Article

Efficient State Synchronization in Distributed Electrical Grid Systems Using Conflict-Free Replicated Data Types

Arsentii Prymushko [1,*], Ivan Puchko [1], Mykola Yaroshynskyi [1], Dmytro Sinko [1], Hryhoriy Kravtsov [1] and Volodymyr Artemchuk [1,2,3,4]

[1] Department of Mathematical and Computer Modeling, G.E. Pukhov Institute for Modelling in Energy Engineering of the NAS of Ukraine, 15 General Naumov Str., 03164 Kyiv, Ukraine; ivan.puchko@pimee.ua (I.P.); mykola.yaroshynskyi@pimee.ua (M.Y.); dmytro.sinko@pimee.ua (D.S.); hryhoriy.kravtsov@pimee.ua (H.K.); volodymyr.artemchuk@pimee.ua (V.A.)

[2] Department of Environmental Protection Technologies and Radiation Safety, Center for Information-Analytical and Technical Support of Nuclear Power Facilities Monitoring of the NAS of Ukraine, 34a Palladin Ave., 03142 Kyiv, Ukraine

[3] Department of Information Systems in Economics, Kyiv National Economic University Named after Vadym Hetman, 54/1 Peremohy Ave., 03057 Kyiv, Ukraine

[4] Department of Intellectual Cybernetic Systems, State Non-Profit Enterprise State University "Kyiv Aviation Institute", 1 Liubomyra Huzara Ave., 03058 Kyiv, Ukraine

* Correspondence: arsentii.prymushko@pimee.ua

Abstract: Modern electrical grids are evolving towards distributed architectures, necessitating efficient and reliable state synchronization mechanisms to maintain structural and functional consistency. This paper investigates the application of conflict-free replicated data types (CRDTs) for representing and synchronizing the states of distributed electrical grid systems (DEGSs). We present a general structure for DEGSs based on CRDTs, focusing on the Convergent Replicated Data Type (CvRDT) model with delta state propagation to optimize the communication overhead. The Observed Remove Set (ORSet) and Last-Writer-Wins Register (LWW-Register) are utilized to handle concurrent updates and ensure that only the most recent state changes are retained. An actor-based framework, "Vigilant Hawk", leveraging the Akka toolkit, was developed to simulate the asynchronous and concurrent nature of DEGSs. Each electrical grid node is modelled as an independent actor with isolated state management, facilitating scalability and fault tolerance. Through a series of experiments involving 100 nodes under varying latency degradation coefficients (LDK), we examined the impact of network conditions on the state synchronization efficiency. The simulation results demonstrate that CRDTs effectively maintain consistency and deterministic behavior in DEGSs, even with increased network latency and node disturbances. An effective LDK range was identified (LDK effective = 2 or 4), where the network remains stable without significant delays in state propagation. The linear relationship between the full state distribution time (FSDT) and LDK indicates that the system can scale horizontally without introducing complex communication overhead. The findings affirm that using CRDTs for state synchronization enhances the resilience and operational efficiency of distributed electrical grids. The deterministic and conflict-free properties of CRDTs eliminate the need for complex concurrency control mechanisms, making them suitable for real-time monitoring and control applications. Future work will focus on addressing identified limitations, such as optimizing message routing based on the network topology and incorporating security measures to protect state information in critical infrastructure systems.

Keywords: conflict-free replicated data type; state; synchronization; distributed electrical grid system; consensus; fault tolerance

Academic Editor: Amiya Nayak

Received: 28 November 2024
Revised: 31 December 2024
Accepted: 9 January 2025
Published: 11 January 2025

Citation: Prymushko, A.; Puchko, I.; Yaroshynskyi, M.; Sinko, D.; Kravtsov, H.; Artemchuk, V. Efficient State Synchronization in Distributed Electrical Grid Systems Using Conflict-Free Replicated Data Types. *IoT* **2025**, *6*, 6. https://doi.org/10.3390/iot6010006

1. Introduction

Modern electrical grid development is driven by research in various industry fields to satisfy modern society's needs. The integration of renewable energy sources like photovoltaic cells [1], differences in the supply chain [2], distributed management [3,4], the active digitalization of monitoring [5], analysis [6], and management processes [7–9], as well as decentralization and scaling [10], leads to a wide range of problems, starting from performance to complex requirements for network objects in terms of fault tolerance and resilience. A key challenge is the active synchronization of states [11,12] within a distributed network to ensure its structural and functional consistency. Such synchronization is crucial for critical infrastructure and is one of the necessary conditions for trouble-free functioning and stability [13,14]. CRDTs can ensure consistency in a deterministic and conflict-free manner. They are mainly used to provide fault-tolerant and robust systems. Therefore, we assume that CRDTs can potentially be used to maintain the overall resilience of the distributed electrical grid.

The main purpose of this study is to explore and evaluate the application of CRDTs in distributed electrical grid systems, particularly in high-load, real-time environments.

The rest of the paper is structured as follows: In Section 2, related work is discussed; in Section 3, a general representation of the distributed electrical grid's state with formal definitions and consensus model representation for state coordination is described; in Section 4, the experiment based on the actor-based framework to model the distributed message delivery process using the CRDT-based state is outlined. In Sections 5 and 6, the modelling results with further discussion and conclusions are presented.

2. Literature Overview

In the last decade, digital systems have radically transformed and adapted to the needs of modern society to meet the latest demands for speed, efficiency, security, fault tolerance, and resilience. The massive shift from centralized computing architectures urges us to find a new way to address problems with applications that involve new approaches, such as the Internet of Things (IoT), where we can collect and preprocess data at the very end of the network.

Galesky et al. [15] presented the VCuve-Sync protocol, where the state of the distributed application is maintained by a CRDT, which allows operations to be performed simultaneously while ensuring a deterministic convergent final state. The results show that the solution demonstrates excellent performance in terms of message delivery latency and good efficiency in terms of bandwidth usage, especially in scenarios with multiple publishers and subscribers. A similar approach [16] was proposed by Leonardo de Freitas Galesky and Luiz Antonio Rodrigues. Simić et al. [17] proposed a concept of the distributed system where the machine state is represented by CRDT data, allowing for operations like a real-time consensus with proper synchronization. Zhang et al. [18] showed how different protocols for state synchronization can be used in distributed systems. The characteristics of consistency were also analysed, and a blockchain system called CChain was proposed, which for the first time integrated the eventual consistency method (CRDT) into the Fabric.

In a distributed system, strong consistency ensures that all clients observe consistent atomic updates of data across all servers [19]. However, due to the influence of the CAP theorem [20,21], such systems must sacrifice availability during synchronization. Xin Zhao and Philipp Haller [22] examined the model of observed atomic consistency (OAC), which ensures both fast convergent updates and non-convergent operations that require synchronization. The presented observable atomic consistency protocol (OACP) ensures OAC and can only be implemented if the communication subsystem ensures eventual delivery. This assumption is shared by CRDTs.

Barreto et al. [23] provided an overview of the use of CRDTs for synchronization in rapidly changing distributed systems. The authors proposed a PS-CRDT (publisher/subscriber) model to provide spatial and temporal decoupling of update dissemination. Such a solution ensures CRDT compatibility with the dynamic entry and exit of nodes in unstable environments. Lv et al. [24] proposed a new CRDT-based synchronization approach for real-time Co-CAD systems.

As we can see, there is a huge interest in using different CRDTs for the representation of the state of a distributed system for resolving disturbances and perturbations that can be caused by conflicts between system states. Numerous studies [25–27] show the efficiency, high performance, low latency, and effective scalability of CRDT-based approaches in the context of message distribution, especially for large-scale systems using the Pub/Sub model. These results indicate that the solution is a promising option for ensuring strong eventual consistency [28,29] in distributed data systems. It can be very beneficial for systems that must be fault-tolerant and highly resilient.

3. General Representation of the Distributed Electrical Grid's States Using CRDTs

The overall state of a distributed electrical grid can be represented as a set of states of all its nodes, reflecting the current state of each one and their interactions [30]. Among the aspects of the state, the following can be highlighted [31,32]:

- Node availability: This state can indicate whether nodes in the network are available for data exchange. If a node fails or disconnects, this will affect the network's state [33].
- Resource information: This state may include information about resources provided by the nodes.
- Network load: This state can indicate the overall load on the network. If many nodes are actively exchanging data, this may lead to network congestion.
- Connection stability: This state can reflect the stability of the connections between nodes. If connections are unstable or nodes frequently lose connectivity, this can impact the network performance.

The transmission of electricity to consumers can involve two types of events that should be distinguished [34]:

- Continuous phenomena, meaning deviations from nominal values that occur continuously. These phenomena mainly result from load characteristics, load power variations, or the nonlinear nature of the load.
- Random voltage events, meaning sudden and significant deviations from the normal or desired voltage waveform. These are most likely due to unforeseen events (such as accidents) or external factors (such as weather conditions or actions by third parties).

Continuous phenomena include the frequency and variation of the supply current. For example, the nominal supply voltage frequency in Ukraine should be 50 Hz [34]. Random voltage events include power supply interruptions and voltage sags. The cause of voltage sags is usually accidents occurring in public networks or consumer equipment. Overvoltages are generally caused by switching actions and load disconnections. Both phenomena are unpredictable and random, with their frequency varying significantly over the year depending on the type of power supply system and the observation point. Moreover, their distribution throughout the year can be highly uneven [34].

It should be noted that the characteristics of the supply voltage—the frequency, level, waveform, and symmetry of phase voltages—undergo changes during the normal operation of the system due to load power fluctuations or disturbances generated by certain types of equipment, as well as during faults, which are mainly caused by external events.

These characteristics change randomly over time at any chosen connection point, and furthermore, they can vary simultaneously at different connection points. As a result of these changes, in a reasonable number of cases, it can be expected that the characteristics will exceed the set values. Some phenomena affecting voltage, especially unpredictable ones, make it very difficult to establish appropriate values for their characteristics [34].

In creating the distributed electrical grid model, the following principles were followed: the model should be used to describe the path of an object's state change depending on its current state and additional information about the states of other objects, which is received externally.

The general state management model of an electrical system consists of a set of peer nodes, each representing an energy provider model with certain static and dynamic characteristics. The static characteristics include the node identifier and its geographical region. The dynamic characteristics include parameters such as the nominal and actual power [35,36]. The combination of these characteristics forms the individual state of each node in the model.

To represent the state of a node, it is necessary to identify characteristics that are unique, precise, and relatively infrequently changing while also impacting management system decisions. Such characteristics include the indicators of nominal and actual power.

Since the model describes the state of the system, it would not be practical to store the actual values of these indicators within it, as this would lead to an infinite number of possible states that constantly change and would require replication. As minor fluctuations in indicators are acceptable, as noted above, and often do not result in a change in the overall state, an infinite set of states is unnecessary. Instead, we will represent it as a set of three possible states (Figure 1).

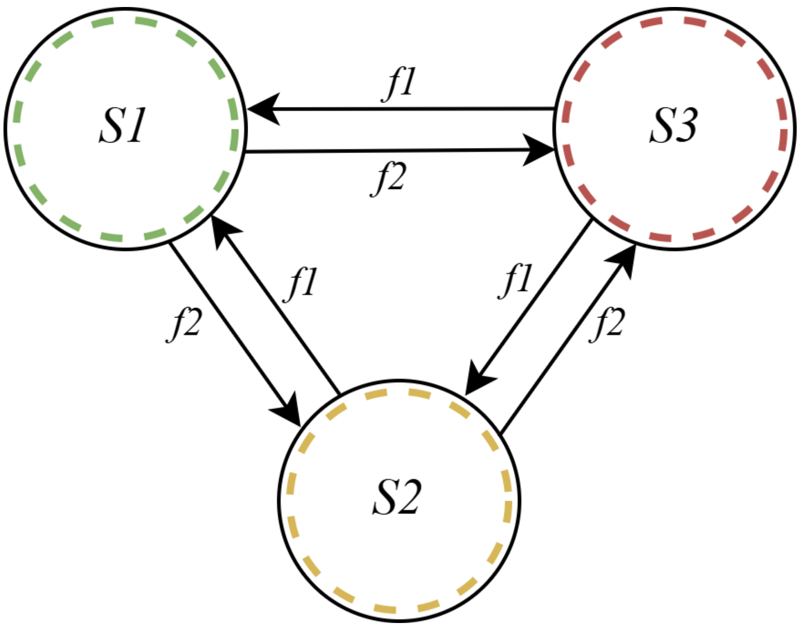

Figure 1. Transition diagram of the power unit state.

In this figure, $S1$ (green) indicates that the supplier has excess capacity, $S2$ (yellow) indicates that the supply meets the demand, and $S3$ (red) indicates that the demand exceeds the supply.

The desired state of any node is $S1$, while $S3$ represents the critical state with the highest balancing priority among neighbouring system nodes. $f1$ denotes a transition from a more desirable state to a less desirable one, defined by the condition $f1 : P_{actual} < P_{nominal}$, while $f2$ denotes the opposite transition, governed by the condition $f2 : P_{actual} > P_{nominal}$.

A state space, represented as a set of potential states relative to the current state, is defined as follows: $\{S1 : \{S2, S3\}, S2 : \{S1, S3\}, S3 : \{S1, S2\}\}$. This state space, with transitions linked to their corresponding conditions, is detailed in Table 1.

Table 1. State space for the node.

	S1	**S2**	**S3**
S1	0	f2	f2
S2	f1	0	f2
S3	f1	f1	0

Data synchronization describes the concept of reconciling a set of data copies (states) or maintaining data integrity. The state reconciliation operation, or consensus, in distributed networks is a procedure for reaching a common agreement on a distributed platform. It is essential for the nodes or entities that are involved in the distributed system to achieve a consensus to ensure reliability and fault tolerance. Reliability and fault tolerance mean that, in a decentralized environment, there are several independent parties that can make their own decisions; some of these parties may act maliciously, but the system must maintain a unified perspective despite their presence [37].

In data exchange within networks where each node has its own copy of the data, ensuring consistency and state alignment is a critical issue. When a large number of nodes attempt to modify data simultaneously, conflicts arise that need to be resolved continuously and efficiently to avoid data discrepancies. Thus, the solution lies in developing methods and approaches that prevent conflicts and ensure data consistency in distributed networks [38].

A conflict-free replicated data type (CRDT) [39] is an abstract data type with a statically defined interface that allows for distributed replication over multiple nodes. It allows for optimistic replication [40] in a principled way. Each replicate can be changed on demand and independently without any overhead coordination between them. When any two replicas have received the same set of updates, even if they were received in different orders, they reach the same state deterministically by adopting mathematically grounded rules to ensure state convergence [41]. Moreover, if concurrent updates of a given data point commute, and all its replicas execute all updates in a causal order, then the values of the replicas converge. This is called a Commutative Replicated Data Type [41]. It is designed to commute concurrent operations. This eliminates the need for complex concurrency control, allowing operations to be executed in different orders while having replicas converge to the same result. This approach guarantees no conflicts, thus removing the need for consensus-based concurrency control [42,43].

Using CRDTs to model the state of a distributed system has the following advantages:

- The data structures are designed to be commutative and convergent [44]. This means that multiple copies of the same data can be independently updated and then merged together without conflicts [45]. This allows for efficient and accurate data analysis, even in cases where the network is disconnected or has high traffic [46,47].
- The CRDT approach relies on predetermined conflict resolution rules, which dictate their semantics. These rules are typically data structure specific [45]. CRDTs provide a mechanism for detecting and resolving conflicts, ensuring consistency across all copies of the data. This is particularly useful in the electrical grid, where multiple copies of state information can be stored in different locations.
- By choosing a state-based CRDT structure [44], which stores the current state of the data rather than the history of changes, they are suitable for real-time monitoring and analysis of data in the electrical grid.

- Scalability: CRDTs enable the construction of distributed systems that can scale horizontally [48] without the need for complex coordination mechanisms [46].
- Fault tolerance: CRDTs are fault-tolerant, as they allow data to be replicated between nodes and recovered in the event of a failure of one or more nodes [49].

Compared to lock/consensus-based methods [50,51], CRDTs have several advantages, as presented in Table 2, which are specifically bounded to synchronization challenges that are unique to distributed electrical grid systems, such as the synchronization of power electronic interfaced distributed generators [52] or for controlling the synchronization of cascading failures in power grids [53].

Table 2. Comparison of CRDT-based synchronization and lock/consensus-based methods.

Criterion	CRDT-Based	Lock/Consensus-Based
Consistency Model	Eventual consistency	Strong consistency
Latency	Low (local updates)	Higher (requires coordination)
Availability	High	Dependent on quorum/leader
Conflict Handling	Automatic and deterministic	Explicit locking or quorum decisions
Scalability	High	Moderate
Use Cases	Collaborative, real-time systems	Mission-critical, transactional systems

All the above advantages allow for the building of highly resilient systems. CRDTs are divided into two types [44]:

1. State-based CRDTs: Convergent Replicated Data Type (CvRDT) [54]—These CRDTs store a complete copy of the state on each network node, which is merged later. These CRDTs support state merge operations to avoid conflicts. A replica periodically propagates its local changes to other replicas by shipping its entire state. A received state is incorporated within the local state via a merge function that deterministically reconciles both states. To maintain convergence, a merge is defined as a join: at least an upper bound over a join semilattice [55]. A major drawback of state-based CRDTs is the communication overhead of shipping the entire state, which can become very large in size [56].
2. Operation-based CRDTs: Commutative Replicated Data Type (CmRDT) [57]—These CRDTs store the operations that have been performed on each network node rather than a complete copy of the state. These CRDTs support the execution order of operations to ensure compatibility. The approach ensures that there are no conflicts and, hence, no need for consensus-based concurrency control [43]. CmRDTs have some advantages, as they can allow for simpler implementations, a concise replica state, and smaller messages; however, they are subject to some limitations. First, they assume a message dissemination layer that guarantees reliable exactly-once causal broadcast; these guarantees are hard to maintain, since large logs must be retained to prevent duplication, even if a TCP is used [58]. Second, membership management is a hard task in op-based systems, especially once the number of nodes becomes larger or due to churn problems, since all nodes must be coordinated by the middleware. Third, the op-based approach requires operations to be executed individually (even when batched) on all nodes [56].

The key comparisons between CvRDT and CmRDT are presented in Table 3.

Based on these types, specific data types have been developed, such as counters, registers, sets, and graphs. Conflict-free replicated data types (CRDTs) can resolve conflicts if the data have certain mathematical properties, such as commutativity, associativity, and idempotency. The problems created by CRDTs include a predefined structure that cannot be changed. Therefore, they cannot be used for all tasks, especially if the operations are not commutative [17].

Table 3. Key comparisons between CvRDT and CmRDT.

Aspect	State-Based (CvRDT)	Operation-Based (CmRDT)
Communication	Full state exchanges	Operation deltas
Complexity	Simple (state merge function)	Complex (requires reliable, causal operation delivery)
Overhead	High (state size grows with dataset)	Low (operation size)
Consistency	Idempotence and associativity ensure eventual consistency	Operations commute to ensure consistency
Message Loss	Resilient (full state re-propagation handles loss)	Vulnerable unless using reliable delivery mechanisms

CRDTs raise important security concerns [59]. A malicious replica could attempt to disrupt the global convergence or consistency by interfering with other replicas; however, this risk can be managed using standard authentication methods [60]. Another challenge, further accentuated with the decentralized and geodistributed nature of cloud-based deployments, is related to the privacy of the data that are stored within the CRDT. This cannot be approached trivially using standard encryption techniques, as secure CRDT protocols would require replica-side computation—on encrypted data—to propagate operations. The outcomes in [27] introduce a foundational security model for CRDTs and provide specialized examples of secure CRDT constructions. Each construction must be carefully designed to use dedicated cryptographic techniques so that the CRDT computations between replicas can be performed over encrypted data [61].

In the current work, a CvRDT with the possibility of spreading the delta of changes between nodes was used. Choosing a state-based CRDT [44], which stores the current state of the data rather than the history of changes, makes it suitable for real-time monitoring and analysis of data in the electrical grids.

To describe the overall state of the system based on the CRDT, which represents a Map data type that can be replicated across the entire set of nodes, according to [62] (p. 6), a modification of the Observed Remove Set (ORSet) [44] (p. 26) can be used. Since the container-type Map is based on Set [44] (p. 21), the implementation of Map will have the same semantics as ORSet, except for the issue of concurrent updates to values, which must be merged in a specific manner. To address this issue, the CRDT LWW-Register [44] (p. 19) was chosen. Using this data type ensures that only the most recently written value is retained, while the previous value (if any) is discarded. Since we are interested in the most recent (current) state of a specific node in the overall system, this strategy is desirable.

Formally, LWW-Register can be defined as a set that is either empty or contains one value: $\{v | wr(v) \in O \wedge \nexists wr(u) \in O \cdot wr(v) < wr(u)\}$ [39] (p. 4). It should also be noted that both ORSet and LWW-Register are Delta State Replicated Data Types [54,56], which reduces the need to send the full state during updates. In other words, when the state of an object changes, only the change itself will be sent, rather than the entire object with the new state. For example, adding elements c and d to the set $\{a, b\}$ will result in sending the delta $\{c, d\}$ and merging this state with the state on the receiving side, resulting in the set $\{a, b, c, d\}$. Therefore, we can formulate a simple formula for state change as follows:

$$S_i' = S_i \circ \Delta_i \tag{1}$$

where S_i in a new state of i's node, S_i, is the previous state of i's node; Δ_i is a change applied on i's node; $\circ$ is the merging changes operation.

The overall state of the system will be represented by a Map data type based on ORSet, which has similar semantics for handling keys. However, in the case of concurrent updates to values, any potential conflict will be resolved in favour of the most recently written value.

A conceptual overview of how CRDT state changes propagate and converge during conflicts among different nodes for the system described in this article can be seen in Figure 2. The state propagation process depicted in Figure 2 is typical of any competitive DEGS network changes, such as instantaneous switch state updates due to low-level sensor changes.

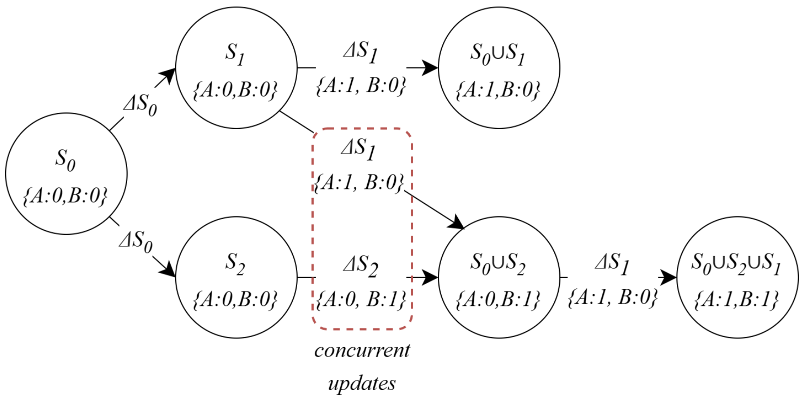

Figure 2. Conceptual view of CRDT state distribution between concurrently updated nodes.

On the basis of Equation (1), we can determine the dataset, which is to be shared by the nodes, as follows:

$$\Delta_i = S'_i - S_i \tag{2}$$

Semantically, each delta represents a set of changes for particular parameters of the node's state, which will be propagated to other nodes and merged with their inner state. For example, based on Figure 2, we can define different deltas: ΔS_1 and ΔS_2. Using Equation (2), the deltas can be rewritten as follows:

$$\Delta S_1 = S'_1 - S_1 = \{A:1, B:0\} - \{A:0, B:0\} = \{A:1, B:0\},$$

$$\Delta S_2 = S'_2 - S_2 = \{A:0, B:1\} - \{A:0, B:0\} = \{A:0, B:1\}$$

We can see how the different deltas, ΔS_1 and ΔS_2, converge during concurrent updates by performing sequential operations on the state S_2. The resulting state will be a union set of changes from ΔS_1 and ΔS_2, which can be rewritten as follows:

$$S_0 \cup S_1 \cup S_2 = (S_0 \circ \Delta S_1) \circ \Delta S_2 = (\{A:0, B:0\} \circ \{A:1, B:0\}) \circ \{A:0, B:1\} = \{A:1, B:1\}$$

Deltas are formed based on the operations defined by the CRDT data structure, which must satisfy idempotence, commutativity, and associativity.

The presented dataset is discrete in nature, effectively representing real-life scenarios, as distributed state representations often operate in quantifiable values or categories rather than continuous ranges. Discrete states simplify modelling by capturing key system behaviours, transitions, and interactions, such as fault conditions, operating modes, or resource availability.

The general representation of the DEGS based on a CRDT can be written as follows:

$$Map[K, LWWRegister[V]] \rightarrow ORSet[(K \rightarrow LWWRegister)]$$

where *Map* denotes a *key* $\rightarrow$ *value* data structure with a generic type *K* for the key and defined node state *V* for the value (Figure 3).

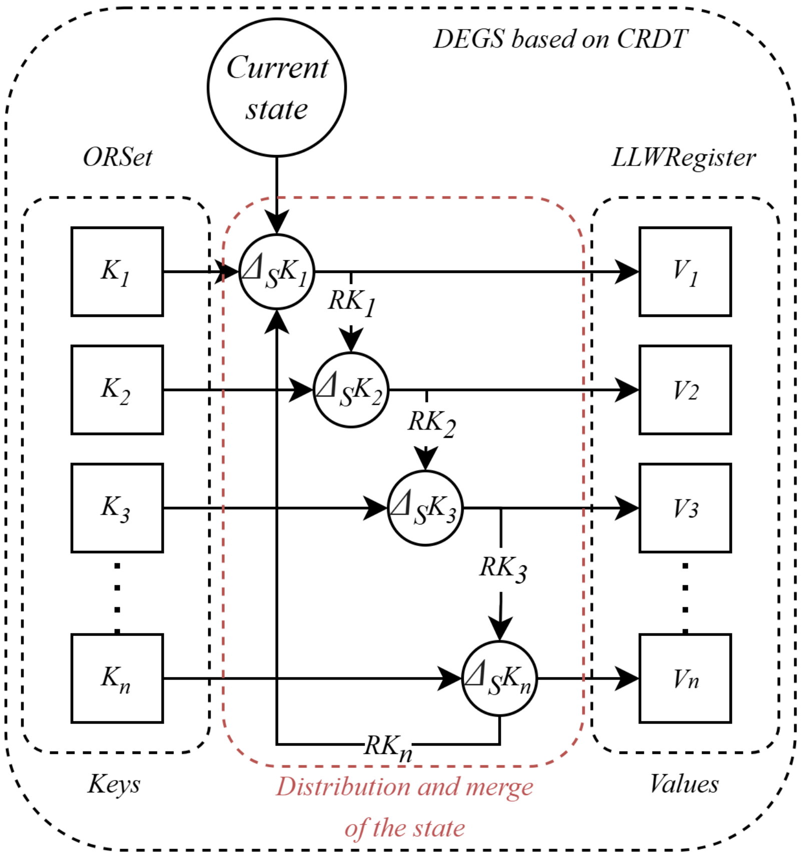

Figure 3. General representation of DEGS based on CRDT.

Here, *Keys* represent the ORset of all possible identifiers of nodes, and *Values* represent LLWRegister structures containing their corresponding states. The *Current state* denotes the state S of a particular node after an initial disturbance Δ_S, which is used to perturb the system's balance through continuous synchronization with other nodes via delta propagation. The synchronization of a particular node produces a unique result RK_i, which is shareable with the next node $i+1$. The result RK_n represents the synchronization outcome for the n-th node, incorporating feedback from the initial conditions. Therefore, achieving balance across the entire DEGS implies no delta transmission or a condition where $RK_n = 0$. This representation highlights the nature of updates, driven by the implementation specifics of the ORset and LLWRegister structures, which is crucial for understanding the experimental results.

If a data type satisfies all three properties—idempotence, commutativity, and associativity—then it is proven to be a Convergent Replicated Data Type.

- Idempotence: A CRDT consists of a set of all added elements and a set of all removed elements. If an attempt is made to *add* or *remove* the same element again, the CRDT remains unmodified. This property avoids duplication and maintains the integrity of the data type, allowing the CRDT to behave consistently across distributed systems by supporting convergence and eventual consistency. Adding (*add* operation) or removing (*remove* operation) an element from the CRDT can be described using Equation (2), where *add* and *remove* are idempotent operations:

$$\begin{aligned} add(a,\mathrm{S}_i) &= (\mathrm{S}_i' \cup \{a\}) = (\mathrm{S}_i' \cup \Delta_i) = (\mathrm{S}_i' \cup (\mathrm{S}_i' \cup -\mathrm{S}_i)) = add(a, add(a,\mathrm{S}_i)) \\ remove(a,\mathrm{S}_i) &= (\mathrm{S}_i' \cup \{a\}) = (\mathrm{S}_i' \cup \Delta_i) = (\mathrm{S}_i' \cup (\mathrm{S}_i' \cup -\mathrm{S}_i)) = remove(a, remove(a,\mathrm{S}_i)) \end{aligned} \quad (3)$$

- Commutativity: Since the *add* and *remove* operations do not depend on each other, the order in which they are executed does not affect the result. This property can be described using Equation (3) as follows:

$$\begin{aligned} add(a,S_i) \to add(b,S_i) = add(b,S_i) \to add(a,S_i) \\ remove(a,S_i) \to remove(b,S_i) = remove(b,S_i) \to remove(a,S_i) \end{aligned} \tag{4}$$

- Associativity: The result of the operations does not depend on the order in which they are applied, regardless of how these operations are grouped. This property can be described based on Equation (4) as follows:

$$add(a, remove(b,S_i)) = remove(a, add(a,S_i))$$

Idempotence, commutativity, and associativity establish fundamental rules that define the set of possible functions and operators that are associated with the described CRDTs. These principles also delineate the range of permissible deltas that can be propagated within specific system topologies. Adhering to these rules ensures the full utilization of CRDT data structures while maintaining the predictability and structure in computations related to continuous and concurrent updates, which is beneficial, even though each node only updates its own part of the state. Any CRDT-based operations mentioned later in this article should be considered a combination of the previously described *add* and *remove* operations.

4. Modelling

For simulation purposes, a special actor-based [63] framework (Vigilant Hawk) was developed by the authors. Its implementation is based on the Akka toolkit [64], which enables the building of highly concurrent systems that are scalable, highly efficient, and resilient by using Reactive Principles [65] and is a great tool to simulate dynamic systems. For example, in [66], the authors introduced a semantics-driven framework to enable demand flexibility control applications in real buildings, and in [67,68], the authors used the Scala/Akka toolkit to create computer frameworks to model large-scale, high-performance, desynchronized information propagation simulations. Using the actor model offers several advantages, particularly for building distributed, asynchronous, and high-load systems. The main benefits of using the actor model [69–71] are as follows:

- Natural Modelling of Asynchronous Processes: The actor model is ideal for asynchronous programming, where tasks run in parallel and independently. Actors communicate via messages without blocking, making it easy to create parallel processes without the need for thread synchronization.
- Improved Scalability: The actor model allows for the addition of new actors or the distribution of actors across nodes with minimal code changes. This makes it possible to scale applications both vertically (on a single server) and horizontally (across multiple servers or nodes).
- State Isolation: Each actor has its own state, which is not directly accessible to other actors. This eliminates the need for locks on shared states, reducing errors related to synchronization. Actors can update their state independently, making the system more robust.
- High Resilience and Self-Recovery: Actors have a built-in supervision system in which one actor can monitor another. If an actor fails, its supervisor can restart it or take other corrective actions, ensuring automatic recovery. This is crucial in distributed systems that need to maintain continuous operation.

- Simplification of Distributed Computing: Since actors communicate through messages, they can exist on different nodes of a distributed system. Actors can interact with each other both locally and over the network, simplifying the development of distributed systems where actors can exchange data regardless of physical location.
- High Performance via Asynchronous Processing: Since actors do not block threads when processing messages, the system can handle a large number of requests simultaneously. This allows for high performance even under heavy loads, as there is no downtime due to resource locking.
- Natural Management of Parallelism: In an actor system, each actor is an isolated unit of parallel computation. This makes it easy to distribute workloads across different CPU cores or even different machines without the complexity that is typically associated with multi-threading.
- Ease of Developing Complex Systems: The actor model is well suited for systems where objects have complex behaviours or interact frequently. Instead of creating complex locking and synchronization mechanisms, actors use messaging to manage their behaviour, simplifying code creation and maintenance.

The developed framework contributes to modelling the asynchronous nature of distributed electrical grids, allowing for a generative nature and the flexibility to represent the nodes of the distributed electric power network by using the actor model. However, the application and further expansion of the software require system domain knowledge and relevant skills of engineers for their implementation.

The framework was developed for the modelling of clustered high-load systems like distributed electrical grids systems, focusing on using different protocols for message delivery. Source files can be found in [72]. Within this framework, clustered electrical grid systems were modelled using a CRDT to represent the state of the cluster nodes. Each node of the cluster is represented as an independent actor, with its internal state [73] being encapsulated and modified only through message passing. This ensures that the state of each node is isolated, reducing the complexity of handling shared mutable states across the system. The cluster's nodes communicate asynchronously without blocking their execution [74]. This allows nodes to remain responsive, even when waiting for external resources or handling other tasks, improving the scalability and reducing latency. An actor is a unique concept [63] that allows us to model each node in the grid, such as a generator, substation, or consumer. Such representations can be created using any paradigm, such as object-oriented or functional representations. The actors are designed to be fault-isolated, i.e., if an actor fails, the system can recover by creating a new actor or reassigning tasks to other actors. Each node as an actor can process requests in parallel, allowing the system to handle a large number of operations concurrently. This approach allows us to gain advantages in terms of concurrency, scalability, and fault tolerance. When a node changes its state, it notifies the replicator of its new state, and these changes become available to all other nodes in the system. Since each node can see the overall state of the system, a node with excess capacity can temporarily redistribute it to meet the urgent needs of other nodes lacking power at a given moment. This redistribution considers priority, giving preference to the nearest nodes based on each node's geographical region.

The simulated cluster consists of electrical unit nodes. Each node has its unique identifier (ID), a region identifier (RI), its own nominal and actual power, two sets with identifiers of other units, and the values of borrowed and returned electricity from/to these units. The structural view of the node can be seen in Figure 4, where the replicator is an abstract unit that is responsible for message distribution, including read and write operations for any Δ_S, across all nodes. A merger is an abstract unit that is responsible for the merging strategy, which defines the process of updating the current node state No_0 to a particular result R with concurrent updates from different nodes. Both the replicator and merger are unified components of each node in the system.

In addition, each electrical unit has its own state (P_i), which reflects the relationship between the nominal ($\mathrm{P}_{nominal}$) and actual (P_{actual}) power:

$$\mathrm{P}_i = \begin{cases} green \to if \quad \mathrm{P}_{actual} \leq \mathrm{P}_{nominal} \\ yellow \to if \quad \mathrm{P}_{actual} = \mathrm{P}_{nominal} \\ red \to if \quad \mathrm{P}_{actual} > \mathrm{P}_{nominal} \end{cases}$$

Each unit has its own view of the state of the grid as a whole $\mathrm{S}_{i,t} \to \mathrm{ID} \cup \mathrm{RI} \cup \mathrm{P}_{i,t} \cup \mathrm{S}_{general,t} - \mathrm{S}_{i,t-1}$, where $\mathrm{S}_{i,t}$ is the state of the node at some time point t and $\mathrm{S}_{general,t} - \mathrm{S}_{i,t-1}$ points to the general state of the system with respect to a previous state of i's node in $t-1$, including the states of other nodes. Each node is responsible for publishing changes to its state in this data structure and reading updates from other nodes whenever there are changes. In our experiment, we have 100 nodes that are in a green state. Through an external influence, we increase the actual power in the observed node of the system so that it moves to the red state. We observe the state of the electrical grid as a whole (from the perspective of a random node in the system) over time while the system balances (nodes in a red state transition to a yellow state by borrowing power from neighbouring nodes and nodes in a green state transition to a yellow state after borrowing this power). Therefore, we compose a view of the whole system based on a single node, which allows us to calculate the messaging based on dynamic distancing.

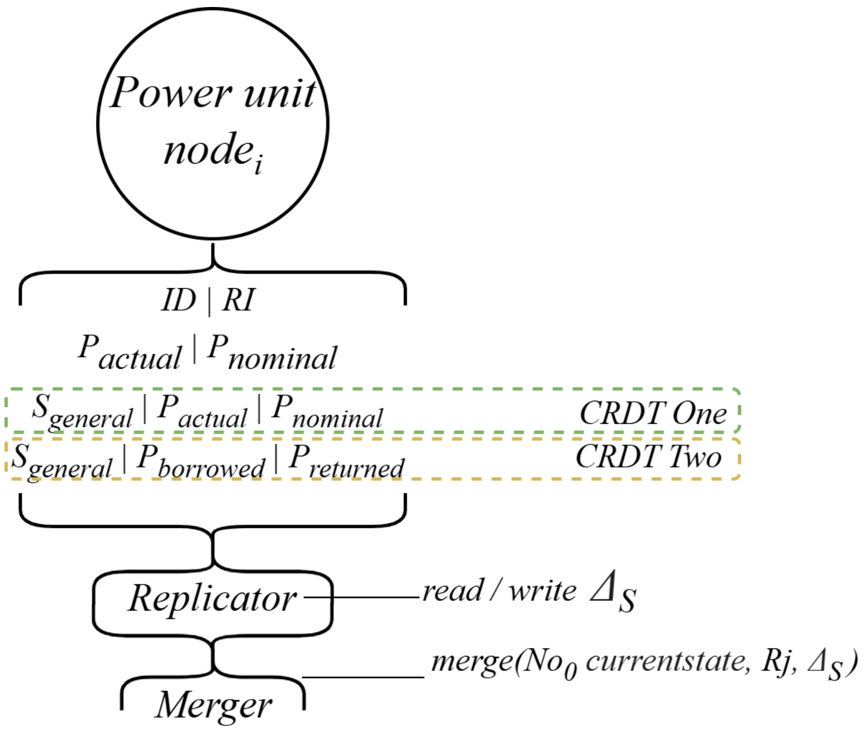
Power unit node$_i$
ID | RI
P_{actual} | $P_{nominal}$
$S_{general}$ | P_{actual} | $P_{nominal}$
CRDT One
$S_{general}$ | $P_{borrowed}$ | $P_{returned}$
CRDT Two
Replicator
read / write Δ_S
merge(No_0 currentstate, Rj, Δ_S)
Merger

Figure 4. Structural view of the node.

The experiment flow in terms of the inner node communication is presented in Figure 5, where observed is the observable node, D is the actual disturbance (in our case, it is based on the actual and nominal power of the node), Δ_S is the delta of the state to be shared with the nearest nodes, D_k is the depth coefficient, which shows the number of layers of nodes from the perspective of the observed node, R is the resulting state, $f(D, O)$ is the process of CRDT message distribution from the observed node to other nodes and vice versa, and

No is the nearest node, which has established a communication channel with the observed node. NoO means that each $f(D,O)$ transfer of Δ_S occurs in the same way as if each No was becoming an observer; then, the actual observer becomes the corresponding No, and M is the merge function (which incorporates the merging strategy), which accepts Δ_S and (if it exists) the corresponding R for inner state changes and the creation of an intermediate result. M can be interpreted using Equation (1). For example, the merge function for No_i can be written as $\left(No_i currentstate, \mathrm{R}_j, \Delta_S\right) \rightarrow \mathrm{R}_i \vee No_i statechange$.

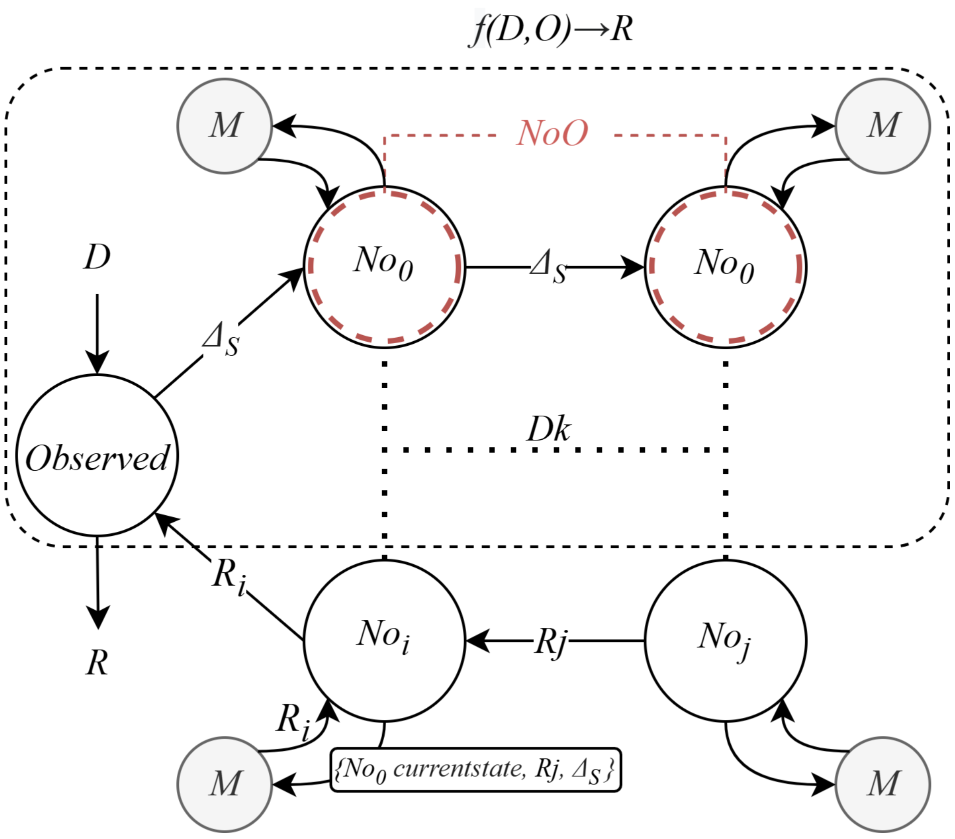

Figure 5. Experimental flow of inner node communication.

It should be noted that each node incorporates a merge function between its actual state, Δ_S, and incoming result; therefore, there may be collisions between Δ_S and R, but based on the previous sections of this paper, we can assume that any conflicts will be resolved during the CRDT message distribution.

The actual experiment consisted of several phases with different timestamps to simulate latency degradation, where the interval ranged from 9 to 45 s. The hardware configuration for the simulation consisted of an Apple M3 Pro, 36 GB RAM, and a 512 GB SSD disk. The specific set of experiments, including the latency degradation and message distribution density, was concluded.

The latency degradation coefficient (LDK) serves as a dynamically generated multiplier for modelling the distance between nodes, enabling the introduction of delays in message delivery and making the system non-ideal.

The full state distribution time (FSDT) is defined as the time that is required for the state change of a specific node to propagate to 100% of the other nodes. This metric provides a qualitative estimate of the effect of delays on the state propagation speed across the system.

To show how messages spread across the network, the view from the perspective of undisturbed nodes in the system was used. We considered a generalized network, where nodes were interconnected in a peer-to-peer manner. The network topology was assumed to be fully connected, or nearly so, with LDK serving as a uniform parameter across all links in this experiment.

Deterministic or stateful grid initialization, non-blocking monitoring, and carefully encapsulated states ensured the reproducibility and reliability of our results under any conjecture of grid disturbances, leveraging the previously mentioned CRDT properties and actor-based implementation.

5. Modelling Results

The simulation results are presented in Figures 6 and 7, where time zero is defined as the moment when all nodes reach an initial state of balance. This balance is characterized by the complete synchronization of system states and the seamless representation of communication through the transmission of state change deltas.

After introducing a disturbance at a single node, we monitor how quickly the updated state propagates to the other nodes. This is achieved using a non-blocking recording of state indicators in a separate file in a raw format: $\{node_{id}, P_{actual}, P_{nominal}, state_{current}, time_{current}\}$. These data are later utilized to generate graphs and calculate the percentage ratio of informed nodes, as well as FSDT, since the actual state change is quantified numerically, while the interval between the first and last updates represents the distribution time.

In the graphs, the $x - axis(n)$ represents the number of nodes, which is a statically bounded value, while the $y - axis(t)$ represents the time in milliseconds.

A general overview of the representation of the distributed electrical grid's states using a CRDT was made. A formal model with a corresponding structure was reviewed. A special actor-based framework was developed for the simulation of the distributed high-load electrical grids and used for the deeper examination of the actual advantages of CRDTs in terms of message delivery in distributed systems.

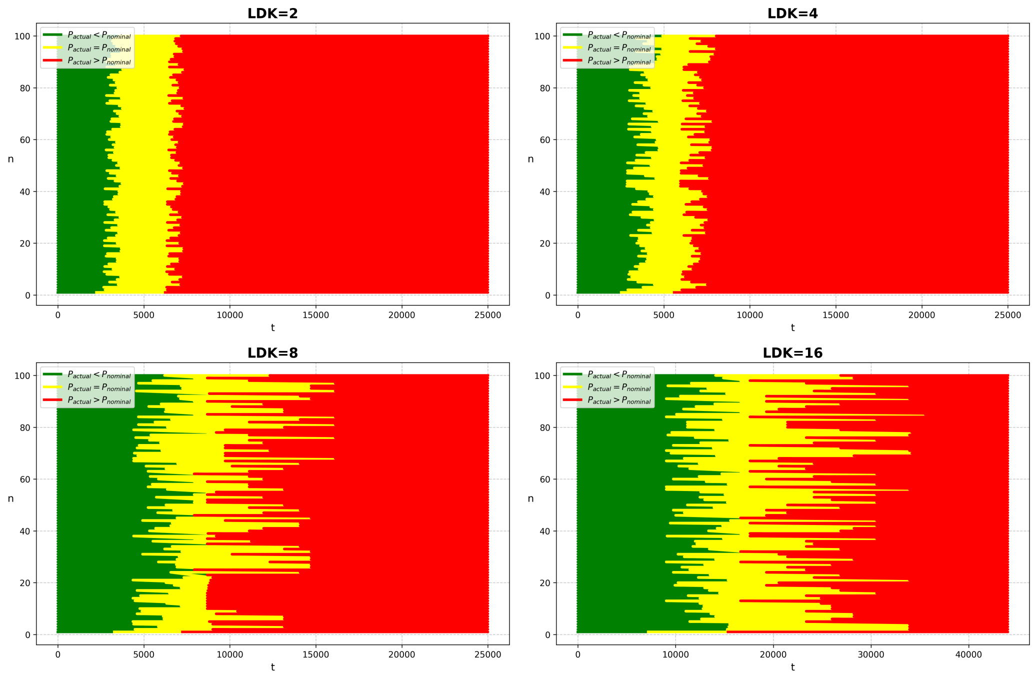

Figure 6. Simulation results for CRDT message distribution density within electrical grid with LDK = 2, 4, 8, 16.

It was found that increasing the distancing between nodes with latency degradation can cause turbulences, making the system more complex to manage in various forms. We could also observe that there is an effective LDK value (in our case, $LDK_{effective} = \{2, 4\}$)

at which the network does not experience significant disturbances. Therefore, to optimize $LDK_{effective}$, it is advisable to divide the cluster into smaller sub-clusters. Naturally, such a division would require a specific topology and a specialized inter-cluster communication protocol to maintain connectivity within larger network groupings. Additionally, it is worth noting that under $LDK_{effective}$, the system responded smoothly and predictably.

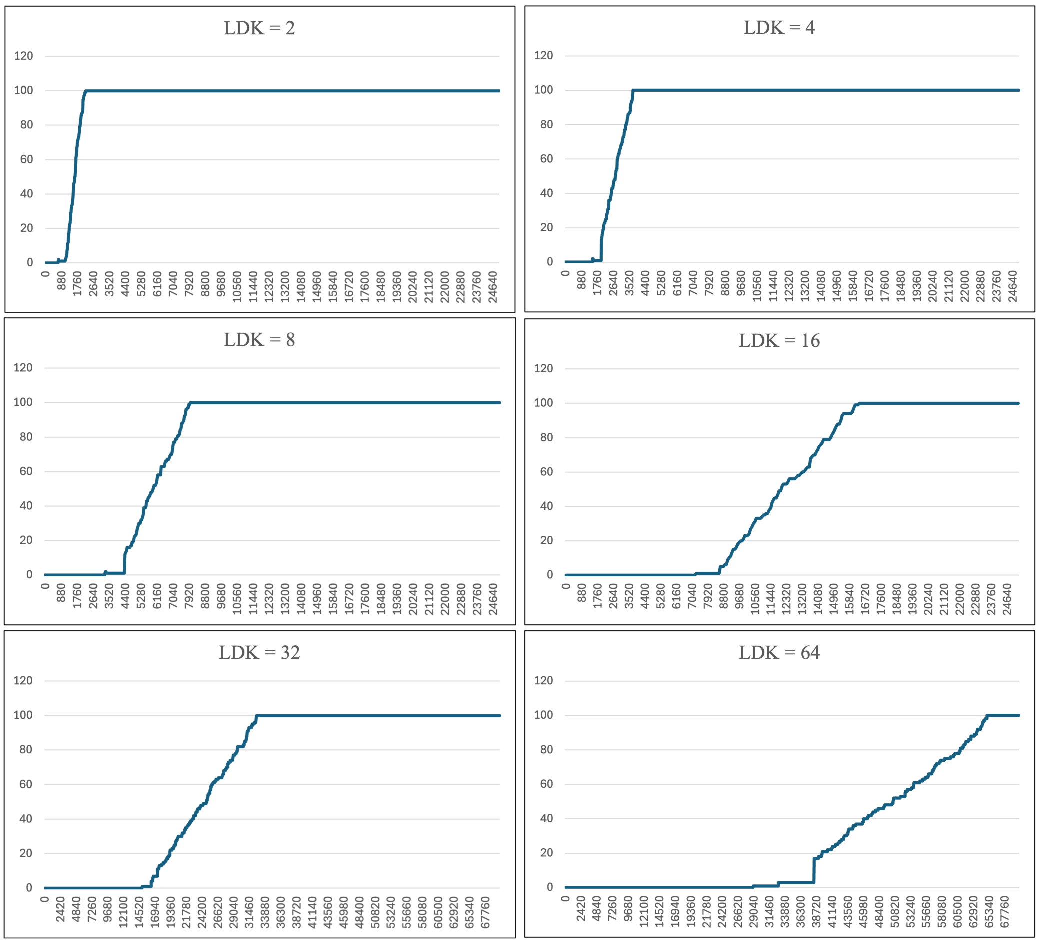

Figure 7. Simulation results for network state synchronization with LDK = 2, 4, 8, 16, 32, 64 and FSDT.

We observe that the FSDT increases with the LDK; however, the process of message distribution among other nodes remains stable and predictable. Essentially, there is a linear relationship between the FSDT and LDK, allowing for the current solution to scale. It is also worth noting that an inflated average value of the LDK was used in this experiment to demonstrate the performance of the DEGS based on a CRDT during a general network connectivity degradation.

6. Discussion and Conclusions

The application of conflict-free replicated data types (CRDTs) to distributed electrical grid systems (DEGSs) presents a novel approach to addressing the challenges of state synchronization in decentralized environments. Our study demonstrates that CRDTs can effectively maintain consistency and ensure deterministic behaviour in DEGSs, even under conditions of network latency and node disturbances.

One of the key findings from our simulations is the identification of an effective latency degradation coefficient (LDK), specifically $LDK_{effective} = \{2, 4\}$, where the network maintains stability without significant disturbances. This suggests that DEGSs can tolerate certain levels of latency without compromising state synchronization, which is critical for real-time monitoring and control. The linear relationship observed between the full state distribution time (FSDT) and LDK indicates that the system's scalability is achievable without introducing nonlinear complexities in message dissemination. Such a relationship between the FSDT and LDK will allow for better resource allocation and scalability planning for optimized network topologies.

Our use of the actor-based framework, leveraging the Akka toolkit, has proven advantageous in modelling the asynchronous and concurrent nature of the DEGS. The actor model's inherent support for scalability, fault tolerance, and high concurrency aligns well with the requirements of modern electrical grids, which are increasingly distributed and subjected to high loads. By encapsulating each grid node as an independent actor with isolated state management, we have minimized the risks associated with shared mutable states and have enhanced the system's resilience to node failures.

The experimental results affirm that CRDTs can facilitate efficient state synchronization in DEGSs. The CRDTs' properties of idempotence, commutativity, and associativity ensure that concurrent updates from multiple nodes converge to a consistent global state, without the need for complex concurrency control mechanisms. This is particularly beneficial in scenarios where nodes may experience intermittent connectivity or when the network is subject to high traffic volumes.

However, this study also highlights certain limitations and areas for further exploration. The increase in FSDT with higher LDK values indicates that while the system remains stable, the time that is required for state changes to propagate throughout the network can become significant. This delay may not be acceptable in scenarios where rapid response times are critical, such as in the mitigation of sudden load imbalances or fault conditions. To address this, it may be necessary to implement strategies such as dividing the network into smaller sub-clusters to reduce propagation delays, albeit at the cost of increased complexity in inter-cluster communication and coordination.

Another consideration is the potential impact of the network topology on the message distribution efficiency. Although our simulations assumed a certain degree of homogeneity in node connectivity, real-world electrical grids may exhibit more complex and heterogeneous network structures. Future work should investigate how different topologies influence CRDT-based synchronization and whether adaptive algorithms, such as adaptive message routing [75] or the Gossip protocol [76], can optimize the message routing based on real-time network conditions. For example, adaptive message routing improves a network's performance, fault tolerance and resilience, resource utilization, latency, and energy efficiency.

Security concerns associated with CRDTs, as noted in prior research [59,60,77], are also pertinent to DEGSs. Ensuring the integrity and confidentiality of state information is paramount, especially given the critical nature of electrical grid infrastructure. Future development should focus on incorporating robust authentication mechanisms and exploring secure CRDT protocols that can operate over encrypted data without compromising performance. These are essential areas for addressing potential conflicts or breaches in state updates caused by malicious nodes during CRDT-based synchronization.

The simplification of node states to three discrete levels (green, yellow, red) provides a practical approach to modelling system behaviour without the overhead of managing continuous state variables. The proposed simplified strategy can also consider stochastic aspects, such as a variable grid's topology. However, this abstraction may overlook nuances in grid dynamics that could be important for finer-grained control and optimization, such as the effect of the quality of the power flow representation [78] and the need for a certain degree of complexity to obtain sufficiently accurate results in energy system models [79]. This can impact the system reliability and precision by reducing the set of possible states of the system, which leads to deconcretization and additional complications when extrapolating conclusions to implement similar simulations. Incorporating additional state levels or adopting a hybrid approach that combines discrete and continuous state representations might offer improved fidelity without sacrificing the benefits of CRDTs.

Overall, the adoption of CRDTs for state synchronization in DEGSs shows significant promise in enhancing system resilience and operational efficiency. The deterministic and conflict-free nature of CRDTs aligns well with the demands of distributed electrical grids, where timely and accurate state information is critical for maintaining balance and preventing failures. By addressing the identified limitations and extending the framework to accommodate more complex network conditions and security requirements, the proposed approach can contribute to the development of more robust and scalable DEGS architectures.

In conclusion, the system behaved predictably and maintained consistency after the disturbance. Despite the fact that the state synchronization time increased during significant disturbances, the nodes remained in a deterministic state, which prevented additional errors or network failures. This allows for efficient and accurate data analysis. The experimental results presented above demonstrated that the proposed solution is both scalable and fault-tolerant within the context of the effectiveness of the application of the CRDT for the synchronization of the distributed electric power grid system. Thanks to the state-based CRDT structure, the corresponding DEGSs are well suited for real-time monitoring and analysis of data in the electrical grids.

The next research steps could cover a wider range of different topics. Directions could include the development of special protocols for inner and outer electrical grid communication using heterogeneous latencies to account for varying communication delays between nodes, which can significantly impact the synchronization efficiency and overall system performance, including cybersecurity aspects of the replicated data types, since it is expected that larger clustered systems can be divided into separate ones. The main challenge is performing cryptographic operations efficiently at scale, ensuring that nodes can validate the authenticity and integrity of states without a high overhead. Techniques like threshold signatures or secure enclaves could be explored. Developing a state management system based on the described data state would be highly beneficial. Such a system would enable the integration of various approaches to data distribution, as well as monitoring, threading, and the analysis, management, and control of associated risks across the entire system. The authors also consider it expedient to develop a universal approach to creating the topology of a distributed electrical grid system.

This would allow for the use of CRDTs on different levels without any overhead based on the design of individual parts of the system. Independently clustered topologies would also provide a direct way to achieve scalability and resolve the issues mentioned earlier. This can be a very tough problem, because each cluster can have its own architecture and non-functional requirements (NFRs). It is also crucial to know the behaviour of the message distribution regarding the distance between nodes and state management issues.

Author Contributions: A.P.: Conceptualization, methodology, formal analysis, writing—original draft preparation, and visualization. I.P.: Methodology, software, formal analysis, data curation, and writing—original draft preparation. M.Y.: Investigation, resources and editing, and visualization. D.S.: Investigation and resources and editing. H.K.: Supervision, data curation, and writing—review and editing. V.A.: Supervision, data curation, visualization, and writing—review and editing. All authors have read and agreed to the published version of the manuscript.

Funding: The research was conducted as part of the projects 'Development of Methods and Means of Increasing the Efficiency and Resilience of Local Decentralized Electric Power Systems in Ukraine' and 'Development of Distributed Energy in the Context of the Ukrainian Electricity Market Using Digitalization Technologies and Systems', implemented under the state budget program 'Support for Priority Scientific Research and Scientific-Technical (Experimental) Developments of National Importance' (CPCEL 6541230) at the National Academy of Sciences of Ukraine.

Data Availability Statement: The raw data supporting the conclusions of this article will be made available by the authors upon request.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Don Chua, W.F.; Lim, C.L.; Koh, Y.Y.; Kok, C.L. A Novel IoT Photovoltaic-Powered Water Irrigation Control and Monitoring System for Sustainable City Farming. *Electronics* **2024**, *13*, 676. [CrossRef]
2. Tsao, Y.C.; Vu, T.L. Power supply chain network design problem for smart grid considering differential pricing and buy-back policies. *Energy Econ.* **2019**, *81*, 493–502. [CrossRef]
3. Dkhili, N.; Eynard, J.; Thil, S.; Grieu, S. A survey of modelling and smart management tools for power grids with prolific distributed generation. *Sustain. Energy Grids Netw.* **2020**, *21*, 100284. [CrossRef]
4. Shan, L.-Z.; Yamane, K.; Ono, T.; Kawamura, T.; Wu, W.C.; Hu, Z.C.; Wang, Q.; Wen, Y.L. Distributed Energy Resource Management System with improved convergence. *Appl. Energy* **2024**, *371*, 123566. [CrossRef]
5. Monaco, R.; Bergaentzle, C.; Leiva Vilaplana, J.A.; Ackom, E.; Nielsen, P.S. Digitalization of power distribution grids: Barrier analysis, ranking and policy recommendations. *Energy Policy* **2024**, *188*, 114083. [CrossRef]
6. Sapatnekar, S.S.; Su, H. Analysis and optimization of power grids. *IEEE Des. Test Comput.* **2003**, *20*, 7–15. [CrossRef]
7. Liu, T.; Song, Y.; Zhu, L.; Hill, D.J. Stability and Control of Power Grids. *Annu. Rev. Control. Robot. Auton. Syst.* **2022**, *5*, 689–716. [CrossRef]
8. Chen, Z.; Amani, A.M.; Yu, X.; Jalili, M. Control and Optimisation of Power Grids Using Smart Meter Data: A Review. *Sensors* **2023**, *23*, 2118. [CrossRef]
9. Khan, N.; Shahid, Z.; Alam, M.M.; Abu Bakar Sajak, A.; Mazliham, M.S.; Ahmed Khan, T.; Rizvi, S.S.A. Energy Management Systems Using Smart Grids: An Exhaustive Parametric Comprehensive Analysis of Existing Trends, Significance, Opportunities, and Challenges. *Int. Trans. Electr. Energy Syst.* **2022**, *2022*, 1–38. [CrossRef]
10. Ajaz, W.; Bernell, D. Microgrids and the transition toward decentralized energy systems in the United States: A Multi-Level Perspective. *Energy Policy* **2021**, *149*, 112094. [CrossRef]
11. Carareto, R.; Baptista, M.S.; Grebogi, C. Natural synchronization in power-grids with anti-correlated units. *Commun. Nonlinear Sci. Numer. Simul.* **2013**, *18*, 1035–1046. [CrossRef]
12. Taher, H.; Olmi, S.; Scholl, E. Enhancing power grid synchronization and stability through time-delayed feedback control. *Phys. Rev. E* **2019**, *100*, 062306. [CrossRef] [PubMed]
13. Cuadra, L.; Salcedo-Sanz, S.; Del Ser, J.; Jiménez-Fernández, S.; Geem, Z.W. A Critical Review of Robustness in Power Grids Using Complex Networks Concepts. *Energies* **2015**, *8*, 9211–9265. [CrossRef]
14. Budrikis, Z. Stability of power grids. *Nat. Rev. Phys.* **2022**, *4*, 635. [CrossRef]
15. Galesky, L.F.; Rodrigues, L.A.; Duarte, E.P., Jr.; Arantes, L. Efficient Synchronization of CRDTs using VCube-PS. In Proceedings of the 12th Latin-American Symposium on Dependable and Secure Computing (LADC'23), La Paz, Bolivia, 16–18 October 2023; Association for Computing Machinery: New York, NY, USA, 2023; pp. 50–59. [CrossRef]
16. Galesky, L.F.; Rodrigues, L.A. Efficient CRDT Synchronization at Scale using a Causal Multicast over a Virtual Hypercube Overlay. In Proceedings of the 11th Latin-American Symposium on Dependable Computing (LADC'22), Fortaleza, Brazil, 21–24 November 2022; Association for Computing Machinery: New York, NY, USA, 2022; pp. 84–88. [CrossRef]
17. Simić, M.; Stojkov, M.; Sladić, G.; Milosavljević, B. CRDTs as replication strategy in large-scale edge distributed system: An overview. In Proceedings of the ICIST 2020: Proceedings of the 10th International Conference on Information Systems and Technologies, Lecce, Italy, 4–5 June 2020; Zdravković, M., Konjović, Z., Trajanović, M., Eds.; pp. 46–50.
18. Zhang, Y.; Wang, J.K.; Han, Y.J. CChain: A high throughput blockchain system. In Proceedings of the Proc. SPIE 12599, Second International Conference on Digital Society and Intelligent Systems (DSInS 2022), 125990K (3 April 2023), Chengdu, China, 2–4 December 2022. [CrossRef]
19. Shi, P.; Cui, Y.; Xu, K.; Zhang, M.; Ding, L. Data Consistency Theory and Case Study for Scientific Big Data. *Information* **2019**, *10*, 137. [CrossRef]
20. Gilbert, S.; Lynch, N. Brewer's conjecture and the feasibility of consistent, available, partition-tolerant web services. *SIGACT News* **2002**, *33*, 51–59. [CrossRef]
21. Kleppmann, M. A Critique of the CAP Theorem. *arXiv* **2015**, arXiv:1509.05393.
22. Zhao, X.; Haller, P. Replicated data types that unify eventual consistency and observable atomic consistency. *J. Log. Algebr. Methods Program.* **2020**, *114*, 100561. [CrossRef]

23. Barreto, A.; Paulino, H.; Silva, J.A.; Preguiça, N. PS-CRDTs: CRDTs in highly volatile environments. *Future Gener. Comput. Syst.* **2023**, *141*, 755–767. [CrossRef]
24. Lv, X.; He, F.; Cheng, Y.; Wu, Y. A novel CRDT-based synchronization method for real-time collaborative CAD systems. *Adv. Eng. Inform.* **2018**, *38*, 381–391. [CrossRef]
25. Acher, Q.; Ignat, C.; Ibrahim, S. Quantifying the Performance of Conflict-free Replicated Data Types in InterPlanetary File System. In Proceedings of the 4th International Workshop on Distributed Infrastructure for the Common Good (DICG'23), Bologna, Italy, 11–15 December 2023; Association for Computing Machinery: New York, NY, USA, 2023; pp. 19–24. [CrossRef]
26. Li, C.; Porto, D.; Clement, A.; Gehrke, J.; Preguiça, N.M.; Rodrigues, R. Making geo-replicated systems fast as possible, consistent when necessary. In Proceedings of the 10th USENIX Symposium on Operating Systems Design and Implementation (OSDI 12), Hollywood, CA, USA, 8–10 October 2012; USENIX Association: Berkeley, CA, USA, 2012; pp. 265–278.
27. Cai, W.; He, F.; Lv, X. Multi-core accelerated CRDT for large-scale and dynamic collaboration. *J. Supercomput.* **2022**, *78*, 10799–10828. [CrossRef]
28. Gomes, V.B.F.; Kleppmann, M.; Mulligan, D.P.; Beresford, A.R. Verifying strong eventual consistency in distributed systems. *Proc. ACM Program. Lang. (PACMPL)* **2017**, *1*, 1–28. [CrossRef]
29. Ahuja, A.; Gupta, G.; Sidhanta, S. Edge Applications: Just Right Consistency. In Proceedings of the 2019 38th Symposium on Reliable Distributed Systems (SRDS), Lyon, France, 1–4 October 2019; pp. 351–3512. [CrossRef]
30. Al-Darrab, A.I.; Rushdi, A.M.A. Multi-State Reliability Evaluation of Local Area Networks. In Proceedings of the 2021 National Computing Colleges Conference (NCCC), Taif, Saudi Arabia, 27–28 March 2021; pp. 1–6. [CrossRef]
31. Weikert, D.; Steup, C.; Mostaghim, S. Availability-Aware Multiobjective Task Allocation Algorithm for Internet of Things Networks. *IEEE Internet Things J.* **2022**, *9*, 12945–12953. [CrossRef]
32. Tariverdi, A.; Torresen, J. Rafting Towards Consensus: Formation Control of Distributed Dynamical Systems. *arXiv* **2023**. [CrossRef]
33. Alahmad, Y.; Agarwal, A. Multiple objectives dynamic VM placement for application service availability in cloud networks. *J. Cloud Comp.* **2024**, *13*, 46. [CrossRef]
34. *DSTU EN 50160:2014*; Voltage Characteristics of Electricity Supplied by Public Electricity Networks. State Standard of Ukraine (DSTU); Derzhspozhyvstandart Ukrainy: Kyiv, Ukraine, 2024. Available online: https://dnaop.com/html/61662/doc-%D0%94%D0%A1%D0%A2%D0%A3_EN_50160_2014 (accessed on 27 November 2024).
35. *IEEE 1547-2018*; IEEE Standard for Interconnection and Interoperability of Distributed Energy Resources with Associated Electric Power Systems Interfaces. Available online: https://standards.ieee.org/ieee/1547/5915/ (accessed on 27 November 2024).
36. *IEEE C37.90-2005*; IEEE Standard for Relays and Relay Systems Associated with Electric Power Apparatus. Available online: https://standards.ieee.org/ieee/C37.90/3284/ (accessed on 27 November 2024).
37. Masood, F.; Faridi, A.R. Consensus Algorithms In Distributed Ledger Technology For Open Environment. In Proceedings of the 2018 4th International Conference on Computing Communication and Automation (ICCCA), Greater Noida, India, 14–15 December 2018; pp. 1–6. [CrossRef]
38. Tanenbaum, A.S.; van Steen, M. *Distributed Systems: Principles and Paradigms*, 2nd ed.; CreateSpace: Scotts Valley, CA , USA, 2016.
39. Preguiça, N. Conflict-free Replicated Data Types: An Overview. *arXiv* **2018**, arXiv:1806.10254.
40. Saito, Y.; Shapiro, M. Optimistic replication. *ACM Comput. Surv.* **2005**, *37*, 42–81. [CrossRef]
41. Almeida, P.S. Approaches to Conflict-free Replicated Data Types. *arXiv* **2024**, arXiv:2310.18220. [CrossRef]
42. Shapiro, M.; Preguiça, N.; Baquero, C.; Zawirski, M. *Conflict-Free Replicated Data Types*; Lecture Notes in Computer Science; Springer: Berlin/Heidelberg, Germany, 2011; pp. 386–400. [CrossRef]
43. Letia, M.; Preguiça, N.; Shapiro, M. CRDTs: Consistency without Concurrency Control. *arXiv* **2009**, arXiv:0907.0929.
44. Shapiro, M; Preguiça, N.; Baquero, C.; Zawirski, M. *A Comprehensive Study of Convergent and Commutative Replicated Data Types*; RR-7506, Inria—Centre Paris-Rocquencourt; INRIA: Le Chesnay, France, 2011; p. 50. Available online: https://hal.inria.fr/file/index/docid/555588/filename/techreport.pdf (accessed on 27 November 2024).
45. Saquib, N.; Krintz, C.; Wolski, R. Log-Based CRDT for Edge Applications. In Proceedings of the 2022 IEEE International Conference on Cloud Engineering (IC2E), Pacific Grove, CA, USA, 26–30 September 2022; pp. 126–137. [CrossRef]
46. Qayyum, O.; Yu, W. Coordination-Free Replicated Datalog Streams with Application-Specific Availability. In *Advances in Databases and Information Systems. ADBIS 2024*; Tekli, J., Gamper, J., Chbeir, R., Manolopoulos, Y., Eds.; Lecture Notes in Computer Science; Springer: Cham, Switzerland, 2024; Volume 14918, pp. 155–169. [CrossRef]
47. Kleppmann, M.; Mulligan, D.P.; Gomes, V.B.; Beresford, A.R. A Highly-Available Move Operation for Replicated Trees. *IEEE Trans. Parallel Distrib. Syst.* **2022**, *33*, 1711–1724. [CrossRef]
48. Biletskyy, B.O. Horizontal and vertical scalability of machine learning methods. *Probl. Program.* **2019**, *2*, 69–80. [CrossRef]
49. Balazinska, M.; Balakrishnan, H.; Madden, S.R.; Stonebraker, M. Fault tolerance in the borealis distributed stream processing system. *ACM Trans. Database Syst.* **2018**, *33*, 1–44. [CrossRef]

50. Ongaro, D.; Ousterhout, J. In search of an understandable consensus algorithm. In Proceedings of the 2014 USENIX Conference on USENIX Annual Technical Conference (USENIX ATC'14), Philadelphia PA, USA, 19–20 June 2014; Association for Computing Machinery: Philadelphia, PA, USA, 2024; pp. 305–320. https://dl.acm.org/doi/10.5555/2643634.2643666.
51. Bolosky, W.J.; Bradshaw, D.; Haagens, R.B.; Kusters, N.P.; Li, P. Paxos replicated state machines as the basis of a high-performance data store. In Proceedings of the 8th USENIX Conference on Networked Systems Design and Implementation (NSDI'11), Boston, MA, USA, 30 March–1 April 2011; USENIX Association: Boston, MA, USA, 2011; pp. 141–154. https://dl.acm.org/doi/10.5555/1972457.1972472.
52. Oshnoei, A.; Peyghami, S.; Mokhtari, H.; Blaabjerg, F. Grid Synchronization for Distributed Generations. In *Encyclopedia of Sustainable Technologies*, 2nd ed.; Elsevier: Amsterdam, The Netherlands, 2024; pp. 517–537. [CrossRef]
53. Olmi, S.; Gambuzza, L.V.; Frasca, M. Multilayer control of synchronization and cascading failures in power grids. *Chaos Solitons Fractals* **2024**, *180*, 517–537. [CrossRef]
54. Enes, V.; Almeida, P.S.; Baquero, C.; Leitão, J. Efficient Synchronization of State-Based CRDTs. In Proceedings of the 2019 IEEE 35th International Conference on Data Engineering (ICDE), Macao, China, 8–11 April 2019; pp. 148–159. [CrossRef]
55. Nation, J.B. Semilattices, Lattices and Complete Lattices. In *Revised Notes on Lattice Theory*; University of Hawaii Math Department: Honolulu, HI, USA, 1990. Available online: https://math.hawaii.edu/~jb/books.html (accessed on 27 November 2024).
56. Almeida, P.S.; Shoker, A.; Baquero, C. Delta State Replicated Data Types. *J. Parallel Distrib. Comput.* **2018**, *111*, 162–173. [CrossRef]
57. Baquero, C.; Almeida, P.S.; Shoker, A. Making Operation-Based CRDTs Operation-Based. In Proceedings of the 14th IFIP International Conference on Distributed Applications and Interoperable Systems, DAIS 2014, Berlin, Germany, 3–5 June 2014; Springer: Berlin/Heidelberg, Germany, 2014; Volume 8460, pp. 126–140._11. [CrossRef]
58. Helland, P. Idempotence Is Not a Medical Condition: An essential property for reliable systems. *Queue* **2012**, *10*, 30–46. [CrossRef]
59. Preguiça, N.; Baquero, C.; Shapiro, M. Conflict-Free Replicated Data Types (CRDTs). In *Encyclopedia of Big Data Technologies*; Springer: Berlin/Heidelberg, Germany, 2019.
60. Jannes, K.; Lagaisse, B.; Joosen, W. Secure replication for client-centric data stores. In Proceedings of the DICG 2022, Quebec, QC, Canada, 7 November 2022; ACM: New York, NY, USA, 2022; pp. 31–36. [CrossRef]
61. Portela, B.; Pacheco, H.; Jorge, P.; Pontes, R. General-Purpose Secure Conflict-free Replicated Data Types. In Proceedings of the 2023 IEEE 36th Computer Security Foundations Symposium (CSF), Dubrovnik, Croatia, 10–14 July 2023; pp. 521–536. [CrossRef]
62. An Optimized Conflict-Free Replicated Set. Available online: https://hal.inria.fr/file/index/docid/738680/filename/RR-8083.pdf (accessed on 27 November 2024).
63. Hewitt, C. Actor Model of Computation: Scalable Robust Information Systems. *arXiv* **2015**, arXiv:1008.1459.
64. Akka Toolkit. Available online: https://akka.io/ (accessed on 27 November 2024)
65. The Reactive Manifesto. Available online: https://www.reactivemanifesto.org/ (accessed on 27 November 2024).
66. Pereira, F.d.A.; Katsigarakis, K.; Rovas, D.; Pritoni, M.; Shaw, C.; Paul, L.; Prakash, A.; Martin-Toral, S.; Finn, D.; O'Donnell, J. A semantics-driven framework to enable demand flexibility control applications in real buildings. *Adv. Eng. Inform.* **2025**, *64*, 103049. [CrossRef]
67. Janczykowski, M.; Turek, W.; Malawski, M.; Byrski, A. Large-scale urban traffic simulation with Scala and high-performance computing system. *J. Comput. Sci.* **2019**, *35*, 91–101. [CrossRef]
68. Bujas, J.; Dworak, D.; Turek, W.; Byrski, A. High-performance computing framework with desynchronized information propagation for large-scale simulations. *J. Comput. Sci.* **2019**, *32*, 70–86. [CrossRef]
69. Garnock-Jones, T. History of Actor, Lecture Notes Harvard University. 2016. Available online: https://groups.seas.harvard.edu/courses/cs252/2016fa/12.pdf (accessed on 27 November 2024).
70. Camilleri, C.; Vella, J.G.; Nezval, V. Horizontally Scalable Implementation of a Distributed DBMS Delivering Causal Consistency via the Actor Model. *Electronics* **2024**, *13*, 3367. [CrossRef]
71. Agha, G.A.; Mason, I.A.; Smith, S.F.; Talcott, C.L. A foundation for actor computation. *J. Funct. Program.* **1997**, *7*, 1–72. [CrossRef]
72. Vigilant Hawk. Available online: https://github.com/ipk0/vigilant-hawk (accessed on 27 November 2024).
73. To, Q.; Soto, J.; Markl, V. A Survey of State Management in Big Data Processing Systems. *arXiv* **2018**, arXiv:1702.01596. [CrossRef]
74. ACTORS: A Model of Concurrent Computation in Distributed Systems. Available online: https://dspace.mit.edu/handle/1721.1/6952 (accessed on 27 November 2024).
75. Fu, L. An adaptive routing algorithm for in-vehicle route guidance systems with real-time information. *Transp. Res. Part B Methodol.* **2001**, *35*, 749–765. [CrossRef]
76. Bakhshi, R.; Gavidia, D.; Fokkink, W.; van Steen, M. An analytical model of information dissemination for a gossip-based protocol. *Comput. Netw.* **2009**, *53*, 2288–2303. [CrossRef]
77. Barbosa, M.; Ferreira, B.; Marques, J.; Portela, B.; Preguiça, N. Secure Conflict-free Replicated Data Types. In Proceedings of the 22nd International Conference on Distributed Computing and Networking, ICDCN 21, Nara, Japan, 5–8 January 2021; ACM: New York, NY, USA, 2021; pp. 6–15. [CrossRef]

78. Biener, W.; Garcia Rosas, K.R. Grid reduction for energy system analysis. *Electr. Power Syst. Res.* **2020**, *185*. [CrossRef]
79. Priesmann, J.; Nolting, L.; Praktiknjo, A. Are complex energy system models more accurate? An intra-model comparison of power system optimization models. *Appl. Energy* **2019**, *255*, 113783. [CrossRef]