a_2 2$ choosing $s = 1$ would be user $i = 2's$ best reply to the other users' choice. The same argument applies to show that any profile ρ, such that only one user chooses $s = 2$, while the others choose $s = 1$, is a pure strategy Nash Equilibrium of the game. It is also immediate to check that the total number of such equilibria would be equal to $\frac{u!}{1!(u-1)!} = u$.

Finally, since $A_1 - a_1(u-1) > A_2 - a_2 2$ implies that $A_1 - a_1(u-l) > A_2 - a_2(l+1)$, for all $l = 2, .., u-1$, it follows that with $j = 1$ the above u strategy profiles, with just one user selecting $s = 1$, are the only Nash Equilibria of the game.

In fact, for example, in this case the profile $\rho' = (s_1 = 2, s_2 = 2, s_3 = 1, ., s_u = 1)$ could not be a Nash Equilibrium for the following reason. Consider user $i = 1$; if she chooses $s_2 = 2$ she would obtain as payoff $A_2 - a_2 2$ while selecting $s = 1$ would provide a payoff of $A_1 - a_1(u-1)$, and since $A_1 - a_1(u-1) > A_2 - a_2 2$ her best reply would be $s = 1$ and not $s = 2$, which prevents ρ' from being a Nash Equilibrium. This is true for all such profiles with two users choosing $s = 2$, but the reasoning immediately extends to any other strategy profile with at least two users selecting $s = 2$.

Consider now any other $u_2 \geq 2$; by an argument similar to the one for $u_2 = 1$, the result follows.

(iii) The proof is analogous to (i)

As for the above result, some comments are in order. First observe that the performance indicator and the transaction fees are key to determine how many users, in equilibrium, will choose the two shards.

A unanimous choice, namely all users selecting either $s = 1$ or $s = 2$, can be the case when one shard *dominates* the other, namely $A_1 - a_1 u > A_2 - a_2$ or $A_1 - a_1 < A_2 - a_2 u$. That is when, even if all the users choose the same shard, and so the transaction costs would be maximum, the performance indicator is sufficiently large to more than compensate for the high transaction fees. Of course, such two equilibria with unanimous choice of the shards could be possible only if the platform does not introduce an upper limit to the number of users lower than u and/or increases the costs of the, other, dominating shard. In this case, based on the platform rules, some of the users will have to select the other shard, even if for them would be sub-optimal and/or less performing.

In the remaining Nash Equilibria users will be distributed across the two shards, in a way that is again due to the performance and transaction fees' structure. More specifically, the number of users choosing $s = 1$ will be determined by the number of users for which the payoff in shard $s = 1$ is larger than in $s = 2$, and the same for the other shard. Furthermore, the payoffs of the users in the two shards is likely to be approximately the same. Indeed, for example, we have seen that with $u_2 = 1$, it is $A_1 - a_1 u < A_2 - a_2$ and $A_1 - a_1(u-1) > A_2 - a_2 2$ which suggests that $A_1 - a_1(u-1)$ may not be too different from $A_2 - a_2$. Similar considerations hold also for the other equilibria.

Before concluding, for completeness, it is worth observing the following point. In the above proposition rather than, for example, $A_1 - a_1 u > A_2 - a_2$ suppose that $A_1 - a_1 u = A_2 - a_2$. In this case, it is easy to check that the profile where all users choose $s = 1$ is still a Nash Equilibrium, but it is not unique. Indeed, there are u profiles, where $(u-1)$ users choose $s = 1$ and one of them selects $s = 2$, which are also equilibria. That is, the equality in

the payoff, obtained by introducing indifference between the two shards, induces a larger number of equilibria compared to strict inequality.

In this case, it would be $A_1 - a_1(u-1) > A_2 - a_2 2$, which however provide no additional equilibria to the previous ones. Therefore, an equilibrium where $(u-1)$ users choose $s=1$ and only one selects $s=2$ could be generated by two different payoff structures. Similar considerations apply to other equilibria.

To conclude, despite the model being simple below we shall see that its insights are in fact much more general, and extend to a large class of models. Indeed, the main message which remains valid is that the shards' performance and transaction fees functions will be driving the distribution of users across the shards in a way that, at a Nash Equilibrium, a user's payoff may be approximately equal across alternative shards, except for the case of *dominance*. Indeed, as we discuss in the next section, the same conclusion can be obtained with any type of heterogenous preferences, and any number of shards, as long as the utility functions, are increasing in the performance and decreasing in the number of users adopting the same shard.

3.2 The General Model

In this section, we extend the above finding to any number of shards and any types of preferences, as long as the utility function increases with a shard's performance indicator and decreases with its number of users. More specifically, suppose now there are S shards, with $S = 2, 3, ..,$ and that for the generic agent i the utility function is

$$U_i(A_s, c_s(u_s)) \text{ with } i = 1, ., u \tag{16}$$

which we assume increasing A_s, decreasing in $c_s(u_s)$, and where $c_s(u_s)$ is increasing in u_s, so that Equation (16) would be decreasing also in u_s.

Proposition 2. *Suppose that at a Nash Equilibrium the number of users adopting the S shards is $(u_1, u_2, ., u_S)$, with $u_s \geq 0$ and $u_1 + u_2 + . + u_S = u$. Then if user i, in equilibrium, chooses shard s_i, it must be*

$$U_i(A_{s_i}, c_{s_i}(u_{s_i} + 1)) < U_i(A_s, c_s(u_s))$$

and

$$U_i(A_{s_i}, c_{s_i}(u_{s_i})) \geq U_i(A_s, c_s(u_s + 1)) \, for \, all \, i \, and \, all \, s \neq s_i.$$

Proof. It is an immediate extension of the proof for Proposition 1. □

As a simple illustration of the above proposition, consider now three shards, that is, $s = 1, 2, 3$, where A_s are their performance indicators and $c_s(u_s) = a_s u_s$, their costs. Moreover, consider also two types of users, characterised by different utility functions. In particular, assume that a share $0 < \alpha < 1$ of the u users have preferences defined as in Equation (15), whom we shall call Type I users, while the remaining $0 < 1 - \alpha < 1$ share of users, Type II, have the following utility function

$$U(A(\lambda_{tm}(s)), c_s(u_s)) = \frac{A(\lambda_{tm}(s))}{c_s(u_s)} \tag{17}$$

Notice that, as well as Equation (15), also Equation (17) is increasing in $A(\lambda_{tm}(s))$ and decreasing in $c_s(u_s)$. To start with suppose $A_1 - a_1 u > A_s - a_s$ and $\frac{A_1}{a_1 u} > \frac{A_s}{a_s}$, for $s = 2, 3$, then the only pure strategy Nash Equilibrium of the game would be $\rho = (s_1 = 1, s_2 = 1., s_u = 1)$ since every user, regardless of her preferences, finds it convenient to choose $s = 1$ when all the other users are choosing it too.

Suppose now $A_1 - a_1 u > A_s - a_s$ with $s = 2, 3$ while $\frac{A_1}{a_1 u} > \frac{A_2}{a_2}$, $\frac{A_1}{a_1 u} < \frac{A_3}{a_3}$ and $\frac{A_1}{a_1(u-1)} > \frac{A_3}{a_3 2}$. Since, as we saw, $A_1 - a_1 u > A_s - a_s$ implies $A_1 - a_1(u-1) > A_s - a_s 2$ for $s = 2, 3$, it follows that there would be $(1-\alpha)u$ Nash Equilibria where all Type I users select $s = 1$, one Type II user chooses $s = 3$ and the rest $s = 1$, while none chooses $s = 2$. All type I users will obtain the same utility level, while type II users approximately the same.

Analogous considerations would hold for other Nash Equilibria.

4 Limitations and Future Directions

As mentioned in previous sections, the analysis presents a number of limitations, which we now discuss as a basis for future developments.

The first aspect to focus upon is the main point of the article, namely that in real blockchains, users typically do not have the possibility to choose among shards. For example, the NEAR platform assigns the shard to a transaction [NEAR, 2021] and regularly *rebalances* the shards, reallocating blocks and smart contracts across them, in such a way that computational and storage loads would be similar across shards, avoiding congestions and delays. Therefore, such rebalancing operates *as if* a dominant strategy, when users are free to choose as in our article, could not emerge.

Hence, once more, this article should not be taken as modelling the current situation in sharded blockchains rather as a possible, future, scenario where users could be allowed to freely choose the shard where to store their transactions. For this reason, the article cannot be empirically validated at the moment since, to the best of our knowledge, so far no blockchain has adopted a free market for shards and so data are missing.

Additionally, the utility function that we introduced in Equation (13) exhibits some limitations, since it hinges only on shards/validators' reputation as represented by the number of blocks stored in the shard. Moreover, users are assumed to be rational and perfectly informed, which is leaving out several, potentially important, elements to understand and predict users' behaviour. For example, issues related to cognitive and psychological biases, security and trust [Azzedin, 2010; Azzedin, 2014; Bao et al., 2019; Bellini et al., 2020] can play a role in deciding which shard to choose for users. Indeed, the main goal of our work is to suggest possible insights on how a free market for shards could establish.

For example, defining $t_s(u_s) \geq 0$ as a numerical indicator of *trust* for each shard s, non-decreasing with u_s, then Equation (15) could be reformulated as

$$U(A(\lambda_{tm}(s)), t_s(u_s),\ c_s(u_s)) = A(\lambda_{tm}(s)) + t_s(u_s) - c_s(u_s) \tag{18}$$

which would imply that trust can perfectly replace reputation for that user. If instead

$$U(A(\lambda_{tm}(s)), t_s(u_s),\ c_s(u_s)) = A(\lambda_{tm}(s))t_s(u_s) - c_s(u_s) \tag{19}$$

then the user would consider reputation and trust as complementary as well, and not only as substitutes. Likewise, if $\varphi_s(u_s) \geq 0$ is a numerical indicator of security for shard s, also non-decreasing with u_s, then Equation (18) could become, for example,

$$U(A(\lambda_{tm}(s)), t_s(u_s),\ c_s(u_s)) = \varphi_s(u_s)[A(\lambda_{tm}(s)) + t_s(u_s)] - c_s(u_s) \tag{20}$$

which may imply that there is a minimum, acceptable, level of security φ_s^* for shard s to be chosen.

As a result of the above extensions, shard dominance, based only on $A(\lambda_{tm}(s))$ and $c_s(u_s)$, may be less likely when trust and security are taken into account. Indeed, while $A(\lambda_{tm}(s))$ is the same for all users, $\varphi_s(u_s)$ and $t_s(u_s)$ may differ across users, depending upon which indicators they consider as meaningful.

Furthermore, in the above discussion, we provided no policy suggestions on how to prevent the emergence of dominant shards as the above findings, although based on a very simple framework, may suggest some initial insights for the platform on how to prevent the emergence of a dominant shard. Indeed, in case a shard would dominate the others, accommodating all the users request, the ultimate goal of scaling through shards, would be missed. For instance, one simple policy, in the two shards model, could be to make the unit costs such that $a_1 > a_2$ for $A_1 > A_2$. In general, the platform could set up higher unit costs for better performing shards and lower costs for the less performing ones. Such a policy should be dynamic, adjusting the costs over time as performances change. By doing so it would prevent the more attractive shards to get congested. Alternative criteria could increase the unit costs with the number of users in that shard.

Finally, in the above discussion, we did not consider how a market for shards could affect validators' behaviour. If paid by the number of blocks, and if validators would not be rotated across shards, then validators should

be interested in attracting as many users as possible in the shards where they operate. Therefore, dominance may emerge if some shard overperforms the others, unless its cost would increase in the way we discussed above. If validators rotate across shards, we may expect performance in terms of validated blocks to be similar over different shards and so, again, costs may be adjusted by the platform to prevent their discrepancies and dominance.

5 Conclusions

Sharded chains can be a solution to the platforms' scalability problem, by avoiding blockchain congestions and long delays in approving transactions, which may discourage future usage of the chain. The article intends to be a contribution to discuss how users of a sharded blockchain like NEAR would choose across different shards, in case it would be possible for them to do so. The discussion provides early insights as to whether congestion and scalability will be solved spontaneously by the users, that is, whether they would find it convenient to validate their transactions on different shards. The analysis suggests that this might be the case if their payoffs do not differ *too much* across alternative chosen shards, so that they could obtain approximately equal benefits. Such possibility can realise, even with homogeneous preferences and differently performing shards, as long as transactions fees would depend on the number of users willing to select a shard. This could indeed be an element discouraging users to concentrate their demand in the same shard, since an increase in demand for a specific shard would spur competition for validating transactions, and an increase in their fees.

Despite the simplicity of the models discussed, we believe the analysis delivers some interesting insights on how users may behave in a sharded blockchain, when they are free to choose shards.

Acknowledgements

I would like to thank the NEAR Foundation for financial support of an earlier version of this article. I'm also grateful to the Editors and the anonymous reviewers for constructive comments, which much improved earlier versions of the article.

References

Alchemy Team. 2022. Ethereum Sharding: An Introduction to Blockchain Sharding. Retrieved from https://www.alchemy.com/overviews/ethereum-sharding-an-introduction-to-blockchain-sharding

F. Azzedin. 2010. Trust-based taxonomy for free riders in distributed multimedia systems. In *Proceedings of the 2010 International Conference on High Performance Computing and Simulation*. IEEE, 362–369.

F. Azzedin. 2014. Taxonomy of reputation assessment in peer-to-peer systems and analysis of their data retrieval. *The Knowledge Engineering Review* 29 (2014), 463–483.

X. Bao, C. Su, Y. Xiong, W. Huang, and Y. Hu. 2019. Flchain: A blockchain for auditable federated learning with trust and incentive. In *Proceedings of the 5th International Conference on Big Data Computing and Communications (BIGCOM)*. IEEE, 151–159.

E. Bellini, Y. Iraqi, and E. Damiani. 2020. Blockchain-based distributed trust and reputation management systems: A survey. *IEEE Access* 28 (2020), 21127–21151.

C. Chen, Q. Ma, X. Chen, and J. Huang. 2021. User distributions in shard-based blockchain network: Queueing modeling, game analysis, and protocol design. In *Proceedings of the 22nd International Symposium on Theory, Algorithmic Foundations, and Protocol Design for Mobile Networks and Mobile Computing (MobiHoc '21)*. ACM, New York, NY, USA, 10 pages. DOI: https://doi.org/10.1145/3466772.3467051

H. Dang, T. Dinh, D. Loghin, E. Chang, Q. Lin, and B. Ooi. 2019. Towards scaling blockchain systems via sharding. In *Proceedings of the International Conference on Management of Data*, 123–140.

M. Di Maggio. 2024. *Blockchain, Crypto and DeFi: Bridging Finance and Technology*. Wiley.

H. Halaburda, M. Sarvary, and G. Haeringer. 2022. *Beyond Bitcoin* (2nd ed). Palgrave-McMillan.

R. Hang, J. Yu, H. Lin, S. Chen, and P. Esteves-Verissimo. 2021. On the security and performance of blockchain sharding. *Crytpology ePrint Archive*, 1276.

E. Kokoris-Kogias, P. Jovanovic, L. Gasser, N. Gailly, E. Syta, and B. Ford. 2018. OmniLedger: A secure, scale-out, decentralized ledger via sharding. In *Proceedings of the IEEE Symposium on Security and Privacy*, 583–598.

Y. Liu, J. Liu, M. Vaz Salles, Z. Zhang, T. Li, B. Hu, F. Henglein, and R. Lu. 2022a. Building blocks of sharding blockchain systems: Concepts, approaches, and open problems. *Computer Science Review* 46 (2022), 100513.

Y. Liu, J. Liu, Q. Wu, H. Yu, Y. Hei, and Z. Zhou. 2022b. SSHC: A secure and scalable hybrid consensus protocol for sharding blockchains with a formal security framework Y. *IEEE Transactions on Dependable and Secure Computing* 19, 3 (2022), 2070–2088.

Y. Liu, X. Xing, H. Cheng, D. Li, Z. Guan, J. Liu, and Q. Wu. 2023. A flexible sharding blockchain protocol based on cross-shard byzantine fault tolerance. *IEEE Transactions on Information Forensics and Security* 18 (2023), 2276–2291.

L. Luu, V. Narayanan, C. Zheng, K. Baweja, S. Gilbert, and P. Saxena. 2016. A secure sharding protocol for open blockchains. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security (CCS '16)*, 17–30.

NEAR. 2021. White Paper. Retrieved from https://pages.near.org/papers/the-official-near-white-paper/#how-near-works

N. Okanami, R. Nakamura, and T. Nishide. 2020. Load balancing for sharded blockchains. In *Proceedings of the Financial Cryptography and Data Security (FC '20)*. M. Bernhard, et al. (Eds.), Lecture Notes in Computer Science, Vol. 12063, Springer, Cham.

C. Sguanci, R. Spatafora, and A. Vergani. 2021. Layer 2 blockchain scaling: A survey. arXiv:2107.10881. Retrieved from https://arxiv.org/abs/2107.10881

G. Wang, J. Shi, M. Nixon, and S. Han. 2019. SoK: Sharding in blockchains. In *Proceedings of the 1st ACM Conference on Advances in Financial Technologies*, 41–61.

J. Wang and H. Wang. 2019. Monoxide: Scale out blockchain with asyncronous consensus zones. In *Proceedings of the 16th USENIX Conference on Networked Systems Design and Implementation (NSDI '19)*, 95–112.

G. Yu, X. Wang, K. Yu, W. Ni, J. Zhang, and R. Liu. 2020. Survey: Sharding in blockchains. *IEEE Access* 8 (2020), 14155–14181.

M. Zamani, M. Movahedi, and M. Raykov. 2018. RapidChain: Scaling blockchain via full sharding. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security (CCS '18)*, 931–848.

Received 21 August 2023; revised 14 January 2025; accepted 24 January 2025