Article

Not peer-reviewed version

Securing Snapshot Pruning in IOTA Tangle 2.0: A Cooperative Deep Reinforcement Learning Approach for Edge-Cloud Smart Meter Networks

Basker Palaniswamy * and Paolo Palmieri

Posted Date: 6 August 2025

doi: 10.20944/preprints202508.0413.v1

Keywords: IOTA tangle;Web 3.0; blockchain security; blockchain for IoT; smart meters; energy trading

 doi:10.20944/preprints202508.0413.v1

Article

Securing Snapshot Pruning in IOTA Tangle 2.0: A Cooperative Deep Reinforcement Learning Approach for Edge-Cloud Smart Meter Networks

Basker Palaniswamy * and Paolo Palmieri

Insight Centre for Data Analytics, School of Computer Science and Information Technology, University College Cork, Cork, Ireland

* Correspondence: basker170889@gmail.com

Abstract

Decentralized security architectures powered by distributed ledger technology (DLT) have become prominent solutions for elevated security in Web 3.0. The IOTA Tangle is an open-source DLT, available for the Internet of Things (IoTs), as transactions can be carried out freely. IOTA 2.0 is the recent version of the IOTA Tangle, yet its local snapshot mechanism has certain drawbacks — security violations, increased costs and a lack of centralized coordination. Specifically, the situation-unaware decision for pruning increases the cost for energy service providers in the smart meter setting. Data analytics becomes infeasible due to the localized pruning policies. To address these issues, we develop a deep-Q actor-critique network-based cooperative pruning algorithm for IOTA 2.0 that preserves cryptographic properties for stored data in every edge server, namely, first and second preimage resistance, collision resistance and integrity. Our informal security analysis against active adversaries reveals the satisfaction of the security properties. Extensive simulation using the Monte Carlo method unveils that the proposed method is efficient regarding 30% improvement in storage reduction and 35% improvement in energy reduction, committing to the lower bound in solution. Our study quantifies the cost benefits of the proposed algorithms for five random cities with varying population densities for the years 2025 to 2029. We implement and test the algorithms in a real-world setting to benefit diverse stakeholders. Our solutions can be adopted by energy companies for cost-benefit and the IOTA Tangle foundation for future up-gradation of its snapshot mechanism.

Keywords: IOTA tangle; Web 3.0; blockchain security; blockchain for IoT; smart meters; energy trading

1. Introduction

Web 3.0 is the next evolution in Internet technology, shifting the paradigm of centralized trusted architecture to decentralized architecture with the utility of Artificial Intelligence [1]. Blockchain is the underlying technology in Web 3.0 that fosters decentralized trust on the Internet, controlling identities and data by users rather than companies. Smart contracts, decentralized finance, and bitcoin are examples of blockchain applications [2]. Industries are adopting blockchain in their service architecture as it facilitates a unique dimension of user autonomy. Blockchains are categorized into three types: public, private and consortium. The public blockchain is decentralized and open, such as Bitcoin, Ethereum and Solana. The private blockchain is permissioned. For example, Hyperledger Fabric, Alibaba blockchain, IBM blockchain, Microsoft Azure blockchain, Meta blockchain and Google Cloud blockchain are private blockchain offerings from big tech companies. The consortium blockchains, like R3 Corda and Quorum, are partially decentralized. Needless to say the sole purpose of blockchain is lost when they are offered as a service on a payment basis. However, targeting the **Internet of Things (IoTs)**, the **IOTA Tangle** is an open-source **Distributed Ledger Technology (DLT)** where there are

 doi:10.20944/preprints202508.0413.v1

no miners and chargeable transactions like Ethereum [1]. Especially, as the transactions (smart meter readings) and utility are not charged, anyone can use the stack to avail of free blockchain in their service architecture. On the flip side, it is under development, which means that people are actively researching to improve it from known vulnerabilities and bottlenecks in deployment [3].

Differing from the traditional blockchain with the proof of work (PoW) and Proof of Stake (PoS), the IOTA Tangle works by **Directed Acyclic Graph (DAG)**. Every transaction in the Tangle network is referred to as a site (approved). Accordingly, the first approved transaction is the genesis site in the network. To add new transactions in the Tangle, transactions are locally verified (signature) by a node. To add the verified transactions (site) in the network, the local node attaches the transaction (tip) to two sites with the tip (not yet approved). Thereafter, the Tip Selection Algorithm (TSA) executes the process to approve the tip in order to convert it to a site [4]. It is noted that every site in the Tangle carries the specifics of a traditional blockchain, say transaction hashes, block signature and Merkle root. The recent version of the IOTA Tangle is **IOTA 2.0 (Coordicide)**, and its previous version is **IOTA 1.5 (Chrysalis)**. Using conventional blockchain for the energy industry is in practice [5], and its applicability is even wider, starting from the energy-producing sector to the energy-selling market [6].

We consider a network architecture with the IoT setting, shown in Figure 1 where the smart meters from homes and industries are connected to the edge servers via LORA technology. The edge servers are rented by the energy companies from **cloud service providers (CSPs)**, such as Amazon Web Services (AWS), Microsoft Azure, Oracle Cloud and Google Cloud. CSPs and energy companies are in a **software-level agreement (SLA)**. Energy companies utilize IOTA 2.0 as the blockchain architecture (Tangle network) to offer services, like smart contracts and smart meter users. Particularly, the Tangle network utilizes a deep Q actor-critic network-enables a reinforcement learning algorithm, where the actor (DQ-AN) and critic (DQ-CN) jointly work to obtain the optimal policy of storage, energy and pruning frequency. By pruning, we mean removing expired blocks. For this, an agent trains the network to minimize the loss by updating the parameters, and the agent works in a model-free architecture using the replay memory as a minibatch for sampling the (state, action, reward) tuple. Our architecture is an extension of the blockchain architecture considered for the smart meter application [7]. Notably, blockchain technology is widely applied in the energy sector, focusing the energy trading [8].

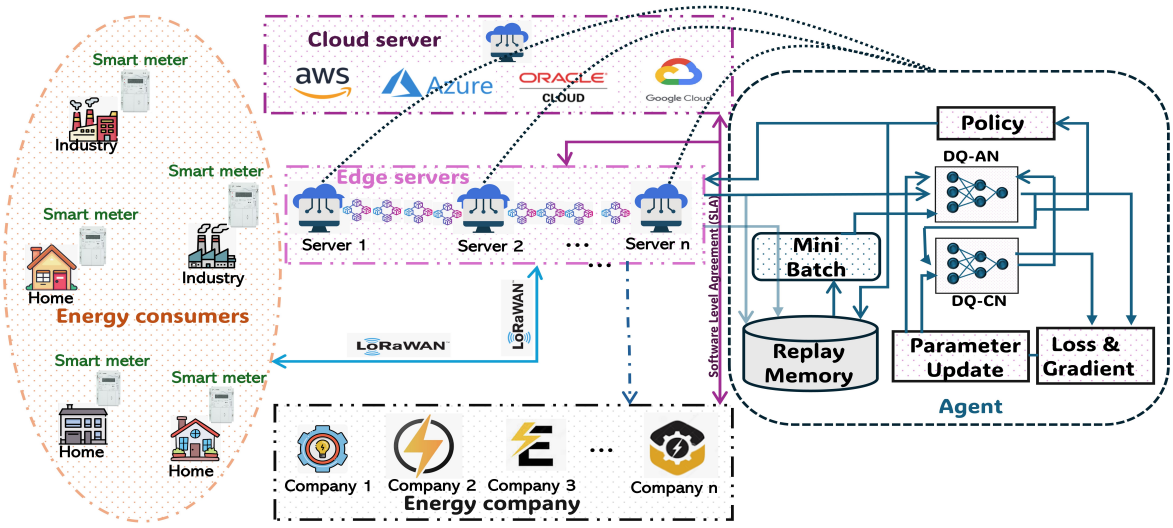

Figure 1. Overall Architecture.

When IOTA 2.0 is utilized, energy providers cannot attain security, cost-effectiveness, and data analytics capability, which are prominent in securing the network and offering sophisticated services to users from the supplier's point of view. For optimizing the network, the pre-Chrysalis version used the **global snapshot** mechanism, facilitating network-wide coordination. The global snapshot

[1] https://www.iota.org/

 doi:10.20944/preprints202508.0413.v1

requires a coordinator to carry out the task, and it demands network-wide consistency to attain the pruning. Nonetheless, IOTA 2.0, is upgraded by the **local snapshots** where individual nodes can execute the pruning to optimize the storage [2]. Though the local snapshot mechanism solves the issue of the requirement for a global coordinator and the process for maintaining network-wide unanimity, further improvement is still required to safeguard the security of the network. Specifically, the local snapshot is governed by static rules, i.e., nodes execute the pruning decision when the threshold limit for the storage is reached locally. This decision can lead to over-pruning or under-pruning, violating the storage requirement of minimum transactions to safeguard the security of the network. Also, as the global snapshot is deprecated in IOTA 2.0, we highlight that the lack of a coordinator can lead to failure in data analytics, blocking the utility of machine learning methods for electricity price forecasting [9]. Figure 2 shows the problems associated with the local snapshots and how they are overcome in the proposed method by the Actor-Critic-based Deep Q network. Considering the challenges faced by the local snapshot mechanism in IOTA 2.0, we pose the following question: "*How can pruning decisions in IOTA 2.0 be made secure and efficient in decentralized edge networks?*" This research question opens avenues to solve the problem in two-fold: developing methods to minimize the cost of the energy providers and developing methods to solve the issues associated with the local snapshot mechanism of IOTA 2.0. As optimising the storage directly reduces the number of edge servers and imposing security constraints while reducing the storage prevents the adversarial attacks, this research question is worth solving.

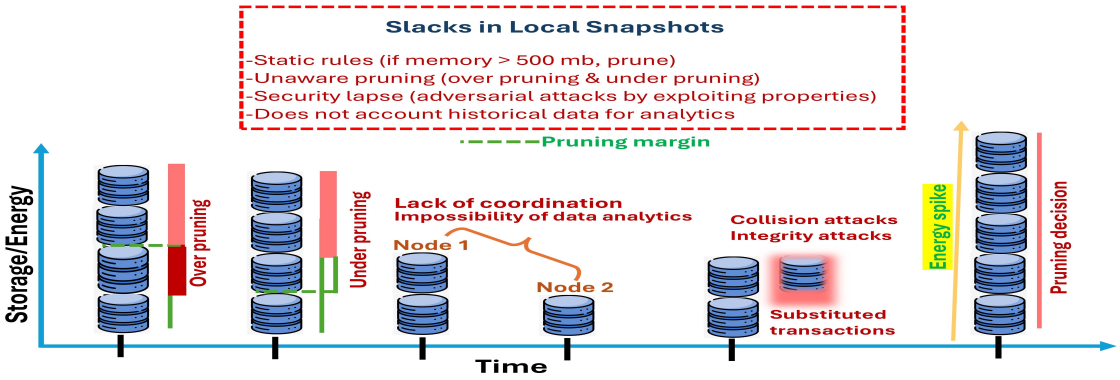

Figure 2. Issues in Local Snapshot.

By answering the research question, we lay out the following contributions of this article:

- We formulate the cost optimization problem as a function of the storage, energy and pruning frequency optimization for optimizing the policy of energy and storage consumption and pruning frequency, respectively. Precisely, we optimize storage and energy by strictly satisfying security constraints in sub-problem 1. Then, we jointly optimize the pruning frequency in sub-problem 2, where constraints are relaxed to attain a feasible solution.
- We develop Actor-Critic method-based deep reinforcement learning algorithms (DQACN) for deep policy optimization, working cooperatively when they are deployed in the edge servers with a cloud service provider.
- Extensively, we perform the Monte Carlo simulation to gain confidence in the performance improvement of proposed algorithms over conventional local snapshots, global snapshots, ordinary deep Q network and random elimination methods.
- We implement and test the proposed algorithms to evaluate their performance robustly in the real-world environment.

The remainder of the article is organized as follows: Section 2 presents the related work. Section 3 presents problem formulation, introducing IOTA 2.0. Section 4 put forth the solution approach,

[2] https://files.iota.org/papers/Coordicide_WP.pdf

 doi:10.20944/preprints202508.0413.v1

convergence proofs for the proposed algorithms and formal security analysis in the universal composability framework. The next section (5) evaluates the proposed algorithms and other methods using the Monte Carlo simulation. Section 6 carries out the implementation study of the proposed algorithm in a real-world setting. Finally, the work is concluded in Section 7.

2. Related Work

Aiming to present the taxonomy in storage optimization of the blockchain, the study [10] categorizes storage optimization in blockchain into three types: replication-based, redaction-based and content-based. Storage optimization in conventional blockchains is targeted by replication-based methods, whereas the redaction-based and content-based methods find optimization in the service and application layers. This study clarifies that storage optimization in the IOTA Tangle has not been done, so we take our breadth-first approach towards optimizing the storage and its associated cost in the IOTA Tangle by preserving security.

Xu et al. [11] presented the storage optimization in blockchain. Their work formulates the objective function as a multi-objective function, and their storage model captures the disk space of the servers used in the cloud environment. However, they propose the genetic algorithm-based solution approach, which usually takes a longer time to find the global optimum solution.

Zuo et al. [12] put forth a three-level, involving IoT devices, a base station and a cloud server, Stackelberg game-based optimal cloud resource utilization model involving IoT devices, base station, and cloud server. This method shows how to cooperatively allocate cloud resources for blockchain storage. Moreover, it lays out the resource allocation model, computing service pricing model and computing, storage and communication model. This work captures mining in the blockchain; however, their work does not involve optimizing the blockchain itself, which directly contributes to the storage space.

Targeting the minimization of energy consumption and carbon emission, Alofi et al. [13] studied the multi-objective optimization for the Bitcoin environment. Their work considers a variety of objectives ranging from reputation to carbon emissions. They present the Pareto front for the different combinations of objective functions, minimizing the convex function with the obtained hypervolume set. Nonetheless, they used the evolutionary algorithm as the solution approach, which takes a longer time to converge and is unsuitable for the considered application in the edge-based cloud network.

Akrasi-Mensah et al. [14] discussed the usage of the deep reinforcement learning algorithm for solving storage optimization in the cloud. Specifically, they considered a blockchain network for the cloud where storage cost is optimized. The solution approach consists of an Actor-Critic method-based optimal policy selection algorithm. Though this approach presents the applicability of the deep reinforcement algorithm as the solution approach, this method does not consider edge networks, i.e., the method applies to the traditional cloud network model.

Zhou et al. [15] presented a way of optimizing the storage cost in the blockchain. The problem is formulated as a multi-objective optimization and reduced to the block selection problem. For optimizing the blockchain, the storage model captures the blockchain as individual blocks with a finite size. Their solution approach considers the traditional deep Q network, which overcomes the curse of dimensionality in the conventional reinforcement learning methods. However, the blocks in the blockchain do not capture the nuance of transactions, signatures and the Merkle root of the block, which is essential to assess the security of the optimization.

Considering a consortium blockchain network, Tanis et al. [16] proposed a cooperative framework for energy trading using the Nash equilibrium [17]. In contrast to the traditional single-energy model, this work takes into account the multi-energy model. Q-algorithm in reinforcement learning is used to solve the nonlinear multi-objective optimization. However, this work does not consider storage optimization, though it leverages the cooperative framework.

In general, in the storage optimization of the blockchain and the granularity of modeling the blocks (precise modeling of components of a block/site) in the blockchain are absent, though optimization

 doi:10.20944/preprints202508.0413.v1

for improving throughput is evident [18]. Importantly, security constraints have not been considered to preserve security. Further, blockchain optimization for the edge-assisted cloud network has not been considered, specifically implementing the IOTA Tangle. Cost optimization for energy service providers renting the cloud server from CSPs using the IOTA Tangle network is not reported in the literature, which is a thriving market.

3. Problem Formulation

In this section, we formulate the energy model and storage model of IOTA 2.0 (Coordicide), followed by formulating optimization problems for energy cost, storage cost and pruning frequency. While IOTA 2.0 does not use the proof of work of the traditional blockchain, it retains many features of the blockchain, such as signature computation and Merkle root retention for transactions (sites), so we introduce the features of IOTA 2.0 and compare them with the conventional blockchain. Then, we prove that the presented optimization problem is NP-hard.

3.1. IOTA 2.0 (Coordicide)

IOTA 2.0 is referred to as a Tangle rather a blockchain since the network is formed by a DAG. Transactions in IOTA 2.0 are called sites. Each site is referenced by two previous transactions (tips). When a node wants to add a transaction, it places the site (unapproved transaction) in the tangle by referring to two previous tips (approved transactions). Then, the Tip Selection Algorithm (TSA) approves the added site to a tip with the help of the referenced two tips. Thus, every site in the tangle is referenced by two tips.

The proof of work is replaced by the mana-based system and Fast Probabilistic Consensus (FPC). Mana is a reputation-based resource bound with every node. This is introduced to have a full-fledged decentralization, as the coordinator in the previous version (IOTA 1.5) is replaced. Every node generates mana by holding IOTA tokens (stake-based) and through its activity (transactions). There are two types of manas, viz, access mana and consensus mana. The access mana determines the node's access to the network, that is, the number of transactions issued by the node. Consensus mana defines the weight of the node when it participates in voting during FPC. The mana of a node is used for at least three purposes: protection, control and weighing. Briefly, Mana prevents the Sybil attack by giving importance to token ownership. This prevents malicious nodes from gaining disproportionate access in the tangle by creating fake identities. Mana regulates the transaction issuance rate. Nodes with higher mana are preferred during network congestion. Consensus mana of every node carries wight during voting in FPC. That is, nodes with higher consensus mana have an advantage in conflict resolution as their votes are weighed higher. Table A1 (Appendix B) compares IOTA 2.0 with the proof of work (PoW) and proof of stake (PoS) of the traditional blockchain.

Remark: While the coordinator (centralized and recommended in IOTA 1.5) is deprecated in IOTA 2.0, deployment of the global coordinator has some benefits. For instance, aggregated data retention for future data analytics can be easily accomplished with such a central coordinator though it does not hold any important roles centrally, like the global snapshot (pruning decision is centrally coordinated).

3.2. Energy Model

Let the total number of edge servers be $N = \{1, 2, ..., i, ..., N\}$. Out of the edge servers, a subset of servers ($C[n]$) is responsible for carrying out FPC. Table 1 elaborately presents the different notations and their meaning. The pruning decision for optimizing the storage relates to energy consumption.

The cost for edge servers rented by the energy companies depends on the energy consumed by the tangle network. Therefore, it is essential to precisely model the energy cost associated with the tangle network. Without loss of generality, we present the energy cost for the tangle network as the cost incurred for transaction, tip, consensus and pruning.

 doi:10.20944/preprints202508.0413.v1

Table 1. Nomenclature.

Notation	Description
Generic Variables	
N	Total number of edge servers
n	Time
$C[n] \subset N$	Subset of edge servers participating in FPC consensus at time n
k_h	Energy cost per hash computation
k_s	Energy cost per signature computation
k_p	Energy cost per pruning operation
$k_{\text{fpc},i}$	Energy cost per FPC query/vote operation at server i
$k_{\text{mana},i}$	Energy cost per mana update operation at server i
$k_{\text{attach},i}$	Energy cost per message attachment operation at server i
α_i	Efficiency factor of server i
$m_i[n]$	Mana of server i at time n
ψ	Nodes involved in the consensus (binary variable)
δ_m	Mana decay rate
κ_m	Mana scaling factor
S_i	Set of smart meters pledging mana to server i
Message Variables	
$\lambda_T^i[n]$	Number of messages processed by server i at time n
$n_{\text{hash}}^i[n]$	Number of hash computations per message at server i
$n_{\text{sig}}^i[n]$	Number of signature verifications per message at server i
$Q_i[n]$	Number of FPC queries/votes by server i at time n
$M_i[n]$	Number of mana updates due to message issuance by server i at time n
$D_i[n]$	Number of mana decay updates by server i at time n
Pruning Variables	
$\lambda_P^i[n]$	Fraction of messages pruned by server i at time n
$M_{\text{backlog}}^i[n]$	Total unpruned messages at server i at time n
$M_{\text{finalized}}^i[n]$	Number of finalized messages at server i at time n

The total energy consumption across all edge servers at time n in IOTA 2.0 is given by .

$$E_{\text{total}}[n] = \sum_{i=1}^{N} \left(E_{\text{T},i}[n] + E_{\text{tip},i}[n] + E_{\text{C},i}[n] + E_{\text{P},i}[n] \right). \tag{1}$$

The cost associated with a transaction can be decomposed into the cost incurred for the hash computation, signature computation, attaching cost and the number of transactions. Accordingly, Equation (2) presents the transaction cost for the i^{th} server.

$$E_{\text{T},i}[n] = \lambda_T^i[n] \cdot \left(k_h \cdot n_{\text{hash}}^i[n] + k_s \cdot n_{\text{sig}}^i[n] + k_{\text{attach},i} \right). \tag{2}$$

The consensus cost is presented in Equation (3).

$$E_{\text{C},i}[n] = \psi_{i \in C[n]} \cdot \frac{k_{\text{fpc},i} \cdot Q_i[n] + k_{\text{mana},i} \cdot M_i[n]}{\alpha_i}, \tag{3}$$

where $M[n]$ denotes the mana, and it is given in (4). We assume the smart meters delegate their mana to the edge servers (pledging) as they have limited computing resources. The mana of server i at time n is updated to include pledged mana from the smart meters in the below equation that presents the

 doi:10.20944/preprints202508.0413.v1

time evolution model of the mana. This equation, also, captures the notion of decreasing mana of a server in the event of inactivity.

$$m_i[n] = m_i[n-1] \cdot e^{-\delta_m} + \kappa_m \cdot \lambda_T^i[n-1] + \sum_{j \in S_i} m_j[n-1] \tag{4}$$

The energy associated with tip selection for server i at time n in IOTA 2.0 can be expressed as follows:

$$E_{\text{tip},i}[n] = \lambda_T^i[n] \cdot \left(k_h \cdot n^i_{\text{hash-tip}}[n] + k_s \cdot n^i_{\text{sig-tip}}[n]\right). \tag{5}$$

The energy associated with pruning operations (local snapshot pruning in IOTA 2.0) for server i at time n in IOTA 2.0 is given by:

$$E_{\text{P},i}[n] = \psi_i \cdot \lambda_P^i[n] \cdot k_p. \tag{6}$$

The total energy associated with the i^{th} server is presented in the Equation (8) by substituting Equations (2), (3), (5) and (6).

$$\begin{aligned} E_{\text{total}}[n] = \sum_{i=1}^{N} E_{\text{server},i}[n] = \sum_{i=1}^{N} \Bigg(& \lambda_T^i[n] \cdot \left(k_h \cdot n^i_{\text{hash}}[n] + k_s \cdot n^i_{\text{sig}}[n] + k_{\text{attach},i}\right) \\ & + \psi_{i \in C[n]} \cdot \frac{k_{\text{fpc},i} \cdot Q_i[n] + k_{\text{mana},i} \cdot M_i[n]}{\alpha_i} \\ & + \lambda_T^i[n] \cdot \left(k_h \cdot n^i_{\text{hash-tip}}[n] + k_s \cdot n^i_{\text{sig-tip}}[n]\right) \\ & + \psi_i \cdot \lambda_P^i[n] \cdot k_p \Bigg) \end{aligned} \tag{7}$$

$$\begin{aligned} E_{\text{total}}[n] = & \sum_{i=1}^{N} \lambda_T^i[n] \cdot \left(k_h \cdot n^i_{\text{hash}}[n] + k_s \cdot n^i_{\text{sig}}[n] + k_{\text{attach},i}\right) \\ & + \sum_{i=1}^{N} \psi_{i \in C[n]} \cdot \frac{k_{\text{fpc},i} \cdot Q_i[n] + k_{\text{mana},i} \cdot M_i[n]}{\alpha_i} \\ & + \sum_{i \in P[n]} \lambda_T^i[n] \cdot \left(k_h \cdot n^i_{\text{hash-tip}}[n] + k_s \cdot n^i_{\text{sig-tip}}[n]\right) \\ & + \sum_{i=1}^{N} \psi_i \cdot \lambda_P^i[n] \cdot k_p \end{aligned} \tag{8}$$

3.3. Storage Model

The storage dynamics over time are captured by the storage model of the time evolution equation. For the discrete time instance n, the storage is given by $S[n-1]$, i.e., storage contributed from the previous time instance, where $S[0] \in \varnothing$. Equation (9) depicts the total storage contributed at n by the i^{th} server, where S_{tx} and $\lambda_T^i[n] \in [0,1]$ denotes the storage size per transaction and coefficient for the transaction of the i^{th} server, respectively.

$$S_S^i[n] = \lambda_T^i[n] \cdot S_{\text{tx}} \tag{9}$$

The quantity of storage freed by pruning is modeled in the Equation (10). S_P specifies the storage cost eliminated by pruning, and $\lambda_P^i[n] \in [0,1]$ denotes the act of pruning done at the i^{th} server at the instance n.

$$S[n]_F^i = \lambda_P^i[n] \cdot S_{\text{P}} \tag{10}$$

 doi:10.20944/preprints202508.0413.v1

The total storage cost for N servers can be framed as per Equation (12) by combining Equations (9) and (10).

$$S[n] = S[n-1] + \sum_{i=1}^{N} \left(\lambda_T^i[n] \cdot S_{\text{tx}} - \lambda_P^i[n] \cdot S_{\text{P}} \right) \tag{11}$$

$$S[n] = S[n-1] + \sum_{i=1}^{N} \lambda_T^i[n] \cdot S_{\text{tx}} - \sum_{i=1}^{N} \lambda_P^i[n] \cdot S_{\text{P}} \tag{12}$$

3.4. Problem Formulation

In this study, our aim is to optimize storage and energy as they directly contribute to the cost of an edge server, thereby increasing the monetary benefit. The key performance indicator of the objective function is linked to the cost-benefit. While optimizing the storage and energy, constraints, namely security constraints, software level agreement (SLA), bill settlement and retention of historical data have to be strictly satisfied. Therefore, we formulate sub-problem 1 (**SP 1**) to optimize the storage and energy by strictly meeting these constraints (constraint satisfaction problem (CP)). Sub-problem 1 is formulated to optimize storage and energy individually. Then, in the sub-problem 2 (**SP 2**), we jointly optimize the pruning frequency, energy and storage. We formulate each objective function using regret, as the regret-based formulation captures dynamic scenarios in comparison with the usual static formulation. The proposed optimisation is replication-based.

$$R(T) = \sum_{t=1}^{T} (C^*[t] - C[t]) \tag{13}$$

In the above equation, $C[t]$ represents the cost at time t, and $C^*[t]$ specifies the best possible cost in hindsight. By Equation (14), we intend to minimize the energy consumption by choosing the best policy $\pi_i^* \in \Pi$ from the set of policies $\{\pi_i\}_{i=1}^{i=n}$, where $\pi : S \to A$, i.e. a mapping function from state to action (refer to next section).

SP 1:

$$\max_{\pi_i^* \in \Pi} \min_{E[n]} R_{\text{storage}}(T) = \max_{\pi \in \Pi} \min_{S[n]} \sum_{t=1}^{T} (S_{\text{total}}^*[t] - S_{\text{total}}[t]) \tag{14}$$

Subject to:

$$C_1: \quad S[n] \geq \min \left(\frac{2^{256}}{B_{\text{hash}}}, \frac{2^{256}}{B_{\text{tx}}}, \frac{2^{128}}{B_{\text{hash}}}, \frac{2^{256}}{B_{\text{sig}}} \right) \in [10^{28}, 10^{71}] \tag{14a}$$

$$C_2: \quad \lambda_P^i[n] \leq B_{\text{paid},i}[n], \quad \forall B_{\text{paid},i}[n] \in \{0,1\} \tag{14b}$$

$$C_3: \quad S_H[n] \geq N_{\text{ret}} \times S_{\text{tx}}, \quad \forall S_{\text{tx}} \in [0,\infty) \tag{14c}$$

$$C_4: \quad S_{\text{edge}}[n] \leq S_{\text{edge}}^{\max}, \quad \forall S_{\text{edge}}^{\max} \in (0,\infty) \tag{14d}$$

$$C_5: \quad S_{\text{cloud}}[n] \geq S_{\text{cloud}}^{\min}, \quad \forall S_{\text{cloud}}^{\min} \in (0,\infty) \tag{14e}$$

$$C_6: \quad \lambda_P^i[n] \leq \lambda_P^{\max}, \quad \forall \lambda_P^{\max} \in [0,1] \tag{14f}$$

$$C_7: \quad H_{\text{cloud}}[n] \geq H_{\text{cloud}}^{\min}, \quad \forall H_{\text{cloud}}^{\min} \in (0,\infty) \tag{14g}$$

$$C_8: \quad S_H[n] \geq S_H^{\min}, \quad \forall S_H^{\min} \in (0,\infty) \tag{14h}$$

Security constraints for a 256-bit security level are presented in C_1. In C_1, $\frac{2^{256}}{B_{hash}}$ presents the security constraint for the first preimage resistance (given a hash, finding its input is computationally infeasible). This constraint imposes search space complexity bounded to computation time for a hash (B_{hash}), i.e., when time $t \to \infty$, $2^{256} \to 0$. $\frac{2^{256}}{B_{block}}$ presents the security constraint for the second pre-image resistance, that is, given a hash and its input, finding its second input is computationally infeasible. $\frac{2^{128}}{B_{hash}}$ presents the constraint for collision resistance property, that is, finding two distinct

 doi:10.20944/preprints202508.0413.v1

inputs for the same hash output is computationally infeasible. Due to the birthday paradox, 2^{256} is reduced to 2^{128}. $\frac{2^{256}}{B_{sig}}$ presents the integrity constraint. We assume that 1 billion hashes can be produced in a second, though in Bitcoin mining, 1 trillion hashes per second are produced. Accordingly, the minimum values obtained are in the order of 10^{28}. For the integrity, the maximum value is obtained in the order of 10^{71} as IOTA 2.0 uses Winternitz One-Time Signatures (WOTS), which accounts for 1 million signature verifications per second. C_2 presents the pruning constraints on blocks that are settled for bills. $B_{\text{paid},i}[n]$ is 1 if the bill is settled; otherwise, its value is 0, indicating not ready for pruning. C_3 presents the number of blocks retailed as part of the historical data. Constraints from C_4 to C_8 present software level agreement (SLA) constraints. Electricity and cloud service providers agree on an SLA to rent resources. Specifically, C_4 depicts the storage capacity of the edge server should not exceed its maximum capacity. C_5 presents the constraint that at the cloud level, the minimum storage has to be ensured for future audits. C_6 insists the pruning frequency should not exceed its limit as it increases energy consumption. C_7 presents at the cloud, minimum storage has to be ensured for analytics purposes. C_8 ensures the retention of local historical data for the smart meter.

Now that we present the objective function based on the regret definition (refer to Equation (13)) for energy optimization.

$$\max_{\pi^* \in \Pi} \min_{E[n]} R_{\text{energy}}(T) = \max_{\pi^* \in \Pi} \min_{E[n]} \sum_{t=1}^{T} (E^*_{\text{total}}[t] - E_{\text{total}}[t]) \tag{15}$$

Subject to:

$$C_1: \quad E_{\text{total}}[n] \geq \min\left(\frac{2^{256}}{P_{\text{compute}}}, \frac{2^{256}}{P_{\text{block}}}, \frac{2^{128}}{P_{\text{hash}}}, \frac{2^{256}}{P_{\text{sig}}}\right) \in [10^{28}, 10^{71}] \tag{15a}$$

$$C_2: \quad \lambda^i_P[n] \leq B_{\text{paid},i}[n], \quad \forall B_{\text{paid},i}[n] \in \{0,1\} \tag{15b}$$

$$C_3: \quad E_H[n] \geq N_{\text{ret}} \times P_{\text{block}} \tag{15c}$$

$$C_4: \quad E_{\text{edge}}[n] \leq E^{\max}_{\text{edge}}, \quad \forall E^{\max}_{\text{edge}} \in (0, \infty) \tag{15d}$$

$$C_5: \quad E_{\text{cloud}}[n] \geq E^{\min}_{\text{cloud}}, \forall E^{\min}_{\text{cloud}} \in (0, \infty) \tag{15e}$$

$$C_6: \quad \lambda^i_P[n] \cdot P_{\text{prune}} \leq \lambda^{\max}_P, \quad \forall \lambda^{\max}_P \in (0,1) \tag{15f}$$

$$C_7: \quad E_H[n] \geq E^{\min}_H, \quad \forall E^{\min}_H \in (0, \infty) \tag{15g}$$

$$C_8: \quad E_{\text{edge}}[n] + E_{\text{cloud}}[n] \geq E_H[n], \quad \forall E_H \in (0, \infty) \tag{15h}$$

C_1 ensures the minimum energy is allotted to maintain security. In particular, $\frac{2^{256}}{P_{\text{compute}}}$ ensures energy consumption for the first pre-image resistance. $\frac{2^{256}}{P_{\text{block}}}$ assures the energy consumption for the second pre-image resistance. $\frac{2^{128}}{P_{\text{hash}}}$ ensures the energy consumption for maintaining the collision resistance property. $\frac{2^{256}}{P_{\text{sig}}}$ warrant the energy consumption for maintaining the integrity. C_2 confirms only if pruning is allowed bill after bill settlement. C_3 corroborates that energy is allotted for retaining the relevant blocks as historical data. C_4 and C_5 meet the energy consumption of the edge server does not exceed its maximum capacity, and the energy capacity of the cloud is allotted to meet its minimum energy utilization. The pruning frequency should not consume much energy. This is met in C_6. While C_3 allocates the minimum energy consumption for maintaining the historical data, C_7 prevents that energy reduction from falling below the minimum level of energy for maintaining the historical data. The final constraint C_8 ensures the balance between the energy used by cloud and edge w.r.t. the energy maintained for retaining historical data.

SP 2:

We jointly optimize the pruning frequency by considering the energy consumption and storage consumption costs, and turning security constraints into the objective function.

 doi:10.20944/preprints202508.0413.v1

$$J_{\text{prune}}(\lambda_P[n]) = w_E \cdot (E^*_{\text{total}}[n] - E_{\text{total}}[n]) + w_S \cdot (S^*_{\text{total}}[n] - S_{\text{total}}[n]) + w_C \cdot C_{\text{security}}(\lambda_P[n]) \tag{16}$$

$$C_{\text{security}}(\lambda_P[n]) = \alpha_1 \cdot \frac{\lambda_P[n]}{S[n]} \times \frac{2^{256}}{B_{\text{hash}}} + \alpha_2 \cdot \frac{\lambda_P[n]}{S[n]} \times \frac{2^{256}}{B_{\text{block}}} + \alpha_3 \cdot \frac{\lambda_P[n]}{S[n]} \times \frac{2^{128}}{B_{\text{hash}}} + \alpha_4 \cdot \frac{\lambda_P[n]}{S[n]} \times \frac{2^{256}}{B_{\text{merkle}}} \tag{17}$$

$$\max_{\pi^*_i \in \Pi} \min_{\lambda_P[n]} R_{\text{joint}}(T) = \sum_{t=1}^{T} \left(J_{\text{prune}}(\lambda_P[t]) - J_{\text{prune}}(\lambda^*_P[t]) \right) \tag{18}$$

Subject to:

$$\left(E_{\text{hash}}[n] + E_{\text{block}}[n] + E_{\text{sig}}[n] + \underbrace{\frac{D[n] \cdot P_{\text{receive}}}{R_{\text{LoRa}}}}_{P_{receive}} \right) \leq E_{\max}[n], \quad \forall E_{\max}[n] \in (0, \infty) \tag{18a}$$

$$\lambda_P[n] \times \left(\underbrace{N_{\text{tx}} \times B_{\text{tx}}}_{B_{block}} + B_{\text{hash}} + B_{\text{sig}} + B_{\text{merkle}} \right) \leq S_{\min}[n], \forall S_{\min}[n] \in (0, \infty) \tag{18b}$$

$$S_{\text{total}}[n] \geq S_H[n] + N_{\text{ret}} \times B_{\text{block}} \in (0, \infty) \tag{18c}$$

In Equation (16), w_E, w_S and w_C are weights for energy consumption, storage and security at time n, respectively. E^*_{total} and S^*_{total} are energy and storage at hindsight. $C_{security}$ is the objective function capturing the security constraint. $C_{security}$ is the objective capturing the security constraints. Precisely, the first pre-image resistance term is covered in $\frac{\lambda_P[n]}{S[n]} \times \frac{2^{256}}{B_{\text{hash}}}$. The second pre-image resistance is absorbed by the term $\frac{\lambda_P[n]}{S[n]} \times \frac{2^{256}}{B_{\text{block}}}$. The collision resistance and integrity are modelled by the terms $\frac{\lambda_P[n]}{S[n]} \times \frac{2^{128}}{B_{\text{hash}}}$ and $\frac{\lambda_P[n]}{S[n]} \times \frac{2^{256}}{B_{\text{merkle}}}$, respectively. $\alpha_1, ..., \alpha_4$ are the weights for these security properties. From Equation (18), $J_{prune}(\lambda^*_P[t])$ is the best known pruning frequency at hindsight. Equation (18a) imposes a constraint for pruning energy, i.e. total pruning energy should not exceed the maximum energy. The pruning should not be overwhelmed, and it is met in the constraint (18b) by ensuring minimum blocks. At any instance, $S_{total}[n]$ should be more than the number of blocks retained as historical data (Equation (18c)).

We show that SP 2 is NP-hard as it includes SP 1 and its components as constraints. The problem of finding the optimal pruning frequency is an NP-hard problem. We prove that the Pruning Optimization Decision Problem (PODP) is NP-hard by reducing it to the Boolean Satisfiability Problem (SAT), which is NP-complete (PODP is polynomial-time reducible to SAT).

Decision Problem formulation: Given N edge servers, time slot T, pruning decisions $\lambda^i_P[n] \in \{0, 1\}$, costs $J^i_{\text{total}}[n]$, and threshold K, does there exist a pruning schedule $\{\lambda^i_P[n]\}_{i \in N, n \in [T]}$ such that:

$$\sum_{n=1}^{T} \sum_{i=1}^{N} J^i_{\text{total}}[n] \leq K, \tag{19}$$

subject to the constraints of SP 2. The proof is presented in Appendix A.

 doi:10.20944/preprints202508.0413.v1

4. Solution Approach Based on Deep Q Actor Critic Network (DQACN)

Figure 3 depicts the architecture of the deep Q actor-critic network in which the actor network is optimized by the gradient ascent method, and the critic network is optimized by the gradient descent method. Initially, the agent observes the state of the environment, and it utilizes the actor network. (policy function network) to take action based on transition probabilities. The action is applied to the environment, resulting in a new state and reward. Then, the critic network evaluates the new state value and current state value by computing the temporal difference (TD) error, followed by tuning the critic and actor networks. The problem of selecting a random server is solved by using the SoftMax probability function P^k, i.e., this function turns the energy cost of every server into the probability distribution, where servers can be selected by knowing their lowest probability score. Once the server is selected, it is allowed to solve optimization problems in order to find the cost-efficient storage, energy and pruning frequency. The server broadcast its solutions to the remaining servers. The recipient servers check whether the received solutions satisfy the constraints. If the constraints are satisfied, then other servers accept the current solution; otherwise, they reject the new solution. Once the pruning is executed locally, the edge server updates the retained data to the rented cloud server, facilitating future data analytics.

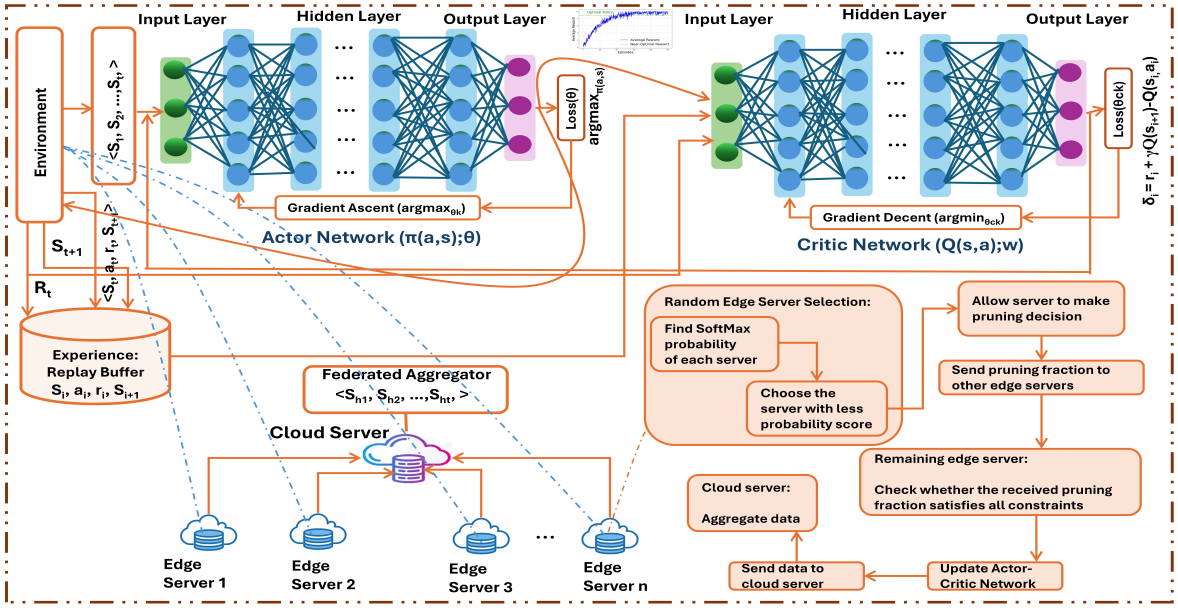

Figure 3. Deep Q Actor-Critic Network Architecture with Method.

Now, we present the state encoding variables, action space encoding variables, and reward and penalty to deploy the deep Q actor-critic network-based agent, representing the environment. The state $s^k[n]$ for server k at time n is defined as follows, and it is common for SP 1 and SP 2. Note that for the mana state, we consider the old value and the recent value, as backlog and finalized, respectively:

$$s^k[n] = \left(S^k_{\text{total}}[n], E^k_{\text{total}}[n], \lambda^k_T[n], m_k[n], M^k_{\text{backlog}}[n], M^k_{\text{finalized}}[n]\right) \tag{20}$$

The action space server k at time n is defined for SP1 and SP 2 as:

$$a^k[n] = S^k_{total}[n] \tag{21}$$

$$a^k[n] = E^k_{total}[n] \tag{21a}$$

$$a^k[n] = \lambda^k_P[n] \tag{21b}$$

 doi:10.20944/preprints202508.0413.v1

The reward $r^k[n]$ for the agent is defined as the negative of the total cost minus a penalty for violating constraints. For SP 1 and SP 2, the reward is defined as below.

$$r^k[n] = -J^k_{\text{total}}[n] - P^k[n] \tag{22}$$

$$r^k[n] = -S^k_{\text{total}}[n] - P^k[n] \tag{22a}$$

$$r^k[n] = -E^k_{\text{total}}[n] - P^k[n] \tag{22b}$$

Let the total cost be $J^k_{\text{total}}[n]$.

- For Algorithm 1 (SP 1):

$$E^k_{\text{total}}[n]; S^k_{\text{total}}[n] \tag{23}$$

- For Algorithm 2 (SP 2):

$$J^k_{\text{total}}[n] = w_E E^k_{\text{total}}[n] + w_S S^k_{\text{total}}[n] + w_C C^k_{\text{security}}[n] \tag{24}$$

The penalty term $P^k[n]$ accounts for violations of key constraints, e.g., security constraints. This ensures that the agent chooses the appropriate pruning frequency by respecting constraints and other objectives.

$$P^k[n] = \beta_1 \cdot \max(0, S_{\text{MIN}} - S^k_{\text{total}}[n]) + \beta_2 \cdot \max(0, S^k_{\text{edge}}[n] - S^{\max}_{\text{edge}}) + \beta_3 \cdot \max(0, \lambda^k_P[n] - \lambda^{\max}_P)... \tag{25}$$

We brief out the working of Algorithm 1 and Algorithm 2. We note that Algorithm 1 must be executed to preserve the security, before Algorithm 2 is executed. Algorithm 1 solves SP 1, thereby producing the solution for optimal storage and energy cost. Then, Algorithm 2 consumes the solution of Algorithm 1 and produces the optimal pruning frequency. We discuss the generic layout of Algorithms 1 and 2. At first, parameters are initialised, and the softmax probabilities are calculated for the edge servers. Then, storage and energy in hindsight are proposed, and the regret for those is computed. If the regrets for the storage and energy are non-negative, it is discarded. Otherwise, the solution is accepted. Then, the actor and critic networks are updated, and this process is iterated for the maximum number of iterations. In Algorithm 1, the parameters of the actor-critic network are initialised in all edge servers (Steps 3 to 6). Among the edge servers, a server (k^{th}) is selected (Steps 8 to 11) by the choice of SoftMax probability function (Step 10). Steps 13 and 14 present the environment to the agent (now, k^{th} server). Using the storage and energy equations, the agent proposes a new value for the storage and the energy by respecting the strict constraints (Steps 15 and 16). In Steps 17 and 18, the agent evaluates the regret. The agent finalises the storage and energy value in Step 19, which is broadcast to other servers. From Steps 21 to 28, other servers evaluate the solution, i.e. whether the new value for the storage and energy satisfies the constraints. The solution is accepted by the remaining servers if the constraints are satisfied (from Steps 29 to 33). In Step 35, the agent computes the advantage function as the computation of the temporal difference learning (the function is computed by targeting future reward as the dominant factor rather than the immediate reward). Additionally, the agent works out such a step by bootstrapping. Step 36 shows the gradient descent step of the critic network with the learning rate (α), whereas Step 37 shows the gradient ascent step of the actor network. At last, the agent outputs the storage and energy values. Note that here the value of the pruning frequency is still null, and it is solved in Algorithm 2. Until Step 11, the agent follows the steps of Algorithm 1. In Step 14, the agent turns the security constraints into part of the objective function. Step 18, additionally, includes the pruning frequency. In contrast to Algorithm 1, the agent updates its actor and critic network parameters after Step 29 by evaluating the advantage function. Then, the new solution is broadcast to the remaining edge servers. From Steps 30 to 38, the edge servers are updating their actor and critic parameters as per the received solution. Finally, every edge server gets the updated storage value, energy value and pruning value.

 doi:10.20944/preprints202508.0413.v1

Next, we analyse the convergence of the algorithms to depict whether the algorithms attain such solutions in sub-linear time, as the regret assures such efficiency.

Algorithm 1 Storage and Energy Optimization (SP 1) with Softmax-based Random Server Selection and Actor-Critic Evaluation

Input: N; $s^k[0]$; α, γ, $T_{\max}$; ψ_i, $m_i[0]$, δ_m, κ_m, S_i; k_h, k_s, $k_{\text{attach},i}$, $k_{\text{fpc},i}$, $k_{\text{mana},i}$, k_p, α_i; S_{tx}, S_{P}; $\lambda_T^k[n]$, $n_{\text{hash}}^k[n]$, $n_{\text{sig}}^k[n]$, $Q_i[n]$, $M_i[n]$, $D_i[n]$; $\lambda_P^k[n]$, $M_{\text{backlog}}^k[n]$, $M_{\text{finalized}}^k[n]$; $S^k[0]$, $E_{\text{total}}^k[0]$;
Initialization:
for $\forall\, k, k \in N$ **do**
  $rand \leftarrow \theta_l^k$ (actor parameter) and $rand \leftarrow \theta_l^{lk}$ (critic parameter);
  $n_l \leftarrow 0$ (learning step counter) and $T \leftarrow 0$ (time step);
  $S^k[0]$, $E_{\text{total}}^k[0]$, $\lambda_P^k[0]$;
end for
while $T < T_{\max}$ **do**
  for $\forall\, k, k \in N$ **do**
    $P^k = \frac{\exp(-E_{\text{total}}^k[n])}{\sum_{j \in N} \exp(-E_{\text{total}}^j[n])}$;
  end for
  $k_{\text{selected}}[k]$;
** Prioritize servers with lower energy usage for optimization **
  $s^{k_{\text{selected}}}[n]$: $S^{k_{\text{selected}}}[n]$, $E_{\text{total}}^{k_{\text{selected}}}[n]$, $\lambda_T^{k_{\text{selected}}}[n]$, $m_{k_{\text{selected}}}[n]$, $M_{\text{backlog}}^{k_{\text{selected}}}[n]$, $M_{\text{finalized}}^{k_{\text{selected}}}[n]$ (observed local states);
  $s^{k_{\text{selected}}}[n]$: $\lambda_P^{k_{\text{selected}}}[n] \sim \pi_l(s^{k_{\text{selected}}}[n] \mid \theta_l^{k_{\text{selected}}})$ (actor's proposal);
  k_{selected}: $S_{\text{proposed}}^{k_{\text{selected}}}[n+1]$ using Equation 9;
  k_{selected}: $E_{\text{proposed}}^{k_{\text{selected}}}[n+1]$ using Equation 8;
  k_{selected}: $R_{\text{storage}}^{k_{\text{selected}}}[n] \leftarrow S_{\text{proposed}}^{k_{\text{selected}}}[n+1] - S_{\text{total}}^{k_{\text{selected}}}[n]$;
  k_{selected}: $R_{\text{energy}}^{k_{\text{selected}}}[n] \leftarrow E_{\text{proposed}}^{k_{\text{selected}}}[n+1] - E_{\text{total}}^{k_{\text{selected}}}[n]$;
  k_{selected}: $S_{\text{proposed}}^{k_{\text{selected}}}[n+1]$, $E_{\text{proposed}}^{k_{\text{selected}}}[n+1]$, $\lambda_P^{k_{\text{selected}}}[n]$ (broadcast);
  solution_accepted $\leftarrow$ True;
  for $\forall\, k, k \in N \setminus \{k_{\text{selected}}\}$ **do**
    $R_{\text{storage}}^k[n] \leftarrow S_{\text{proposed}}^{k_{\text{selected}}}[n+1] - S_{\text{total}}^k[n]$;
    $R_{\text{energy}}^k[n] \leftarrow E_{\text{proposed}}^{k_{\text{selected}}}[n+1] - E_{\text{total}}^k[n]$;
    if $R_{\text{storage}}^k[n] < 0$ or $R_{\text{energy}}^k[n] < 0$ **then**
      solution_accepted $\leftarrow$ False;
      break;
    end if
  end for
** Ensure the solution benefits all servers before proceeding **
  if solution_accepted **then**
    for $\forall k, k \in N$ **do**
      $\lambda_P^k[n] \leftarrow \lambda_P^{k_{\text{selected}}}[n]$;
      $S^k[n+1] \leftarrow S_{\text{proposed}}^{k_{\text{selected}}}[n+1]$;
      $E_{\text{total}}^k[n+1] \leftarrow E_{\text{proposed}}^{k_{\text{selected}}}[n+1]$;
    end for
    k_{selected}: $A^{k_{\text{selected}}}[n] \leftarrow R_{\text{storage}}^{k_{\text{selected}}}[n] + R_{\text{energy}}^{k_{\text{selected}}}[n] + \gamma V^{\pi_l}(s^{k_{\text{selected}}}[n+1] \mid \theta_l^{lk_{\text{selected}}}) - V^{\pi_l}(s^{k_{\text{selected}}}[n] \mid \theta_l^{lk_{\text{selected}}})$;
    k_{selected}: $\theta_l^{lk_{\text{selected}}} \leftarrow \theta_l^{lk_{\text{selected}}} + \alpha \cdot A^{k_{\text{selected}}}[n] \cdot \nabla_{\theta_l^{lk_{\text{selected}}}} V^{\pi_l}(s^{k_{\text{selected}}}[n] \mid \theta_l^{lk_{\text{selected}}})$;
    k_{selected}: $\theta_l^{k_{\text{selected}}} \leftarrow \theta_l^{k_{\text{selected}}} + \alpha \cdot A^{k_{\text{selected}}}[n] \cdot \nabla_{\theta_l^{k_{\text{selected}}}} \log \pi_l(s^{k_{\text{selected}}}[n] \mid \theta_l^{k_{\text{selected}}})$;
** Apply the solution globally and improve the policy through learning **
  end if
  $T \leftarrow T + 1$;
end while
** Provide the optimised results for storage and energy across servers **
Output:
$\lambda_P^k[n]$, $S^k[n]$, $E_{\text{total}}^k[n]$, $\forall\, k, k \in N$.

 doi:10.20944/preprints202508.0413.v1

Algorithm 2 Joint Pruning Optimization (SP 2) with Softmax-based Random Server Selection and Actor-Critic Evaluation

Input: N; $s^k[0]$; α, γ, $T_{\max}$; ψ_i, $m_i[0]$, δ_m, κ_m, S_i; w_E, w_S, w_C; α_1, α_2, α_3, α_4, B_{hash}, B_{block}, B_{merkle}; k_h, k_s, $k_{\text{attach},i}$, $k_{\text{fpc},i}$, $k_{\text{mana},i}$, k_p, α_i; S_{tx}, S_{P}; $\lambda_T^k[n]$, $n_{\text{hash}}^k[n]$, $n_{\text{sig}}^k[n]$, $Q_i[n]$, $M_i[n]$, $D_i[n]$; $\lambda_P^k[n]$, $M_{\text{backlog}}^k[n]$, $M_{\text{finalized}}^k[n]$; $S^k[0]$, $E_{\text{total}}^k[0]$;
Initialization:
for $\forall\, k, k \in N$ **do**
  $rand \leftarrow \theta_l^k$ and $rand \leftarrow \theta_l^{lk}$;
  $n_l \leftarrow 0$ and $T \leftarrow 0$;
  $S^k[0]$, $E_{\text{total}}^k[0]$, $\lambda_P^k[0]$;
end for
while $T < T_{\max}$ **do**
  for $\forall\, k, k \in N$ **do**
    $P^k = \frac{\exp(-E_{\text{total}}^k[n])}{\sum_{j \in N} \exp(-E_{\text{total}}^j[n])}$;
  end for
  $k_{\text{selected}}[k]$;
** Prioritize servers with lower energy usage for optimization **
  $s^{k_{\text{selected}}}[n]$: $S^{k_{\text{selected}}}[n]$, $E_{\text{total}}^{k_{\text{selected}}}[n]$, $\lambda_T^{k_{\text{selected}}}[n]$, $m_{k_{\text{selected}}}[n]$, $M_{\text{backlog}}^{k_{\text{selected}}}[n]$, $M_{\text{finalized}}^{k_{\text{selected}}}[n]$;
  k_{selected}: $C_{\text{security}}(\lambda_P^{k_{\text{selected}}}[n])$ using Equation 15;
  k_{selected}: $S_{\text{proposed}}^{k_{\text{selected}}}[n+1]$ using Equation 9;
  k_{selected}: $E_{\text{proposed}}^{k_{\text{selected}}}[n+1]$ using Equation 8;
  k_{selected}: J_{proposed} using Equation 16;
  k_{selected}: $S_{\text{proposed}}^{k_{\text{selected}}}[n+1]$, $E_{\text{proposed}}^{k_{\text{selected}}}[n+1]$, $\lambda_P^{k_{\text{selected}}}[n]$ (broadcast);
  solution_accepted $\leftarrow$ True;
  for $\forall\, k, k \in N \backslash \{k_{\text{selected}}\}$ **do**
    J_{local}^k using Equation 16;
    if $J_{\text{local}}^k < J_{\text{proposed}}$ **then**
      solution_accepted $\leftarrow$ False;
      break;
    end if
  end for
** Ensure the proposed solution improves the joint cost for all servers **
  if solution_accepted **then**
    k_{selected}: $R_{\text{joint}}^{k_{\text{selected}}}[n] \leftarrow J_{\text{proposed}} - J_{\text{prune}}(\lambda_P^{k_{\text{selected}}}[n])$;
    k_{selected}: $A^{k_{\text{selected}}}[n] \leftarrow R_{\text{joint}}^{k_{\text{selected}}}[n] + \gamma V^{\pi_l}(s^{k_{\text{selected}}}[n+1] \mid \theta_l^{lk_{\text{selected}}}) - V^{\pi_l}(s^{k_{\text{selected}}}[n] \mid \theta_l^{lk_{\text{selected}}})$;
    for each edge server $k \in N$ **do**
      $\lambda_P^k[n] \leftarrow \lambda_P^{k_{\text{selected}}}[n]$;
      $S^k[n+1] \leftarrow S_{\text{proposed}}^{k_{\text{selected}}}[n+1]$;
      $E_{\text{total}}^k[n+1] \leftarrow E_{\text{proposed}}^{k_{\text{selected}}}[n+1]$;
      k: $R_{\text{joint}}^k[n] \leftarrow J_{\text{proposed}} - J_{\text{prune}}(\lambda_P^k[n])$;
      k: $A^k[n] \leftarrow R_{\text{joint}}^k[n] + \gamma V^{\pi_l}(s^k[n+1] \mid \theta_l^{lk}) - V^{\pi_l}(s^k[n] \mid \theta_l^{lk})$;
      $\theta_l^{lk} \leftarrow \theta_l^{lk} + \alpha \cdot A^k[n] \cdot \nabla_{\theta_l^{lk}} V^{\pi_l}(s^k[n] \mid \theta_l^{lk})$;
      $\theta_l^k \leftarrow \theta_l^k + \alpha \cdot A^k[n] \cdot \nabla_{\theta_l^k} \log \pi_l(s^k[n] \mid \theta_l^k)$;
    end for
    $n_l \leftarrow n_l + 1$;
** Apply the solution globally and refine the policy with joint cost optimization **
  end if
  $T \leftarrow T + 1$;
end while
** Provide the optimized results for storage and energy across servers **
Output:
  $\lambda_P^k[n]$, $S^k[n]$, $E_{\text{total}}^k[n]$, $\forall k, k \in N$.

4.1. Convergence Proof

While the deep Q actor-critic-based method is used for solving optimization problems in different contexts, we attempt to present the formulated regret functions indeed converge to the optimal solution in sub-linear time.

We present the assumption for proving theorems in Appendix A.

 doi:10.20944/preprints202508.0413.v1

Theorem 1. *Under the assumptions of finite state and action spaces, bounded rewards, Robbins-Monro learning rates, and an ergodic MDP, Algorithm 1 converges to a locally optimal policy π^* for SP1, minimizing energy and storage costs subject to constraints C_1 to C_8, with sublinear regret $R(T) = O(\sqrt{T})$.*

The proof of this theorem is presented in the appendix section (Appendix A).

Theorem 2. *Under the assumptions of finite spaces, bounded rewards, Robbins-Monro learning rates, an ergodic MDP, Lipschitz continuity, and bounded security costs, Algorithm 2 converges to a locally optimal policy π^* forming a Nash equilibrium for SP2, jointly optimizing pruning frequency, energy, and storage, with sublinear regret $R_{joint}(T) = O(\sqrt{T})$.*

The proof is presented in Appendix A.

Now, we show that the proposed algorithm for SP 1 truly satisfies security properties, namely, first pre-image resistance, second pre-image resistance, collision resistance and integrity against active adversaries.

4.2. Adversary Model

We consider the Dolev-Yao adversary model for the adversary $\mathcal{A}$. $\mathcal{A}$ can overhear communications between the smart meters and edge servers, taking the passive role. The attack surface for the adversary is considered at the edge servers. In the active role, $\mathcal{A}$ can substitute fake transactions in the storage held at every edge server. The adversary can violate the first preimage resistance, i.e., it can find the inputs for a targeted hash. By violating the second preimage resistance property, $\mathcal{A}$ can identify the other inputs that share the same hash value. In the event of breaking the collision resistance property, the adversary can find other inputs for a targeted input and hash. While counterfeating the integrity, $\mathcal{A}$ can substitute fake transactions to claim that they are legitimate.

4.3. Formal Security Analysis

We formally analyze how Algorithm 1 safeguards security against a classical active adversary using the universal composability framework, so we start with the relevant terminology of the framework. We claim the sense of security for the only properties claimed in the below propositions. The security is attained by satisfying the properties.

- **Real World:** Parties execute the protocol (Algorithm 1) in the presence of an adversary $\mathcal{A}$.
- **Ideal World:** Parties interact with an ideal functionality $\mathcal{F}$ that securely implements the desired behavior, with a simulator $\mathcal{S}$ emulating the adversary.
- **Security Definition:** A protocol μ UC-realizes $\mathcal{F}$ if, for any adversary $\mathcal{A}$, there exists a simulator $\mathcal{S}$ such that every environment $\mathcal{Z}$ cannot distinguish the real-world execution (with μ and $\mathcal{A}$) from the ideal-world execution (with $\mathcal{F}$ and $\mathcal{S}$).

Refer to Appendix A for the ideal functionality ($\mathcal{F}_{PRUNE}$).

- **Proposition 1**: Algorithm 1 ensures the first preimage resistance with non-negligible probability.
- **Proposition 2**: Algorithm 1 satisfies the second preimage resistance with non-negligible probability.
- **Proposition 3**: Collision resistance with non-negligible probability is achieved by Algorithm 1.
- **Proposition 4**: Integrity with non-negligible probability is ratified by Algorithm 1.

We prove the propositions by showing that Algorithm 1 UC-realizes $\mathcal{F}_{\text{PRUNE}}$, satisfying the respective security property (refer to Appendix A for the proofs).

5. Simulation and Analysis

In this section, to asses the effectiveness of the proposed algorithms, we ask the following sub-research questions (SRQs):

 doi:10.20944/preprints202508.0413.v1

- **SRQ 1:** How much efficiency is attained by DQACN (Algorithm 1 and Algorithm 2) w.r.t. the local snapshot mechanism in IOTA 2.0?
- **SRQ 2:** How far do DQACN (Algorithm 1 and Algorithm 2) satisfy the constraints strictly?
- **SRQ 3:** What is the cost-benefit of the proposed algorithms over local snapshot in IOTA 2.0?

To answer **SRQ 1**, we perform the Monte Carlo simulation as per the network setting of IOTA 2.0. Moreover, we simulate the network behavior of the IOTA 2.0 to capture the actual effectiveness of the proposed algorithms. We consider the local and global snapshots supported in IOTA 2.0 and IOTA 1.5, respectively, and the deep Q network and random elimination methods for comparison.

We consider a random process of transactions sent by the smart meters received by edge servers, with the IOTA Tangle network between smart meters and edge servers. We capture the input variables with the appropriate probability distribution, modeling the activity of the mana with the appropriate probability distribution.

Refer Appendix C for the probability distributions considered for the random process.

We use the values specified in Table A3 for different parameters. As we considered the temporal difference learning method in the actor-critic network, the network learns the optimal policy by bootstrapping (initial guess followed by iterative refinement), attaining convergence in the critic network. Figure 4 presents the simulation result. 1 in Figure 4 demonstrates the total cost, whereas Sub-figures 2 and 3 show the storage and energy costs, respectively. In the three sub-figures, Algorithm 2 (Alg 2) outperforms all schemes, typically consuming lower cost than the local snapshot. Sub-figure 1 depicts the total cost w.r.t. increasing edge servers, summing the energy cost and storage cost presented in sub-figures 2 and 3, respectively. When the edge server reaches 100, Algorithm (Alg 1) attains approximately 50% cost reduction in comparison with the local snapshot as it is driven by the static pruning rules. Alg 2 shows the cost reduction of 60% when compared with the local snapshot. This reduction trend is reflected in the ordinary deep-Q network as well; however, the actor-critic-based method outperforms all methods, namely local and global snapshots, random elimination and deep Q method.

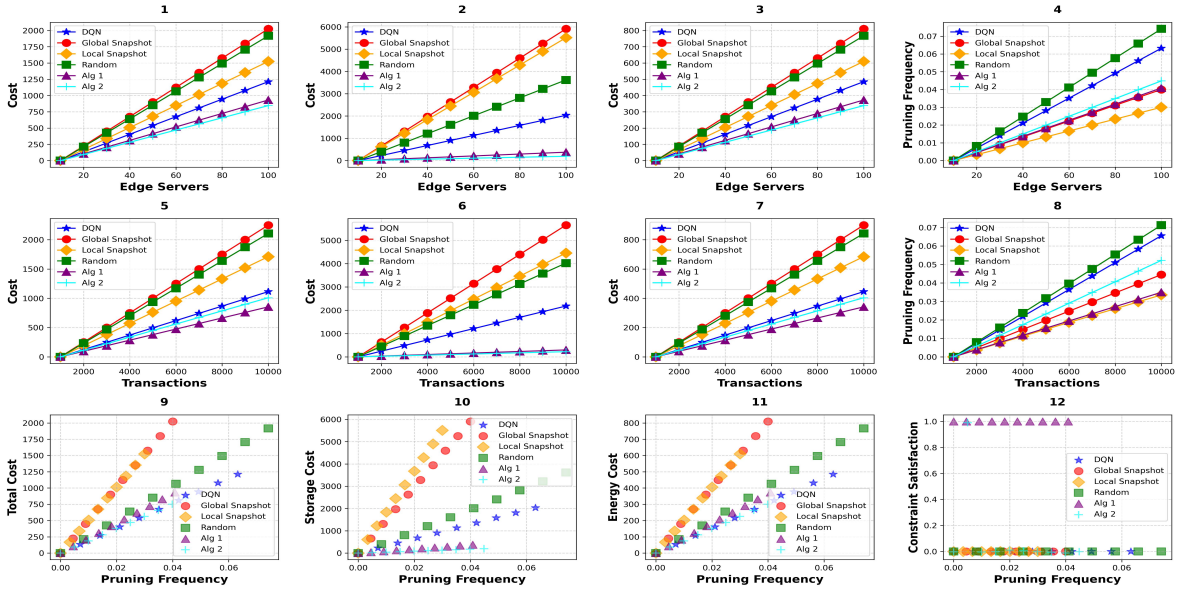

Figure 4. Simulation Result: Edge servers Vs Total Cost (1), Storage Cost (2), Energy Cost (3) and Pruning Frequency (4), Transactions Vs Total Cost (5), Storage Cost (6), Energy Cost (7) and Pruning Frequency (8), Pruning Frequency Vs Total Cost (9), Storage Cost (10), Energy Cost (11) and Constraints Satisfaction (12).

In Sub-figure 2, the proposed algorithms' storage cost is almost zero for proposed algorithms because of the aggressive pruning frequency. This trend shifts for increasing servers due to the time evolution model of the storage function and security constraint satisfaction. Sub-figure 3 demonstrates the energy cost of the proposed algorithms, showing lower consumption of energy. Notably, Algorithm

 doi:10.20944/preprints202508.0413.v1

2 consumes the least energy, whereas the global snapshot incurs the highest energy cost. Remaining methods contribute to the middle values between the proposed algorithms and the global snapshot.

Sub-figure 4 shows the amount of pruning frequency versus the number of servers where the highest values are attained by the random elimination since the pruning decisions are not cost aware; however, the least value comes from the local snapshot because of static rules, say prune only if the storage capacity exceeds 85% of the total storage. Proposed algorithms have higher pruning frequencies to optimize the storage and energy costs.

Sub-figure 5 shows the total cost in relation to transaction volume, with Algorithm 1 performing best by minimizing costs, followed by Algorithm 2 in energy consumption. Sub-figure 6 indicates that both Algorithms 1 and 2 achieve the same lowest storage cost. Sub-figure 7 highlights that Algorithms 1 and 2 outperform others in energy costs. Regarding pruning frequency, the local snapshot shows the lowest costs, but effective pruning decisions are necessary for optimal savings, leading to a higher pruning frequency in the proposed algorithms. Sub-figures 9, 10, and 11 demonstrate the quick convergence of pruning decisions for total cost, storage cost, and energy cost in the proposed algorithms compared to others.

SRQ 2 is answered in Sub-figure 12. For all the pruning decisions, Algorithm 1 strictly satisfies all constraints. Algorithm 2 occasionally satisfies the constraints, whereas the remaining methods, such as local and global snapshots, and the deep Q method, do not satisfy constraints, resulting in security violations and infeasibility in data analytics.

We answer **SRQ 3** as follows. We consider five cities, namely Amsterdam, Beijing, Cincinnati, Delhi and Milan for our study. The profile of the cities, including population, smart meter usage, number of edge servers used and renting rate, are depicted in Table A4 (Appendix C). We present the case for Amsterdam. For all cities, we consider the rental rate of the edge server as 5000 USD/year (Appendix C).

Table 2. Average Cost Savings Using Algorithms 1 and 2 for Smart Meters in Five Cities (USD, 2025-2029).

City	1-Year	2-Year	3-Year	4-Year	5-Year
Amsterdam	5,585,700	11,171,400	16,757,100	22,342,800	27,928,500
Beijing	136,652,240	273,304,480	409,956,720	546,608,960	683,261,200
Cincinnati	1,857,440	3,714,880	5,572,320	7,429,760	9,287,200
Delhi	204,978,360	409,956,720	614,935,080	819,913,440	1,024,891,800
Milan	8,693,120	17,386,240	26,079,360	34,772,480	43,465,600

6. Implementation Study

In this section, we implement the proposed algorithms to answer the following research question.
SRQ 4: What is the effectiveness of the proposed algorithms in the actual edge server environment?
SRQ 5: How do energy and other industries, and the IOTA community benefit from this research?

To answer **SRQ 4**, we consider the following environment for implementing and testing the proposed algorithms: Linux (operating system type), x86_64 (processor), Intel(R) Xeon(R) CPU @ 2.20GHz (processor brand and operating frequency), 12 GB (RAM) and 2 (CPU cores). We use Python 3 as the programming language for the implementation. In order to assess the proposed algorithms in a real-world setting, we implement the IOTA node with mana and the DQACN agent. We consider the test cases mentioned in Table A2. We test the algorithms to know what extent the storage, energy and pruning frequency are used to understand their baseline operation. For this case, we measure the memory consumed, energy consumed and pruning fraction held initially. To check the robustness of the algorithm towards growing transactions, we subject the second case, which is detailed in Table A2 (Appendix B). To check the security assurance of the proposed algorithms, we use the third test case. To understand the reliability of the algorithms, we use the final test case. Figure A1 (Appendix B) shows the result of the test cases. It is apparent that all of the test cases succeeded, making the proposed algorithms suitable for real-world deployment.

 doi:10.20944/preprints202508.0413.v1

Our developed algorithms can be found in the link [3], and the algorithms can be used for the real-time deployment, or it can serve as the guideline for extended implementation. This answers our **SRQ 5**. We made the developed algorithms as a standalone file ("dr_agent.py") so that it can be easily used for deployment. Our implementation can be used by the energy companies for the upgradation. Also, the IOTA Tangle foundation can use the developed algorithms as an alternative for the local snapshot.

7. Conclusions

This article identifies key limitations associated with the local snapshot of IOTA 2.0, particularly security violations and the infeasibility of data analytics. We proposed two algorithms based on the deep Q actor-critic network in reinforcement learning. Algorithm 1 optimizes storage cost and energy cost by strictly satisfying constraints (inclusive of security constraints) for a random edge sever, and the solutions are accepted by the remaining servers provided that constraints are satisfied, and Algorithm 2 optimizes pruning frequency without the restriction on the constraint satisfaction. Monte Carlo simulation confirms the practical efficacy of the proposed algorithms. Typically, a significant amount of improvement in efficiency is attained by the algorithms, satisfying security requirements. The study evaluated the cost-benefit of the proposed algorithms by considering five cities. This article studied the real-world implementation of the proposed algorithms and made the algorithms available, ready for real-world deployment. The solutions proposed in this article can benefit the IOTA Tangle foundation for upgrading its future snapshot mechanism. Further, this research can benefit energy companies renting CSPs by following the proposed algorithms over the conventional local snapshot in IOTA 2.0.

Acknowledgments: Funded in part by the European Union (EU), Grant Agreement no. 101095717 (SECURED). Views and opinions expressed are those of the authors and do not necessarily reflect those of the EU or the Health and Digital Executive Agency. Neither the EU nor the granting authority are responsible for them.

Appendix A

Appendix A.1. Proof for the NP-Hardness of SP 2

Proof. We reduce the problem, beginning from SAT. Given a SAT instance with v variables and m clauses: Set $N = v$, $T = 1$ in Equation (19). Map variable x_j to server $i = j$. Let $\lambda_P^j[1] = 1$ if $x_j =$ true, 0 if $x_j =$ false. For each clause c_k, the cost is defined as $C^k_{\text{clause}} = 1$ if c_k is not satisfied (all literals false), 0 otherwise. Set $J^i_{\text{total}}[1] = \sum_{k=1}^{m} C^k_{\text{clause}}$. Set $K = 0$. This satisfies constraints.

If SAT is satisfiable, there exists $\{\lambda_P^i[1]\}$ such that all clauses are satisfied ($C^k_{\text{clause}} = 0$), so $\sum_{k=1}^{m} C^k_{\text{clause}} = 0 \leq K$, and PODP returns "True". If SAT is not satisfiable, at least one clause is unsatisfied ($C^k_{\text{clause}} = 1$), so $\sum_{k=1}^{m} C^k_{\text{clause}} \geq 1 > K$, and PODP returns "False."

The time complexity associated with the construction takes $O(m \cdot v)$. Since SAT is NP-complete and reduces to PODP in polynomial time, PODP is NP-hard, which is NP-complete. Thus, the optimization problem (SP 2) is NP-hard. □

Appendix A.2. Assumptions

Finite Spaces (A1): To ensure bounded space for the agent, let the state spaces S and action spaces A are finite and discrete [19]. This makes the finite landscape for the agent to explore and make decisions.

Bounded Rewards (A2): Reward $r(s,a) = -E_{\text{total}}[n] + P[n]\text{r} - S_{\text{total}}[n] + P[n]$ satisfies $|r(s,a)| \leq R_{\max}$ [20]. This ensures the reward is finite and convergent rather than being divergent.

Learning Rates (A3): The learning rate α_t satisfies $\sum_t \alpha_t = \infty$, $\sum_t \alpha_t^2 < \infty$, and $0 < \gamma < 1$ a.k.a. Robbins-Monro learning rates [21]. The divergent condition indicates that the learning step is not

[3] https://github.com/Scriptsuser/IOTA_2_0

 doi:10.20944/preprints202508.0413.v1

stuck with local optimality, whereas the convergent condition implies that the learning step indeed converges to a stationary point.

Ergodic Markov Decision Process (MDP) (A4): The MDP has a stationary distribution $\mu^\pi(s)$ [20]. By the property of the ergodicity, the time average equals the ensemble average, that is, the statistical property observed in different states (pruning decision) is equal to the time average of the system.

Lipschitz Continuity (A5): $\nabla_\theta J(\theta)$ is Lipschitz continuous with constant L [22]. In other words, the gradient of the function is continuous and it is bounded by a constant.

Convex Weights (A6): $w_E, w_S, w_C > 0$, $w_E + w_S + w_C = 1$. This condition underscores the importance of the objective function, meaning that every objective function in a joint optimization problem contributes to certain importance.

Bounded Security (A7): C_{security} is continuous and bounded. This assumption ensures that security constraints converted to the objective function have a finite landscape while exploring for solutions.

Appendix A.3. Proof of Theorem 1:

Proof. We prove Theorem 1 by establishing the convergence of the Actor-Critic method in Algorithm 1, leveraging the Policy Gradient Theorem [23] and TD(0) convergence [24].

We begin with A1 and A2.

The critic updates the value function using TD(0) (initial guess):

$$\delta^k[n] = r^k[n] + \gamma V^\pi(s^k[n+1]|\theta^{c,k}) - V^\pi(s^k[n]|\theta^{c,k}), \tag{A1}$$

$$\theta^{c,k} \leftarrow \theta^{c,k} + \alpha_n \delta^k[n] \nabla_{\theta^{c,k}} V^\pi(s^k[n]|\theta^{c,k}). \tag{A2}$$

The TD error's expectation is:

$$\mathbb{E}[\delta^k[n]|s^k[n]] = Q^\pi(s^k[n], a^k[n]) - V^\pi(s^k[n]). \tag{A3}$$

By [24], TD(0) is a contraction mapping with factor γ:

$$\|V^\pi_{n+1} - V^\pi\|_\infty \leq \gamma \|V^\pi_n - V^\pi\|_\infty. \tag{A4}$$

Given the Robbins-Monro conditions (A3), $\theta^{c,k}$ converges almost surely to parameters yielding $V^\pi(s)$ [20].

The actor updates:

$$\theta^k \leftarrow \theta^k + \alpha_n A^k[n] \nabla_{\theta^k} \log \pi(a^k[n]|s^k[n], \theta^k), \tag{A5}$$

$$A^k[n] = r^k[n] + \gamma V^\pi(s^k[n+1]) - V^\pi(s^k[n]). \tag{A6}$$

The policy gradient theorem states:

$$\nabla_\theta J(\theta) = \mathbb{E}_{\mu^\pi}[\nabla_\theta \log \pi(a|s,\theta) Q^\pi(s,a)]. \tag{A7}$$

Since $A^k[n] \to Q^\pi(s,a) - V^\pi(s)$ as the critic converges, the gradient is unbiased.

Server selection uses:

$$P^k = \frac{\exp(-E^k_{\text{total}}[n])}{\sum_{j \in N} \exp(-E^j_{\text{total}}[n])}. \tag{A8}$$

By Jensen's inequality for the concave function $\exp(-x)$:

$$\mathbb{E}[\exp(-E^k_{\text{total}}[n])] \geq \exp(-\mathbb{E}[E^k_{\text{total}}[n]]). \tag{A9}$$

As $E^k_{\text{total}}[n]$ stabilizes, P^k favors low-cost servers, ensuring exploration [19].

 doi:10.20944/preprints202508.0413.v1

Regret is given by:

$$R(T) = \sum_{t=1}^{T} (C^*[t] - C[t]), \quad C[t] = E_{\text{total}}[t] \text{ or } S_{\text{total}}[t]. \tag{A10}$$

The performance difference lemma [25] gives:

$$J(\theta_{t+1}) - J(\theta_t) \geq \frac{1}{1-\gamma} \mathbb{E}_{\mu^{\pi_t}}[A^{\pi_{t+1}}(s,a)]. \tag{A11}$$

With $A^k[n] > 0$ for improvements and constraints enforced, $J(\theta_t)$ increases. By Lipschitz continuity (A5), $R(T) = O(\sqrt{T})$.

As $n_l \to \infty$, $\nabla_\theta J(\theta) \to 0$, and the ergodic MDP (A4) ensures convergence to a local optimum π^* satisfying C_1 to C_8.

Thus, Algorithm 1 converges to π^* with sublinear regret. □

Appendix A.4. Proof of Theorem 2:

Proof. We prove Theorem 2 using cooperative Actor-Critic convergence [26] and Nash equilibrium properties [27].

The reward is (A7 holds in addition to A2):

$$r^k[n] = -J^k_{\text{total}}[n], \quad J^k_{\text{total}}[n] = w_E E^k_{\text{total}}[n] + w_S S^k_{\text{total}}[n] + w_C C^k_{\text{security}}[n], \tag{A12}$$

where $C^k_{\text{security}}[n] = \sum_{i=1}^{4} \alpha_i \frac{\lambda^k_P[n]}{S^k[n]} \frac{2^{k_i}}{B_i}$ is bounded ($\lambda^k_P[n] \in [0,1]$, $S^k[n] \geq S_{\min}$) (A7).

As in SP1:

$$\theta^{c,k} \leftarrow \theta^{c,k} + \alpha_n \delta^k[n] \nabla_{\theta^{c,k}} V^\pi(s^k[n]|\theta^{c,k}), \tag{A13}$$

$$\delta^k[n] = r^k[n] + \gamma V^\pi(s^k[n+1]) - V^\pi(s^k[n]). \tag{A14}$$

TD(0) ensures $V^\pi(s^k[n]|\theta^{c,k}) \to V^\pi(s)$ [24].

The actor updates:

$$\theta^k \leftarrow \theta^k + \alpha_n A^k[n] \nabla_{\theta^k} \log \pi(a^k[n]|s^k[n], \theta^k), \tag{A15}$$

with $A^k[n] = r^k[n] + \gamma V^\pi(s^k[n+1]) - V^\pi(s^k[n])$, and:

$$\nabla_\theta J(\theta) = \mathbb{E}_{\mu^\pi}[\nabla_\theta \log \pi(a|s,\theta) Q^\pi(s,a)]. \tag{A16}$$

A solution is accepted if:

$$J^k_{\text{total}}[n] < J_{\text{proposed}}, \quad \forall k \in N \setminus \{k_{\text{selected}}\}. \tag{A17}$$

This ensures cooperative improvement, approximating a Nash equilibrium [27].

For the convex combination:

$$\mathbb{E}[J^k_{\text{total}}] \leq w_E \mathbb{E}[E^k_{\text{total}}] + w_S \mathbb{E}[S^k_{\text{total}}] + w_C \mathbb{E}[C^k_{\text{security}}]. \tag{A18}$$

This bounds the expected cost, stabilizing optimization.

Regret is:

$$R_{\text{joint}}(T) = \sum_{t=1}^{T} (w_E E^*_{\text{total}}[t] + w_S S^*_{\text{total}}[t] - J_{\text{total}}[t]). \tag{A19}$$

Cooperative updates ensure:

$$J(\theta_{t+1}) \leq J(\theta_t) - \epsilon \text{ or } J(\theta_{t+1}) = J(\theta_t). \tag{A20}$$

 doi:10.20944/preprints202508.0413.v1

By Lipschitz continuity:

$$\|J(\theta_{t+1}) - J(\theta_t)\| \leq L\alpha_t \|A^k[n]\nabla_\theta \log \pi\|. \tag{A21}$$

As $\alpha_t \to 0$, $\nabla_\theta J(\theta) \to 0$ (A3), and $R_{\text{joint}}(T) = O(\sqrt{T})$.

At convergence, no server can improve J^k_{total} unilaterally, forming a Nash equilibrium [26].

Thus, Algorithm 2 converges to π^* with sublinear regret. □

Appendix A.5. Ideal Functionality $\mathcal{F}_{PRUNE}$:

We define an ideal functionality $\mathcal{F}_{\text{PRUNE}}$ that captures the secure pruning mechanism for IOTA 2.0, ensuring the cryptographic properties claimed in Propositions 1–4. The functionality interacts with edge servers $(P_1, \ldots, P_N)$, smart meters $(SM_1, \ldots, SM_M)$, and a CSP, handling transactions and pruning while enforcing security.

1. **Initialization**:
   - Stores a ledger of transactions $\mathcal{L} = \{(tx_i, h_i, sig_i, merkle_i)\}$, where $h_i = H(tx_i)$, sig_i is the Winternitz One-Time Signature (WOTS), and $merkle_i$ is the Merkle root.
   - Maintains storage $S_i[n]$ and energy $E_i[n]$ for each server P_i at time n.
   - Security parameters: $k = 256$ (hash length), ensuring 2^{256} preimage resistance, 2^{128} collision resistance (birthday paradox), and 2^{256} integrity for WOTS.
2. **Transaction Submission**:
   - On input (tx, send) from SM_j:
     - Compute $h = H(tx)$, $sig = \text{WOTS.Sign}(tx)$, and update Merkle root.
     - If H is collision-resistant ($\Pr[\text{find } tx' \neq tx : H(tx') = h] \leq 2^{-128}$) and sig verifies, add $(tx, h, sig, merkle)$ to $\mathcal{L}$.
     - Send (tx, h, sig) to P_i for storage, updating $S_i[n] \leftarrow S_i[n-1] + S_{\text{tx}}$.
3. **Pruning Request**:
   - On input $(\text{prune}, \lambda^i_P[n])$ from P_i:
     - Check constraints:
       - $S_i[n] \geq S^{\min}_H$ (historical data retention).
       - $\lambda^i_P[n] \cdot S_P \leq S_{\max}$ (pruning limit).
       - Bills settled ($B_{\text{paid},i}[n] = 1$).
       - Security: Ensure pruning retains 2^{256} first/second preimage resistance, 2^{128} collision resistance, and 2^{256} integrity (via sufficient transactions for hash/Merkle checks).
     - If valid, remove fraction $\lambda^i_P[n]$ of transactions, updating $S_i[n] \leftarrow S_i[n-1] - \lambda^i_P[n] \cdot S_P$.
     - Output $(S_i[n], \text{success})$.
4. **Adversary Interaction**:
   - $\mathcal{A}$ can submit fake transactions (tx', h', sig').
   - $\mathcal{F}_{\text{PRUNE}}$ rejects if:
     - $h' \neq H(tx')$ (first preimage resistance).
     - $tx' \neq tx$ but $H(tx') = H(tx)$ (second preimage/collision resistance).
     - sig' fails WOTS verification (integrity).
   - $\mathcal{A}$ can query $\mathcal{L}$, but cannot modify valid entries.
5. **Output**:
   - Return updated $S_i[n]$, $E_i[n]$, and $\mathcal{L}$ to P_i, SM_j, and CSP, ensuring no security violations.

This functionality ensures pruning preserves cryptographic properties by rejecting adversarial attempts to violate preimage resistance, collision resistance, or integrity.

 doi:10.20944/preprints202508.0413.v1

Appendix A.5.1. Proof of Proposition 1

Algorithm 1 ensures that an adversary $\mathcal{A}$ cannot find an input tx' such that $H(tx') = h$ for a given hash $h = H(tx)$ with non-negligible probability.

- **Ideal World**: In $\mathcal{F}_{\text{PRUNE}}$, $\mathcal{A}$ submits (tx', h'). The functionality checks $h' = H(tx')$, rejecting if false. Since H is modeled as a random oracle with 2^{256} output space, $\Pr[H(tx') = h] \leq 2^{-256}$, ensuring first preimage resistance.
- **Real World**: Algorithm 1 prunes transactions while ensuring $S_i[n] \geq S_H^{\min}$, retaining enough transactions to maintain hash security (constraint C_1: $\frac{2^{256}}{B_{\text{hash}}}$). The hash function H (e.g., SHA-256) is preimage-resistant, and pruning decisions enforce sufficient storage to verify $h = H(tx)$.
- **Simulator** $\mathcal{S}$: $\mathcal{S}$ simulates $\mathcal{A}$'s view by generating transactions (tx, h, sig) using a random oracle H. If $\mathcal{A}$ submits tx', $\mathcal{S}$ computes $h' = H(tx')$ and forwards to $\mathcal{F}_{\text{PRUNE}}$, which rejects unless $h' = h$. Since $\mathcal{S}$ uses the same H, the probability of finding tx' such that $H(tx') = h$ remains 2^{-256}.
- **Indistinguishability**: The environment $\mathcal{Z}$ sees identical outputs in both worlds: valid transactions are stored, and invalid ones rejected. The constraint $S_i[n] \geq S_H^{\min}$ ensures enough data to enforce 2^{256} preimage resistance, matching $\mathcal{F}_{\text{PRUNE}}$'s rejection of fake inputs.
- Algorithm 1 UC-realizes $\mathcal{F}_{\text{PRUNE}}$ for first preimage resistance, as $\mathcal{A}$'s success probability is negligible ($\leq 2^{-256}$).

Appendix A.5.2. Proof of Proposition 2

Algorithm 1 sanctions $\mathcal{A}$ cannot find $tx' \neq tx$ such that $H(tx') = H(tx)$ with non-negligible probability.

- **Ideal World**: $\mathcal{F}_{\text{PRUNE}}$ rejects (tx', h') if $h' = H(tx)$ but $tx' \neq tx$. For a random oracle H, $\Pr[H(tx') = H(tx) \mid tx' \neq tx] \leq 2^{-256}$, ensuring second preimage resistance.
- **Real World**: Algorithm 1's constraint C_1 ($\frac{2^{256}}{B_{\text{hash}}}$) ensures pruning retains transactions to verify $H(tx)$. The hash function's second preimage resistance guarantees that finding $tx' \neq tx$ with $H(tx') = H(tx)$ is computationally infeasible.
- **Simulator** $\mathcal{S}$: $\mathcal{S}$ simulates $\mathcal{A}$'s view, computing $h = H(tx)$ for valid transactions. If $\mathcal{A}$ submits $tx' \neq tx$, $\mathcal{S}$ computes $h' = H(tx')$ and forwards to $\mathcal{F}_{\text{PRUNE}}$, which rejects unless $h' = h$, with probability 2^{-256}.
- **Indistinguishability**: $\mathcal{Z}$ cannot distinguish real and ideal worlds, as both reject $tx' \neq tx$ with $H(tx') = H(tx)$. Algorithm 1's storage constraints ensure verification data persists, aligning with $\mathcal{F}_{\text{PRUNE}}$.
- Algorithm 1 UC-realizes $\mathcal{F}_{\text{PRUNE}}$ for second preimage resistance, with $\mathcal{A}$'s success probability $\leq 2^{-256}$.

Appendix A.5.3. Proof of Proposition 3

Algorithm 1 satisfies $\mathcal{A}$ cannot find $tx_1 \neq tx_2$ such that $H(tx_1) = H(tx_2)$ with non-negligible probability.

- **Ideal World**: $\mathcal{F}_{\text{PRUNE}}$ rejects pairs (tx_1, tx_2) if $H(tx_1) = H(tx_2)$ and $tx_1 \neq tx_2$. For a random oracle, the birthday paradox bounds $\Pr[H(tx_1) = H(tx_2)] \leq 2^{-128}$, ensuring collision resistance.
- **Real World**: Algorithm 1 enforces C_1 ($\frac{2^{128}}{B_{\text{hash}}}$), retaining transactions to detect collisions. The hash function's collision resistance ensures finding tx_1, tx_2 with $H(tx_1) = H(tx_2)$ requires $\approx 2^{128}$ operations.
- **Simulator** $\mathcal{S}$: $\mathcal{S}$ generates $h_i = H(tx_i)$ for valid transactions. If $\mathcal{A}$ submits (tx_1, tx_2), $\mathcal{S}$ computes $h_1 = H(tx_1)$, $h_2 = H(tx_2)$, forwarding to $\mathcal{F}_{\text{PRUNE}}$, which rejects if $h_1 = h_2$. The probability of a collision is 2^{-128}.
- **Indistinguishability**: Both worlds reject collisions identically, with Algorithm 1's constraints ensuring sufficient storage for hash verification, matching $\mathcal{F}_{\text{PRUNE}}$.

 doi:10.20944/preprints202508.0413.v1

- Algorithm 1 UC-realizes $\mathcal{F}_{\text{PRUNE}}$ for collision resistance, with $\mathcal{A}$'s success probability $\leq 2^{-128}$.

Appendix A.5.4. Proof of Proposition 4

Algorithm 1 ratifies that $\mathcal{A}$ cannot substitute fake transactions to violate integrity with non-negligible probability.

- **Ideal World**: $\mathcal{F}_{\text{PRUNE}}$ verifies transactions via WOTS signatures (sig_i) and Merkle roots. If $\mathcal{A}$ submits (tx', sig'), it's rejected unless sig' verifies for tx' and matches $\mathcal{L}$'s Merkle root. WOTS ensures $\Pr[\text{forgery}] \leq 2^{-256}$.
- **Real World**: Algorithm 1 uses WOTS signatures and Merkle roots, with constraint C_1 ($\frac{2^{256}}{B_{\text{sig}}}$) ensuring enough transactions remain to verify integrity. Pruning only occurs for settled bills (C_2), preserving valid data.
- **Simulator** $\mathcal{S}$: $\mathcal{S}$ simulates $\mathcal{A}$'s view, generating valid (tx, sig, merkle). If $\mathcal{A}$ submits (tx', sig'), $\mathcal{S}$ checks sig' via WOTS and forwards to $\mathcal{F}_{\text{PRUNE}}$, which rejects invalid signatures. Forgery probability is 2^{-256}.
- **Indistinguishability**: $\mathcal{Z}$ sees identical ledger updates, with both worlds rejecting fake transactions. Algorithm 1's constraints ensure that signature verification data persists.
- Algorithm 1 UC-realizes $\mathcal{F}_{\text{PRUNE}}$ for integrity, with $\mathcal{A}$'s success probability $\leq 2^{-256}$.

Appendix B

```
platform linux -- Python 3.11.11, pytest-7.4.0, pluggy-1.5.0 -- /usr/bin/python3
cachedir: .pytest_cache
rootdir: /content
plugins: typeguard-4.4.2, anyio-4.9.0, langsmith-0.3.23
collecting ... collected 4 items

test_pruning.py::TestPruning::test_baseline_operation PASSED                [ 25%]
test_pruning.py::TestPruning::test_high_transaction_load PASSED             [ 50%]
test_pruning.py::TestPruning::test_security_constraint PASSED               [ 75%]
test_pruning.py::TestPruning::test_long_term_stability PASSED               [100%]
```

Figure A1. Test Cases Result.

Table A1. Comparison between IOTA 2.0 and Traditional Blockchain Consensus Mechanisms.

Metrics	IOTA 2.0 (Mana & FPC)	Blockchain (PoW)	Blockchain (PoS)
Consensus Mechanism	Mana (reputation-based) + FPC (voting-based)	PoW (hash-based)	PoS (stake-based)
Energy Consumption	Low	High	Low
Storage	Low	High	High
Decentralization	High	High	High
Scalability	High	Low	Moderate
Sybil Protection	Mana	Computational power	Stake
Pruning	Local snapshot	Possible	Possible

 doi:10.20944/preprints202508.0413.v1

Table A2. Test Cases for IOTA 2.0 Pruning with DQACN.

Test Case	Description	Parameters	Test Result
Baseline Operation	Run a single edge server with simulated IOTA transactions for 1 hour.	- Storage usage (MB) - Energy usage (CPU %) - Pruning frequency (λ_P)	Pass
High Transaction Load	Simulate 10,000 transactions/hour on 3 edge servers.	- Storage reduction (%) - Energy reduction (%) - Mana value	Pass
Security Constraint	Inject 1000 fake transactions; monitor pruning impact.	- Storage for security (MB) - Collision resistance (2^{128}) - Preimage resistance (1 & 2) (2^{256})	Pass
Long-Term Stability	Run 5 edge servers for 24 hours with variable transaction rates (1000–5000/hour).	- Storage usage - Energy usage - Agent's regret (Is it sublinear?)	Pass

Appendix C

Table A3. Simulation Values.

Notation	Value
N	10-100
$\lambda_T^i[n]$	1000-10000
$\lambda_P^i[n]$	0 to 1
$S_{\text{edge}}^{\max}$	1000
$E_{\text{total}}[n]$	5000 (base)
$S[n]$	$5000 \times 0.6 = 3000$ (base)
$S_{\text{storage}}^i[n]$	$N \times 0.1$
$S[n]_{\text{freed}}^i$	0 to 1000
γ	0.9
α	0.1
w_E	0.4
w_S	0.4
w_C	0.2
κ_m	0.05
$m_j[n]$	Uniform(0.01, 0.1)
η	0.0001

The pruning decision should be proportional to the mana, i.e. servers receiving more transactions have to prune frequently. The base pruning rate is modeled by the beta distribution in Equation (A22) with the gamma (Γ) distribution:

$$\text{Beta}(x;\alpha_p,\beta_p) = \frac{x^{\alpha_p-1}(1-x)^{\beta_p-1}}{\frac{\Gamma(\alpha_p)\Gamma(\beta_p)}{\Gamma(\alpha_p+\beta_p)}}, \quad 0 \le x \le 1 \tag{A22}$$

$$\lambda_P^i[n] = \text{Beta}(2,5) \cdot \frac{M_{\text{finalized}}^i[n]}{M_{\text{backlog}}^i[n]} \cdot \frac{m_i[n]}{\max(m_j[n])} \tag{A23}$$

$\text{Beta}(2,5)$ creates the skewed value, leading to the lower pruning rate. The pruning frequency is scaled further with the ratio of finalized messages by FPC and unpruned messages, and normalizing the

 doi:10.20944/preprints202508.0413.v1

factor, enabling active servers to prune more. The mana decay in the energy model is as follows. The energy consumption for mana management by server i at time n, accounting for both transaction issuance and decay updates, is given by:

$$E_{\text{mana},i}[n] = k_{\text{mana},i} \cdot (M_i[n] + D_i[n]) \tag{A24}$$

$D_i[n]$ is the number of decay updates, modeled as $D_i[n] \sim \text{Poisson}(d_i)$, with $d_i \sim \text{Uniform}(1, 3)$. The Poisson distribution for $D_i[n]$ and $M_i[n]$ is defined by the probability mass function:

$$\text{Poisson}(k; \mu) = \frac{\mu^k e^{-\mu}}{k!}, \quad k = 0, 1, 2, \ldots \tag{A25}$$

The Uniform distribution for d_i is defined by the probability density function:

$$\text{Uniform}(x; a, b) = \begin{cases} \frac{1}{b-a}, & \text{if } a \leq x \leq b \\ 0, & \text{otherwise} \end{cases}, \quad a = 1, \quad b = 3 \tag{A26}$$

To handle the network congestion, the transaction rate ($\mu_i[n]$) of the i^{th} server is controlled according to its mana value. This is given by below equation.

$$\mu_i[n] = \mu_i \cdot \frac{m_i[n]}{\max(m_j[n])} \cdot \frac{1}{1 + \eta \cdot \sum_{j=1}^{N} \lambda_T^j[n-1]} \tag{A27}$$

μ_i is the base transaction rate for server i, $\max(m_j[n])$ is the maximum mana across all servers at time n and $\sum_{j=1}^{N} \lambda_T^j[n-1]$ represents the total network transaction rate at the previous time step $n-1$. The FPC query rate for server i at time n, scaled by conflict rate, is given by:

$$Q_i[n] \sim \text{Poisson}\left(q_i[n] \cdot \frac{m_i[n]}{\max(m_j[n])} \right) \tag{A28}$$

q_{base} is the baseline FPC query rate, q_{conflict} is the additional query rate per conflict, $C_{\text{rate}}[n]$ is the conflict rate at time n.

$$q_i[n] = q_{\text{base}} + q_{\text{conflict}} \cdot C_{\text{rate}}[n] \tag{A29}$$

Amsterdam Illustration:

We present the example calculation for Amsterdam. For the population of 900,000 ([28]), a 70% smart meter rate ([29]) yields 630,000 smart meters (900,000 × 0.7). Each 50,000 meters requires one edge server ([30]), so there are 630,000 / 50,000 = 12.6 (13) servers. With a rental rate of $5,000 USD per year for an edge server, the baseline rental cost is 13 × $5,000 = $65,000. Without optimization, operational costs become $5,040,000 annually (630,000 × $8/meter, IOTA 2.0), offset savings of $3,780,000 (630,000 × $6/meter, billing efficiency), resulting in a net baseline savings of $3,715,000 ($3,780,000 - $65,000). Algorithm 1, optimizing storage and energy individually, applies a 35% energy reduction and 30% storage reduction (applying lower bound to the solution), increasing savings to $5,475,450 ($3,715,000 × 1.474, reflecting combined energy and storage benefits), a 47.4% gain over the baseline (($5,475,450 - $3,715,000) / $3,715,000 × 100 = 47.4%). Algorithm 2, jointly optimizing pruning frequency, energy, and storage, applies a 40% energy reduction and 35% storage reduction, yielding $5,695,950 ($3,715,000 × 1.534), a 53.4% increase (($5,695,950 - $3,715,000) / $3,715,000 × 100 = 53.4%). Averaging these, the 1-year savings become ($5,475,450 + $5,695,950) / 2 = $5,585,700, reflecting a 50.4% improvement (($5,585,700 - $3,715,000) / $3,715,000 × 100 = 50.4%). This analysis highlights the significant cost efficiency of the cooperative deep Q actor-critic approach, scalable to a 5-year average of $27,928,500 ($5,585,700 × 5), offering substantial benefits for energy providers in smart meter deployments. For the remaining cities, the cost saving for 1 year to 5 years is highlighted in Table 2. From our analysis, it

 doi:10.20944/preprints202508.0413.v1

is apparent that energy providers can obtain enormous cost benefits. The cost-benefit is enormously high for highly populated cities like Delhi and Beijing.

Table A4. Smart Meter Deployment Across Five Cities (as on 2025).

City	Population	Smart Meters [29]	Edge Servers [30]	Renting Rate
Amsterdam	900,000 [28]	630,000	13	5,000
Beijing	22,000,000 [31]	15,400,000	308	5,000
Cincinnati	300,000 [32]	210,000	5	5,000
Delhi	33,000,000 [33]	23,100,000	462	5,000
Milan	1,400,000 [34]	980,000	20	5,000

References

1. Wan, S.; Lin, H.; Gan, W.; Chen, J.; Yu, P.S. Web3: The Next Internet Revolution. *IEEE Internet of Things Journal* **2024**, *11*, 34811–34825. https://doi.org/10.1109/JIOT.2024.3432116.
2. Liu, Z.; Xiang, Y.; Shi, J.; Gao, P.; Wang, H.; Xiao, X.; Wen, B.; Li, Q.; Hu, Y.C. Make Web3.0 Connected. *IEEE Transactions on Dependable and Secure Computing* **2022**, *19*, 2965–2981. https://doi.org/10.1109/TDSC.2021.3079315.
3. Bu, G.; Gürcan, Ö.; Potop-Butucaru, M. G-IOTA: Fair and confidence aware tangle. In Proceedings of the IEEE INFOCOM 2019 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), 2019, pp. 644–649. https://doi.org/10.1109/INFCOMW.2019.8845163.
4. Guo, F.; Xiao, X.; Hecker, A.; Dustdar, S. Characterizing IOTA Tangle with Empirical Data. In Proceedings of the GLOBECOM 2020 - 2020 IEEE Global Communications Conference, 2020, pp. 1–6. https://doi.org/10.1109/GLOBECOM42002.2020.9322220.
5. Koštál, K.; Bahar, M.N.; Numyalai, S.; Ries, M. Beyond Integration: Advancing Electricity Metering with Secure and Transparent Blockchain-IoT Solutions. In Proceedings of the 2024 47th International Conference on Telecommunications and Signal Processing (TSP), 2024, pp. 324–331. https://doi.org/10.1109/TSP63128.2024.10605954.
6. Bhavana, G.; Anand, R.; Ramprabhakar, J.; Meena, V.; Jadoun, V.K.; Benedetto, F. Applications of blockchain technology in peer-to-peer energy markets and green hydrogen supply chains: a topical review. *Scientific Reports* **2024**, *14*, 21954.
7. Tang, W.; Xie, X.; Huang, Y.; Qian, T.; Li, W.; Li, X.; Xu, Z. Exploring the potential of IoT-blockchain integration technology for energy community trading: Opportunities, benefits, and challenges. *CSEE Journal of Power and Energy Systems* **2025**.
8. Wang, H.; Ma, S.; Guo, C.; Wu, Y.; Dai, H.N.; Wu, D. Blockchain-based power energy trading management. *ACM Transactions on Internet Technology (TOIT)* **2021**, *21*, 1–16.
9. O'Connor, C.; Collins, J.; Prestwich, S.; Visentin, A. Electricity price forecasting in the irish balancing market. *Energy Strategy Reviews* **2024**, *54*, 101436.
10. Heo, J.W.; Ramachandran, G.S.; Dorri, A.; Jurdak, R. Blockchain data storage optimisations: a comprehensive survey. *ACM Computing Surveys* **2024**, *56*, 1–27.
11. Xu, M.; Feng, G.; Ren, Y.; Zhang, X. On cloud storage optimization of blockchain with a clustering-based genetic algorithm. *IEEE Internet of Things Journal* **2020**, *7*, 8547–8558.
12. Zuo, Y.; Jin, S.; Zhang, S.; Zhang, Y. Blockchain storage and computation offloading for cooperative mobile-edge computing. *IEEE Internet of Things Journal* **2021**, *8*, 9084–9098.
13. Alofi, A.; Bokhari, M.A.; Bahsoon, R.; Hendley, R. Optimizing the energy consumption of blockchain-based systems using evolutionary algorithms: A new problem formulation. *IEEE Transactions on Sustainable Computing* **2022**, *7*, 910–922.
14. Akrasi-Mensah, N.K.; Agbemenu, A.S.; Nunoo-Mensah, H.; Tchao, E.T.; Ahmed, A.R.; Keelson, E.; Sikora, A.; Welte, D.; Kponyo, J.J. Adaptive storage optimization scheme for blockchain-IIoT applications using deep reinforcement learning. *IEEE Access* **2022**, *11*, 1372–1385.
15. Zhou, Y.; Ren, Y.; Xu, M.; Feng, G. An improved nsga-iii algorithm based on deep q-networks for cloud storage optimization of blockchain. *IEEE Transactions on Parallel and Distributed Systems* **2023**, *34*, 1406–1419.
16. Tanis, Z.; Durusu, A. Cooperative Behaviors and Multienergy Coupling through Distributed Energy Storage in the Peer-to-Peer Market Mechanism. *IEEE Access* **2025**.
17. Kreps, D.M. Nash equilibrium. In *Game theory*; Springer, 1989; pp. 167–177.

 doi:10.20944/preprints202508.0413.v1

18. Ma, J.; Zhang, X.; Li, X. Demystifying Blockchain Scalability: Sibling Chains with Minimal Interleaving. In Proceedings of the International Conference on Security and Privacy in Communication Systems. Springer, 2023, pp. 265–286.
19. Sutton, R.S.; Barto, A.G. *Reinforcement Learning: An Introduction*, 2nd ed.; MIT Press: Cambridge, MA, 2018.
20. Bertsekas, D.P.; Tsitsiklis, J.N. *Neuro-Dynamic Programming*; Athena Scientific: Belmont, MA, 1996.
21. Tsitsiklis, J.N. Asynchronous Stochastic Approximation and Q-Learning. *Machine Learning* **1994**, *16*, 185–202. https://doi.org/10.1023/A:1022637812044.
22. Konda, V.R.; Tsitsiklis, J.N. Actor-Critic Algorithms. *SIAM Journal on Control and Optimization* **2000**, *42*, 1143–1166. https://doi.org/10.1137/S0363012901385691.
23. Sutton, R.S.; McAllester, D.A.; Singh, S.P.; Mansour, Y. Policy Gradient Methods for Reinforcement Learning with Function Approximation. In Proceedings of the Advances in Neural Information Processing Systems 12 (NIPS 1999); Solla, S.A.; Leen, T.K.; Müller, K.R., Eds. MIT Press, 2000, pp. 1057–1063.
24. Tsitsiklis, J.N.; Van Roy, B. An Analysis of Temporal-Difference Learning with Function Approximation. *IEEE Transactions on Automatic Control* **1997**, *42*, 674–690. https://doi.org/10.1109/9.580874.
25. Kakade, S. A Natural Policy Gradient. In Proceedings of the Advances in Neural Information Processing Systems 14 (NIPS 2001); Dietterich, T.G.; Becker, S.; Ghahramani, Z., Eds. MIT Press, 2002, pp. 1531–1538.
26. Williams, R.J. Simple Statistical Gradient-Following Algorithms for Connectionist Reinforcement Learning. *Machine Learning* **1993**, *8*, 229–256. https://doi.org/10.1007/BF00992696.
27. Filar, J.; Vrieze, K. *Competitive Markov Decision Processes*; Springer: New York, NY, 1997. Chapter 3 for Nash equilibrium in competitive MDPs, https://doi.org/10.1007/978-1-4612-4054-9.
28. UN Population Division. World Urbanization Prospects, 2018. 2018 Revision.
29. International Energy Agency. Smart Meters in Buildings, 2023.
30. AWS IoT Architecture Guidelines, 2023.
31. National Bureau of Statistics of China. China Statistical Yearbook, 2023.
32. U.S. Census Bureau. American Community Survey, 2023. 2023 Estimates.
33. UN Data. India Population Projections, 2023.
34. Eurostat. Urban Population Statistics, 2023.