IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 5 June 2024, accepted 14 June 2024, date of publication 19 June 2024, date of current version 26 June 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3416612

RESEARCH ARTICLE

Performance Evaluation of a Distributed Ledger-Based Platform for Renewable Energy Trading

NEBOJŠA HORVAT, (Member, IEEE), DUŠAN B. GAJIĆ, (Member, IEEE), PETAR TRIFUNOVIĆ, VELJKO B. PETROVIĆ, DINU DRAGAN, AND VLADIMIR A. KATIĆ, (Life Senior Member, IEEE)
Faculty of Technical Sciences, University of Novi Sad, 21000 Novi Sad, Serbia

Corresponding author: Dušan B. Gajić (dusan.gajic@uns.ac.rs)

This research has been supported by the Ministry of Science, Technological Development and Innovation (Contract No. 451-03-65/2024-03/200156) and the Faculty of Technical Sciences, University of Novi Sad through project "Scientific and Artistic Research Work of Researchers in Teaching and Associate Positions at the Faculty of Technical Sciences, University of Novi Sad" (No. 01-3394/1).

ABSTRACT Distributed ledger technology (DLT) and blockchain-based energy trading solutions are often labeled as resource-hungry, excessive power consumers which consequently create a significant carbon footprint. Using this approach to energy trading, without simultaneously minimizing energy consumption, defeats the purpose of using such trading solutions to encourage renewable energy production and may even cancel out any ecological benefits. This paper demonstrates that it is possible to create a DLT-based energy trading system which provides security, transparency, autonomy, scalability, and decentralization, provided by using a DLT, but with significantly lower penalties than other previously known solutions. This is best illustrated through the fact that the carbon footprint per transaction of the solution presented in this paper is 0.19 grams of CO_2 compared to 20 grams of CO_2 created by single transaction of similar complexity on the Ethereum blockchain. This makes per-transaction carbon footprint of our system approximate 100 times smaller than that of the most commonly used Ethereum blockchain. We present the architecture and features of the proposed platform, as well as a thorough analysis of its performance, including power consumption and estimated carbon footprint. All experiments are done on a dedicated Beowulf cluster comprised of general-purpose computers. The cluster mimics a microgrid environment and presents a testing ground for real-world performance and power consumption analysis of a system used for trading energy predominantly produced from prosumers and their renewable sources.

INDEX TERMS Energy trading, distributed ledger technology, distributed systems, renewable energy, smart grid, performance evaluation, blockchain.

I. INTRODUCTION

Welive in an era where environmental conscious [1], and sustainable living [2] are an omnipresent topic in global discussions.At the same time, we are witnessing a fast shift towards renewable energy sources [3]. While large-scale initiatives and governmental policies undoubtedly play an important role, the silent revolution is happening right at the heart of smaller communities, within individual households and small energy producers [3]. An increasing number of individuals are embracing renewable energy solutions. This movement is redefining the whole energy system, decentralizing power generation and distribution and putting the ability to harness sustainable resources directly into the hands of everyday citizens. From rooftop solar panels to backyard wind turbines, these small-scale energy producers are turning homes into power hubs and contributing significantly to the reduction of carbon footprints.

These changes are welcome, but they also introduce new challenges. These challenges encompass the whole

The associate editor coordinating the review of this manuscript and approving it for publication was Hao Wang.

technological spectrum including new ways of storing and delivering power, but also necessitating the need for truly decentralized systems of management, payment, and control. These distributed systems can be vastly more complex than centralized ones. Managing data in such systems in non-trivial and it requires more computational and storage resources. They are also more vulnerable which increases effort required to guarantee security. However, if these challenges are overcome, decentralized, distributed systems demonstrate high resilience, availability, and a near-limitless capacity to scale, which make the additional engineering and research efforts well worth the trouble.

A web platform, we called DREAM (Distributed Renewable Energy Automated Marketplace) and presented in its early form in [4], is one of possible approaches to the problem of distributed renewable energy trading. We find that DREAM might be a unique solution in a sense that it combines private distributed ledger technology (DLT) with an architecture that supports energy trading with reduced resource consumption. Based on broad field reviews by Andoni et al. in 2019 [5], and two studies by Khezami at al. [6] and Al-Abri at al. [7] from 2022 we can see that there is no shortage of similar solutions based on the use of public blockchain technology. Therefore we devised a series of experiments to measure performance of our system and compare it to other existing solutions in order to truly evaluate its performance. Of particular interest to us is power consumption and, consequently, the carbon footprint.

Due to the inherent overhead issues, it is common for distributed systems to have performance problems. This is even more characteristic when working with DLTs and blockchains in particular. The great emphasis on security and decentralization of these systems is often achieved at the expense of performance. While DLT performance may not be crucial in a lot of financial use cases, when it comes to renewable energy trading, performance becomes a very important part of equation. Small energy transactions that are the principal use-case for such a system concern monetary amounts that are fairly small which means that even limited transaction and gas fees can render them unprofitable for either party. Furthermore, it would be counterproductive to use a power hungry information system for tracking renewable energy transactions as it would go against the idea of maximizing renewable energy efficiency and impact.

Seeing as performance can make or break an energy trading system, determining whether it can be used in a real world scenario at all or not, we have developed and conducted an extensive battery of performance-focused experiments. We have measured both raw transaction throughtput, total memory use, but also measured direct power usage in selected cases. Experiments were performed using a small cluster configured so as to resemble a microgrid in organization. Experiments cover network performance in six different configurations, with the number of nodes ranging from 6 to 33. Further, we extended the original implementation of the system as presented in [4] with new node types and functionalities in order to measure scalability of the system and then we ran all six experiments again. Therefore, twelve total experiments were conducted and more than ten different performance variables were captured.

The current enhanced implementation of the DREAM system is likely its final version in many aspects, as it contains all features needed for information-rich power trading. It now allows energy storage providers to be a part of the network. They can buy energy when it is abundant, store it, and sell it later for a profit. Assuming the spread in the energy costs exceeds recurring storage costs and amortized storage facility costs, this can be highly profitable. Storage providers help stabilize a network with a lot of renewable producers since they can help mitigate the variation in the production capacity caused by outside sources. Additionally, the system supports verification agencies which verify producer claims regarding the manner in which their power is produced. These agencies are incentivised to earn and keep consumer confidence, and can help consumers who wish to purchase power whose production meets ecological, ethical, adn other criteria.

The DLT subsystem of DREAM is based on Corda [8] which provides all the advantages of using a DLT solution, such as improved security, distribution, and scalability, without introducing significant performance penalties. Unlike Corda, public blockchains are only now moving away from resource hungry Proof-of-Work (POW) consensus algorithms, and they still require a significant amount of resources to run properly [9]. We can show that DREAM, on the other hand, consumes orders of magnitude less energy than Ethereum-based systems which are currently predominantly used for solutions in the energy sector [5].

We find that the following are main contributions of the paper:

- A proposal of an architecture for a DLT-based renewable energy trading web marketplace which is very light on required resources and yet provides high throughput and overall performance;
- Power consumption and carbon footprint analysis of distributed energy trading systems based on experimental results;
- A proposal of a selection of performance metrics which can be used for the evaluation of DLT-based energy trading platforms;
- An approach, including data sets and algorithms, which can be used for characterizing the performance of DLT solutions through experiment, measurement, and analysis.

The remainder of the paper is organized as follows. In order to make the paper self-contained, in the next section we offer a short overview of distributed ledgers and blockchain as its subset. Then, in Section III, we discuss both current and future applications of DLTs in the energy sector and summarize the related work on DLT performance measurements and their energy consumption. In Section IV, we present the DREAM platform for energy trading, with

all its extensions. Afterwards, we use Section V to explain the experimental process we used for performance testing and present the infrastructure used. We then present the results and discuss performance and power consumption aspects of the system. We conclude the paper by summarizing its main contributions and sharing plans for future work.

II. BACKGROUND THEORY

This section offers a brief introduction to DLTs and blockchain. It also discusses the most important concepts in the Corda DLT which is used to implement the approach presented in this paper.

Distributed ledger represents a type of a distributed database in which data is distributed among the participants that are all part of an environment in which no level of mutual trust can be assumed [10], [11]. DLTs and blockchain are commonly used as synonyms, but, fundamentally, these are not the same. Blockchain is a specific implementation of a DLT, thus, ''every blockchain is a distributed ledger, but not every distributed ledger is a blockchain'' [12]. Specifically, a blockchain is an immutable, append-only DLT where data is stored in the form of a linked list of blocks of transactions and each block is cryptographically connected to its parent by hash computations [10], [13], [14]. The first ever blockchain is Bitcoin, originally described in [15]. The actions which occur on the blockchain are automated through smart contracts [16], which represent a specialized type of a computer program which defines the way and the conditions under which the chain of blocks changes [17]. In a DLT, and thus, in a blockchain environment, it is crucial for all the participants to have the same version of data stored on their private ledger copies. The component which ensures this global data uniformity is called the consensus mechanism. Such a mechanism ensures that each copy of the chain of blocks looks the same across the whole network [10], [18]. There is a great profusion of approaches to consensus algorithms but those most often mentioned and relevant to public blockchains are Proof of Work (PoW) and Proof of Stake (PoS). Both are based on ensuring that actors that help validate transactions as correct are acting in good faith by risking, as it were, something. In the case of PoW what's at risk is a large amount of computation, and in the case of PoS, it is a stake of a crypto-currency that is at risk. Either thing makes participating in the validation process expensive in some way which increases decentralized security but at a considerable cost.

Distributed ledgers can be categorized into four groups: public permissionless, private permissionless, public permissioned, and private permissioned. The two most frequently used are public permissionless and private permissioned. The former has its data open to public access and writing new data to such a ledger requires no special permission. This type of DLT is what usually comes to mind when the word *blockchain* is mentioned, with popular examples such as Bitcoin [10], [15] and Ethereum [10], [19]. The later represents a much stricter environment — the data is not available to anyone outside of the system, and, in order to become its participant, users must first be granted necessary permissions. These DLTs are mostly used in enterprise solutions, and the most popular examples are Hyperledger Fabric [20] and R3 Corda [8].

Corda differs from the standard blockchain structure, since it stores individual transactions on the ledger rather than using blocks of multiple transactions. Although it does not have a standard chain of blocks data layout, it still provides safe and secure data storage which is one of its core functionalities. Every node in a network has its own database called a *vault*. Contrary to traditional blockchains [15], [19], [20], the vault stores only those transactions its associated node participated in thereby reducing data redundancy and increasing performance [21]. Vaults store *states* which represent facts about the system and can be either spent or not. Spent states are replaced by updated ones and represent historical information about the system. States become spent and are replaced by new ones trough transactions while transactions are a product of *flows* [22]. Flows are Corda's equivalent of what other blockchain and DLT systems call smart contracts. Corda also has a structure it calls contracts, but these serve only to enforce limitations on which states and transactions are valid. Therefore they operate far more like their real-world counterparts then is typical for smart contracts which, in most other systems, both define validity but also do data processing.

Communication between two Corda parties (nodes) is also performed by flows by sharing unfinished transactions. Cooperating nodes that represent interested parties in a multi-lateral transaction create transactions and send them, still unfinished, to their counterparties which process, examine, and endorse (by signing) them. Aside from the parties of the multi-lateral transaction this communication also includes *notaries* which are specialized network participants introduced by Corda which monitor and validate transactions and serve as an extra layer of security not otherwise provided by normal counterparty interests. For in-depth understanding about the way Corda works refer to it's documentation [23].

III. RELATED WORK

The research in this paper is an extension of our earlier work presented in [4]. Although the first DREAM system was a solid starting point for testing our energy trading platform, that implementation was limited and tests were done only in a local environment. In this paper we significantly extend the implementation and introduce an experimental setup which can be used to produce results comparable to a real-world use-case of such a system. In this section we discusses related work in three directions. We first consider existing research on DLT applications in the energy domain. We then proceed with analysis of work done on blockchain performance measurements and conclude with related work on blockchain energy consumption in particular.

A. DLT APPLICATIONS IN THE ENERGY SECTOR

Use of DLT and blockchain has been in the rise across many different fields including the energy domain [5], [6], [7]. Several papers already present broad studies of blockchain use in electrical grids and in the energy sector as a whole. The study by Andoni et al. [5] offers one of the first systematic reviews of blockchain applications in the energy sector. Although Andoni's study is comprehensive, it was published in 2019, which can make some of it's aspects outdated since the field is evolving so rapidly. Therefore, we compared two newer broad field reviews [6], [7] from 2022 with Andoni et al.'s [5] 2019 work, in effort to obtain up-to-date field trends. The percentage of publications about applications of blockchain technology in the energy sector in shown in Fig. 1. Out of 769 articles, 25.1%, are focused on use of blockchain in energy trading, 14.44% on energy storage and 7,8% on smart girds [7]. Almost 50% of articles tackle problems which are also the subject of our study which offers us some confidence about the direction of our work.

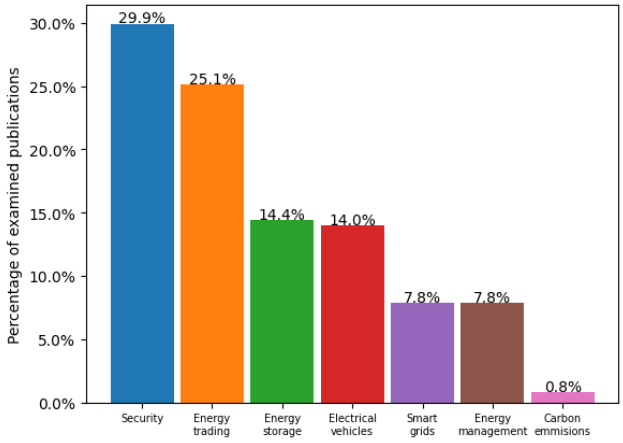

FIGURE 1. Distribution of publications on DLT use in the energy sector.

Results described in [5] and [6] both include classification of the DLT used in the energy sector. Both articles cover a wide range of papers and are published three years apart from one another which can help in tracking technology trends. Fig. 2 shows that 50% of 140 examined studies use Ethereum as technology of choice, followed by Hyperledger Fabric and Energy Web. Fig. 3 is based on a more recent study, and shows that 90% out of 769 articles use private or public Ethereum networks for building energy blockchains, while only 2,8% use Hyperledger Fabric. This indicates that trends lean towards heavier use of well established and battle-tested blockchain technologies like Ethereum. There are advantages to using well-established and battle-tested technologies as the basis of one's work, but it does mean accepting as immutable the fundamental disadvantages that come with their use. Public Ethereum networks share resources between all participants and can get congested in times of high demand. Private Ethereum networks solve this problem and can be used only be selected group of participants, but still do not provide enough throughput required for managing large energy systems. Ethereum scaling is only possible to a limited extent due to it's architecture [24]. Ethereum is evolving and it recently had an upgrade from a Proof of Work (PoW) to a Proof of Stake (PoS) consensus algorithm. That did make Ethereum more desirable as no excessive power is now needed to maintain the network [25], but it did not transform its performance statistics. Therefore, Ethereum is still not an ideal choice of technology if the requirement is a high throughput.

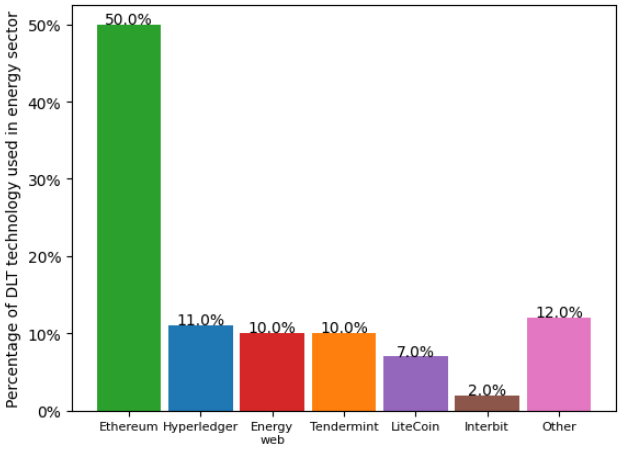

FIGURE 2. Distribution of DLTs used in the energy sector in 2019 [5].

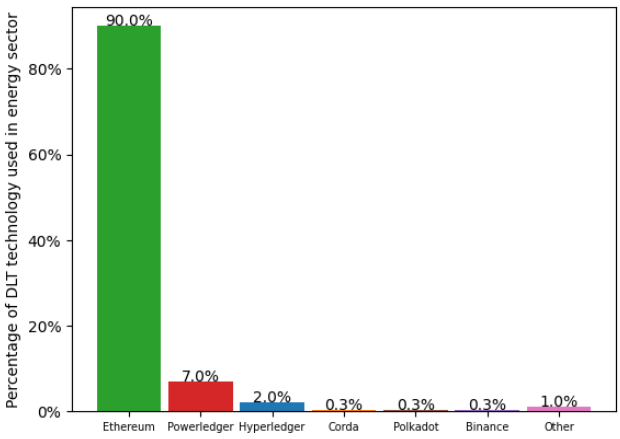

FIGURE 3. Distribution of DLTs used in the energy sector in 2022 [6].

1) EXISTING DLT-BASED ENERGY TRADING SOLUTIONS

One of the pioneering attempts at using DLTs for supporting energy trading is the Brooklyn Microgrid project [26]. They conducted a study which successfully implemented a peer-to-peer (P2P) energy trading system in a small microgrid in Brooklyn, New York, USA, and tested it for three months. The technologies which allowed prosumers to directly sell energy to other network participants were public Ethereum smart contracts combined with Practical Byzantine

Fault Tolerance (PBFT) [27] consensus implemented by Tendermint.

Power Ledger [28] represents a well tested and mature trading solution. It performs a wide range of services, from managing grid stability and flexibility to P2P energy trading, electrical vehicle solutions, etc. It is currently used by more than 30 clients in 10 countries. In order to avoid the sole use of public blockchains and to reduce energy consumption, Power Ledger uses a mix of public and private blockchains described in [28]. Additionally, they offer tokens like POWR and Sparkz which can be publicly traded and through which customers can make potential profits.

There is a considerable number of academic papers and enterprise solutions which apply DLTs for energy trading [29], [30], [31], [32], [33], [34]. Some solutions [31] still use a PoW algorithm which is power hungry and requires prolonged settlements. This problem is mitigated by use of credit-based payment schemata where nodes have a credit strategy and apply for loans accordingly. The research presented in [34] uses Ethereum smart contracts for an energy trading schema on Virtual Power Plants (VPP). Ethereum's transition from a PoW to PoS algorithm does make this solutions a bit more attractive, but the use of the public blockchain network can still cause high fee payments. Other solutions, like [29] and [30], use a private permissioned Hyperledger Fabric system for energy P2P trading. Authors in [30] utilize a two-phase algorithm in which the first phase schedules energy consumption days ahead, while the second phase schedules consumption hours ahead and is focused on real-time network management. Further, in [29] entities such as a energy node, energy aggregator, and smart energy meters are recognized. Energy nodes sell or buy energy according to their needs and energy status, while energy aggregators serve as brokers that regulate the trade. General Electric is investigating use of Digital Power Plants [35] (DPP) which will also use Hyperledger Fabric for power plants with small and medium generators in order to satisfy the diverse needs of power assets with high-speed and intelligent infrastructure.

Further examples of DLT-based energy trading solutions can be found in academic research [36], [37], [38], [39], [40]. Hamouda et. al. [36] use smart meters to create transacting energy market framework. They implement a market model solution which creates both expected energy transactions as well as monitors real-time transaction data from smart meters. All of these are bundled in blocks that are a part of a custom blockchain solution coded in Python. The authors in [37] present a solution that allows prosumers and other small producers to offer their excess power through an intelligent brokerage that is based on Ethereum blockchain technology. In [38], a blockchain based private solution that works as a secure mediator between local micro grid controllers and utility operators is presented. It strives to solve the problem of power flow in a distributed network implemented using private Ethereum solution. The authors in [40] use a simulated Ethereum blockchain and execute its smart contracts to coordinate the decentralized approach to the economic dispatch problem of distributed generation units.

B. BLOCKCHAIN PERFORMANCE MEASUREMENT

Although early efforts in blockchain applications were primarily aimed at maintaining high security (Fig. 1) and achieving decentralization, the popularity of this form of distributed systems also led lately to performance becoming of great importance. Public blockchain systems in general are not great at quickly processing large amount of transactions, which is a problem caused by their design and the need for maintaining high levels of security and decentralization. Vitalik Buterin introduced the concept known as the blockchain trilemma, stating that a blockchain system can only have two properties of the following three — scalability, decentralization, and security [41]. Recent work aimed towards resolving this problem in public blockchain environments involves solutions such as switching to other, less computationally demanding consensus mechanisms [42], partitioning the ledger [43], or adding new layers to the chain [44]. These solutions do not eliminate the problem, but rather aim to mitigate it, since they are always present in blockchain systems as currently conceptualized by their very nature and cannot be completely eradicated. However, no matter which approach to mitigation is taken, public blockchains can only choose to sacrifice scalability, and occasionally some level of security, but rarely decentralization. Private blockchains, on the other hand, are designed in such a way that they are not fully decentralized, and sacrificing strict decentralization enables transactions to execute faster. This results in private blockchains having higher throughput and lower latency. Such a claim is supported [45] through a detailed performance comparison of Ethereum and Hyperledger Fabric. According to [46], the throughput for Bitcoin on 4 October 2023 was, on average, only 3 transactions per second (TPS), and for Ethereum it was, on average, equal to 11 TPS on 30 September 2023. Comparing public to private blockchains just by TPS cannot be considered a fully objective measure, since the environments in which these blockchains are running differ greatly and the transactions are too heterogeneous. However, certain sources [47], [48], [49] state that Hyperledger Fabric and R3 Corda can, in certain situations, achieve thousands of TPS. These numbers are strongly dependent on the configuration of the network, number of participants, consensus protocol chosen and similar factors, but can serve well enough to show that private permissioned blockchains should be considered and may have performance advantages over public blockchains in certain scenarios.

As far as private permissioned blockchains are concerned, several studies compare the performance of some of the most popular frameworks. In [50], an evaluation of five blockchains is given — private Ethereum deployment, Hyperledger Fabric, Quorum, Multichain, and R3 Corda. Each of these is given a final grade on a scale from 1 to 5.

Hyperledger Fabric was given the highest final grade of 4.4, Ethereum got 3.3, Corda and Quorum both got 2.2, and Multichain was given a 2. Although a methodology with explicit grading instructions was proposed, the grading system is not built upon an experimental setup, but rather on the information acquired from other research papers. This does not give an objective insight into the performance of these blockchains, since the information about it originates from various sources and different experiments and tests. Moreover, the final grade is a mix of technical performance metrics, such as latency and throughput, and non-technical metrics such as community activity on Github and Twitter, and the rate to which the blockchains are commercially adopted and used by big corporations. Further, the number of research papers used as the source of information is not balanced among the five blockchains — 8 papers were used for Fabric, 6 for Ethereum, and only one for each of the remaining three. Thus, the final grades cannot be considered absolutely relevant. The only paper used in [50] to evaluate R3 Corda is [51], where Corda shows the worst performance among the evaluated blockchains. The blockchain frameworks compared in [51] are Hyperledger Fabric v0.6, Hyperledger Fabric v1.0, Ripple, Tendermint, and R3 Corda. The Corda version used in this evaluation is v3.2 with Raft consensus used for Corda Notaries. Each framework was scaled to about 30 nodes, except for Corda, which could not support more than 4, but the v3.2 version is from the year 2018, and the newer Corda versions perform much better than this one, with improved scalability and overall performance [48].

The work in [21], [51], and [52] consists of a well-described experimental setup and a description of the used methodology, and thus gives a more thorough comparison of private blockchain solutions than [50]. Further, [51] does not focus on comparing different blockchains, but rather on measuring their scaling capabilities which, a few years ago, were significantly worse than today. On the other hand, [21] and [52] do compare different private blockchains. In both of them, Hyperledger Fabric is the solution that shows the best overall performance. The methodology proposed in [52] can be used to measure latency, throughput, total execution time for a batch of transactions, and the maximum number of concurrent transactions. In the presented work, the methodology compares a private Ethereum instance and Hyperledger Fabric, observing their behaviour when the number of requests sent to the blockchain for processing is increased. The only aspect in which Ethereum performs better is in the number of concurrent transactions it can handle, while Hyperledger Fabric performs significantly better in terms of latency, throughput and the total execution time. The results in [21] are quite similar, but the experiments done here are somewhat extended. In [21], four blockchains are compared — Hyperledger Fabric, private Ethereum, R3 Corda and Quorum. The experiments vary both in the number of initiated transactions and in the number of nodes included. Hyperledger Fabric again shows the best overall performance, although it can scale up to 16 nodes with 1000 concurrent transactions, and only 4 with 10000 transactions, while other blockchains can handle more nodes. The second best in this evaluation is R3 Corda. This paper also shows the significant improvement in Corda's performance which occurred in only three consecutive version upgrades (v4.3, v4.4 and v4.5). Despite being the second best, Corda is not far away from Hyperledger Fabric, and even shows better throughput when the number of concurrent transactions is in the range of 10. The results from [21] prove that R3 Corda performs much better than it did in the earlier versions evaluated in [50] and [51].

As far as tools for benchmarking blockchains are concerned, [53] gives a good overview of the existing frameworks which can be used to measure performance of blockchains. Some of the mentioned tools are BlockBench [54], BCTMark [55], and Hyperledger Caliper [56]. BlockBench [54] is a tool for benchmarking private blockchains, and measures latency, throughput, scalability (how latency and throughput change with the addition of new nodes) and fault tolerance (how latency and throughput change with node failure). When [54] was published, the tool supported Hyperledger Fabric, private Ethereum deployments and Parity. It was later expanded, and, as stated on BlockBench's github repository,[1] it supports multiple Hyperledger Versions, private Ethereum, Parity and Quorum. BCTMark [55] is another one used to benchmark different blockchain platforms. It can measure network-related parameters, such as latency and throughput, as well as infrastructure-related ones, such as power consumption and CPU usage. It is developed in a way which abstracts both the underlying infrastructure, and the blockchain system it is used for, making it portable to various devices and reusable for different blockchains. Initially, it was created to support Ethereum Ethash, Ethereum Clique, and Hyperledger Fabric. In order to use it for other blockchains, developers must implement an adequate driver. Hyperledger Caliper [56] is a tool which can, currently, be used to benchmark Hyperledger Besu, Hyperledger Fabric v1.x and v2.x, Ethereum, and FISCO BCOS. It can measure throughput and latency for read and write operations, as well as resource consumption, such as CPU, memory and network I/O.

As it can be seen from the description of the beforementioned tools, none of them supports R3 Corda. In the official Corda documentation, however, there is some information about collecting useful data about the network's properties, nodes' execution, and monitoring of the system. In [57] a list of Corda node's attributes is given, each of which can be used to monitor a specific part of a node's execution. A test suite which uses a Corda-adapted wrapper for Apache Jmeter [58] in order to test the transaction execution and measure throughput is described here [59].

All of the described tools require additional setup steps or driver implementations and cannot be used out-of-the box to measure arbitrary performance metrics of an R3 Corda

[1] https://github.com/ooibc88/blockbench, last accessed on 3 April 2024

network. This is why we decided to use our own experimental setup and write custom algorithms and scripts in order to monitor the specific metrics. Details of these measurements and the experimental environment are described in V.

C. BLOCKCHAIN ENERGY CONSUMPTION

Impact of blockchain on the climate and the environment has become a topic of great interest. Research results from [60], [61], and [62] show that the early blockchain platforms which use PoW as the consensus protocol are not very energy efficient. Calculations from [60] and [61] predicted that in 2020 the annual energy consumption of Bitcoin would be between the amount of energy which Austria and Norway consume per year (in the range from 75 GWh to 125 GWh). The data available in [63] shows that their predictions were not far off, and that Bitcoin's energy consumption has increased further from the one measured in 2020. As described in [60] and [61], other Proof of Work blockchains consume a bit less energy than Bitcoin, but their consumption is still very high. Although such behavior is not desirable, [60] argues that it is the result of PoW being designed to be computationally intensive in order to maintain high levels of security. Many other consensus protocols are much more efficient in terms of the required energy, such as PoS [25], or other mechanisms which are less strict and used in private permissioned blockchains, such as PBFT [60], [61], [62]. In [62], an overview of various consensus protocols is given, with a brief analysis of their energy efficiency. Apart from the consensus mechanisms, what influences the amount of required energy is the redundancy present in the system. It can manifest as computational redundancy (the number of nodes which participate in computation tasks in the network) and storage redundancy (number of data replicas) [60]. In certain private permissioned blockchains used in enterprises, not all of the nodes must validate and confirm each transaction, and data is not necessarily replicated on all of the nodes. This results in enterprise blockchain systems being more energy efficient than public ones [60]. Such behavior is also true for R3 Corda, since nodes do not record the whole ledger history, but only the data they are interested in. Furthermore, in Corda, the consensus for a transaction involves only the interested parties, instead of the whole network. All of this implies that Corda's energy consumption is lower than that of public permissionless blockchains. A power consumption comparison of Corda to other blockchain systems like Bitcoin, Ethereum, and Polkadot can be found in [64]. Although the energy consumption depends on a lot of factors, such as the workload, the number of participants in the network and the infrastructure used, the results in [64] show that the amount of energy used in Corda is significantly lower than in Ethereum, Bitcoin, and Polkadot.

IV. SYSTEM DESIGN

This section describes the structure of the system. First, we present the overall design of the Distributed Ledger-Based Automated Marketplace (DREAM). Then we discuss the extensions done on the original implementation.

A. BASIC DREAM IMPLEMENTATION

This subsection is fundamentally an overview of the system structure we originally presented in [4]. We offer this brief description here in order to make this paper self-contained and to explain the details of the extended system.

1) STRUCTURE AND PURPOSE OF THE DREAM NETWORK

All participants in the DREAM network are connected through an environment structured as a single grid. This grid provides participants with the ability to trade electrical power. This trading is made possible through the use of promises. The producing party promises to supply a specific amount of power, and thus creates what we call a *power promise*, while the consuming party promises to use the power it buys. Ideally, the participants of the trade fulfill their promises, which results in transferring money to the power producer which in turn provides the promised power to the consumer.

In a real-world scenario, the sides involved in the trade may default on their promise. Regardless of whether this is caused by an error or by potentially malicious intent, the system provides a way of recovering from potential power surpluses or shortages. A network party called the *grid authority* is responsible for monitoring the network, connecting the digital state of the DREAM network with the real-world grid state and keeping track of whether the participants have fulfilled their end of the bargain. If not, the grid authority acts accordingly — in case of the producer failing to provide the energy, the authority makes sure that a last-resort provider makes up for the power shortage; in case of the consumer failing to use what it promised to, the excess energy is accepted by the grid authority. The DREAM network as a software system deals with processes in the grid from the time of power promise creation, to the point in which it is delivered to the consumer. The actual way in which the energy surplus is stored is a part of the physical infrastructure of a grid and does not directly influence the software implementation of the system described.

Regardless of the system's ability to respond to unfulfilled agreements, there can be a way in which the participants are discouraged from make unrealistic promises. This is necessary because constantly referring to the last-resort energy providers or relying on energy storage is expensive. This mechanism has not yet been fully implemented since DREAM testing in this phase wasn't focused on economic incentives, but its details have been fully developed. A calculated amount is deposited to Grid Authorities escrow account and to be used only in case a producer fails to deliver energy as promised. The higher the promised amount of kWh, the higher the amount the producer has to put into escrow and the higher the penalty they will potentially pay if the promise is not fulfilled. This amount will be used by the Grid Authority to pay the supplier of last resort. The amount is calculated

by using the worst case historical price for last resort energy and increasing it by further 30%. All this is done in an effort to make sure that the Grid Authority suffers no losses when dealing with unfulfilled power promises while strongly financially incentivizing the producers to make only those promises they are able to keep. If a power promise is kept, then the full escrowed amount is available for withdrawal or reuse as the backing for another power promise.The consumer, too, must be discouraged from making promises they cannot keep. A power grid must have an equal sum of inputs and outputs or it cannot function correctly. Therefore, it is just as much trouble to have an excess as it is to have a shortfall, and the consumer must be prevented from causing excesses by failing to utilize that which they have bought. This can be done by limiting their purchasing power by the size of the security deposit they have lodged with the system. In case they do not utilize their power promise (or sell it on themselves) they would be penalized for the costs of handling this excess power by the consumers of last resort.

The specific way in which the DREAM network enables power trading is through an auction system. Any participant capable of power production can create a power promise and put it up for auction. Other network participants can then become auctioneers by placing bids and competing to buy the power promise. When the auction deadline passes, the winner, that is, the highest bidder, becomes the owner of the power promise. In a real-world scenario, the power should be delivered to the winner after a specified amount of time, and the whole process would be monitored by the grid authority. A power promise represents both the right to a given amount of power and the obligation to accept this power as specified over a certain period of time.

An important part of the structure of the DREAM network is a component which enables communication between the outer world and the core of the network. We refer to this component as *the proxy*. This component creates a bridge between the operators which represent the network participants in the real world and the software representation of these participants in the DREAM system. The proxy exposes a REST API [65] and a set of endpoints which can be accessed from outside the internal DREAM network. In the prototype version, one instance of this proxy is capable of contacting all the participants of the network and forwarding them requests as needed. In a real-world installation, multiple proxy instances would be required in order for the network to be properly scaled. Any action the operators of the network participants wish to perform must start by invoking the proxy which transforms these invocations into a form that the DREAM network understands. These requests are then processed and further translated into Corda flows described in IV-A3.

2) PARTICIPANTS IN THE DREAM NETWORK

This subsection gives a brief explanation of the participants which exist in the original DREAM network and which are more thoroughly described in [4]. New types of participants are added and explained in detail in subsection IV-B. The architecture of the complete DREAM system, which includes new participants, is presented in Fig. 4.

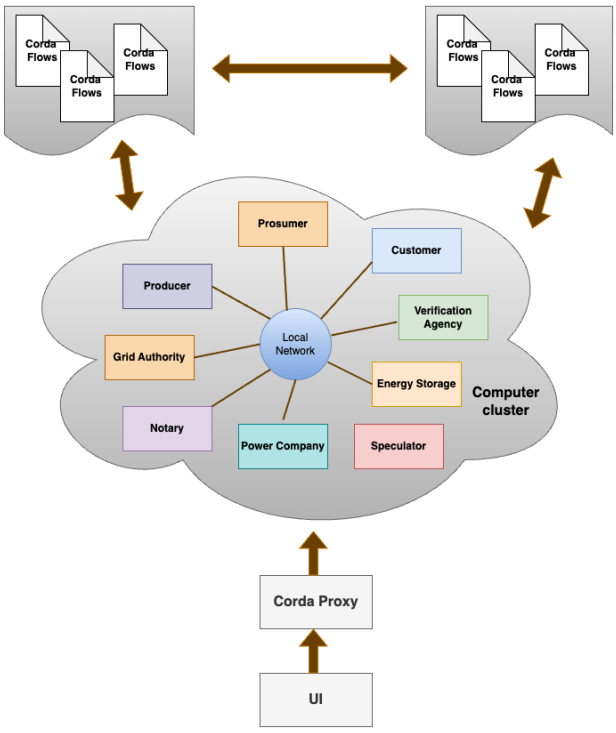

FIGURE 4. A high-level overview of the architecture of the DREAM system.

- **Grid Authority**. The grid authority is an institution responsible for the grid, which oversees the network and connects the real-world structure to the software implementation. It is the only party which is centralized and trusted by other participants. The necessity for such a centralized party in an essentially decentralized network is better explained in [4].
- **Power Company**. The power company is a large producer that can both buy and sell vast amounts of power in relation to the total throughput of the DREAM system. Commonly there may be just one in which case it is the producer and consumer of last resort (and shares an off-chain identity with the grid authority), but the system must adapt to a situation where multiple such companies share a market in which case the grid authority takes the role of consumer and producer of last resort.
- **Producer**. Producers are enterprises whose purpose in the network is to create the electricity in order to sell it.
- **Prosumer**. Prosumers are individuals, not enterprises, which produce a small amount of energy (compared to the producer) for personal use, and sell the rest.
- **Consumer**. An individual or an enterprise which participates in the network only in order to purchase and use the energy.

3) TYPES OF FLOWS IN THE DREAM NETWORK

The processes and transactions which occur in the Corda network which the DREAM system relies on are performed through what is known in Corda as 'flows' [22]. A flow represents a way to automate the execution of series of steps in a Corda network and these steps instruct the nodes on what actions to perform in order to update the state of the system.Using Corda flows increases the speed of transaction processing compared to traditional blockchains by reducing the number of participants that have to partake in a flow. Each flow can have an individual list of participants and each of them can execute unique code after receiving a transaction. This allows us to target only specific parties in the distributed system that truly need to be part of the transaction. Because only the flow participants need to save the executed transaction, Corda flows reduce the redundancy in the system.

This subsection gives a brief overview of the Corda flows implemented in DREAM:

- **Create power promise** — represents a flow during which a producing participant of the network creates a promise to supply the grid with a certain amount of electricity at a specific time. Participants which can create power promise are: producer, prosumer, and power company. The output of this flow is a newly created power promise which can be put up for auction by the owner of the promise.
- **Create power auction** — represents a flow for putting a power promise up for auction. This can be performed by the party which owns a power promise that has not yet expired, that is, a power promise whose delivery time has not yet passed. Participants which can create power auction are: producer, prosumer, consumer and power company. The output of this flow is a new auction which other participants can bid on.
- **Bid on power promise** — represents a flow for placing a bid on an active auction. This flow cannot be started by the grid authority, by the party which created the auction and by a participant of the type Producer. In the case of the first bid for an auction, the price offered in the bid must be equal or higher than an amount called *base price* which is set by the auction creator. If not the bid is not the first, the price must be higher than the best price offered so far. The result of this flow is a new highest bidder.
- **Auction settlement** — represents a flow which ends an auction. It can be started only by the highest bidder and only after the auction deadline has expired. The owner of the power promise gets the money from the highest bidder, and the highest bidder becomes the new owner.
- **End auction** — represents a flow which marks an auction as inactive. This flow is automatically triggered by the creator of the auction when the auction deadline is reached. The result is only the change of the state of the auction from *active* to *inactive*.
- **Check delivery** — represents a flow which ends the whole process of trading power. It is automatically initiated by the grid authority when the power promise deadline is reached. The grid authority then checks if the right amount of power was delivered by one and used by the other party. In the prototype version, the fulfillment of the promises by the two parties is randomly determined, while in a real-world scenario the grid authority would use the electrical grid infrastructure in order to check whether the parties have indeed held their ends of the agreement. If the power promise is fulfilled, the deposit held in escrow is returned to the power promise creator. Otherwise, the money is given to the Grid Authority/Power Company which supplies the promised power instead.

It should be noted that the grid authority can not initiate any of these flows except for delivery checking, which starts automatically. A much more detailed description of these flows is given in [4].

B. EXTENDED DREAM IMPLEMENTATION

In this subsection, we describe the extended version of the DREAM network. In essence, the system serves the same purpose as explained in previous subsections. The original version of DREAM was extended by adding certain metadata fields, and by introducing several new types of participants, which in turn resulted in the extension of the logic behind certain flows.

1) RICH POWER METADATA

New metadata fields are added to the implementation of a power promise in order to give information about the manner of production of the energy being sold. This gives participants of the grid insight into the origin of the energy they are consuming and gives them an opportunity to prioritize renewable energy sources.

The first metadata field added is the *production manner*. As mentioned, this informs the rest of the grid how the energy is produced.

The second field added to enrich the description of a power promise is a field which can take the value of either true or false. If true, it indicates that the creator of the power promise claims that the energy the promise refers to is created from a renewable source. If false, it indicates that the source is not renewable. This field is meant to be combined with the first one. This combination of metadata fields enables specialized agencies to verify or disprove claims by producers about the manner of the energy production.

Current state of these metadata fields is used in our prototype of the system in order to enable proper implementation of a new participant type, called *verification agency*, as explained in the next section. The values for these metadata fields were randomly determined in our experiments in order to simulate expected scenarios. A real-world application would require a more sophisticated method of labeling the energy production manner.

2) NEW PARTICIPANTS

New participants added to the system allow us to better replicate possible scenarios in a real-world microgrid. These participants are:

- **Energy storage provider**. As explained in [4], energy storage providers resemble prosumers in many ways. Both of these are capable of creating and trading power promises. The difference between them arises in the manner in which this behavior is reflected on the real world. Unlike prosumers, energy storage providers should not be using any power for personal needs. Instead, the power it buys and accepts it stores in some manner. Later, trading on the stored power, it can create power promises. This allows it to, on one hand, provide greatly improved liquidity to the system, and on the other reap economic benefits since it can buy power when there is a surfeit of it, and sell when there is a shortfall.
- **Verification agency**. Verification agencies are similar to the grid authority in that they too should be trusted by other participants. However, while the grid authority is a centralized unit which should be trusted in advance, the verification agency must earn this trust throughout its lifetime. These agencies verify the claims by the producers on the manner in which the power is produced. A verification agency should perform certain examinations in the real world and put itself in a suitable position to verify or disprove producers' claims. Performing bad examinations could lead to wrong verification results which would decrease the level of trust that other participants are willing to put into verification agencies. Thus, these agencies have no interest in deceiving the rest of the network, since this would result in low credibility for the agency. Verification agencies are directly included into the process of creating a power promise in the current DREAM implementation. The specific way in which they take part in the power promise creation flow is explained in the next subsection.
- **Speculator**. The speculator participates in the network solely for the purpose of profit. It does so through trading existing power promises. This, again, seems similar to prosumers, or even more to energy storage providers. However, prosumers and storage providers can buy a power promise and wait until the energy is delivered, without bearing any consequences. This is prohibited for a speculator. As mentioned, a speculator can only trade power promises, not the power itself. Additionally, the speculator cannot create a power promise, only buy one, since it cannot act as a producer. Also, other restrictions should be made for speculators in order to limit the allowed volume of trading. This should be done in order to prevent speculators from holding the primacy in trading on the network and overpowering individuals willing to trade energy for their private homes and businesses. The way in which speculators influence the Corda flows in the network is explained in the next section.

Extending application implementation with environmental considerations can benefit all parties involved. It allows for decentralized trading of power and incentivizes the collaboration between traditional energy companies and prosumers in order to maintain a stable energy grid. Additionally, regulators are given a platform through which they can reliably check the source of produced energy and its carbon footprint. The mentioned improvements increases collaboration and utilization of renewable energy sources in order to reduce CO_2 emissions globally.

3) EXTENSIONS IN CORDA FLOWS

The introduction of new participants in the network influences certain parts of logic in the existing Corda flows. The addition of the verification agencies extends the flow of creating a power promise. When a promise is created, it is sent to a verification agency (or multiple, if present) for examination. The agency checks the values of the two added metadata fields — the production manner, and the one which simulates the producer's claim on whether the power comes from a renewable source or not. In the current prototype, the production manner is randomly set to either 'solar', 'thermal' or 'wind'. The other field is randomly set to 'true' or 'false'. If the manner is 'wind' or 'solar', the agency does not check the other field's value, since a renewable source is considered a benefit regardless of the producer's claim. If the manner is set to 'thermal', then the agency verifies the creation of the power promise only if the other field is set to 'false', showing that the producer in a way ''admits'' that the energy source is not renewable. The way in which verification is done is through signing. If the agency signs the transaction, the flow continues and the promise is verified. Otherwise, the flow is stopped and the power promise does not get created.Of course this is merely a simplified system meant to simulate the real world. In production, not all verification agencies would take part, and an unverified power promise, i.e. one not attested by anyone but the producer or prosumer, would be permitted, just less trusted.

The introduction of speculator nodes changed the power promise creation flow, and the flow for the delivery check. The power promise is extended by checking if the flow was initiated by a speculator. If yes, the flow is stopped, since the speculator is prohibited from creating a power promise. The flow for delivery check is the last flow which, in a way, completes the life cycle of a power promise and is triggered when the power promise deadline is reached. After the delivery of the promised power, it is checked whether the owner of the power promise is a speculator. If yes, the speculator pays a fine, since it is prohibited from buying energy and can only own power promises before they get delivered.

V. EXPERIMENTS

We conducted performance and memory use experiments on both basic and extended implementations of the DREAM system. Some basic performances measurements were described in [4], but all experiments were done solely on one computer. Experiments presented in this paper use an actual distributed network of nodes. The DREAM system is designed and created using Corda DLT technology which is meant to be used on a network of connected nodes. Systems can be tested on a single machine like in our previous paper, but such setups should be only used for preliminary results, since they do not take into account network latency, time synchronization, different physical node machines, inconsistent node data and other overheads associated with distributed systems.

Therefore, we created a distributed environment in which we could properly test DREAM's performance. In order to better capture the scaling capabilities of the application, we performed twelve experiments on a Beowulf computer cluster described in the next subsection. The first six experiments were done using the original implementation introduced in [4], while the rest use the same cluster to test the extended implementation of the application IV-B.

A. EXPERIMENTAL SETTINGS

Developing applications on public blockchains (and similar publicly accessible distributed systems) is relatively easy from a developer's perspective as the underlying architecture is provided by the network itself. Therefore, the whole effort there is focused on implementing only application logic and, in some way, paying for the use of the network infrastructure (in most cases with native cryptocurrency). This can initially seem as an advantage but it inherently makes it impossible to accurately predict performance or make any kind of performance guaranties. Additionally, operation costs of public blockchains vary with demand. In times of increased demand, different users are practically out-bidding one another for resources and even the highest bidders can feel performance crunches. This behavior can increase operation costs of applications up to a factor of ten and make effective performance testing quite impossible and non-deterministic [66]. Contrary to public distributed ledger networks and blockchains, private ones are fully managed by network owners which are in charge of the setup and maintenance. This is challenging form the perspective of network managers and the barrier for bootstrapping such a system is high because of extensive work needed to start and setup all the nodes. But, when initial work is done, the higher level of stability, control and cost efficiency can be achieved when using private distributed ledgers. Therefore, our decision to choose Corda as the underlying technology for developing the DREAM application brought with it full control over the network and opened up a possibility for producing deterministic and precise performance and memory use metrics.

1) EXPERIMENTAL INFRASTRUCTURE

All experiments were performed on a Beowulf cluster composed of four general purpose computers. Different computers run different Corda nodes, so, for the sake of convenience we named them: Omega, Alpha, Beta and Gamma. Because Corda network participants are called nodes and computers which are part of the cluster are also called nodes, in order to avoid confusion, in further text we will call Corda nodes simply *nodes* and nodes in the Beowulf cluster *physical nodes*. A high level overview of physical nodes and their connections in the cluster is presented in Fig.5. All physical nodes are connected through a local network and can communicate with one another. Omega is additionally given a public IP address and it acts as a gateway to the rest of the network. Therefore, to connect to the network from outside, Omega is used as an entry point. Consequently, the proxy web server which is used to interact with the DREAM network is also launched on the Omega physical node. From there it can access all other Corda nodes in the network while also being easily accessible to all other applications and users who want to use network trough the proxy's REST API.

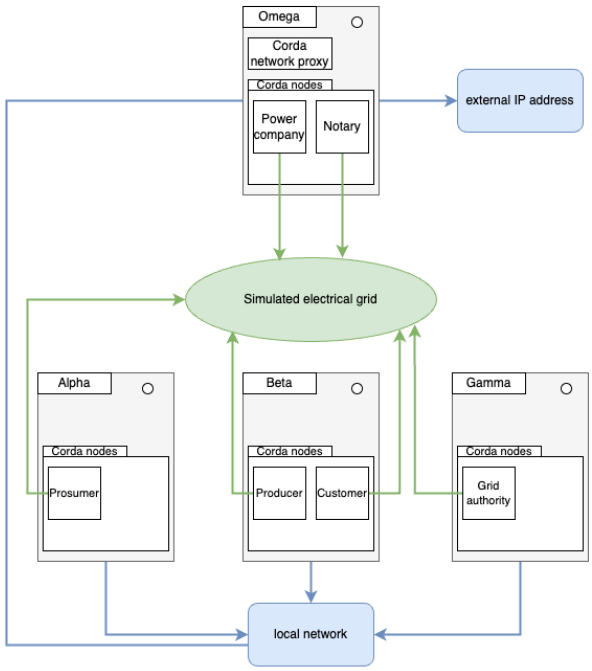

FIGURE 5. Cluster infrastructure for experiment auto-exp-0.

In all experiments, each physical node in the cluster is exclusively used to run different Corda nodes. All software nonessential for tests is terminated on all physical nodes. This will probably not be the case in a real-world setting, but we decided to go with such a setup in order to make accurate measurements and to try to reduce noise.

Fig. 5 represents state of the network in the first experiment we conducted (auto-exp-0). In it, Omega hosts the Power

company and Notary Corda nodes, Alpha hosts only the Prosumer, Beta hosts the Producer and Customer while Gamma hosts the Grid authority. Each following test only increases the number of Corda nodes on each physical node in the cluster, and the general infrastructure is not changed. One physical node can hosts up to 7 Corda nodes with a total maximum number of 33 Corda nodes in the whole network. This should represent a middle sized microgrid and help us see how the application performance scales as we increase the number of nodes in the network. As we increased the number of nodes we also made sure that the type of nodes present in the experiments will mirror real world scenarios. Prosumers, customers and producers are represented in greater numbers than power companies and producer entities which are expected to be less numerous in the real world scenarios.

All the Corda nodes in the network should be in some way connected to the power grid in order to monitor power production and consumption. Gathered data represents input to the system and requires smart electricity meters. Our experiments simulate a connection to the power grid which makes the performance testing much faster, safer, and easier.

The physical nodes in the cluster are general purpose computers and not all of them have identical configurations. Such a setup more closely resembles the real-world scenario in which different participants run DREAM nodes on their computers which do not necessarily have the same specifications. The detailed specification of the physical nodes used in the testing is described in Table 1.

TABLE 1. Configuration of nodes in cluster.

Node	Omega	Alpha	Beta	Gamma
RAM slots	8	4	4	4
RAM slots used	2	2	2	2
RAM installed	32 GiB	32 GiB	32 GiB	32 GiB
RAM type	DDR4	DDR4	DDR4	DDR4
RAM speed	2133 MHz	2400 MHz	2400 MHz	2400 MHz
CPU model name	Intel(R) Xeon(R) CPU E5-1620	AMD Ryzen 7 1800X	AMD Ryzen 7 1800X	AMD Ryzen 7 1800X
CPU architecture	x86-64	x86-64	x86-64	x86-64
CPU Physical address size	46 b	43 b	43 b	43 b
CPU Virtual address size	48 b	48 b	48 b	48 b
CPU Cores	4	8	8	8
CPU threads per core	2	2	2	2
CPU number of logical cores	8	16	16	16
CPU max freq.	3600 MHz	3600 MHz	3600 MHz	3600 MHz
CPU min freq.	1200 MHz	2200 MHz	2200 MHz	2200 MHz

All four nodes in the cluster have been equipped with the same software. The nodes are running a Ubuntu Server 22.04 LTS which is the only operating system installed on all the nodes in the cluster. The remainder of the used software and their respective versions are shown in Table 2.

TABLE 2. Software used on nodes.

Use	Software	Version
Operating system	Ubuntu Server	22.04 LTS
DLT infrastructure	R3 Corda	4.8
Running Corda nodes	Java	8u321
Running proxy server	Spring boot	2.1.3.
Gradle	Building Corda nodes and proxy	5.0.12

For measuring power consumption and other electricity metrics, we used a C.A. 8336 (Qualistar+) [67] with three-phase network analyzer and color graphic display Fig 6. It can have three roles and be used to:

- Measure RMS values, powers, and perturbations of electric distribution networks.
- Deliver a snapshot of the principal characteristics of a three-phase network.
- Track the variations of various parameters over time.

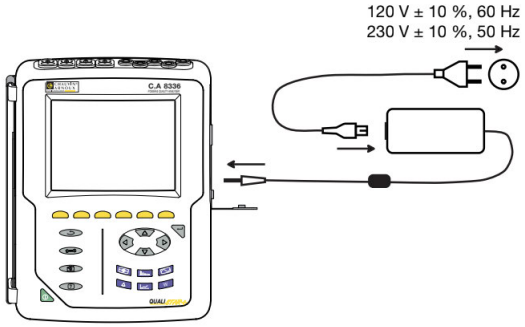

FIGURE 6. The C.A. 8336 measurement instrument [67].

Our main use of the instrument was to track the power consumption of the computer cluster over time. All the four physical nodes were connected to one power source and the instrument was measuring power characteristics of the whole network of nodes. A schema which shows how we connected the measurement instrument to our cluster is shown in Fig 7.

B. EXPERIMENTAL PROCESS

In this subsection we describe the experimental process and explain reasoning behind measurements we conducted. We performed two different sets of performance measurements. The first set was focused on performance of the network from the computer science perspective. We measured the use of computer resources while performing different kinds of transactions on the DREAM platform. In the second set of experiments we used the instrument to measure power consumption and other characteristics of the electrical grid which was used to power the platform during some of our experiments performed in the first set.

System is created in a way that each participant in the network is represented by one Corda node. As the network is distributed and new participants are incentivized to join, we concluded that this will be the most common change happening in the network. Therefore, as natural

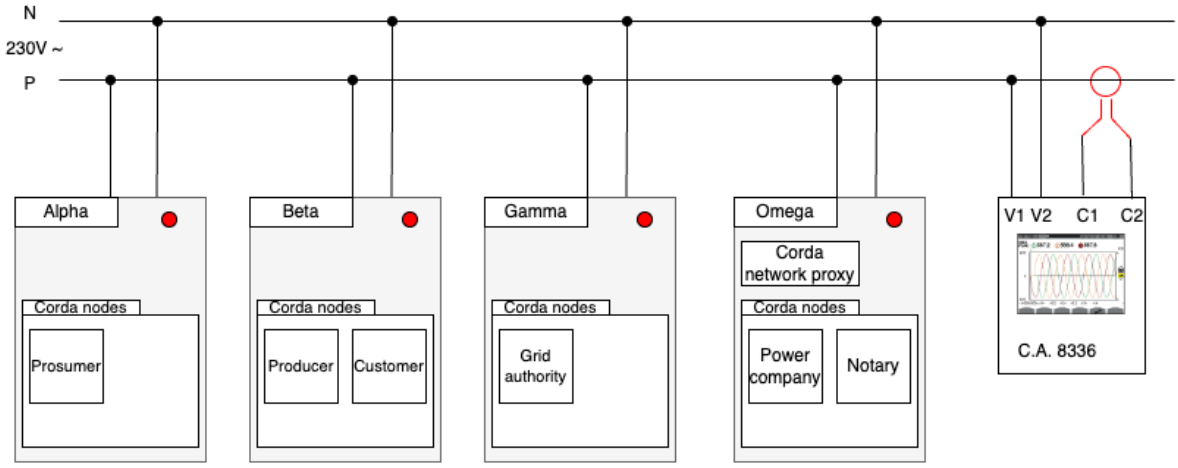

FIGURE 7. Power measurement schema .

scalability of the network is manifested by an increased number of Corda nodes, this was the first thing we set out to test. Conveniently, increasing the number of nodes in the distributed system is generally a good strategy for testing its scalability. We followed performance metrics from the minimal sized network of 7 nodes to the maximal size network of 33 nodes. Additional increase of Corda node numbers required hardware improvements which are not currently possible. Furthermore, we theorized that existing experiments were enough to distinguish trends in resource consumption which ended up clearly forming and are further discussed in Section VI.

Twelve experiments which measure the use of computer resources were done in total and they are clustered in two batches. The first batch consists of six experiments and they differ in the amount of Corda nodes which are started up on each physical node. The conducted experiments are:

- **auto-exp-0** - Represents the minimal number of nodes which need to be started up in order for the DREAM network to run properly. This means running each different kind of Corda node only once. The layout of the Corda nodes over the physical nodes is:
  - Omega: Power company and Notary.
  - Alpha: Prosumer.
  - Beta: Producer and Customer.
  - Gamma: Grid authority.
- **auto-exp-1** - Adds new producer and customer nodes to the ones featured in auto-exp-0. The layout of the Corda nodes over the physical nodes in this experiment is:
  - Omega: Power company and Notary.
  - Alpha: Prosumer and Producer1.
  - Beta: Producer and Customer.
  - Gamma: Grid authority and Customer1.
- **auto-exp-2** - Adds new producer and customer nodes on top of the existing auto-exp-1 nodes. The layout of the Corda nodes over the physical nodes is:
  - Omega: Power company, Notary and Producer2.
  - Alpha: Prosumer and Producer1.
  - Beta: Producer, Customer and Customer2.
  - Gamma: Grid authority and Customer1.
- **auto-exp-3** - Adds five prosumer nodes on top of the auto-exp-2 nodes as they are the most likely new nodes to appear in real world use case. The layout of the Corda nodes over the physical nodes is:
  - Omega: Power company, Notary, Producer2 and Prosumer1.
  - Alpha: Prosumer, Prosumer1 and Prosumer2.
  - Beta: Producer, Customer, Customer2 and Prosumer3.
  - Gamma: Grid authority, Customer1, Prosumer4 and Prosumer5.
- **auto-exp-4** - Adds five customer nodes on top of the auto-exp-3 nodes. The layout of the Corda nodes over the physical nodes is:
  - Omega: Power company, Notary, Producer2, Prosumer1 and Customer3.
  - Alpha: Prosumer, Prosumer1, Prosumer2, Customer4 and Customer 5.
  - Beta: Producer, Customer, Customer2, Prosumer3 and Customer6.
  - Gamma: Grid authority, Customer1, Prosumer4, Prosumer5 and Customer7.
- **auto-exp-5** - Adds three producer, three prosumer and four customer nodes on top of the auto-exp-4 nodes. The layout of the Corda nodes over the physical nodes is:
  - Omega: Power company, Notary, Producer2, Prosumer1, Customer3, Producer3, Producer4 and Producer5.
  - Alpha: Prosumer, Prosumer1, Prosumer2, Customer4, Customer 5, Prosumer6, Prosumer7 and Prosumer8.

– Beta: Producer, Customer, Customer2, Prosumer3, Customer6, Customer8 and Customer9.
– Gamma: Grid authority, Customer1, Prosumer4, Prosumer5, Customer7, Customer10 and Customer11.

These experiments compose the first batch. The second batch of the experiments was done on the extended version of the DREAM platform (as described in Section IV-B). We wanted to do exactly the same experiments as in the first batch with added new node types and the introduction of additional functionalities. This was done in effort to see how the system scales, not just by adding more of the existing nodes, but also by extending the application logic and introducing new types of nodes which perform novel responsibilities in a distributed network. All the new experiments in the second batch are named by their counterpart in the first batch with the addition of letter *e*. Therefore, auto-exp-e-0 is the counterpart of auto-exp-0, auto-exp-e-1 is counterpart of auto-exp-1, etc. The layout of the Corda nodes over the physical nodes in extended experiments (second batch) mirrors layout from the basic experiment (first batch) with addition of new node types. Thus, we will not write a detailed node layout for extended experiments as it can be easily deduced by just adding the following nodes to their basic experiment counterparts:

- Alpha: Verification agency.
- Beta: Speculator.
- Gamma: Energy storage provider.

All basic and extended experiments are each conducted ten times. This is done in order to measure the variance. Consequently, running experiments in ideal conditions (without any technical problems) can take weeks. This is not a problem while measuring only the use of computer resources, as it can be done fully remotely. On the other hand, measuring power consumption is more demanding and cumbersome undertaking, so only a subset of all experiments has power consumption statistics attached to them. This subset consists of: auto-exp-0, auto-exp-3, auto-exp-5, auto-exp-e-0, auto-exp-e-3, and auto-exp-e-5.

Although all the listed experiments differ in number and type of nodes that are started in the experimental process, they all follow the same basic flow we named the atomic experiment flow. The atomic experiment flow contains four steps:

- Corda nodes are started on all physical nodes.
- One of the following commands is run: create power promise, create auction, or create bid in batches of 1, 10, and 100 transactions respectively. Corda node performance is measured on all the four physical nodes.
- The performance data is gathered.
- The Corda nodes are stopped on all the physical nodes and their databases are purged. Additionally, all the remaining data produced by previous steps is removed from the physical nodes. This step is done in order to insure that the next experiment will be executed in the same settings as the previous one.

Therefore one experiment, e.g., auto-exp-0 is composed of thirty individual atomic flows defined above. At the beginning of the experiment ten atomic flows which measure the performance of the create power promise command are executed. Thereafter, ten atomic flows which measure the performance of the create auction command and create bid command are executed respectively.

This all resulted in a total of $n = 1080$ different measurements. Half of the measurements represent the basic experiments, and half are of the extended form. Of the total number of experimental cases $n' = 12$ have had their energy consumption measured.The data originates from filtering and analyzing the logs of the experiments conducted on the Beowulf cluster. The logs were filtered to only include the relevant information, and then aggregated on the level of a single experiment instance. This was done by a CLI Python code whose source code is available on request alongside the raw data as well as the processed data, and the R source code used to process it.

1) EXPERIMENT AUTOMATION

The experimental process we use requires numerous steps and can be very tedious if done manually. Consequently, all the experiments are conducted automatically by using Shell and Python scripts. The scripts were written in advance and distributed on all physical nodes. They are used for process orchestration, performance measurements, result collection, data pruning, and any additional steps required for running one experiment. The scripts use the SSH and HTTP protocols for communication and data transfer between the physical nodes and the experiment initiating machine. Fig. 8 gives an overview of the scripts used for the performance testing and their distribution over physical nodes.

The experiment initiating machine is not part of the computer cluster and does not run any Corda nodes. Therefore, its specifications is not relevant for presenting and discussing experimental results. However, it plays an instrumental role in the experimental pipeline and scheduling of the execution steps. The *prepare-build.sh*, *run-nodes.sh*, *run-experiment.sh* and *stop-nodes.sh* scripts are run exclusively on the experiment initiating machine and used to remotely initiate all the experimental steps on the Beowulf cluster by the use of SSH protocol. The *prepare-build.sh* is the entry point for each of twelve conducted experiments. It is used for building all Corda nodes according to the predefined Corda over physical node layout associated with each experiment, copying the binaries required for running Corda nodes on physical nodes, and, finally, installing any other dependencies on the physical nodes.

The *run-nodes.sh* script initiates each atomic experiment flow by running all the Corda nodes included in a given experiment on the remote computer cluster while also running the Corda network proxy on Omega. The *run-experiment.sh*

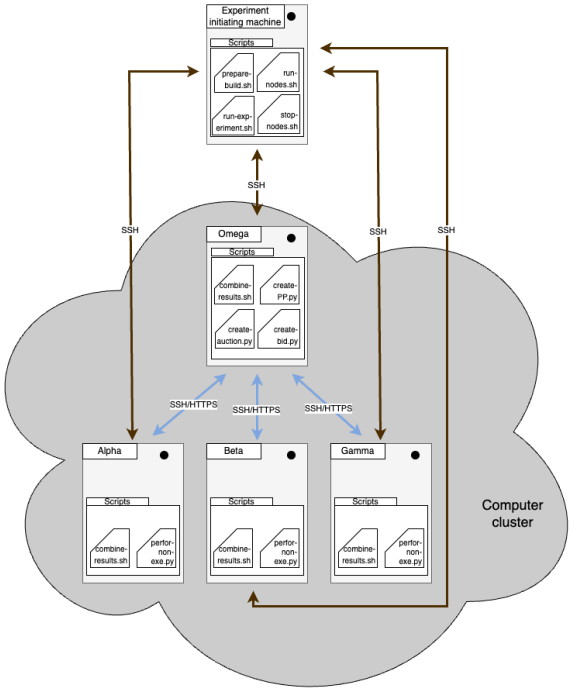

FIGURE 8. Automated performance measurements scripts.

script is started after the *run-nodes.sh* and is responsible for running and orchestrating all the python scripts distributed on the physical nodes. It first runs the *create-PP.py* script on the Omega node which calls the Corda network proxy over HTTPS and begins transaction creation. Additionally, the script runs the *performance-non-exec.py* scripts in parallel on Alpha, Beta, and Gamma nodes which are used for gathering performance data. After finishing the performance measurements for power promise creation, *run-experiment.sh* then runs the *create-auction.py* and the *create-bid.py* scripts respectively, in the same manner as *create-PP.py*.

Successful termination of *run-experiment.sh* means that performance data on all tree commands (create power promise, create auction, and create bid) is successful for the given experiment and that data can be gathered by the use of the *combine-results.sh* script.

Finally, after performance data creation and result gathering, the *stop-nodes.sh* script is started in order to finish the atomic experimental flow by stopping Corda nodes and deleting all data generated by the previous scripts (including Corda node databases). This step provides a clean slate for running the next atomic experimental flow in exactly the same conditions as previous ones.

VI. RESULTS

In this section we present the results of performance and power consumption measurement as described in Section V.The analysis is focused on the time and memory usage of the experiments, and the carbon footprint of the energy used in the experiments. Time and memory analysis are result of the first set of measurements which are focused on network performance from computer science perspective. While carbon footprint analysis is the result of the second set of measurements which measured system's power consumption. Each analysis focuses on one specific outcome variable and tracks how it changes when observed over the number of requests processed, the number of nodes used, and the type of experiment.

The analysis has been conducted using a specialized Python analysis tool in order to parse the experiment logs and create from them a comprehensive database of measurements. Selected elements of this database are aggregated and exported in CSV form for analysis by a script written in the R programming language.

The analysis is split into three sections:

1. Time analysis
2. Memory analysis
3. Carbon footprint analysis

A. TIME ANALYSIS

The first plot (Fig 9) shows the variation of time with the number of nodes versus the command executed and the number of requests. This plot can be interpreted so as to analyze the relative processing time required for any given command, as well as the sensitivity of this relationship to the number of nodes communicating, and the scaling factor showing how the numbers change as the number of requests changes.

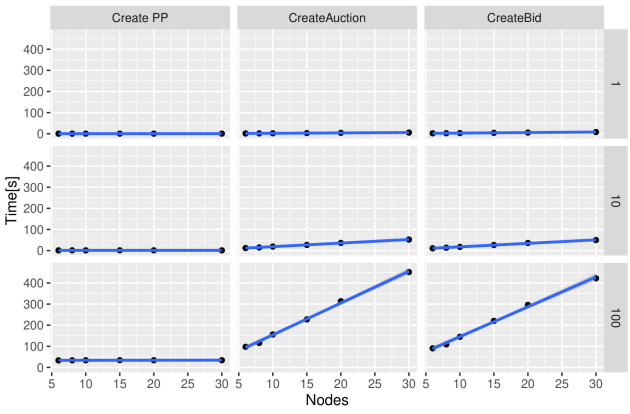

FIGURE 9. Time vs Nodes viewed over Command and Number of Requests (Non-Extended implementation).

As can be seen clearly from the plot, the number of nodes is entirely irrelevant to the Create Power Promise command, and interacts very weakly when there are only 1 or 10 requests. It takes 100 requests for the number of nodes to start having a noticeable effect. The number of nodes has a much more pronounced effect on the time taken for the other commands. In those instances, it is clear that the relationship is a linear one, meaning that the communication, consensus delay, seems to be one that scales linearly, an important insight for future work.

The same can be repeated using extended experiments which feature more complex processing (Fig 10). Exact same pattern emerges, without any apparent deviation.

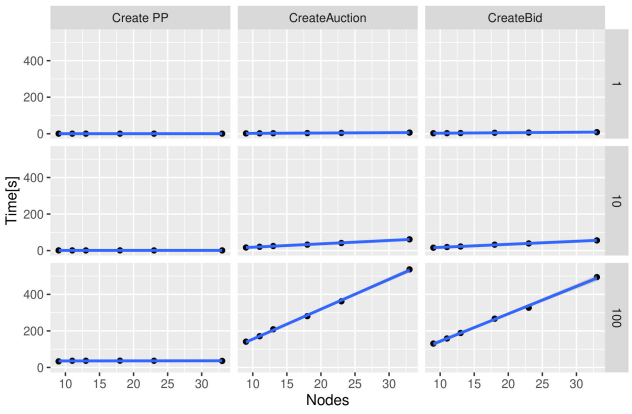

FIGURE 10. Time vs Nodes viewed over Command and Number of Requests (Extended implementation).

Next analysis step is to see if there is additional information to be gleaned when it comes to requests versus time. To this end, data is filtered to select the cases with very few nodes, a medium number of nodes and a very large number of nodes. Then we plot the relationship between time and number of requests (Fig 11). We start with the non-extended cases.

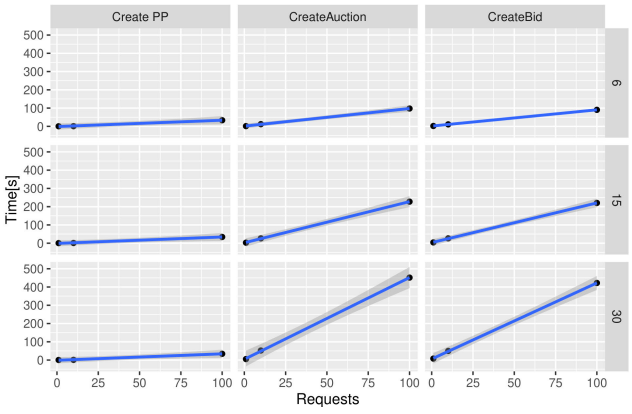

FIGURE 11. Time vs Requests viewed over Nodes and Commands (Non-Extended implementation).

The analysis of the graph clearly shows that there is a predictable linear increase as the number of requests increases. This is true for all commands and all numbers of nodes, though the slopes differ. Note that the sharpness of the increase varies with the number of nodes, and with the command. The Create Power Promise command shows relatively modest increase and is not, as previously observed, affected by the number of nodes. The other two, however, do show an increase, and show, furthermore, that the slope of the increase is steeper as the number of nodes increases. This indicates increasing scaling costs as the number of nodes increases. We can compare this to the plot (Fig 12) of the extended case and note that the exact same pattern can be observed.

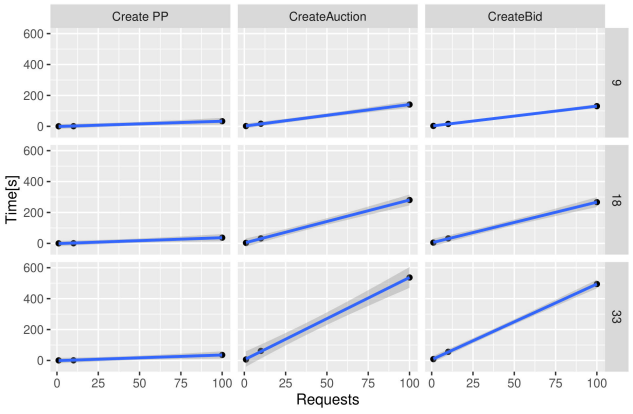

FIGURE 12. Time vs Requests viewed over Nodes and Commands (Extended implementation).

B. MEMORY ANALYSIS

The amount of memory used in the experiment is measured in bytes, but will be converted to megabytes for display and ease of interpretation. Note that this means megabytes, as in 10^6 bytes, not Mebibytes, i.e. 2^{20} bytes. For purposes of this analysis the memory usage measured is based on what Linux reported as the RSS (Resident Set Size) of the process. Naturally, this means that the result isn't *precise* but it should be sufficiently correlated to actual memory usage for purposes of comparison.

The first plot (Fig 13) to consider is the variation of memory usage with the the number of nodes versus the command executed and the number of requests. Much like in the case of time analysis this permits simultaneous analysis of the relative memory consumption of various commands, as well as how this is influenced by the number of requests processed and the number of nodes. Unlike as in the time analysis, we do not necessarily expect a linear relationship to become apparent as it is not unreasonable to expect that the memory consumption of processing one request is not much different than processing e.g. ten, assuming reasonably optimized execution.

The results of memory analysis are absolutely clear, the usage of memory is entirely dominated and determined by the number of nodes executed. Hardly anything else matters. This indicates that the memory footprint, once the infrastructure is running is not high. No effect seems to be visible from the number of requests, and the command executed seems to have a very limited effect. The same can be repeated using extended experiments which feature far more complex processing (Fig 14) while exhibiting same memory use patterns.

As with the analysis of time taken, analysis of the number of requests is important. We select the number of nodes in such a way to create three groups representing the low, medium, and high number, and plot the memory use by

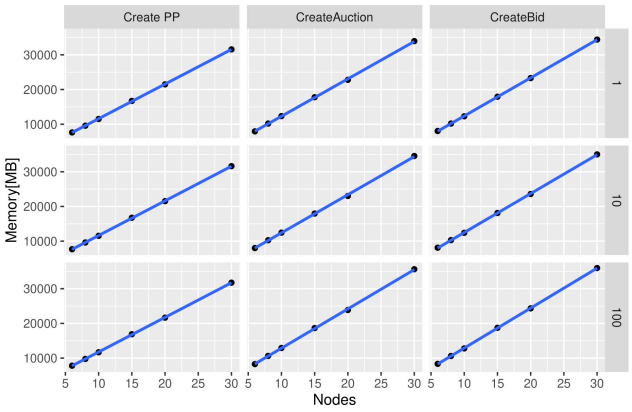

FIGURE 13. Memory usage vs nodes and requests (Non-Extended implementation).

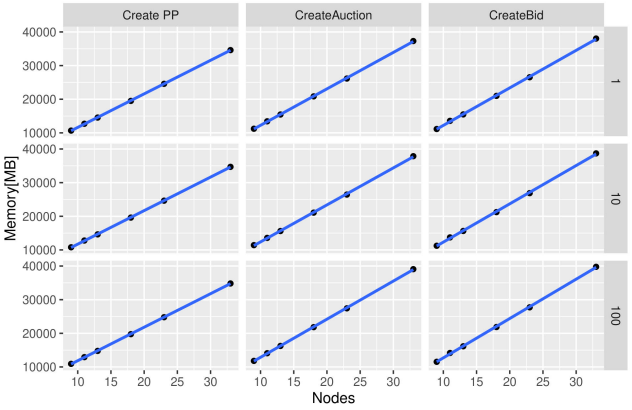

FIGURE 14. Memory usage vs nodes and requests(Extended implementation).

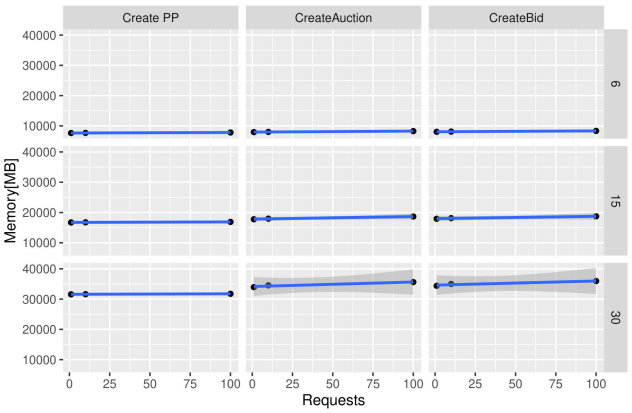

FIGURE 15. Memory usage vs nodes and requests(Non-Extended implementation).

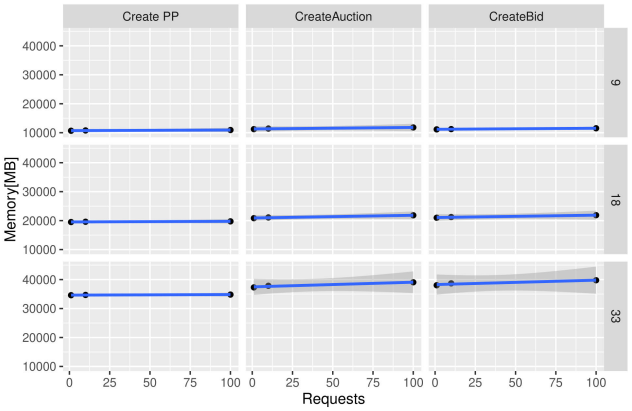

FIGURE 16. Memory usage vs nodes and requests(Extended implementation).

number of requests (Fig 15). As can be clearly seen, memory use does not interact to any significant degree with the number of requests processed. There is no adverse scaling related to processing large volumes of data visible at these scales: the number of nodes is the dominant factor. The exact same thing can be seen when looking at the extended experiment set (Fig 16).

C. CARBON FOOTPRINT ANALYSIS

Before we begin carbon footprint analysis we need to address main assumptions made when computation of the carbon footprint of the energy used in the experiments is conducted. The carbon footprint for a given amount of electrical energy can vary dramatically. We have based our estimate on Ember's 2023 [68] and 2022 [69] yearly electricity reviews and Energy Institute's Statistical Review of World Energy from 2023 [70]. These datasets provide the carbon intensity of electricity generation in grams of CO_2 per kilowatt-hour for a large number of countries. We have used the average carbon intensity for the countries in which the computing cluster is located. This is a simplification, as the carbon intensity of electricity generation can vary within a country, and the carbon intensity of electricity generation is not the only factor that determines the carbon footprint of electricity use. However, we believe that this is a reasonable simplification for the purposes of this analysis.

The value we have chosen to use is the average based on all countries in the cited dataset for the year 2022. This is the most recent year for which data is available. The average carbon intensity of electricity generation for the countries in which the computing cluster is located is 365.3805708 in $\frac{gCO_2}{kWh}$ which is 0.1014946 $\frac{gCO_2}{kJ}$, and is the value used for the computations in this analysis.

Carbon footprint analysis is done on a subset of the data, since not all experiments have had their energy consumption measured. Given a smaller data set, it is not practical to plot the data in the manner we have used thus far, nor do descriptive statistics hold much value. Therefore, the data gathered by energy consumption measurement are presented in Table 3.

Dominant factor in carbon footprint is the number of nodes, as it can be observer fairly comprehensively from Table 3. This same relationship can be seen visually either when the carbon footprint is plotted against nodes for all commands,

TABLE 3. Measured carbon footprint of the energy used in the experiments.

ID	Experiment number	Try	Command	Requests	Nodes	Time	Extended	Memory	Energy	Carbon
32	0	1	CreateAuction	100	6	104.26685	False	8207.278	43549.4	4.420029
62	0	1	CreateBid	100	6	86.33326	False	8357.782	43296.1	4.394321
302	3	1	CreateAuction	100	15	234.75075	False	18631.086	90022.9	9.136838
332	3	1	CreateBid	100	15	219.92399	False	18647.433	104629.7	10.619350
482	5	1	CreateAuction	100	30	464.64646	False	35227.861	155311.0	15.763228
512	5	1	CreateBid	100	30	421.17342	False	35740.602	156665.7	15.900723
572	1000	1	CreateAuction	100	9	133.19794	True	11791.016	58781.6	5.966015
602	1000	1	CreateBid	100	9	133.64989	True	11662.049	86462.5	8.775477
842	1003	1	CreateAuction	100	18	289.44234	True	22204.789	106169.9	10.775672
872	1003	1	CreateBid	100	18	263.96090	True	21870.674	120994.4	12.280279
1022	1005	1	CreateAuction	100	33	521.85225	True	38960.144	155081.0	15.739885
1052	1005	1	CreateBid	100	33	502.22050	True	39343.661	185146.0	18.791320

as on Fig 17. Viewed all on one plot the same data is shown on Fig. 18. What we can see is that even when running together all commands and the extended versus the non-extended experiments, the carbon footprint is entirely dominated by the number of nodes. The command executed and the type of experiment have a predictable effect, but not a dramatic one.

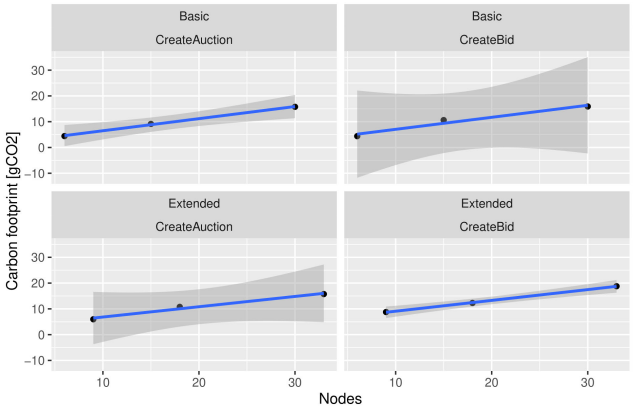

FIGURE 17. Carbon footprint vs Node.

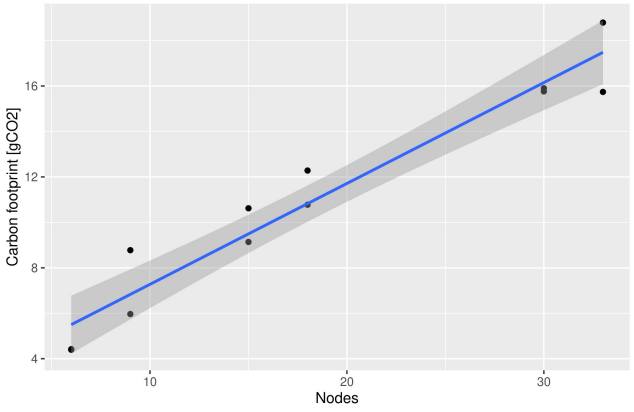

FIGURE 18. Carbon footprint vs Node in single plot.

A way to put the carbon footprint results into perspective is to estimate the carbon footprint per transaction and compare it to a typical technology used in similar solutions. We have selected the Ethereum blockchain as the point of comparison and we have focused on the PoS variant of the chain since that is the far more energy efficient and modern iteration. We will compare the most carbon-intensive experimental case on our end (0.1869132 grams of CO_2 with the carbon footprint of an Ethereum transaction as reported in the literature, specifically by Tian [71]. We have selected an NFT minting transaction as the one to compare against as it represents a reasonable but not excessive amount of work with the error being in Ethereum's favor to make sure that the comparison is maximally fair to Ethereum.

The amount reported for NFT minting is 0.02 kg of CO_2 per transaction which means that in this comparison our solution is approximately 106 times more efficient than Ethereum when comparing the most carbon intensive case with an NFT minting transaction. Presented results show that a significant amount of saving can be achieved by using DLT systems like Corda and should encourage future researchers and software developers to test alternative technologies when designing their applications.

D. RESULTS SUMMARY

The results of this analysis can be summarized by the following principal findings:

1. The execution time scales linearly with the number of requests processed. The slope is dependent on the command being executed and the number of nodes used.
2. The number of nodes in most cases does not increase execution time. However, an increase is visible in case of a large number of nodes executing a command other than the creation of a power promise.
3. The memory usage is entirely determined by the number of nodes used with an increase of the number of nodes leading directly to an increase of memory used. The number of requests processed have no significant impact.
4. The extended implementation of DREAM introduces valuable new features without any noticeable performance penalties. This indicates that application

scales well when it comes to vertical feature enhancements.
5. The carbon footprint, based on available data, is strongly determined by the number of nodes used, with more nodes leading to a larger impact. The command executed has only a limited impact.
6. The carbon footprint of the most carbon intensive experiment is 0.1879132 grams of CO_2 per transaction. This is 106.43212 times less carbon intensive than the average comparable transaction on the Ethereum used most often in similar solutions.

Presented findings show that computer resource consumption has linear tendencies. More importantly, the paper shows that power consumption of the whole distributed application is much smaller compared to similar applications which use other popular technologies like Ethereum. Findings have potential to impact the industry and more parties should consider using alternative DLT technologies like Corda instead of defaulting on popular Ethereum-based solutions. This is especially true in situations where moderate resource consumption is a higher priority than excess security combined with abundant data redundancy.

VII. CONCLUSION

In this paper we presented a performance evaluation of a DLT-based platform for renewable energy trading. It is based on the Corda DLT and allows users to trade with information-rich electrical power promises. Corda allows our solution to be scalable while using very little additional energy. The system is designed to allow users to have privacy levels which are not easily achievable using public blockchains, such as Ethereum, which are currently a typical choice for DLTs used in similar energy trading systems.

The performance analysis we performed which measured time taken, memory consumption, and power consumption (which was subsequently used to estimate carbon footprint) shows mostly expected results with some surprises. The expected results include results such as that the execution time of all commands scales linearly with the number of requests processed, while number of request does not have a significant impact on use of memory. Further, memory use is entirely dominated by the number of nodes which, in most of the cases, does not negatively impact execution time. The carbon footprint is strongly determined by the number of nodes where commands executed have only limited impact. The most important result which we found and which was unexpected is that the carbon footprint of a typical DREAM transaction is around 100 times smaller than that of an comparable Ethereum transaction.

Our plans for future work are focused on connecting DREAM to a real power grid and replacing simulated data with real data gathered directly from smart power meters. Although we have tested the system in a distributed setting and gathered data which emulates real word use, we still need to connect system to a micro grid and gather additional performance results.

Further application improvements will be focused on the user interface and usability of the application. The automation of trading based on user preferences would also be beneficial and is planed as further work. It would increase the usability of the system by reducing human input required for receiving full system benefits. These features, along with actual test results of DREAM working within a micro grid environment, will allow future production use of the proposed platform.

REFERENCES

[1] J. Liu, H. Wang, B. Zhang, and Y. Xiong, "An analysis of some changes in public environmental awareness in Beijing," in *Proc. Int. Conf. Remote Sens., Environ. Transp. Eng.*, Jun. 2011, pp. 8045–8049.
[2] A. Vasiljevic-Shikaleska, G. Trpovski, and B. Gjozinska, "Environmental awareness and of pro-environmental consumer behavior," *J. Sustain. Develop.*, vol. 8, no. 20, pp. 4–17, 2018.
[3] T. Kurbatova and T. Perederii, "Global trends in renewable energy development," in *Proc. IEEE KhPI Week Adv. Technol. (KhPIWeek)*, Oct. 2020, pp. 260–263.
[4] D. B. Gajić, V. B. Petrović, N. Horvat, D. Dragan, A. Stanisavljević, V. Katić, and J. Popović, "A distributed ledger-based automated marketplace for the decentralized trading of renewable energy in smart grids," *Energies*, vol. 15, no. 6, p. 2121, Mar. 2022.
[5] M. Andoni, V. Robu, D. Flynn, S. Abram, D. Geach, D. Jenkins, P. McCallum, and A. Peacock, "Blockchain technology in the energy sector: A systematic review of challenges and opportunities," *Renew. Sustain. Energy Rev.*, vol. 100, pp. 143–174, Feb. 2019.
[6] N. Khezami, N. Gharbi, B. Neji, and N. B. Braiek, "Blockchain technology implementation in the energy sector: Comprehensive literature review and mapping," *Sustainability*, vol. 14, no. 23, p. 15826, Nov. 2022.
[7] T. Al-Abri, A. Onen, R. Al-Abri, A. Hossen, A. Al-Hinai, J. Jung, and T. S. Ustun, "Review on energy application using blockchain technology with an introductions in the pricing infrastructure," *IEEE Access*, vol. 10, pp. 80119–80137, 2022.
[8] R3. (2024). *Corda Platform*. Accessed: Apr. 2, 2024. [Online]. Available: https://docs.r3.com/
[9] P. Woitschig, G. S. Uddin, T. Xie, and W. K. Härdle, "The energy consumption of the Ethereum-ecosystem," *SSRN Electron. J.*, Jul. 2023.
[10] M. J. M. Chowdhury, MD. S. Ferdous, K. Biswas, N. Chowdhury, A. S. M. Kayes, M. Alazab, and P. Watters, "A comparative analysis of distributed ledger technology platforms," *IEEE Access*, vol. 7, pp. 167930–167943, 2019.
[11] M. Gorbunova, P. Masek, M. Komarov, and A. Ometov, "Distributed ledger technology: State-of-the-art and current challenges," *Comput. Sci. Inf. Syst.*, vol. 19, no. 1, pp. 65–85, 2022.
[12] S. Ray. *The Difference Between Blockchains & Distributed Ledger Technology | by Shaan Ray | Towards Data Science*. Accessed: Apr. 2, 2024. [Online]. Available: https://towardsdatascience.com/the-difference-between-blockchains-distributed-ledger-technology-42715a0fa92
[13] M. Di Pierro, "What is the blockchain?" *Comput. Sci. Eng.*, vol. 19, no. 5, pp. 92–95, 2017.
[14] F. Wang, Y. Chen, R. Wang, A. O. Francis, B. Emmanuel, W. Zheng, and J. Chen, "An experimental investigation into the hash functions used in blockchains," *IEEE Trans. Eng. Manag.*, vol. 67, no. 4, pp. 1404–1424, Nov. 2020.
[15] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," *Decentralized Bus. Rev.*, Aug. 2008, Art. no. 21260.
[16] S. Wang, L. Ouyang, Y. Yuan, X. Ni, X. Han, and F.-Y. Wang, "Blockchain-enabled smart contracts: Architecture, applications, and future trends," *IEEE Trans. Syst. Man, Cybern. Syst.*, vol. 49, no. 11, pp. 2266–2277, Nov. 2019.
[17] M. Bartoletti and L. Pompianu, "An empirical analysis of smart contracts: Platforms, applications, and design patterns," in *Financial Cryptography and Data Security*. Cham, Switzerland: Springer, 2017, pp. 494–509.
[18] D. P. Oyinloye, J. S. Teh, N. Jamil, and M. Alawida, "Blockchain consensus: An overview of alternative protocols," *Symmetry*, vol. 13, no. 8, p. 1363, Jul. 2021.

[19] V. Buterin. (2024). *A Next-generation Smart Contract and Decentralized Application Platform*. Accessed: Feb. 2, 2024. [Online]. Available: https://ethereum.org/en/whitepaper/
[20] Hyperledger. (2024). *Open Source Blockchain Technologies*. Accessed: Apr. 2, 2024. [Online]. Available: https://hyperledger-fabric.readthedocs.io/en/release-2.5/
[21] A. A. Monrat, O. Schelén, and K. Andersson, "Performance evaluation of permissioned blockchain platforms," in *Proc. IEEE Asia–Pacific Conf. Comput. Sci. Data Eng. (CSDE)*, Dec. 2020, pp. 1–8.
[22] R. Corda. (2024). *Flows-enterprise 4.8*. Accessed: Apr. 2, 2024. [Online]. Available: https://docs.r3.com/en/platform/corda/4.8/enterprise/key-concepts-flows
[23] R. Corda. (2024). *Documentation (2024)*. Accessed: Feb. 25, 2024. [Online]. Available: https://docs.r3.com
[24] M. Schäffer, M. Di Angelo, and G. Salzer, "Performance and scalability of private Ethereum blockchains," in *Proc. Int. Conf. Bus. Process Manag.*, Vienna, Austria. Cham, Switzerland: Springer, Sep. 2019, pp. 103–118.
[25] E. Org. (2024). *Ethereum Energy Consumption*. Accessed: Apr. 2, 2024. [Online]. Available: https://ethereum.org/en/energy-consumption/
[26] E. Mengelkamp, J. Gärttner, K. Rock, S. Kessler, L. Orsini, and C. Weinhardt, "Designing microgrid energy markets: A case study: The Brooklyn Microgrid," *Appl. Energy*, vol. 210, pp. 870–880, Jan. 2018.
[27] L. Lamport, R. Shostak, and M. Pease, "The Byzantine generals problem," in *Concurrency: Works of Leslie Lamport*. New York, NY, USA: Association for Computing Machinery, Oct. 2019, pp. 203–226.
[28] (2024). *Powerledger*. Accessed: Apr. 2, 2024. [Online]. Available: https://www.powerledger.io/
[29] R. K. Kodali, S. Yerroju, and B. Y. K. Yogi, "Blockchain based energy trading," in *Proc. TENCON IEEE Region 10 Conf.*, Oct. 2018, pp. 1778–1783.
[30] S. Wang, A. F. Taha, J. Wang, K. Kvaternik, and A. Hahn, "Energy crowdsourcing and peer-to-peer energy trading in blockchain-enabled smart grids," *IEEE Trans. Syst. Man, Cybern. Syst.*, vol. 49, no. 8, pp. 1612–1623, Aug. 2019.
[31] Z. Li, J. Kang, R. Yu, D. Ye, Q. Deng, and Y. Zhang, "Consortium blockchain for secure energy trading in industrial Internet of Things," *IEEE Trans. Ind. Informat.*, vol. 14, no. 8, pp. 3690–3700, Aug. 2018.
[32] S. J. Pee, E. S. Kang, J. G. Song, and J. W. Jang, "Blockchain based smart energy trading platform using smart contract," in *Proc. Int. Conf. Artif. Intell. Inf. Commun. (ICAIIC)*, Feb. 2019, pp. 322–325.
[33] N. Wang, X. Zhou, X. Lu, Z. Guan, L. Wu, X. Du, and M. Guizani, "When energy trading meets blockchain in electrical power system: The state of the art," *Appl. Sci.*, vol. 9, no. 8, p. 1561, Apr. 2019.
[34] S. Seven, G. Yao, A. Soran, A. Onen, and S. M. Muyeen, "Peer-to-peer energy trading in virtual power plant based on blockchain smart contracts," *IEEE Access*, vol. 8, pp. 175713–175726, 2020.
[35] G. Electric. *The Digital Power Plant*. Accessed: Nov. 12, 2023. [Online]. Available: https://www.ge.com/digital/sites/default/files/downloadassets/GE-Digital-Power-Plant-Brochure.pdf
[36] M. R. Hamouda, M. E. Nassar, and M. M. A. Salama, "A novel energy trading framework using adapted blockchain technology," *IEEE Trans. Smart Grid*, vol. 12, no. 3, pp. 2165–2175, May 2021.
[37] E. K. Markakis, Y. Nikoloudakis, K. Lapidaki, K. Fiorentzis, and E. Karapidakis, "Unification of edge energy grids for empowering small energy producers," *Sustainability*, vol. 13, no. 15, p. 8487, Jul. 2021.
[38] C. Shah, J. King, and R. W. Wies, "Distributed ADMM using private blockchain for power flow optimization in distribution network with coupled and mixed-integer constraints," *IEEE Access*, vol. 9, pp. 46560–46572, 2021.
[39] A. A. G. Agung and R. Handayani, "Blockchain for smart grid," *J. King Saud Univ. Comput. Inf. Sci.*, vol. 34, no. 3, pp. 666–675, Mar. 2022.
[40] I. Kappos, I. Kouveliotis-Lysikatos, and N. Hatziargyriou, "A blockchain platform for the decentralized operation of active distribution networks," in *Proc. IEEE Madrid PowerTech*, Jun. 2021, pp. 1–6.
[41] CoinMarketCap. (2024). *Blockchain Trilemma Definition*. Accessed: Apr. 2, 2024. [Online]. Available: https://coinmarketcap.com/academy/glossary/blockchain-trilemma
[42] E. Org. (2024). *The Merge*. Accessed: Apr. 2, 2024. [Online]. Available: https://ethereum.org/en/roadmap/merge/
[43] (2024). *What Is Sharding*. Accessed: Apr. 2, 2024. [Online]. Available: https://crypto.com/university/what-is-sharding
[44] Chainlink. (2024). *What is Layer 2*. Accessed: Apr. 2, 2024. [Online]. Available: https://chain.link/education-hub/what-is-layer-2
[45] M. Dabbagh, M. Kakavand, M. Tahir, and A. Amphawan, "Performance analysis of blockchain platforms: Empirical evaluation of hyperledger fabric and Ethereum," in *Proc. IEEE 2nd Int. Conf. Artif. Intell. Eng. Technol. (IICAIET)*, Sep. 2020, pp. 1–6.
[46] Blockchair. (2024). *Universal Blockchain Explorer and Search Engine*. Accessed: Apr. 2, 2024. [Online]. Available: https://blockchair.com/
[47] H. Foundation. (2023). *Benchmarking Hyperledger Fabric 2.5 Performance*. Accessed: Apr. 2, 2024. [Online]. Available: https://www.hyperledger.org/blog/2023/02/16/benchmarking-hyperledger-fabric-2-5-performance
[48] R. Corda. (2024). *Performance Benchmarking Results of Corda 4.8*. Accessed: Apr. 2, 2024. [Online]. Available: https://docs.r3.com/en/platform/corda/4.8/enterprise/performance-testing/performance-results.html
[49] R. Corda. (2024). *Transactions Per Second (TPS)*. Accessed: Apr. 2, 2024. [Online]. Available: https://corda.net/blog/transactions-per-second-tps/
[50] J. Polge, J. Robert, and Y. Le Traon, "Permissioned blockchain frameworks in the industry: A comparison," *ICT Exp.*, vol. 7, no. 2, pp. 229–233, Jun. 2021.
[51] R. Han, G. Shapiro, V. Gramoli, and X. Xu, "On the performance of distributed ledgers for Internet of Things," *Internet Things*, vol. 10, Jun. 2020, Art. no. 100087.
[52] S. Pongnumkul, C. Siripanpornchana, and S. Thajchayapong, "Performance analysis of private blockchain platforms in varying workloads," in *Proc. 26th Int. Conf. Comput. Commun. Netw. (ICCCN)*, Jul. 2017, pp. 1–6.
[53] M. Touloupou, M. Themistocleous, E. Iosif, and K. Christodoulou, "A systematic literature review toward a blockchain benchmarking framework," *IEEE Access*, vol. 10, pp. 70630–70644, 2022.
[54] T. T. A. Dinh, J. Wang, G. Chen, R. Liu, B. C. Ooi, and K.-L. Tan, "BLOCKBENCH: A framework for analyzing private blockchains," in *Proc. ACM Int. Conf. Manage. Data*, May 2017, p. 1085.
[55] D. Saingre, T. Ledoux, and J.-M. Menaud, "BCTMark: A framework for benchmarking blockchain technologies," in *Proc. IEEE/ACS 17th Int. Conf. Comput. Syst. Appl. (AICCSA)*, Nov. 2020, pp. 1–8.
[56] H. Organization. (2024). *Caliper is a Blockchain Performance Benchmark Framework, Which Allows Users to Test Different Blockchain Solutions With Predefined Use Cases, and Get a Set of Performance Test Results*. Accessed: Apr. 2, 2024. [Online]. Available: https://hyperledger.github.io/caliper/
[57] (2024). *Metrics Data and Monitoring Enterprise 4.8*. Accessed: Apr. 2, 2024. [Online]. Available: https://docs.r3.com/en/platform/corda/4.8/enterprise/operations/monitoring-logging/metrics-monitoring-scenarios.html
[58] Apache. (2024). *Apache JMeter*. Accessed: Apr. 2, 2024. [Online]. Available: https://jmeter.apache.org/
[59] R. Corda. (2024). *Introduction Enterprise 4.8*. Accessed: Apr. 2, 2024. [Online]. Available: https://docs.r3.com/en/platform/corda/4.8/enterprise/performance-testing/introduction.html
[60] J. Sedlmeir, H. U. Buhl, G. Fridgen, and R. Keller, "The energy consumption of blockchain technology: Beyond myth," *Bus. Inf. Syst. Eng.*, vol. 62, no. 6, pp. 599–608, Dec. 2020.
[61] J. Sedlmeir, H. U. Buhl, G. Fridgen, and R. Keller, "Ein blick auf aktuelle entwicklungen bei blockchains und deren auswirkungen auf den energieverbrauch," *Informatik Spektrum*, vol. 43, no. 6, pp. 391–404, Nov. 2020.
[62] A. O. Bada, A. Damianou, C. M. Angelopoulos, and V. Katos, "Towards a green blockchain: A review of consensus mechanisms and their energy consumption," in *Proc. 17th Int. Conf. Distrib. Comput. Sensor Syst. (DCOSS)*, Jul. 2021, pp. 503–511.
[63] Digiconomist. (2024). *Bitcoin Energy Consumption Index*. Accessed: Apr. 2, 2024. [Online]. Available: https://digiconomist.net/bitcoin-energy-consumption
[64] R. Corda. (2024). *Just How Energy Efficient is Your Blockchain*. Accessed: Apr. 2, 2024. [Online]. Available: https://r3.com/blog/just-how-energy-efficient-is-your-blockchain/
[65] M. Masse, *REST API Design Rulebook: Designing Consistent RESTful Web Service Interface*. Sebastopol, CA, USA: O'Reilly Media, 2011.
[66] C. Fan, S. Ghaemi, H. Khazaei, and P. Musilek, "Performance evaluation of blockchain systems: A systematic survey," *IEEE Access*, vol. 8, pp. 126927–126950, 2020.
[67] *Chauvin Arnoux C.A 8336 Measurement Instrument*. Accessed: Oct. 6, 2023. [Online]. Available: https://www.chauvin-arnoux.com/sites/default/files/HLHBXUP5.PDF

[68] Ember. (2023). *Yearly Electricity Data (2023)*. Accessed: Feb. 7, 2024. [Online]. Available: https://ember-climate.org/data-catalogue/yearly-electricity-data
[69] Ember. (2022). *Yearly Electricity Data (2022)*. Accessed: Feb. 7, 2024. [Online]. Available: https://ember-climate.org/insights/research/european-electricity-review-2022/
[70] E. Institute. (2023). *Statistical Review of World Energy (2023)*. Accessed: Feb. 7, 2024. [Online]. Available: https://www.energyinst.org/statistical-review
[71] Z. Tian, "Post-merge carbon footprint analysis and sustainability in the NFT art market," *Arts*, vol. 12, no. 5, p. 211, Sep. 2023.

NEBOJŠA HORVAT (Member, IEEE) received the B.S. and M.S. degrees in electrical and computer engineering from the Faculty of Technical Sciences, University of Novi Sad, Republic of Serbia, in 2018 and 2019, respectively. He is currently pursuing the Ph.D. degree in electrical and computer engineering with the University of Novi Sad.

He is also a Teaching Assistant with the Faculty of Technical Sciences, where he teaches courses in distributed systems and blockchain technology, computer architecture, and parallel computing. His research interests include distributed systems, parallel programming, and private and public blockchain technologies.

DUŠAN B. GAJIĆ (Member, IEEE) received the M.Sc. and Ph.D. degrees (Hons.) in electrical engineering and computing from the Faculty of Electronic Engineering, University of Nis, Serbia, in 2009 and 2014, respectively.

He is currently an Associate Professor with the Faculty of Technical Sciences, University of Novi Sad. He has authored or coauthored five chapters in international scientific monographs, more than 70 papers published in peer-reviewed journals or presented at scientific conferences, and four technical solutions. His research interests include parallel and distributed computing, blockchain, GPGPU, algorithm design, and machine learning.

PETAR TRIFUNOVIĆ received the bachelor's degree from the Faculty of Electronic Engineering, University of Nis, and the master's degree in electrical and computer engineering from the Faculty of Technical Sciences, University of Novi Sad, in 2022, where he is currently pursuing the Ph.D. degree.

He is also a Teaching Assistant with the Faculty of Technical Sciences, where he teaches courses in computer architecture, parallel computing, and high-performance computing. His research interests include computer languages, distributed environments, and blockchain.

VELJKO B. PETROVIĆ received the M.Sc. and Ph.D. degrees (Hons.) in electrical engineering and computing from the Faculty of Technical Sciences, University of Novi Sad, Serbia, in 2010 and 2018, respectively.

He is currently an Assistant Professor with the Faculty of Technical Sciences, University of Novi Sad. He has authored or coauthored more than 30 papers published in scientific monographs and peer-reviewed journals or presented at scientific conferences. His research interests include machine learning, statistical software applications, computer vision, data visualization, AR/VR software, cryptography, and computer security.

DINU DRAGAN received the M.Sc. and Ph.D. degrees (Hons.) in electrical engineering and computing from the Faculty of Technical Sciences, University of Novi Sad, Serbia, in 2008 and 2013, respectively.

He is currently an Associate Professor with the Faculty of Technical Sciences, University of Novi Sad. He has authored or co-authored more than 60 papers published in scientific monographs, peer-reviewed journals or presented at scientific conferences, and three technical solutions. His research interests include data compression, human–computer interaction, UX/UI, computer graphics and computer vision, VR/AR, algorithms, and machine learning.

VLADIMIR A. KATIĆ (Life Senior Member, IEEE) received the B.Sc. degree in electrical engineering from the University of Novi Sad, Serbia, in 1978, and the M.Sc. and Ph.D. degrees in electrical engineering from the University of Belgrade, Serbia, in 1981 and 1991, respectively. Since 1978, he has been with the Faculty of Technical Sciences, University of Novi Sad, where he was appointed to a Full Professor, in 2002. He has been a Honorary Professor with the Polytechnic University of Timisoara, Romania, since 2019. He has participated or a main Researcher in 65 national and international research and development projects. He is the author or coauthor of 46 textbooks and a scientific monograph. He has published more than 600 scientific papers in international and national journals and conferences proceedings. His research interests include power electronics converters, power quality, renewable energy sources, and electric vehicles.