Modeling of Distributed Ledgers: Challenges and Future Perspectives

Modeling of Distributed Ledgers: Challenges and Future Perspectives

Sergey Smetanin[1], Aleksandr Ometov[2], Niclas Kannengießer[3,4],
Benjamin Sturm[3], Mikhail Komarov[1], Ali Sunyaev[3]

[1] Faculty of Business and Management, National Research University Higher School of Economics, Moscow, Russia
[2] Faculty of Information Technology and Communication Sciences, Tampere University, Tampere, Finland
[3] Institute of Applied Informatics and Formal Description Methods, Karlsruhe Institute of Technology, Karlsruhe, Germany
[4] Blockchain Center EU, University of Kassel, Kassel, Germany
Emails: ssmetanin@hse.ru, aleksandr.ometov@tuni.fi (✉), niclas.kannengiesser@kit.edu,
benjamin.sturm@kit.edu, mkomarov@hse.ru, sunyaev@kit.edu

***Abstract*—Interest in applications based on distributed ledger technology (DLT) is on the rise, with corporations worldwide shifting from simply exploring DLT's potential to creating productive business cases. However, despite the existence of numerous DLT applications, developers are still lacking proper tools and instruments for evaluating system behavior (e.g., performance) of their applications on different distributed ledgers before deployment. Since the behavior of such applications is highly dependent on the characteristics of the distributed ledger they are built upon and changing the distributed ledger after deployment of an application is difficult, selecting the wrong ledger can have severe negative consequences. To address the issue, we conducted an extensive literature review to identify and synthesize published modeling and simulation approaches for distributed ledgers. Based on the results, this paper also presents a research agenda to improve modeling of the system behavior of distributed ledgers and to support the development of distributed ledgers and viable DLT applications. In doing so, we facilitate informed decision-making for suitable modeling or simulation approach, which helps application developers to identify a suitable distributed ledger before implementing an application. In addition, our work contributes to science, as we provide a comprehensive overview and analysis of extant simulation and modeling approaches in DLT that accumulates the current state-of-the-art in the rapidly growing DLT field.**

***Index Terms*—distributed ledger technology, modeling, blockchain, analytical modeling, simulation**

I. Introduction

Initially designed for the exchange of digital assets such as cryptocurrencies, distributed ledger technology (DLT) has already gained a remarkable impact on numerous industries worldwide (e.g., finance or health care). The trend to adopt DLT can be found in almost any digitized industry, which increases the pressure for all companies to leverage DLT to their advantage [1]. Already employed DLT applications range from public record-keeping (e.g., financial transactions [2], [3]) to private storage of inter-organizational transactions (e.g., provenance tracking of products [4], [5]).

From a technical perspective, DLT allows us to set up a tamper-resistance, append-only distributed database (referred to as a distributed ledger), where (in most cases) data is replicated across various computing and storage devices (referred to as nodes). Each stakeholder (e.g., organizations or individuals) individually maintains and controls such a geographically separated node. The essential advantage of distributed ledgers is their ability to handle the arbitrary occurrence of Byzantine failures [6]. Therefore, distributed ledgers can cope with malicious behavior of nodes, crashed or unreachable nodes, and network delays and allow for tamper-resistant data storage as well as a reliable transfer of digital assets without the need for a trusted third party [7]. To advance the basic protocols of distributed ledgers, developers can deploy DLT applications (e.g., for financial transactions in Bitcoin) to such a distributed ledger [7]. Thereby, a distributed ledger may serve as a distributed, digital infrastructure, where numerous DLT applications can be executed reliably.

Due to various technical constraints in the designs of distributed ledgers (e.g., the trade-off between availability and consistency [7], [8]), distributed ledgers cannot meet all requirements of all DLT applications simultaneously. Therefore, various distributed ledgers were designed to fulfill the requirements of certain DLT applications. Developers must choose one among these distributed ledgers, which suits their DLT applications' requirements best. The selection of a suitable distributed ledger needs to be done as early as possible during the development process because the migration of a DLT application to a different distributed ledger is costly and highly complex and, when performed after its deployment, may even lead to data loss. Since distributed ledger often serves as an infrastructure for various DLT applications, the retroactive reconfiguration of a distributed ledger to match the requirements of a certain DLT application is usually hard. To avoid serious negative consequences for the DLT application's long term viability, the development of a DLT application has to include an early and thorough evaluation of distributed ledgers to match the requirements of the DLT application.

However, such predictive suitability assessments are challenging to conduct due to the complexity of distributed ledgers and their numerous unapparent dependencies between characteristics (e.g., block size and transaction-confirmation latency) [7]. Setting up multiple distributed ledgers simply for testing purposes is time-consuming and expensive. Besides, real-world networks may only allow for limited control of

a testing environment. To reduce time efforts and cost and increase control over the testing environment, a promising approach for the suitability assessment of distributed ledgers for a specific DLT application is through simulations. Such simulations employ analytical models of distributed ledgers, which allow replicating their real-world system behavior in a controlled environment.

We consider a model to be a physical or digital product that represents an actual or theoretical system. It is similar to, but simpler than, the system it represents, "while approximating most of the same salient features of the real system as close as possible" [9]. Accordingly, *modeling* describes the act of building a model. In this paper, we investigate approaches used to model the system behavior of distributed ledgers in simulations, while we consider *simulating* as the process of running a model in a controlled execution environment (referred to as simulation environment). Modeling distributed ledgers and simulating their system behavior promise to increase our understanding of the complex dependencies between DLT characteristics which are otherwise difficult to identify or even assess (e.g., block propagation delays and consistency [10]), to identify potential drawbacks for DLT applications caused by the use of a distributed ledger, and to optimize a distributed ledger's configuration to meet the requirements of specific DLT applications [7]. However, modeling and simulating the system behavior of distributed ledgers remains challenging due to a large number of dependencies between DLT characteristics that need to be considered when creating an appropriate simulation of the system behavior.

Extant research has already presented various approaches to compare distributed ledgers from a technical perspective (e.g., considering availability or throughput) [11], [12] or from an organizational perspective (e.g., trust in other stakeholders) [13], [14]. These studies focus on comparing distributed ledgers' system behaviors in a static manner, for instance, by comparing the maximum throughput of different distributed ledgers in order to asses their suitability for a specific DLT application [11], [12]. However, in order to gain insights into the system behavior of distributed ledgers, either exhaustive field tests or controlled simulations that allow understanding the complex dependencies between DLT characteristics are required. To this end, various approaches for the simulation of the system behavior of distributed ledgers have been proposed and partially applied [10], [15]–[18]. However, the proposed modeling and simulation approaches are scattered across various sources, which is why it is hard to understand how to model and eventually simulated distributed ledgers' system behavior. To obtain profound insights into extant approaches for the simulation of the system behavior of distributed ledgers and to help decide for appropriate modeling and simulation approach, a synthesis of extant approaches for distributed ledgers is needed. To overcome this challenge, this paper addresses the following research question: *What approaches exist to analyze system behavior of distributed ledgers?*

To answer our research question, we conducted an extensive literature review to identify and synthesize published modeling and simulation approaches for distributed ledgers. Based on the results, we concluded the paper by presenting a research agenda to improve modeling of the system behavior of distributed ledgers and to support the development of distributed ledgers and viable DLT applications. In doing so, we facilitate informed decision-making for suitable modeling or simulation approach, which, in turn, helps developers to identify a suitable distributed ledger before developing a DLT application. In addition, our work contributes to science, as we provide a comprehensive overview and analysis of extant simulation and modeling approaches in DLT that accumulates the current state-of-the-art in the rapidly growing DLT field. Our research thereby serves as a fundamental knowledge base and starting point for future research advancing the simulation-driven assessment and evaluation of complex system behavior with a focus on but not limited to DLT.

II. Background

A. Distributed Ledger Technology

DLT allows to establish shared, digital infrastructures for applications (e.g., for financial transactions, logistics, production chain monitoring, etc.) by enabling the operation of a highly available, append-only distributed database (referred to as distributed ledger) in an untrustworthy environment [6], where separated storage and computing devices (referred to as nodes) maintain a local replication of the ledger [7]. Nodes are maintained and controlled by individuals or organizations. An untrustworthy environment is characterized by the arbitrary occurrence of Byzantine failures [6], such as crashed or (temporarily) unreachable nodes, network delays, and malicious behavior of nodes [19]. The data are transferred and appended to the ledger in the form of transactions and is further stored in a chronologically-ordered sequence. Each transaction contains meta-data (e.g., a transaction recipient or timestamp) and a digital representation of certain assets (e.g., coins or program code of a smart contract) [3]. When a node receives a new transaction, the transaction is validated by the selected consensus protocol, for example, by proof-of-ownership for the digital representation of the asset based on digital signatures and public-key cryptography [20] or by involving more heterogeneous nodes as part of proof-of-activity [21].

Distributed ledgers can be grouped into three concepts: *blockchains*, *block-based directed acyclic graphs (BlockDAG)* [22], and *transaction-based directed acyclic graphs (TDAG)* [7]. In blockchains, the applied consensus mechanisms target a single chain of chronologically ordered blocks representing the transaction history. In BlockDAGs, blocks can also have multiple predecessors and successors, which requires customization of the consensus protocols applied for blockchains. In TDAGS, transactions are directly linked with each other, and blocks are not even used. Another classification of distributed ledgers can be made by distinguishing between public and permissioned DLT designs (e.g., [23], [24]) or public, consortium, and private DLT designs [25]. In this paper, we distinguish between public and private DLT designs depending on whether a new

node can directly join a network (i.e., a public DLT) to read the ledger as a node or permission is required (i.e., a private DLT). While *public* and *private* specifies the read permissions, in terms of write permissions, nodes can either all have the same permission (i.e., permissionless) or require explicit permission for validating new transactions for the distributed ledger and initiating to append validated transactions to the distributed ledger (i.e., permissioned).

A central concept in distributed databases such as distributed ledgers is finality. In large distributed ledgers, like Bitcoin or Ethereum, nodes can randomly join and leave the network, which complicates or even prevents a consensus among all nodes before new data is added to the distributed ledger [26]. Due to this circumstance, newly appended data cannot be considered finalized, and only probabilistic finality is given. It means that the data cannot be removed or altered with a certain probability [18], [27]. A transaction's probability of finality increases with each additional block (or transaction) appended to the distributed ledger after the transaction [2], [10], [18]. Thus, the trust model of probabilistically final DLT designs allows for network partitions that will converge over time toward a certain state. While some nodes can agree on a state $s_{n,1}$ of the ledger and others agree on $s_{n,2}$ with $s_{n,1} \neq s_{n,2}$. Network partitions maintaining different states are referred to as *forks*. In a distributed ledger, any number of forks may exist, and a DLT design needs to employ rules that decide which blocks (or transactions) are included in the main branch of the ledger and which not. The latter are referred to as stale blocks (or stale transactions). Rules that determine a certain state of the ledger and, thus, allow the system to return to a consistent state, are called fork resolution rules (e.g., longest-chain rule in Bitcoin [2] or the Greedy Heaviest Sub Graph (GHOST) in Ethereum [3]).

Although distributed ledgers employ a Byzantine fault-tolerant threat model, distributed ledgers are not immune to attacks (e.g., [6], [7], [28]). Most attacks target double-spending, where an adversary aims to spend the same digital asset multiple times without getting the assets transferred back. Other attacks aim to compromise the integrity of a distributed ledger. As one of such attacks, selfish-mining attacks describe a phenomenon where a set of nodes work on their own local version of a blockchain (ledger or branch) without publishing their blocks to the main branch until their branch would be chosen as a future main branch by the fork resolution rule applied to the distributed ledger [7], [28], [29]. A selfish-mining attack is carried out by attackers to obtain excessive rewards or waste the computing power of honest validating nodes [29]. It was found that a successful selfish-mining attack can be performed in Bitcoin if at least one-third of the validating nodes collude [28].

B. Analysis of System Behavior of Distributed Ledgers

Since every distributed ledger has individual characteristics affecting its suitability for specific applications, developers must either identify an existing distributed ledger that matches the requirements of their DLT application or find the right configuration of these characteristics to set up a new distributed ledger. However, this requires a profound knowledge of the dependencies between DLT characteristics. To this end, extant research discusses different means (i.e., benchmarking [11], emulation [16], and simulation [30]) that enable the controlled evaluation of the system behavior of distributed ledgers with, for instance, different configurations, protocol modifications, or within different contextual settings (e.g., [31]). The most direct approach for assessing the behavior of a distributed ledger is to test a DLT application on a running system with real users. Since changes in a distributed ledger's configuration and other invasive interventions may disrupt its regular operation, evaluations of live systems are usually limited to black-box approaches like benchmarking (e.g., [11], [12]).

Benchmarks are concerned with the comparison of the running system with regards to specific characteristics (e.g., scalability or throughput). For example, BLOCKBENCH [11] is a software framework for benchmarks between private blockchains, which measures the behavior of entire systems and individual components in terms of scalability, throughput, latency, and fault-tolerance. However, evaluating running distributed ledgers can become costly when comparing a large number of different distributed ledgers due to, for example, high set-up and operational costs. Furthermore, insights into system behavior gained through black-box tests are limited to observations about system outputs and do not reveal the underlying causes.

An alternative to the evaluation of real-world systems is *emulation* of a distributed ledger (e.g., BlockLite [16]). Emulation describes the process of imitating a system on another system by replicating its function, behavior, and inner states as closely as possible [32]. Emulations employ emulation models that replace parts of the real system (e.g., hardware or software layers). Emulation is focused on the accurate replication and observation of system behavior during operation in real-time. While providing a highly faithful replication of the actual system behavior under controllable conditions, emulators require significant overhead to run the emulated interface and functionality while providing the layers of virtualization needed to emulate a whole system [33].

In contrast, *simulations* attempt to approximate a system's behavior and its development over time by running a simulation model [34]. Simulation refers to "the process of using a model to study the behavior and performance of an actual or theoretical system" [9]. Unlike an emulation that intends to replicate the operation of a system or individual system components, a simulation "can use a model to explore states that would not be possible in the original system" [9]. Another difference between simulation and emulation is that a simulation is not necessarily bound to the clock of the imitated system [32]. Thus, in contrast to an emulation that usually runs in real-time, a simulation can accelerate time or stop the system clock to make calculations and advance after a decision has been made. Simulations are built on models that must be constructed upfront [35] and have generally simplified abstractions of a simulated system that aims at covering the

specific level of details required to achieve research goals. By changing variables and conditions in the implemented simulation model, insights into how the original system behaves can be derived [35]. The simulation also allows investigating large scale systems, like distributed ledgers, with minimal use of resources by simulating thousands of nodes on a single platform [33]. Both *emulation* and *simulation* may be based on analytical models that describe specific aspects of the system under investigation.

Analytical models are a subclass of mathematical models that have a closed-form solution. In the DLT context, a set of equations can be formulated as a mathematical equation system to describe the system behavior of a distributed ledger. Such an analytical model can be used to evaluate, explain, and predict the system behavior of distributed ledgers with regards to, for example, performance (e.g., [11], [18], [36]) or security (e.g., [18], [37], [38]).

III. Method

In order to identify key publications on the analysis of distributed ledgers through modeling or simulation, we performed a literature search in scientific databases that cover leading computer science journals and conferences: *IEEE Xplore*, *ACM Digital Library*, *ScienceDirect*, *SAGE Journals Online*, and *Springer Link*. To find relevant articles and papers for our research, we applied the following search string: *(Blockchain OR DLT OR "Distributed Ledger") AND (Model OR Modeling OR Simulation OR Emulation)*. In addition to the database search, we also included pre-prints on the modeling of distributed ledgers to cover the latest advances. In total, we gathered a set of 842 potentially relevant publications, excluding grey literature and pre-prints. We then analyzed the titles, keywords, and abstracts of the publications to identify papers and articles that described at least one modeling or simulation approach for distributed ledgers. In doing so, we selected a total of 28 publications. To further extend our literature sample, we analyzed the references of the selected publications for additional papers or articles relevant to our research. Following this process resulted in a total of 32 publications. Once the literature selection process was completed, we carefully read the selected publications and used an open coding approach to identify the described modeling and simulations for distributed ledgers as well as details including, for instance, the approaches' functioning, underlying models, assumptions, and intended purposes. Next, we classified the extracted approaches into *modeling* and *simulation approaches*. Modeling approaches were further categorized into four categories based on the main approaches identified in extant literature (e.g., [31]): *Markov process*, *Markov decision processes*, and *queueing models*, and *random walks*.

IV. State-of-the-Art in Modeling of Distributed Ledgers

A. Modeling Approaches of Distributed Ledgers

In our literature review, we identified four modeling approaches applied in the context of DLT, which will be presented in the following: *Markov process*, *Markov decision processes*, and *queueing models*, and *random walks*. In general, most of the identified modeling approaches focus on the analysis of dependencies between DLT characteristics (e.g., [39], [40]) and their effects on the *transaction-confirmation probability* and *transaction-confirmation* time (e.g., [41], [42]). These DLT characteristics were found of particular importance in DLT [7], [41], [42], as they indicate the degree of security to which a submitted transaction will be appended and never be removed from the distributed ledger (see Section II-A.

1) Markov Processes: A Markov process can be used as a mathematical tool for performance evaluation of distributed ledgers [43]. A Markov process describes a process whose future probabilities are determined by its most recent values. A stochastic process can be called a Markov process only if

$$P(X_{t_n} \lneq X_n | X_{t_{n-1}}, ..., X_{t_1}) = P(X_{t_n} \lneq X_n | X t_{n-1}) \quad (1)$$

is true for every n and every $t_1 < t_2 < ... < t_n$ [44]. From the mathematical point of view, a Markov process is a first-order auto-regressive model $X_t = c + aX_{t-1} + \epsilon_t$, where X_t is a time series, a is a parameter of the model, c is a constant, and ϵ_t is a white noise.

In DLT, Markov processes are often used to model *block generation* in and*block propagation* through the distributed ledger (e.g., [41]) to evaluate performance of a distributed ledger [28], [31], [45], [46]. In such models, nodes essentially conduct Bernoulli trials [41] at a vast pace, each with an extremely low success probability, meaning that successes roughly occur as a Poisson process. In Markov processes, block generation roughly follows an exponential distribution [41]. The targeted outcome in these studies mainly focused on the examination of incentive-compatible properties of distributed ledgers (e.g., [28], [41]), studies on strategies for selfish-mining attacks (e.g., [45], [47]), and evaluation of consistency in distributed ledgers (e.g., [10].

However, our literature review shows that Markov processes become highly complex when trying to make them applicable to a wide range of scenarios, for instance, when including observations of both long and short period attacks for Nakamoto consensus or GHOST in one model [10] or when conducting analysis with standard concentration bound for Markov chains (e.g., Chernoff's bound [48]) [49]. In addition, blockchain models have been tried to be generalized from Poisson processes and exponential distribution regarding block generation [31].

2) Markov Decision Processes: A Markov decision process is a discrete time stochastic process, which is widely used as a mathematical framework for sequential decision-making task with a fully observable environment described by a Markov transition model and additional rewards. Markov decision process can be defined as a 4-tuple (S, A, P_a, R_a), where S is the finite set of states, A is the finite set of actions. $P_a(s, s')$ is the probability that action a in state s will move to state s' at the state s_{t+1}

$$P_a(s, s') = Probability(s_{t+1} = a | s_t = s, a_t = a), \quad (2)$$

where $R_a(s, s')$ is an expected reward received immediately after transitioning from state s to s' by performing action a.

In the DLT context, several studies utilized a Markov decision process as mathematical modeling to identify the optimal mining policy in blockchains in several attacks. For instance, several studies [18], [50], [51] utilized a Markov decision process to evaluate different double-spending and selfish-mining strategies. In these evaluations, the applied security model is based upon Markov decision processes.

As an extension of Markov process models, models based on Markov decision processes often obtain relatively better results on examining selfish-mining attack strategies (e.g., [18], [50], [51]).

3) Queueing Models: A queueing model is a mathematical model, which allows for predictions of queue lengths and waiting times. In the context of DLT, queueing models have been predominantly used to represent and analyze the transaction-confirmation process on the block-level (e.g., [52]) or transaction-level (e.g., [42]). All queueing models examined in this literature review can be characterized as a single-server queue with batch service [40], [42], [53] and applied an exponential distribution for block generation (e.g., [40], [53]). Transactions are considered to be processed in batches represented by blocks. The arrival of the transactions (or blocks) has mainly been assumed as a Poisson process (e.g., [53]).

Queueing models have often been applied to analyze the mining process for distributed ledgers (e.g., [39], [41], [54], [55]). The authors proposed a stochastic model, which aims at capturing the dynamics in distributed ledgers (e.g., in terms of block generation and block arrivals) and evolution [41], [55]. Ricci et al. [54] proposed a model that incorporates machine learning in combination with a queueing model. The model aimed to predict which transactions will be confirmed and the corresponding confirmation time.

Several studies investigated fluid limits – a subsection in the queueing theory describing deterministic processes, which aims at an approximation of the evolution of the analyzed stochastic process. For instance, the research group from the University of Amsterdam developed a Bitcoin-inspired infinite-server model with a random fluid limit [56]. King et al. [57] considered the fluid limit of a random graph model for distributed ledgers.

4) Random Walks: A random walk is a stochastic mathematical model that describes a path that consists of random changes or steps on mathematical space at discrete points in the time being widely utilized for various information and communications technology (ICT) systems analysis [58]–[60]. Given independent and identically distributed random variables $X_1, X_2, ..., X_n$, where $X_i \in R_n$, the pure structure of random walks can be defined as

$$Y_n = \sum_{1}^{n} X_i. \tag{3}$$

To study double-spending attacks in DAG-based distributed ledgers, Goffard [61] refined Nakamoto's model [2] with a primary focus on the probability of attack success. Jang and Lee [62] proposed a new model, which takes into account block confirmation release, and, thus, advances Goffard's model. Grunspan and Perez-Marco computed the minimal number of confirmations to be requested by the recipient such that the double-spend strategy is non-profitable [63].

B. Simulation of Distributed Ledgers

Simulations of distributed ledgers can be described using three groups of characteristics: *purpose*, *model*, and *usability*.

In terms of potential simulation *purposes*, our literature review identified four categories: *testing and verification*, *evaluation and validation*, *prediction*, and *understanding*. First, simulations can be used by developers to *test* software modules and to enable agile software development (e.g., [64]) as well as to verify models for system behavior of distributed ledgers. In the latter case, developers of a model can check if they implemented it correctly (e.g., verify a match between specifications and assumptions deemed acceptable for the given purpose of an application) and try to fix errors in the model [34], [65]. Second, *evaluation and validation* comprises two perspectives. First, evaluation aims at assuring that a distributed ledger fulfills certain properties, for example, a certain mean transaction-confirmation latency (the period between the time a transaction is appended to a distributed ledger and the time that the probability of that transaction being excluded from the distributed ledger becomes negligible). During the validation, model developers evaluate in cooperation with experts if their final model meets requirements in terms of, for example, the accuracy of certain observations (e.g., [10], [50] [65]). Third, simulations can be used to *predict* system behavior of distributed ledgers in contexts that are critical for the system (e.g., [10], [28], [65]). Fourth, *understanding* is concerned with analyzing dependencies between DLT characteristics. To this end, analysts use the controlled simulation to manipulate certain DLT characteristics (e.g., block size [28], block generation difficulty levels [41], or network delays [42]), and observe resulting effects on other DLT characteristics (e.g., number of forks [41] or consistency [10]). Table I illustrates the different purposes of simulations, correspondingly applied modeling approaches and distributed ledgers that can be simulated based upon the proposed model. According to the reviewed literature, some authors used simulations for more than one purpose (e.g., evaluation, understanding, and prediction [50]).

Simulations typically build upon *models* that are generated based on mathematical models (see Section IV-A). Thus, the *purpose* of a simulation determines the minimum capabilities of the underlying model (see Section IV-A). Analysts need to be able to manipulate the starting configuration of a model (e.g., block size [28] or network dynamics [66]), to observed DLT characteristics (e.g., consistency [10] or a number of forks [41]), and derive the results that help to fulfill the particular purpose of the simulation (e.g., certain mining strategies with high success probability [10], [50], [51]). In order to make a model applicable for a variety of scenarios, the model needs to be flexible enough in terms of configurations (e.g., stochastic distribution of events like block

generation) and the type of distributed ledger to be simulated (e.g., different consensus mechanisms [10]).

Usability is "the extent to which a system, product or service can be used by specified users to achieve specified goals with effectiveness, efficiency, and satisfaction in a specified context of use" [67]. Usability of the distributed ledgers' system behavior simulator concerns the appropriate configuration of simulation scenarios, the required execution environment, and an understandable output of observations and targeted results (e.g., [16], [30]). In most papers on the modeling of distributed ledgers, the authors tested and verified their proposed models in simulations (e.g., [10], [42]) without a graphical user interface (GUI) and, thus, provide little usability for the configuration of the simulation environment. In contrast, some simulators for distributed ledgers allow for a comfortable use through a web interface (e.g., [30]). Such web interfaces allow users to set simulation parameters (e.g., number of nodes [30], [52]) and may even visualize the DLT characteristics to be observed (e.g., network growth [30]). Regarding the execution environment, most simulations run on a single machine (e.g., [10], [68]) or outsource to a cloud service (e.g., [30]). By doing so, simulations can be conducted independently from other machines, and users gain better control over simulated network dynamics. However, one of the critical aspects of constructing simulation models tends to be the consumption of computational resources. Resource consumption becomes particularly critical for simulations running on single nodes that cannot scale in their computational resources, like virtualized machines. There are several strategies to tackle the issue of resource consumption, including simplification of resource-competitive modules [15], utilization of multi-threading approaches, and/or multiple CPU cores for the simulation [33], [68], and performing simulations in the cloud [30], [66]. The first strategy (simplification) is also applicable when dealing with resource constraints on a single machine, but may come at the cost of the accuracy of the model, due to over-simplification (e.g., [16]). The second strategy (parallelization) may require certain hardware (e.g., multiple CPU cores) and, thus, is only applicable for a fraction of potential users of the simulation (e.g., [16]). The third strategy (virtualization) appears to combine the strategies *simplification* and *parallelization* by making use of virtualization of computational resources in a cloud ([30]). Hence, the third strategy may achieve high usability due to its high scalability.

In the following, we show the applicability of our classification and illustrate dependencies between *purpose*, *model*, and *usability* by real-world simulators for distributed ledgers.

V. Principle Findings and Future Directions

A. Principle Findings

In this paper, we identified four approaches for modeling system behavior of distributed ledgers, the modeling approaches' respective purpose and observations, and the modeling approaches' use in simulations (see Table I). Our findings show that most modeling approaches and simulations focus on the analysis of transaction-confirmation latency. The phenomenon of transaction-confirmation latency has been examined from various perspectives such as the identification of optimal attack strategies (e.g., [28]), resilience mechanisms of the distributed ledger (e.g., [50], or by focusing on consistency of distributed ledgers (e.g., [10]).

In addition, we identified three groups of characteristics that help to differentiate simulations of distributed ledgers (i.e., *purpose*, *model*, and *usability*), explicate interactions between these groups, and illustrate the identified interactions through exemplary simulations (e.g., Bitcoin-Simulator [52]).

One principle finding indicates a best practice for distributed ledger modeling is that the investigated modeling approaches make several similar assumptions regarding block generation and block propagation. Various models assume an exponential distribution for block generation in a Poisson process [40], [50], [53], where the block generation is considered a *batch* because the block contains various transactions [40], [42], [50], [53]. Receiving blocks and transactions are often modeled in a Poisson process (e.g., [41]). These assumption also apply for DLT concepts other than blockchains (e.g., BlockDAG or TDAG; see Section II) [15], [66].

The examined models strongly differ in their representation of the PoW for distributed ledgers that make use of PoW-based consensus mechanisms. With the high scalability of later simulations as their common goal, two modeling approaches for PoW are being used: *adding timestamps* and *real PoW*. First, *adding timestamps* assumes the elapsed time based on equivalent real-world computations for PoW. Since no actual computations are performed, it is the least resource-consuming modeling approach of the three and, thus, scalable to a large number of nodes, but also the less realistic than the other modeling approaches. *Fast-forwarding* – a subcategory to adding timestamps – models PoW more realistically while maintaining high scalability to a large number of nodes. Adding timestamps fast-forwarding allows accelerating simulation since the actual computations will not take as long as the approximated elapsed time for the simulated PoW. Third, the use of *real PoW* executed by a single node as a representation for all nodes has been found a realistic approximation to real-world distributed ledgers [16]. Nevertheless, real PoW is the most resource-consuming approach in terms of time and computational resources and requires a calibration of the PoW difficulty so that it matches the block creation interval of the simulated distributed ledger [16]. Block generation based on mining can also be modeled as Bernoulli trials (e.g., miner guessed a correct nonce for the block or not), each with an extremely low success probability for block generation. Thus, successful block generation roughly occurs as a Poisson process [41]. Nonetheless, to the best knowledge of the authors, there is no comparative study on the differences between fast-forwarding and real PoW in terms of model accuracy. Modeling of the system behavior of distributed ledgers is predominantly focused on the most popular public-permissionless blockchains, namely Bitcoin, Ethereum (e.g., [10], [18], [28], [50], and IOTA as one of

TABLE I
OVERVIEW OF SIMULATIONS OF DISTRIBUTED LEDGERS

Simul. Purpose	Model. Appr.	Observation(s)	Exampl. Referen.	Modeled for							
				BCH	BTC	CHW	DODGE	ETH	IOTA	LTC	SYS
Testing & Verification	MP	n/a	–								
	MDP	n/a	–								
	QM	Development and evaluation of software components	[16]		X						
		Performance analysis and optimization	[36]		X						
		Profitability of double-spending attacks	[54]								
		Rapid prototyping of new features and fixes	[68]		X						
	RW	Check analytical calculations for DAG-based distributed ledgers	[15]						X		
		Profitability of double-spending attacks	[62]	X	(X)		(X)	(X)		(X)	X
Evaluation & Validation	MP	Evaluation of consistency of distributed ledgers	[10]		X	X		X			
		Double-Spending, "Parasite Chain", and Lazy Nodes Attacks	[69]						X		
	MDP	Evaluation of selfish-mining strategies	[18]		X		X	X		X	
		Evaluation of selfish-mining and stubborn-mining strategies	[45]		X		X	X		X	
		Evaluation of selfish-mining strategies	[50]		X			X			
		Evaluation of selfish-mining strategies	[51]		X						
	QM	Evaluation of selfish-mining strategies	[41]		X						
	RW	Double-Spending Attacks	[61]		X						
Prediction	MP	n/a	–								
	MDP	Best selfish-mining strategies for various contexts	[50]		X			X			
	QM	Long term growth rate of blockchains	[41]					X			
	RW	n/a	–								
Understanding	MP	Mean number of transactions in the Memory Pool, mean number of transactions in a block, mean transaction-confirmation time	[42]		X						
	MDP	Profit threshold for selfish mining attacks	[50]		X						
	QM	Long term growth rate of blockchains	[41]				X				
		Mean transaction-confirmation time	[53]		X						
		Mean transaction-confirmation time	[54]		X						
		Cost and effectiveness of vulnerabilities in the Bitcoin software	[68]		X						
	RW	Impact of network latency on transaction-attachment probabilities	[15]						X		

QM	–	Queueing models	MP	–	Markov process	MDP	–	Markov decision process	RW	–	Random walks
BCH	–	Bitcoin Cash	CHW	–	Chainweb	BTC	–	Bitcoin	CHW	–	Chainweb
ETH	–	Ethereum	LTC	–	Litecoin	LTC	–	Litecoin	SYS	–	Syscoin
X	–	Modeled and compared to real data				(X)	–	Claimed to be modeled but not evaluated			

the major TDAGs (e.g., [66], [69]). Since various distributed ledgers build upon the protocols of Bitcoin and Ethereum, several models claim to be generalizable to distributed ledgers that employ a certain type of consensus mechanism (e.g., PoW-based [10], [18]). Nevertheless, the generalizability of existing models is still not verified and, thus, is considered a challenge for future research (e.g., [10]).

As illustrated in Table I, the *evaluation and validation* of existing distributed ledgers gained increased interest in recent years. Foremost, the analysis of inconsistencies between nodes of a distributed ledgers due to transaction propagation delays raised particular interest in the security domain. Studies on mean transaction-confirmation intervals of distributed ledgers revealed various trade-offs between security and performance (e.g., [28], [41]). It is reasonable to assume that focusing on *evaluation and validation* of distributed ledgers' system behaviors is a first step toward ensuring the security of distributed ledgers before even building real-world DLT applications for it. Demand for simulations that facilitate test-driven development of DLT applications may grow, once appropriate models are available.

Our final principal finding concerns the current state of simulators available to developers. Although modeling of the system behavior of popular distributed ledgers (i.e., Bitcoin, Ethereum, and IOTA) already achieved high accuracy in terms of replicating the behavior of the respective real-world system (e.g., [10]), their application is still complex and they are often not even accessible for developers. Most simulators identified in the literature review were in an early prototypical stage and predominantly focused on the software architecture

of the simulator, including a user interface (e.g., command-line interface or web interface [30]), configurable parameters (e.g., number of nodes [16], [30], [64] or network bandwidth [16], [64]), and the execution environment of the simulation (e.g., a single node or cloud [30]). However, the actual models used by simulators identified in our review are often not described (e.g., [30]).

B. Future Research Opportunities

Based on the analysis of the identified literature, the following six future research opportunities were derived. In general, future research needs to thoroughly investigate the modeling approaches presented in this paper in order to identify potential synergies between the individual approaches to allow for a more comprehensive analysis of system behavior of distributed ledgers.

1) Model simplifications and their effect on the accuracy of the model: Event-based simulation models (e.g., [30], [70]) operate by abstracting and/or simplifying parts of a distributed ledger's protocol. Such abstractions increase scalability and decrease resource consumption. However, such abstractions and simplifications may lead to neglecting traits of distributed ledgers [71]. To identify and assess effects that emerge from abstraction and simplification of a distributed ledger's protocol, an empirical analysis should be conducted included comparative tests between different levels of abstraction and/or simplifications. Such insights will support the development of more accurate and less resource consuming simulations.

2) Consideration of multiple transaction classes: Most examined papers deal with the modeling of blockchains, their individual transaction-confirmation latency, and the probability of transaction-confirmation (e.g., [42], [53], [54]). To model realistic transaction-confirmation latency and transaction-confirmation probability, a prevailing challenge is to represent different classes of transactions and to model a system dealing with them similar to real-world distributed ledgers [40], [53]. The fee may determine classes of transactions a user is willing to pay [40] or by the time a transaction is already in a miner's MemPool. While approaches exist to tackle this challenge (e.g., [53], [54]), modeling approaches for real-world prioritization mechanisms for different classes of transactions require further investigations [42], [54].

3) Dependencies between DLT characteristics: While certain relations or even trade-offs between quantitative DLT and underlying blockchain characteristics were deeply examined (e.g., the effect of block size limits on transaction-confirmation times [40], [53]), the relations between other DLT characteristics have remained open. For example, trade-offs between a degree of decentralization of a distributed ledger and charged transaction fees (e.g., [7]). In order to simulate distributed ledgers more comprehensively, while taking into account such DLT characteristics, more detailed and testable criteria must be generated. Prior research on trade-offs between DLT characteristics (e.g., [7], [8]) can be used as a starting point for more comprehensive modeling and simulations of distributed ledgers.

4) Interoperability: Currently, it is commonly held that there cannot be a one-size-fits-all distributed ledger that suits all requirements of all DLT applications at the same time (e.g., [7], [72]). This is why new means for facilitating interaction between distributed ledgers and external (non-DLT) services that help to meet specific DLT application requirements is a highly relevant field of study (e.g., [73], [74]). However, even though respective interoperability protocols are being developed (e.g., BTC Relay or Town Crier), it remains hard to predict the implications for DLT applications when implementing these protocols. Hence, interoperability between different distributed ledgers as well as between distributed ledgers and external non-DLT services are worth consideration in future modeling and simulation approaches, for example, with regard to the employed threat models and performance characteristics.

5) Machine learning in simulation modeling: While machine learning approaches demonstrated an ability to perform successfully in system modeling tasks [54], [75]–[78], only a few studies applied machine learning to model distributed ledgers. For instance, Markov Chain Neural Networks [79] may have great potential for the simulation of Markov chains for modeling distributed ledgers to predict system behavior, while decreasing run-time. At the same time, the Markov decision process extraction network [80] potentially can be used in order to automatically extract minimal relevant aspects of the dynamics from observations to model a Markov Decision Process.

6) Testing environment for DLT application development: Extant modeling approaches for distributed ledgers predominantly focus on the *evaluation and validation* of distributed ledgers from a security perspective (see Section V-A and Table I). However, these models are frequently the subject of research projects that are not concerned with their practical application, even though the development of DLT applications would benefit from the use of such models in simulations to test DLT applications. It would require, for example, easy-to-use simulators integrated into development environments. To this end, further investigations are required to clarify to which extent existing models are applicable in practice and how to best interact with simulations using such models.

C. Limitations

This research is subject to two major limitations. First, since the simulation of distributed ledgers is still in its infancy, the majority of papers in our literature review present research that still resides at an early stage. As to be expected, this results in an overall lack of empirical evidence for the effectiveness of the individual approaches. Second, we limited our literature search to already proposed or even implemented modeling approaches in the field of DLT. Therefore, our results do not include alternative modeling approaches from other fields, which might be applicable in the DLT context. In addition to the aforementioned research opportunities, we would like to encourage further investigations of such extant modeling approaches and their contextual evaluation.

VI. Conclusion

In this work, we surveyed extant approaches to model and simulate system behavior of distributed ledgers in order to support the development of viable applications on suitable distributed ledgers. We identified four modeling approaches applied in the context of DLT: Markov processes, Markov decision processes, queueing models, and random walks. We further were able to synthesize and systematically characterize existing simulations approaches by their purpose, employed modeling approach, and their usability. Finally, this paper presents a research agenda to improve modeling of the system behavior of distributed ledgers and to support the development of distributed ledgers and viable DLT applications. In doing so, the contributions of this paper to practice and research are threefold. First, we help practitioners and researchers selecting an appropriate distributed ledger simulator for particular purposes. Thereby, we help developers to predict potential drawbacks for DLT applications to be developed and, consequently, to make more sophisticated decisions for a certain distributed ledger to be employed. Second, developers can obtain initial best practices in simulation development for distributed ledgers as we describe individual benefits and challenges. Third, we contribute to research as we outline potentials future research directions to improve distributed ledger simulations.

Acknowledgements

This work was carried out in the scope of the project COOLedger (Helmholtz Association of German Research Centers: HRSF-0081, Russian Science Foundation: Project No. 19-41-06301).

References

[1] L. Pawczuk, R. Massev, and D. Schatsky, *Breaking Blockchain Open: Deloitte's 2018 Global Blockchain Survey*. Deloitte Development LLC, 2018, (accessed June 24, 2020). [Online]. Available: https://www2.deloitte.com/content/dam/Deloitte/us/Documents/financial-services/us-fsi-2018-global-blockchain-survey-report.pdf

[2] S. Nakamoto, "Bitcoin: A Peer-to-Peer Electronic Cash System," Manubot, Tech. Rep., Nov 2019.

[3] V. Buterin. (2019, Mar) Ethereum Whitepaper – A Next-Generation Smart Contract and Decentralized Application Platform. (accessed June 24, 2020). [Online]. Available: https://whitepaper.io/coin/ethereum

[4] F. Tian, "A Supply Chain Traceability System for Food Safety Based on HACCP, Blockchain Internet of Things," in *Proc. of International Conference on Service Systems and Service Management*, 2017, p. 1–6.

[5] T. Lehikoinen. (2018, Jul) Food Supply Chain This Summer, Fishing in Finland Means Food Traceability on the Menu. (accessed June 24, 2020). [Online]. Available: https://www.ibm.com/blogs/blockchain/2018/07/this-summer-fishing-in-finland-means-food-traceability-on-the-menu/

[6] L. Lamport, R. Shostak, and M. Pease, "The Byzantine Generals Problem," *ACM Transactions on Programming Languages and Systems*, vol. 4, no. 3, pp. 382–401, 1982.

[7] N. Kannengießer, S. Lins, T. Dehling, and A. Sunyaev, "Trade-offs between distributed ledger technology characteristics," *ACM Computing Surveys*, vol. 53, no. 2, April 2020.

[8] O. O'Donoghue, A. A. Vazirani, D. Brindley, and E. Meinert, "Design Choices and Trade-Offs in Health Care Blockchain Implementations: Systematic Review," *Journal of Medical Internet Research*, vol. 21, no. 5, p. e12426, May 2019.

[9] A. Maria, "Introduction to Modeling and Simulation," in *Proc. of 29th Conference on Winter Simulation*, 1997, pp. 7–13.

[10] L. Kiffer, R. Rajaraman, and A. Shelat, "A Better Method to Analyze Blockchain Consistency," in *Proc. of ACM SIGSAC Conference on Computer and Communications Security*, 2018, pp. 729–744.

[11] T. Dinh, T. Anh, J. Wang, G. Chen, R. Liu, C. Ooi, and K.-L. Tan, "Blockbench: A Framework for Analyzing Private Blockchains," in *Proc. of ACM International Conference on Management of Data*, 2017, pp. 1085–1100.

[12] F. Gräbe, N. Kannengießer, S. Lins, and A. Sunyaev, "Do Not Be Fooled: Towards a Holistic Comparison of Distributed Ledger Technology Designs," in *Proc. of 53th Hawaii Internantional Conference on System Sciences*, 2020.

[13] S. K. Lo, X. Xu, Y. K. Chiam, and Q. Lu, "Evaluating Suitability of Applying Blockchain," in *Proc. of 22nd International Conference on Engineering of Complex Computer Systems*, Nov 2017, pp. 158–161.

[14] K. Wüst and A. Gervais, "Do You Need a Blockchain?" in *Proc. of Crypto Valley Conference on Blockchain Technology (CVCBT)*, June 2018, pp. 45–54.

[15] M. Zander, T. Waite, and D. Harz, "DAGsim: Simulation of DAG-based Distributed Ledger Protocols," *ACM SIGMETRICS Performance Evaluation Review*, vol. 46, no. 3, pp. 118–121, Dec 2018.

[16] X. Wang, A. Al-Mamun, H. Lin, F. Yan, M. Sadoghi, and D. Zhao, "BlockLite: A Lightweight Emulator for Public Blockchains," *arXiv preprint arXiv:1905.02157*, May 2019.

[17] D. Marmsoler and L. Eichhorn, "Simulation-Based Analysis of Blockchain Architectures," *Unpublished*, 2018, (accessed June 24, 2020). [Online]. Available: http://rgdoi.net/10.13140/RG.2.2.19898.44481

[18] A. Gervais, G. O. Karame, K. Wüst, V. Glykantzis, H. Ritzdorf, and S. Capkun, "On the Security and Performance of Proof of Work Blockchains," in *Proc. of ACM SIGSAC Conference on Computer and Communications Security*, 2016, pp. 3–16.

[19] A. Sunyaev, *Distributed Ledger Technology*, 1st ed. Springer Nature, 2019, p. 265–292.

[20] D. Chaum, *Blind Signatures for Untraceable Payments*. Springer US, 1983, pp. 199–203.

[21] K. Zhidanov, S. Bezzateev, A. Afanasyeva, M. Sayfullin, S. Vanurin, Y. Bardinova, and A. Ometov, "Blockchain Technology for Smartphones and Constrained IoT Devices: A Future Perspective and Implementation," in *Proc. of 21st Conference on Business Informatics (CBI)*, vol. 2. IEEE, 2019, pp. 20–27.

[22] Y. Lewenberg, Y. Sompolinsky, and A. Zohar, "Inclusive Block Chain Protocols," in *Proc. of International Conference on Financial Cryptography and Data Security*. Springer, 2015, pp. 528–547.

[23] D. Massessi. (2018, Dec) Public vs Private Blockchain in a Nutshell. (accessed June 24, 2020). [Online]. Available: https://medium.com/coinmonks/public-vs-private-blockchain-in-a-nutshell-c9fe284fa39f

[24] P. Cabianca, "What's the difference between Public, Private and Permissioned Blockchains?" *Medium*, 2018.

[25] Z. Zheng, S. Xie, H. N. Dai, X. Chen, and H. Wang, "Blockchain Challenges and Opportunities: A Survey," *International Journal of Web and Grid Services*, vol. 14, no. 4, p. 352, 2018.

[26] R. Pass and E. Shi, "The Sleepy Model of Consensus," in *Proc. of Advances in Cryptology (ASIACRYPT)*, T. Takagi and T. Peyrin, Eds., 2017, p. 380–409.

[27] K. Saito and Y. Hairoyuki, "What's So Different about Blockchain? – Blockchain is a Probabilistic State Machine," in *Proc. of 36th International Conference on Distributed Computing Systems Workshops*. IEEE, Jun 2016, p. 168–175.

[28] I. Eyal and E. G. Sirer, "Majority is Not Enough: Bitcoin Mining is Vulnerable," in *Proc. of International Conference on Financial Cryptography and Data Security*. Springer, 2014, pp. 436–454.

[29] J. Göbel, P. Keeler, A. E. Krzesinski, and P. G. Taylor, "Bitcoin Blockchain Dynamics: The Selfish-mine Strategy in the Presence of Propagation Delay," *Performance Evaluation*, vol. 104, p. 23–41, 2016.

[30] L. Stoykov, K. Zhang, and H.-A. Jacobsen, "VIBES: Fast Blockchain Simulations for Large-scale Peer-to-Peer Networks," in *Proc. of 18th ACM/IFIP/USENIX Middleware Conference: Posters and Demos*, 2017, pp. 19–20.

[31] Q.-L. Li, J.-Y. Ma, Y.-X. Chang, F.-Q. Ma, and H.-B. Yu, "Markov Processes in Blockchain Systems," *Computational Social Networks*, vol. 6, no. 1, pp. 1–28, 2019.

[32] I. McGregor, "The Relationship Between Simulation and Emulation," in *Proc. of the Winter Simulation Conference*, vol. 2, Dec 2002, pp. 1683–1688.

[33] C. Faria and M. Correia, "BlockSim: Blockchain Simulator," in *Proc. of IEEE International Conference on Blockchain*. IEEE, 2019, pp. 439–446.

[34] J. Banks, *Discrete Event System Simulation*. Pearson Education, 2005.

[35] F. Landriscina, *Simulation and Learning: A Model-centered Approach*. Springer, 2013.

[36] R. A. Memon, J. P. Li, and J. Ahmed, "Simulation Model for Blockchain Systems Using Queuing Theory," *Electronics*, vol. 8, no. 2, p. 234, 2019.

[37] C. Natoli and V. Gramoli, "The Balance Attack or Why Forkable Blockchains are Ill-Suited for Consortium," in *Proc. of 47th Annual IEEE/IFIP International Conference on Dependable Systems and Networks*, Jun 2017, p. 579–590.

[38] I. Weber, V. Gramoli, A. Ponomarev, M. Staples, R. Holz, A. B. Tran, and P. Rimba, "On Availability for Blockchain-Based Systems," in *Proc. of 36th Symposium on Reliable Distributed Systems*. IEEE, Sep 2017, p. 64–73.

[39] R. A. Memon, J. Li, J. Ahmed, A. Khan, M. I. Nazir, and M. I. Mangrio, "Modeling of Blockchain Based Systems Using Queuing Theory Simulation," in *Proc. of 15th International Computer Conference on Wavelet Active Media Technology and Information Processing (ICCWAMTIP)*. IEEE, 2018, pp. 107–111.

[40] S. Kasahara and J. Kawahara, "Effect of Bitcoin Fee on Transaction-confirmation Process," *Journal of Industrial & Management Optimization*, vol. 15, no. 1, pp. 365–386, 2019.

[41] N. Papadis, S. Borst, A. Walid, M. Grissa, and L. Tassiulas, "Stochastic Models and Wide-Area Network Measurements for Blockchain Design and Analysis," in *Proc. of IEEE Conference on Computer Communications (INFOCOM)*. IEEE, 2018, pp. 2546–2554.

[42] Q.-L. Li, J.-Y. Ma, and Y.-X. Chang, "Blockchain Queue Theory," in *Proc. of International Conference on Computational Social Networks*. Springer, 2018, pp. 25–40.

[43] G. Bolch, S. Greiner, H. De Meer, and K. S. Trivedi, *Queueing Networks and Markov Chains: Modeling and Performance Evaluation with Computer Science Applications*. John Wiley & Sons, 2006.

[44] A. T. Bharucha-Reid, *Elements of the Theory of Markov Processes and Their Applications*. Courier Corporation, 2012.

[45] K. Nayak, S. Kumar, A. Miller, and E. Shi, "Stubborn Mining: Generalizing Selfish Mining and Combining with an Eclipse Attack," in *Proc. of IEEE European Symposium on Security and Privacy (EuroS&P)*. IEEE, 2016, pp. 305–320.

[46] D. Huang, X. Ma, and S. Zhang, "Performance Analysis of the Raft Consensus Algorithm for Private Blockchains," *IEEE Systems, Man, and Cybernetics Society*, p. 1–10, 2019.

[47] Q. Bai, X. Zhou, X. Wang, Y. Xu, X. Wang, and Q. Kong, "A Deep Dive into Blockchain Selfish Mining," in *Proc. of IEEE International Conference on Communications (ICC)*. IEEE, 2019, pp. 1–6.

[48] K.-M. Chung, H. Lam, Z. Liu, and M. Mitzenmacher, "Chernoff-Hoeffding Bounds for Markov Chains: Generalized and Simplified," *arXiv preprint arXiv:1201.0559*, p. 12, 2012.

[49] R. Pass, L. Seeman, and A. Shelat, "Analysis of the Blockchain Protocol in Asynchronous Networks," in *Proc. of Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 2017, pp. 643–673.

[50] A. Sapirshtein, Y. Sompolinsky, and A. Zohar, "Optimal Selfish Mining Strategies in Bitcoin," in *Proc. of International Conference on Financial Cryptography and Data Security*. Springer, 2016, pp. 515–532.

[51] R. Zhang and B. Preneel, "Publish or Perish: A Backward-compatible Defense Against Selfish Mining in Bitcoin," in *Proc. of Cryptographers' Track at the RSA Conference*. Springer, 2017, pp. 277–292.

[52] J. Garay, A. Kiayias, and N. Leonardos, "The Bitcoin Backbone Protocol with Chains of Variable Difficulty," in *Proc. of Annual International Cryptology Conference*. Springer, 2017, pp. 291–323.

[53] Y. Kawase and S. Kasahara, "Transaction-Confirmation Time for Bitcoin: A Queueing Analytical Approach to Blockchain Mechanism," in *Proc. of International Conference on Queueing Theory and Network Applications*. Springer, 2017, pp. 75–88.

[54] S. Ricci, E. Ferreira, D. S. Menasche, A. Ziviani, J. E. Souza, and A. B. Vieira, "Learning Blockchain Delays: A Queueing Theory Approach," *ACM SIGMETRICS Performance Evaluation Review*, vol. 46, no. 3, pp. 122–125, 2019.

[55] R. Bowden, H. P. Keeler, A. E. Krzesinski, and P. G. Taylor, "Block Arrivals in the Bitcoin Blockchain," *arXiv preprint arXiv:1801.07447*, Jan 2018.

[56] M. Frolkova and M. Mandjes, "A Bitcoin-Inspired Infinite-Server Model with a Random Fluid Limit," *Stochastic Models*, vol. 35, no. 1, pp. 1–32, 2019.

[57] C. King, "The Fluid Limit of a Random Graph Model for a Shared Ledger," *arXiv preprint arXiv:1902.05050*, 2019.

[58] Y. Orlov, E. Kirina-Lilinskaya, A. Samuylov *et al.*, "Time-dependent SIR Analysis in Shopping Malls Using Fractal-based Mobility Models," in *Proc. of International Conference on Wired/Wireless Internet Communication*. Springer, 2017, pp. 16–25.

[59] F. Spitzer, *Principles of Random Walk*. Springer Science & Business Media, 2013, vol. 34.

[60] D. Moltchanov, A. Ometov, and Y. Koucheryavy, "Analytical Characterization of the Blockage Process in 3GPP New Radio Systems with Trilateral Mobility and Multi-connectivity," *Computer Communications*, vol. 146, pp. 110–120, 2019.

[61] P.-O. Goffard, "Fraud Risk Assessment within Blockchain Transactions," *Advances in Applied Probability*, vol. 51, no. 2, pp. 443–467, Jun. 2019.

[62] J. Jang and H.-N. Lee, "Profitable double-spending attacks," *arXiv preprint arXiv:1903.01711*, 2019.

[63] C. Grunspan and R. Pérez-Marco, "On Profitability of Nakamoto Double Spend," *arXiv preprint arXiv:1912.06412*, 2019.

[64] X. Wang, A. Al-Mamun, F. Yan, and D. Zhao, "Toward Accurate and Efficient Emulation of Public Blockchains in the Cloud," in *Proc. of International Conference on Cloud Computing*, 2019, pp. 67–82.

[65] J. Carson, "Model Verification and Validation," in *Proc. of the Winter Simulation Conference*, vol. 1. IEEE, 2002, p. 52–58.

[66] M. R. A. Lathif, P. Nasirifard, and H.-A. Jacobsen, "CIDDS: A Configurable and Distributed DAG-based Distributed Ledger Simulation Framework," in *Proc. of 19th International Middleware Conference (Posters)*, 2018, pp. 7–8.

[67] ISO/TC 159/SC 4. (2018, Mar) ISO 9241-11:2018(en) Ergonomics of human-system interaction – Part 11: Usability: Definitions and concepts. (accessed June 24, 2020). [Online]. Available: https://www.iso.org/standard/63500.html

[68] A. Miller and R. Jansen, "Shadow-Bitcoin: Scalable Simulation via Direct Execution of Multi-Threaded Applications," in *Proc. of 8th USENIX Conference on Cyber Security Experimentation and Test*, ser. CSET'15. USA: USENIX Association, 2015, p. 7.

[69] S. Popov. (2018, April) The Tangle–Version 1.4.3. (accessed June 24, 2020). [Online]. Available: https://assets.ctfassets.net/r1dr6vzfxhev/2t4uxvsIqk0EUau6g2sw0g/45eae33637ca92f85dd9f4a3a218e1ec/iota1_4_3.pdf

[70] Y. Aoki, K. Otsuki, T. Kaneko, R. Banno, and K. Shudo, "SimBlock: A Blockchain Network Simulator," in *Proc. of IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*. IEEE, 2019, pp. 325–329.

[71] L. Alsahan, N. Lasla, and M. Abdallah, "Local Bitcoin Network Simulator for Performance Evaluation using Lightweight Virtualization," *arXiv preprint arXiv:2002.01243*, 2020.

[72] V. Buterin. (2016) Chain Interoperability [White Paper]. (accessed June 24, 2020). [Online]. Available: http://www.r3cev.com/s/Chain-Interoperability-8g6f.pdf

[73] J. Heiss, J. Eberhardt, and S. Tai, "From Oracles to Trustworthy Data On-chaining Systems," in *Proc. of IEEE International Conference on Blockchain*, 2019.

[74] J. Eberhardt and J. Heiss, "Off-chaining Models and Approaches to Off-chain Computations," in *Proc. of the 2nd Workshop on Scalable and Resilient Infrastructures for Distributed Ledgers*, 2018, p. 7–12.

[75] F. Recknagel, "Applications of Machine Learning to Ecological Modelling," *Ecological Modelling*, vol. 146, no. 1-3, pp. 303–310, 2001.

[76] H. Tariq, H. Al-Sahaf, and I. Welch, "Modelling and Prediction of Resource Utilization of Hadoop Clusters: A Machine Learning Approach," in *Proc. of 12th IEEE/ACM International Conference on Utility and Cloud Computing*, 2019, pp. 93–100.

[77] P. D. Dueben and P. Bauer, "Challenges and Design Choices for Global Weather and Climate Models Based on Machine Learning," *Geoscientific Model Development*, vol. 11, no. 10, pp. 3999–4009, 2018.

[78] A. Mosavi, M. Salimi, S. Faizollahzadeh Ardabili et al., "State of the Art of Machine Learning Models in Energy Systems, a Systematic Review," *Energies*, vol. 12, no. 7, p. 1301, 2019.

[79] M. Awiszus and B. Rosenhahn, "Markov Chain Neural Networks," in *Proc. of IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2018, pp. 2180–2187.

[80] S. Duell, A. Hans, and S. Udluft, "The Markov Decision Process Extraction Network," in *Proc. of European Symposium On Artificial Neural Networks, Computational Intelligence and Machine Learning*, 2010.