IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received December 27, 2021, accepted January 17, 2022, date of publication January 26, 2022, date of current version January 31, 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3146343

An Intelligent Distributed Ledger Construction Algorithm for IoT

CHARLES C. RAWLINS, (Member, IEEE), AND S. JAGANNATHAN, (Fellow, IEEE)
Department of Electrical and Computer Engineering, Missouri University of Science and Technology, Rolla, MO 65409, USA

Corresponding author: Charles C. Rawlins (crawlins@mst.edu)

This work was supported in part by the Graduate Assistance in Areas of National Need (GAANN) National Fellowship Program; and in part by the Department of Energy's Kansas City National Security Campus, operated by Honeywell Federal Manufacturing and Technologies, LLC, under Contract DE-NA0002839.

ABSTRACT Blockchain is the next generation of secure data management that creates near-immutable decentralized storage. Secure cryptography created a niche for blockchain to provide alternatives to well-known security compromises. However, design bottlenecks with traditional blockchain data structures scale poorly with increased network usage and are extremely computation-intensive. This made the technology difficult to combine with limited devices, like those in Internet of Things networks. In protocols like IOTA, replacement of blockchain's linked-list queue processing with a lightweight dynamic ledger showed remarkable throughput performance increase. However, current stochastic algorithms for ledger construction suffer distinct trade-offs between efficiency and security. This work proposed a machine-learning approach with a multi-arm bandit that resolved these issues and was designed for auditing on limited devices. This algorithm was tested in a reinforcement-learning environment simulating the IOTA ledger's construction with a decision tree. This study showed through regret analysis and experimentation that this approach was secure against impulse manipulation attacks while remaining energy-efficient. Although the IOTA protocol was a pioneer for lightweight distributed ledgers, it is expected that future blockchain protocols will adopt techniques similar to those presented in this work.

INDEX TERMS Blockchain security, distributed ledger technology, Internet of Things, machine learning, multi-arm bandit, regret analysis.

I. INTRODUCTION

Blockchain is one of the most revolutionary methods for securing network data with adoption that has exploded into a variety of data management communities, like government documents [1] and financial products [2]. Recently, Internet of Things (IoT) devices have received attention for data security in industries like supply-chain [3] and consumer households [4]. Data gathered from these devices efficiently tracks inventory and products at distant third-party servers [5]. Limited hardware and mobility of many devices usually implies reduced security [6]. Distributed IoT data lost to adversaries can be used in social engineering or behavior prediction attacks [7], creating a need for improved trust.

Blockchain is a potential solution with decentralized data management. However, there are many challenges to implementing blockchain on IoT due to traditional blockchain's design decisions. The computationally-intense queue processing of Sybil-prevention mechanisms like Proof-of-Work (PoW) in Bitcoin [8] create bottlenecks for transaction throughput and network growth that directly conflicts with the features of IoT. Recent works offloaded computation with machine-learning [9], [10] but compromise decentralization. Only efforts that alternate ledger storage formats [11]–[13] and allow concurrency with lightweight verification can be feasibly implemented on IoT devices. Of these, the IOTA protocol [11] successfully found a balance between security and performance for IoT.

An open problem in the IOTA-related literature is the development of lightweight and secure transaction selection algorithms for ledger construction [14]. The selection schemes are required to approve two existing transactions in creation of a new transaction, which decides what data persists as ground-truth and can be vulnerable to ledger-specific attacks. There are works that discuss improvements to ledger construction by expanding on basic

The associate editor coordinating the review of this manuscript and approving it for publication was Peng-Yong Kong.

methods [11], [14]. For example, E-IOTA [15] randomly varied several construction algorithms to improve transaction selection. These approaches were randomized to prevent adversarial gaming, but do not deliberately remove the distinct possibility that a transaction will go unconfirmed in the ledger. Use of complex features in transaction selection to prevent this issue, such as using a ledger or network state, were absent in the literature.

This paper proposes a ledger construction algorithm with a robust multi-arm bandit approach to resolve the issue of unconfirmed transactions while remaining secure. The bandit observes consistent rewards from the ledger environment to successfully avoid impulse attacks from adversaries and increase security. The approach also considered limited IoT devices in its design with an auditable decision tree using Q-learning from environment context. Analysis and simulations of the proposed work show superior performance for confirming late transactions compared to other schemes. Based on the author's understanding, this is the first blockchain protocol to actively use machine-learning in decision-making, which is an open opportunity in general blockchain research [16], [17].

The contributions for this work include secure reinforcement learning (RL) analysis for lightweight ledger construction and a robust multi-arm bandit algorithm. Regret analysis and simulation for performance and security guarantees were also included. Section II provides general background related to blockchain and learning mechanisms. Section IV discusses the methodology for analyzing the IOTA ledger and its security model. Section V shows results for the algorithm's effectiveness and regret analysis.

II. BACKGROUND

Combining IoT with blockchain is crucial for data security, but not feasible with traditional blockchain design. Concurrency with recent protocols' execution presented opportunities for lightweight machine learning in optimization and attack avoidance.

A. IoT SECURITY AND IOTA

Internet of Things (IoT) consists of numerous heterogeneous devices communicating to improve either consumer quality-of-life or manufacturing efficiency. Trivial schemes of collecting IoT data include traditional database systems with network-distant servers [18]. In these schemes, a user must accept consequences of not controlling their own data [19]. An advancement towards securely recording data management activity arose with blockchain networks. The core idea behind blockchain is creating a secure ledger, where transactions recorded describe network activity. The ledger is then distributed in the network and shared equally between all 'full nodes' participating and organized in block chunks. Blocks are then linked together via cryptographic hashes, hence the term 'blockchain.' Each node in a basic blockchain network contains the entire history of the ledger, though offloading can be introduced with reliance on full blockchain nodes for limited devices [9].

An example of an IoT blockchain transaction is a digitally formatted agreement between a user and a third-party to supply them data generated from a user's devices, such as temperature sensor readings from a device swarm. Since the agreement has been made publicly available on the blockchain network, users are given confidence that their data is only being used by the agreed-upon third-party, improving IoT trust and user privacy. Effectively employing limited IoT hardware as a full node remains one of the blockchain's most prevalent challenges, as the two paradigms have several characteristics that directly conflict, including: network scalability, computational demand, and large ledger storage.

The blockchain industry used to address these issues by using alternative ledger structures. Recent protocols proposed generalized schemes that form a Directed Acyclic Graph (DAG) instead of a single linked-list like in traditional blockchain. These ledgers fundamentally differ but accomplish the same tasks as traditional blockchain, so they and blockchain have been termed Distributed Ledger Technology (DLT). A DAG ledger consists of interconnected single transactions that each reference one or more previous transactions. This form factor was adopted by various DLT networks such as Hadera Hashgraph [12], Nano [13], and Obyte [20]. One pioneering DLT network offering protocol flexibility is IOTA. This protocol is an IoT-focused scheme that emphasised direct use on IoT devices with its ledger, called the Tangle.

The Tangle's dynamic benefits in ledger construction and data storage make it practical for use in IoT networks for a variety of reasons. One of the key benefits of the Tangle is the removal of strict ledger policies enforcing the same view of the ledger like with traditional blockchain. Nodes are allowed to have differing views of the ledger and are not required to directly confirm the same transactions. For example, a mobile IoT device like a smart car with intermittent network access is able to continue making transactions while disconnected from the network and then broadcasts records when reconnecting. This is the primary reason for the change in ledger structure, as relying on a linked-list blockchain would introduced bottlenecks in ledger growth. It is important to understand how the IOTA protocol is groundbreaking for allowing direct IoT device usage in DLT networks, as it presents many research opportunities for securing distributed IoT data. A simple example implementation of the IOTA network is shown in Fig. 1.

IOTA, at the time of writing, is experiencing a massive research effort that removes a centralized coordinator for verifying transactions [14]. Though the new type of ledger shape offers greater concurrency than a blockchain, it has additional challenges in protocol efficiency and security. As transactions are selected by individual nodes and appended to the tip of the DAG, there are no enforced protocols for devices to confirm a specific transaction or group of transactions like blockchain. An open area of research noted by the IOTA foundation is effective transaction selection algorithms [14].

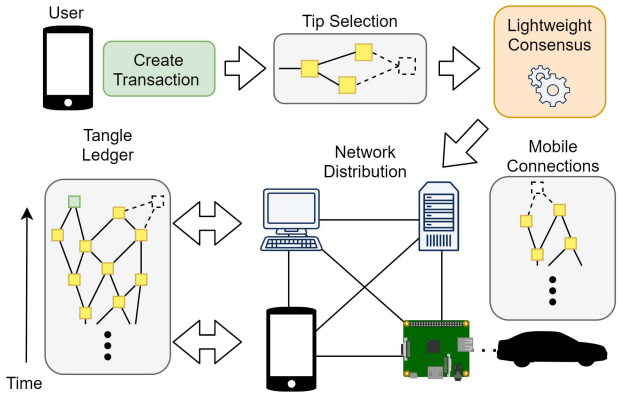


FIGURE 1. IOTA ledger construction overview.

The class of algorithms accomplishing this are simply called tip selection algorithms, which are used for constructing the DAG ledger by selecting transactions at the tip of the growing graph. A general overview of the ledger protocol and how a single new transaction is appended is shown in Fig. 1. More discussion on the security of tip selection is provided in Section IV.

B. REINFORCEMENT LEARNING

The general machine-learning methodology chosen to create an intelligent tip selection algorithm was RLQ-learning [21]. In this scheme, a learner chose an action in the action space $A \in \mathcal{A}$ to take for a given state in the state space $S \in \mathcal{S}$ of a Markov Decision Process (MDP). Each A was associated with a reward value R from a value function V, called Q-values. The optimal policy from these Q-values, Q_*, is usually approximated with a machine learner like a Multilayer Perceptron (MLP). Other lightweight algorithms like policy gradient were considered [22], but since Q-learning was simple and memory-efficient, it was chosen for an IoT environment.

A design decision considered for directly applying machine learning with blockchain was maintaining auditability. A variety of machine learning algorithms are powerful in approximating unknown nonlinear functions, but are difficult for average humans and researchers to understand. The key strength of blockchain and DLT is that anyone can verify the account balances in the ledger, providing strong security. A Decision Tree (DT) from Conservative Q-improvement [23] was chosen as the base learner for this scheme. This DT creation decides how to grow the tree based on a perceived increase in Q-values.

The Multi-arm Bandit (MAB) is a classic RL problem for determining an optimal selection from a set of discrete actions in the case of uncertain rewards. The contextual bandit was an extension of the MAB that considers an environment with different states of learning for $S \in \mathcal{S}$ [24]. The main decision-making algorithms behind a MAB are called bandit algorithms. Bandits have been analyzed through regret analysis for their performances and security guarantees. Regret is simply a value of measure used to record how the ideal action differs from the action taken by the learner and is denoted as $|R_1 - R_i|$, where R_1 is the reward for the optimal action. It was noted in related works that several lightweight bandits were gameable by adversaries, which was accomplished by poisoning rewards [25]. One effort showed that bandits like ϵ-greedy and Upper Confidence Bound (UCB) [24] (using random exploration and reduced-uncertainty of the maximal arm, respectively), were gameable in $\mathcal{O}(log(n))$ time [26]. One of the simplest robust bandit algorithms was Thompson Sampling (TS) [27]. TS is the first bandit algorithm that successfully balanced exploration and exploitation with use of the beta distribution for posterior distribution sampling. With a beta posterior distribution (denoted as *Beta*), TS finds an optimal arm for I actions with shape parameters $A_{i,0} = B_{i,0} = 1 : a = \text{argmax}_{i \in I}\{Beta(A_i, B_i)\}$. As the learner encounters changes in the environment through Bernoulli rewards $R \in [0, 1]$, the update for each shape parameter associated with the chosen action j is

$$\alpha_t, \beta_t = \begin{cases} A_i, B_i & \text{if } j \neq i \\ A_i + R,\ B_i + (1 - R) & \text{if } j = i \end{cases}$$

The security of TS showed robustness to adversarial attacks under arm manipulation [28], but [25] showed an effective scheme to poison rewards with offline data. It is important to understand the vulnerabilities of TS for creation of a robust version in Section IV.

III. RELATED WORKS

Though blockchain is a very recent technology, it has experienced a plethora of works with applications to various sectors. Recent developments with DLT networks like IOTA have also received attention for the performance and security opportunities in related research.

A. MACHINE LEARNING FOR BLOCKCHAIN WITH IoT DEVICES

Blockchain has been extensively used to improve privacy for various technologies [1]–[3] even with only recent development. Research opportunities with blockchain and machine learning have also been heavily explored [29]. While many works focus on using deep learning with blockchain as a basis for storing data [30] or cryptocurrency price predictions [31], learning optimization with blockchain for IoT has also seen many efforts in networking [9], [10], [32] and other applications [33]. However, from the surveyed literature, machine learning has not been used as the active security measure in a DLT protocol.

While there is much discussion about how blockchain can improve the efficiency and security of artificial intelligence [16], cyber-physical systems [17], and IoT [34], the opposite has not received as much attention. The primary reason for this is believed to be due to traditional blockchain's rigid data structure and execution, presenting

very few improvements for machine learning besides improving efficiency with consensus calculations [35]. As newer DLT protocols like IOTA present freedom in protocol execution, more opportunities emerge. This work is believed to be the first discussion exploring the security of the underlying blockchain protocol and improvements with machine learning for general blockchain technology, in addition to DLT protocols designed for an IoT setting.

B. IOTA LEDGER CONSTRUCTION

Ledger construction, or tip selection, in the IOTA protocol has seen several works exploring alterations to basic protocols, though there is still believed to be room for novelty at the time of writing. Original simplistic algorithms proposed in [11] were proposed for basic functionality and security against attacks. Additional work done by the IOTA foundation analyzed the performance of tip selection and the Tangle in discrete [36] and continuous time [37]. Further work by the IOTA foundation explored other methods of analyzing ledger security [38]. As the IOTA foundation worked to remove the centralized coordinator as a part of the Coordicide effort [14], the authors proposed a lighter tip selection algorithm with reliance on an external voting system [39].

Approaches by other parties were conducted to identify novel ledger construction schemes. Wang, Jia, Wang, Shen, Zhao, Chen, and Shao [40] explored increasing the efficiency of basic tip selection algorithms through heavy use of concurrency for lightweight nodes. Other works use existing algorithms, similar to this work, as a base for selection of multiple algorithms. E-IOTA [15], which expanded on G-IOTA [41], explored stochastically varying selection of several basic algorithms to improve transaction selection while remaining secure. Ferraro, Shorten, and King [42] explored used liquid modelling with IOTA's ledger and proposed their own hybrid scheme for high transaction confirmation rates. Other approaches take more novel directions to develop novel tip selection algorithms. Bramas [43] explores a two-step algorithm for resolving conflict resolution for sets of tips. Chafjiri and Mehdi Esnaashari Esfahani [44] used an adaptive random walk to improve selection efficiency.

IV. METHODOLOGY

After a brief discussion of ledger construction and security, a threat model for transaction selection is presented before providing the secure algorithms. For construction of the IOTA ledger, analysis of ideal behavior was necessary for reward function design. Security concerns and known vulnerabilities were then considered for robust decision-making. Finally, a multi-arm bandit algorithm was proposed for protection against reward attacks. This algorithm uses batch processing with highlighted rewards and filtering to prevent adversary impulse attacks from influencing learning.

A. LEDGER CONSTRUCTION AND SECURITY

The IOTA Tangle ledger is a DAG $G = (V, E)$. Each $v \in V$ contains a transaction that represented an exchange of tokens between multiple parties. Directed edge $e \in E$ represent a cryptographic link approving a previous transaction in the ledger. The only requirement in the IOTA protocol for a node approving a new transaction is confirming two previous transactions [11]. In addition, as networking delays and disconnects occur, some nodes may create subgraphs of G with differing v and e. For simplicity of analysis, these differences were ignored to create only a single version of the ledger. Any time a transaction is generated, the following steps are performed:

1) A network node creates a new transaction x
2) Nodeconducts a selection algorithm k times, where $k \geq 2$ to confirm k previous transactions
3) Nodeconductsa verification scheme to generate k cryptographic links $e \in E$
4) Broadcast e with x to the network
5) Network verifies the integrity of the cryptographic link.

It was assumed the neighbor and node verifies data with PoW, though any verification scheme could be used. At step 2, the node has an opportunity to analyze the ledger state for intelligent ledger construction.

The ledger state viewed from a single transaction x can be viewed as a discrete Markov chain [45]. Between each state, the total number of unconfirmed transactions $L(t)$ can vary along with the accumulated weight of an individual transaction $W(x)$. $L(t)$ can be modelled as a Poisson process [11] with mean rate λ. $W(x)$ is the accumulation of individual weights of all future transactions approving x, where each weight is proportional to the amount of work done to verify those transactions. It was assumed $W_0 = 1$ and all future weights are also 1 for this study.

The value $|L(t)|$ depends highly on the tip selection algorithm chosen to construct the ledger [11]. The simplest tip selection algorithm is called Uniform Random Tip Selection (URTS), which randomly selected transactions with probability $1/|L(t)|$. Each transaction has a probability > 0 of being selected, so URTS is guaranteed to not leave transactions behind [11]. The stable value for URTS is [11]

$$L(t) = \frac{k\lambda h}{k-1}, \tag{1}$$

where h is the time taken to confirm a transaction (it is assumed $h = 1$). Generally nodes choose $k = 2$, $L(t) \approx 2\lambda = L_{Ideal}$. If $L(t) < L_{Ideal}$ (due to $k > 2$), then nodes will waste resources confirming past transactions. However, one issue with URTS is that it can be easily gamed into selecting transactions approving an attack on the ledger. If an attack created some conflict in the ledger, an adversary can create numerous empty transactions approving their attack. When a node confirmed one of these dummy transactions, it helps the attacker (see Fig. 2).

With these known vulnerabilities, Markov Chain Monte Carlo (MCMC) was developed [11]. With MCMC, a weighted random walk was used to select a new transaction at the ledger edge. MCMC started from transaction in a window $[W, 2W]$, and a random walk was taken from a

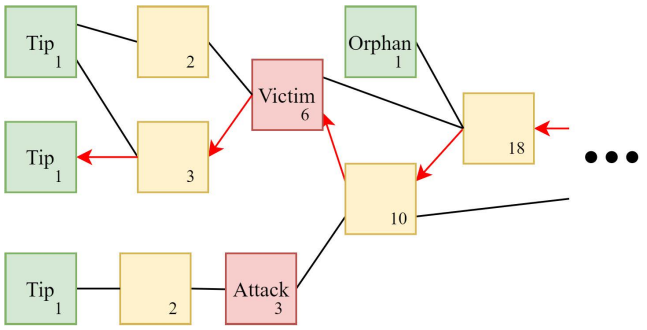


FIGURE 2. Parasite chain attack example.

transaction towards the unconfirmed tip edge. The parameter α biases selection towards transactions with higher weight accumulated from future transaction selection (see Fig. 2). $\alpha > 0$ prevents selection of transactions that would help this type of attack be successful, called a parasite chain attack. The probability of a random walker transitioning from transaction x to y when y approves x is found by [11]

$$P_{xy} = \frac{exp(\alpha W_y)}{\Sigma_{z:z\rightsquigarrow x} exp(\alpha W_z)} \tag{2}$$

where W_z is the cumulative weight of the transaction z and the sum is taken over all transactions z which approve x. High bias will not likely explore the Tangle for legal transactions that have less weight compared to the frequently-selected, meaning they will be selected with probability $\approx$0. As the majority of the network will produce higher weights with legal transactions, it will outpace the attacker. This also applies to other attacks that target cumulative weight, like ledger splitting [11].

Information was presented to a machine learner to find $S \in \mathcal{S}$. The goal of the learner is to avoid attacks while simultaneously confirming late transactions. The distribution of unapproved transactions with times falling behind the current timestamp of the next arriving transaction was represented as a distribution. Though the transactions arrive from the Poisson process with an exponential interval time, the state of current and past transaction events without approval was represented with the normal distribution mean μ_{tip} and standard deviation σ_{tip}. The following was used to represent S as a vector,

- Perceived network transaction rate λ_{obv}
- The difference $L_{Err} = |L_{Ideal} - L(t)|$
- Average tip time μ_{tip}
- Tip time standard deviation σ_{tip}

The actions chosen for the learner consisted of standard tip selection algorithms. These actions were:

- URTS with $k = 2$
- URTS with $k = 3$
- MCMC with $\alpha = 0.1,\ k = 2$
- MCMC with $\alpha = 0.9,\ k = 2$

A learner conducting either URTS or MCMC creates either exploratory or secure actions. Altering α allows a learner to alter the distribution of tips at the side or front edge of the DAG [37].

With these considerations, a reward function for ledger construction was created. The reward function with $R \in [0, 1]$ is

$$R = 0.9^{\sigma_{tip}} - 0.05(k-2) - L_B^{k/\sigma_{tip}} + B_{MCMC} + B_{URTS}, \tag{3}$$

where L_B is the number of transactions left behind,

$$L_B = \frac{|\{x \in L(t) \mid x_{t,\text{Ledger}} < (\mu_{tip} - 3 * \sigma_{tip})\}| - |M|}{|L(t)|},$$

while B_{MCMC} and B_{URTS} are

$$B_{MCMC} = 0.1\alpha_{MCMC} * \mathbb{1}\{MCMC\}$$
$$B_{URTS} = 0.01 * L_B * \mathbb{1}\{URTS\}.$$

The set M containsmalicious transactions associated with any previous attack. Behavior letting transactions fall behind was inhibited while incentivizing secure actions. This was done using the two binary (hence B_x) terms in the later part of the reward function, where each binary term is only evaluated when the respective x algorithm is selected. If *URTS* is selected, the learner is given incentive to continue selecting *URTS* if $|L_B| > 0$. When $|L_B| = 0$, no reward is gained from *URTS* and the only reward can be earned from selecting *MCMC* with high α. The constant 0.9 in (3) presents a baseline reward that was chosen arbitrarily, though it needs to be less than 1 to give reward for *MCMC* actions.

Other flat reward values for individual actions (0.1, 0.05, 0.01) were chosen so the largest constant value was rewarded to the action with the desired behavior (*MCMC*), though individual values were also chosen arbitrarily. Actions with $k > 2$ were also discouraged from selection to only when necessary in the second term, but encouraged in the third term by decreasing the negative reward for $|L_B| > 0$. $x_{t,\text{Ledger}} = t - t_{\text{Creation}}$ denotes the difference in time that a node creates or receives a transaction from the current time.

With most terms in (3), an adversary would not be able influence the reward. However, if unconfirmed transactions were spammed, L_B would increaseand createa drastic drop in reward. This was addressed with an algorithm that is robust to dramatic reward changes.

B. THREAT MODEL

Although one of the main benefits of using a DAG is the increased throughput, the downside is the additional vulnerabilities. The overall goal of an attacker is to create an attack transaction and convince the network it is correct. This is well-known in blockchain literature as a double-spend attack [46]. This attack can be accomplished through various methods with their own attack name, such as the 51% attack [46]. The exact percentage of network computing power a successful attack needs varies depending on blockchain consensus mechanisms and strategies [47]. This study considered either the brute-force strategy, where the

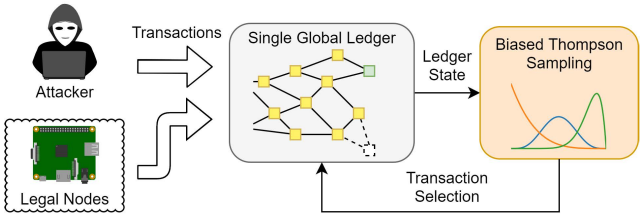


FIGURE 3. Intelligent ledger construction general threat model.

Algorithm 1 General Ledger Construction Algorithm

Initialize Tangle, $L(t)$, M, C, $i \in I$ arms
$S_0 \in \mathcal{S} : A_i = B_i = 0$
$\forall S \in \mathcal{S} : t_S = 0$
while receive $x \in L(t)$ **do**
  $S_{obv} = \{\lambda_{obv}, L_{Err}, \mu_{tip}, \sigma_{tip}\}$
  Find $S_{obv} \approx S \in \mathcal{S}$ with DT
  $j, R =$ BiasedTS(S)
  Execute $A_j \in \mathcal{A}$ and create k cryptographic edges
  Broadcast x with k links
  Update DT according to [23] with R
  if SplitNode(S) **then**
    S_{Left}, S_{Right} inherit A_i, B_i, H_i from S

attacker can only leverage blank transactions for the attack, or other ledger-specific attacks and was in the network minority. It was assumed the attacker does not control information sent to an IoT node in an Eclipse or Sybil-related attack [48]. The attacker also had perfect knowledge of the ledger learner and all internal parameters, because it was white-box. An overview of this is shown in Fig. 3.

The additional vulnerabilities from adding RL were also addressed. An attacker will try to game the decision-making process of an innocent node to help in their attack strategy through several means, such as [49]: evasion, poisoning, and exploratory. It was assumed that an adversary could alter the inputs to influence the reward during testing, meaning robustness against attacks was needed. Exploratory attacks were not beneficial to the attacker, since the learner was already white-box.

C. INTELLIGENT TRANSACTION SELECTION

The general approach for intelligent machine-learning ledger construction is shown in Algorithm 1. The core intelligence behind this approach is an extension to Bernoulli TS with an approach that emphasizes learning from consistent reward increases in Algorithm 2, called Biased TS. The comes from the adjusted shape values for each arm created by frequency of arm pulls. If an adversary poisons several reward values for an non-optimal arm, the few impulses will eventually be forgotten by another arm that creates a more consistent average from the innocent network majority for the *Beta* update in [0, 1]. The inspiration for this approach came from batch processing with TS [50] and posterior sampling [51]. Note that the weight of individual transactions is not considered in the algorithm.

Algorithm 2 BiasedTS(S)

Get $\{t_S, A_i, B_i, H_i, f_i\} \in S$
$j = \text{argmax}_{i \in I}\{Beta(A_i, B_i)\}$
$f_j += 1$
$R =$ Reward for $A_j \in \mathcal{A}$ from (3)
$H_j.append(R)$
FilteredMean(H_j)
if $t_S \geq C$ **then**
  for $i \in I$ **do**
    $l = |H_i| - 1$
    $J = max(1, f_i * \mathbb{E}(H_i[l - f_i : l]))$
    $K = max(1, f_i * (1 - \mathbb{E}(H_i[l - f_i : l])))$
    $U = Beta(J, K)$
    $A_i = A_i + U$
    $B_i = B_i + (1 - U)$
$t_S = 0$
$f_i = 0$
return j, R

Algorithm 3 FilteredMean(H_j)

$l = |H_j| - 1$
if $|H_j| < C$ **then**
  Return $H_j[l]$
$X = \{sgn(H_j[l - C + 1] - H_j[l - C]), \ldots,$
  $sgn(H_j[l] - H_j[l - 1])\}$
$Y = \{|H_j[l - C + 1] - H_j[l - C]|, \ldots, |H_j[l] - H_j[l - 1]|\}$
$W = [\,]$
$\Delta R_\mu = \mathbb{E}(Y)$
$\Delta R_\sigma = \sigma(Y)$
for y in $1 : |Y|$ **do**
  $W.append(Y[y] > \Delta R_\mu + 3 * \Delta R_\sigma)$
for w in $1 : |W|$ **do**
  if $W[w]$ **then**
    Add transactions causing attack $W[w]$ to M
    for z in $w : C - 1$ **do**
      $H_j[l - C + z + 1] = H_j[l - C + z] - X[z] * Y[z]$
Return $\mathbb{E}(H_j)$

Given the literature discussing the adversarial security of TS [25], [52], it was assumed that Biased TS was still vulnerable to manipulation with large reward values. A simple prevention scheme against reward poisoning attacks is presented in Algorithm 3. This was done with a light form of filtering shown in Algorithm 3. If an adversary were to create a parasite chain that drastically changed $|L(t)|$, it would not influence the behavior of the learner. This approach does not completely prevent selection of parasite chain transactions, but it does prevent adversarial gaming. While [53] originally presented the idea of a trimmed mean to a UCB MAB, this approach directly removes the adversary influence by

filtering out large adversarial attacks that would otherwise influence an arm's mean reward. Notice the careful indexing in Algorithm 3, where actions selected for the current batch are also compared to the previous experience if $f < C$. Legal actors can avoid transactions being added to M by slowly publishing their transactions to the network, or the node with intelligent selection can slowly adjust to the live network.

Stepping through the algorithms collectively, the IOTA node with intelligent transaction selection is initialized with the DT from [23]. Biased TS shape parameters are initially $A_i = B_i = 0$ for the first tree node S_0 and are inherited by child nodes as the tree is generated and split. The current network state S_{obv} and processed through the DT before it is passed to Biased TS. The action arm index j is chosen like the standard Bernoulli TS, where the number of arm pulls f_j are recorded during a batch count C. The history of rewards H_j is then used in Algorithm 3 to detect if the change in reward for the new sample is anomalous compared to a normal distribution of the previous C reward samples. When C actions have been taken between all arms, the beta distributions are updated for Biased TS using the previous f_i samples. Using an intermediary *Beta* distribution to update arm samples slows the learning capabilities of standard TS, but allows for identification of consistent reward updates. Consistently positive rewards will create higher values for the intermediary *Beta* over time and update the shape parameters with greater rewards than other arms.

V. RESULTS

To demonstrate the methodology in Algorithm 1-3 and from other approaches in the literature, the IOTA ledger simulator[1] was used. A RL environment for the Tangle was created in OpenAI gym in Python [54]. All simulations are provided in the repository.[2]

A. COMPARISON OF ALGORITHMS

The tip selection algorithm with Biased TS was compared to two other complex approaches. The current algorithm used by the IOTA foundation is called almostURTS [14]. The almostURTS algorithm is an extension of URTS with an added weight bias towards transactions that were confirmed with recent timestamps. To simulate network delays, noise was added to a transactions received timestamp. Without this noise, almostURTS behaved like regular URTS. The other algorithm selected was 'E-IOTA' [15]. E-IOTA varies algorithm selection randomly with MCMC and varying levels of α. The parameters for this algorithm were selected based on the best parameters presented in [15]. Each approach was then compared to pure selection of URTS and MCMC.

The training and DT parameters are shown in Table 1. The value for $\lambda = 10$ was chosen in simulation to mimic a similar value for the real IOTA network at the time of writing.[3] As the network rate can increase depending on node participation, training values for λ were varied uniformly with a bound of [5, 50]. No attacks were simulated, so line 13 in Algorithm 3 was not executed. After eachtrail, tree nodes not frequently visited were pruned. The bandit had difficulty balancing actions without prior reward experience, so learning was frozen until tree nodes were generated. New nodes in the tree inherited A_k, B_k upon splitting, and H_S was reset to 1 each time a node was split. For optimization, the parameter optimization framework Optuna [55] was used to explore the parameter space for 1600 trials. The best parameters minimizing *URTS* selection and tips left behind were then chosen manually for the final results.

TABLE 1. DT training and testing parameters.

Q-learning/Training	
Learning Rate α	0.77
Reward Bias γ	0.75
Trials	26
Biased TS Batch C	25
λ Range	[5,50]
Testing Parameters	
λ	10
Trials per. Algo.	10
MCMC α	0.9
almostURTS (C_1, C_{1p}, C_2, M)	$(1.1, 1, 1, 10)$
E-IOTA (p_1, p_2)	$(0.1, 0.35)$
Init. Split Threshold H_S	1
Visit Decay d	0.973
Split Threshold Decay D	0.749
Visit Trim Perc.	38%
Max. Tree Depth	75
Regret Simulation Parameters	
Time Steps T	$1 \cdot 10^4$
Trials	10
Arm Means	[0.3, 0.5, 0.7]
Arm Var.	0.001
Biased TS Batch C	20
Arm Budget B_k	30

The performance of each algorithm was compared by looking at the amount of transactions left behind. These were measured by counting timestamps more than three standard deviations of tip time for the pure URTS selection from the respective algorithm mean. The best-case (URTS) and worst-case (MCMC) scenarios for leaving transactions behind were also performed (see Fig. 4).

With the low number of transactions left behind for DT in Fig. 4, this shows that the approach was more effective at taking actions to confirm late transactions compared to other stochastic algorithms. The confirmation performance was more comparable to almostURTS while accomplishing secure actions. Since similar performance is achievable with secure algorithms without relying on an external voting mechanism [39], IoT devices in a distributed will have stronger security and likely avoid ledger attacks more compared to other approaches.

[1] IOTA Ledger Simulator Repository.
[2] Intelligent Ledger Construction Repository.
[3] IOTA Network Explorer.

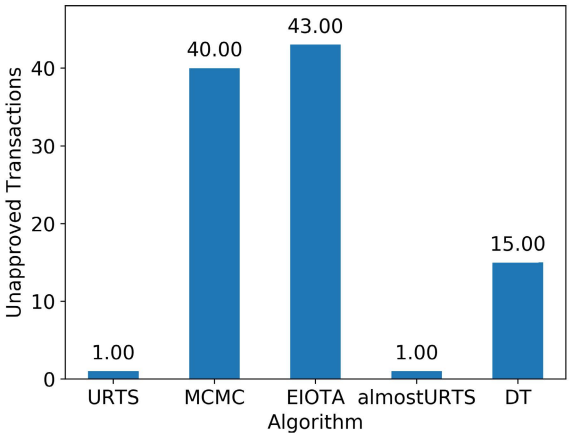


FIGURE 4. Average transactions left behind comparison.

TABLE 2. Average algorithm computation comparison (10 trials).

	URTS	MCMC	E-IOTA	almostURTS	DT
URTS	1998	0	0	1998	90
MCMC (α=0)	0	0	202	0	0
MCMC	0	1998	1796	0	1908

The number of computations for each basic algorithm were also compared in Table 2. For all algorithms besides the DT approach, the total number of tip selection executions conducted was 1998 for 1000 transactions (two for each transaction minus two for the starting transaction). Comparing the DT approach to the other selection schemes like E-IOTA with *MCMC* with $\alpha = 0$, intelligent selection was able to execute insecure actions (with URTS) fewer times for ledger construction to confirm late transactions. Other results from the parameter study were able to further reduce left behind tips, but executed insecure actions slightly more compared to the results in in Table 2. The low amount of URTS actions was achievable for the DT with a relatively high learning rate and reward bias. The slow learning of Biased TS helped child nodes in the DT improve from experience, with the drawback of long training time. Using a 'static' configuration for varying actions like E-IOTA showed that more transactions were being left behind according to the metrics used in testing.

B. REGRET ANALYSIS AND SIMULATION

An extension was provided to existing work [52] for regret with Bernoulli TS to estimate the upper bound for expected regret in Biased TS . Calculating the upper bound for regret in a control scenario provides a mathematical metric for comparing the worst-case performance of multiple MAB algorithms. Theorem 1 shows how Biased TS can be expected to behave under normal operating conditions.

Theorem 1: Let each arm $i = 1, \ldots, I$ and constant batch number C. The expected amount of regret for Biased TS in Algorithm 2 is

$$\mathbb{E}[R(T)] \leq \sum_{i}^{I} \frac{4\ lnT}{C} \Delta i + \frac{48}{\mu_i \Delta i^2}.$$

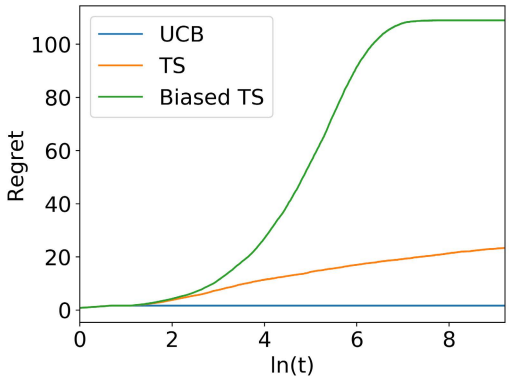


FIGURE 5. Control regret comparison ($B_i = 0$).

An approach similar to [28] was used to find expected regret of Biased TS under an adversarial setting. This analysis assumed that the adversary had a strategy of manipulating an arm with total budget B_i. According to (4) in [28], the upper bound for expected regret in TS is dependant on B_i. In the context of blockchain, this factor is crucial as an adversary with a large budget (of computation power to generate transactions) would be able to easily manipulate normal TS by spamming transactions. A tighter regret bound for Biased TS was proven without reliance on B_i and instead shows that the worst-case performance is dependant on smaller changes in reward that are able to shift the perceived distribution of an arm.

Theorem 2: Let each arm $i = 1, \ldots, k$ and constant batch number C. The expected amount of regret for Biased TS under an adversarial setting with arm budget B_i is

$$\mathbb{E}[R(T)] \leq \sum_{i}^{I} \frac{4\ lnT}{C} \Delta i + \max\{\frac{H}{2}, H\sigma logT\},$$

where $H = |3\sigma_{\Delta R} + \mu_{\Delta R}|$ for the reward change distribution.

The full proof of Theorem 1 and 2 with full term definitions is provided in the Appendix. Compared to the upper regret bounds for Gaussian TS in [28], Biased TS has worse regret in the control case from Theorem 1 due to the large constant in the second term. For the adversarial setting, a tighter regret bound with the logarithmic regret was proven compared to the natural logarithm for Gaussian TS.

Other bandit algorithms, ϵ-greedy and Gaussian TS, were compared against for their robustness to arm manipulation. The adversary had a total budget for manipulating each individual arm B_i and individual trial manipulation values $\alpha_{t,i}$ to increase a reward. Each algorithm was tested with and without the Lump Sum Investment (LSI) attack strategy outlined in [28], where the attacker spends the rest of its remaining budget on attacking an arm to convince the bandit it had higher rewards at times τ: $R_i = R_i + B_i - \Sigma_{t=1}^{\tau} \alpha_{t,i}$. Fig. 5 and Fig. 6 compare each algorithm's standard and adversarial regret respectively. The regret for each trial $|R_1 - R_i|$ was plotted and averaged over 10 trials. The simulation parameters for regret are shown in Table 1.

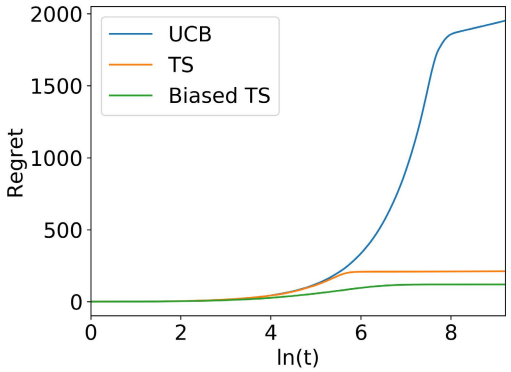


FIGURE 6. Adversarial regret comparison ($B_i = 30$).

TABLE 3. Average computation experiment comparison (50 trials).

Algorithm	Power (W)	Instr. [10^9]	Cycles [10^9]
URTS	2.367	209	117
MCMC	2.369	351	217
E-IOTA	2.368	318	196
almostURTS	2.370	234	133
DT	**2.372**	**408**	**260**

Biased TS shows noticeably worse regret under control conditions compared to Gaussian TS and UCB . The UCB algorithm was able to obtain the optimal action quickly with little increase in regret, while the Gaussian TS took longer on average. For a small attack budget in Fig. 6, Biased TS maintained near-identical regret levels to the control case This shows that Biased TS is a more secure algorithm for preventing arm manipulation compared to the other approaches for strong impulse attacks, which would be caused by a spam of transactions and a drop in reward for (3). Gaussian TS had slightly worse regret for $B_i = 30$ and increased from higher values in further testing. The UCB algorithm had drastically worse regret for the small attack and continued to increase with stronger attacks.

One note about the attack results in Fig. 6 is that the LSI attack strategy outlined in [28] is not believed to be the most effective strategy for Biased TS. Since rewards for Biased TS are limited to $[0, 1]$ while other approaches have rewards in $\mathbb{R}$, comparing the attack performance of the standard MAB algorithms to Biased TS is not fair. Scaling the reward proved difficult to obtain stable regret results from all bandits simultaneously. More effective attacks against Biased TS and comparison to other algorithms will be explored in future work.

C. PERFORMANCE MEASUREMENTS

For consideration of IoT devices, tip selection with Biased TS was measured for its energy consumption and throughput performance.

1) ENERGY CONSUMPTION

A Raspberry Pi 4 Model B with the Raspbian OS was running Python 3.7 with code provided in the repository using a standard 5V power supply. The Pi uses a 1.5 GHz CPU with the Arm Cortex-A72 architecture. Each algorithm in Table 3 was executed for 10 trials of creating 1000 transactions with $\lambda = 10$. Each group of trials was repeated 5 times for a total of 50 trials. The Pi was monitored with a MakerHawk USB Power monitor and recorded on a separate computer. The voltage and current were measured and averaged over each trial group and recorded in Table 3. In addition, the Linux tool `perf stat` was used to estimate the number of cycles and instructions generated by each algorithm. The results showed that the DT did not cause a drastic power draw or computation compared to tested algorithms. Note the scale to 0.5 V in Fig. 7, where the spikes created form the DT approach are likely correlated to underlying operating systems execution and not a concern for the tip selection energy draw.

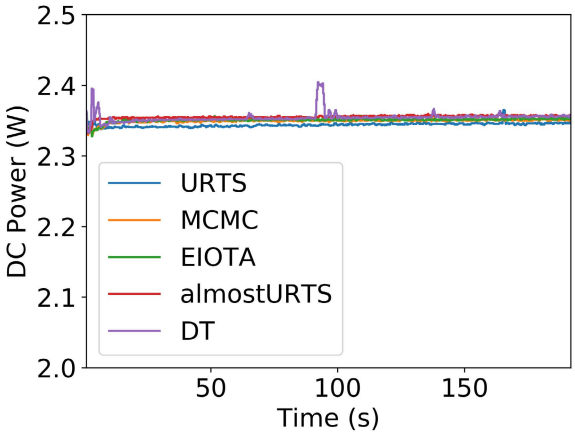


FIGURE 7. Tip selection energy comparison.

2) THROUGHPUT PERFORMANCE

Each algorithm's throughput to conduct transaction selection was also compared. Insight into how the approach with Biased TS will perform can be shown by running isolated ledger constructions algorithms. For this approach, the same setup used to generate Fig. 8 for the energy analysis was used with the Raspberry Pi and optimized decision tree to run experiments recording the individual execution time of tip algorithm selection for each type of ledger construction in Fig. 8. Each algorithm was executed for a total of 50 trials with the average recorded.

The optimized decision tree algorithm takes longer to execute than other existing approaches due to execution of the DT code, including pure MCMC. In response to this, a separate DT limited to 10 nodes (but not optimized) was also executed for comparison using the same setup. This limited DT was able to execute similarly to pure MCMC. It may be possible to further reduce the execution time of transaction selection by finding a more powerful greatly optimized decision tree, though this performance is not expected to decrease dramatically as the base tip selection algorithms still need to be executed. It should be acknowledged that these results could be better optimized for low-level hardware if

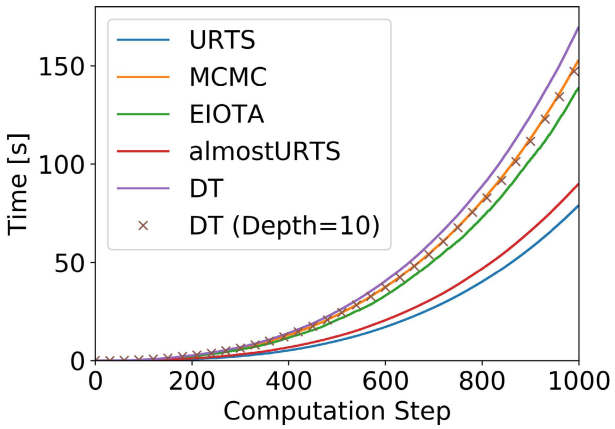


FIGURE 8. Algorithm cumulative execution delay comparison.

executed in a lower-level programming language, or rewritten for concurrency [40].

D. REFLECTION AND FUTURE WORK

Ledger construction with Biased TS showed improved performance with intelligent selection. Highlighting consistent reward changes and filtering outliers successfully prevents impulse attacks. Bounding filtered values by outliers in the reward change distribution were key for logarithmic regret bounds. One issue with the learning strength of DT is it needed incentive to minimize selection of URTS by adding B_{URTS} and B_{MCMC} to (3), thus increasing complexity. Another issue worth noting is that an adversary may have more success with manipulation if they are able to create consistent poisoned rewards, though only if they are not network minority. Attacks against intelligent transaction selection were not simulated (also without executing line 13 in Algorithm 3), so incorporating a more realistic live simulations may also alter results and could be explored by future researchers. This study is still believed to be, at the time of writing, the first to discuss active use of machine learning in blockchain technology. Future researchers could explore some of the issues presented with this work to expand the discussion and strengthen the security of intelligent ledger construction.

With creation of Biased TS and effective construction of the Tangle ledger, what are the practical applications beyond this approach? Comparing the base IOTA protocol to other blockchains, individual transactions are processed by each node before ledger appending. In standard execution, transactions are approved blindly with conflict resolution done in post. There is an opportunity for distributed nodes to intelligently analyze individual transactions before they are approved in the ledger by comparing them to previous attack history M in (3). With intelligent analysis, it may be possible to detect the attack and rerun transaction selection to avoid the adversary ledger influence. Using the robust tip selection presented in this work, it would be difficult for adversaries to manipulate the input to analysis algorithms, increasing security. Future work will explore this topic further with efforts to improve analysis of rare attack transactions through data augmentation techniques.

VI. CONCLUSION

This effort presented a machine learning-based DLT ledger construction algorithm for IoT with Biased TS. The algorithm's performance was compared to other schemes through simulation and improved transaction confirmation performance was found. This approach was also comparable in energy consumption and execution time to other tested algorithms. Proof of security against manipulation for Biased TS was shown with better regret under attacks compared to other bandits. Future work will explore directions for more effective detection of rare attacks.

APPENDIX
PROOFS

A. DEFINITIONS AND ASSUMPTIONS

The definitions below are for understanding the proceeding analysis. Several definitions are also found in [52], while others are unique to this work.

Definition 1: *($n_i(t)$, $\hat{\mu}_i$, μ_1, $\mu_{i,C}$, Δ_i, $\theta_i(t)$) Let $n_i(t)$ be the number of times arm i was pulled. $\hat{\mu}_i$ is the empirical mean for arm i, which is defined as*

$$\hat{\mu}_i = \frac{\sum_{\tau=1:i(\tau)=i}^{t-1} r_i(\tau)}{n_i(t)+1}$$

μ_i is the reward mean for arm i and μ_1 is the optimal arm mean. $\mu_{i,C}$ is the mean regret created by the beta distribution update in Algorithm 2, This value is expanded upon in the adversarial proof but used on its own in the general proof. $\Delta_i = |\mu_1 - \mu_i|$ is the remaining difference between the optimal and non-optimal arms for each arm i. $\theta_i(t)$ is a posterior distribution sample.

Definition 2 (x_i and y_i): x_i and y_i are two thresholds for $\theta_i(t)$ where $\mu_i < x_i < y_i < \mu_1$.

Definition 3 ($E_i^{\mu}(t)$ and $E_i^{\theta}(t)$): Event $E_i^{\mu}(t)$ occurs when $\hat{\mu}_i \leq x_i$ and $E_i^{\theta}(t)$ occurs when $\theta_i(t) \leq y_i$.

Definition 4 (Distribution Measurements): $F_{n,p}^{B}$ and $f_{n,p}^{B}$ denote the binomial distribution's cdf and pmf respectively. $F_{\alpha,\beta}^{Beta}$ is the cdf of the beta distribution. In addition, to relate the binomial and beta distributions: $F_{\alpha,\beta}^{Beta}(y) = 1 - F_{\alpha+\beta-1,y}^{B}(\alpha-1)$.

Definition 5 ($\mathcal{H}_t$): $\mathcal{H}_t$ is the arm pull history where $\mathcal{H}_t = \{i(\tau), r_{i(\tau)}(\tau), \tau = 1, \ldots, t\}$, $i(\tau)$ is the arm played for time τ, $r_{i(\tau)}(\tau)$ is the reward observed from time τ.

Definition 6 (Multiplicative Chernoff Bound): The multiplicative Chernoff Bound defines a bound for the tails of a distribution. This general bound applies to any random variable with $\mu = \mathbb{E}(x)$. (Theorem 4.4 in [56])

$$P(X \geq (1+\delta)\mu) \leq exp(\frac{-\mu\delta^2}{3})$$

Definition 7 (H_i): H_i is the maximum value that a change in reward for arm i can take before it is filtered out by

Algorithm 3: $H = |3\sigma_{\Delta R} + \mu_{\Delta R}|$. *This value assumes that the reward and absolute change distributions are sub-Gaussian.*

Definition 8 (B_i and Attack Strategy): *Let B_i note the budget an adversary has to influence the reward for non-optimal arms. We assume the adversary chooses the optimal attack strategy with the LSI approach [28]. For a general attack, the adversary influence is $\beta_i = \frac{B_i}{n_i(t-1)}$. For the LSI attack, the adversary influence is a massive influence over the arm by using the accrued values $\alpha_i = \frac{B_i}{t}$. The adversary applies the influence using the LSI strategy by $\beta_{i,t} = B_i - \sum_{\tau=1}^{t-1} \alpha_{i,\tau}$.*

B. BANDIT ALGORITHM GENERAL REGRET

The approach to finding the general regret analysis upper bound is an extension of thorough work in [52]. Where this work deviates is changing the reward to TS with an augmented beta distribution found in Algorithm 2.

Lemma 1:

$$\sum_{t=1}^{T} P(i(t) = i, \overline{E_i^{\theta}(t)}, E_i^{\mu}(t)) \leq \frac{4\ lnT}{C} + 1$$

Proof of Lemma 1: A function that bounds $n_i(t)$ for each arm $Q_i(T)$ is found by

$$\begin{aligned}\sum_{t=1}^{T} P(i(t) = i, \overline{E_i^{\theta}(t)}, E_i^{\mu}(t)) \\ = \sum_{t=1}^{T} P(i(t) = i, n_i(t)\ \leq Q_i(T), \overline{E_i^{\theta}(t)}, E_i^{\mu}(t)) \\ + \sum_{t=1}^{T} P(i(t) = i, n_i(t) > Q_i(T), \overline{E_i^{\theta}(t)}, E_i^{\mu}(t))\end{aligned}$$

$Q_i(t)$ bounds $n_i(t)$ in the first term. For the second term,

$$\begin{aligned}\sum_{t=1}^{T} P(i(t) = i, n_i(t) > Q_i(T), \overline{E_i^{\theta}(t)}, E_i^{\mu}(t)) \\ \leq \mathbb{E}[\sum_{t=1}^{T} \mathbb{I}(n_i(t) > Q_i(T), \hat{\mu}_i(t) \leq x_i) \cdot P(\theta_i(t) > y_i | \mathcal{H}_{t-1})]\end{aligned}$$

When $\hat{\mu}_i(t) \leq x_i$, then the biased reward update in Algorithm 2 stochastically dominates $\theta_i(t)$:$Beta(U, (1-U))$, If $U = Beta(f_i \cdot \mu_{i,C}, f_i \cdot (1-\mu_{i,C}))$, the standard beta distribution average reduces the update to $\mu_{i,C}$. $\theta_i(t)$ is then dominated by beta $Beta(x_i B_i \mu_{i,C} + \mu_{i,C}, (1-x_i)B_i\mu_{i,C})$, where B_i is the number of batches where i is selected. Given an instance H_{t-1} of $\mathcal{H}_{t-1}$ where $\hat{\mu}_i(t) \leq x_i$ and $n_i(t) > Q_i(T)$,

$$\begin{aligned}P(\theta_i(t)\ > y_i | \mathcal{H}_{t-1} = H_{t-1}) \\ \leq 1 - F^{Beta}_{\mu_{i,C}(x_i B_i + 1), \mu_{i,C}(1-x_i)B_i}(y_i)\end{aligned}$$

According to Def. 4,

$$\begin{aligned}1 - F^{Beta}_{\mu_{i,C}(x_i B_i + 1), \mu_{i,C}(1-x_i)B_i}(y_i) \\ = F^{B}_{\mu_{i,C}(1+B_i), y_i}(\mu_{i,C}(x_i B_i + 1))\end{aligned}$$

Using Def. 6 and substituting $B_i = \lfloor \frac{n_i(t)}{C} \rfloor$ for $C > 1$, set $Q_i(T) = \frac{4\ lnT}{C}$

$$\begin{aligned}F^{B}_{\mu_{i,C}(1+B_i), y_i}(x_i \mu_{i,C} B_i + x_i) \\ \leq exp(-\frac{\mu_{i,C}(x_i B_i)^2}{3}) \\ \leq exp(-\frac{\mu_{i,C}(x_i \lfloor \frac{n_i(t)}{C} \rfloor)^2}{3}) \leq exp(-\frac{\mu_{i,C}(x_i Q_i(T))^2}{3} \\ \leq \frac{1}{T} \leq 1.\end{aligned}$$

Putting in the bounds for the original equation,

$$\sum_{t=1}^{T} P(i(t) = i, \overline{E_i^{\theta}(t)}, E_i^{\mu}(t)) \leq \frac{4\ lnT}{C} + 1.$$

□

Lemma 2:

$$\sum_{t=1}^{T} P(i(t) = i, \overline{E_i^{\mu}(t)}) \leq \frac{48}{\mu_i \Delta_i^2}$$

Proof of Lemma 2: Let τ_n denote the n^{th} trial that arm i is pulled,

$$\begin{aligned}\sum_{t=1}^{T} P(i(t) = i, \overline{E_i^{\mu}(t)}) \leq \sum_{n_i=0}^{T-1} P(\overline{E_i^{\mu}(\tau_{n_i+1})}) \\ = \sum_{n_i=0}^{T-1} P(\hat{\mu}_i(\tau_{k+1}) > x_i).\end{aligned}$$

At time τ_{k+1} for $k \geq 1$, $\hat{\mu}_i(\tau_{n+1}) = \frac{\mu_{i,C}}{n_i+1} \leq \frac{\mu_{i,C}}{n_i}$. Using Def. 6 to define a bound,

$$\begin{aligned}\sum_{n_i=0}^{T} P(\hat{\mu}_i(\tau_{k+1}) > x_i) \leq \sum_{n_i=0}^{T} P(\frac{\mu_{i,C}}{n_i} > x_i) \\ \leq \sum_{n_i=0}^{T} exp(-\frac{n_i \mu_{i,C} \delta_i^2}{3}).\end{aligned}$$

Letting $\delta_i = \frac{\Delta_i}{4}$,

$$\begin{aligned}\sum_{n_i=0}^{T} exp(-\frac{n_i \mu_{i,C} \delta_i^2}{3}) \leq 1 + \sum_{n_i=1}^{T} exp(-\frac{n_i \mu_{i,C} \delta_i^2}{3}) \\ \leq 1 + \frac{48}{\mu_i \Delta_i^2}.\end{aligned}$$

Thus, the bound is

$$\sum_{n_i=0}^{T} P(i(t) = i, \overline{E_i^{\mu}(t)}) \leq 1 + \frac{48}{\mu_i \Delta_i^2}.$$

□

Proof of Theorem 1: Following a similar strategy to Theorem 1.1 and Lemmas 2.9-2.12 in [52], regret analysis followed a martingale approach by finding individual term

bounds for Biased TS. The expected number of arm pulls can be found by

$$\mathbb{E}[n_i(T)] = \sum_{t=1}^{T} P(i(t) = i, E_i^{\mu}(t), E_i^{\theta}(t)) + \sum_{t=1}^{T} P(i(t) = i, E_i^{\mu}(t), \overline{E_i^{\theta}(t)}) + \sum_{t=1}^{T} P(i(t) = i, \overline{E_i^{\mu}(t)}). \quad (4)$$

The first term is bounded by $\mathcal{O}(1)$ according to Lemma 2.10 in [52]. The remaining two terms are influenced by the beta distribution reward and their bounds are determined by Lemmas 1 and 2. Plugging in the bounds for these terms,

$$\mathbb{E}[n_i(T)] \leq \mathcal{O}(1) + 1 + \frac{4ln(T)}{C} + 1 + \frac{48}{\mu_i \Delta_i^2} = \frac{4ln(T)}{C} + \frac{48}{\mu_i \Delta_i^2},$$

and then finding the standard regret,

$$\mathbb{E}[R(T)] \leq \sum_i \Delta_i \mathbb{E}[n_i(T)] \leq \sum_i \frac{4ln(T)}{C} \Delta_i + \frac{48}{\mu_i \Delta_i}.$$

□

C. ADVERSARIAL REGRET

The adversarial regret analysis approach also extends on the work of [28], but an adversary has a limited budget to influence a bandit algorithm towards non-optimal arms.

Lemma 3:

$$\sum_{t=1}^{T} P(i(t) = i, \overline{E_i^{\mu}(t)}) \leq max\{\frac{H}{2\Delta_i}, \frac{H\sigma logT}{\Delta_i}\} + 1$$

Proof of Lemma 3: We find a function that bounds $n_i(t)$ for each arm $Q_i(T)$, but under an adversarial setting like Lemma C.5 in [28].

$$\sum_{t=1}^{T} P(i(t) = i, \overline{E_i^{\mu}(t)}) = \sum_{t=1}^{T} P(i(t) = i, n_i(t) \leq Q_i(T), \overline{E_i^{\mu}(t)}) + \sum_{t=1}^{T} P(i(t) = i, n_i(t) > Q_i(T), \overline{E_i^{\mu}(t)})$$

The first term is bounded by $Q_i(T)$, but the adversary influences the second term.

$$\sum_{t=1}^{T} P(i(t) = i, n_i(t) > Q_i(T), \overline{E_i^{\mu}(t)}) \leq \sum_{t=1}^{T} P(\hat{\mu}_i + \beta_i \geq x_i | n_i(t-1) \geq Q_i(T))$$

To represent the adversary's influence in the second term, we establish how the adversary influences $\hat{\mu}_i$ with the security algorithms from μ_i. Since any large changes in reward are removed by algorithm 3, $\beta_i \leq H$. The bias introduced by f_i in Algorithm 2 also limits the adversaries attacks by $\frac{f_i}{C}$, where we assume $f_i \leq \frac{C}{2}$ for $i \neq 1$. With both of these limits,

$$\beta_i = \frac{f_i}{C}(\frac{B_i}{n_i(t)}) \leq \mu_i \leq \frac{f_i}{C} H_i \leq \frac{H_i}{2\Delta_i}$$

β_i is bounded by $\frac{H\sigma\ logT}{\Delta_i}$,

$$Q_i(T) = max\{\frac{H}{2\Delta_i}, \frac{H\sigma logT}{\Delta_i}\}.$$

Using the above bound and Def. 6,

$$\sum_{t=1}^{T} P(\hat{\mu}_i + \beta_i \geq x_i | n_i(t-1) \geq Q_i(T)) = \sum_{t=1}^{T} P(\hat{\mu}_i + \frac{f_i}{C}(\frac{B_i}{n_i(t-1)}) \geq x_i | n_i(t-1) \geq Q_i(T)) \leq \sum_{t=1}^{T} exp(-\frac{\hat{\mu}_i (\frac{f_i}{C}(\frac{B_i}{n_i(t-1)}))^2}{3}) \leq \frac{1}{T} \leq 1$$

Plugging in the appropriate terms, the bound is

$$\sum_{t=1}^{T} P(i(t) = i, \overline{E_i^{\theta}(t)}, E_i^{\mu}(t)) \leq max\{\frac{H}{2\Delta_i}, \frac{H\sigma logT}{\Delta_i}\} + 1.$$

□

Proof of Theorem 2: Following the same approach as Theorem 1, we find regret bounds for individual terms of the expected number of arm pulls in (4). Bounds for the first two terms are found like Theorem 1, the third term is bounded by Lemma 3 to show adversary influence. Plugging in these terms,

$$\mathbb{E}[n_i(T)] \leq \mathcal{O}(1) + 1 + \frac{4\ ln(T)}{C} + 1 + max\{\frac{H}{2\Delta_i}, \frac{H\sigma logT}{\Delta_i}\} = \frac{4ln(T)}{C} + max\{\frac{H}{2\Delta_i}, \frac{H\sigma logT}{\Delta_i}.\}$$

Following with standard regret, the bound is

$$\mathbb{E}[R(T)] \leq \sum_i \Delta_i \mathbb{E}[n_i(T)] \leq \sum_i \frac{4ln(T)}{C} \Delta_i + max\{\frac{H}{2}, H\sigma logT\}$$

□

REFERENCES

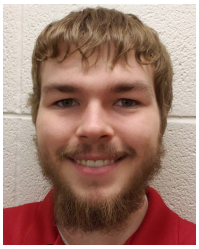
[1] A. Mohite and A. Acharya, "Blockchain for government fund tracking using hyperledger," in *Proc. Int. Conf. Comput. Techn., Electron. Mech. Syst. (CTEMS)*, Dec. 2018, pp. 231–234.

[2] B. Chen, Z. Tan, and W. Fang, "Blockchain-based implementation for financial product management," in *Proc. 28th Int. Telecommun. Netw. Appl. Conf. (ITNAC)*, Nov. 2018, pp. 1–3.

[3] Y. P. Tsang, K. L. Choy, C. H. Wu, G. T. S. Ho, and H. Y. Lam, "Blockchain-driven IoT for food traceability with an integrated consensus mechanism," *IEEE Access*, vol. 7, pp. 129000–129017, 2019.

[4] T. Song, R. Li, B. Mei, J. Yu, X. Xing, and X. Cheng, "A privacy preserving communication protocol for IoT applications in smart homes," *IEEE Internet Things J.*, vol. 4, no. 6, pp. 1844–1852, Dec. 2017.
[5] J. H. Jeon, K. Kim, and J. Kim, "Block chain based data security enhanced IoT server platform," in *Proc. IEEE Int. Conf. Inf. Netw. (ICOIN)*, Jan. 2018, pp. 941–944.
[6] F. Meneghello, M. Calore, D. Zucchetto, M. Polese, and A. Zanella, "IoT: Internet of Threats? A survey of practical security vulnerabilities in real IoT devices," *IEEE Internet Things J.*, vol. 6, no. 5, pp. 8182–8201, Oct. 2019.
[7] M. Ghasemi, M. Saadaat, and O. Ghollasi, "Threats of social engineering attacks against security of Internet of Things (IoT)," in *Fundamental Research in Electrical Engineering* (Lecture Notes in Electrical Engineering), S. M. Kouhsari, Ed. Singapore: Springer, 2019, pp. 957–968.
[8] S. Nakamoto. (2009). *Bitcoin: A Peer-to-Peer Electronic Cash System*. [Online]. Available: https://bitcoin.org/bitcoin.pdf
[9] X. Qiu, L. Liu, W. Chen, Z. Hong, and Z. Zheng, "Online deep reinforcement learning for computation offloading in blockchain-empowered mobile edge computing," *IEEE Trans. Veh. Technol.*, vol. 68, no. 8, pp. 8050–8062, Aug. 2019.
[10] N. Mhaisen, N. Fetais, A. Erbad, A. Mohamed, and M. Guizani, "To chain or not to chain: A reinforcement learning approach for blockchain-enabled IoT monitoring applications," *Future Gener. Comput. Syst.*, vol. 111, pp. 39–51, Oct. 2020.
[11] S. Popov, "The tangle," Apr. 2018.
[12] D. L. Baird, M. Harmon, and P. Madsen. (2020). *Hedera: A Public Hashgraph Network & Governing Council*. [Online]. Available: https://hedera.com/hh-whitepaper
[13] C. LeMahieu. (2017). *Nano: A Feeless Distributed Cryptocurrency Network*. [Online]. Available: https://content.nano.org/whitepaper/Nano_Whitepaper_en.pdf
[14] *The Coordicide*, IOTA Foundation, Berlin, Germany, 2019.
[15] G. Bu, W. Hana, and M. Potop-Butucaru, "Metamorphic IOTA," 2019, *arXiv:1907.03628*.
[16] K. Salah, M. Rehman, N. Nizamuddin, and A. Al-Fuqaha, "Blockchain for AI: Review and open research challenges," *IEEE Access*, vol. 7, pp. 10127–10149, 2019.
[17] W. Zhao, C. Jiang, H. Gao, S. Yang, and X. Luo, "Blockchain-enabled cyber–physical systems: A review," *IEEE Internet Things J.*, vol. 8, no. 6, pp. 4023–4034, Mar. 2021.
[18] F. Loi, A. Sivanathan, H. H. Gharakheili, A. Radford, and V. Sivaraman, "Systematically evaluating security and privacy for consumer IoT devices," in *Proc. Workshop Internet Things Secur. Privacy*. New York, NY, USA: ACM, Nov. 2017, pp. 1–6, doi: 10.1145/3139937.3139938.
[19] H. A. Abdulghani, N. A. Nijdam, A. Collen, and D. Konstantas, "A study on security and privacy guidelines, countermeasures, threats: IoT data at rest perspective," *Symmetry*, vol. 11, no. 6, p. 774, Jun. 2019. [Online]. Available: https://www.mdpi.com/2073-8994/11/6/774
[20] A. Churyumov. (2016). *Byteball: A Decentralized System for Storage and Transfer of Value*. [Online]. Available: https://obyte.org/Byteball.pdf
[21] C. Watkins, "Learning from delayed rewards," Ph.D. dissertation, King's College, Cambridge, U.K., 1989.
[22] R. S. Sutton, D. McAllester, S. Singh, and Y. Mansour, "Policy gradient methods for reinforcement learning with function approximation," in *Advances in Neural Information Processing Systems*, vol. 12. Cambridge, MA, USA: MIT Press, 2000. [Online]. Available: https://papers.nips.cc/paper/1999/hash/464d828b85b0bed98e80ade0a5c43b0f-Abstract.html
[23] A. M. Roth, N. Topin, P. Jamshidi, and M. Veloso, "Conservative Q-improvement: Reinforcement learning for an interpretable decision-tree policy," 2019, *arXiv:1907.01180*.
[24] R. S. Sutton and A. G. Barto, *Multi-Armed Bandits*. Cambridge, MA, USA: MIT Press, 2018, ch. 2, pp. 25–27.
[25] F. Liu and N. Shroff, "Data poisoning attacks on stochastic bandits," 2019, *arXiv:1905.06494*.
[26] K.-S. Jun, L. Li, Y. Ma, and X. Zhu, "Adversarial attacks on stochastic bandits," 2018, *arXiv:1810.12188*.
[27] W. R. Thompson, "On the likelihood that one unknown probability exceeds another in view of the evidence of two samples," *Biometrika*, vol. 25, nos.3–4, pp. 285–294, Dec. 1933. [Online]. Available: https://www.jstor.org/stable/2332286
[28] Z. Feng, D. C. Parkes, and H. Xu, "The intrinsic robustness of stochastic bandits to strategic manipulation," 2019, *arXiv:1906.01528*.
[29] S. Tanwar, Q. Bhatia, P. Patel, A. Kumari, P. K. Singh, and W.-C. Hong, "Machine learning adoption in blockchain-based smart applications: The challenges, and a way forward," *IEEE Access*, vol. 8, pp. 474–488, 2019.
[30] W. Meng, E. W. Tischhauser, Q. Wang, Y. Wang, and J. Han, "When intrusion detection meets blockchain technology: A review," *IEEE Access*, vol. 6, pp. 10179–10188, 2018.
[31] P. Jaquart, D. Dann, and C. Weinhardt, "Short-term bitcoin market prediction via machine learning," *J. Finance Data Sci.*, vol. 7, pp. 45–66, Nov. 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2405918821000027
[32] C. Qiu, X. Wang, H. Yao, J. Du, F. R. Yu, and S. Guo, "Networking integrated cloud–edge–end in IoT: A blockchain-assisted collective *Q*-learning approach," *IEEE Internet Things J.*, vol. 8, no. 16, pp. 12694–12704, Aug. 2021.
[33] A. A. Sadawi, M. S. Hassan, and M. Ndiaye, "A survey on the integration of blockchain with IoT to enhance performance and eliminate challenges," *IEEE Access*, vol. 9, pp. 54478–54497, 2021.
[34] N. Waheed, X. He, M. Ikram, M. Usman, S. S. Hashmi, and M. Usman, "Security and privacy in IoT using machine learning and blockchain: Threats and countermeasures," 2020, *arXiv:2002.03488*.
[35] J. Weng, J. Weng, J. Zhang, M. Li, Y. Zhang, and W. Luo, "DeepChain: Auditable and privacy-preserving deep learning with blockchain-based incentive," *IEEE Trans. Dependable Secure Comput.*, vol. 18, no. 5, pp. 2438–2455, Oct. 2021.
[36] B. Kusmierz. (2017). *The First Glance at the Simulation of the Tangle: Discrete Model*. [Online]. Available: https://www.semanticscholar.org/paper/The-first-glance-at-the-simulatio%n-of-the-Tangle-%3A-Kusmierz/e2f388505670608c20d6d801cb152c18f7a8a56f
[37] B. Kusmierz, P. Staupe, and A. Gal. (2018). *Extracting Tangle Properties in Continuous Time via Large-Scale Simulations*. [Online]. Available: https://www.semanticscholar.org/paper/Extracting-Tangle-Properties-in-C%ontinuous-Time-via-Kusmierz-Staupe/176d4a6361bd5cbc6f1b0df939e41f901bc05c11
[38] A. Penzkofer, B. Kusmierz, A. Capossele, W. Sanders, and O. Saa, "Parasite chain detection in the IOTA protocol," 2020, *arXiv:2004.13409*.
[39] S. Popov. (2021). *(Almost) URTS on a Subset*. [Online]. Available: https://iota.cafe/t/almost-urts-on-a-subset/234
[40] Q. Wang, Z. Jia, T. Wang, Z. Shen, M. Zhao, R. Chen, and Z. Shao, "A highly parallelized PIM-based accelerator for transaction-based blockchain in IoT environment," *IEEE Internet Things J.*, vol. 7, no. 5, pp. 4072–4083, May 2020.
[41] G. Bu, Ö. Gürcan, and M. Potop-Butucaru, "G-IOTA: Fair and confidence aware tangle," 2019, *arXiv:1902.09472*.
[42] P. Ferraro, R. Shorten, and C. King, "On the stability of unverified transactions in a DAG-based distributed ledger," *IEEE Trans. Autom. Control*, vol. 65, no. 9, pp. 3772–3783, Sep. 2020.
[43] Q. Bramas, "Efficient and secure TSA for the tangle," in *Networked Systems* (Lecture Notes in Computer Science), K. Echihabi and R. Meyer, Eds. Cham, Switzerland: Springer, 2021, pp. 161–166.
[44] F. S. Chafjiri and M. M. E. Esfahani, "An adaptive random walk algorithm for selecting tips in the tangle," in *Proc. 5th Int. Conf. Web Res. (ICWR)*, Apr. 2019, pp. 161–166.
[45] Y. Li, B. Cao, M. Peng, L. Zhang, L. Zhang, D. Feng, and J. Yu, "Direct acyclic graph-based ledger for Internet of Things: Performance and security analysis," *IEEE/ACM Trans. Netw.*, vol. 28, no. 4, pp. 1643–1656, Aug. 2020.
[46] M. Saad, J. Spaulding, L. Njilla, C. Kamhoua, S. Shetty, D. Nyang, and A. Mohaisen, "Exploring the attack surface of blockchain: A systematic overview," 2019, *arXiv:1904.03487*.
[47] I. Eyal and E. Gun Sirer, "Majority is not enough: Bitcoin mining is vulnerable," 2013, *arXiv:1311.0243*.
[48] M. Castro, P. Druschel, A. Ganesh, A. Rowstron, and D. S. Wallach, "Secure routing for structured peer-to-peer overlay networks," *ACM SIGOPS Operating Syst. Rev.*, vol. 36, no. SI, pp. 299–314, Dec. 2002, doi: 10.1145/844128.844156.
[49] A. Chakraborty, M. Alam, V. Dey, A. Chattopadhyay, and D. Mukhopadhyay, "Adversarial attacks and defences: A survey," 2018, *arXiv:1810.00069*.
[50] S. Ray Chowdhury and A. Gopalan, "On batch Bayesian optimization," 2019, *arXiv:1911.01032*.
[51] I. Osband, D. Russo, and B. Van Roy, "(More) efficient reinforcement learning via posterior sampling," 2013, *arXiv:1306.0940*.
[52] S. Agrawal and N. Goyal, "Near-optimal regret bounds for Thompson sampling," *J. ACM*, vol. 64, no. 5, pp. 30:1–30:24, Sep. 2017, doi: 10.1145/3088510.
[53] L. Niss and A. Tewari, "What you see may not be what you get: UCB bandit algorithms robust to ε-contamination," 2019, *arXiv:1910.05625*.

[54] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba, "OpenAI Gym," 2016, *arXiv:1606.01540*.
[55] T. Akiba, S. Sano, T. Yanase, T. Ohta, and M. Koyama, "Optuna: A next-generation hyperparameter optimization framework," in *Proc. 25th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Jul. 2019, pp. 2623–2631.
[56] M. Mitzenmacher, E. Upfal, and C. U. Press, *Probability and Computing: Randomized Algorithms and Probabilistic Analysis*. Cambridge, U.K.: Cambridge Univ. Press, 2005. [Online]. Available: https://books.google.com/books?id=0bAYl6d7hvkC



CHARLES C. RAWLINS (Member, IEEE) received the B.S. degree (Hons.) in electrical engineering from Montana Technological University with an interest in computer science. He is currently pursuing the Ph.D. degree in computer engineering with the Missouri University of Science and Technology. His research interests include the Internet of Things devices and security.

S. JAGANNATHAN (Fellow, IEEE) worked as the Site Director of the NSF Industry/University Cooperative Research Center on Intelligent Maintenance Systems. He is currently with the Missouri University of Science and Technology (former the University of Missouri-Rolla), where he is also a Rutledge-Emerson Distinguished Professor of electrical and computer engineering. His research interests include machine learning and cyber-physical system security, neural network control, prognostics/big data analytics, and autonomous systems/robotics.