A Theoretical Model Characterizing Tangle Evolution in IOTA Blockchain Network

Fengyang Guo, *Student Member, IEEE*, Xun Xiao, Artur Hecker, and Schahram Dustdar, *Fellow, IEEE*

***Abstract*—IOTA blockchain system is lightweight without heavy proof-of-work mining phases, which is considered a promising service platform of Internet of Things applications. IOTA organizes ledger data in a directed acyclic graph (DAG), called *Tangle*, rather a chain structure as in traditional blockchains. With arriving messages, IOTA tangle grows in a special way, as multiple messages can be attached to the tangle at different locations in parallel. Hence, the network dynamics of an operational IOTA system would justify a thorough study, which is currently unexplored in the literature. In this article, we present the first theoretical modeling for the evolving IOTA tangle based on stochastic analysis. After analyzing snapshots of the real-world IOTA ledger data, our key finding suggests that IOTA tangle follows a rather atypical double Pareto Lognormal (dPLN) degree distribution. In contrast, typical power-law and exponential distributions do not accurately reflect the fact. For model parameter estimation, we further realize that using generic optimization solvers cannot yield quality fitting results. Thus, we design an alternative algorithm based on expectation–maximization (EM) framework. We evaluate the proposed model and fitting algorithm with official data provided by the IOTA Foundation. Quantitative comparisons confirm the fitting quality of our proposed model and algorithm. The whole analysis reveals a deeper understanding of the internal mechanism of the IOTA network.**

***Index Terms*—Expectation–maximization (EM) algorithm, IOTA blockchain, network dynamics, parameter estimation, theoretical modeling.**

I. Introduction

In 2016, IOTA Foundation (IF)—The official IOTA development and operation consortium—launched a new type of blockchain system, called IOTA. Rather than using a chain topology, the ledger of the IOTA system is organized as a directed acyclic graph (DAG), called *Tangle*, wherein every vertex represents a single message record (either a value transaction or a data payload) [1]. In IOTA, every participating node holds a copy of the tangle, responsible for committing incoming messages independently and forming consensus in a distributed manner among participants.

Manuscript received 22 April 2022; revised 5 August 2022; accepted 5 September 2022. Date of publication 19 September 2022; date of current version 6 January 2023. *(Corresponding author: Xun Xiao.)*

Fengyang Guo is with the Munich Research Center, Huawei Technologies, 80992 Munich, Germany, and also with the Distributed Systems Group, TU Wien, 1040 Vienna, Austria (e-mail: fengyang.guo@dsg.tuwien.ac.at).

Xun Xiao and Artur Hecker are with the Munich Research Center, Huawei Technologies, 80992 Munich, Germany (e-mail: drxiaoxun@gmail.com; artur.hecker@huawei.com).

Schahram Dustdar is with the Distributed Systems Group, TU Wien, 1040 Vienna, Austria (e-mail: dustdar@dsg.tuwien.ac.at).

Digital Object Identifier 10.1109/JIOT.2022.3207513

IOTA is lightweight and feeless without heavy proof-of-work mining phases. Hence, IOTA is considered suitable for a decentralized service platform of IoT applications, characteristic to a massive exchange of instant, typically tiny information. Although IOTA recently has gained high research popularity, most of the works focus on its empirical studies [2], protocol extensions [3], and applications [4], [5]. Nevertheless, we are not aware of any graph- and network-theoretical analysis on the ledger tangle evolution, especially for the operational IOTA network deployed in the real world. Undoubtedly, theoretically understanding how the tangle evolves is important to capture the core mechanism underlying IOTA network dynamics.

Due to the particular structure and the distributed consensus mechanism, the evolving ledger tangle in IOTA would justify such a network dynamics analysis. Specifically, when a new message arrives, it is attached as a new vertex, with directed edges pointing to the existing vertices. As in a DAG, many candidate vertices exist, the location the vertex will attach to is determined by a *selection algorithm*—part of the IOTA distributed consensus protocol. In IOTA, a directed edge represents an approval from the source vertex to the referred vertex. The key principle is to encourage new messages approving yet unapproved vertices, so-called *tip* vertices, whose in-degree is zero. The ledger tangle chronologically grows in such a manner over time. More details about IOTA's mechanism will be introduced in a later section.

In this article, we study how the tangle topology evolves in IOTA. Particularly, we try to answer, if there exists a theoretical network model governing this process; and if so, what a degree distribution would best represent it. Several typical network models, such as the random graph model [6], [7], [8] and Barabási's preferential attachment (PA) model [9] have gained a wide recognition, after they were shown to have good fitting properties for many naturally occurring processes. However, during our initial investigation, we realized that the existing network models did not fit well with the observed data sets generated from the IOTA network. The key reasons are explained as follows.

First, IOTA tangle grows with a batch arrival mode, in which multiple new vertices may come and every new vertex may add more than one new edge. The key fact behind this is that a copy of the tangle exists on every participating node; and every node can attach new messages to its local tangle independently; thus, after individual ledger copies are merged, multiple messages and edges can appear to one vertex at burst. Existing models, however, often assumed a single vertex sequential arrival mode, where only one new vertex is added at each time. Most even further assumed a single edge addition. Hence, it is inaccurate to simplify IOTA tangle developing with such a simple way. The network modeling for IOTA's tangle evolution has a different growing behavior.

Second, vertex and edge addition in IOTA tangle do not follow a similar logic of the PA model (or of its variants). In the PA model, a vertex is randomly selected proportional to (or modeled as a function with explicit form of) its degree value. In IOTA, however, vertex selection is a much more complicated process, which involves evaluating other existing vertices (i.e., historical ledger records) in a subtangle topology. Clearly, it cannot be attributed to a simple vertex property (e.g., a degree value) characteristic to the PA model. The above two key features of the formation process makes IOTA's tangle evolution show a unique behavior, which was not studied in the scope of network modeling research.

In addition to deriving the model, another technical problem that is equally important is parameter estimation. Unfortunately, the issue we encounter is that a generic optimization solver (typically gradient-descent (GD)-based methods) cannot give a satisfactory parameter estimation for the derived model. We then develop a dedicated algorithm based on expectation–maximization (EM) framework [10] as an alternative. In summary, our main contributions are listed as follows.

1) Using stochastic analysis, we characterize operational IOTA network dynamics with an stochastic differential equation (SDE) that approximates vertex degree evolution over time.
2) Based on the analytical solution of the modeled SDE, we derive that IOTA tangle dynamics follow a double Pareto Lognormal (dPLN) distribution.
3) For parameter estimation, we develop an EM-based algorithm, which can provide more reliable and higher quality fitting results than using generic GD-based solvers; our source code is also published to benefit the community.[1]
4) We evaluate the fitting quality of the derived model and proposed algorithm with realistic snapshot data generated from IOTA *mainnet*,[2] and the results justify our findings.

To the best of our knowledge, in short, this work is the first trying to model the tangle evolution in IOTA, whose network dynamics behaviors combine a batch vertex arrival and a complex attachment process. However, modeling of such a unique network dynamic, meanwhile providing a more efficient fitting algorithm, were not seen so far in the past literature.

The remainder of this article is organized as follows. Related work is reviewed in Section II and IOTA preliminary is introduced as a background in Section III; Section IV presents our model and Section V introduces our model fitting algorithm; after that, Section VI shows the evaluation results; Section VII concludes this article.

[1]Github: https://github.com/goldrooster/IOTA-Tangle-Evolution-Model.

[2]IOTA mainnet is an IOTA network deployed on Internet by IF and ledger data were created by anonymous users over the world.

II. Related Work

With the high popularity of blockchain, there are many survey works on research activities of blockchain and IoT systems, such as [15], [16], [17], [18], [19], and [20]. Most of them focused on inventing/proposing consensus protocols, improving system performances, applications of blockchain for IoT services, and security issues. For example, as an application presented, Dhall *et al.* [16] provided a solution to utilize blockchain platform for reducing fake information propagation on social media/messaging systems; additionally, Hayyolalam *et al.* [20] provided a comprehensive review on using edge-assisted solutions for healthcare systems based on IoT devices. Nevertheless, few of them mentioned the theoretical analysis research about blockchain systems; and even fewer had an eye on the theoretical modeling on the tangle dynamics of the IOTA blockchain network.

A. Theoretical Work in IOTA (DAG-Based) Blockchain

Though there are very limited relevant theoretical works, we observed several attempts on analytical performance modeling of DAG-based blockchain systems. Kusmierz *et al.* [11], [12] built a rule-based discrete- and continuous-time models for IOTA, in order to build a relationship of the number of tip vertices and the vertices' cumulative weights over time. In [14], it theoretically analyzed the probability of being left-behind of confirmation of a message in IOTA tangle by simulating the IOTA protocol. Popov *et al.* [13] analyzed the message attachment behavior of the IOTA network and proved that there exists a Nash equilibrium, revealing that selfish nodes will cost more than nonselfish nodes. Our interest in this work targets to a different goal, which aims to theoretically model how the tangle topology evolves and what a degree distribution could best represent it.

B. Network Graph Models

Network graph modeling is an active research area. The famous growing/evolving network model (i.e., PA model) was proposed in [9]. In this model, new vertices prefer to attach on existing vertices with higher degrees, which models a common phenomenon where the rich becomes richer. The authors proved that the graph will become a scale-free network (i.e., a power-law (PL) distribution) at the end.

As a variant of the PA model, cyclic PA (CPA) was introduced in [21]. The attachment probability of the CPA model depends on the shortest path from the node to all other nodes. The author used this model to analyze the real-world network, such as Facebook and company directors. They showed that the CPA model can provide more flexibility to model the networks in the real life. Furthermore, in [22], another PA model's variant is proposed to model a phenomenon where a vertex acquires a new vertex depending on the density of its local area in a graph. It also shows that a PL distribution appears. The work in [23] introduced a burst model based on the PA model. However, this burst model only extends the PA model with a random vertex mutation behavior where a new vertex randomly duplicates to multiple ones at its original point.

TABLE I
COMPARISON OVERVIEW OF THE SELECTED LITERATURE

Subject / Reference	*IOTA System Related*				*Network Dynamics Modeling*			
	KPI Analysis	System Enhancement	Application/ Survey	Tangle Evolution	Vertex Batch Arrival	Complex Vertex Attachment	Fitting Algorithm	Modeling Theory
[11]–[14]	✓	✗	✗	✗	N/A			
[15]–[20]	✗	✓	✓	✗	N/A			
[9], [21]–[30]	N/A				✗	✗	✓	✓
[31], [32]	N/A				✓	✗	✗	✓
Ours	✗	✗	✗	✓	✓	✓	✓	✗

Recently, Pandey and Adhikari [24], based on the PA model, proposed a network reconstruction model for structural reconstruction of scale-free real networks. Liu *et al.* [25] used two jointly evolving graphs, i.e., K-partite graph and generated graph, to characterize intertype and intratype interactions among nodes, respectively, and establishes the evolving process of them. Its underlying assumption is also based on the PA model where higher degree vertices are preferred when the graphs evolve. Tajeuna *et al.* [26] modeled the community structure changes of social networks to facilitate predictions of critical events. It applied a sliding window analysis from which it developed a model that simultaneously exploits an autoregressive model and survival analysis techniques. Qiao *et al.* [27] proposed a variant of stochastic block models in order to characterize clusters or community structures of network data with PL degree features.

In summary, although the PA model and its variants provide decent modeling for a large number of evolving networks, to our problem, IOTA tangle evolution cannot be simplified like that due to its special burst arrival mode and the vertex selection mechanism, explained before.

In reality, many phenomena do not follow the logic of a PA model. The degree distributions of their topology are also not PL/Exponential (Expon) distributions. For instance, the authors, respectively, showed that the file size [28], the city size [29], and mobile call graphs [30] follow dPLN distributions [31]. Comparing to them, the main challenge of this work is that IOTA is a distributed network system and its network dynamics are implicit. Hence, a correct modeling with rigorous verification is needed. In fact, initial results in [33] already realized that the PL model does not fit the empirical data of IOTA mainnet.

C. Modeling Tools

Technically, there are two main approaches used for network modeling: 1) master equation system (MES) and 2) SDE approaches.

The MES approach uses the Markov chain theory to derive a set of differential equations that describe the transition of the probability distribution of an interested system state [34]. For example, Wing *et al.* [23] used this approach and presented a generalized framework to unify different evolution stages of complex networks. Its network growing strategy is similar to the PA model. The advantages of using the MES approach are its accuracy and flexibility, while its disadvantage is that modeling with MES may render the problem intractable. We will see that our problem drops into this case. This also explains why most of the existing works only covered simplified network behaviors.

The SDE approach describes a dynamic system in a probabilistic view by introducing stochastic terms in modeling [32]. Reed and Jorgensen [31] explained the genesis of dPLN distribution with such an approach. The advantage of using the SDE approach is its simplicity. It can help to simplify the original problem to an easier case and get a decent approximation. Its disadvantage is that sometime it may oversimplify the problems thus lose its original properties. We will see that our problem can benefit from the SDE approach, where after approximation, the original problem becomes solvable without sacrificing any key property.

D. Summary

In order to give a more concise and direct comparison, we compose Table I to highlight the aimed subjects of our work and the mismatch of the past literature.

First, this work mainly targets to theoretically understand and model the tangle ledger dynamics in IOTA. This differs to the works analyzing the performance KPIs as in Section II-A (refer the 1st row of Table I), and this work is also orthogonal to those work exploring IOTA's applications, such as [15], [16], [17], [18], [19], and [20] (refer the 2nd row of Table I).

Second, in the scope of network dynamics modeling, due to the unique features of IOTA, modeling its tangle evolution faces the challenges combining both batch vertex arrivals and a complex vertex attachment process. This differs to the existing network modeling works (as discussed in Section II-B) often with oversimplified assumptions, which cannot accurately reflect the behaviors of IOTA's ledger evolution over time (refer the 3rd row of Table I).

Third, we utilize the long-standing mathematical tools in literature (refer the 4th row of Table I). However, the theory does not tell how to translate IOTA's tangle evolution into the mathematical language. This is one of our key contributions in this work, where not only a full modeling with stochastic analysis will be provided, but also a customized fitting algorithm will be designed for parameter estimation of the derived model.

In short, only our work (refer the last row of Table I) covers all the highlighted subjects and problem features shaded with yellow color in Table I comparing with the past literature.

III. IOTA PRELIMINARY

For improving the readability of this article, here we briefly review IOTA network's two key mechanisms as follows.

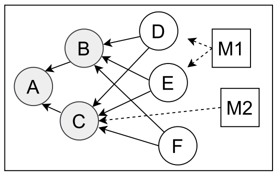

Fig. 1. Ledger tangle evolution in IOTA ("A"–"F" are existing messages in the tangle; "M1" and "M2" are new messages).

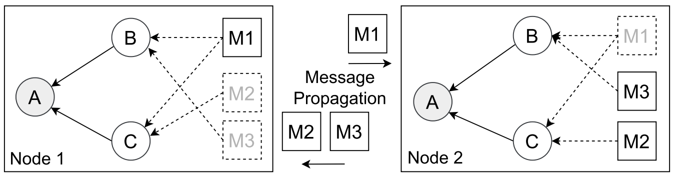

Fig. 2. Tangle consolidation via message propagation.

A. Message Attachment

A standard way is to attach new messages (e.g., "M1" or "M2" as shown in Fig. 1) to vertices with in-degree 0 (e.g., "D," "E" and "F" in Fig. 1), i.e., *tips*. In IOTA's syntax, messages at tips are unapproved yet; thus, their confidence is low. In contrast, the more attachments a message gets (through direct and indirect approvals), the more confidence the message owns. The motivation behind this is to create a mechanism, encouraging every node approving other nodes' messages. In IOTA, every node should run a tip selection algorithm (TSA) module dedicated for tip selection. So far, IOTA released two protocol versions, i.e., IOTA 1.0 (released in June 2016) and 2.0 (released in June 2021). Although TSAs are different in the two versions, the purpose of the two different protocol versions are always the same. They all aim to identify the most appropriate tips where a new message can attach to. Their main features are compared as follows.

IOTA protocol 1.0's TSA is based on weighted DAG random walk. Jumping on a DAG along with reversed edge directions always stops at one tip. Additionally, the outcomes are biased, where some tips get higher preferences after random walk. In general, the weighted random walk gives one way to select the most "appropriate" tip(s). IOTA protocol 2.0's TSA proposed a more efficient mechanism by introducing many auxiliary modules in order to prepare extra information in advance to facilitate the tip selection. For example, every node now has to prepare and update unspent transaction output (UTXO) graphs ahead to maintain individual groups of nonconflict messages (transactions). Thus, a nonconflict tip can be identified faster for a new message. Therefore, expensive random walk as in IOTA 1.0 can be avoided.

However, no matter which protocol version is used, this cannot change the fact of conflicting messages in the ledger. A tip vertex that will be selected by weighted DAG random walk (IOTA 1.0) will probably be selected by the new selection mechanism (IOTA 2.0) and *vice versa*. In short, IOTA 1.0 and 2.0 differ with efficiency and internal auxiliary data structures. The change or outcome resulted to the evolution of the tangle topology is similar.

In addition to the standard IOTA protocols, IOTA does not prohibit using other principles for message attachment. For example, a node can attach a message to a nontip vertex (e.g., "M2" attaching to "C" in Fig. 1). Attaching to nontip messages makes a message easier get approved later on. Selfish nodes may utilize a nonstandard way to launch parasite chain attacks [35]. Such abnormal behaviors reflect self-interest behaviors and were indeed observed in IOTA mainnet online.

Note that since the distributed consensus, as an intermediate phase, is already integrated within message attachment when every node makes its vertex selection decision, it does not fundamentally change the manner of network dynamics.

B. Tangle Consolidation

Every node in IOTA has a local ledger view in form of a tangle. Tangle consolidation aims to propagate messages across the network so that tangle copies are merged into one (e.g., "M1" as "Node 1" s newly attached message is propagated to "Node 2"; and "M2" and "M3" as "Node 2" s newly attached messages are propagated to Node 1 in Fig. 2). If a forwarded message exists in the local tangle, the node ignores it but still forwards to other neighbors (except to the expedient); otherwise, a node saves the message and checks, if the referenced messages can be found in its tangle; if so, the node merges the message; if missing, the message is suspended, until all previous messages are found from neighbors by sending message requests recursively.

In summary, the key mechanism behind IOTA is that new messages always approve old messages. Attached messages will be propagated across the whole network thus ledgers converge and consensus opinions are formed. For more details, readers are referred to the full IOTA protocol specifications [1], [36].

IV. IOTA Tangle Network Dynamics

A. Modeling

Our modeling consists of two components: 1) a batch attachment model and 2) a state transition model.

1) Batch Attachment Model: As explained, messages in the IOTA network arrive in batches, because different nodes may independently select the same message to attach new messages to their own tangle copies. Hence, a vertex can get multiple referencing messages after consolidation. A typical random process to model this phenomenon is a multivariate $\texttt{Poisson}$ process $\texttt{Poi}(\lambda_t, \lambda_m)$, where one or more messages arrive with an average rate λ_t and an average size λ_m.

Denoting all new messages arriving at time t as a set M_t, IOTA requires each new message (vertex) to select $s \in [2, 8]$ existing vertices[3] in the tangle for approval. This would create maximally $s \cdot |M_t|$ new directed edges, denoted as an edge set

[3]The parameter s will be configured to a *fixed* value in a deployment (e.g., in IOTA mainnet, $s = 2$). Whenever a new participating node joins in, this parameter s will be shared or synchronized with other nodes. Hence, every new message can attach to at most 2 different existing vertices. An example with $s = 2$ is shown in Figs. 1 and 2, every vertex has at most two egress edges.

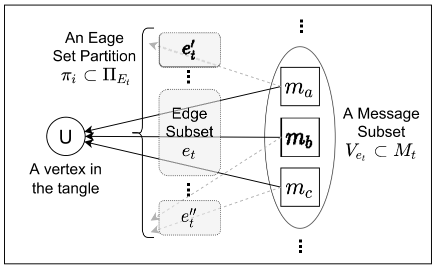
An Eage Set Partition $\pi_i \subset \Pi_{E_t}$
e_t'
Edge Subset e_t
e_t''
U
A vertex in the tangle
m_a
m_b
m_c
A Message Subset $V_{e_t} \subset M_t$

Fig. 3. New messages and edge set partition at time t.

E_t. We further denote the subset of new messages selecting the same vertex as $V_{e_t} \in M_t$; these messages will introduce a subset of new edges $e_t \subset E_t$ to the selected vertex. Fig. 3 illustrates such an example, where three new messages (m_a, m_b, and m_c) select the same vertex U and bring three new edges to U. Note that a new message can have its new edges in multiple edge subsets at the same time (e.g., m_a has its second edge to e_t').

The total new message set M_t splits into several subsets (such as V_{e_t}), which results in a partition π_i on the whole edge set E_t split into many edge subsets (such as e_t' and e_t''). We denote all possible partitions on E_t caused by M_t's attachment as Π_{E_t}. Obviously, there are many possible ways to partition E_t, depending on where the new messages in M_t are exactly attached/clustered. In the following analysis, it is sufficient to analyze the outcome of an edge partition $\pi_i \subset \Pi_{E_t}$, because only newly attached edges increase the vertex degree.

2) State Transition Model: The total new message set M_t always changes degrees of selected vertices and likewise, the size of the tangle. Hence, the *system state* of the tangle can be described with a 2-D state vector $\langle k, n\rangle$, representing a state of vertex degree type k given the current tangle size n. There are three possible state transitions involving the system state $\langle k, n\rangle$, which are illustrated in Fig. 4 and elaborated as follows.

1) *Transition* $\mathfrak{g}_{\tau_1}$*:* Suppose that the current tangle size is $n - |V_{e_t}|$, an edge subset e_t attaches to a type of vertex whose original degree is $k - |e_t|$. It changes the vertex's degree type to k and increases the tangle size to n, thus transiting into the state $\langle k, n\rangle$

$$\left\langle k-|e_t|, n-|V_{e_t}|\right\rangle \overset{\mathfrak{g}_{\tau_1}}{\rightarrow} \langle k, n\rangle$$

2) *Transition* $\mathfrak{g}_{\tau_2}$*:* Suppose that the current tangle size is $n - |V_{e_t}|$, an edge subset e_t attaches to any type of vertices whose degree $\forall k_i \neq k - |e_t|$. It keeps vertices whose degree type is already k untouched while only increases the tangle size to n, thus also transiting into the state $\langle k, n\rangle$

$$\left\langle k, n-|V_{e_t}|\right\rangle \overset{\mathfrak{g}_{\tau_2}}{\rightarrow} \langle k, n\rangle$$

3) *Transition* $\mathfrak{g}_{\tau_3}$*:* Suppose that the current tangle size is n, an edge subset e_t attaches to any type of vertices possibly with any degree. If the selected vertex has its degree type $k_j == k$, this changes the vertex's degree type to $k + |e_t|$ (i.e., the right horizontal transition in Fig. 4); if the selected vertex has its degree type $\forall k_j \neq k$, this changes the vertex's degree type to another $k' \neq k + |e_t|$ (i.e., the upper right transition in Fig. 4). In either case, the tangle size increases to $n + |V_{e_t}|$. This makes the state jump out of the state $\langle k, n\rangle$

$$\langle k, n\rangle \overset{\mathfrak{g}_{\tau_3}}{\rightarrow} \left\langle k', n+|V_{e_t}|\right\rangle.$$

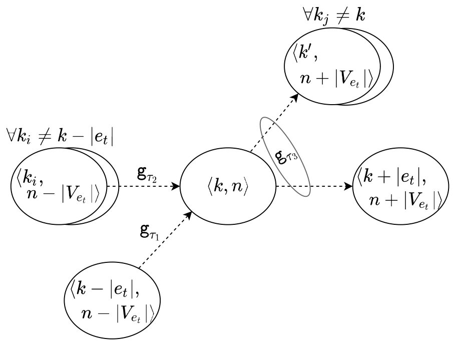
$\forall k_j \neq k$
$\langle k', n+|V_{e_t}|\rangle$
$\forall k_i \neq k-|e_t|$
$\langle k_i, n-|V_{e_t}|\rangle$
$\mathfrak{g}_{\tau_2}$
$\langle k, n\rangle$
$\mathfrak{g}_{\tau_3}$
$\langle k+|e_t|, n+|V_{e_t}|\rangle$
$\mathfrak{g}_{\tau_1}$
$\langle k-|e_t|, n-|V_{e_t}|\rangle$

Fig. 4. State transition graph. (For the case $k = 0$, $\mathfrak{g}_{\tau_1}$ and $\mathfrak{g}_{\tau_2}$ do not exist; for the case $k = K$, $\mathfrak{g}_{\tau_3}$ does not exist.)

The state transition graph in Fig. 4 is a 2-D Markov chain. The evolution of the probability distribution of the system state $p_{k,n}(t)$ follows the Chapman–Kolmogorov equation [37] below:

$$\frac{dp_{k,n}(t)}{dt} = \mathtt{Poi}(\lambda_t, \lambda_m) \cdot \sum_{\pi_i \subset \Pi_{E_t}} \sum_{e_t \subset \pi_i} \left(\underbrace{\mathfrak{g}_{\tau_1}(k-|e_t|) \cdot p_{\tau_1}(t)}_{\text{Gain term 1}} + \underbrace{\sum_{\forall k_i \neq k} \mathfrak{g}_{\tau_2}(k_i) \cdot p_{\tau_2}(t)}_{\text{Gain term 2}} - \underbrace{\sum_{\forall k_j} \mathfrak{g}_{\tau_3}(k_j) \cdot p_{k,n}(t)}_{\text{Loss term}} \right) \quad (1)$$

where each $\mathfrak{g}_{\tau_i}(\cdot)$ defines the generalized transition rate. Such a differential equation system is called a MES in statistical mechanics [38], formulating the probability distribution change of a system state by aggregating all possible "Gain" and "Loss" transitions. If only the standard way of attachment is preferred, we can limit the state transitions to tip vertices with an indicator function $\mathbb{1}(k == 0)$.

Unfortunately, the MES in (1) does not permit an analytical solution thus hinders our further analysis, because: 1) the MES enumerates over set partitions Π_{e_t} and further over subsets (i.e., every edge subset e_t) of every possible edge set partition $\pi_i \subset \Pi_{E_t}$. Both of them are set permutations thus do not have explicit expressions and 2) transition rate functions $\mathfrak{g}_{\tau_i}(\cdot)$ do not possess an analytical form either, as it represents a vertex selection algorithm involving subtangle operations. Clearly, a new approach is needed for the problem.

B. Modeling Approximation

Instead of analyzing the detailed transitions between degree types, our idea is to analyze a *macro* effect resulted from the

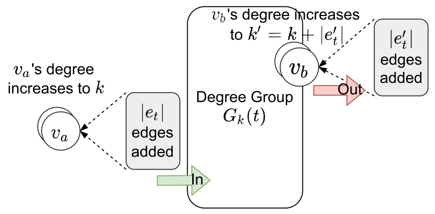
v_b's degree increases to $k' = k + |e'_t|$
$|e'_t|$ edges added
v_a's degree increases to k
$|e_t|$ edges added
Degree Group $G_k(t)$
v_a
v_b
In
Out

Fig. 5. Dynamics of DGS $s_k(t)$ at time t.

new message set M_t. Recalling from Section IV-A, an edge set partition $\pi_i \subset \Pi_{E_t}$ simultaneously leads to degree changes on multiple vertices, this motivates us to model the size change of a degree group $G_k(t)$ in a tangle, denoted as the degree group size (DGS) $s_k(t)$. A degree group $G_k(t)$ represents vertices all having the same degree k. In IOTA, DGS $s_k(t)$ may dynamically change due to the following two events, which are illustrated in Fig. 5 and explained as follows.

1) *"In"-Event:* DGS $s_k(t)$ may increase, because there could be a vertex v_a, whose original degree is less than k, but an edge subset e_t makes v_a's degree increase to k with adding $|e_t|$ new edges;
2) *"Out"-Event:* DGS $s_k(t)$ may decrease, because there could be a vertex v_b, whose original degree is already k, but another edge subset e'_t makes v_b degree increase to k' with adding $|e'_t|$ new edges.

In IOTA, *"In"-Event* can happen to any vertex whose degree value is between $[0, k)$, and *"Out"-Event* can happen to any vertex whose degree value is equal to k, thus covering all degree groups in a tangle.

From a probabilistic point of view, the macro effect of "In"- and "Out"-Events to a degree group G_k can be roughly viewed as a Brownian motion [39], because whether or not the DGS $s_k(t)$ will eventually change is uncertain, which is driven by the random vertex selections from participating nodes. Mathematically, the rate of the changing ratio can be either positive, zero, or negative during an infinitesimal period. Such a stochastic process can be formulated with an SDE of $s_k(t)$ as follows:

$$\frac{ds_k(t)}{s_k(t)} = \omega(t)dt + \sigma(t)dB(t) \tag{2}$$

where $\omega(t)$ is a growing rate coefficient, and $\sigma(t)$ is a fluctuation coefficient of random behaviors modeled as a Brownian motion $dB(t)$. Note that, we did not use the absolute change of $s_k(t)$, because the variation of $s_k(t)$ might be negative due to the Brownian motion term, which would conflict with the reality, as size cannot be negative. A benefit of using a relative ratio here is that it guarantees $s_k(t)$ a nonnegative value.

C. Degree Distribution

Based on the SDE modeling, we sketch the main theoretical results regarding the stationary distribution of s_k. Since related properties are well studied, interested readers are kindly referred to [31] for the concrete steps to derive the results below.

First, the SDE in (2) takes the form of geometric Brownian motion (GBM). If $\omega(t)$ and $\sigma(t)$ are independent of time t, this SDE is analytically solvable, and we have

$$s_k(t) = s_k(0) \cdot \exp\left\{ \underbrace{\left(\omega - \frac{\sigma^2}{2}\right)}_{\mu \text{ term}} t + \sigma B_t \right\} \tag{3}$$

where the μ term is referred in the following equations.

Second, the probability density function (PDF) of DGS $s_k(t)$ at any observation time t follows a lognormal (LN) distribution:

$$f_{\mathsf{LN}}(x) = \frac{1}{\sigma\sqrt{2\pi}x} \exp\left\{ \frac{-(\log x - \mu)^2}{2\sigma^2} \right\}. \tag{4}$$

Additionally, if the observation time t is exponentially distributed as $p_T(t) = \xi e^{-\lambda t}$, the PDF of $s_k(t)$ follows a dPLN distribution as follows:

$$\begin{aligned} f_{\mathsf{dPLN}}(x) = \frac{\alpha\beta}{\alpha+\beta}\Bigg[& x^{-\alpha-1}A(\alpha)\Phi\left(\frac{\log x - \mu - \alpha\sigma^2}{\sigma}\right) \\ & + x^{\beta-1}A(-\beta)\Phi^c\left(\frac{\log x - \mu + \beta\sigma^2}{\sigma}\right)\Bigg] \end{aligned} \tag{5}$$

where $A(z) = \exp(z\mu + (z^2\sigma^2/2))$, $\Phi(\cdot)$ is the cumulative distribution function (CDF) of a standard normal distribution, and $\Phi^c(\cdot)$ is the complementary CDF of $\Phi(\cdot)$. Although the form of dPLN distribution in (5) looks complicated, it can be interpreted as a multiplicative process of LN quantities over exponentially distributed observation time t.

To our problem, the interpretation is that the DGS $s_k(t)$ grows along with the tangle over time t and the stoppage time t is assumed exponentially distributed. Importantly, the PDF in (5) tells what the probability density the size of a certain degree group $G_k(t)$ will be. After normalized with the total tangle size n, it represents exactly the *degree distribution* of a tangle that we target.

V. Parameter Estimation

A. Problem Formulation

The PDF of a dPLN distribution can be converted to a more friendly form—normal-Laplace (nLP) distribution—by substituting $y = \log x$

$$\begin{aligned} f_{\mathsf{nLP}}(y) = \frac{\alpha\beta}{\alpha+\beta}\Phi\left(\frac{y-\mu}{\sigma}\right)\Bigg[& R\left(\alpha\sigma - \frac{(y-\mu)}{\sigma}\right) \\ & + R\left(\beta\sigma + \frac{(y-\mu)}{\sigma}\right)\Bigg] \end{aligned} \tag{6}$$

where $R(\cdot) = ([1 - \Phi^c(\cdot)]/\Phi(\cdot))$ is Mills' ratio/survival function. The model parameter $\boldsymbol{\theta}$ is $[\alpha, \beta, \mu, \sigma^2]$ for both dPLN and nLP distributions. In the following sections, we use the nLP distribution in (6) for parameter estimation due to its simplicity.

Denoting the observed data (i.e., the observed degree distribution of a tangle) as $\boldsymbol{\mathcal{Y}}$, the log-likelihood is written as

$$\ell_{\mathsf{nLP}}(\boldsymbol{\theta}; \boldsymbol{\mathcal{Y}}) = \sum_{i=1}^{n} \log f_{\mathsf{nLP}}(\boldsymbol{\theta}; y_i). \tag{7}$$

A corresponding maximization likelihood estimation (MLE) problem is

$$\boldsymbol{\theta}^* \leftarrow \arg\max_{\boldsymbol{\theta}} \ell_{\mathsf{nLP}}(\boldsymbol{\theta}; \boldsymbol{\mathcal{Y}}). \tag{8}$$

The problem in (8) is usually not a concave (convex) problem due to the sum of a series of log-PDF terms. Therefore, the rest of this article focuses on the parameter estimation for the derived model, especially after we realize that in our trials generic optimization solvers cannot provide quality estimation results.

B. Main Idea

According to the result in [40], the visible/observed random variable Y of an nLP distribution can be considered a sum of two invisible/latent variables Z and W (i.e., $Y = Z + W$), following `Normal` distribution $f_Z(\mu, \sigma^2)$ and `Skewed-Laplace` distribution $f_W(\alpha, \beta)$, respectively.

Based on this feature, it is possible to construct an EM algorithm [10]. An EM algorithm moves to a maximized likelihood in iterations with the help of an augmented likelihood function of *complete* data by introducing auxiliary latent variables. Such an augmented likelihood function usually enables a simplification to the original likelihood function. Specifically, the simplified version calculates a set of expectation quantities of the augmented latent variables. This, in turn, eliminates the introduced latent variables after the expectation operation; in addition, that simplified version usually becomes a linear function of the unknown parameters, much easier for optimization.

The key benefits of an EM algorithm are: 1) neither a gradient nor Hessian matrix is needed, unlike generic optimization techniques such as Newton–Raphson methods and 2) iteration steps usually enjoy closed forms, thus quite efficient for computation. Nevertheless, a known obstacle of adopting EM framework is that no generic way exists to transform an MLE problem automatically into a form suitable in the EM framework. Always, a case-by-case design/transformation is needed, for which we will develop upon next.

C. dPLN *EM Algorithm*

The PDF of an nLP distribution with the visible random variable Y can be considered the marginal PDF of a joint distribution $f_{Y,Z}(\cdot)$ integrating over an introduced latent variable z. Hence, the likelihood in (7) is extended as

$$\ell_{\mathsf{nLP}}(\boldsymbol{\theta}; \boldsymbol{\mathcal{Y}}) = \sum_{i=1}^{n} \log \int f_{Y,Z}(y_i, z; \boldsymbol{\theta})\, dz. \tag{9}$$

The following result gives a lower bound of $\ell_{\mathsf{nLP}}(\boldsymbol{\theta}; \boldsymbol{\mathcal{Y}})$.

Theorem 1: A lower bound $Q(\boldsymbol{\theta})$ of the likelihood $\ell_{\mathsf{nLP}}(\boldsymbol{\theta}; \boldsymbol{\mathcal{Y}})$ in (9) is

$$\ell_{\mathsf{nLP}}(\boldsymbol{\theta}; \boldsymbol{\mathcal{Y}}) \geq Q(\boldsymbol{\theta}) \overset{\text{def.}}{=} \sum_{i=1}^{n} \mathsf{E}\big[\log f_{Y,Z}(y_i, z, \boldsymbol{\theta})\big] \tag{10}$$

where $\mathsf{E}[\,\cdot\,]$ is the expectation over an arbitrary distribution $g(z)$ of Z.

Proof: See Appendix A. ■

Theorem 1 says that we can consider to maximize $Q(\cdot)$ instead of $\ell_{\mathsf{nLP}}(\cdot)$ in (9). The question is how to find a proper $g(z)$.

Since $g(z)$ used in $Q(\cdot)$ can be arbitrary, it is convenient to use the conditional probability $f_{Z|Y=y_i}(\cdot)$ as $g_i(z)$, which represents, how likely z will be in terms of an observed data point $y_i \in \boldsymbol{\mathcal{Y}}$ with a specified parameter $\boldsymbol{\theta}^{(s)}$. This gives us an explicit $g_i(z)$ as follows:

$$\begin{aligned} g_i(z) &\overset{\text{def.}}{=} f_{Z|Y=y_i}\left(z; \boldsymbol{\theta}^{(s)}\right) \\ &= \frac{f_{Y,Z}\left(y_i, z; \boldsymbol{\theta}^{(s)}\right)}{f_Y\left(y_i; \boldsymbol{\theta}^{(s)}\right)} = \frac{f_Z\left(z; \boldsymbol{\theta}^{(s)}\right) f_W\left(y_i - z; \boldsymbol{\theta}^{(s)}\right)}{p_{\mathsf{nLP}}\left(y_i; \boldsymbol{\theta}^{(s)}\right)}. \end{aligned} \tag{11}$$

Note that $g_i(z)$'s exact form in (11) is completely given since $\boldsymbol{\theta}^{(s)}$ has a specific value and all PDFs are known to us.

With $g_i(z)$, the lower bound $Q(\cdot)$ in Theorem 1 also gets an explicit form as follows (see Appendix B for its derivation):

$$\begin{aligned} Q\left(\boldsymbol{\theta}; \boldsymbol{\theta}^{(s)}\right) &= n \log\left(\frac{1}{\sqrt{2\pi\sigma^2}}\right) - \frac{n\mu^2}{2\tau^2} + n \log\left(\frac{\alpha\beta}{\alpha + \beta}\right) \\ &\quad + \frac{\mu}{\sigma^2} \sum_{i=1}^{n} \underbrace{\mathsf{E}[z_i]}_{(\sharp 1)} - \frac{1}{2\sigma^2} \sum_{i=1}^{n} \underbrace{\mathsf{E}\left[z_i^2\right]}_{(\sharp 2)} \\ &\quad + \beta \sum_{i=1}^{n} \underbrace{\mathsf{E}\left[y_i - z\right]_{-\infty}^{y_i}}_{(\sharp 3)} - \alpha \sum_{i=1}^{n} \underbrace{\mathsf{E}\left[y_i - z\right]_{y_i}^{+\infty}}_{(\sharp 4)} \end{aligned} \tag{12}$$

where $\boldsymbol{\theta}^{(s)}$ is absorbed in $g_i(z)$, which is only used when calculating the four $\mathsf{E}[\cdot]$ quantities ($\sharp$1)-($\sharp$4) with every y_i. Although $Q(\boldsymbol{\theta}; \boldsymbol{\theta}^{(s)})$ looks complicated, its structure is much simpler than $\ell_{\mathsf{nLP}}(\cdot)$ in (9). This is the augmented likelihood function with complete data Z and $Y - Z$ (i.e., W) mentioned before. Let us review the two important features of $Q(\boldsymbol{\theta}; \boldsymbol{\theta}^{(s)})$ as follows.

1) If the four summation terms with the four $\mathsf{E}[\cdot]$ quantities ($\sharp$1)–($\sharp$4) are treated as coefficients, $Q(\boldsymbol{\theta}; \boldsymbol{\theta}^{(s)})$ only contains seven terms, much simpler than (9) with n terms (i.e., the number of data points).
2) $Q(\boldsymbol{\theta}; \boldsymbol{\theta}^{(s)})$ is (almost) a linear function of elements $[\alpha, \beta, \mu, \sigma^2]$ of $\boldsymbol{\theta}$ after the values of the four $\mathsf{E}[\,\cdot\,]$ quantities ($\sharp$1)–($\sharp$4) are determined, then much easier for optimization.

These two features match our initial expectations. More importantly, these two features also provide the algorithmic procedures of our dedicated EM algorithm.

Feature 1 defines an "E-Step" to calculate the four $\mathsf{E}[\,\cdot\,]$ quantities (so as the summation terms) by assigning $\boldsymbol{\theta}^{(s)}$ a specific value, starting with an initial guess $\boldsymbol{\theta}^{(0)}$. When calculating the four $\mathsf{E}[\,\cdot\,]$ quantities, the introduced latent variable z is thus eliminated with expectation operations. Since summation terms become coefficients, $Q(\cdot)$ reduces to a linear form of parameter $\boldsymbol{\theta}$. The closed forms to calculate the four $\mathsf{E}[\,\cdot\,]$ quantities are provided in Appendix C. $\boldsymbol{\theta}^{(s)}$ will be repeatedly updated with a new value in an "M-Step" below.

Feature 2 defines the M-Step to optimize the $Q(\cdot)$ function. In this step, only the model parameter $\boldsymbol{\theta}$ is treated as a variable,

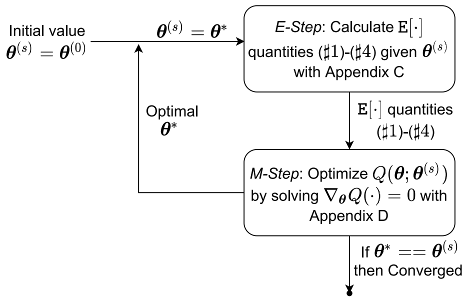
Initial value $\boldsymbol{\theta}^{(s)} = \boldsymbol{\theta}^{(0)}$
$\boldsymbol{\theta}^{(s)} = \boldsymbol{\theta}^{*}$
E-Step: Calculate $\mathbb{E}[\cdot]$ quantities (♯1)-(♯4) given $\boldsymbol{\theta}^{(s)}$ with Appendix C
Optimal $\boldsymbol{\theta}^{*}$
$\mathbb{E}[\cdot]$ quantities (♯1)-(♯4)
M-Step: Optimize $Q(\boldsymbol{\theta};\boldsymbol{\theta}^{(s)})$ by solving $\nabla_{\boldsymbol{\theta}} Q(\cdot) = 0$ with Appendix D
If $\boldsymbol{\theta}^{*} == \boldsymbol{\theta}^{(s)}$ then Converged

Fig. 6. Algorithm 1: dPLN model parameter estimation.

because the four $\mathsf{E}[\,\cdot\,]$ quantities, which were already fixed in the E-Step, have become coefficients. To yield an optimal $\boldsymbol{\theta}^*$ in this iteration, we maximize $Q(\cdot)$ by taking partial derivatives in terms of $\boldsymbol{\theta}$ and solving an equation system $\nabla_{\boldsymbol{\theta}} Q(\cdot) = 0$. As $Q(\cdot)$ is linear to $\boldsymbol{\theta}$, this equation system has an analytical solution. Hence, the optimal $\boldsymbol{\theta}^*$ can be directly calculated with the closed form given in Appendix D.

Iterating between the two steps defines the body of our parameter estimation algorithm. As said, the old value $\boldsymbol{\theta}^{(s)}$ is updated with $\boldsymbol{\theta}^*$ derived in the M-Step, becoming $\boldsymbol{\theta}^{(s+1)}$. Based on $\boldsymbol{\theta}^{(s+1)}$, the E-Step recalculates the four new $\mathsf{E}[\,\cdot\,]$ quantities, and this once again modifies $Q(\cdot)$. Then, the M-Step is repeated with the new $Q(\cdot)$ function, and it gives another $\boldsymbol{\theta}^*$ used to replace $\boldsymbol{\theta}^{(s+1)}$, and so on so forth. Once the optimized parameter in the M-Step does not change anymore or fulfill a predefined threshold, then a (local) optimal estimate is obtained. A diagram of the algorithm is shown in Fig. 6.

D. Remarks

In the literature, EM-based methods are often used for estimating parameters with a mixed `Gaussian` model. Differently, the nLP (dPLN) model is rather a mixture of `Normal` and `Skewed-Laplace` distributions, which cannot trivially reuse the existing algorithms developed for other models. How a dPLN model can be estimated under an EM framework is partly discussed in [31] and [41]. Unfortunately, none of them shows evidences of executable implementations and reports comprehensive performance evaluations; some of them even contain errors after our examination. In contrast, we not only provide explicit mathematical derivations, publish our source code but also compare its fitting performance against using existing optimization solvers.

VI. Results

Considering the nature of the technical contributions, this work neither modified any existing system nor proposed any new system. Instead, this work theoretically analyzed the dynamics of a real-world system—IOTA network—by deriving a *new model* and designing a *fitting algorithm*. Therefore, the rationale of our evaluation plan is to directly evaluate: 1) whether the derive model (dPLN) does fit better with the observed data and 2) whether the proposed fitting algorithm does provide a better parameter estimation.

A. Settings

1) Data Set Information: Our evaluation uses real-world historical data generated from IOTA mainnet. Some information about the used data are introduced as follows.

First, IOTA mainnet was launched on 11 July 2016. IOTA mainnet regularly maintains a network scale of more than 400 active nodes running the IOTA protocol on the Internet.

Second, the used data were officially published by IF.[4] The whole data set contains message records from two archive periods: Period I is November 2016–June 2019 (generating 96 tangle snapshots) and Period II is April 2020–August 2020 (generating 16 tangle snapshots). Except these two periods, we do not see any newer official data set available.

Third, tangle snapshots vary in size, which is mainly determined by the message arrival rate and the number of active participating nodes during the period of the tangle snapshot was created. The former information can be calculated by dividing the number of messages over the period length. However, for the latter information, it is difficult to restore because IF does not make the history record of participating nodes public.

Last but not least, the tangle size refers the total number of messages contained in an archived ledger *snapshot*. Snapshots are periodically created and archived by IF every two or three months since June 2016. After a snapshot is created, IOTA mainnet resets and starts over a new empty ledger. Hence, the tangle size is not additive between two snapshots. For a vertex's original degree, it particularly refers to its in-degree value in this work, which equals the total number of direct references from other messages.

An overview of some properties about the tangle snapshots is provided in Table II. Averagely, a tangle contains approximately 1.7 millions of messages.

2) Data Set Preparation: To prepare the reference data, given a snapshot, the in-degree of a vertex (message) can be calculated by summarizing the total number of messages that reference to the considered message. After that, for each tangle snapshot, we first count the DGS s_k of every degree group G_k, and then we calculate its proportion $y_k = (s_k/n)$ in the tangle, giving the observed degree distribution of a tangle. This is the reference data $\mathcal{Y} = \{G_k, y_k\}_{k=1}^{K}$ for parameter estimation of one tangle snapshot, where each degree group G_k corresponds s_k data points (vertices).

3) Candidate Models: The candidate models for comparisons are listed in Table III. Besides the dPLN model, the other three candidates are chosen because they are widely acknowledged network models representative for many natural phenomena. The complexity of candidate models increases from the simplest ones (i.e., PL and Expon) to complicated ones (i.e., LN and dPLN). The number of their model parameters also increases from $1 \rightarrow 2 \rightarrow 4$. Table III also lists the solution methods of MLE for each candidate model.

4) Fitting Quality Metric: We use root mean squared logarithmic error (rMSLE) to quantify the fitting quality. Its classic

[4]Online archive: https://dbfiles.iota.org/.

TABLE II
STATISTICAL PROPERTIES OF THE IOTA MAINNET SNAPSHOT DATA SETS

Year	Month Period	Tangle Snapshot Num.	Tangle Index	Total Size (10^6)	Average Size (10^6)	Daily Message Num. (10^3)	Daily Variance (10^8)
2016	10-12	2	♯1, ♯2	0.15	0.08	2.2	0.02
2017	01-03	2	♯3, ♯4	2.5	1.3	16.6	46.61
	04-06	1	♯5	3.5	3.5	57.8	6.80
	07-09	1	♯6	2.1	2.1	45.8	3.31
	10-12	4	♯7-♯10	4.9	1.2	42.7	71.20
2018	01-03	7	♯11-♯17	5.2	0.7	48.7	69.53
	04-06	10	♯18-♯27	19.3	1.9	104.5	200.59
	07-09	28	♯28-♯55	24.7	0.9	158.8	285.60
	10-12	22	♯56-♯77	50.6	2.3	237.8	613.3
2019	01-03	19	♯78-♯96	37.8	2.0	147.3	257.3
2020	04-08	16	♯97-♯112	40.0	2.5	220.2	836.1
Total		112	-	191.3	1.7	124.1	347.1

TABLE III
LIST OF CANDIDATE MODELS (ζ AND ξ: NORMALIZATION CONSTANTS)

Model	PDF	Model Parameters	MLE Solution Method
PL	$\zeta x^{-\gamma}$	γ	Closed-Form
Expon	$\xi e^{-\lambda x}$	λ	Closed-Form
LN	Eq. (4)	$[\mu, \sigma^2]$	Closed-Form
dPLN	Eq. (5)	$[\alpha, \beta, \mu, \sigma^2]$	Algorithm 1 (Fig. 6)

definition is given as follows:

$$\text{rMSLE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(\log y_i - \log \hat{y}_i\right)^2} \tag{13}$$

where n is the total number of observed data points, and $\hat{y}_i$ is the predicted value of y_i. rMSLE can be considered a *relative error* of the predicted and actual values. The smallest rMSLE is zero when every predicated value $\hat{y}_i$ is equal to its observed value y_i.

The key reason to choose rMSLE is because it is a unit/scale-independent metric. Note that in our problem, the probabilities (proportions) of degree groups may differ significantly in scale. In this situation, unit-dependent measures like the mean absolute error (MAE) and the root mean squared error (RMSE) turn out to be unsuitable, because the absolute error distances from the prediction on data points with smaller proportions will be insignificant. With those metrics, since only dominant features matter, this will falsely reflect the fitting quality. rMSLE solves this issue by taking a log-difference/relative ratio so that it becomes unit independent.

For our evaluation, we can get a more succinct form, as the log-difference term for every data point (vertex) in one degree group G_k is identical. Therefore, we only need to calculate once the log-difference for all data points in every degree group G_k weighted by its proportion y_k as follows:

$$\text{rMSLE} = \sqrt{\sum_{k=1}^{K} y_k\left(\log y_k - \log \hat{y}_k\right)^2}. \tag{14}$$

The benefits of using (14) are as follows:

First, it speeds up the calculation, because the summation in (14) only has K terms (i.e., K observed degree groups), much less than the summation of n terms in (13). In our data sets, n means millions of vertices while K means only hundreds types of degree groups.

Second, if we remove the root and square operators in (14), rMSLE recovers to Kullback–Leibler (KL) divergence[5] that is widely used to measure the divergence between two probability distributions. Therefore, one metric acts as two. More importantly, rMSLE removes a cumbersome constraint in using KL divergence where both y_k and $\hat{y}_k$ $\forall k = 1, \ldots, K$ must be perfect probability distributions (i.e., the sum of probability values equal to 1). Practically, since the predicted value $\hat{y}_k$ will be sampled from a continuous PDF of a candidate model, this constraint is not always met, leading an invalid KL divergence, thus making the evaluation failed. Then, it is inevitable to introduce extra techniques to discretize every candidate model's PDF. However, this may cause uncertainty to our evaluation, as it is unknown yet which discretization technique fits the best for our case. Instead, rMSLE measures in the similar way as KL divergence does but free of such a constraint.

B. Model Selection

1) Quantitative Comparison: In this part, we examine the fitting quality of the four candidate models. We rank the candidate models in a decreasing order in terms of their fitting qualities. The model getting rMSLE closest to the optimal value 0 (highlighted with yellow bar) is put at the top.

The first comparison is on overall interval where all degree groups are considered, shown in Fig. 7. In this comparison, we show both the CDFs of rMSLEs of the four candidate models and a statistical boxplot on top. We can observe that dPLN model achieves the least average rMSLE below 0.2 with a concentrated variance distribution (shown as the short green boxplot). Besides, the LN model (in blue) ranks as the second best but its mean rMSLE (around 0.55) is already worse two

[5]$\text{KL}(y_k \,||\, \hat{y}_k) = \sum_{k=1}^{K} y_k \log(y_k/\hat{y}_k)$, where y_k and $\hat{y}_k$ $\forall k = 1, \ldots, K$ are two discrete probability distributions.

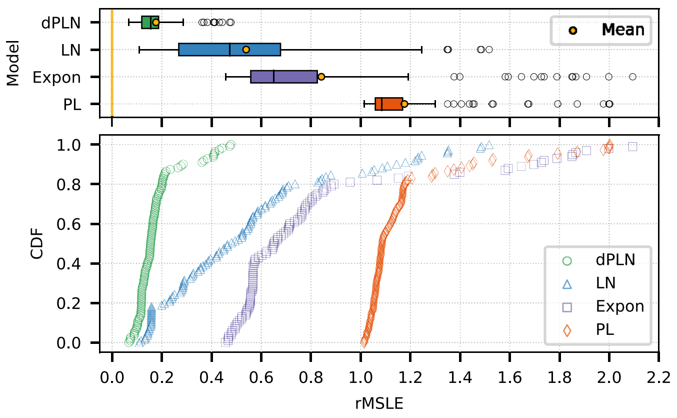

Fig. 7. rMSLE comparison on overall interval (degree group $G_k \in [1, \max]$, EM's $\boldsymbol{\theta}^{(0)} = [0.1, 0.1, 0.1, 0.1]$).

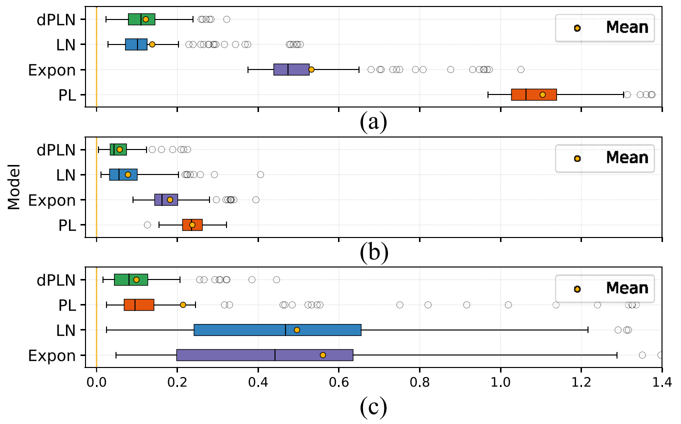

Fig. 8. rMSLE comparison on segmented intervals (EM's $\boldsymbol{\theta}^{(0)} = [0.1, 0.1, 0.1, 0.1]$).

times more than dPLN model's; in addition, the LN model has the largest variance of rMSLE. Furthermore, neither Expon nor PL models (in purple and orange, respectively) seems a correct model to explain the degree distribution of the realistic tangles generated in the real world, where both of them have much worse rMSLE, especially the PL model.

The large variances of rMSLE values show the instability of the three compared candidate models when they fit the observed in-degree distributions. Actually, their mean values of rMSLE are also worse than the dPLN model's performance. In contrast, the variance of dPLN model's rMSLE is concentrated and much less than the variances of the other three models. This again justifies that the performance of the dPLN model is not only better on average but also relatively more stable than the other three models can do.

The second comparison is on segmented intervals as shown in Fig. 8. We compare three separate intervals that split the data points into three parts: 1) header; 2) middle; and 3) rear parts. Specifically, the header part contains vertices in degree groups $G_k \in [1, 2]$, which often roughly occupy 45%; the middle part contains vertices in degree groups $G_k \in [3, 5]$, which occupy another 30%–45%; and the rear part is all the rest kinds of vertices (i.e., in degree groups $G_k \in [6, \max]$).

Specifically, in the header part [shown in Fig. 8(a)], the ranking is the same as in the overall interval but the performances of dPLN and LN models become closer, though dPLN model's rMSLE is slightly smaller. This implies that for vertices with degree values in $[1, 2]$, both models fit well and achieve small errors. In the middle part [shown in Fig. 8(b)], the ranking is also the same but every candidate model gets a smaller rMSLE, meaning that all candidate models fit better to the distribution of vertices with degree values between $[3, 5]$. Particularly, dPLN and LN models even get their rMSLE smaller than 0.1. In the rear part [shown in Fig. 8(c)], the ranking becomes different, where the second-best model is now PL, the third place is LN, and the last one is Expon. In fact, LN and Expon show much worse performances when fitting to the degree distribution of vertices with large degree values.

Note that there are deeper reasons behind the observation where every model performs well in the middle part. We explain as follows.

1) The head part (vertices with in-degree between $[1, 2]$) is also a majority ($\sim 45\%$). However, due to the shape of their distributions bending down, the other three models cannot cover the both head and middle parts. Specifically, PL model as a straight line cannot bend obviously (the worst), Expon model can slightly bend (the second worst), and LN model can do more (the third worst). Only the dPLN model can nicely balance the two parts. This is why we see distinct performances at the head part in Fig. 8(a).
2) The rear part (vertices with in-degree ≥ 6) is not a majority ($< 5\%$). For the same reason, not all the other three models are able to cover this part. The order of the fitting performance changes, the Expon model becomes the worst, the LN model becomes the second worst, and the PL model (as a straight line to fit the right tail) becomes the third worst. Still, dPLN performs the best in Fig. 8(c).

This observation can be seen more directly in our graphical comparison next.

In summary, we can clearly observe that on any considered interval dPLN model ranks always the best and its rMSLEs are relatively stable.

2) Graphical Comparison: We then give a graphical comparison. This helps readers to understand how the four candidate models fit the reference data in a visual way. Here, in Fig. 9 we pick three tangle instances, to which dPLN model yields its min, median, and max rMSLEs, respectively. The subplots therein particularly zoom in on the fittings of degree groups $G_k \in [1, 3]$.

In the graphical fitting comparison, we can visually observe that dPLN's fittings (the green solid curves) indeed stick much closer to the actual distribution of the data points (gray empty circles). In contrast, we can see that the other three models are far away either to the header part (such as Expon and PL models) or to the rear part (such as the LN model).

In the zoom-in subplots (on upper right corners), we can see that only dPLN and LN models can fit the data points

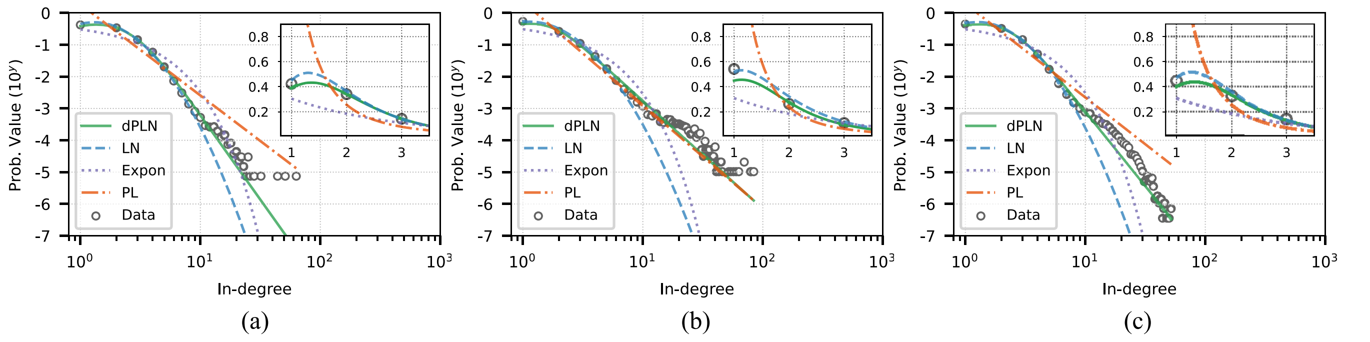

Fig. 9. Graphical fitting comparison of candidate models. (a) Tangle♯34 (rMSLE$_{\text{min}}$: 0.06). (b) Tangle♯39 (rMSLE$_{\text{Median}}$: 0.15). (c) Tangle♯64 (rMSLE$_{\text{max}}$: 1.08).

in the header part (curves in green and blue, respectively). They are slightly different, where the LN model tends to overestimate while the dPLN model relatively underestimates the data points. Nevertheless, neither Expon nor PL model performs reasonably in the header part fitting, where the Expon model largely underestimates (purple curves) and the PL model significantly overestimates (orange curves) the data points.

The key factor making the dPLN model better than the other models is that it can not only characterize dominant features like majority vertices with degrees in [1, 3] (as LN model), but also reflect special features like existences of high degree vertices (as the PL model does), which is unique to the real-world tangles. Generally, we can conclude that the dPLN model can explain much better the observed degree distributions of tangle data generated in the IOTA mainnet. This confirms our theoretical modeling for IOTA network dynamics.

C. Fitting Algorithm Comparison

We then evaluate the performance of our algorithm in Fig. 6 named "EM" against "BFGS" [42]. The BFGS algorithm is a typical example of GD-based methods already implemented in the `Python.scipy` package, considered a default optimization Python library. Both algorithms aim to find the optimal solution for the MLE problem defined in (8).

We set the maximum iteration number equal to 2000 for both algorithms. We set the convergence threshold to 10^{-4} for BFGS. Particularly, we set the convergence threshold to our algorithm as follows:

$$\max\left\{\Delta_s \middle| \Delta_s = \sqrt{\left(\boldsymbol{\theta}^{(s+1)} - \boldsymbol{\theta}^{(s)}\right)^2}\ \forall s \in W\right\} \leq 10^{-4} \quad (15)$$

where it requires the maximum norm of the difference in W consecutive $\boldsymbol{\theta}^{(s)}$ less than 10^{-4}. Note that our threshold is harsher than BFGS uses.

For both algorithms, we evaluated ten different initial $\boldsymbol{\theta}^{(0)}$ values generated with the following rules. We first fix (μ, σ^2) pair but triple the (α, β) pair from $0.1 \to 2.7$ (giving four initial values). Then, we fix (α, β) pair but triple (μ, σ^2) pair from $0.1 \to 2.7$ (giving another 3 initial values). Last, we triple both pairs $0.1 \to 2.7$ (giving the last three initial values). This gives a set of initial values differ with several magnitudes.

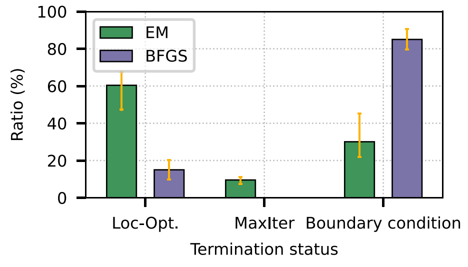

Fig. 10. Termination status comparison.

We did not use a random strategy to generate the initial values in order to guarantee the reproducibility of all presented results.

1) Algorithm Termination Status: There are three possible termination states of the two algorithm candidates as follows.

1) *"Loc-Opt.":* An algorithm terminates, because it fulfilled its convergence condition before reaching the configured maximum iteration number;
2) *"MaxIter":* An algorithm terminates, because it reached the configured maximum iteration number (i.e., 2000 here);
3) *"Boundary Condition":* An algorithm terminates, because some temporal solution violated some boundary conditions. In our case, this bound is that all elements of $\boldsymbol{\theta}$ should be positive.

It has to emphasize that all three termination statuses give parameter estimation solutions but with different fitting qualities. With the ten different initial values and 112 tangle snapshots, the termination states of the 112×10 times' fitting tests with the two algorithms are summarized in Fig. 10.

Our EM-based algorithm shows 60.36% of convergence rate, versus 15.18%, when using BFGS. In contrast, 84.82% times of using the BFGS solver triggered boundary condition versus 30.08%, when using our EM-based algorithm. Additionally, less than 10% of using our EM-based algorithm reached the maximum iteration number. Using the BFGS solver never reached the maximum iteration number, because we have seen that BFGS easily terminated due to boundary condition violation. This confirms that our algorithm has higher chances to get a local optimal solution, which is several magnitudes higher than using the existing solver—BFGS.

2) Fitting Quality: We again evaluate the two algorithms with rMSLE shown in Fig. 11. In all intervals, the

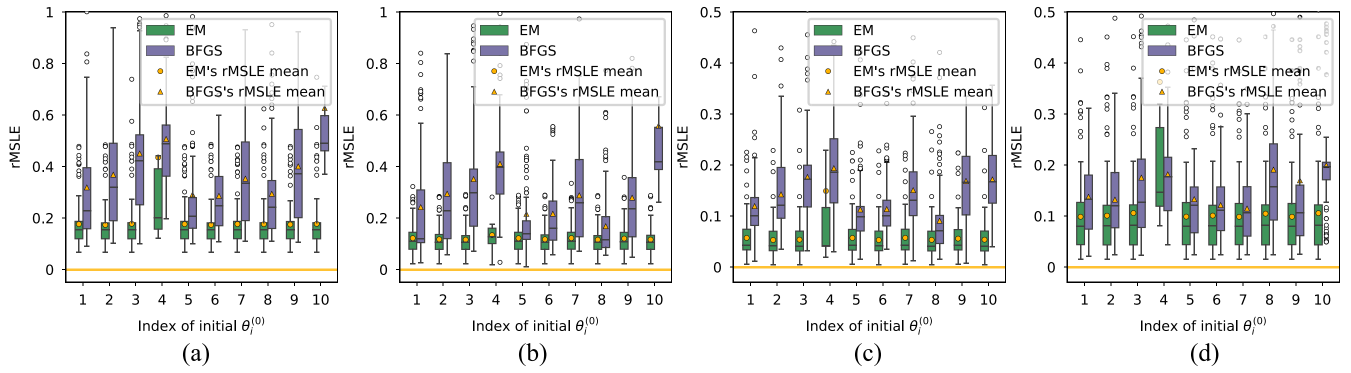

Fig. 11. rMSLE comparison of using EM and BFGS on (segmented) intervals. (a) Overall ($G_k \in [1, \max]$). (b) Header part ($G_k \in [1, 2]$). (c) Middle part ($G_k \in [3, 5]$). (d) Rear part ($G_k \in [6, \max]$).

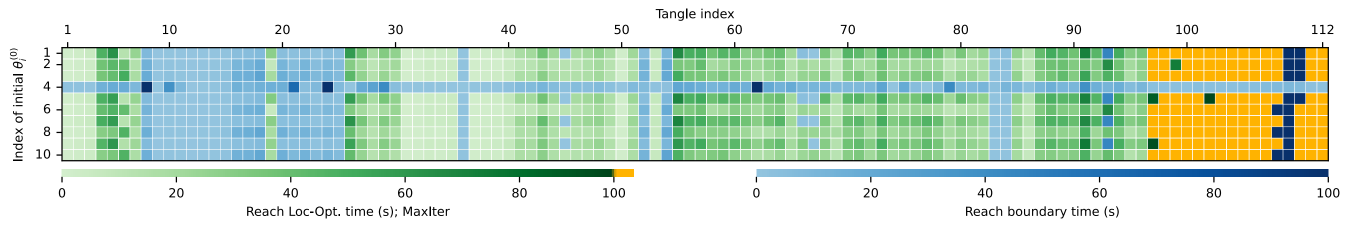

Fig. 12. Time to reach termination statuses in all estimation tests with the proposed EM-based algorithm.

performance of using BFGS is worse than using our EM-based algorithm. We can observe a larger variation of the rMSLE of the fitting results given by the BFGS solver. Instead, the rMSLEs of our EM-based algorithm are closer to the optimal value 0 and the variations are not only more consistent but also much smaller than BFGS has. With different initial values, we observe a similar result where our EM-based algorithm achieves a better fitting quality (i.e., smaller rMSLE) than using BFGS.

The results from the termination status and fitting quality suggest that for parameter estimation of a dPLN model, an EM-based algorithm is recommended. It also shows that it is difficult for GD-based methods to handle optimization problems within a high-dimensional space (our problem has four elements in $\boldsymbol{\theta}$ thus it is 4-D). In fact, we had also tested Nelder–Mead (downhill simplex) method as a third candidate, which was chosen as an opponent that is without calculating gradient/Hessian matrix [43]. Since its fitting quality was even much worse than BFGS can provide, it is less valuable to report its results here.

3) Fitting Time: Finally, we report the time cost spent on fitting every tangle with our EM-based algorithm in Fig. 12. The heatmap plot indicates individual execution time to reach one of the three termination states in every tangle fitting test (1120 times in total). Specifically, blocks in green, blue, and yellow colors represent termination states of “Loc-Opt.,” “Boundary Cond.” violation, and “MaxIter,” respectively. The proportions of the three color blocks correspond to the summary result in Fig. 10.

First, we observe that when estimating parameters for tangles ♯8–♯18, ♯20–♯25, ♯36, ♯73, ♯74, and ♯109, our EM-based algorithm triggered the boundary conditions with any initial value. This is a known outcome when the given data are not *perfectly* dPLN distributed [31]. Second, we observe that $\boldsymbol{\theta}_4^{(0)}$ seems to be a challenging initial value. With this particular initial value, our EM-based algorithm did not terminate at the Loc-Opt. status. Actually, for BFGS, with $\boldsymbol{\theta}_4^{(0)}$, it also yielded poorer rMSLEs. It needs further investigation to check whether such an initial value is at a location blocked to a local optimal in the solution space. Except $\boldsymbol{\theta}_4^{(0)}$, our algorithm performs coherently, where we observe not only similar termination states but also similar execution times. Third, we observe that our algorithm reaches the MaxIter termination status when fitting tangles ♯97–♯112. This can be because of our the harsh convergence threshold defined in (15).

One important reason why the fitting time may vary among different tangles is because of the tangle size. If the tangle size n is large, the number of terms in (9) grows as well. As a result, a fitting algorithm may take longer time in each iteration when evaluating (9). In contrast, a different vertex’s degree will not immediately influence the fitting performance. Instead, the *population distribution* of vertices with a certain degree will statistically influence the fitting quality of a model. The nature of the fitting data (i.e., the observed in-degree distribution) determines whether or not a model can fit well.

The execution time falls in the range with an upper limit of 100s (except for those reaching MaxIter status). From the color distribution, execution time seems more relevant to the size of the tangles (i.e., the number of vertices). Instead, it seems rather less dependent to the initial values because no matter which initial value is used, the variation of execution time across the entire data sets are similar. On average, a tangle has a million vertices, we can expect approximately 40–60 s with our EM-based algorithm.

D. Statistics of Estimated Parameters

Finally, we provide a summary of the estimated parameters for the tangle data sets of IOTA mainnet, shown in Table IV.

TABLE IV
STATISTICS OF ESTIMATED MODEL PARAMETERS
($\boldsymbol{\theta}^{(0)} = [0.1, 0.1, 0.1, 0.1]$)

	α	β	μ	σ^2
Mean	4.638	2.002	0.497	0.096
Variance	22.325	0.800	0.039	0.002
Median	2.762	1.631	0.579	0.101
1/4 Quartile	2.076	1.526	0.432	0.071
3/4 Quartile	3.766	1.881	0.624	0.131

As part of our future work, with the estimated parameters, the derived network model gives a new way to design an IOTA simulator. Specifically, it can initialize a dPLN distribution, then sample from the distribution, and rewire sampled vertices to construct a tangle topology. This can largely improve the efficiency because simulating heavy network protocols is avoided completely.

VII. CONCLUSION

In this article, we developed a theoretical model for IOTA network dynamics with stochastic analysis. The key finding is that realistic tangles follow a dPLN distribution, which is not as usual belief, such as PL and Expon distributions. We designed a dedicated model estimation algorithm that can provide more reliable and quality solutions, which overcomes the deficiencies of using the existing solvers. Based on the real-world official data sets, the evaluation results confirmed our finding where our proposed model outperforms the existing popular network models; the evaluation results also justified the performance of our proposed parameter estimation algorithm. The whole work also gave a deeper understanding on the internal mechanisms of the IOTA network.

APPENDIX A
SKETCH PROOF OF THEOREM 1

Given an arbitrary PDF $g(z)$, the log-likelihood $\ell_{\mathsf{nLP}}(\boldsymbol{\theta}; \boldsymbol{\mathcal{Y}})$ can be transformed as follows:

$$\begin{aligned}\ell_{\mathsf{nLP}}(\boldsymbol{\theta}; \boldsymbol{\mathcal{Y}}) &= \sum_{i=1}^{n} \log \int f_{Y,Z}(y_i, z; \boldsymbol{\theta})\, dz \\ &= \sum_{i=1}^{n} \log \int \left\{\frac{f_{Y,Z}(y_i, z; \boldsymbol{\theta})}{g(z)}\right\} g(z)\, dz.\end{aligned}$$

According to Jensen's inequality (i.e., $\log \mathsf{E}[X] \geq \mathsf{E}[\log X]$), after moving "log" into the integration, we have

$$\begin{aligned}\ell_{\mathsf{nLP}}(\boldsymbol{\theta}; \boldsymbol{\mathcal{Y}}) &\geq \sum_{i=1}^{n} \int \log\left\{\frac{f_{Y,Z}(y_i, z; \boldsymbol{\theta})}{g(z)}\right\} g(z)\, dz \\ &= \sum_{i=1}^{n} \int \log\{f_{Y,Z}(y_i, z; \boldsymbol{\theta})\} g(z)\, dz \\ &\quad - \sum_{i=1}^{n} \int \log\{g(z)\} g(z)\, dz \\ &= \underbrace{\sum_{i=1}^{n} \mathsf{E}\big[\log f_{Y,Z}(y_i, z; \boldsymbol{\theta})\big]}_{Q(\boldsymbol{\theta})} \underbrace{- \sum_{i=1}^{n} \mathsf{E}\big[\log g(z)\big]}_{\text{Constant} \geq 0}.\end{aligned}$$

Since $-\mathsf{E}[\log g(z)] \geq 0$, $\ell_{\mathsf{nLP}}(\boldsymbol{\theta}; \boldsymbol{\mathcal{Y}})$ is at least no less than the $Q(\boldsymbol{\theta})$ part. This proves the result.

APPENDIX B
SIMPLIFICATION OF $Q(\boldsymbol{\theta})$

We directly substitute with all PDFs and $g_i(z)$. Note that $g_i(z)$ is the PDF with known parameters set to $\boldsymbol{\theta}^{(s)}$. Let $Q_i(\cdot) = \mathsf{E}[\log f_{Y,Z}(y_i, z; \boldsymbol{\theta})]$ for simplicity of notations

$$\begin{aligned}Q_i(\boldsymbol{\theta}) &= \mathsf{E}\big[\log f_{Y,Z}(y_i, z; \boldsymbol{\theta})\big] \\ &= \mathsf{E}\big[\log\{f_Z(z; \boldsymbol{\theta}) f_W(y_i - z; \boldsymbol{\theta})\}\big] \\ &= \mathsf{E}\big[\log f_Z(z; \boldsymbol{\theta}) + \log f_W(y_i - z; \boldsymbol{\theta})\big] \\ &= \mathsf{E}\big[\log f_Z(z; \boldsymbol{\theta})\big] + \mathsf{E}\big[\log f_W(y_i - z; \boldsymbol{\theta})\big] \\ &= \int \log\left[\frac{\exp\left(-\frac{(z-\mu)^2}{2\tau^2}\right)}{\sqrt{2\pi\tau^2}}\right] g_i(z)\, dz \\ &\quad + \int \log\left[\frac{\alpha\beta}{\alpha+\beta}\begin{cases} e^{\beta(y_i - z)}, & y_i - z \leq 0 \\ e^{-\alpha(y_i - z)}, & y_i - z > 0\end{cases}\right] g_i(z)\, dz.\end{aligned} \tag{16}$$

After integration and rearrangement, it becomes

$$\begin{aligned}Q(\boldsymbol{\theta}) &= \sum_{i=1}^{n} Q_i(\boldsymbol{\theta}) \\ &= n\log\left(\frac{1}{\sqrt{2\pi\sigma^2}}\right) - \frac{n\mu^2}{2\sigma^2} + n\log\left(\frac{\alpha\beta}{\alpha+\beta}\right) \\ &\quad + \frac{\mu}{\sigma^2}\sum_{i=1}^{n}\underbrace{\mathsf{E}[z_i]}_{(\sharp 1)} - \frac{1}{2\sigma^2}\sum_{i=1}^{n}\underbrace{\mathsf{E}\big[z_i^2\big]}_{(\sharp 2)} \\ &\quad + \beta\sum_{i=1}^{n}\underbrace{\mathsf{E}\big[y_i - z\big]_{-\infty}^{y_i}}_{(\sharp 3)} - \alpha\sum_{i=1}^{n}\underbrace{\mathsf{E}\big[y_i - z\big]_{y_i}^{+\infty}}_{(\sharp 4)}.\end{aligned}$$

This obtains the results.

APPENDIX C
CLOSED FORMS OF EXPECTATION QUANTITY IN E-STEP

Here, we directly give the calculations

$$\begin{aligned}\mathsf{E}[z_i] &= \int_{-\infty}^{\infty} z g_i(z) dz = \mu + \sigma^2\frac{\alpha R(p_i) - \beta R(q_i)}{R(p_i) + R(q_i)} \\ \mathsf{E}\big[z_i^2\big] &= \int_{-\infty}^{\infty} z^2 g_i(z) dz = \mu^2 + \sigma^2 - \frac{\sigma^2(p_i + q_i)}{R(p_i) + R(q_i)} \\ &\quad + \sigma^2\frac{(2\nu\alpha + \alpha^2\sigma^2)R(p_i) + (\beta^2\sigma^2 - 2\mu\beta)R(q_i)}{R(p_i) + R(q_i)} \\ \mathsf{E}\big[y_i - z\big]_{-\infty}^{y_i} &= \int_{-\infty}^{y_i}(y_i - z) g_i(z) dz = \frac{\sigma(1 - p_i R(p_i))}{R(q_i) + R(p_i)} \\ \mathsf{E}\big[y_i - z\big]_{y_i}^{+\infty} &= \int_{y_i}^{+\infty}(y_i - z) g_i(z) dz = \frac{\sigma(q_i R(q_i) - 1)}{R(q_i) + R(p_i)}\end{aligned} \tag{17}$$

where $p_i = \alpha\sigma - (y_i - \mu)/\sigma$, $q_i = \beta\sigma + (y_i - \mu)/\sigma$ and $R(\cdot)$ is also the Mills ratio.

APPENDIX D
CLOSED FORMS OF OPTIMAL PARAMETERS IN M-STEP

The optimal solutions of the parameters can be easily calculated by evaluating the partial derivative $\nabla_{\boldsymbol{\theta}} Q(\cdot)$ equal to zero, to which we have

$$\nabla_{\boldsymbol{\theta}} Q(\cdot) = 0$$
$$\Rightarrow \boldsymbol{\theta}^* = \begin{cases} \mu^* &= \frac{1}{n}\sum_{i=1}^{n} \mathsf{E}[z_i] \\ (\sigma^2)^* &= \frac{1}{n}\sum_{i=1}^{n}\left(\mu^* - 2\mu^*\mathsf{E}[z_i] + \mathsf{E}\left[z_i^2\right]\right) \\ \alpha^* &= \frac{1}{\mathsf{P}+\sqrt{\mathsf{PQ}}} \\ \beta^* &= \frac{1}{\mathsf{Q}+\sqrt{\mathsf{PQ}}} \end{cases} \tag{18}$$

where

$$\mathsf{P} = \frac{1}{n}\sum_{i=1}^{n} \mathsf{E}\left[y_i - z\right]_{y_i}^{+\infty}, \quad \mathsf{Q} = \mathsf{P} - \frac{1}{n}\sum_{i=1}^{n} \mathsf{E}\left[y_i - z\right]_{-\infty}^{y_i}$$

REFERENCES

[1] S. Popov, "The tangle," IOTA Found., Berlin Germany, WhitePaper, Feb. 2018.

[2] C. Fan, H. Khazaei, Y. Chen, and P. Musilek, "Towards a scalable DAG-based distributed ledger for smart communities," in *Proc. IEEE 5th World Forum Internet Things (WF-IoT)*, Limerick, Ireland, Apr. 2019, pp. 177–182.

[3] P. Perazzo, A. Arena, and G. Dini, "An analysis of routing attacks against IOTA Cryptocurrency," in *Proc. IEEE Int. Conf. Blockchain (Blockchain)*, 2020, pp. 517–524.

[4] P. C. Bartolomeu, E. Vieira, and J. Ferreira, "IOTA feasibility and perspectives for enabling vehicular applications," in *Proc. IEEE Globecom Workshops (GC Wkshps)*, Abu Dhabi, United Arab Emirates, Dec. 2018, pp. 1–7.

[5] X. Xiao, F. Guo, and A. Hecker, "A lightweight cross-domain proximity-based authentication method for IoT based on IOTA," in *Proc. IEEE Globecom Workshops (GC Wkshps*, Taipei, Taiwan, Dec. 2020, pp. 1–6.

[6] X. Zhang, C. Moore, and M. E. Newman, "Random graph models for dynamic networks," *Eur. Phys. J. B*, vol. 90, no. 10, pp. 1–14, 2017.

[7] W. Aiello, F. Chung, and L. Lu, "A random graph model for power law graphs," *Exp. Math.*, vol. 10, no. 1, pp. 53–66, 2001.

[8] D. Garlaschelli, "The weighted random graph model," *New J. Phys.*, vol. 11, no. 7, Art. 2009, no. 073005.

[9] A.-L. Barabási and R. Albert, "Emergence of scaling in random networks," *Science*, vol. 286, no. 5439, pp. 509–512, 1999.

[10] A. Dempster, N. Laird, and D. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *J. Roy. Statist. Soc.*, vol. 39, no. 1, pp. 1–38, 1977.

[11] B. Kusmierz, "The first glance at the simulation of the tangle: Discrete model," IOTA Found., Berlin Germany, WhitePaper, pp. 1–10, 2017.

[12] B. Kusmierz, P. Staupe, and A. Gal, "Extracting tangle properties in continuous time via large-scale simulations," IOTA Found., Berlin Germany, Rep. 2018–08, 2018.

[13] S. Popov, O. Saa, and P. Finardi, "Equilibria in the tangle," *Comput. Ind. Eng.*, vol. 136, pp. 160–172, Oct. 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0360835219304164

[14] B. Kusmierz and A. Gal, "Probability of being left behind and probability of becoming permanent tip in the tangle," IOTA Found., Berlin Germany, Rep. 2, 2020.

[15] M. A. Ferrag, M. Derdour, M. Mukherjee, A. Derhab, L. Maglaras, and H. Janicke, "Blockchain technologies for the Internet of Things: Research issues and challenges," *IEEE Internet Things J.*, vol. 6, no. 2, pp. 2188–2204, Apr. 2019.

[16] S. Dhall, A. D. Dwivedi, S. K. Pal, and G. Srivastava, "Blockchain-based framework for reducing fake or vicious news spread on social media/Messaging platforms," *ACM Trans. Asian Low-Resour. Lang. Inf. Process.*, vol. 21, no. 1, pp. 1–33, Jan. 2022. [Online]. Available: https://doi.org/10.1145/3467019

[17] H.-N. Dai, Z. Zheng, and Y. Zhang, "Blockchain for Internet of Things: A survey," *IEEE Internet Things J.*, vol. 6, no. 5, pp. 8076–8094, Oct. 2019.

[18] M. A. Ferrag and L. Shu, "The performance evaluation of blockchain-based security and privacy systems for the Internet of Things: A tutorial," *IEEE Internet Things J.*, vol. 8, no. 24, pp. 17236–17260, Dec. 2021.

[19] K. Peng, M. Li, H. Huang, C. Wang, S. Wan, and K.-K. R. Choo, "Security challenges and opportunities for smart contracts in Internet of Things: A survey," *IEEE Internet Things J.*, vol. 8, no. 15, pp. 12004–12020, Aug. 2021.

[20] V. Hayyolalam, M. Aloqaily, Ö. Özkasap, and M. Guizani, "Edge-assisted solutions for IoT-based connected healthcare systems: A literature review," *IEEE Internet Things J.*, vol. 9, no. 12, pp. 9419–9443, Jun. 2022.

[21] D. Kasthurirathna and M. Piraveenan, "Cyclic preferential attachment in complex networks," *Procedia Comput. Sci.*, vol. 18, pp. 2086–2094, 2013. [Online]. Available: https://doi.org/10.1016/j.procs.2013.05.378

[22] L.-N. Wang, J.-L. Guo, H.-X. Yang, and T. Zhou, "Local preferential attachment model for hierarchical networks," *Physica A Stat. Mech. Appl.*, vol. 388, no. 8, pp. 1713–1720, 2009.

[23] D. S. Wing *et al.*, "A unified framework for complex networks with degree trichotomy based on Markov chains," *Sci. Rep.*, vol. 7, no. 1, pp. 1–12, 2017.

[24] P. K. Pandey and B. Adhikari, "A parametric model approach for structural reconstruction of scale-free networks," *IEEE Trans. Knowl. Data Eng.*, vol. 29, no. 10, pp. 2072–2085, Oct. 2017.

[25] J. Liu, L. Fu, Y. Yao, X. Fu, X. Wang, and G. Chen, "Modeling, analysis and validation of evolving networks with hybrid interactions," *IEEE/ACM Trans. Netw.*, vol. 27, no. 1, pp. 126–142, Feb. 2019.

[26] E. G. Tajeuna, M. Bouguessa, and S. Wang, "Modeling and predicting community structure changes in time-evolving social networks," *IEEE Trans. Knowl. Data Eng.*, vol. 31, no. 6, pp. 1166–1180, Jun. 2019.

[27] M. Qiao, J. Yu, W. Bian, Q. Li, and D. Tao, "Adapting stochastic block models to power-law degree distributions," *IEEE Trans. Cybern.*, vol. 49, no. 2, pp. 626–637, Feb. 2019.

[28] M. Mitzenmacher, "Dynamic models for file sizes and double pareto distributions," *Internet Math.*, vol. 1, no. 3, pp. 305–333, 2004.

[29] K. Giesen, A. Zimmermann, and J. Suedekum, "The size distribution across all cities–Double Pareto lognormal strikes," *J. Urban Econ.*, vol. 68, no. 2, pp. 129–137, 2010.

[30] M. Seshadri, S. Machiraju, A. Sridharan, J. Bolot, C. Faloutsos, and J. Leskove, "Mobile call graphs: Beyond power-law and lognormal distributions," in *Proc. 14th ACM SIGKDD Int. Conf. Knowl. Discov. Data Min.*, 2008, pp. 596–604.

[31] W. J. Reed and M. Jorgensen, "The double Pareto-lognormal distribution—A new parametric model for size distributions," *Commun. Statist.-Theory Methods*, vol. 33, no. 8, pp. 1733–1753, 2004.

[32] N. Van Kampen, "Stochastic differential equations," *Phys. Rep.*, vol. 24, no. 3, pp. 171–228, 1976.

[33] F. Guo, X. Xiao, A. Hecker, and S. Dustdar, "Characterizing IOTA tangle with empirical data," in *Proc. IEEE Globecom*, Taipei, Taiwan, Dec. 2020, pp. 1–6.

[34] S. M. Ross, *Stochastic Processes*, vol. 2. New York, NY, USA: Wiley, 1996.

[35] A. Cullen, P. Ferraro, C. King, and R. Shorten, "On the resilience of DAG-based distributed ledgers in IoT applications," *IEEE Internet Things J.*, vol. 7, no. 8, pp. 7112–7122, Aug. 2020.

[36] S. Popov and W. J. Buchanan, "FPC-BI: Fast probabilistic consensus within byzantine infrastructures," *J. Parallel Distrib. Comput.*, vol. 147, pp. 77–86, Jan. 2021.

[37] J. Karush, "On the chapman-kolmogorov equation," *Ann. Math. Statist.*, vol. 32, no. 4, pp. 1333–1337, 1961.

[38] R. C. Tolman, *The Principles of Statistical Mechanics*. North Chelmsford, MA, USA: Courier Corporat., 1979.

[39] J. T. Lewis and J. V. Pulè, "Dynamical theories of Brownian motion," in *Proc. Int. Symp. Math. Problems Theor. Phys.*, 1975, pp. 516–519.

[40] W. J. Reed, *The Normal-Laplace Distribution and Its Relatives*. Boston, MA, USA: Birkhäuser Boston, 2006, pp. 61–74.

[41] E. Calderín-Ojeda, K. Fergusson, and X. Wu, "An EM algorithm for double-Pareto-lognormal generalized linear model applied to heavy-tailed insurance claims," *Risks*, vol. 5, no. 4, p. 60, 2017.

[42] C. Zhu, R. H. Byrd, P. Lu, and J. Nocedal, "Algorithm 778: L-BFGS-B: Fortran subroutines for large-scale bound-constrained optimization," *ACM Trans. Math. Softw.*, vol. 23, no. 4, pp. 550–560, Dec. 1997.

[43] J. A. Nelder and R. Mead, "A simplex method for function minimization," *Comput. J.*, vol. 7, no. 4, pp. 308–313, 1965.

Fengyang Guo (Student Member, IEEE) received the B.Sc. degree in communication engineering from Shandong Normal University, Jinan, China, in 2015, and the M.Sc. degree in computer science from RWTH Aachen University, Aachen, Germany, in 2019. He is currently pursuing the Ph.D. degree with Munich Research Center, Huawei Technologies, Munich, Germany.

He affiliates with the Distributed Systems Group, TU Wien, Vienna, Austria. His research interests include blockchain and computer network, more specifically focusing on the consensus mechanism of the blockchain based on the directed acyclic graph.

Xun Xiao received the B.Sc. and M.Sc. degrees in computer science from the South-Central University for Nationalities, Wuhan, China, in 2006 and 2009, respectively, and the Ph.D. degree in computer science from the City University of Hong Kong, Hong Kong, in 2012.

He is currently a Senior Researcher with Munich Research Center, Huawei Technologies, Munich, Germany. He was a Postdoctoral Researcher with the Max Planck Institute of Molecular Cell Biology and Genetics, Dresden, Germany, from 2012 to 2014. His research interests mainly focus on algorithms, networking, and computing in distributed systems.

Artur Hecker received the M.Sc. degree from the Universität Karlsruhe, Karlsruhe, Germany, in 2001, and the Ph.D. degree from ENST, Paris, France, in 2005.

He is currently the Director of Networking Research with the Advanced Wireless Technology Laboratory and an Expert with Huawei Munich Research Center, Munich, Germany. He was a full-time Associate Professor with Télécom ParisTech, Paris, from 2006 to 2013, where he was a Leader of the Security and Networking Research Group. Overall, he looks back at almost 20 years of entrepreneurial, academic, and industry experience in networks, systems, and system security.

Dr. Hecker has been serving as a Chair for the Vision and Societal Challenges Working Group of 6GIA since 2021.

Schahram Dustdar (Fellow, IEEE) received the M.Sc. and Ph.D. degrees from the University of Linz, Linz, Austria, in 1989 and 1992, respectively.

He is a Full Professor of Computer Science heading the Research Division of Distributed Systems with TU Wien, Vienna, Austria. He is the Co-Founder of an EdTech company in the U.S. (edorer.com) and SinoAus.net based in Nanjing, China, an R&D Lab focusing on AI and Edge Intelligence.

Dr. Dustdar is a recipient of multiple awards, such as the TCI Distinguished Service Award in 2021, the IEEE TCSVC Outstanding Leadership Award in 2018, the IEEE TCSC Award for Excellence in Scalable Computing in 2019, the ACM Distinguished Scientist in 2009, the ACM Distinguished Speaker in 2021, and the IBM Faculty Award in 2012. He is the Founding Co-Editor-in-Chief of the *ACM Transactions on Internet of Things* as well as the Editor-in-Chief of *Computing* (Springer). He is an Associate Editor of the IEEE TRANSACTIONS ON SERVICES COMPUTING, IEEE TRANSACTIONS ON CLOUD COMPUTING, *ACM Computing Surveys*, *ACM Transactions on the Web*, and *ACM Transactions on Internet Technology*, as well as on the editorial board of IEEE INTERNET COMPUTING and IEEE Computer. He is an Elected Member of the Academia Europaea: The Academy of Europe, where he is the Chairman of the Informatics Section, as well was as an Asia–Pacific Artificial Intelligence Association President in 2021 and Fellow in 2021. He was a Fellow of EAI and I2CICC in 2021.