The Sui Smart Contracts Platform: Economics and Incentives

The Mysten Labs Team
econ@mystenlabs.com

May 2022

1 Introduction

The Sui Smart Contracts Platform is an environmentally-friendly, cost-efficient, high-throughput, and low-latency permissionless blockchain. Sui's capabilities are well beyond the frontier of existing blockchain systems. Recent tests show that an unoptimized single-worker Sui validator running on an 8-core M1 Macbook Pro can execute and commit 120,000 token transfer transactions per second (TPS).[1] This achievement augurs the genesis of a platform that can meet the Herculean requirements needed to serve billions of users across a wide range of web3 applications.

Sui's cutting-edge performance is enabled by major advancements in the fields of distributed systems, cryptography, and programming languages. In the same spirit, the Sui economy has been designed at the frontier of blockchain economic and incentives research. The overarching goal has been to implement an economic system aligning incentives across the various entities participating in the Sui ecosystem. The aim is for Sui's financial plumbing to be at par with its engineering design so that Sui delivers a flourishing economy with billions of participants. This tokenomics paper describes the core elements of the Sui economy. We refer the reader to the "Sui Smart Contracts Platform" white paper at https://sui.io/whitepaper for details on Sui's technical design and recommend both papers be read in parallel.

The Sui economy is characterized by three sets of entities:

- Users submit transactions to the Sui platform in order to create, mutate, and transfer digital assets or interact with more sophisticated applications enabled by smart contracts, interoperability, and composability.

[1] A full performance report will be published when Sui's testnet is released.

- Owners of Sui's native asset – called SUI and available in finite supply – bear the option of delegating their holdings to validators and participating in the proof-of-stake mechanism. SUI owners also hold the rights to participate in Sui's governance.
- validators manage transaction processing and execution on the Sui platform.

The Sui economy has four core components:

- The SUI token is the Sui platform's native asset and provides on-chain liquidity for the Sui economy.
- Gas fees are charged on all network operations and used to reward participants of the proof-of-stake mechanism and prevent spam and denial-of-service attacks.
- The proof-of-stake mechanism is used to select, incentivize, and reward honest behavior by the Sui platform's operators – i.e. validators and the SUI delegators.
- On-chain governance is used to modify and improve the functioning of the Sui protocol across time.

Sui's main economic novelty is its specific implementation of the proof-of-stake mechanism – a direct consequence of Sui's object-centric design. Sui objects can encode any type of asset – including fungible and non-fungible tokens – and determine Sui's global state.[2] Transactions on Sui take objects as inputs and deliver new objects as outputs.

Conditional on the object-centric design, Sui processes and executes transactions using a causal-ordering approach. Loosely speaking, if the objects in two transactions are fully independent, then it matters not which transaction is processed first. Indeed, it will often be the case that some validators process one transaction first and some validators the other.[3] This design is remarkably powerful because it lets Sui parallelize transaction processing of non-shared objects – i.e. objects owned by a single address (see section 2.4). Hence, each validator can scale itself horizontally and increase its transaction throughput by adding more computing power. This delivers highly efficient unit economics in that both throughput and costs scale linearly with network activity on independent data. Sui's "multi-lane" design contrasts markedly with traditional blockchain designs relying on total ordering – where every single transaction is ordered relative to each other, even fully independent transactions.

Sui's proof-of-stake mechanism leverages the causal ordering approach such that processing and executing each transaction requires a quorum of 2/3's of the validators by stake. Validators participate passively by receiving incoming transactions, validating their authenticity, and sending back signatures to the users. Hence, Sui both processes

[2]Sui objects are not limited to digital assets, but can also encode smart contracts and the Move packages used to create and manage other objects. For the purposes of this paper, objects will be treated as if they were synonymous with digital assets. See the Sui white paper for a detailed discussion of Sui objects.

[3]Causal ordering does require some degree of ordering, however. For example, if transaction B utilizes transaction A's output objects as inputs then transaction B must necessarily follow transaction A. Ordering is also important in the case of shared objects – where objects are owned by multiple addresses.

different transactions in parallel and enables validators to process transactions shortly after submission. Since Sui is leaderless and every validator has an equal role in validation and execution, all honest validators reap the benefits and obtain stake rewards according to their share of delegated stake. Consequently, Sui requires no wasteful computation and avoids the "rich-get-richer" forces present in other proof-of-stake implementations where high-stake validators are more likely to obtain protocol rewards.

Sui's gas pricing mechanism achieves the triple outcomes of delivering users with low, predictable transaction fees, of incentivizing validators to optimize their transaction processing operations, and of preventing denial of service attacks. Importantly, a unique feature of Sui's gas mechanism is that Sui users pay separate fees for execution and storage. Execution or, computation gas prices, are determined thorough a three-step process operating repeatedly across Sui epochs (time is divided into consecutive periods lasting roughly 24 hours each):

1. A gas price survey asks validators to submit reservation prices at the epoch start – that is, the minimum gas price at which they are willing to process transactions. The protocol sets the 2/3's percentile by stake as the epoch's reference gas price.

2. As the epoch progresses with users submitting transactions and validators processing them, validators obtain signals over the operations of other validators.

3. At the epoch close, each validator submits their (subjective) beliefs over every other validator's behavior and this information is used as an input into the stake reward distribution rule. Validators who submitted low price quotes during the gas survey – namely, lower than the reference price – or who processed all transactions above their self-declared reservation price promptly get boosted rewards. Contrarily, validators who submit high price quotes during the gas survey or who do not honor their self-declared reservation price get penalized with discounted rewards.

Sui's gas pricing mechanism provides end users with good user experience and creates the incentives for validators to operate sustainable business models. On the user side, Sui does not require gas prices to be above or below the reference gas price – indeed, users are free to submit any gas price. However, the gas pricing mechanism is designed so that validators are incentivized to both elicit their true reservation gas price and to honor such prices. Consequently, Sui users can expect transactions submitted with gas prices close to or at the reference price to be processed promptly. Sui users thus avoid the inefficiencies of having to forecast the current gas price and overpaying as a result. On the validator side, a quorum of validators should always be able to operate with healthy gross margins since they collectively decide the reference gas price. Moreover, since the most efficient validators receive boosted rewards, Sui's gas mechanism includes incentives to avoid cartel-like behavior at the time of price setting. In sum, Sui's gas price mechanism creates a healthy competition for fair prices: Validators are incentivized to set low gas prices but not too low – lest they be penalized for failing to honor such prices.

Sui's gas pricing mechanism bestows Sui users with an important monitoring role. On the one hand, users want their transactions to be processed as quickly and efficiently as possible. User clients such as wallets encourage this by prioritizing communication with the most responsive validators. Such efficient operations are compensated with boosted rewards relative to less responsive validators. On the other hand, SUI token delegators receive the same boosted or penalized rewards as their delegate validator. An unresponsive validator is thus doubly exposed to the gas pricing mechanism: they lose directly through slashed rewards and indirectly through reduced delegated stake in future epochs as stakers move their tokens to more responsive validators.

Sui also includes an efficient and sustainable economic mechanism for pricing data storage. Beyond Sui's high throughput and low latency, a key Sui feature is its ability to handle arbitrary amounts of on-chain data. Financially, this feature introduces a severe intertemporal challenge: Validators who process and write data into storage today may differ from the future validators needing to store that data. If users were to pay only the fees for computation power at write, effectively, future users would need to subsidize past users for their storage and pay disproportionately high fees. This negative network externality can become highly taxing for Sui in the long-run if left unaddressed.

Sui's economic design includes a storage fund that redistributes past transaction fees to future validators. In a nutshell, users pay fees upfront for both computation and storage. The storage fees are deposited into a storage fund used to adjust the share of stake rewards distributed to validators relative to SUI delegators. When on-chain storage requirements are high, validators receive substantial additional rewards in order to compensate their costs. Vice versa when storage requirements are low. Importantly, the storage fund never distributes rewards directly out of its principal, thus providing an economic mechanism that is viable in the long-run and that can fund storage costs indefinitely.

The storage fund introduces various desirable incentives into the Sui economy. First, it includes a "deletion option" by which users obtain a storage fee rebate whenever they delete previously stored on-chain data.[4] This introduces a useful self-regulating throttle mechanism by which users delete data whenever storage no longer makes sense financially. Second, because the Sui storage fund is denominated in SUI, increased activity leads to larger storage requirements and to more SUI removed from circulation. The storage fund thus kills two birds with one stone: it delivers a financially viable storage model and also creates deflationary pressure on SUI – benefitting the network's owners and users. Third, the storage fund is capital efficient in that it is economically equivalent to a rent model where users pay for storage through a pay-per-period model. The storage fund is arguably cleaner, however, since it needs not rely on the vast complexities in establishing rent models where a myriad of users individually pay for rent each period.

[4]This should not be confused with deleting past transactions. Activity on Sui is finalized at each epoch boundary and thus past transactions are immutable and can never be reversed. The type of data that can be deleted is, for example, data corresponding to objects that are no longer live such as an NFT's metadata, tickets that have been redeemed, auctions that have concluded, etc.

The Sui economics white paper proceeds as follows. Section 2 begins by describing the main primitives and operations of the Sui platform. Section 3 introduces the main building blocks of the Sui economy and offers an overview of Sui's proof-of-stake economic model. Sections 4 and 5 delve deeply into the design and incentives present in Sui's gas price mechanism and storage fund, respectively. Section 6 discusses the long-term dynamics of Sui's economic model. Finally, section 7 offers some concluding thoughts. The appendix contains a summary of the model's free parameters.

2 Primitives of the Sui Platform

2.1 The SUI Token

The Sui platform's native asset is called SUI – and we will generally use the capitalized version of SUI to distinguish the token from the Sui platform.

The Sui platform divides time into sequential epochs that we index with the time subscript $e = 0, 1, 2, \ldots$ We denote the total supply of SUI at epoch e as $\mathbf{M}_e$. The SUI token's monetary rule is such that supply is non-decreasing over time – i.e. SUI tokens are never burnt and $\mathbf{M}_e \leq \mathbf{M}_{e+1}$ for all e. The long-run SUI supply is capped at $\lim_{e\to\infty} \mathbf{M}_e = 10,000,000,000$ tokens.[5] We will refer to $e = 0$ as the genesis epoch, at which point a non-zero amount of SUI tokens $\mathbf{M}_0 > 0$ are minted.

The SUI token serves four purposes on the Sui platform. First, the SUI token can be staked within an epoch in order to participate in the proof-of-stake mechanism. Second, the SUI token is the asset denomination needed for paying the gas fees to execute transactions or other operations on the Sui platform. Third, SUI can be used as a versatile and liquid asset for various applications including the standard features of money – a unit of account, a medium of exchange, or a store of value – and more complex functionality enabled by smart contracts, interoperability, and composability across the Sui ecosystem. Fourth, and finally, the SUI token plays an important role in governance by acting as a right to participate in on-chain voting on issues such as protocol upgrades.

2.2 Sui Objects and Transactions

The Sui platform relies on objects as its main building block. Sui objects can represent any type of digital asset, including fungible and non-fungible tokens. We will refer to actions on the Sui platform – such as object creations, deletions, mutations, or transfers – as transactions. A generic transaction takes objects as inputs, operates a specified set of instructions on the inputs, and produces subsequent objects as outputs.

Non-shared objects – objects owned by a single address – have three important characteristics. First, every object is tied with single-ownership by including an "address"

[5]While the Sui protocol does not include mechanisms for burning tokens explicitly, in practice various forces have a similar deflationary effect as that of burning tokens (see section 6.1). Note each SUI token is divisible up to a large number of decimal places.

field. Second, objects can be used in a transaction but only when authenticated by the signature of the owning address. Third, objects include a digest indicating the transaction that had said object as an output.

The set of objects that have not yet been used as inputs in a transaction is called the set of "live objects." The Sui platform's programming language is built such that non-live objects – i.e. objects already used as inputs in previous transactions – cannot be used again by future transactions. Consequently, the full set of objects and transactions across all epochs can be used to construct a directed acyclical graph (DAG) representing the evolution of Sui's state across time. In this DAG, objects correspond to vertices, transactions correspond to edges, and the set of live objects correspond to childless vertices and vertices with fewer outgoing edges than the transaction has outputs.

While objects represent the core elements of the Sui platform, the economics of Sui are best understood through the lens of transactions. For this reason, we will use the notation τ to refer to a generic transaction and refrain from modeling objects explicitly. The reader should keep in mind, however, that all transactions τ are associated with a list of object inputs, outputs, and actions.

2.3 Staking

The Sui platform relies on delegated proof-of-stake to determine the set of validators who process transactions. Within each epoch e, operations are processed by a set V_e of validators, with each validator $v \in V_e$ participating with an amount $S_e(v)$ of stake. The amount of stake is relevant in that it determines the share of voting power each validator has to process transactions. Call the collection of validators and delegated stake $C_e = (V_e, S_e(\cdot))$ a committee and denote the total amount of delegated stake $\mathbf{S}_e = \sum_{v \in V_e} S_e(v)$. It will be useful to define an validator's stake share as $\sigma_e(v) = S_e(v) / \mathbf{S}_e$. By construction, the following conditions hold: $\mathbf{S}_e \leq \mathbf{M}_e$ and $\sum_{v \in V_e} \sigma_e(v) = 1$ for all epochs e.

The Sui platform implements delegation by allowing any owner of the SUI token to delegate all or part of their holdings to a specific validator and participate in the staking rewards earned by such validator. When SUI token holders delegate SUI, the SUI tokens are locked at the chosen validator for the entire epoch.

SUI token holders can unlock their SUI or delegate them to different validators when the epoch changes. As a result of changes in delegation, committees evolve across epochs with both the set of live validators and distribution of managed stake potentially changing at the epoch boundary. That is, between two epochs e and $e+1$, both $V_e \neq V_{e+1}$ and $S_e(v) \neq S_{e+1}(v)$ for $v \in V_e, V_{e+1}$ will be generally true. Rewards from Sui's operations are distributed across various entities, including the set of validators and SUI delegators, at the epoch close. The next section discusses the procedure by which users, clients, and validators submit, process, and record transactions on the Sui platform.

2.4 System Operations

Sui's operations are secure as long as less than 1/3rd of the validators (weighted by stake) are Byzantine – i.e. that they deviate arbitrarily from the protocol (see the Sui white paper for further details). Processing a transaction on Sui requires two broad steps:

1. In the first step, a user cryptographically signs transaction τ with their private key and sends it to the current epoch's validator set V_e. Each validator validates the transaction and, in case of success, signs the transaction with their own private key and sends the signed transaction back to the user.

2. The second step occurs once signatures from at least 2/3's of the validators by stake have been received. Formally, this takes place once the user receives signatures from a quorum $Q_e \subset V_e$ such that $\sum_{v \in Q_e} \sigma_e(v) \geq 2/3$. These responses are then collected to form a transaction certificate. This certificate is subsequently sent to the validators, who check its signatures and execute the transaction. Finality is achieved once a quorum of validators has executed the certificate.

Note that the user was required to cryptographically sign their transaction only at the very beginning, when submitting the transaction for validation by the validator set. Hence, in practice, the subsequent process need not be carried out by the user itself but can instead be managed by a third-party client or gateway service.

The main economic benefit of Sui's transaction flow is it can be parallelized. For example, take two transactions τ and τ' such that their two sets of mutable input objects is disjoint. It is easy to see that the above two steps can be processed simultaneously for τ and τ' with the only requirement being that each validator devote separate resources to process each transaction. Without loss of generality, the same argument applies to the case of thousands or millions of simultaneous transactions. The ability to parallelize transactions emanates from Sui's object-centric design, which makes it trivial for the protocol to keep track of which transactions can be parallelized and in what manner.

The Sui platform thus scales throughput linearly by adding more computing power to each validator, while also scaling costs linearly. This delivers a cost-effective platform that remains fast and cheap regardless of the aggregate demand for the network's resources. More generally, the case of shared objects – where τ and τ' call the same input objects – is more complex. Shared objects imply that not all transactions can be fully parallelized and, moreover, that validators must run a consensus protocol to agree on the current state of a shared object. In these cases, a degree of parallelization is still possible by noting that while shared objects create causal dependencies, different non-causally-dependent shared objects can be parallelized. Together with this lighter parallelization, the Sui platform obtains agreement through a high-throughput DAG-based consensus mechanism to process shared objects. The reader can refer to the Sui white paper for technical details.

3 The Sui Economy: Basic Building Blocks

We now describe the economics of the Sui platform both within and across epochs. For the purposes of this discussion, we abstract away from some of the engineering intricacies that are less relevant to the network economics and incentives.

The Sui platform generates rewards to incentivize its operators and distributes these SUI tokens across network participants. We split this process into three steps. First, we describe the platform's ability to generate rewards through gas fees. Second, we introduce the Sui storage fund and show how it enables Sui to shift rewards across different epochs. Third, we review the platform's economic model for distributing rewards within a given epoch.

3.1 Gas Fees

The Sui platform generates rewards by charging users with gas fees.[6] Let τ be an arbitrary transaction on Sui – for example, an object creation, mutation, transfer, or deletion. The gas fees associated with processing transaction τ during epoch e equal:

$$\text{GasFees}_e[\tau] = \underbrace{\text{ComputationUnits}_e[\tau] \times P_e^C[\tau]}_{\text{computation fees in SUI}} + \underbrace{\text{StorageUnits}_e[\tau] \times P_e^S}_{\text{storage fees in SUI}}. \tag{1}$$

The gas functions $\text{ComputationUnits}_e[\tau]$ and $\text{StorageUnits}_e[\tau]$ measure the amount of computation and storage resources, respectively, required to process and store the data associated with τ. We index the gas functions with a time subscript since the computing and storage cost may change across epochs due to protocol upgrades, improvements in software and hardware, and other factors. Within an epoch, however, the gas function is deterministic and common across all network participants. The gas prices $P_e^C[\tau]$ and P_e^S capture the cost of one unit of computation or storage, respectively, in SUI units. Both computation and storage fees are invoiced in and must be paid for with the SUI token. Importantly, note that the computation gas price may differ across transactions both within and across epochs while the storage gas prices is constant within an epoch but varies across epochs.

In practice, the average user of the Sui platform uses fiat as their standard unit of account. This implies that for most users, what matters is not the SUI value of gas but the dollar value of gas. Let $P_e^{\$}$ be the dollar price of the SUI token at the start of epoch e. The dollar cost of processing transaction τ equals:

$$\begin{aligned}\text{GasFees}_e^{\$}[\tau] &= \text{GasFees}_e[\tau] \times P_e^{\$}, \\ &= \underbrace{\text{ComputationUnits}_e[\tau] \times P_e^C[\tau] \times P_e^{\$}}_{\text{computation fees in \$}} + \underbrace{\text{StorageUnits}_e[\tau] \times P_e^S \times P_e^{\$}}_{\text{storage fees in \$}}.\end{aligned}$$

[6]Gas fees have the added benefit of discouraging spam by introducing non-zero costs of network utilization.

The Sui economy is designed to keep gas fees low in dollar terms. As discussed above, the gas functions $\text{ComputationUnits}_e\,[\tau]$ and $\text{StorageUnits}_e\,[\tau]$ are determined by technological constraints while SUI's dollar price $P_e^{\$}$ is determined by market forces. Hence, the only degree of freedom in $\text{GasFees}_e^{\$}\,[\tau]$ lies in the gas prices $P_e^C\,[\tau]$ and P_e^S. Keeping gas fees low in dollar terms thus requires gas prices to move counter-cyclically with SUI's dollar price. When $P_e^{\$}$ is high then $P_e^C\,[\tau]$ and P_e^S should be low; vice-versa when $P_e^{\$}$ is low. The Sui economy achieves this by incorporating market-based incentives to keep the products $P_e^C\,[\tau] \times P_e^{\$}$ and $P_e^S \times P_e^{\$}$ roughly low and constant both within and across epochs. The gas price mechanism achieving this property is described in section 4.

3.2 Sui Storage Fund

The Sui platform is optimized to deliver high throughput and low latency even while storing, potentially, arbitrary amounts of on-chain data. This imposes an important challenge from an economic standpoint. The Sui network operates by relying on validators to process and execute transactions. Providing these services, however, requires having the data associated with past transactions on hand.

The challenge is that if users only pay gas fees for computation, then validators will have to fund both their current operations and storage overhead with gas fees from current computations. This represents a tax on the system since current users do not internalize the storage cost they're imposing on future validators. This is further complicated by the fact that the validator set V_e changes over time, implying that future validators will have to store data associated with past transactions from which they might not have obtained any rewards. Since future validators need a viable business model to survive, future computation fees would have to cover the costs of storage. In other words, future users would have to subsidize past users – an inefficient economic outcome.

Sui's economic model addresses the storage challenge by charging users storage fees upfront: a user submitting transaction τ must pay both for current execution and for future storage. In practice, operationalizing such a model in a sustainable manner becomes itself a complicated endeavor. To see why, note that charging storage fees delivers a finite amount of SUI tokens while storage costs are potentially infinite since data might need to be stored forever. Moreover, storage costs themselves are volatile and hard to predict. On one extreme, a solution is to simply charge upfront for storage for a finite amount of time and delete the data automatically if the user does not renew their storage fees at the time of expiry. On the other extreme, a solution is to design a storage fee model that can cover storage costs indefinitely. We take the view that the latter approach is better for user experience and the platform's overall economic model.

Sui's economic model includes a storage fund designed to provide a sustainable and viable long-run mechanism for compensating validators for the cost of storage. In a nutshell, the storage fund is used to adjust the staking rewards paid to validators, so that validators obtain an additional source of rewards to help offset their storage costs.

The storage fund has three key features. First, the storage fund is funded by past transactions. This ensures that future validators are compensated for their storage costs by the past users who created those storage requirements in the first place. In other words, the storage fund provides a tool for shifting stake rewards across different epochs. Second, the storage fund distributes tokens indirectly through the stake rewards accrued to its SUI deposits but does not actually pay out the deposits directly. This preserves the fund's capitalization and guarantees it can survive indefinitely. Third, the storage fund's mechanics incentivize users to delete data and obtain a rebate on their storage fees when the cost of storing such data exceeds the value obtained from maintaining that data on-chain. Hence, this design is efficient since it distributes rewards to compensate for existing storage and also includes a market-based mechanism for eliminating storage when it is no longer attractive from an economic standpoint.

At a high-level, mechanics of the storage fund are as follows (details described in section 5). The storage fund's size is fixed throughout the duration of an epoch and adjusted at the epoch boundary. Inflows correspond to the fund's reinvestment of a share of the return on its capital into new principal, plus the epoch's gas storage fees. Outflows correspond to the rebates accrued to users who delete data. Formally, the storage fund at the epoch boundary between e and $e+1$ is given by:

$$\mathbf{F}_{e+1} = \mathbf{F}_e + \underbrace{\text{Reinvestment}_e + \sum_{\tau \in T_e} \text{StorageUnits}_e\,[\tau] \times P_e^S}_{\text{inflows}} - \underbrace{\sum_{\tau \in R_e} \text{Rebates}_e\,[\tau]}_{\text{outflows}}, \tag{2}$$

where T_e represents the set of transactions processed throughout epoch e, R_e represents the set of past transactions – i.e. from e or before – whose data was eliminated throughout epoch e, and $\text{Rebates}_e\,[\tau]$ is function capturing the rebates accrued by the users deleting the data associated with τ. Note that the storage fund is denominated in SUI units.

3.3 An Economic Model with Proof-of-Stake and Storage

We now discuss how the above elements interact with each other in order to introduce Sui's economic model. Throughout this section, we will use the visual representation in Figure 1 to aid the discussion. There are two key high-level differences between Sui's economic model and traditional proof-of-stake systems.

First, entities participating in the system's operations can expect to achieve a smooth source of rewards across time, as opposed to the volatile reward streams in some alternative models. This is a consequence of Sui validators playing a passive role as opposed to the active role validators play elsewhere. Since processing each transaction requires that a quorum of validators participate, all validators can benefit in proportion to their share of total stake during every epoch if they behave honestly. This is in contrast to other reward systems where the probability of receiving rewards at a given moment in time is proportional to the share of stake. This feature has important implications for the evolution of the stake distribution across validators over time (see section 6.3.) This design also has important implications for the quality of service provided by each validator. While a

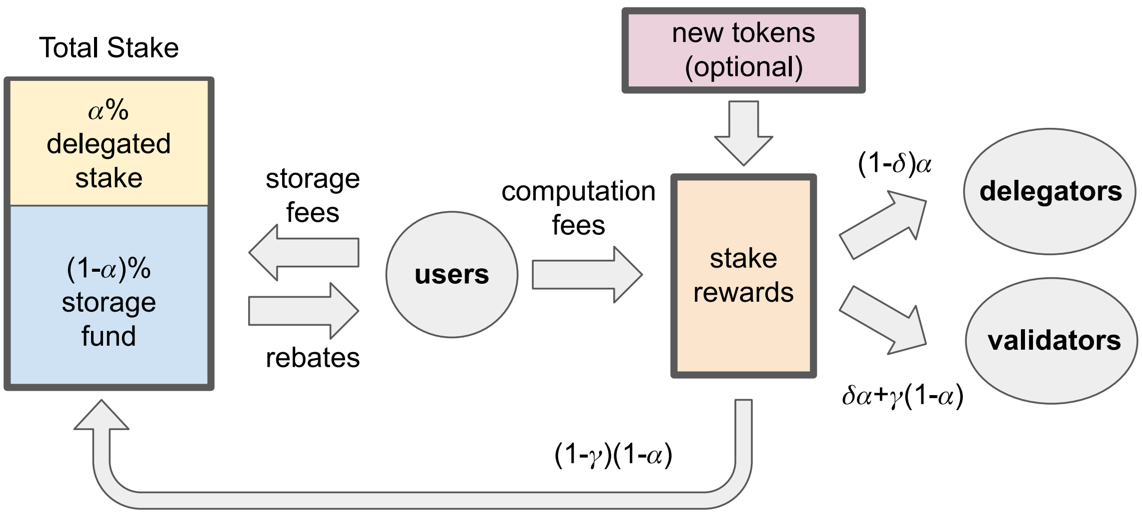

Figure 1: The Sui Economy

larger stake share lets validators reap more stake rewards, large validators are also more likely to be prioritized by clients during regular network operations. Consequently, larger rewards are partially offset by the increased costs of scaling operations; thus ensuring all validators enjoy viable business models regardless of their delegated stake size.

Second, the presence of the Sui storage fund delivers the ability to shift rewards across epochs. This implies that the proof-of-stake mechanism needs to be adjusted to account for the storage fund's presence. In particular, the economic model needs to be carefully designed in order to preserve the incentives arising from the proof-of-stake mechanism while accommodating the additional incentives arising from the storage fund.

The Sui economic model works as follows:

- At the beginning of epoch e: Three important things happen at the epoch boundary between $e-1$ and e. First, SUI holders delegate (some of their) tokens to validators, and a new validator committee $C_e = (V_e, S_e(\cdot))$ is formed. Second, the reference gas prices are set (see section 4). Third, the size of the storage fund is updated to $\mathbf{F}_e$ as described in equation (2). This last action is important because the Sui economic model will assume that the total amount of stake is given by the sum of delegated stake plus the storage fund. In other words, the total amount of staked SUI during epoch e is given by: $\mathbf{S}_e + \mathbf{F}_e$, where remember that $\mathbf{S}_e = \sum_{v \in V} S_e(v)$. It will be useful to define the auxiliary variable α_e as the share of delegated stake:

$$\alpha_e = \frac{\mathbf{S}_e}{\mathbf{S}_e + \mathbf{F}_e}.$$

  Note that α_e is an endogenous variable that changes over time in response to the aggregate decisions of Sui's users, delegators, and validators.

- During epoch e: Users submit transactions to the Sui platform, and validators process them. Remember that T_e is the set of transactions processed during the epoch. For each transaction $\tau \in T_e$ users pay the $\text{GasFees}_e[\tau]$ described in equation (1). If τ corresponds to a transaction deleting the data associated with a past transaction τ' then the user receives a SUI transfer of $\text{Rebate}_e[\tau']$.

- At the end of epoch e: validators vote to end the current epoch and exchange information to commit to a checkpoint with the aid of an agreement protocol. The union of all transactions processed by a quorum of validators is calculated in order to agree on the current state of the Sui platform. The final step is to distribute the epoch's rewards to the different entities. This occurs in two steps:

  - First, we must calculate the total amount of rewards generated throughout the epoch. These rewards are distributed to the entities who participated in the staking process. In Sui, there are two sources of stake rewards: computation fees and new token issuance. Formally:

    $$\text{StakeRewards}_e = \sum_{\tau \in T_e} \text{ComputationUnits}_e[\tau] \times P_e^C[\tau] + (\mathbf{M}_{e+1} - \mathbf{M}_e).$$

    As Figure 1 shows, stake rewards from new token issuance are optional in the sense that some epochs may see zero stake rewards raised through this channel. Indeed, in the long-term it is necessarily the case that no new tokens are issued given that the total amount of SUI in circulation is capped. More likely, most of the new token issuance designated as stake rewards will be paid out in Sui's initial epochs to subsidize validators when network activity is still nascent.

  - Second, we determine the split of stake rewards across network participants. To do this we need to discuss the role of the storage fund, one of the most important elements of the Sui economy.

    Simply put, the storage fund lets validators increase the number of SUI tokens they receive relative to delegators by adjusting their share of staking rewards. Since the storage fund is tallied into the calculation of total stake, a share $1 - \alpha_e$ of the staking rewards accrue to the storage fund. In contrast to delegated stake, however, the storage fund is not owned by the delegators; and this opens the question of who should receive these rewards. The Sui economic model takes the view that the rewards accruing to the storage fund should be used to compensate for storage. Since validators are the entities storing data, they should be the entities entitled to these rewards.

    Formally, the distribution of staking rewards is as follows. Assume that delegators enter into contracts with their respective validators such that validators are entitled to a commission $\delta \in [0, 1]$ for their services.[7] Delegators receive the

[7] This is easily generalizable to a setting where each validator negotiates a separate commission with its delegators and where this commission changes over time. In such a case, we would index the commission paid to validator $v \in V_e$ as $\delta_e(v)$.

following amount of staking rewards:

$$\text{DelegatorRewards}_e = (1 - \delta) \times \alpha_e \times \text{StakeRewards}_e.$$

Meanwhile, validators receive the remaining rewards corresponding to delegated stake and a share $\gamma \in [0, 1]$ of the rewards corresponding to the stake from the storage fund:

$$\text{ValidatorRewards}_e = (\delta \times \alpha_e + \gamma \times (1 - \alpha_e)) \times \text{StakeRewards}_e. \tag{3}$$

Setting $\gamma < 1$ and distributing less than the full amount of storage fund rewards to validators is useful for preserving the storage fund's long-term financial health. In practice, γ will likely be close to 1 and updated infrequently through governance proposals depending on the storage fund's health. The storage fund's capital inflow from stake rewards is given by:

$$\text{Reinvestment}_e = (1 - \gamma) \times (1 - \alpha_e) \times \text{StakeRewards}_e.$$

This scheme represents a full accounting schedule:

$$\text{StakeRewards}_e = \text{DelegatorRewards}_e + \text{ValidatorRewards}_e + \text{Reinvestment}_e.$$

In sum, the storage mechanism allows validators to obtain additional rewards beyond the share corresponding to their delegated stake in order to fund their storage overhead. The storage fund acts as a wedge between SUI delegators and validators permitting the latter to increase their share of overall staking rewards. To see this note that:

$$\underbrace{(\delta \times \alpha_e + \gamma \times (1 - \alpha_e))}_{\substack{\text{validator rewards with} \\ \text{storage pricing}}} \quad > \quad \underbrace{\delta}_{\substack{\text{validator rewards without} \\ \text{storage pricing}}} \quad \Leftrightarrow \quad \gamma > \delta.$$

Effectively, it is as if the validators were able to borrow the SUI deposited in the storage fund at a lower interest rate than the SUI borrowed from delegators. This is true whenever $\gamma > \delta$ and, in practice, will be the case since delegators only delegate if served with a low commission (low δ) while the protocol is designed to reward for storage (high γ).

4 Gas Price Mechanism: Design and Incentives

The Sui gas price mechanism is designed to achieve two overarching goals. First, gas prices should be low in $-terms and predictable both within and across epochs. This delivers good user experience to Sui users, who can focus on using the Sui network without worrying about the level and volatility of transaction fees. Second, the gas price mechanism is designed to encourage and reward good validator behavior throughout Sui's regular operations. This arrangement aligns incentives between the SUI token holders, the network's operators (i.e. validators), and its users.

4.1 Computation Gas Prices

Remember from equation (1) that computation gas prices $P_e^C[\tau]$ are set at the transaction level and thus vary both within and across epochs. More specifically, the Sui network unbundles computation gas prices into separate fixed and tip components:

$$P_e^C[\tau] = \underbrace{\overline{P}_e^C}_{\text{fixed component}} + \underbrace{\zeta[\tau]}_{\text{tip}}, \qquad \text{s.t. } P_e^C[\tau] > \underbrace{\underline{P}_e^C}_{\text{price floor}}.$$

The fixed component $\overline{P}_e^C$ is set at the network level for the duration of the epoch while the tip $\zeta[\tau]$ is at the discretion of the user. Since $\zeta[\tau]$ can be negative, but constrained to keep the overall gas price positive and above the price floor $\underline{P}_e^C$, the user submitting τ is simply stating how much they are willing to pay relative to the network-wide fixed component: $\zeta[\tau] = P_e^C[\tau] - \overline{P}_e^C$. The price floor exists to prevent the network being flooded from spam, and should not affect the processing of regular activity. In practice, the price floor can be set in proportion to the reference price, such as $\underline{P}_e^C = \beta \overline{P}_e^C$ with $\beta < 1$.

We will refer to the fixed component $\overline{P}_e^C$ as the reference gas price. Sui's gas mechanism is designed to make the reference gas price a credible anchor for users to use when submitting transactions on the network. That is, users can be reasonably confident that submitting transactions with gas prices at or close to the reference gas price, i.e. $P_e^C[\tau] \approx \overline{P}_e^C$ or $\zeta[\tau] \approx 0$, will be processed in a timely manner.

The gas pricing mechanism has three elements:

1. Gas Price Survey: An validator-wide survey is used to set the reference gas price at the beginning of each epoch. This delivers a coordination price point $\overline{P}_e^C$ around which users can submit their gas price quotes $P_e^C[\tau]$.
2. Tallying Rule: A validator-wide survey is used as an input into the distribution of stake rewards at the end of each epoch. This delivers the incentives for validators to honor the reference gas price $\overline{P}_e^C$ determined during the gas price survey.
3. Incentivized Stake Reward Distribution Rule: The amount of stake rewards distributed to each validator is adjusted using information from the gas survey and tallying rule. This delivers the incentives for validators to set low reference gas prices $\overline{P}_e^C$ in the long run and prevents validators from gaming the system.

Jointly, these three elements create a gas price mechanism delivering a low, stable, and credible reference gas price $\overline{P}_e^C$ for users while ensuring that validators honor such prices and process transactions in a timely fashion. We now describe each element in detail.

4.1.1 Gas Price Survey: What's the Gas Price?

The gas price survey occurs right before the epoch boundary, at the moment of committee formation. This occurs in two steps:

- First, when validators propose the next epoch's validator set and stake distribution $C_e = (V_e, S_e(\cdot))$, they also include a gas price proposal $\overline{p}_e^C(v)$ for each $v \in V_e$.

- Second, the $|V_e|$ bids are aggregated to deliver a reference price such that 2/3's of the proposals by stake are at or below this threshold. Formally, without loss of generality, the validators are ordered such that $v \leq v'$ implies that $\overline{p}_e^C(v) \leq \overline{p}_e^C(v')$. The reference gas price is set at:

$$\overline{P}_e^C = \overline{p}_e^C(v^*), \quad \text{with } v^* \in V_e \quad \text{s.t.} \quad \sum_{v=1}^{v^*-1} \sigma_e(v) < \frac{2}{3} \quad \text{and} \quad \sum_{v=1}^{v^*} \sigma_e(v) \geq \frac{2}{3}.$$

Essentially, the gas price survey asks each validator: at what price are you willing to process transactions? Aggregating the responses delivers a reference gas price $\overline{P}_e^C$ around which users can reasonably assume that a 2/3's quorum of validators by stake will process their transaction promptly.

Two challenges remain. First, what incentivizes validators to truthfully reveal their reservation gas price during the gas survey and ensure a quorum will actually process transactions around the reference gas price? Second, even if validators honor their price quotes, what prevents validators from setting an arbitrarily high reference gas price?

4.1.2 Tallying Rule: How to Split the Pie?

The tallying rule is applied at the close of epoch e, once the current validator set reach full agreement on the transactions processed during the epoch and before stake rewards are paid out. The tallying rule is used by each validator to construct a subjective measure over how much staking rewards should be distributed to every other validator.

The tallying rule's goal is to have a community-enforced system for encouraging validators to honor the quotes $\overline{p}_e^C(v)$ submitted during the gas price survey and thus incentivize validators to reveal their true reservation prices. In particular, by punishing validators who do not honor their quotes, these incentives discourage validators who attempt to game the system by submitting arbitrarily low gas price quotes.

The tallying rule has three elements:

- Executed Gas Price Distribution: Let T_e be the set of transactions executed during epoch e. Since each transaction $\tau \in T_e$ includes a computation gas price $P_e^C[\tau]$, validators can construct the executed gas price distribution:

$$T_e[p] = \left\{ \tau \in T_e \text{ s.t. } P_e^C[\tau] \geq p \right\}.$$

  This distribution requires data known with certainty at the epoch boundary and is thus a common, objective metric known by all validators.

- Reasonable Execution Metric: Each validator v makes a subjective evaluation regarding the transactions that every other validator v' processed during the epoch.

In particular, this estimate is relative to the quote $\overline{p}_e^C(v')$ submitted during the gas price survey. Formally:

$$\hat{T}_e^v(v') = \left\{\tau \in T_e\left[\overline{p}_e^C(v')\right] \text{ s.t. } v' \text{ processed } \tau \text{ in reasonable time}\right\}.$$

The main intuition is if validator v' submitted a gas price quote of $\overline{p}_e^C(v')$, then it should have processed all transactions $\tau \in T_e$ such that $P_e^C[\tau] \geq \overline{p}_e^C(v')$ promptly.

The reasonable execution metric is a subjective measure since it depends on a combination of data collected throughout the epoch by each individual validator v and objective data known to all validators at the epoch boundary. For example, validators can implement gossip between them, with each validator listening to a few others and receiving notifications on their processed transactions. Differences across validators can be used to estimate the relative performance of each validator v' from the point of view of the listening validator v. Additional information such as provable Byzantine behavior, known delays in providing information, observing which validators sign which transactions, and other strategies can be used as further information sources. Ultimately, though, this metric is subjective since it depends on each validator's ability to obtain informative signals on its peers.

- Tallying rule: The executed gas price distribution and reasonable execution metric are used to construct an estimate of relative validator performance. Specifically, validator v proposes the following multiplier for each other validator v':

$$\hat{\mu}_e^v(v') = \phi^v \times \frac{\sum_{\tau \in \hat{T}_e^v(v')} \text{ComputationUnits}_e[\tau] \times P_e^c[\tau]}{\sum_{\tau \in T_e\left[\overline{p}_e^C(v')\right]} \text{ComputationUnits}_e[\tau] \times P_e^c[\tau]},$$

where each ϕ^v is a normalizing constant such that

$$\frac{1}{|V_e| - 1} \times \sum_{v' \in V_e \setminus \{v\}} \hat{\mu}_e^v(v') = 1.$$

The multiplier's numerator sums computation gas fees across all transactions validator v' executed within reasonable time, out of all the transactions with gas prices above the validator's self-declared reservation gas price. The denominator sums computation gas fees across all transactions executed in the epoch in which the gas price was at least as high as the reservation price of validator v'. In other words, the denominator includes transactions validator v' should have processed promptly but did not. Both the numerator and denominator are weighted by the executed gas fees since the relevant metric is not the number of transactions an validator processed, but the amount of computation it processed, relative to what it should have.[8]

Finally, the normalization ϕ^v is included so that each validator v submits a set of multipliers $\hat{\mu}_e^v(v')$ for all other validators $v' \in V_e \setminus \{v\}$ that average out to 1 but

[8]Note that all of the tallying rule variables, including $T_e[p]$, $\hat{T}_e^v(v')$, and $\hat{\mu}_e^v(v')$, can be approximated with sampling techniques to speed up calculation in epochs when the executed transaction set T_e is large.

in which validators with relatively good performance get a boost $\hat{\mu}_e^v(v') > 1$ and validators with relatively bad performance get a discount $\hat{\mu}_e^v(v') < 1$.[9]

In sum, the tallying rule delivers a multiplier whereby each validator v says: If validator v' operated well in the sense that it processed all transactions above its self-declared reservation gas price in reasonable time then its stake rewards should be boosted. If not, then its stake rewards should be discounted/punished.

The tallying rule thus creates community-enforced incentives for validators to honor the gas price quotes submitted during the gas survey. These incentives trickle upstream and encourage validators to submit honest quotes to begin with since, by providing quotes they can honor, validators avoid getting their rewards slashed.

4.1.3 Incentivized Stake Reward Distribution Rule: A Healthy Competition for Fair Prices

The tallying rule incentivizes validators to submit gas price quotes they can honor, but the gas mechanism is still missing incentives to keep gas prices low. The incentivized stake reward distribution rule encourages an equilibrium where the validator set collectively proposes a low reference gas price. This rule is implemented in three steps:

- First, the protocol computes epoch e's total validator stake rewards as described in equation (3).
- Second, the protocol computes a set of global multipliers using the set of validator-submitted multipliers from the tallying rule. Formally, the global multiplier for validator v is given by:

$$\hat{\mu}_e(v) = \text{Median}\left\{\hat{\mu}_e^1(v), \ldots, \hat{\mu}_e^{v-1}(v), \hat{\mu}_e^{v+1}(v), \ldots, \hat{\mu}_e^{V_e}(v)\right\},$$

  where the median is weighted by the distribution of validator stake $\sigma_e(v')$. The median rule helps guard Sui's economic model against Byzantine behavior, where a subset of Sui validators attempt to appropriate a disproportionate amount of rewards by giving each other excessively high multipliers. Note that validator v does not submit a quote over its own performance.
- Third, the total amount of validator rewards in equation (3) is distributed to individual validators according to the following incentivized distribution rule:

$$\text{ValidatorRewards}_e(v) = \hat{\sigma}_e(v) \times \text{ValidatorRewards}_e$$

  where the share of validator v equals:

$$\hat{\sigma}_e(v) = \begin{cases} \psi \times (1+\kappa) \times \hat{\mu}_e(v) \times \sigma_e(v), & \text{if } v \le v^*, \\ \psi \times (1-\kappa) \times \hat{\mu}_e(v) \times \sigma_e(v), & \text{if } v > v^*. \end{cases}$$

[9] The normalization is important since it focuses attention on the relative performance of other validators from the listening validator's perspective instead of focusing on absolute performance (i.e. focuses on variance in multipliers instead of levels). Consequently, the multipliers contain useful information even in cases where validators differ vastly in their ability to obtain subjective information about each other.

	if validator does process transactions promptly $\hat{\mu}_e(v) \geq 1$	if validator does not process transactions promptly $\hat{\mu}_e(v) < 1$
if validator submits low quote: $\overline{p}_e(v) \leq \overline{P}_e$	$\hat{\sigma}_e(v) \geq \psi \times (1+\kappa) \times \sigma_e(v)$	$\hat{\sigma}_e(v) < \psi \times (1+\kappa) \times \sigma_e(v)$
if validator submits high quote: $\overline{p}_e(v) > \overline{P}_e$	$\hat{\sigma}_e(v) \geq \psi \times (1-\kappa) \times \sigma_e(v)$	$\hat{\sigma}_e(v) < \psi \times (1-\kappa) \times \sigma_e(v)$

Table 1: Incentivized Stake Reward Distribution Rule: The gas mechanism creates incentives for validators to submit low gas price quotes, but only to the point at which they can reasonably honor those gas fees.

Remember that $v \leq v^*$ indexes validators submitting quotes below the reference price, i.e. $\overline{p}_e^C(v) \leq \overline{P}_e^C$, while $v > v^*$ corresponds to validators above: $\overline{p}_e^C(v) > \overline{P}_e^C$. The parameter ψ is a normalizing constant such that $\sum_{v \in V_e} \hat{\sigma}_e(v) = 1$. This normalization is important since it prevents gaming the system: the set of multipliers $\hat{\mu}_e(v)$ are zero-sum in the sense that if some validators get boosted rewards then other validators must necessarily face discounted rewards.

The key innovation in the incentivized rule is that $\kappa > 0$ is included as an additional multiplier to boost the rewards obtained by validators who submit low gas price quotes – specifically, quotes below the 2/3's percentile. Analogously, validators who submit high gas price quotes receive a reduction in their rewards.

Table 1 summarizes validator incentives. Two key forces are present: the tallying rule incentivizes validators to honor the quotes submitted during the gas survey while the distribution rule incentivizes validators to submit low gas prices. The interaction of these two forces is critical. On net, the gas price mechanism encourages validators to submit low gas price quotes – but not too low since then they will be punished for not honoring those bids. Sui's gas price mechanism thus encourages a healthy competition for fair prices. In the ideal equilibrium, all validators optimize their operations and behavior to deliver good performance. In such a symmetric equilibrium, validators receive a share of rewards proportional to their share of overall stake, i.e. $\hat{\sigma}_e(v) = \sigma_e(v)$.

SUI delegators are subject to the same forces since they inherit the proportional share of rewards accruing to their delegate validator. Specifically, the total amount of stake rewards distributed to the delegators of validator v equals:

$$\text{DelegatorRewards}_e(v) = \hat{\sigma}_e(v) \times \text{DelegatorRewards}_e.$$

SUI delegators thus play an important monitoring role by optimizing their delegation decisions according to validator behavior. validators are doubly incentivized to good behavior, otherwise they get punished directly through slashed rewards and indirectly through losing delegated stake in future epochs.

4.2 Storage Gas Prices

In contrast to computation gas prices $P_e^C[\tau]$, storage gas prices P_e^S are constant for all transactions within an epoch and only (infrequently) vary across epoch boundaries.

Setting storage gas prices requires a different mechanism from computation gas prices for two reasons. First, storage prices are charged on transactions executed by current validators but used to reward future validators. This creates a wedge between the incentives that current and future validators care about. Second, storage prices are solely intended to create a sustainable business model for future validators and not primarily intended to incentivize proper network operations in the way computation gas prices do. For these two reasons, Sui's storage pricing framework is more straightforward than its computation pricing mechanism.

Sui's storage prices are set through governance proposals for the duration of various epochs (e.g. for the period of a few months). Specifically, a storage pricing target is set exogenously by fixing the dollar value of one unit of storage. Call $\$x$ the dollar cost of storing one unit of storage for one epoch. The storage gas price is then set to:

$$\overline{P}^S = \frac{\$x}{rP^{\$}},$$

where r is the average nominal return on stake rewards (non-annualized) and $P^{\$}$ is SUI's average dollar price, both taken over a preceding window (e.g. over the last week). The storage price in each subsequent epoch $P_e^S = \overline{P}^S$ is set at this level until a new governance proposal is passed. This targeting ensures that storage fees are roughly fixed in dollar terms for as long as the target is applied, with the user submitting transaction τ paying for storage in \$-terms equal to:

$$\text{StorageUnits}_e[\tau] \times P_e^S \times P_e^{\$} \approx \text{StorageUnits}_e[\tau] \times \frac{\$x}{r}.$$

Since validators receive the returns on the storage fund's SUI, they receive a multiple $r\%$ of the above during each epoch.

We expect storage prices to be updated through governance proposals when SUI's dollar price exhibits a substantial level shift. In the long-run, $\overline{P}^S$ will likely tend to fall as the dollar-cost of storage falls with technological improvements.

4.3 Gas Prices as a Coordination Mechanism

Sui's gas price mechanism provides end users with credible reference points for submitting their transactions. By incentivizing validators to elicit their true reservation prices and to honor these quotes, Sui users can credibly assume transactions submitted at or close to the computation reference price will be processed in a timely manner. Likewise, since the protocol requires storage fees be deposited into the storage fund, validators have no incentive to charge users more or less than the reference storage gas price.[10]

Overall, users submitting transactions τ with gas prices $P_e^C[\tau] = \overline{P}_e^C$ and P_e^S face good user experience and clients, such as wallets, should automatically feed these prices to users. Sui's gas mechanism avoids the pitfalls of first-price, auction-based settings where users typically overpay for gas. Similarly, Sui's gas mechanism is consistent with Sui's ability to scale horizontally. When network activity increases, validators add more workers, increase their costs linearly, and are still able to process transactions at low gas prices. In cases of extreme network congestion where validators cannot scale fast enough, the tip's presence provides a market-based regulating mechanism that discourages further demand spikes by increasing the cost of transacting on the Sui platform.

In the long run, Sui's gas mechanism creates incentives for validators to optimize their hardware and operations. validators who invest in becoming more efficient are able to honor lower gas prices and obtain a reward boost of $1 + \kappa$. Sui validators are thus encouraged to innovate and improve the experience of end users.

5 Storage Fund: Design and Incentives

The Sui storage fund is designed to provide future validators with a viable business model: To compensate future validators for storing on-chain data they did not obtain computation gas fees from at the moment of write. We now describe the detailed workings of the storage fund and explain how its design covers storage costs in perpetuity.

5.1 The Storage Fund's Long-Term Viability

There are two key concerns regarding the storage fund's long-term viability. First, it is critical that the fund's assets are never depleted. An empty storage fund is useless. Second, the storage fund's size should be correlated with the amount of data held in storage by validators. Otherwise, validators will not be able to align their storage cost structure with the rewards proceeding from the storage fund.

Sui's economic model was designed to preserve the storage fund's capital, which is never used directly as a source of SUI tokens. Rather, the storage fund simply distributes the return on its capital (i.e the staking rewards) to validators. By never touching the fund's principal, this design protects the fund's ability to distribute rewards for storage indefinitely. This feature is further buttressed by the capital reinvested at the end of each epoch, equal to a $1 - \gamma$ share of the fund's returns.

[10]Storage gas fees are reminiscent of the base fee in Ethereum's post-EIP-1559 world. In Ethereum, validators must charge users with the base fees since the protocol burns these fees. In Sui, validators must charge users with the storage fees since the protocol deposits these fees into the storage fund.

Sui's economic model was designed to ensure the storage fund's size is commensurate with the amount of data held in storage. This goal is achieved by denominating data deletion rebates in terms of the storage fees originally paid when the data was written. Formally, deleting the data associated with a transaction $\tau \in T_e$ executed in epoch e, during epoch $e' \geq e$ delivers:

$$\text{Rebates}_{e'}\left[\tau\right] = \theta \times \text{StorageUnits}_{e}\left[\tau\right] \times P_e^S \tag{4}$$

where $\theta \in [0,1]$. In the extreme case where $\theta = 1$, rebates fully return the storage fees. The rebate function is justified by the fact that storage fees exist to compensate for storage throughout the data's lifecycle. There is no reason to keep charging for storage once data has been deleted, and so these fees are fully rebated. Users thus enjoy a "deletion option" whereby they pay for storage but are also able to obtain a rebate whenever that storage no longer makes financial sense to them. More generally, $\theta < 1$ is useful if some but not all the data associated with a transaction τ can be deleted and a share $1-\theta$ of the storage fees remain in the fund to compensate storage costs in perpetuity.[11]

The key property of the rebate function is that it limits storage fund outflows to be always less than the original storage inflow, at the individual transaction level. In particular, note that the storage gas price $P_{e'}^S$ at the time of deletion in epoch e' is irrelevant since the storage rebate is proportional to the SUI deposited at the time of write. This mechanism guarantees that the storage fund's size moves in line with the amount of data held in storage. A simple way to think of the storage fund is as if it were made out of a collection of individual accounts. Each account corresponds to the objects associated with a past transaction τ and the amount of deposited funds equals the storage fees paid when τ was processed. The owner of τ's output objects is the owner of these accounts and can withdraw the funds as long as they delete the associated objects. This accounting is useful for proving the claim that the storage fund can never be depleted because it always contains at least the storage fees associated with the live objects held in storage.

To conclude, the storage fund's recursive formulation in equation (2) can be rewritten under the above interpretation of a series of individual accounts corresponding to the transactions executed on Sui. Specifically, the storage fund's value at the end of epoch e equals the sum of the fund's initial value at genesis, capitalization inflows from each epoch up to e, and the full amount of storage fee inflows net of deletion rebates:

$$\mathbf{F}_{e+1} = \mathbf{F}_0 + \underbrace{\sum_{\varepsilon=0}^{e} \text{Reinvestment}_{\varepsilon}}_{\text{capitalizations}} + \underbrace{\sum_{\varepsilon=0}^{e} \sum_{\tau \in T_\varepsilon} \left(1 - \theta \times \mathbb{I}\left[\tau \in \bigcup_{\varepsilon'=\varepsilon}^{e} R_{\varepsilon'}\right]\right) \times \text{StorageUnits}_{\varepsilon}\left[\tau\right] \times P_{\varepsilon}^S}_{\text{storage fees net of deletions}}.$$

Notation is such that $\mathbb{I}[\cdot]$ represents the indicator function and if $\tau \in T_\varepsilon$ and $\tau \in \bigcup_{\varepsilon'=\varepsilon}^{e} R_{\varepsilon'}$ are both true, then τ is a transaction processed during epoch ε that has been deleted at some moment between then and the current epoch $e \geq \varepsilon$. In such cases, only a share $1-\theta$ of the originally paid storage fees remain in the storage fund.

[11] In practice, it may occur that a larger share of data can be deleted for some transaction types than for others. In such cases, $\theta[\tau]$ can vary across transaction types to capture this heterogeneity.

6 The Sui Economy: Long-Term Dynamics

6.1 SUI Deflation

The Sui economy does not include any mechanism to burn SUI tokens directly.[12] However, since the long-run supply is capped at ten billion tokens, increased activity on the Sui platform effectively acts as a deflationary force. If Sui unlocks more uses cases and more users migrate to the platform, the dollar price of SUI will likely increase since the relative amount of economic activity on Sui versus the off-chain world increases. As a result, on-chain SUI prices – including gas prices – fall and the Sui economy becomes deflationary.

Beyond the standard deflationary effects derived from SUI's finite supply, the Sui storage fund introduces two additional deflationary forces. One temporary and the other quasi-permanent. The storage fund's temporary effect arises from the storage fund's tokens being locked up and unusable for any other activities. Hence, while overall token supply during epoch e equals $\mathbf{M}_e$, the true amount of SUI tokens available for staking, paying gas fees, and other activities on Sui is given by $\mathbf{M}_e - \mathbf{F}_e$. This effect is only temporarily deflationary since, in principle, users can delete their on-chain data and release SUI tokens from the storage fund. That said, since storage is likely to increase with network activity, this deflationary force is likely to be important in the long run.

The more interesting effect is the storage fund's quasi-permanent effect on the SUI token supply. Let $\tilde{\mathbf{M}}_{e+1}$ be the maximum number of SUI tokens that can be in circulation at the epoch boundary between e and $e+1$. This term can be computed recursively as:

$$\tilde{\mathbf{M}}_{e+1} = \tilde{\mathbf{M}}_e + \underbrace{(\mathbf{M}_{e+1} - \mathbf{M}_e)}_{\text{SUI issuance}} - \underbrace{\text{Reinvestment}_e}_{\substack{\text{storage fund}\\ \text{capitalization}}} - \underbrace{\sum_{\tau \in T_e} \frac{1-\theta}{\theta} \times \text{Rebates}_e[\tau]}_{\text{rebate residual}}.$$

The effective number of tokens in circulation at $e+1$ equals the effective number of tokens in circulation at e, plus the new issuance of SUI tokens, minus the tokens reinvested to capitalize the storage fund, minus the residual of new storage rebates. The capitalization term captures the fact that stake rewards reinvested in the storage fund are deposited there in perpetuity – that is, they are not indirectly owned by any write transaction and thus cannot be withdrawn by any party. Similarly, the rebate residual is given by the share of storage fees remaining in the storage fund in perpetuity to fund the storage of data that cannot be deleted.[13] Since these last two terms represent coins deposited in the

[12] That said, tokens sent intentionally or accidentally to addresses without known private keys are effectively burnt.

[13] Note that the rebate residual is summed over $\tau \in T_e$, the set of transactions processed during e, and not over $\tau \in R_e$, the set of transactions rebated during e. This is the correct accounting since the rules are such that a share $1-\theta$ of a transaction's storage fees will remain in the storage fund when the associated data is deleted in the future. Whether that rebate has already occurred or not is immaterial; for all effective purposes those coins are already locked in perpetuity in the storage fund.

storage fund in perpetuity, $\tilde{\mathbf{M}}_{e+1}$ captures the maximum number of SUI tokens that can be in circulation even in the extreme case where all users delete their on-chain data.

The storage fund's quasi-permanent deflationary effect is not fully permanent because the Sui economy has a safeguard to prevent the storage fund from growing too big. The risk is that network incentives may get out of sync if the share of delegated stake α_e becomes too small.[14] To this end, $\underline{\alpha} \in (0, 1)$ is such that $\alpha_e \leq \underline{\alpha}$ triggers an outflow of the storage fund principal. This outflow will be limited to the portion of the fund's principal endowed through storage fund capitalizations or rebate residuals – not storage fees. This preserves the fund's long-term viability while keeping its size manageable. The lower bound $\underline{\alpha}$ is likely to be updated over time through on-chain governance, and the outflow funds can be set aside as future stake reward subsidies.

In sum, the storage fund introduces two important deflationary effects on the SUI token, each with different depth and lasting impact. The temporary deflation effect is stronger since it removes a larger share of SUI from circulation. But the temporary effect is potentially short-lived since it relies on the current amount of data in storage, which can change at any moment. The quasi-permanent deflation effect has a weaker impact, but potentially lasts forever and depends on the full history of storage on the Sui platform – regardless of whether that storage has been deleted or not.

6.2 Capital Efficiency

The storage fund is a capital efficient way of paying for storage from the user's perspective. While this may seem counterintuitive – since this model requires locking SUI in the storage fund – achieving capital efficiency was a key goal in Sui's economic design.

Capital efficiency follows from the fact that, in equilibrium, the user's opportunity cost of locking up SUI is exactly equal to the fees they would otherwise pay for storage. To see this, assume the Sui economy is in steady state such that all variables are constant across time, markets clear, and SUI's supply has been fully issued. Assume also no storage fund reinvestments, and that deletions deliver full rebates, i.e. $\gamma = \theta = 1$. Define the return on staked SUI as:

$$r = \frac{\text{StakeRewards}}{\mathbf{S} + \mathbf{F}}.$$

The market equilibrium and validator free entry conditions imply that the staking rewards associated with storing the data of transaction τ exactly cover its storage costs. In other words, for any transaction τ it must be the case that $r \times \text{StorageUnits}[\tau] \times P^S \times P^\$$ equals the dollar cost of storing the data associated with τ.

[14]This concern arises mainly because the deletion option does not apply to funds endowed through capitalizations and rebate residuals. Specifically, if network growth is such that SUI's dollar value rises over time, the storage fund will have a large number of SUI tokens associated with early storage when SUI's dollar value was low. If that occurs, this portion of the fund will become disproportionately big relative to the cost of storage but will not auto-regulate as storage fee funds do given the deletion option's absence.

The storage fund is capital efficient in the sense that a user is perfectly indifferent between the following two options:

- Paying for storage indirectly through the storage fund: A user storing τ for a finite number of epochs deposits StorageUnits $[\tau] \times P^S$ SUI tokens in the storage fund at write and receives that same amount of SUI at deletion.
- Paying for storage directly through fees in every epoch (i.e. a rent structure): A user storing τ for a finite number of epochs will pay a fee of $r \times$ StorageUnits $[\tau] \times P^S$ every epoch. This can be achieved by staking StorageUnits $[\tau] \times P^S$ units of SUI, obtaining $(1 + r) \times$ StorageUnits $[\tau] \times P^S$ of SUI at the end of the epoch, and paying the fee. When the user deletes their data, no more storage fees are charged and the user is left with StorageUnits $[\tau] \times P^S$ of SUI.

In sum, the Sui economy achieves the same outcome as a rent model in which users do not lock up SUI to store data. It is as if the storage fund invested the user's SUI profitably to pay for storage, while in the rent model users do this themselves.

While economically equivalent, Sui's design is arguably more effective since it integrates the storage model directly into the Sui economy and needs not rely on millions of users having to individually figure out how to fund their storage costs.

6.3 Stake Distribution Dynamics

A common critique of proof-of-stake systems is they promote "rich-get-richer" schemes by which the distribution of stake across validators is likely to converge to a degenerate distribution in the long run. This occurs in proof-of-stake systems where one or some of the validators obtain the full amount of stake rewards each period, and where the probability of winning is proxied by the validator's share of total stake.

The main intuition for this result is that traditional proof-of-stake enables compounding when validators reinvest their stake rewards. Consequently, a validator with high stake is more likely to start compounding earlier than validators with low stake. This effect is exacerbated over time, leading high-stake validators to end up with the majority of stake with high likelihood. Interestingly, the "rich-get-richer" effect is not driven by malicious or strategic behavior – it will arise even if all validators work honestly. The "rich-get-richer" effect is entirely driven by randomness.

Sui's proof-of-stake model does not deliver a "rich-get-richer" effect since all honest validators receive their share of the staking rewards at the end of each epoch with full certainty – that is, they are not at the mercy of randomness. This fact can be leveraged to prove that the stake distribution remains fixed over time. Formally, this can be shown in the special case where all validators submit the same price quote during the gas survey, process all their transactions in reasonable time, and that all SUI delegators and validators reinvest their stake rewards in the same validators across time:

$$\begin{aligned}\sigma_{e+1}(v) &= \frac{S_e(v) + \sigma_e(v) \times (\text{DelegatorRewards}_e + \text{ValidatorRewards}_e)}{\mathbf{S}_e + (\text{DelegatorRewards}_e + \text{ValidatorRewards}_e)}, \\ &= \sigma_e(v).\end{aligned}$$

That is, each validator $v \in V_e, V_{e+1}$ will have the same share of delegated stake at the beginning of epoch $e+1$ as the share they had at epoch e. By induction, this proof applies to all epochs e and implies the staking distribution is constant across time.

This fact is an important result for Sui's network security since the concern of some validators achieving a disproportionate amount of voting power is vastly diminished. While the above proof corresponds to a stylized setting, the result hints to a valuable force present in Sui's proof-of-stake implementation.

7 Final Thoughts

Sui's design lies at the frontier of both engineering and economic blockchain research. We look forward to working with the community and receiving your feedback on Sui's economic model at econ@mystenlabs.com.

Appendix

The following table summarizes the free system parameters of Sui's economic model. The frequency column categorizes parameters into those modified on an epoch-by-epoch basis, as described in the previous sections, and those changed infrequently through governance proposals.

Variable	Frequency	Description
$\overline{P}_e^C$	every epoch	Computation reference gas price. Set collectively by validators through the gas survey.
$\underline{P}_e^C$	every epoch	Computation floor gas price. Can be set proportionally to $\overline{P}_e^C$, e.g. $\underline{P}_e^C = \beta \overline{P}_e^C$ with $\beta < 1$.
$\overline{P}^S$	infrequently	Storage gas price. Fixed in order to target the dollar cost of storage.
δ	infrequently	Validator commission share. Can be set at the system-level or negotiated by each validator.
γ	infrequently	Share of storage fund stake rewards distributed to validators. System parameter set by governance.
θ	infrequently	Share of storage deletion rebates. System parameter set by governance.
$\underline{\alpha}$	infrequently	Bound on the maximum storage fund size. System parameter set by governance.
κ	infrequently	Stake rewards boost for low gas price submitters. System parameter set by governance.