

RATEM: A Resource-Aware Transaction Evaluation Framework for Scalable and Fair Blockchain Systems

Oleksandr Khoshaba *¹

¹Department of Software, Vinnitsia National Technical University, Ukraine

June 2025

Abstract

Accurate and fair evaluation of blockchain transactions is essential for ensuring system efficiency, economic fairness, and resilience to manipulation. Traditional evaluation mechanisms, such as Ethereum’s gas-based model, use static opcode pricing and market-driven fees, which do not reflect a transaction’s actual computational footprint. This leads to inefficiencies, unfair prioritisation, and limited scalability in heterogeneous or high-load environments.

This paper introduces the Resource-Aware Transaction Evaluation Model (RATEM), a multidimensional framework designed to quantify transaction cost based on observable system-level metrics, including CPU time, memory operations, I/O volume, and execution latency. The RATEM approach addresses the gap between abstract fee mechanisms and real resource consumption by offering three adaptable submodels: ETOM (Execution-Time-Only Model), IREM (Instrumented Resource Evaluation Model), and HGEM (Hybrid Gas Evaluation Model). Each submodel supports a different level of metric availability and validator capability, enabling deployment in both constrained and advanced settings.

The paper presents a formal mathematical formulation of each model, discusses their comparative properties, and provides a taxonomy of resource metrics used in transaction evaluation. Experimental insights and theoretical analysis demonstrate that RATEM improves alignment between transaction cost and system load, supports fairer scheduling decisions, and enhances platform-agnostic comparability.

The proposed framework offers a practical foundation for improving blockchain governance and transaction fee mechanisms in Layer-1 and Layer-2 environments. Future directions include machine learning-based weight calibration, testnet deployment, and integration with smart contract validators. RATEM provides a viable pathway toward transparent, efficient, and resource-aware transaction evaluation in modern blockchain ecosystems.

Keywords: Blockchain, Transaction Evaluation, Resource Metrics, Ethereum, RATEM

Introduction

Blockchain networks have emerged as a transformative infrastructure for secure, decentralised, and verifiable data processing across numerous domains, including finance, logistics, energy, and governance. At the core of every blockchain system lies the transaction: a fundamental unit of operation that triggers state transitions, executes smart contracts, and facilitates trustless value exchange. The structure, validation, and prioritisation of transactions define not only the technical behaviour of the blockchain but also its economic sustainability, scalability, and fairness.

*Corresponding author: Oleksandr.Khoshaba@gmail.com

Due to the distributed and decentralised nature of consensus mechanisms, transaction processing must be rigorously analysed and optimised. Transaction performance metrics – throughput, latency, gas cost, and confirmation time – are critical to public and private blockchains’ operational and economic viability. Yet, as highlighted in recent surveys [7, 9, 11, 8], a comprehensive and adaptable model for evaluating transaction quality remains an open research challenge.

Ethereum, for instance, relies on a gas pricing mechanism to manage computation and storage usage within the Ethereum Virtual Machine (EVM) [10]. While this model has successfully prevented denial-of-service attacks and incentivised miners, it does not capture holistic transaction quality metrics such as semantic utility, system-level load, or long-term prioritisation trends. Moreover, introducing Ethereum Improvement Proposals (EIPs), such as EIP-1559, has introduced dynamic fee structures that aim to stabilise user experience and validator incentives [2].

Recent work by Li et al. [7] categorises blockchain performance evaluation models into architectural, operational, and transactional dimensions, yet notes the lack of standard benchmarks for transaction rating. Wang et al. [9] emphasise the limitations of gas-centric evaluation in capturing performance trade-offs in resource-constrained environments. Meanwhile, studies such as Zhang et al. [11], Gervais et al. [6], and Chen et al. [3] show that performance and security cannot be decoupled from the evaluation framework of transactional activity.

Additionally, recent audits and debugging tools such as GasChecker [3] reveal frequent gas underestimations in smart contracts, causing unexpected failures or vulnerabilities. This further motivates the need to base transaction evaluation on observable system metrics rather than static opcode costs.

As blockchain ecosystems diversify and adopt heterogeneous configurations – such as sharded architectures, Layer-2 solutions, or DAG-based models – the need for flexible and accurate transaction evaluation frameworks grows. The problem is further compounded by the emergence of application-specific blockchains (AppChains) and cross-chain interoperability protocols, which call for generalised metrics that can facilitate informed scheduling, pricing, and policy-making decisions [1].

This paper addresses the core mismatch between gas-based pricing and actual system-level impact. A novel, modular approach is required to integrate multiple dimensions of transaction quality, support cross-platform comparability, and enhance transparency in blockchain operations.

To address these challenges, this work introduces the *Resource-Aware Transaction Evaluation Model (RATEM)* – a multidimensional framework that evaluates transaction cost based on real execution characteristics such as CPU time, memory operations, I/O volume, and latency. RATEM offers an extensible foundation for fair and performance-aligned blockchain transaction governance.

1 Problem Formulation and Motivation

1.1 Background

Blockchain networks rely on transaction evaluation mechanisms to ensure fairness, efficiency, and security in resource-constrained environments. In Ethereum and similar systems, the gas-based model has become the de facto method for quantifying transaction cost. This model statically assigns computational weights to low-level operations and uses market-based bidding (e.g., `gasPrice`) to prioritise transactions. While effective in deterring denial-of-service attacks and supporting incentive structures, this model fails to reflect the actual computational footprint of transactions.

1.2 Identified Gaps

Despite its wide adoption, the gas model suffers from key limitations:

- **Lack of resource correspondence:** Gas usage does not account for actual CPU, memory, or I/O load.
- **Inflexibility under load:** Static cost assignment does not adapt to system congestion or validator capacity.
- **Unfair prioritisation:** Fee-based ordering enables economic abuse and undermines equitable access.
- **Limited cross-platform comparability:** Metrics are not transferable across Layer-2 or sharded environments.
- **Inaccurate simulation assumptions:** Dry-run estimates often diverge from real execution profiles.

1.3 Research Motivation

The evolution of blockchain infrastructure – including roll-up architectures, AppChains, and heterogeneous validator configurations – exposes the inadequacy of static, one-dimensional evaluation models. As performance expectations increase, transaction evaluation must evolve to be dynamic, fair, and context-aware.

1.4 Research Objective

This study aims to develop a *Resource-Aware Transaction Evaluation Model (RATEM)* that quantifies transaction cost based on real system metrics such as CPU time, memory access, I/O operations, and latency. The model should be modular, adaptive, and suitable for deployment in heterogeneous environments. It also seeks to support more efficient transaction ordering, cross-platform fairness, and system resilience.

1.5 Structure of the Paper

The structure of this paper is as follows:

- **Section 1** introduces the research problem, outlines background and motivation, identifies existing limitations in transaction evaluation models, and defines the objectives of the proposed solution.
- **Section 2** outlines key challenges associated with transaction evaluation in blockchain networks, particularly under high and low load conditions.
- **Section 3** analyses the fundamental limitations of the traditional gas-based evaluation model used in Ethereum and similar systems.
- **Section 4** surveys existing approaches in academic and industrial contexts, including enhanced gas models, off-chain execution strategies, and quality-of-service frameworks.
- **Section 5** introduces the proposed Resource-Aware Transaction Evaluation Model (RATEM), describing its rationale, core metrics, and objectives.

- **Section 6** presents a taxonomy and classification of RATEM-based models, including ETOM, IREM, and HGEM, along with their mathematical formulations and implementation considerations.
- **Section 7** discusses theoretical implications of the RATEM framework, such as multidimensional evaluation, assumptions, and comparative advantages.
- **Section 8** concludes the paper with a summary of contributions, potential benefits, and future research and deployment directions.

2 Challenges in Transaction Evaluation in Blockchain Networks

Blockchain transaction evaluation is critical in determining decentralised networks’ efficiency, fairness, and security. However, widely adopted approaches such as gas-based models – particularly those used in Ethereum – face multiple limitations under different network load scenarios. This section categorises these challenges into two complementary conditions: highly competitive and underutilised network states.

2.1 Competitive Load Environments

In high-demand scenarios, users must compete for transaction inclusion by increasing the gas prices they offer. This auction-based mechanism, which underlies Ethereum’s original gas model and is partially modified by EIP-1559 [2], leads to inflation in transaction fees and creates a barrier for low-value or micro-scale applications. Although EIP-1559 introduces a dynamic base fee and burns a portion to stabilise gas pricing, it does not eliminate fee volatility.

Moreover, increased gas prices directly reduce the affordability of blockchain usage, especially for users deploying frequent or high-volume transactions. This restricts the network’s inclusivity and amplifies cost asymmetries between different classes of participants.

2.2 Low Load Inefficiencies

Conversely, during periods of low network utilisation, the base fee set by EIP-1559 drops significantly, allowing users to submit transactions at minimal costs. While economically favourable for users, this introduces unintended consequences: transactions with high computational cost may appear cheap regarding gas fees, yet impose significant burdens on validator nodes.

Such a misalignment between economic cost and system resource usage leads to inefficiencies in block processing. Validators spend more time and resources verifying low-paying, resource-intensive transactions, reducing overall system throughput. Furthermore, this condition opens opportunities for denial-of-service (DoS) attacks via transaction flooding with low gas prices [6, 5].

2.3 Summary of Limitations

The prevailing gas-based transaction evaluation models lack multidimensional awareness. They focus primarily on fee bidding and computational steps, ignoring context-aware factors such as network state, transaction semantics, and real-time system load [9, 11]. This leads to mispriced resources, inefficient block utilisation, and vulnerability to economic exploits.

A more comprehensive transaction evaluation framework must balance economic cost, computational intensity, and systemic impact in high- and low-load scenarios.

3 Limitations of the Traditional Gas-Based Evaluation Model

The gas-based model widely used in Ethereum and similar blockchains is the de facto mechanism for evaluating and pricing transactions. It combines two elements: an objectively measured quantity known as `gasUsed`, and a market-driven price component, `gasPrice` or `maxFeePerGas`. Although this framework has enabled scalable and attack-resistant smart contract execution, it presents multiple structural limitations that hinder accurate transaction assessment.

3.1 Subjectivity and Market Dependency

In Ethereum, the cost of a transaction is determined through a market-based mechanism where users propose their own `gasPrice`. This price fluctuates based on network demand, creating an inherently subjective and volatile evaluation of transaction cost. Even with the introduction of EIP-1559, which separates base fees from tip incentives, user-defined parameters still dominate fee dynamics [2].

Such market dependency results in unpredictable transaction fees and raises barriers for consistent quality-of-service (QoS) guarantees. It also introduces strategic fee bidding behaviour that may distort fair access to the network.

3.2 Abstract Resource Metrics

While `gasUsed` is calculated deterministically by the Ethereum Virtual Machine (EVM), it is based on a static enumeration of opcodes, memory calls, and storage interactions. It does not capture actual computational or system resource consumption. Notably, it lacks correspondence with physical execution metrics such as CPU cycles, I/O bandwidth, memory usage, energy consumption, or latency.

This abstraction decouples transaction evaluation from real-world node performance and obfuscates the actual computational impact of smart contract operations [10, 9].

3.3 Implications for System-Level Optimisation

Due to the absence of resource-sensitive evaluation, system-level optimisation becomes infeasible. Validators and system designers cannot accurately prioritise transactions based on real load impact or resource balancing requirements. This creates inefficiencies in block packing, increases the likelihood of congestion, and impedes the development of adaptive consensus strategies.

To support next-generation blockchain applications and heterogeneous execution environments, more comprehensive evaluation frameworks must go beyond gas accounting and integrate direct resource-awareness.

3.4 Fundamentals of Gas-Based Evaluation

Gas is a virtual unit that reflects the static cost of executing low-level EVM operations. Each opcode has a predefined price specified in the Ethereum Yellow Paper [10]. However, this enumeration does not account for dynamic computational conditions like memory locality, branching, or concurrency.

3.5 Transaction Lifecycle and Evaluation Timing

A transaction undergoes six main stages before inclusion in the blockchain:

1. **Client Submission:** Transaction creation, signing, and broadcast ($\sim 10\text{--}150$ ms).

2. **Mempool Admission:** Initial validation and queuing (validation: $\sim 10\text{--}100$ ms; waiting: 10 ms to several minutes) [5].
3. **Transaction Selection:** Prioritisation by validators ($\sim 1\text{--}50$ ms).
4. **Block Construction:** Assembly and gas limit packing ($\sim 100\text{--}300$ ms).
5. **Consensus and Propagation:** Block broadcast and consensus (~ 12 s in Ethereum PoS) [2].
6. **Finalisation:** Final state update and confirmation (24 s to 6 minutes).

Notably, gas estimation occurs early, during client-side simulation (dry-run), and again in validator-side validation. However, it does not reflect the variability of absolute execution paths.

3.6 Heuristic vs. Actual Resource Consumption

Gas estimation mechanisms [9] rely on simplified heuristics based on instruction cost tables. In reality, transaction behaviour may diverge from dry-run predictions due to:

- data-dependent execution paths;
- memory and storage access patterns;
- changes in global state between simulation and inclusion.

Ethereum documentation acknowledges that gas estimates are best-effort and can under- or overshoot the true cost [4].

3.7 Network Incentives and Inclusion Priorities

Validators do not prioritise "lightweight" transactions per se, but rather those offering the highest fee per gas unit. Portfolio-based transaction selection aims to maximise profit under the block gas limit [6]. This creates a skewed incentive model, favouring expensive and potentially wasteful transactions over efficient ones.

3.8 On "Light" and "Heavy" Transactions

Colloquially, "light" transactions refer to those with low gas usage (e.g., ETH transfers), while "heavy" transactions involve complex smart contracts with high storage or compute demand. Officially, this distinction maps to "compute-intensive", "storage-heavy", or "I/O-bound" classifications [11]. The gas model, however, fails to differentiate them beyond static instruction cost.

3.9 Root Cause of Unfair Evaluation

The critical misalignment arises during *Stage 2 – Mempool Admission*. Here:

- Real resources are consumed during validation (CPU, memory);
- Little to no resources are consumed during mempool waiting;
- Yet, the gas cost is estimated and fixed based on simplistic assumptions.

This decoupling between gas and impact is precisely what RATEM seeks to rectify by incorporating runtime profiling during the validation phase.

3.10 Summary and Implications

This analysis reveals a persistent inefficiency and unfairness in current gas evaluation models:

- Gas does not reliably reflect system-level resource usage;
- Estimation is context-insensitive and potentially misleading;
- Validators optimise for fees, not efficiency;
- The need for resource-aware metrics like RATEM is theoretical and practically urgent.

These findings support the integration of RATEM as a complementary evaluation mechanism aligned with actual resource impact.

4 Overview of Existing Solution Approaches

Various approaches have been proposed in academic literature and industry initiatives to address the limitations of traditional gas-based models. These approaches aim to improve transaction evaluation by incorporating alternative metrics, computational insights, or network-aware mechanisms. This section briefly categorises these solution paths rather than an exhaustive review.

4.1 Enhanced Gas Mechanisms

Some solutions retain the gas-based framework but introduce modifications to improve responsiveness and fairness. Notably, Ethereum’s EIP-1559 proposes a dynamic base fee adjustment to reduce fee volatility [2], while proposals such as EIP-4844 explore blob-carrying transactions for scalable data availability. These approaches maintain backwards compatibility while addressing specific inefficiencies.

4.2 Off-Chain Evaluation Strategies

Layer-2 protocols such as Optimistic Rollups, zkRollups, and Plasma introduce off-chain transaction execution with periodic on-chain commitments. These models decouple transaction execution from gas pricing, enabling context-specific or broker-mediated evaluation mechanisms that optimise cost and throughput.

4.3 Resource-Aware Metrics

A growing body of research advocates using metrics grounded in actual system consumption – such as CPU cycles, FLOPS, memory utilisation, or energy dissipation – to evaluate transaction cost [9]. This approach would align pricing with physical resource use, providing a more accurate load balancing and fairness basis.

4.4 Machine Learning-Based Estimation Models

Recent efforts explore integrating predictive models and machine learning to assess transaction importance or expected load. These systems may adjust evaluation parameters dynamically based on observed behaviours, enabling intelligent scheduling and fee adaptation.

4.5 Quality-of-Service (QoS) Models

Some blockchain designs incorporate QoS principles by introducing transaction classes, priority queues, or service-level agreements (SLAs). These models aim to offer differentiated transaction handling based on application needs or user profiles, often borrowing concepts from network traffic engineering.

4.6 Synthesis and Gaps

While each approach contributes to improving specific aspects of transaction evaluation, they remain largely disjointed. A unified framework that integrates system-level awareness, user-driven policies, and cross-platform compatibility is still lacking. This highlights the need for a holistic model that adapts to blockchain networks’ diverse and evolving conditions.

4.7 Systemic Gaps in Gas-Based Transaction Evaluation

Despite widespread adoption, the gas-based evaluation model used in Ethereum and similar blockchains presents significant misalignments between gas cost and actual resource consumption. This article identifies key systemic gaps and substantiates the need for a more resource-aware evaluation framework.

5 Introduction to the Resource-Aware Transaction Evaluation Models (RATEM)

Contemporary blockchain platforms rely heavily on gas-based models to evaluate and prioritise transactions. While effective for preventing denial-of-service attacks and incentivising validators, these models fail to capture each transaction’s actual computational and systemic impact. This inadequacy leads to misaligned pricing, inefficient resource utilisation, and a lack of transparency in transaction evaluation.

5.1 Rationale

Traditional evaluation metrics such as `gasUsed` abstract away the physical resource footprint of a transaction. They do not account for real-world execution characteristics such as CPU time, memory access patterns, I/O load, or energy consumption. As blockchain applications diversify and hardware configurations become more heterogeneous, this gap becomes increasingly problematic.

5.2 Model Concept

The proposed **Resource-Aware Transaction Evaluation Models (RATEM)** is designed to bridge this gap. RATEM introduces a multi-dimensional evaluation framework that correlates transaction cost with actual system resource usage under diverse execution contexts. Rather than relying on a monolithic gas pricing scheme, RATEM decomposes the transactional footprint into measurable, system-relevant components — such as CPU time, memory access, and I/O load — enabling more granular and equitable evaluation.

5.3 Target Metrics

RATEM captures a broad spectrum of metrics that reflect the real computational load and execution profile of a transaction. Some of these metrics include:

- **CPU execution time** (e.g., milliseconds or CPU cycles)
- **Memory operations** (e.g., bytes read/written per second)
- **I/O volume** (e.g., system-level input/output interactions)
- **Runtime latency** (e.g., total execution delay of a smart contract)

Depending on the platform’s capabilities, these metrics are either measured directly via execution tracing tools or inferred through profiling models.

5.4 Objectives

The overarching objective of RATEM is to enhance transaction evaluation by providing a resource-transparent, platform-adaptable, and fair assessment mechanism. Specific goals include:

- Improving alignment between transaction cost and system load
- Enabling cross-platform comparability and standardisation
- Supporting dynamic scheduling and adaptive consensus mechanisms
- Enriching incentive design and resource pricing policies

RATEM is a foundational step toward a more accountable and efficient transaction evaluation ecosystem in modern blockchain networks.

6 Taxonomy and Classification of RATEM-Based Models

The design of RATEM is not merely a technical enhancement but a conceptual response to the fundamental limitations of gas-based transaction evaluation in blockchain systems. Its modular framework supports several evaluation strategies based on system capabilities (Table 1).

This section outlines the theoretical rationale behind RATEM and the specific objectives it seeks to fulfil, as well as the taxonomy and classification of RATEM-based models.

6.1 RATEM as a Dynamic Adjustment to Stateless Gas Models

Traditional blockchain gas models rely on static opcode pricing tables, which assign a fixed gas cost to each instruction regardless of context. However, the real cost of executing identical instructions may vary significantly depending on system load, memory state, and concurrent network activity.

RATEM introduces a dynamic correction mechanism that adjusts transaction evaluation according to real-time execution context. This allows identical instruction sets to be priced differently if executed under peak load, thus aligning cost with actual resource strain. The model moves from a purely syntactic evaluation (instruction-counting) to a context-aware, semantic assessment of transaction impact.

6.2 Extended Objectives of RATEM

RATEM pursues a set of broader design objectives that address systemic inefficiencies in blockchain transaction processing:

- **Correcting gas model limitations:** Enabling dynamic cost adaptation based on system load, ensuring that instructions executed during peak usage periods are priced proportionally to their execution strain.
- **Ensuring fair resource attribution:** Matching transaction fees to actual consumption of CPU time, memory bandwidth, and I/O operations, rather than relying on abstract gas units.
- **Providing a cross-platform fairness metric:** Offering a platform-agnostic score for transaction comparison, suitable for Layer-1, Layer-2, and sharded environments.

RATEM, therefore, is not just an evaluation formula — it is a governance tool for aligning resource economics with system behaviour. It bridges the gap between execution semantics and economic accountability in blockchain systems.

Table 1: RATEM-based models and their purposes

Model Name	Purpose
ETOM (Execution-Time-Only Model)	Provide resource estimation using only wall-clock execution time as an indirect proxy
IREM (Instrumented Resource Evaluation Model)	Use direct profiling of system metrics to measure resource usage precisely
HGEM (Hybrid Gas Evaluation Model)	Combine execution time and measured resources for robust evaluation under partial information

This taxonomy demonstrates (Table 1) that RATEM is not a monolithic construct but a modular evaluation framework adaptable to various runtime conditions and validator capabilities. The Execution-Time-Only Model (ETOM), in particular, acts as a fallback mechanism when full instrumentation is unavailable while preserving evaluation integrity. Let’s take a closer look at these models.

Distinctiveness of RATEM. Unlike conventional gas-based evaluation models [10], which rely solely on opcode enumeration, RATEM incorporates real-time system metrics such as CPU usage, memory access, and I/O operations. While EIP-1559 [2] introduces dynamic base fees to reduce volatility, it does not account for the actual computational footprint of a transaction. Moreover, quality-of-service models discussed in [7] focus on user-defined priorities rather than system-aware scheduling.

RATEM distinguishes itself by supporting direct instrumentation (IREM) and hybrid evaluation (HGEM), which flexibly combines direct and indirect metrics and gracefully falls back to approximation when complete profiling is unavailable.

In this sense, RATEM provides a unified, extensible framework for performance-sensitive blockchain governance.

6.3 ETOM: Execution-Time-Only Model

The first model in RATEM (Table 2, page 11) is based on indirect measurement and is denoted as the **Execution-Time-Only Model** (ETOM), or alternatively as the **Time-Augmented**

Gas Evaluation. This approach relies exclusively on wall-clock execution time to estimate resource usage (eq. 1). While relatively simplistic, it is beneficial in systems where direct profiling is unavailable or too expensive. However, because execution time can be affected by unrelated factors such as I/O blocking or parallel task contention, this method is considered an approximation, suitable for low-precision evaluations or fallback scenarios.

$$\text{ETOM}(T) = \alpha_{\text{Time_exec}} \cdot \text{Gas}_{\text{base}} \quad (1)$$

Next, let’s look at the Extended Execution-Time-Only Model with Delay-Aware Adjustment. While the basic Execution-Time-Only Model (ETOM) offers a simplified approximation of transaction cost based on wall-clock execution time, it neglects other systemic factors contributing to a transaction’s effective load on the blockchain infrastructure. In particular, network propagation latency and mempool queuing time represent critical delays that impact system responsiveness, block inclusion predictability, and transaction throughput. To address these limitations, we propose an extended formulation of ETOM that integrates additional time-based metrics into the evaluation.

The extended ETOM score for a transaction T is defined as:

$$\text{ETOM}_{\text{extended}}(T) = \alpha_{\text{exec}} \cdot \text{Gas}_{\text{base}} + \beta_{\text{mempool}} \cdot \text{Delay}_{\text{mempool}}(T) + \gamma_{\text{net}} \cdot \text{Delay}_{\text{network}}(T) \quad (2)$$

Here:

- Gas_{base} denotes the static gas cost of transaction T as determined by EVM opcode pricing;
- $\text{Delay}_{\text{mempool}}(T)$ is the queuing time from transaction arrival to inclusion;
- $\text{Delay}_{\text{network}}(T)$ represents the latency between transaction submission and its recognition by validator nodes;
- $\alpha_{\text{exec}}, \beta_{\text{mempool}}, \gamma_{\text{net}}$ are scaling coefficients that control the relative impact of each factor.

This formulation captures the aggregate system impact of a transaction across three temporal dimensions: execution duration, queuing latency, and network dissemination time. By assigning appropriate weights to each component, validators and schedulers can fine-tune evaluation criteria based on their infrastructure constraints and fairness policies.

Such delay-aware modelling is particularly relevant for heterogeneous blockchain environments, including geographically distributed networks, Layer-2 aggregators, and systems with mempool overflow conditions. It provides a mechanism for penalising transactions that contribute disproportionately to congestion or introduce propagation lag, even if their base gas cost is nominally low.

The extended ETOM thus offers a more holistic approach to transaction cost estimation, bridging the gap between local execution metrics and global system-level dynamics. Its modularity allows for incremental deployment, wherein legacy nodes may retain simple timing estimators, while advanced validators incorporate all three components of Equation 2 into their prioritisation logic.

Table 2: Input parameters and advantages of RATEM-based models

Model Name	Input Parameters	Key Advantages
ETOM	Total runtime in milliseconds	Low overhead, usable in environments without profiling support
IREM	CPU cycles, memory accesses, system I/O traces	High accuracy, supports precise matching of fee and load, suitable for advanced validators
HGEM	Runtime, partial profiling data, network load indicators	Flexibility, graceful degradation, deployable in mixed environments

6.4 IREM: Instrumented Resource Evaluation Model

The second model (Table 2, page 11), the **Instrumented Resource Evaluation Model** (IREM), also referred to as the **Direct-Resource-Measured Gas Model** (DRMGM), is a High-accuracy evaluation using direct profiling. It leverages runtime instrumentation to collect system-level metrics such as CPU time, memory usage, and disk/network I/O traces. This model enables a detailed and reproducible accounting of the computational cost of transactions and is well-suited for high-precision environments such as performance-aware validators or enterprise-grade blockchain deployments. However, it typically requires additional infrastructure support and may introduce overhead due to profiling.

This model directly applies Equations 3 and 4 with metrics obtained via system instrumentation to replace static gas pricing with a dynamic evaluation based on actual resource usage.

$$\text{IREM}(T) = \sum_{i=1}^n w_i \cdot R_i(T) \quad (3)$$

where $R_i(T)$ is the normalized value of the i -th resource metric.

To support multi-dimensional evaluation of transaction impact, the IREM model considers a set of system-level metrics reflecting the actual computational and infrastructural cost of executing a transaction. Each transaction T is characterised by a vector of normalised resource indicators $R_i(T)$, where i denotes the i -th resource type. These indicators are dimensionless and scaled to the interval $[0, 1]$, allowing for consistent aggregation through weighted sums as shown in Eq. 3.

For example, the selected resource dimensions currently included in the model are:

- $R_1(T)$ – **CPU Execution Time:** Measures the total processing time consumed by the transaction on the validator node’s CPU, expressed in milliseconds or normalised CPU cycles. It reflects the computational complexity of smart contract execution or transaction validation routines. This metric is critical for identifying compute-intensive operations, such as cryptographic primitives or recursive function calls.
- $R_2(T)$ – **Memory Access Volume:** Quantifies the volume of memory read and write operations performed during transaction execution. It is measured in bytes or operations per second and indicates how intensively the transaction interacts with memory structures. This metric is particularly relevant for state-heavy smart contracts or large data structures operations.
- $R_3(T)$ – **I/O Load:** Represents the intensity of input/output interactions at the system level. This includes disk access (e.g., logs, checkpoints) and system calls that involve storage or network drivers. It is measured in data volume (e.g., bytes) or the number of I/O-related function calls. High I/O load may indicate transactions that stress the underlying hardware beyond what gas-based models capture.
- $R_4(T)$ – **Runtime Latency:** Denotes the wall-clock time elapsed from the beginning to the end of transaction execution, including internal processing delays and scheduling latencies. Although an indirect indicator, this metric helps identify bottlenecks, synchronous dependencies, or blocking operations that are not apparent from static cost analysis.
- $R_5(T)$ – **Network Overhead:** Captures the network-level resource cost imposed by the transaction, including the number of messages transmitted, size of propagated payloads, and communication required for consensus. Measured in bytes or message counts, this metric reflects how the transaction impacts propagation delay, mempool congestion, and validator coordination.

Each metric $R_i(T)$ is obtained through direct profiling, system-level monitoring, or indirect observation, and is normalised using min-max scaling or protocol-defined bounds. Aggregating these metrics via a weighted sum (see Eq. 3) yields a composite IREM score that enables granular and platform-independent transaction evaluation.

In practical settings, this model can be deployed at various layers of the blockchain infrastructure: within validator nodes to assess incoming transactions, in Layer-2 coordinators to optimise batch formation, or within fee prediction systems to support user interfaces.

To combine heterogeneous resource metrics — such as CPU execution time, memory usage, and I/O operations — into a unified evaluation formula, it is essential to normalise these inputs to a standard scale. Directly aggregating raw metric values would be inappropriate due to differences in units, magnitudes, and system-specific variations. For instance, memory usage measured in kilobytes may span vastly different from CPU time expressed in milliseconds, potentially biasing the evaluation outcome if combined without normalisation.

To address this, we adopt a standard min-max normalisation approach for each metric, denoted as $R_i(T)$, where i indexes the resource dimension and T denotes a specific transaction. The normalized metric $R_i^{\text{norm}}(T)$ is computed as follows:

$$R_i^{\text{norm}}(T) = \frac{R_i(T) - \min(R_i)}{\max(R_i) - \min(R_i)} \quad (4)$$

Here:

- $R_i(T)$ is the raw measured value of the i -th resource metric for transaction T ;
- $\min(R_i)$ and $\max(R_i)$ represent the minimum and maximum observed values for metric i within a defined evaluation window (e.g., across a block, epoch, or historical batch);
- $R_i^{\text{norm}}(T) \in [0, 1]$ is the resulting normalised score, where 0 indicates the lowest observed usage and 1 the highest.

This normalisation serves two primary objectives. First, it ensures that all resource metrics contribute comparably to the aggregated transaction cost, preventing any single metric from dominating due to unit magnitude alone. Second, it facilitates the application of weighted aggregation models such as Equation (3), where resource-specific weights w_i can be meaningfully interpreted and calibrated.

It should be noted that the normalisation window (i.e., the range over which min and max are computed) plays a critical role in ensuring stable and representative normalisation. In high-variability environments, such as permissionless public blockchains, dynamic or sliding windows may be used to adapt to system changes. Alternatively, fixed bounds derived from protocol specifications or empirical baselines may be employed to ensure stability and comparability across time.

Overall, Equation (4) provides a mathematically sound and operationally feasible mechanism for integrating diverse system-level metrics into a unified transaction evaluation framework.

6.5 HGEM: Hybrid Gas Evaluation Model

The third (Table 2, page 11) and most adaptable approach is the **Hybrid Gas Evaluation Model** (HGEM), also known as the **Composite RATEM Model**. It integrates indirect and direct measurements, adjusting its behaviour based on the availability of profiling data and the execution context. HGEM provides a robust middle ground between simplicity and accuracy using weighted combinations of available metrics and fallback logic (Alg. 1). It is particularly useful in heterogeneous validator networks where capabilities differ and a unified strategy must accommodate both full-featured and limited nodes.

Algorithm 1 HGEM Evaluation Algorithm

Require: Transaction T , metric set $\{R_1(T), \dots, R_n(T)\}$, execution time $\text{Time}_{\text{exec}}(T)$, base gas $\text{Gas}_{\text{base}}(T)$

Ensure: Hybrid transaction evaluation score $\text{HGEM}(T)$

```
1: Identify available resource metrics:  $K \subseteq \{1, \dots, n\}$  such that  $R_i(T)$  is present
2:  $k \leftarrow |K|$ ,  $n \leftarrow$  total number of supported metrics
3: Compute confidence:  $\lambda \leftarrow \frac{k}{n}$ 
4: if  $k > 0$  then
5:   for each  $i \in K$  do
6:     Normalize  $R_i(T)$  to  $R_i^{\text{norm}}(T)$  using Eq. (4)
7:   end for
8:   Compute direct score:  $\text{RAGEM}(T) \leftarrow \sum_{i \in K} w_i \cdot R_i^{\text{norm}}(T)$ 
9: else
10:   $\text{RAGEM}(T) \leftarrow 0$ 
11: end if
12: Compute fallback score:  $\text{ETOM}(T) \leftarrow \alpha_{\text{exec}} \cdot \text{Gas}_{\text{base}}(T)$ 
13: Compute final score:  $\text{HGEM}(T) \leftarrow \lambda \cdot \text{RAGEM}(T) + (1 - \lambda) \cdot \text{ETOM}(T)$  return  $\text{HGEM}(T)$ 
```

The Hybrid Gas Evaluation Model (HGEM) is proposed as a flexible and context-aware approach for estimating the cost of blockchain transactions in settings where profiling capabilities may vary across validator nodes. HGEM combines indirect (e.g., execution time) and direct (e.g., profiled system metrics) signals to produce a robust and adaptive evaluation score.

Let T denote a transaction. The HGEM score is defined as a weighted combination of the direct resource-aware model and the fallback time-based model:

$$\text{HGEM}(T) = \lambda \cdot \text{IREM}(T) + (1 - \lambda) \cdot \text{ETOM}(T) \quad (5)$$

Here:

- $\text{IREM}(T)$ is the resource-aware evaluation based on direct profiling (see Eq. 3, page 12) where $R_i^{\text{norm}}(T)$ is the normalized value of the i -th system resource (e.g., CPU time, memory, I/O), and w_i is its associated weight.
- $\text{ETOM}(T)$ is the execution-time-only approximation (see Eq. 1, page 11);
- $\lambda \in [0, 1]$ is the confidence coefficient reflecting the availability and reliability of profiling data, where Confidence coefficient:

$$\lambda = \frac{k}{n}$$

and k is the number of available profiled resource dimensions,

n is the total number of supported resource types. This formulation ensures smooth interpolation between the fallback and full-instrumentation cases.

Interpretation. When no profiling data are available ($\lambda = 0$), the model defaults to execution-time approximation. When full profiling is available ($\lambda = 1$), the model relies entirely on direct measurement. The evaluation combines both views proportionally for intermediate cases, enabling consistent and fair estimation even in heterogeneous validator environments.

Advantages. HGEM enables fine-grained transaction cost estimation in mixed-capability networks. It balances simplicity and accuracy, supports incremental deployment, and provides resilience against missing or partial telemetry. It is particularly suited to permissioned blockchains, L2 rollups, and hybrid public-private networks with non-uniform validator capacities.

To further examine the Hybrid Gas Evaluation Model (HGEM), we formulate an algorithm that dynamically combines direct and indirect resource estimates based on data availability. The algorithm accepts a transaction T and its associated metrics, including optionally profiled values for CPU, memory, I/O, and the execution time, as well as the base gas cost. The procedure computes a confidence coefficient λ as the ratio of available direct metrics to the total expected resource dimensions, thereby enabling context-aware interpolation.

Overall, this classification underscores the modularity and adaptability of the IREM framework. Rather than prescribing a single path to transaction evaluation, IREM offers a family of models tailored to varying operational environments, validator capacities, and performance goals. The combination of resource-aware cost evaluation and efficiency-driven transaction scheduling opens the path to a new generation of blockchain transaction governance systems that balance fairness, performance, and economic soundness.

6.6 Classification of Resource Metrics

Table 3 provides a structured classification of transaction-related resource metrics employed in the RATEM framework. Each metric represents a specific dimension of system-level resource consumption associated with transaction execution. The metrics are described with their corresponding measurement units, the nature of their availability (status), and the method of access (access level).

The following clarifications explain the *Status* and *Access Level* columns:

- **Direct Profiling:** Metrics that can be obtained directly through low-level system instrumentation or runtime profilers (e.g., CPU cycles, memory operations).
- **Indirect Observable:** Metrics not directly instrumented can be measured by monitoring system behaviour over time (e.g., execution latency, mempool delay).
- **Derived:** Metrics inferred or computed from other observables, often using validator-specific logic or consensus monitoring (e.g., consensus delay).
- **Fallback:** Proxy metrics such as dry-run execution time used in environments where direct profiling is unavailable. These values enable approximate but applicable estimates of transaction impact.

Table 3: Classification of Transaction Resource Metrics

Metric	Description	Unit	Status	Access Level
CPU time	Execution time of smart contract	ms / cycles	Direct	Profiling
Memory ops	Read/write memory operations	bytes / ops	Direct	Profiling
I/O ops	Disk/network I/O volume	bytes / calls	Direct	Profiling
Exec. latency	Wall-clock time to complete	ms	Indirect	Observable
Mempool delay	Time waiting for inclusion	blocks / ms	Indirect	Observable
Consensus delay	Time from execution to finality	ms / rounds	Indirect	Derived
Dry-run time	Validator pre-execution time	ms	Proxy	Fallback

Including direct (in IREM Model) and indirect (in ETOM Model) metrics ensures that the RATEM framework remains adaptable across different blockchain architectures, validator configurations, and performance monitoring capabilities. This taxonomy underpins RATEM’s flexible and extensible structure for comprehensive transaction evaluation.

7 Conclusion and Future Work

7.1 Summary of Contributions

This paper introduced RATEM (Resource-Aware Transaction Evaluation Model), a modular and extensible framework for assessing blockchain transactions based on real system metrics rather than abstract opcode enumeration. Unlike conventional gas-based evaluation, RATEM incorporates dimensions such as CPU time, memory operations, I/O activity, and runtime latency.

The core contribution lies in the formalisation of transaction scoring as a weighted aggregation of normalised resource indicators (Equation (3), (4)), with flexible integration mechanisms such as fallback timing models (Equation (5)) in hybrid environments. Three submodels – ETOM, IREM, and HGEM – were presented, each addressing a specific range of metric availability and validator capabilities.

7.2 Benefits and Theoretical Value

RATEM provides a principled foundation for resource-sensitive evaluation, enabling:

- **Transparency:** Transaction scores are derived from measurable execution properties;
- **Cross-platform comparability:** Normalised metrics allow evaluation across diverse environments;
- **Operational adaptability:** The hybrid HGEM model ensures resilience to incomplete data;
- **Improved scheduling logic:** Cost attribution reflects real system impact, aiding validator-side prioritisation.

These benefits promote fairness, accountability, and governance-aware transaction handling, particularly in performance-sensitive or heterogeneous blockchain systems.

7.3 Future Work Directions

RATEM opens several practical and theoretical avenues for further research:

Short-term directions:

- *Experimental deployment in testnets or permissioned networks* to measure runtime behaviour, scheduling impact, and system overhead.
- *Integration with EVM-based validation stacks* through middleware or modified node clients supporting profiling and evaluation hooks.

Long-term directions:

- *Machine learning-based calibration of weights* using historical transaction traces to improve accuracy and responsiveness;

- *Formal verification of fairness and robustness* under adversarial submission strategies or incomplete metric scenarios;
- *Adoption by Layer-2 rollup coordinators* to improve batch selection and DoS mitigation by leveraging resource-sensitive scores.

7.4 Final Statement

RATEM represents a viable and theoretically sound step toward performance-aware blockchain governance. Its modularity and metric-driven logic make it compatible with evolving validator infrastructures. With targeted implementation efforts, RATEM may be a foundational component in future transaction scheduling protocols and resource management strategies in public and permissioned blockchain platforms.

References

- [1] Shehar Bano, Alberto Sonnino, et al. Sok: Consensus in the age of blockchains. In *ACM CCS*, 2019. doi: 10.1145/3319535.3339814.
- [2] Vitalik Buterin et al. Eip-1559: Fee market change for eth 1.0 chain. 2021.
- [3] Yao Chen, Chao Lin, et al. Gaschecker: Detecting gas-underestimating bugs in smart contracts. In *USENIX Security Symposium*, 2020. URL <https://www.usenix.org/conference/usenixsecurity20/presentation/chen-yao>.
- [4] Ethereum Foundation. Estimating gas. URL <https://ethereum.org/en/developers/docs/gas/estimating-gas/>. Ethereum Developer Documentation.
- [5] Adem Efe Gencer, Soumya Basu, Ittay Eyal, Robbert Van Renesse, and Emin Gun Sirer. Decentralization in bitcoin and ethereum networks. *Proceedings of the International Conference on Financial Cryptography and Data Security*, pages 439–457, 2018. doi: 10.1007/978-3-662-58820-8_26.
- [6] Arthur Gervais, Ghassan O Karame, Karl Wüst, Vasileios Glykantzis, Hubert Ritzdorf, and Srdjan Capkun. On the security and performance of proof of work blockchains. *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pages 3–16, 2016. doi: 10.1145/2976749.2978341.
- [7] Wei Li, Stefano Andreina, Ghassan Karame, and Srdjan Capkun. A survey on the scalability of blockchain systems. *ACM Computing Surveys (CSUR)*, 55(1):1–36, 2022. doi: 10.1145/3485124.
- [8] Y. Liu, X. Xu, and L. Zhang. Performance analysis of blockchain systems: A survey. *IEEE Systems Journal*, 15(1):411–421, 2021. doi: 10.1109/JSYST.2020.3019505.
- [9] Tao Wang, Yuming Li, Lei Wang, and Hao Chen. Gas usage analysis and optimization in ethereum smart contracts. *IEEE Access*, 9:13156–13170, 2021. doi: 10.1109/ACCESS.2021.3052022.
- [10] Gavin Wood. Ethereum: A secure decentralised generalised transaction ledger. 2014. URL <https://ethereum.github.io/yellowpaper/paper.pdf>.
- [11] Feng Zhang, Tao Xie, Wen Li, Yao Zhou, and Huaimin Wang. A comprehensive evaluation of consensus algorithms in permissioned blockchains. *Future Generation Computer Systems*, 107:475–489, 2020. doi: 10.1016/j.future.2020.01.007.

Appendix

Contents

1	Problem Formulation and Motivation	2
1.1	Background	2
1.2	Identified Gaps	3
1.3	Research Motivation	3
1.4	Research Objective	3
1.5	Structure of the Paper	3
2	Challenges in Transaction Evaluation in Blockchain Networks	4
2.1	Competitive Load Environments	4
2.2	Low Load Inefficiencies	4
2.3	Summary of Limitations	4
3	Limitations of the Traditional Gas-Based Evaluation Model	5
3.1	Subjectivity and Market Dependency	5
3.2	Abstract Resource Metrics	5
3.3	Implications for System-Level Optimisation	5
3.4	Fundamentals of Gas-Based Evaluation	5
3.5	Transaction Lifecycle and Evaluation Timing	5
3.6	Heuristic vs. Actual Resource Consumption	6
3.7	Network Incentives and Inclusion Priorities	6
3.8	On “Light” and “Heavy” Transactions	6
3.9	Root Cause of Unfair Evaluation	6
3.10	Summary and Implications	7
4	Overview of Existing Solution Approaches	7
4.1	Enhanced Gas Mechanisms	7
4.2	Off-Chain Evaluation Strategies	7
4.3	Resource-Aware Metrics	7
4.4	Machine Learning-Based Estimation Models	7
4.5	Quality-of-Service (QoS) Models	8
4.6	Synthesis and Gaps	8
4.7	Systemic Gaps in Gas-Based Transaction Evaluation	8
5	Introduction to the Resource-Aware Transaction Evaluation Models (RATEM)	8
5.1	Rationale	8
5.2	Model Concept	8
5.3	Target Metrics	8
5.4	Objectives	9
6	Taxonomy and Classification of RATEM-Based Models	9
6.1	RATEM as a Dynamic Adjustment to Stateless Gas Models	9
6.2	Extended Objectives of RATEM	10
6.3	ETOM: Execution-Time-Only Model	10
6.4	IREM: Instrumented Resource Evaluation Model	12
6.5	HGEM: Hybrid Gas Evaluation Model	13
6.6	Classification of Resource Metrics	15

7	Conclusion and Future Work	16
7.1	Summary of Contributions	16
7.2	Benefits and Theoretical Value	16
7.3	Future Work Directions	16
7.4	Final Statement	17

List of Tables

1	RATEM-based models and their purposes	10
2	Input parameters and advantages of RATEM-based models	11
3	Classification of Transaction Resource Metrics	15