PhronAI

Table of Contents

PhronAI

1. Introduction

Abstract

This paper introduces Phron, a groundbreaking approach to blockchain technology by integrating artificial intelligence (AI) capabilities into the foundational Layer 0. Building upon the traditional principles of decentralization, security, and scalability, an AI-based Layer 0 blockchain aims to revolutionize the landscape of distributed ledger systems.

The core of our proposed solution lies in the incorporation of AI algorithms, enabling dynamic consensus mechanisms, predictive security measures, and adaptive scalability. By leveraging machine learning, the proposed chain adapts to evolving network conditions, enhancing efficiency and responsiveness in real-time. This adaptive consensus model not only strengthens resistance against attacks but also optimizes the network's performance under various scenarios.

The proposed AI-based Layer 1 to be constructed on the master classes in Layer 0 introduces intelligent contract execution, augmenting the capabilities of smart contracts. Through integrated machine learning algorithms, the system gains the ability to autonomously optimize contract execution, predict potential vulnerabilities, and dynamically adjust gas fees based on current market conditions. This not only streamlines transaction processing but also enhances the overall security and efficiency of smart contract operations.

In this paper, we go over the design principles, technical architecture, the integration of AI master class modules into the Layer 0 blockchain, and how Layer 1 gains access to the underlying Layer 0 blockchain. We introduce novel approaches to solve Layer 1 bootstrapping issues while still securing the chain by Node - Block-Sharing (NBS). The introduction of Adaptive AI Staking (AAIS) builds on the built-in master classes to determine the node reward boost allowing for a more efficient anesthetization. Finally, we introduce the AI arbiter in the chain governance voting mechanism. The arbiter solves the long-standing issue of how to determine the voting power of the users.

We explore the impact of AI on decentralization, security, and scalability, presenting empirical evidence of improved performance through simulations and real-world use cases.

Keywords: Blockchain, AI, Layer 0, Layer 1

2. Vision and Inspiration

PhronAI is the avant-garde Layer-1 blockchain that blends EVM compatibility and Proof-of-Stake, featuring an AI-driven Consensus Mechanism. PhronAi introduces a new proprietary consensus layer, enhanced by machine learning algorithms, with its core technology, Sophia, which enables transaction processing times of under 0.9 seconds at an average cost of $0.00001 and maintains over 31,000 transactions per second, achieving unparalleled network scalability without congestion.

PhronAI is the first chain that establishes the usage of a dynamic consensus algorithm through the appliance of AI tools managed automatically by the Sophia Protocol giving an available testing sandbox to understand and improve the current AI model used for our next application. Once PhronAI is optimally refined, the technology will transition to PhronZero. PhronZero expands each chain built upon it with AI technology, granting it heightened efficiency, simplicity, and communication capabilities.

PhronAI empowers projects to create tailored solutions across various digital and real-world sectors, enabling efficient, secure, interoperable communication. This ecosystem nurtures trustless cooperation among applications, positioning PhronAI as a cornerstone for constructing a Web3 future that leverages the full potential of AI technology [1].

In the information era, blockchain and artificial intelligence are reshaping industries and redefining interactions with data and transactions. Both have emerged as transformative mediums, disrupting sectors ranging from finance to supply chain management. However, a genuine integration of the capabilities of both technologies, which would allow for a synergy that opens new possibilities, has yet to be realized.

The fusion of blockchain and artificial intelligence marks a significant leap forward in technological advancement, introducing a synergy that extends beyond the capabilities of each technology individually. Blockchain's decentralized and secure infrastructure for managing transactions and data is complemented by artificial intelligence's prowess in analyzing extensive datasets to uncover insights and streamline decision-making. This partnership not only can solve problems such as data privacy, security, and transparency but also sets the stage for the development of groundbreaking applications.

3. Problem Statement

In traditional blockchain architectures, scalability, security, and adaptability have often been cited as significant challenges. As the scale of blockchain networks grows, so does the complexity of maintaining consensus, ensuring security, and accommodating diverse transactional requirements. Moreover, existing blockchain solutions often struggle to adapt dynamically to changing network conditions, leading to inefficiencies and vulnerabilities.

PhronAI addresses these challenges by introducing an innovative approach to blockchain technology, integrating AI capabilities at its foundational layer. By leveraging machine learning algorithms, PhronAI seeks to create dynamic consensus mechanisms, predictive security measures, and adaptive scalability, thereby revolutionizing the landscape of distributed ledger systems.

SophiaExec is at the core which is actually responsible for executing the business logic of the protocol as follows:

4. Proof-of-Concept: Phron Layer 1

The Phron AI Chain represents a groundbreaking advancement in blockchain technology, where artificial intelligence is seamlessly integrated into the foundational Layer 0.

Unlike conventional blockchain architectures, which rely solely on static rules and consensus mechanisms, the Phron AI Chain harnesses the power of AI to adapt and optimize its operations in real-time. This dynamic approach not only enhances the efficiency and responsiveness of the network but also fortifies its security against evolving threats.

4.1. Phron AI: What's under the hood?

At the heart of the Phron blockchain lies PhronAI, a sophisticated amalgamation of cutting-edge technologies designed to fuel its decentralized ecosystem. PhronAI operates as the brainpower behind the platform, orchestrating various functions to ensure efficiency, security, and scalability.

At its core, PhronAI harnesses the power of artificial intelligence (AI) to optimize consensus mechanisms, enhance data validation processes, and streamline transaction throughput. Leveraging AI algorithms, PhronAI dynamically adjusts network parameters, adapting to fluctuating demands and maintaining optimal performance levels.

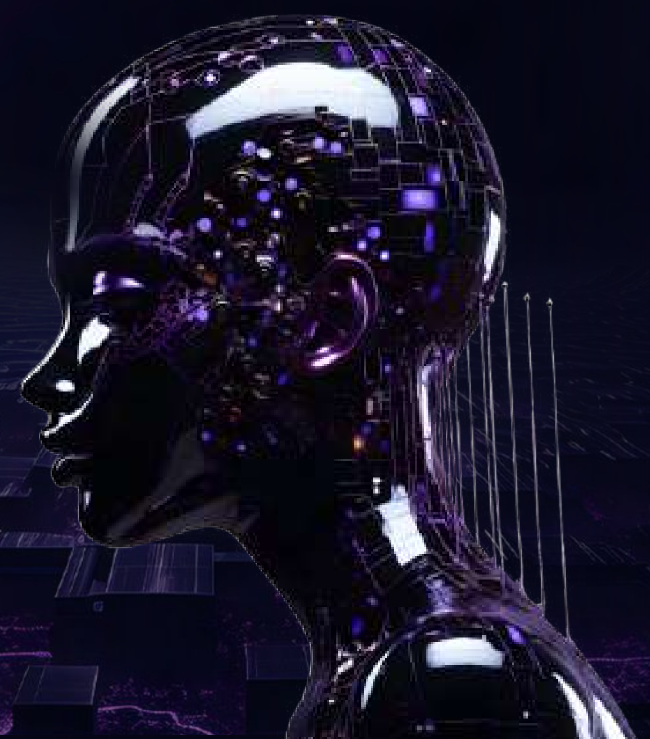
One of the key features of PhronAI is its ability to autonomously detect and mitigate potential security threats, fortifying the network against malicious activities such as DDoS attacks, double-spending, and Sybil attacks. Through continuous monitoring and analysis of network behavior, PhronAI reinforces the blockchain's resilience, safeguarding user assets and preserving the integrity of transactions.

4.2. Sophia Protocol: Statistical Consensus Algorithm

Sophia utilizes a set of rules designed to analyze and interpret the metrics data of nodes, uncovering their functional capacity to participate in the network. Through this application, three categories of validators are activated to process a broad spectrum of transactions submitted at varying fee rates. This mechanism enhances the block production process by expanding the chain's capabilities, setting the standard transaction fee remarkably low based on previously mentioned average metrics. This standard cost applies to all transactions created and submitted by end-users, ensuring their rapid processing is comparable to high-fee transactions on other blockchains.

Periodically, Sophia evaluates individual statistics and generates a list categoriz ing validators into three groups. The Deep Learning Mechanisms oversee the PhronAi block sequence holistically, identifying any anomalies and initiating a Machine Learning auto-response mechanism to mitigate the risk posed by any potentially malicious party. Furthermore, validators within these groups are tasked with processing transactions immediately, based on fee values, benefiting end-users, node owners, and developers alike.

Validator participation within the network is carefully assessed using various metrics, which, after each mechanism cycle, serve as inputs for the next. Should a validator exhibit reduced participation, its metrics are recorded as low. With low metrics as inputs, there is a possibility of category shifts among validators, from super node to fast node or average node. Consequently, PhronAi motivates validators to engage actively in the network by processing blocks that include the maximum number of transactions. The first step involves collecting inputs from useful metrics that a node calculates independently. During the network's initialization phase, these metrics, serving as input values, are supplied by the genesis file and a self-enforcing smart contract.



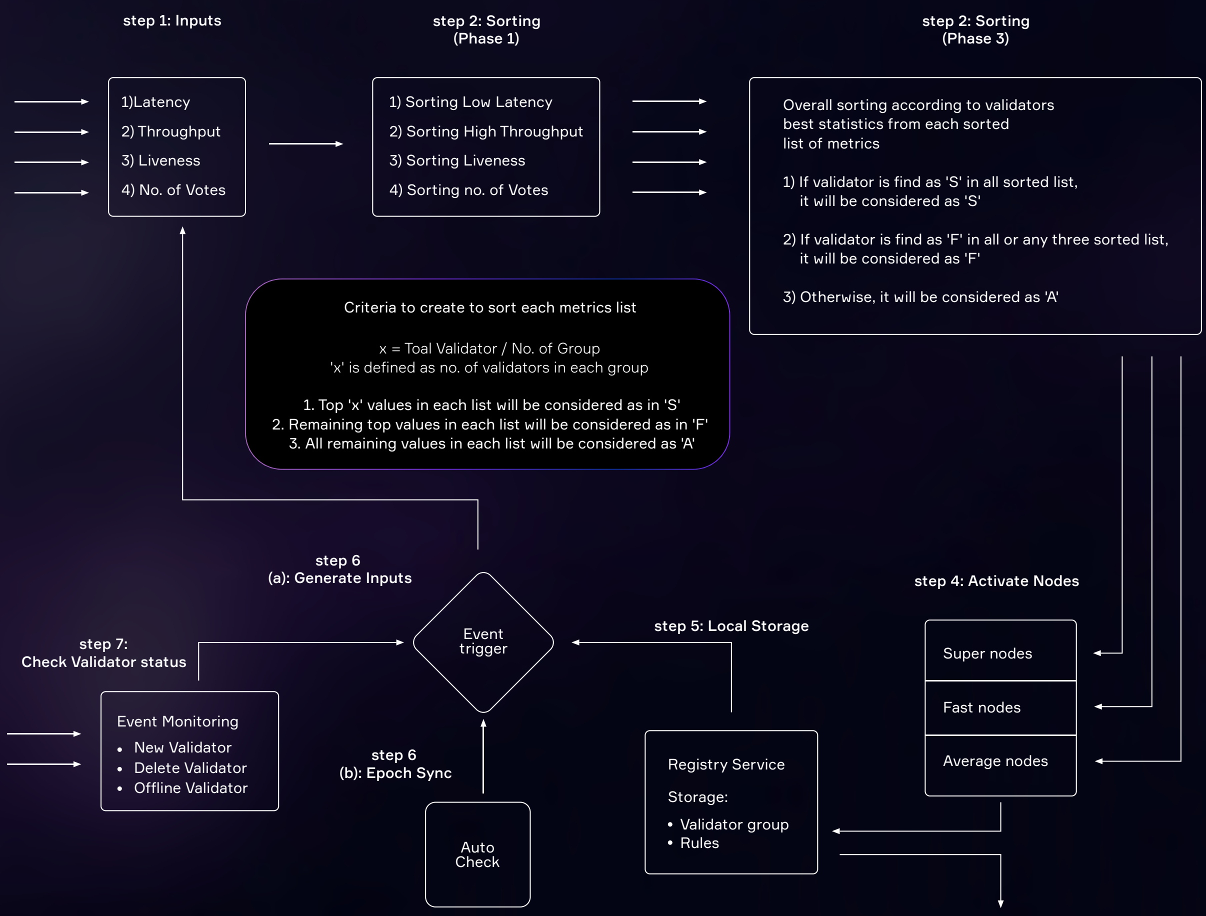


Fig. 1 Sophia Statistical Consensus Algorithm

Once the statistical algorithm becomes fully operational, metrics input values will also be directly obtained from the event trigger functionality.

Low latency, high throughput, the total number of votes, and maximum liveness metrics are sorted separately. For example, in the case of low latency, the individual matrices of all validators will be used for low-latency sorting. After this process, sorted lists from each metric are passed to the next step. The following details the criteria used in sorting each list of metrics for the super, fast, and average categories.

A variable is declared by defining an equation:

x= Total number of Validators / Number of Groups

The number of validator groups is 3. The top ’x’metrics in each sorted list are considered super metrics. Similarly, the remaining top ’x’metrics in each sorted list are considered fast metrics. All remaining metrics will be considered as average metrics.

Validators are categorized into super, fast, and average nodes based on a sorted list of all metrics. A validator is formally designated as a super node if it consistently appears as a super node in every sorted list. Similarly, a validator is classified as a fast node if it is listed as fast in at least three sorted lists. If these criteria are not met, the validator is designated as an average node. At this point, three comprehensive lists containing the IDs of all validators in the network are compiled, categorizing them as super, fast, and average nodes respectively.

In this phase, validator IDs within the super, fast, and average node categories retrieve their records from databases and caches, leading to the creation of fully functional validator objects ready to participate in the block-producing mechanisms. The registry service temporarily stores the active validator groups from these three categories. These groups are also permitted to engage in the event emission and evaluation process for a specific epoch round.

Concurrently, a self-enforcing event trigger service operates in parallel, generating input signals for the initial phase of the statistical algorithm. This service addition ally generates input signals in reaction to particular events, such as the addition or removal of a validator. Consequently, the data-capturing fields of input objects may be initialized or aligned with input matrices. Moreover, this phase is technically regarded as the concluding step of the statistical algorithm.

This final step also operates in parallel and shares its output with the continuous execution of the process already running in the previous step. Its primary purpose is to monitor the liveness of validators within each group. Should any changes in the liveness status occur, such as the addition of new validators or the removal of existing ones, a reporting event is generated and conveyed through the event trigger service

Consists of three individual modules operating within their own boundaries yet transcending the functional capabilities internally, directly governing the consensus committee and managing the transaction pool state e.g the tx queue and gas cost overhead.

A. NeuraClassi (The Arbiter)

A method for intelligently selecting an accounting node, relating to fields of blockchain, virtual currency and artificial intelligence, is provided, which includes:

1. Processing data
2. Training on processed data.

Processing data:
The raw data we have is in the form of numeric values that need to be encoded in categorical values to do so we proposed an algorithm which can convert the given array of data for a variable into its category as super, fast and average depending upon calculations. The raw data shape we get as an input and proposed Arbiter algorithm for calculation are as under.

NODE ID	POWER RATIO (PR)	LATENCY (AL)	SUCCESSFUL THROUGHPUT (ST)	LIVENESS (AL)
Node - 1	0.0714	19 (ms)	1.9 (kbps)	3000 (s)
Node - 2	0.0714	18 (ms)	1.8 (kbps)	2500 (s)
Node - 3	0.1071	22 (ms)	1.6 (kbps)	4000 (s)
Node - 4	0.1071	17 (ms)	1.3 (kbps)	500 (s)
Node - 5	0.0714	15 (ms)	1.5 (kbps)	2000 (s)
Node - 6	0.1071	17 (ms)	1.3 (kbps)	200 (s)
Node - 7	0.1428	14 (ms)	1.0 (kbps)	1500 (s)
Node - 8	0.1428	12 (ms)	1.7 (kbps)	800 (s)
Node - 9	0.1785	11 (ms)	2.1 (kbps)	5000 (s)

Arbiter Algorithm:

1. Get data of different nodes having Power Ratio, Average Latency, Successful Throughput, Liveliness.
2. Sort the values of each data column in the form of arrays.
3. Calculate the X factor using the formula:

   Xfactor = Total number of nodes / Group of nodes

   Where the total number of nodes is the number of nodes running inside the network and Group of nodes are types of nodes that want to classify. In our case we are categorizing the nodes into super, fast and average so the group of nodes is equal to 3.
4. Divide the list into X_factor sublists based on sorted values (Average, Fast, Super) in such a way that first list is assigned as Super, second list assigned as Fast and last list assigned as Average.

   sublistx-1= n=1xsorted(list)xfactor

5. If some values overlap within multiple sublists then these values should be associated with the previous sublists.
6. Encode values in such a way to assign super, fast, average category to sublist 1, sublist 2 and sublist 3 respectively.
7. As the AI models work well on numerical categories we then assign 0 to average, 1 to fast and 2 to super values in the final list.
8. The output of the data after this algorithm is as follows.

NODE ID	CATEGORY: PR	CATEGORY: AL	CATEGORY: ST	CATEGORY: L
Node - 1	0	0	2	2
Node - 2	0	0	2	1
Node - 3	1	0	1	2
Node - 4	1	1	0	0
Node - 5	0	1	1	1
Node - 6	1	1	0	0
Node - 7	2	2	0	1
Node - 8	2	2	1	0
Node - 9	2	2	2	2

AI Arbiter Model

The AI Arbiter Model is a deep learning approach designed specifically for node type detection within blockchain networks. This section outlines the architecture, mathematical formulation, training procedure, and evaluation metrics associated with the AI Arbiter Model.

The AI Arbiter protocol aims to classify nodes within a blockchain network into different types based on their behavior, role, and network attributes. By accurately identifying node types such as Super nodes, Fast nodes and average nodes, the model assists in network management to take governance decisions based on AI module output which will help to overcome the problems described above in protocols.

Model Architecture:

We propose a deep neural network architecture tailored for node type detection in blockchain networks. The model comprises multiple layers, including input, hidden, and output layers. By utilizing dense layers and appropriate activation functions, our model aims to capture intricate patterns and relationships within the input data.

Different types of layers include:

Input Layer:

1. The input data matrix X consists of features representing each node in the blockchain network. These features could include encoded parameters such as Power Ratio type , Average Latency type, Successful Throughput type, and Liveliness type.

$$X = \begin{bmatrix} x_{1,1} & x_{1,1} & \cdots & x_{1,n} \\ x_{2,1} & x_{2,1} & \cdots & x_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ x_{m,1} & x_{m,1} & \cdots & x_{m,n} \end{bmatrix}$$

Here, *m* represents the number of nodes, and n represents the number of features associated with each node.

2. Hidden Layers:

The hidden layers introduce non-linearity into the model, enabling it to capture complex relationships within the input data. Each hidden layer *l* is computed as:

$$H(l)= f(W(l)H(l\text{-}1)+b(l))$$

Where *W(l)* denotes the weight matrix, *b(l)* represents the bias vector, and *f* is the activation function applied element-wise.

3. Output Layer:

The output layer produces predictions for the node types. Since this is a multi-class classification problem, we use a softmax activation function to obtain the probability distribution over the classes. The output *Y* is computed as:

$$Y= Softmax(W(out)\ H(L)+b(out))$$

Where W(out) and b(out) are the weight matrix and bias vector for the output layer, respectively.
L denotes the index of the last hidden layer.

Training Procedure:

The model is trained using a suitable optimization algorithm such as stochastic gradient descent (SGD), Adam, or RMSprop. The objective is to minimize a suitable loss function such as categorical cross-entropy, which measures the dissimilarity between the predicted probabilities and the true labels. The annotated data sample shown above is used during the training process.

Evaluation Metrics:

To assess the performance of the model, evaluation metrics such as accuracy, precision, recall, and F1-score can be computed on a held-out validation set or through cross-validation.

B. NeutraGuard

1. Performs as two way watchguard between NeuraClassi and SophiaExec, providing **Guardrails** to control any **Emergent, Hallucinative or Suspicious** behavior produced from relative modules, takes some predetermined measures and falls back to LockDown state to mitigate the situation as:

    a. Blocking AI influence to consensus or tx pool conditionally

    b. Switching to the state of defaults to keep network going smoothly
        i. The state could be triggered in case of any anomaly detection

    c. Controls how long the LockDown stays or it could only be restored through manual governance agreement.

2. NeutraGuard is also incharge to control the behavior of AI compliance with on-chain state; in case of any disagreement between the states from both modules the protocol will again fallback to LockDown state.

3. And as data channel telemetry across Sophia Protocol

NeutraGuard Anomaly analysis:

Isolation Forests(IF), similar to Random Forests, are built based on decision trees. And since there are no predefined labels here, it is an unsupervised model. Isolation Forest is a technique for identifying outliers in data. The approach employs binary trees to detect anomalies, resulting in a linear time complexity and low memory usage. Isolation. Isolation Forests were built based on the fact that anomalies are the data points that are “few and different”. In an Isolation Forest, randomly sub-sampled data is processed in a tree structure based on randomly selected features. The samples that travel deeper into the tree are less likely to be anomalies as they require more cuts to isolate them. Similarly, the samples which end up in shorter branches indicate anomalies as it was easier for the tree to separate them from other observations.

Isolation Forests for outlier detection are an ensemble of binary decision trees, each referred to as an Isolation Tree (iTree). The algorithm begins by training the data through the generation of Isolation Trees.

The flow is as follows:

1. Given a dataset *D*, a random subsample **Ds** is chosen and assigned to a binary tree.

2. Tree branching initiates by randomly selecting a feature from the set of all *N* features. Subsequently, branching occurs based on a random threshold within the range of minimum and maximum values of the selected feature.

    - Let *xi* denote the feature vector of data point i, and feat be the randomly selected feature.
    - Let thresh be a random threshold.
    - Branching condition: If **xi[feat]<thresh**, then go to the left branch; otherwise, go to the right branch.

3. If the value of a data point is less than the selected threshold, it follows the left branch; otherwise, it proceeds to the right branch. This process splits a node into left and right branches.

    - Let left(n) and right(n) denote the left and right branches of node n, respectively.

4. The process continues recursively until each data point is entirely isolated or until the maximum depth (if defined) is reached.

    - Let *max_depth* be the maximum depth of the tree.
    - Recursive termination condition: If *max_depth* is reached or only one data point remains in the node, stop recursion.

5. The above steps iteratively construct multiple random binary trees.

    - Let T be the set of all constructed trees.

6. Once the ensemble of iTrees (Isolation Forest) is formed, model training concludes. During scoring, each data point traverses through all previously trained trees. Subsequently, an 'anomaly score' is assigned to each data point based on the depth of the tree required to reach that point. This score aggregates the depths obtained from each of the iTrees. An anomaly score of -1 is assigned to anomalies, and 1 is assigned to normal points, based on the provided contamination parameter, which signifies the percentage of anomalies present in the data.

C. SophiaExec

SophiaExec is at the core which is actually responsible for executing the business logic of the protocol as follows

1. Block Authoring / Finalizing Committee

Receives the nodes list in a categorized manner and let each node author blocks according to its position in the list, the total block capacity within the given timeframe will be divided as percentages, the nodes will get to author blocks according to their performance measured by the **NeuraClassi** module. For finality all nodes can take part except for the banned nodes without any distinction for the time being.

Validators categorized for NeutraGuard:

- `validator`: Node that can become a member of committee (or already is) via rotation.
- `validators reserved`: immutable validators, i.e. they cannot be removed from the list.
- `validators non_reserved`: validators that can be banned from the list.

There are two options for choosing validators during election process:

- `Permissionless`: choose all validators that are not banned.
- `Permissioned::reserved`: choose only reserved validators.
- `Permissioned::non_reserved` choose only non_reserved that are not banned.

These conditions provide help in LockDown situations as fallback for different sets of validators.

2. Validator Bans

In order to manage underperforming / misbehaving nodes the ban logic is required which is currently being managed by the root (pallet elections will be replaced once the NeuraClassi matures to involve in committee decisions) but the Ai module is also responsible to ban the misbehaving nodes or choose the committee set according to the given stats.

3. TxPool Rearrangements

Security concern:

Networks that process low fee transactions are always vulnerable to pool flooding aka DDOS attacks for these scenarios the transaction pool and queue is carefully designed to deal with.

Computation:

Tx in Pool (%) / PoolLimit(%) * MaxTTL = TTL
Pool Capacity - Total Tx in Pool / Pool Capacity * MaxTTL * Fee = Priority Factor

- If PF equals or less than last Tx in the pool it won't be accepted and if it beats one in that case the last transaction will be dropped.
- If TTL of a transaction expires that will be dropped automatically.
- All other transactions will be put into the ready and future queue according to the status tags.

Priority Factor will let some extremely low fee transactions pass through, as lucky transactions.

Note: The hardware metrics from the node are not supposed to be stored on chain but the resulting decision from the Ai module is, which will be considered a metric itself for further decision making.

Example - Flooding

Pool Limit = 5000, MaxTTL = 2h (7200000 ms)

INDEX	TX IN POOL	TTL (H/MS)	FREEPAID	PF	STATUS	ACCEPTANCE
1	0	2.0000	0.0001	0.0002	Active	100%
2	1	1.9992	0.0001	0.00019996	Active	99.96%
3	2	1.9984	0.0001	0.00019992	Active	99.92%
4	3	1.9976	0.0001	0.00019988	Active	99.88%
...	...	...	...	...	...	...
5000	4999	0.0400	0.0001	Active	Active	2%
5001	5000	0	0.0001	Dropped	Dropped	0%

Example - Lucky Transaction

INDEX	TX IN POOL	TTL (H/MS)	FREEPAID	PF	STATUS
1	0	2.0000	0.01	0.02	Active
2	1	1.9984	0.02	0.039968	Active
3	2	1.9992	0.002	0.0039984	Active
4	3	1.9976	0.001	0.0019976	Active
...	...	...	...	...	...
4999	4998	0.0008	0.003	0.00000240	Active- Lucky
5000	4999	0.0004	0.0059	0.00000236	Active

4.2.1. AI Arbiter

The voting weight issue in blockchain governance revolves around the complexity of determining the influence each participant holds in decentralized decision-making within a blockchain network [8, 9]. In decentralized governance systems, such as those prevalent in blockchain projects, decisions concerning protocol upgrades, changes, or community initiatives are typically made through a voting mechanism [10]. The introduction of an AI arbiter within Phron's governance system revolutionizes this aspect by harnessing advanced artificial intelligence algorithms. Unlike conventional methods reliant on static metrics like token holdings or stake sizes, the AI arbiter considers a diverse array of dynamic factors to fairly allocate voting influence to each participant.

The incorporation of an AI arbiter within the governance voting mechanism of the Phron chain signifies a breakthrough in addressing the persistent challenge of determining users' voting power in decentralized decision-making processes. Traditionally, this issue has sparked debates regarding fairness, transparency, and susceptibility to manipulation. One pivotal advantage of employing an AI arbiter lies in its capability to analyze intricate datasets and discern patterns, trends, and user behaviors that may elude human observation. Through machine learning techniques, the AI arbiter continually adapts and improves, ensuring precise and equitable distribution of vot ing power over time. The AI arbiter introduces objectivity and impartiality, lacking in human-driven governance systems. By eliminating biases and subjective judgments, it guarantees decisions are based solely on merit and community interests, rather than individual inclinations.

From an efficiency point of view, the AI arbiter enhances governance efficiency and scalability by automating essential tasks such as voter registration, verification, and vote tabulation. This not only streamlines decision-making but also mitigates the risk of human error or manipulation.

Beyond its role in determining voting power, the AI arbiter offers valuable insights and recommendations to inform governance decisions. By analyzing historical voting patterns and market data, aids users in making informed decisions aligned with Phron chain's long-term objectives and sustainability.

4.2.2. Indirect – LTFM Protocol

Sophia implements efficiently a set of processes to manage transactions while avoiding triggering spikes in transactions costs. Implemented indirectly as Low Transaction Fee Management (LTFM) Protocol. The transaction fees are being reduced by an incredible magnitude of 8X, the number is sure to make the chain even cheaper to interact with. This makes the blockchain even more suitable for accommodating, High-Frequency DeFi or other large-scale use cases.

But still dynamic fee adjustment is unavoidable. They create the financial incentives for providing services and guarantee the further development of the project in question.

So theoretically If the block has fewer transactions than the targeted block saturation, the price will diminish by a minor amount. If a block is subject to more transactions, the fees will be accordingly priced higher.

4.3. Adaptive AI Staking (AAIS)

Blockchain staking is a mechanism used to secure and validate transactions on a blockchain network, as well as to incentivize network participants to actively contribute to the network's operation. Staking involves users locking up a certain amount of cryptocurrency tokens as collateral to participate in the network's consensus process. In return for staking their tokens, participants are rewarded with additional tokens as an incentive for helping to maintain the network's security and integrity [11].

Blockchain staking offers several advantages over traditional Proof of Work (PoW) consensus mechanisms, including reduced energy consumption, scalability improvements, and potentially enhanced decentralization. Additionally, staking enables cryptocurrency token holders to earn passive income by contributing to network validation, thereby encouraging sustained investment and involvement in blockchain ecosystems [12].

Phron utilizes Adaptive AI Staking (AAIS), introduced by Dr. Adel ElMessiry, which is an innovative approach to blockchain staking that leverages artificial intelligence (AI) algorithms to dynamically adjust staking parameters based on real-time network conditions, user behavior, and market dynamics. This methodology aims to optimize staking rewards, mitigate risks, and enhance the efficiency of the staking process. The main characteristics of AAIS are expanded in the following sections.

4.3.1. Dynamic Staking Parameters

AAIS utilizes AI algorithms to continuously analyze various factors such as network congestion, transaction volume, token price movements, and user participation. Based on this analysis, AAIS dynamically adjusts staking parameters such as staking duration, reward rates, and token allocation to maximize returns and adapt to changing network conditions.

4.3.2. Risk Management

AAIS incorporates risk management strategies to mitigate potential losses and protect stakers' interests. AI algorithms monitor market volatility, security threats, and other risk factors, and automatically adjust staking parameters to minimize exposure to risks such as price fluctuations and network vulnerabilities.

4.3.3. Customizable

AAIS prioritizes the interests of stakers by tailoring staking parameters to individual preferences, risk tolerance, and investment goals. Users have the flexibility to customize their staking preferences and adjust parameters such as staking duration, reward distribution frequency, and withdrawal options to suit their needs.

4.3.4. Optimized Reward Distribution

AAIS optimizes reward distribution mechanisms to ensure fair and equitable distribution of staking rewards among participants. AI algorithms dynamically adjust reward rates based on factors such as staking duration, token holdings, and network contribution, incentivizing active participation and encouraging long-term engagement.

4.3.5. Continuous Learning

AAIS incorporates machine learning techniques to continuously learn from past performance, user feedback, and market data to refine its algorithms and improve staking efficiency over time. By analyzing historical data and identifying patterns, AAIS can make more accurate predictions and better optimize staking parameters to maxi mize returns for participants. The end goal of AAIS is to adjust the required staking amount and the rewards in a manner that rewards user behavior conducive to the entire ecosystem over the long run.

5. PhronZero: Decentralizing Blockchain Development

A Layer 0 blockchain, sometimes referred to as a "protocol layer" or "foundational layer," represents the underlying infrastructure upon which other blockchain layers operate. Unlike Layer 1, which typically encompasses blockchains like Bitcoin and Ethereum, Layer 0 is not concerned with specific applications or consensus mechanisms. Instead, it focuses on fundamental protocols and infrastructure components that provide the backbone for decentralized networks to function efficiently and securely [2].

Phron Zero blockchain provides the foundational infrastructure and protocols that enable the functioning of decentralized networks. It sets the stage for innovation and development at higher layers, empowering developers to build decentralized applications (dApps), decentralized finance (DeFi) protocols, and other blockchain-based solutions on top of a robust and secure foundation. The main innovations of PhronZero are Master Class and Node-Block-Sharing (NBS).

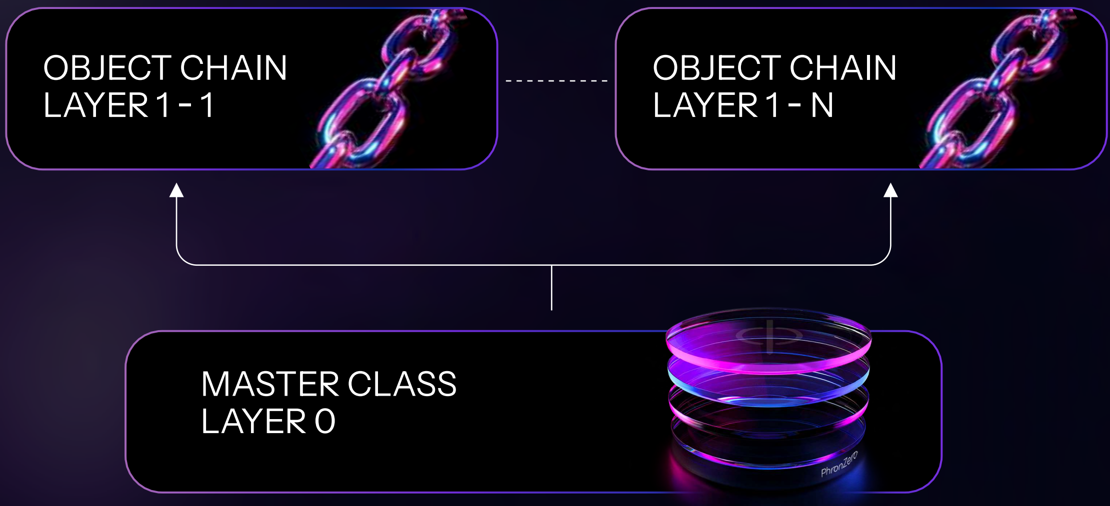
5.1. Master Class

The concept of classes in modern programming languages serves as a cornerstone for creating reusable structures from which objects can be instantiated [3, 4]. Extending this paradigm to the realm of blockchain, the Phron Master Class introduces a pioneering approach to enhancing blockchain functionality and extensibility.

At its core, the Phron Master Class enables the blockchain to evolve and adapt by incorporating new functionalities through the addition of Master Classes. These Master Classes undergo a rigorous validation and regression testing process to ensure their compatibility and reliability within the blockchain ecosystem. Once validated, Master Classes are seamlessly integrated into the chain, becoming readily available for utilization by smart contracts.

The adoption of a Master Class is not only a technical decision but also an economic one. To invoke a Master Class, users are required to pay gas fees in addition to any other fees stipulated by the creator of the Master Class. This token economics frame work is meticulously designed to incentivize the creation of Master Classes and foster a vibrant ecosystem of innovation and collaboration within the blockchain community. By allowing Master Classes to be adopted into the chain, Phron empowers developers and stakeholders to introduce novel functionalities, optimizations, and improvements to the blockchain network.

Whether it's introducing advanced cryptographic techniques, implementing complex algorithms, or enhancing interoperability with external systems, Master Classes serve as the building blocks for unlocking new capabilities and driving the evolution of blockchain technology.



The integration of Master Classes fosters a culture of openness and collaboration, where developers can contribute their expertise and innovations to the broader blockchain ecosystem. Through a transparent and inclusive validation process, PhronAI ensures that Master Classes meet the highest standards of quality and reliability, thereby instilling confidence in their adoption by smart contracts and applications.

The PhronAI Master Class represents a paradigm shift in blockchain development, offering a scalable and extensible framework for incorporating new functionalities and innovations. By incentivizing the creation of Master Classes and fostering a collaborative ecosystem, Phron paves the way for the continuous evolution and advancement of blockchain technology.

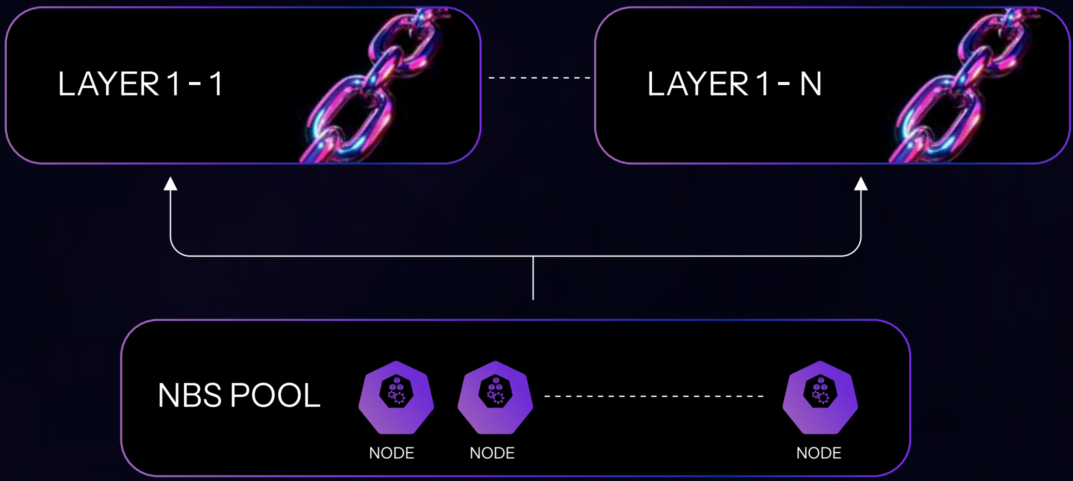
5.2. Node-Block-Sharing (NBS)

The bootstrapping issue and lack of sufficient incentives for node operations are common challenges faced by Layer 1 blockchain networks [5]. These issues can impede the growth and sustainability of newly deployed chains, limiting their ability to operate securely and efficiently [6]. Node Block Sharing (NBS) presents a novel solution to these challenges, offering a comprehensive approach to address not only bootstrapping but also enhancing transaction processing and incentivizing node participation.

NBS leverages the underlying infrastructure of Phron Zero, a foundational protocol shared by all Layer 1 chains built upon it. This interoperability ensures cross-compatibility of transaction hashing, enabling nodes to seamlessly mine transactions across multiple Layer 1 chains. By tapping into a shared pool of participating nodes, each chain can access the necessary computational resources to process transactions efficiently, regardless of its individual node count.

One of the key benefits of NBS is its ability to address the bootstrapping issue by providing a decentralized network of nodes ready to support newly deployed chains. Instead of relying solely on the native node population of a specific chain, newly launched networks can leverage the existing infrastructure of Phron Zero, significantly reducing the time and resources required for network initialization and stabilization. NBS introduces a mechanism for optimizing mining rewards and transaction processing fees based on node status and network demand. By dynamically adjusting mining fees according to node availability and performance, each chain can incentivize node participation while ensuring fair compensation for computational resources contributed. This approach not only promotes a healthy ecosystem of node operators but also enhances the overall security and efficiency of transaction processing across Layer 1 chains.

To further enhance the efficiency and effectiveness of NBS, an AI agent can be employed to optimize the return on investment (ROI) for each participating node. By analyzing network dynamics, transaction volumes, and node performance metrics, the AI agent can dynamically adjust mining strategies and fee structures to maximize profitability for individual nodes while maintaining network stability and security.

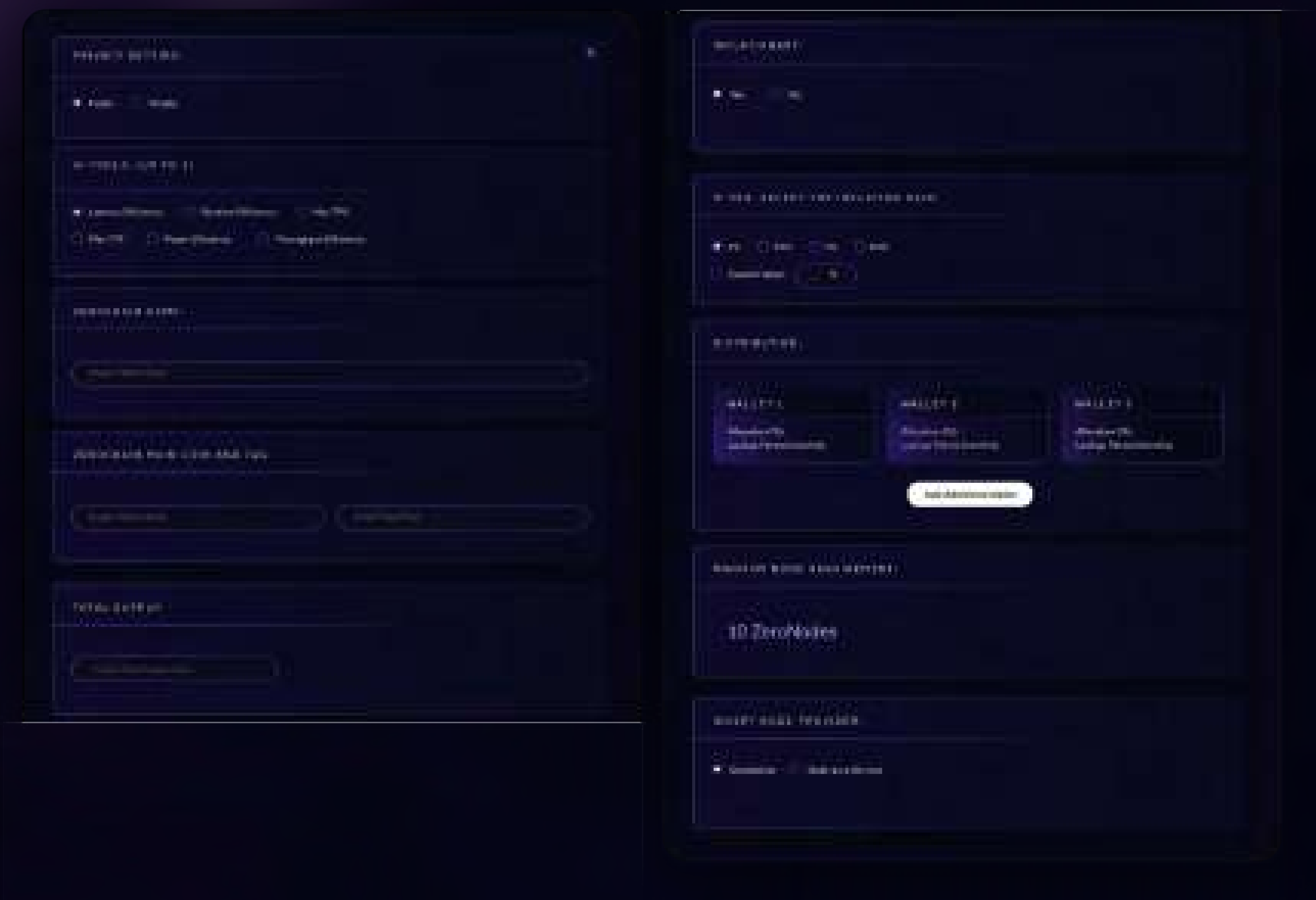


Node Block Sharing represents a pioneering approach to addressing the bootstrapping issue and incentivizing node participation in Layer 1 blockchain networks. By leveraging cross-chain compatibility, dynamic fee structures, and AI-driven optimization, NBS offers a scalable and sustainable solution to the challenges facing decentralized blockchain ecosystems.

5.3. Layer 1 Blockchain Minter Dashboard

Layer 1 Blockchain Minter implementation allows the user to create with a simple array of parameters the configuration of the future blockchain. This blockchain will work under the infrastructure and master classes available of PhronZero, permitting security, efficiency and intercommunication with other Layer 1 Blockchains.

ZEROCHAIN (LAYER 1) MINTER



6. Token Economics

6.1. Introduction

PhronZero presents a pioneering approach in the integration of blockchain and artificial intelligence (AI), offering a layer 0 infrastructure designed to empower layer 1 blockchains with unprecedented AI capabilities. The model aiming to ensure the sustainability, growth, and decentralized governance of the ecosystem and is constructed to incentivize participation, secure the network, and facilitate a vibrant economy centered around AI and blockchain synergy.

6.2. Purpose of the Token

- **Transaction Fees:**
  Used to pay for transactions and services within the PhronZero ecosystem, including smart contract deployments and AI service calls.

- **Staking:**
  Required for participating in network consensus as validators, securing the network, and earning rewards.

- **Governance:**
  Grants holders the right to vote on proposals concerning the network's development, feature integrations, and use of the ecosystem fund.

- **AI Services Access:**
  Enables access to advanced AI capabilities and services provided by PhronZero, acting as a payment mechanism within the AI marketplace.

6.3. Deflationary Mechanism

Transaction Fee Burns:
A portion of transaction fees (e.g., 0.5%) is burned, reducing the total supply over time and creating deflationary pressure.

AI Service Fee Burns:
Similar to transaction fees, a portion of the fees paid for using AI services within PhronZero will be burned.

6.4. Staking and Validator Incentives

Dynamic Staking Rewards:
Adjusted based on network participation levels, total staked amount, and overall network performance to ensure attractive yet sustainable reward levels [14].

6.5. Dynamic Gas Fee Model

Optimizing for network efficiency and user experience, we employ a dynamic gas fee model, drawing inspiration from Ethereum's EIP-1559 [15], formulated thus:

$$G(t) = \int_0^t \text{BaseFee}(t) + \text{Tip} + \epsilon \frac{P_{target} - P(t)}{P(t)} \cdot (\Delta C(\tau) + \Delta N(\tau))\, d\tau \qquad (1)$$

- **BaseFee(t):** Dynamically adjusts based on block space utilization, ensuring adaptability to network demand.

- **Tip:** An optional incentivization for validators to prioritize transactions, enhancing throughput during peak times.

- **$\Delta C(\tau)$ and $\Delta N(\tau)$:** Represent the rate of change in transaction complexity and network congestion, respectively.
- **ϵ**: A sensitivity parameter for the token price stabilization mechanism.

- **Ptarget and P (t):** Target and current token prices, guiding fee adjustments to market conditions.

6.6. Storage Fee Formulation

Reflecting considerations of data size, redundancy, and depreciating storage costs over time:

$$S(D, t, R) = StorageBaseFee \cdot D \cdot R \cdot e^{-\lambda t} \tag{2}$$

- **StorageBaseFee:** Cost per unit of data storage.
- D: Size of the data stored.
- R: Redundancy factor for data reliability.
- λ: Reflects decreasing storage technology costs over time.

6.7. Fee Distribution Mechanism

Encouraging a collaborative network through a model that rewards validators based on performance:

$$F_{Layer1} = F \cdot (\alpha + \beta P) \tag{3}$$

$$F_{Layer0} = F \cdot (1 - (\alpha + \beta P)) \tag{4}$$

- F : Total transaction fees collected.
- α: Base coefficient for fee distribution between layers.
- β: Adjusts distribution based on validator performance.
- P : Performance metric for validators.

6.8. Staking Rewards Dynamics

Enhancing network security and stakeholder engagement via a dynamic staking rewards model:

$$R(S, t) = I \; S(t) + \gamma V(t) + \theta \; \frac{P_{target} - P(t)}{P(t)} \; S(t) \tag{5}$$

- I: Inflation rate for reward distribution.
- S(t): Total amount staked.
- γ: Validator performance coefficient.
- θ: Token price stabilization coefficient.

These mechanisms are crafted to ensure the blockchain remains adaptable, efficient, and economically sustainable, fostering a robust ecosystem conducive to long-term stability.

6.9. Dual Token Architecture

A dual token architecture in blockchain refers to a system where there are two distinct types of tokens operating within the same ecosystem, typically with one token serving as the default base currency and another as a customizable token specific to individual layers or chains built on top of the base protocol. In this scenario, let's explore the architecture with Phron Zero as the default token [1, 16].

6.9.1. Phron Zero Token (Default Token)

Phron Zero token serves as the default base currency within the blockchain ecosystem. It is used for various purposes such as transaction fees, rewards, and value exchange within the network. Phron Zero token is the foundational token upon which the entire ecosystem is built. It ensures interoperability and consistency across different layers and chains within the ecosystem.

6.9.2. Layer 1 Custom Tokens

Any layer 1 blockchain built on top of Phron Zero can opt to incorporate a custom token specific to its chain. These custom tokens can have their own unique features, use cases, and economic models tailored to the specific requirements of the layer or chain. Custom tokens can be used for various purposes including governance, utility, incentivization, and more within their respective chains.

6.9.3. Interoperability Between Layer 1

The dual token architecture ensures interoperability between the default Phron Zero token and the custom tokens. Users can seamlessly transact and exchange value between different chains and layers within the ecosystem, irrespective of the specific

tokens being used. Smart contracts and protocols are designed to accommodate both Phron Zero and custom tokens, facilitating smooth interactions between them. Think of it as the reserve currency of the global monitor system.

6.9.4. Integration and Development

Developers building on top of Phron Zero can choose to integrate the default token or create custom tokens specific to their applications or layer 1 blockchains. Development frameworks, APIs, and toolkits are provided to simplify the process of token creation and integration, enabling developers to focus on building innovative solutions.

6.9.5. Economic Model and Governance

The economic model of the ecosystem may involve mechanisms for governing the issuance, distribution, and utilization of both Phron Zero and custom tokens [17]. Governance structures ensure that the interests of token holders and participants are aligned with the overall goals and sustainability of the ecosystem. In summary, a dual token architecture in blockchain, with Phron Zero as the default token, allows for flexibility, customization, and interoperability within the ecosystem. It empowers developers to build diverse applications and layer 1 blockchains while maintaining a cohesive network supported by the foundational Phron Zero token.

6.10. Incentives

An initial rewards curve for a newly launched protocol focuses on encouraging the use of a higher proportion of trusted Layer-0 (L0) nodes during the critical early stages of development and operation. This approach is strategically advantageous because it ensures a more secure and stable launch by leveraging the established security and reliability of L0 nodes. By incentivizing the utilization of these nodes through a rewards structure that makes it more cost-effective to use more L0 nodes rather than fewer, the protocol can maintain integrity and trustworthiness in its nascent phase.

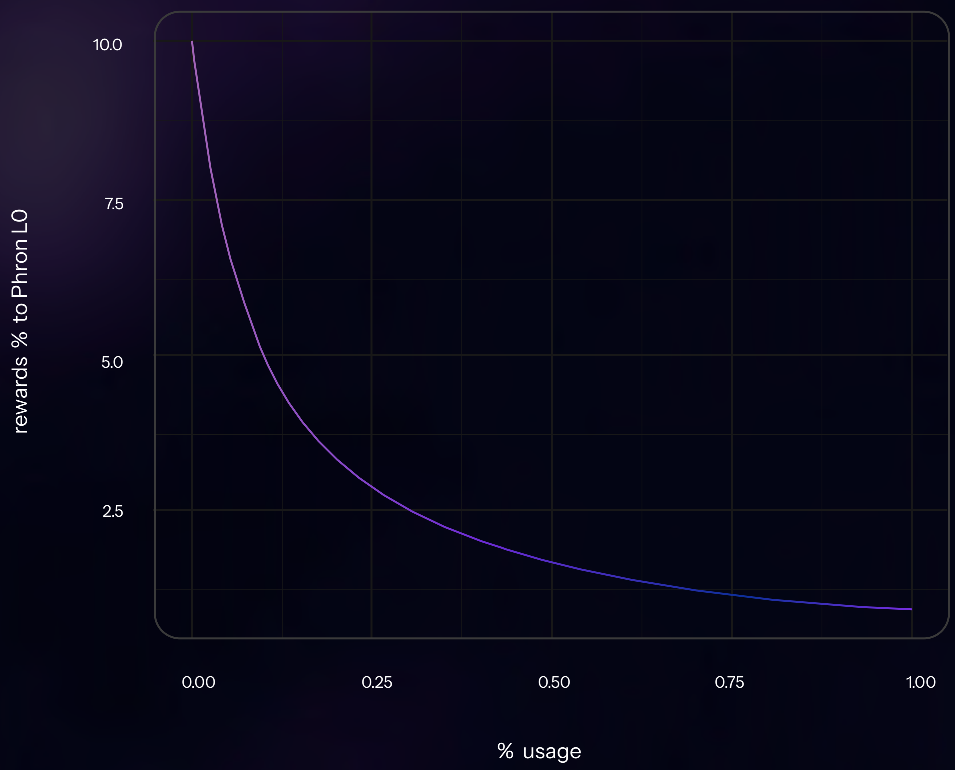


Fig. 2 Bootstrapping Phase Rewards Curve Model.

As the project matures and gains stability, trust, and a wider validator base, the rewards curve can transition. This change reflects the growing confidence in the protocol's own Layer-1 (L1) validators and a deliberate shift towards encouraging a more decentralized model. The upsloping rewards curve now incentivizes a gradual reduction in dependency on L0 nodes, rewarding the protocol for diversifying its validator network. This evolution in the incentives curve aligns with the project's development trajectory, from relying on the foundational security of L0 nodes to fostering its autonomous, decentralized security apparatus as it matures.

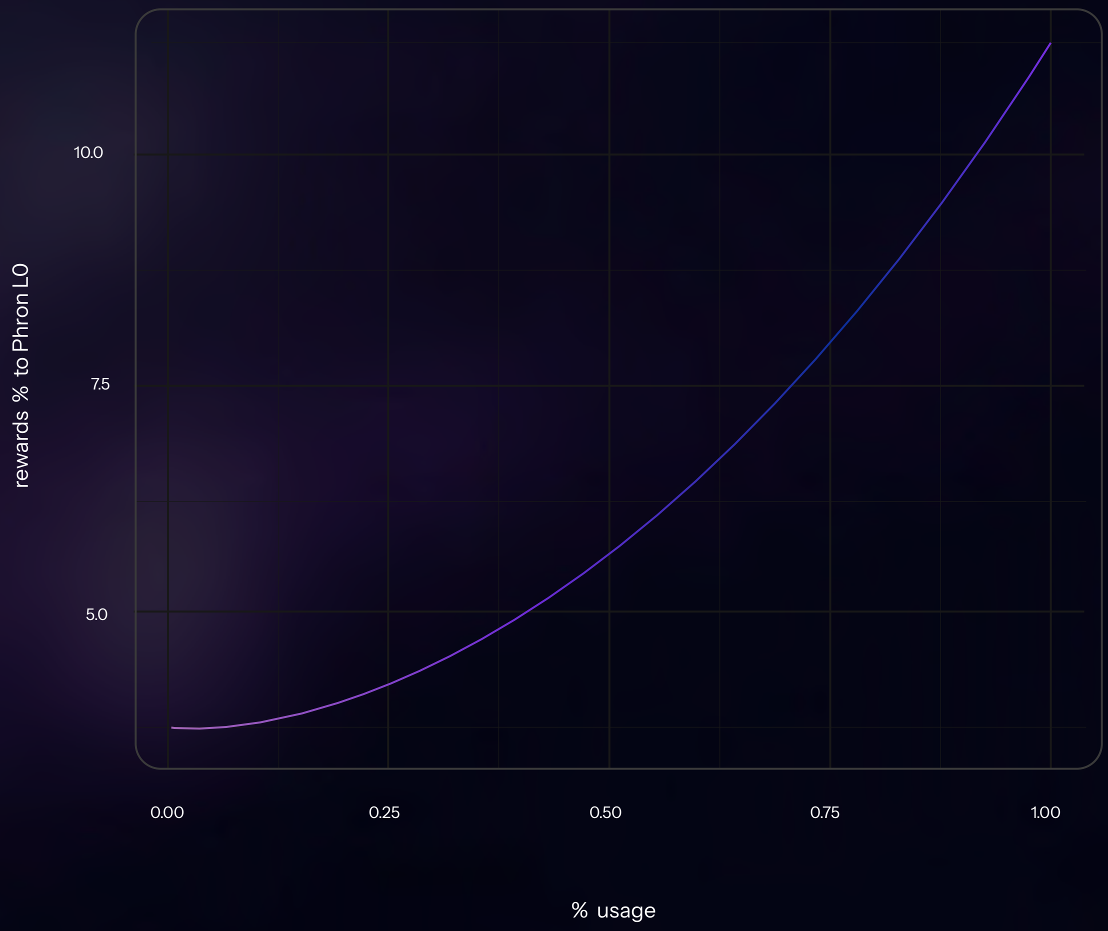


Fig. 3 Stability Phase Rewards Curve Model.

7. Staking

The Phron blockchain employs a novel approach to validator selection, leveraging both user staking and AI-ranked performance to ensure a robust, fair, and meritocratic system. This method prioritizes high-performing nodes while incorporating community trust and randomness to democratize the selection process.

7.1. Meritocratic Selection

Validators are chosen based on a combination of AI-generated performance metrics and user staking, promoting a system where merit and community trust determine validator selection. The AI scores and ranks nodes by their operational efficacy, creating a competitive environment that motivates validators to uphold high standards. This merit-based selection system ensures that the most reliable and efficient validators are prioritized.

7.2. Community Participation

Incorporation of user staking into the validator ranking allows the community to have a direct influence on the selection process. Validators that receive higher stakes from the community are perceived as more trusted, thereby integrating a democratic element into the system. This approach aligns the network's operation with the preferences and trust of its users.

7.3. Fairness and Randomness

To further ensure fairness, the system includes a random lottery element that considers both the AI rankings and the percentage of stakes. This mechanism introduces a degree of randomness, mitigating biases and providing opportunities for newer or smaller validators to participate in network validation.

7.4. Selection Algorithm

The final decision on validator selection is based on the following factors:

1. The AI ranking of a node k, denoted as R^k_{AI}, ranks nodes in ascending order based on their performance, with the best nodes receiving a higher rank.

2. The stake-based ranking of a node k, denoted as R^k_{Stake}, applies the same ranking principle, prioritizing nodes with higher stakes.

3. A random lottery that accounts for both AI and stake-based rankings, calculating the probability P_k for a node k to be selected as a validator.

The algorithm is formalized by the following equations:

$$A = \sum_{k=1}^{n} R^k_{AI}$$

$$B = \sum_{k=1}^{n} R^k_{Stake}$$

$$P_k = \frac{R^k_{AI} + R^k_{Stake}}{A + B}$$

where A is the sum of AI ranks for all nodes, B is the sum of stake-based ranks, and P_k represents the selection probability of node k.

7.5. Reward System

PHRON constitutes the link between the PhronAi Ecosystem and the holder. The PHRON reward system for node validators will work with the APR method; it is described with the following equation:

$$y = -\log_{15\times10^{15}}\left(\frac{x}{2000} + 0.0001\right) - 0.1 \tag{6}$$

where:

- $0 \leq x < \infty$
- y is the APR (in decimal form)
- x is the time (in years)

This APR calculation factors into the broader staking and reward mechanism, ensuring validators are incentivized proportionally to their commitment and performance over time.

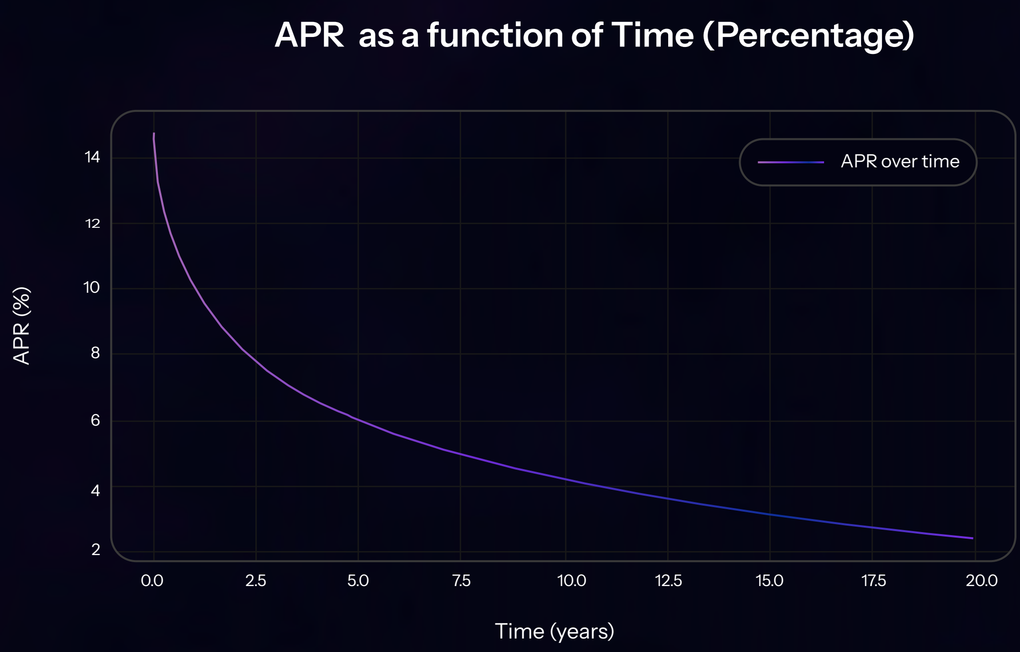


Fig. 4 APR calculation

8. Governance

In an endeavor to ensure a high degree of decentralization, PhronAI introduces an on-chain governance model, predicated on a vote-escrowed mechanism. This approach empowers token holders to influence the ecosystem dynamically, aligning with the principles of decentralized autonomous organization (DAO) governance.

8.1 Vote-Escrowed Tokenomics

Vote-escrowed tokenomics grants token holders the autonomy to determine a lock-up period for their tokens, effectively tying the token's utility to the duration of its lock. The extended commitment to lock up tokens translates into enhanced influence within the network, manifesting in:

- Enhanced governance voting power.
- Increased staking rewards.
- Amplified voting impact on specific liquidity pools.

veTokens Align Incentives:
The veToken model is designed to synchronize the protocol's success with that of the token holders'. A prolonged lock-up period symbolizes a vested interest in the protocol's prosperity.

VeTokens Encourage DAO Participation:
By offering additional voting power for longer lock-up periods, the protocol incentivizes users to "max time-lock" their tokens, thereby strengthening their governance voice. This mechanism ensures that token holders deeply invested in the DAO's future are rewarded with greater influence.

8.2. Quadratic Voting

To further democratize the governance process and mitigate the risks of centralization and collusion, PhronAI will employ a quadratic voting system. This system ensures that as token holders acquire more tokens, the marginal increase in their voting power diminishes, promoting a more equitable distribution of governance influence.

The governance voting power, V , is determined by the equation:

$$V = \sqrt{R \times L}$$

where:

- V represents the vePhron balance, indicating the voting power.
- R denotes the Phron native token quantity.
- L is the token lock-up period multiplier, enhancing the token's voting power.

The vePhron balance declines linearly from the initiation of the lock-up period to its end, at which point stakeholders can reclaim their Phron tokens. However, token holders are afforded the flexibility to extend or renew their lock-up duration at any juncture, enabling them to either augment or maintain their vePhron balance and, by extension, their governance influence.

9. Chain Simulations

9.1. Transaction Throughput Simulation

To evaluate the robustness and scalability of the foundational Layer 0 network, we performed a simulation of transaction throughput for several Layer 1 projects running concurrently. The primary aim was to observe the network's ability to manage and distribute its transaction processing capacity among the projects.

9.1.1. Simulation Parameters

The simulation was conducted under the following assumptions:

Layer 0 Capacity: The maximum transactions per second (tps) capacity was set at 31,000, reflecting a high-throughput blockchain infrastructure.

Even Distribution: The Layer 0 network's tps capacity was evenly divided among the Layer 1 projects, emulating a fair and balanced load-sharing protocol.

Temporal Scope: The simulation covered a 100-second timeframe, providing a snapshot of network activity in a high-velocity environment.

9.1.2. Throughput Simulation Results

The throughput simulation [18] results (Figure 5) showcased the transactions processed by each Layer 1 project over time. The graphical representation illustratedmthat despite the fluctuations typically observed in network conditions, each project maintained a consistent level of activity, indicating a resilient and well-dimensioned network infrastructure.

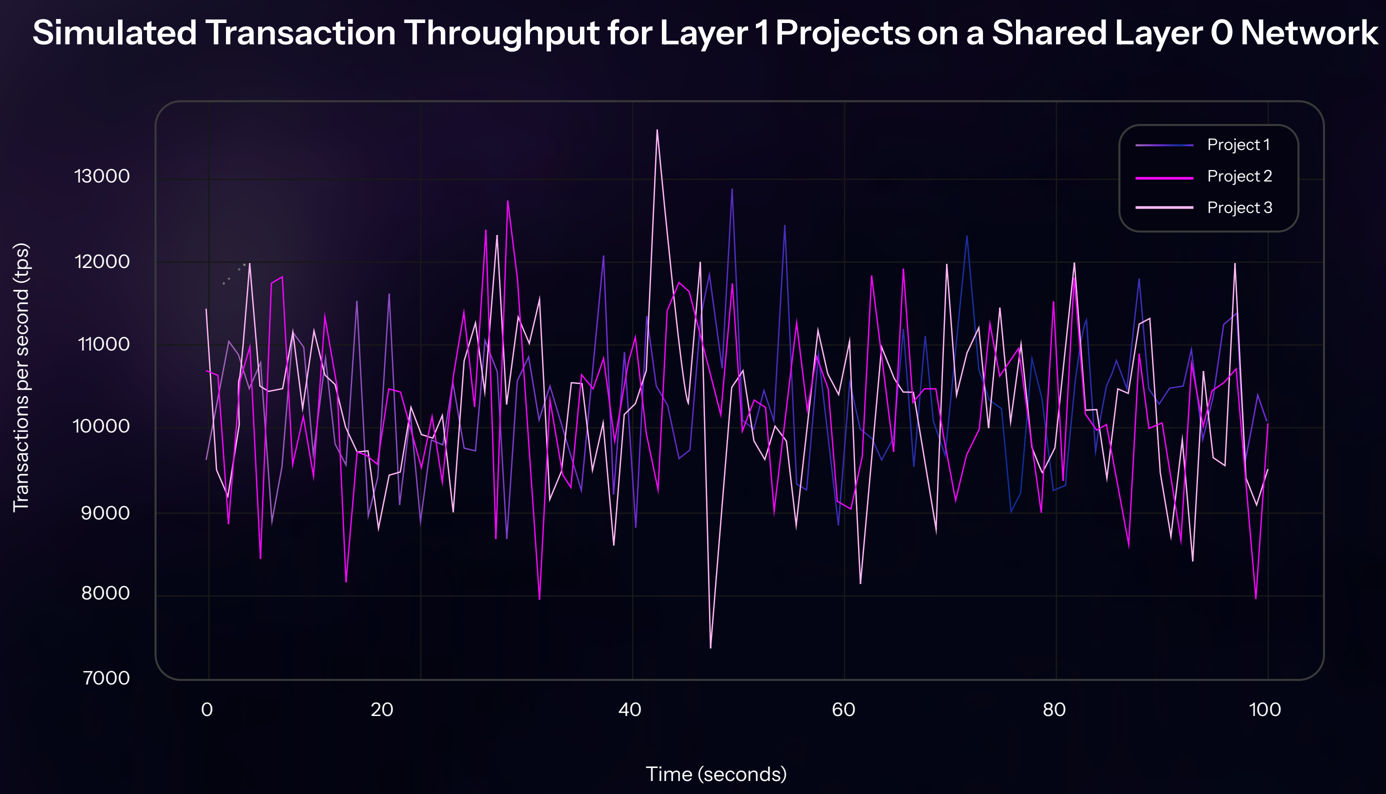


Fig. 5 Simulated Transaction Throughput for Layer 1 Projects on Layer 0 Network.

9.2. Gas Fee Simulation

Complementary to the throughput analysis, a simulation of gas fees was executed [19], capturing the computational and storage demands of various dApp types. This simulation aimed to offer insight into the costs associated with on-chain activity, from simple transactions to complex smart contract interactions.

9.2.1. Assumptions for Gas Fee Simulation

The following assumptions were integral to the gas fee simulation:

- **Computational Complexity:** Each dApp type exhibited a distinct pattern of transaction complexity, informed by common use-case scenarios.

- **Storage Requirements:** dApps with storage needs were attributed higher gas fees, proportional to the size of the data being managed.

- **Network Congestion:** A sinusoidal model was applied to simulate network congestion, affecting the gas fees across all dApp types.

9.2.2. Phron Zero Gas Fee Simulation Results

Since Phron Zero is the foundation on which multiple layer ones will run, we need understand the holistic impact of each layer one chain on layer zero. Let's first take a look on the assumed types of each layer one.

9.2.3. Simulated Gaming Focused Layer One

A gaming-focused blockchain is a specialized blockchain network designed specifically to cater to the needs and requirements of the gaming industry. Such a blockchain leverages the unique characteristics of blockchain technology to offer various features and functionalities tailored to gamers, game developers, and other stakeholders within the gaming ecosystem. The expected gas fees would be an order of magnitude higher for the gaming Dapps rather than the DEX or Storage.

Fig. 6 Simulated Gas Fee Consumption for Gaming Biased Layer 1.

9.2.4. Simulated DEX Focused Layer One

A decentralized exchange (DEX) focused blockchain is a specialized blockchain network specifically designed to facilitate decentralized trading of digital assets, such as cryptocurrencies, tokens, and other blockchain-based assets. This type of blockchain prioritizes features and functionalities that enhance the performance, security, and user experience of decentralized exchange platforms. Naturally, such a chain would generate more swap related gas fees.

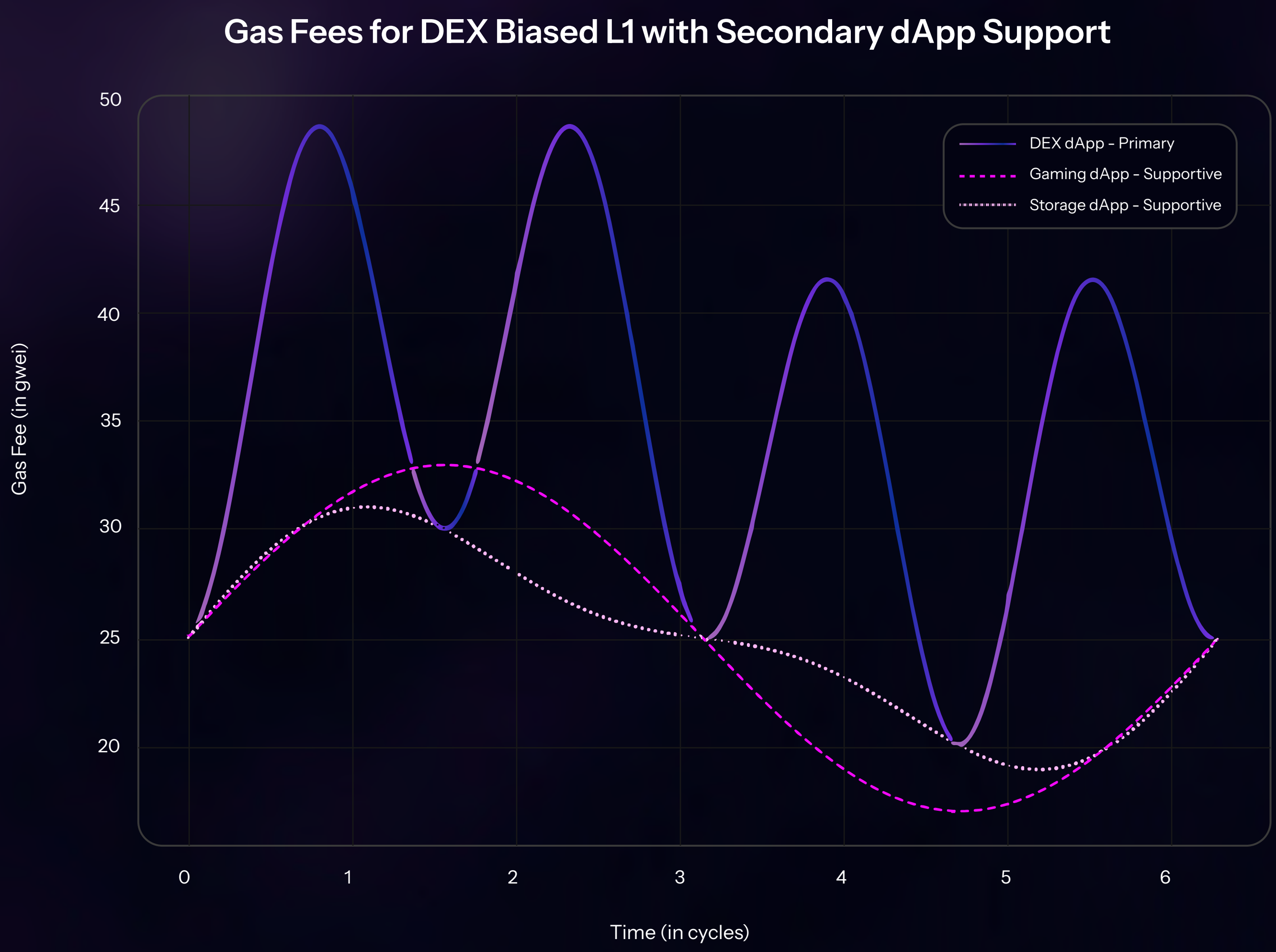


Fig. 7 Simulated Gas Fee Consumption for DEX Biased Layer 1.

9.2.5. Simulated Storage Focused Layer One

A storage-focused blockchain typically prioritizes the efficient and secure storage of data on the blockchain network. Gas fees, which represent the cost of performing transactions or executing smart contracts on the blockchain, play a crucial role in incentivizing network participants and maintaining the security and integrity of the system. In a storage-focused blockchain, gas fees may be structured in a way that reflects the costs associated with storing and accessing data on the blockchain. Gas fees in a storage-focused blockchain are designed to reflect the costs of storing and accessing data on the blockchain while incentivizing efficient resource usage and maintaining network security and performance. By implementing a dynamic and transparent fee structure, the blockchain ensures that gas fees remain competitive, responsive, and aligned with the needs of network participants.

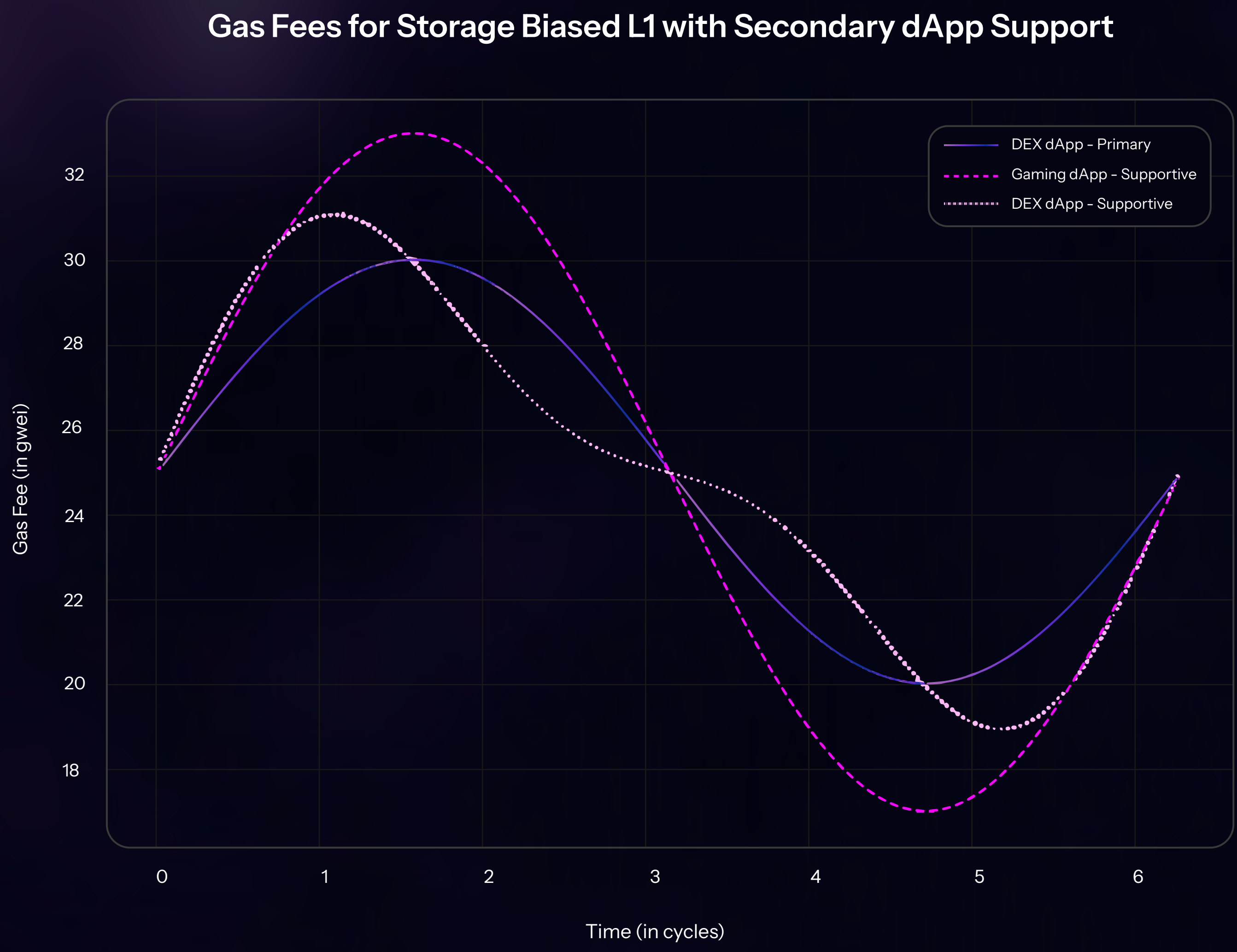


Fig. 8 Simulated Gas Fee Consumption for Storage Biased Layer 1.

9.3. Phron Zero Simulated Gas Fee Consumption

The results, as visualized in Figure 9, depicted the variability of gas fees over time for gaming, DEX, and storage dApps. The simulation reflected that storage-intensive dApps may incur higher fees during peak data operations, whereas gaming and DEX dApps showed variable fees correlated with their interactive and market-driven activities.

Fig. 9 Simulated Gas Fee Consumption for Different dApps.

9.4. Conclusion

The simulations confirm the Layer 0 network's capacity to support a multi-faceted blockchain ecosystem, managing both high-velocity transactions and complex dApp interactions efficiently. By mirroring realistic operational conditions, the simulations validate the network's design philosophy, highlighting its ability to adaptively balance performance and cost for diverse Layer 1 projects.

10. Economic Simulation

The methodology will center on the use of stochastic approximations to model and analyze the system. This approach allows us to estimate the collective behavior of agents within a system under conditions of uncertainty and variability. By leveraging stochastic approximations, we can efficiently simulate and predict outcomes without the need for detailed data on every individual component. This principle underpins our commitment to achieving both accuracy and computational efficiency in our simulations. Simulations will be used with a focus on understanding price dynamics, not with the aim of predicting the exact future price, but rather to comprehend the conditions and environment conducive to price appreciation or identifying factors leading to price declines.

While PhronAI is envisioned to evolve into a blockchain of blockchains, our current evaluation will concentrate exclusively on its initial layer-1. However, the potential of layer-1 should be assessed with the understanding that its scope extends beyond merely its launch. This principle underscores the importance of viewing PhronAI's layer-1 not just as an isolated product, but as a foundational element that contributes to the overall growth and success of the ecosystem.

10.1. Modeling Parameters

We have used the following parameters which we think are reasonable for this type of blockchain.
Run time 5 years.

- Initial price = $0.5
- Starting monthly transaction volume = $30 million
- Final monthly transaction volume = $300 million
- Pricing equation: Equation of exchange P=T/(MV) Allocation and emissions are highlighted above.
- Total tokens: 70% of the full supply (treasury and foundation are assumed to be out of circulation).
- Holding time: Lognormal Distribution.

The holding time data utilized in our analysis was compiled from a collection of holding times extracted from various industry projects. This dataset has been adopted as a robust basis for determining holding times, underpinned by the rationale that observed patterns across these projects offer a substantial foundation for formulating well-informed assumptions about future asset holding durations. By leveraging this historical data, we have established a benchmark for holding times, ensuring our projections are anchored in tangible, real-world observations and trends.

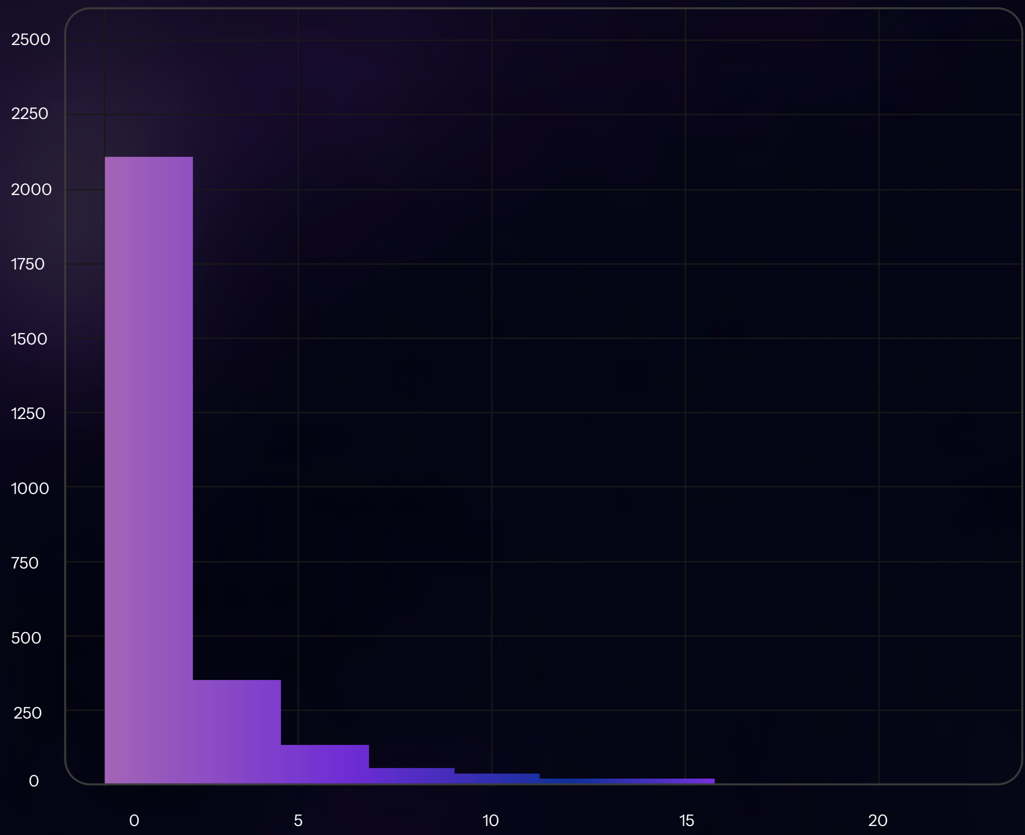


Fig. 10 Visual Holding Time Representation as Lognormal Distribution.

A simulation was conducted to test the resilience of the current assumptions. The simulation ran for 100 iterations. Each iteration consisted of the parameters displayed above. The results are shown below. The solid blue line shows the mean fair price, while the shaded areas show 95% confidence intervals.

10.2. Base Scenario Simulation

It looks like a launch price of $0.5 can lead to a fair value of close to $60 as a best-case scenario over 5 years.

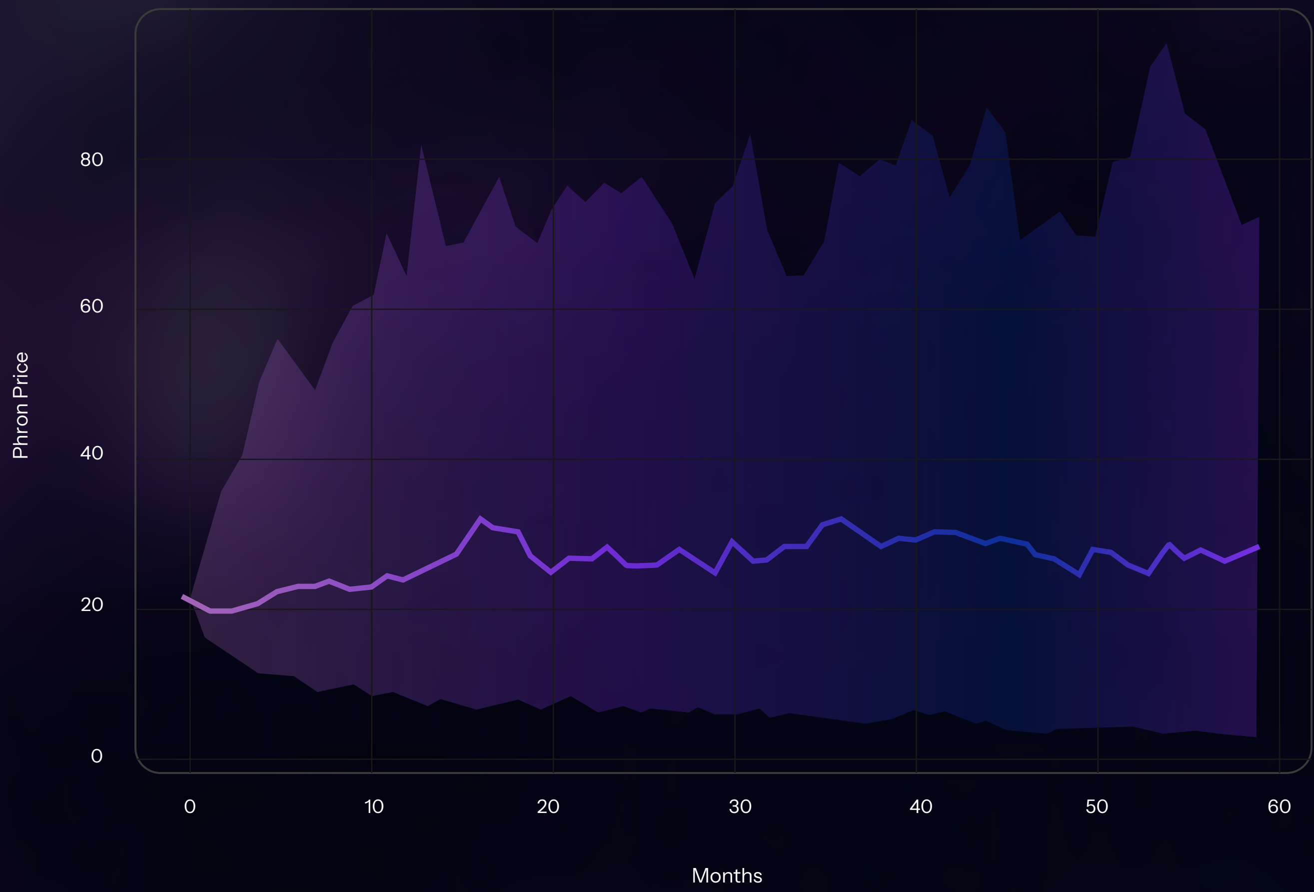


Fig. 11 Base Scenario Simulation.

10.3. Conclusion

Price appreciation is observed even with conservative metrics. This indicates that even modest achievements relative to transaction volume can lead to significant value increases, highlighting the potential for growth despite cautious projections.

Price appreciation, along with the current token supply allocation and emissions, demonstrates that the tokenomics from a quantitative perspective are defensible and can appreciate even without modeling forward multiples, which are often observed in euphoric market conditions to be 5-10x.

Price appreciation is assessed exclusively within the scope of Phron L1, excluding consideration of the interconnectedness with L0 and other L1 networks. This approach underrate the real project's value growth, which could be faster when the project's ultimate vision is considered.

10.4. Token Allocations

Token allocation refers to the distribution of the total supply of tokens in a blockchain project among various stakeholders, according to specific categories and purposes. This allocation is typically outlined in a project's whitepaper or offering documents and is an essential component of a project's economic and governance model. The tokenomics of Phron AI is structured with an initial circulating supply of 2,100,000,000 PHRON, with emissions for additional token creation to support future growth and scalability of the ecosystem.Effective token allocation is designed to align the incentives of developers, investors, users, and other stakeholders with the long-term success and sustainability of the project [20].

Private Sale (16%): Unlocks at Token Generation Event (TGE), followed by a 6- month linear vesting period. This is designed to protect the ecosystem from excessive token infusion during the chain boot-strap phase.

Public Sale (4%): Unlocks at TGE, followed by a 2-month linear vesting period.

Team (15%): Subject to a 9-month cliff, with linear vesting from that point until month 24. The longer vesting period is designed to insure that the team will continue chain support for the next two years at a minimum.

Ecosystem Supportive Nodes (15%): This amount is reserved for the ecosystem nodes Locked indefinitely to support the nodes.

Liquidity and market makers (17%): Fully unlocked.

Advisors (3%): Subject to a 9-month cliff, with linear vesting from that point until month 24.

Foundation (20%): Fully unlocked. The funds will be used for building the L0 and the grants program. The grants program will fund L1s with a strategic focus, facilitating the creation of the Phron ecosystem. The foundation will be utilized to support the construction of Layer 0 and the development of the system. This allocation strategy is designed to prevent the distribution of excessively large stakes to any particular group, thereby helping to manage early-stage volatility.

Treasury (10%): Vesting over 4 years, linearly at 25

Acknowledgements.
We would like to acknowledge the contributions of our community, especially to the completion of this work.

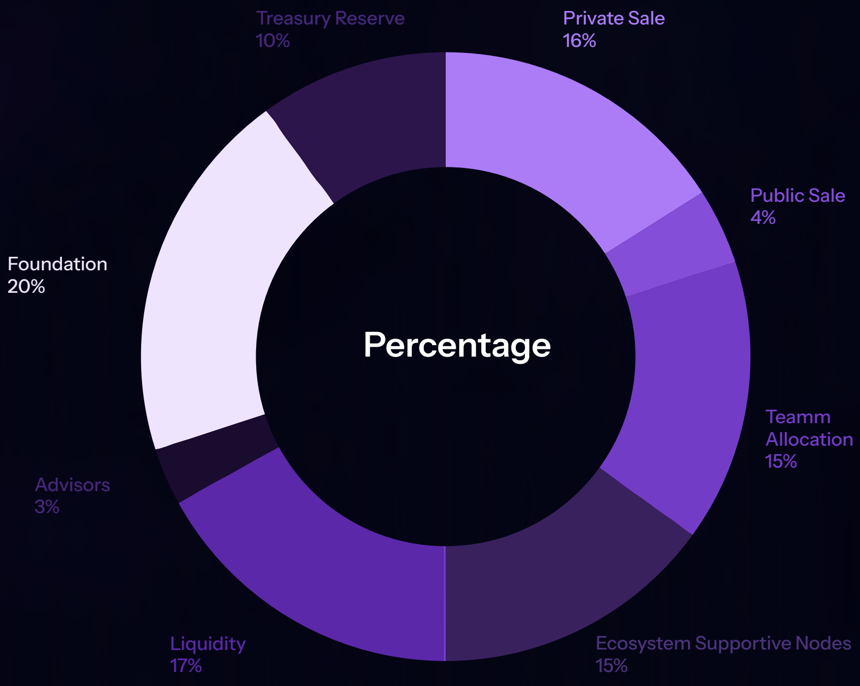


Fig. 12 Phron Token Allocation.

11. References

ElMessiry, M., ElMessiry, A., ElMessiry, M.: Dual token blockchain economy framework. In: International Conference on Blockchain, pp. 157–170 (2019). Springer

Gangwal, A., Gangavalli, H.R., Thirupathi, A.: A survey of layer-two blockchain protocols. Journal of Network and Computer Applications 209, 103539 (2023)

Mitchell, J.C.: Concepts in Programming Languages. Cambridge University Press, ??? (2003)

Pierce, B.C.: Types and Programming Languages. MIT press, ??? (2002)

Garay, J.A., Kiayias, A., Leonardos, N., Panagiotakos, G.: Bootstrapping the blockchain, with applications to consensus and fast pki setup. In: Public-Key Cryptography–PKC 2018: 21st IACR International Conference on Practice and Theory of Public-Key Cryptography, Rio de Janeiro, Brazil, March 25-29, 2018, Proceedings, Part II 21, pp. 465–495 (2018). Springer

Lantz, L., Cawrey, D.: Mastering Blockchain. O'Reilly Media, ??? (2020)

Sguanci, C., Spatafora, R., Vergani, A.M.: Layer 2 blockchain scaling: A survey. arXiv preprint arXiv:2107.10881 (2021)

Kiayias, A., Lazos, P.: Sok: blockchain governance. In: Proceedings of the 4th ACM Conference on Advances in Financial Technologies, pp. 61–73 (2022)

Tapscott, D., Tapscott, A.: Blockchain Revolution: How the Technology Behind Bitcoin Is Changing Money, Business, and the World. Penguin, ??? (2016)

Leonardos, S., Reijsbergen, D., Piliouras, G.: Weighted voting on the blockchain: Improving consensus in proof of stake protocols. International Journal of Network Management 30(5), 2093 (2020)

John, K., Rivera, T.J., Saleh, F.: Equilibrium staking levels in a proof-of-stake blockchain. Available at SSRN 3965599 (2021)

Choi, K.J., Jeon, J., Lim, B.H.: Optimal staking and liquid token holding decisions in cryptocurrency markets. Available at SSRN 4528742 (2023)

Kjorveziroski, V., Filiposka, S., Mishev, A.: Evaluating webassembly for orches- trated deployment of serverless functions. In: 2022 30th Telecommunications Forum (TELFOR), pp. 1–4 (2022). https://doi.org/10.1109/TELFOR56187. 2022.9983733

Tosh, D., Shetty, S., Foytik, P., Kamhoua, C., Njilla, L.: Cloudpos: A proof- of-stake consensus design for blockchain integrated cloud. In: 2018 IEEE 11Th International Conference on Cloud Computing (CLOUD), pp. 302–309 (2018). IEEE

Liu, Y., Lu, Y., Nayak, K., Zhang, F., Zhang, L., Zhao, Y.: Empirical analysis of eip-1559: Transaction fees, waiting times, and consensus security. In: Proceed- ings of the 2022 ACM SIGSAC Conference on Computer and Communications Security, pp. 2099–2113 (2022)

Cai, T., Cai, H., Wang, H., Cheng, X., Wang, L.: Analysis of blockchain system with token-based bookkeeping method. IEEE Access 7, 50823–50832 (2019)

Beck, R., Mu¨ller-Bloch, C., King, J.L.: Governance in the blockchain economy: A framework and research agenda. Journal of the association for information systems 19(10), 1 (2018)

Faria, C., Correia, M.: Blocksim: blockchain simulator. In: 2019 IEEE Interna- tional Conference on Blockchain (Blockchain), pp. 439–446 (2019). IEEE

Memon, R.A., Li, J.P., Ahmed, J.: Simulation model for blockchain systems using queuing theory. Electronics 8(2), 234 (2019)

Lee, J.Y.: A decentralized token economy: How blockchain and cryptocurrency can revolutionize business. Business Horizons 62(6), 773–784 (2019)

Appendix A Simulation Code Example

The following is the code written in Python to generate the simulations used in the document above.

Simulated Transactions Per Second (TPS) Over 24 Hours

```
import numpy as np
import matplotlib.pyplot as plt

base_tps_phronzero = 100000
base_tps_phronlayer1 = 50000

time_hours = np.arange(0, 24, 1)
network_load_factor_phronzero = 0.5 * np.sin(np.pi * time_hours / 12 - np.pi/2) + 1
network_load_factor_phronlayer1 = 0.5 * np.sin(np.pi * time_hours / 12 - np.pi/2) + 1.5

effective_tps_phronzero = base_tps_phronzero * network_load_factor_phronzero
effective_tps_phronlayer1 = base_tps_phronlayer1 * network_load_factor_phronlayer1

plt.figure(figsize=(14, 7))

plt.plot(time_hours, effective_tps_phronzero, label='PhronZero TPS', marker='o')
plt.plot(time_hours, effective_tps_phronlayer1, label='Phron Layer 1 TPS', marker='x')

plt.title('Simulated Transactions Per Second (TPS) Over 24 Hours')
plt.xlabel('Time (Hours)')
plt.ylabel('Transactions Per Second (TPS)')
plt.legend()
plt.grid(True)
plt.xticks(time_hours)
plt.ylim(0, max(effective_tps_phronzero) + 50000)

plt.show()
\end{verbatim*}
\subsection{Simulated Dynamic Gas Fee}
\begin{verbatim*}
    import numpy as np
import matplotlib.pyplot as plt

def calculate_dynamic_gas_fee(base_fee, tip, epsilon, p_target, p_current, delta_c, delta_n):
    """
    Calculate the dynamic gas fee for a transaction based on the provided parameters.

    :param base_fee: Base fee of the transaction
    :param tip: Optional tip to miners/validators
    :param epsilon: Sensitivity parameter for token price stabilization
    :param p_target: Target token price
    :param p_current: Current token price
    :param delta_c: Rate of change in transaction complexity
    :param delta_n: Rate of change in network congestion
    :return: Calculated dynamic gas fee
    """
    price_adjustment = epsilon * (p_target - p_current) / p_current
    gas_fee = (base_fee + tip + price_adjustment) * (delta_c + delta_n)
    return gas_fee

base_fee = 10
tip = 1
epsilon = 0.1
p_target = 1
p_current = 0.65
delta_c = 1
delta_n = 1
```

Listing 1: Python example

2. Gas Fees for Storage Biased L1 with Secondary dApp Support

```
# Simulate dynamic gas fees over time for a single dApp type
time = np.arange(0, 10, 0.1)
gas_fees = [calculate_dynamic_gas_fee(base_fee, tip, epsilon, p_target, p_current +
      t, delta_c, delta_n) for t in time]

plt.figure(figsize=(10, 6))
plt.plot(time, gas_fees, label='Dynamic Gas Fee Over Time')
plt.title('Simulated Dynamic Gas Fee')
plt.xlabel('Time')
plt.ylabel('Gas Fee')
plt.legend()
plt.grid(True)
plt.show()
```

Listing2:Pythonexample

3. Simulated Gas Fees

```

import numpy as np
import matplotlib.pyplot as plt

# Setup
t = np.linspace(0, 2 * np.pi, 400)
BaseRate = 10
StorageRate = 0.05
N = 0.5 * np.sin(t) + 1.5  # Network congestion

# Parameters for complexity (C)
A, B, V, M, S = 1.5, 1, 2, 1.5, 0.8
omega, eta, kappa, alpha, R, T = 2, 0.05, 1, 0.1, 5, 50
i = np.arange(1, int(np.max(t) / T) + 1) * T

# Gaming dApp
C_gaming = A * np.sin(omega * t) + B
D_gaming = np.zeros_like(t)
G_gaming = BaseRate * (1 + C_gaming + N) + StorageRate * D_gaming

# DEX
C_dex = V * np.log(1 + eta * t) + M * np.sin(kappa * t)
D_dex = np.zeros_like(t)
G_dex = BaseRate * (1 + C_dex + N) + StorageRate * D_dex

# Storage dApp
C_storage = np.full_like(t, S)
D_storage = R * np.sum([np.exp(-alpha * (t - iT)**2) for iT in i], axis=0)
G_storage = BaseRate * (1 + C_storage + N) + StorageRate * D_storage

# Plotting
plt.figure(figsize=(12, 8))
plt.plot(t, G_gaming, label='Gaming dApp')
plt.plot(t, G_dex, label='DEX')
plt.plot(t, G_storage, label='Storage dApp', linestyle='--')

plt.title('Simulated Gas Fees')
plt.xlabel('Time (in cycles)')
plt.ylabel('Gas Fee (in gui unit)')
plt.legend()
plt.grid(True)
plt.show()
```

Listing3:Pythonexample

4. Gas Fees for Storage Biased L1 with Secondary dApp Support

```
import numpy as np
import matplotlib.pyplot as plt

# Setup
t = np.linspace(0, 2 * np.pi, 400)
BaseRate = 10
N = 0.5 * np.sin(t) + 1.5  # Simulated network congestion

C_gaming_primary = 1.5 * np.sin(2 * np.pi * t / max(t))
C_dex_secondary = 0.3 * np.sin(4 * np.pi * t / max(t))  # Reduced influence
C_storage_secondary = 0.2 * np.sin(4 * np.pi * t / max(t))  # Reduced influence

G_gaming = BaseRate * (1 + C_gaming_primary + N)
G_dex = BaseRate * (1 + C_dex_secondary + N)
G_storage = BaseRate * (1 + C_storage_secondary + N)

plt.figure(figsize=(12, 8))
plt.plot(t, G_gaming, label='Gaming dApp - Primary', color='blue')
plt.plot(t, G_dex, label='DEX dApp - Secondary', color='orange', linestyle='--')
plt.plot(t, G_storage, label='Storage dApp - Secondary', color='green', linestyle='
    :')

plt.title('Gas Fees for Gaming Biased L1 with Secondary dApp Support')
plt.xlabel('Time (in cycles)')
plt.ylabel('Gas Fee (in gwei)')
plt.legend()
plt.grid(True)
plt.show()

plt.figure(figsize=(12, 8))

G_dex_primary = BaseRate * (1 + 2.0 * (np.sin(4 * np.pi * t / max(t))**2) + N)
G_gaming_supportive = BaseRate * (1 + 0.3 * np.sin(2 * np.pi * t / max(t)) + N)  #
    Increased presence from Gaming
G_storage_supportive = BaseRate * (1 + 0.2 * np.sin(4 * np.pi * t / max(t)) + N)

plt.plot(t, G_dex_primary, label='DEX dApp - Primary', color='orange')
plt.plot(t, G_gaming_supportive, label='Gaming dApp - Supportive', color='blue',
    linestyle='--')
plt.plot(t, G_storage_supportive, label='Storage dApp - Supportive', color='green',
     linestyle=':')

plt.title('Gas Fees for DEX Biased L1 with Secondary dApp Support')
plt.xlabel('Time (in cycles)')
plt.ylabel('Gas Fee (in gwei)')
plt.legend()
plt.grid(True)
plt.show()


plt.figure(figsize=(12, 8))
G_storage_primary = BaseRate * (1 + 0.8 * np.sum([np.exp(-0.1 * (t - iT)**2) for iT
     in np.arange(1, int(np.max(t) / 50) + 1) * 50], axis=0) + N)
G_gaming_supportive = BaseRate * (1 + 0.3 * np.sin(2 * np.pi * t / max(t)) + N)  #
    Slightly increased presence from Gaming
G_dex_supportive = BaseRate * (1 + 0.2 * np.sin(4 * np.pi * t / max(t)) + N)

plt.plot(t, G_storage_primary, label='Storage dApp - Primary', color='green')
plt.plot(t, G_gaming_supportive, label='Gaming dApp - Supportive', color='blue',
    linestyle='--')
plt.plot(t, G_dex_supportive, label='DEX dApp - Supportive', color='orange',
    linestyle=':')

plt.title('Gas Fees for Storage Biased L1 with Secondary dApp Support')
plt.xlabel('Time (in cycles)')
plt.ylabel('Gas Fee (in gwei)')
plt.legend()
plt.grid(True)
plt.show()
```

Listing4:Pythonexample

5. Transaction Throughput for Layer 1 Projects and Total Usage on Layer 0 Network

```python

import numpy as np
import matplotlib.pyplot as plt

# Simulation parameters
SIMULATION_DURATION = 100  # Total time for simulation in seconds
LAYER_0_TPS = 31000  # Layer 0's maximum tps
NUM_PROJECTS = 3  # Number of Layer 1 projects

# Assume an even distribution of Layer 0's TPS across Layer 1 projects
tps_distribution = LAYER_0_TPS / NUM_PROJECTS

# Simulating transaction throughput for each Layer 1 project over time
time_steps = np.linspace(0, SIMULATION_DURATION, SIMULATION_DURATION)
throughput_data = {}

# Total throughput at each time step
total_throughput = np.zeros(SIMULATION_DURATION)

# Create throughput data for each project and calculate total throughput
for i in range(NUM_PROJECTS):
    # Randomly vary the tps for each project to simulate fluctuating network
        conditions
    tps_variation = np.random.normal(0, 1000, SIMULATION_DURATION)
    throughput_data[f'Project {i+1}'] = tps_distribution + tps_variation
    total_throughput += throughput_data[f'Project {i+1}']

# Plotting the multi-line chart for individual projects
plt.figure(figsize=(14, 7))
for project, tps in throughput_data.items():
    plt.plot(time_steps, tps, label=project)

# Adding the total usage curve
plt.plot(time_steps, total_throughput, label='Total Usage', color='black',
    linewidth=2, linestyle='--')

# Chart configurations
plt.title('Transaction Throughput for Layer 1 Projects and Total Usage on Layer 0
    Network')
plt.xlabel('Time (seconds)')
plt.ylabel('Transactions per Second (tps)')
plt.legend()
plt.grid(True)
plt.show()
```

Listing5:Pythonexample

6. The Annual Percentage Rate (APR)

```python
import numpy as np
import matplotlib.pyplot as plt

def apr_function(x):
    base = 15 * 10**15
    apr_decimal = -np.log10(x / 2000 + 0.0001) / np.log10(base) - 0.1
    return apr_decimal * 100

x_values = np.linspace(0, 20, 400)  # Assuming the time range is 0 to 20 years for illustration
y_values = apr_function(x_values)

# Plotting the function
plt.figure(figsize=(10, 5))
plt.plot(x_values, y_values, label='APR over time')
plt.title('APR as a function of Time (Percentage)')
plt.xlabel('Time (years)')
plt.ylabel('APR (%)')
plt.legend()
plt.grid(True)
plt.show()
```

Listing6:Pythonexample