HARB: A Hypergraph-based Adaptive Consortium Blockchain for Decentralised Energy Trading

Samuel Karumba, *Student Member, IEEE,* Salil S. Kanhere, *Senior Member, IEEE,* Raja Jurdak, *Senior Member, IEEE,* and Subbu Sethuvenkatraman

***Abstract*—The emergence of the Internet of Things (IoT) and Distributed Energy Resources (DERs), has given rise to collaborative communities that manage their energy production and consumption load through Peer to Peer Decentralised Energy Trading (P2P DET). To address the issue of distributed trust in these communities, blockchain technology is widely considered as a promising solution due to its ability to provide records provenance and visibility. However, the current blockchain-based platforms are known to compromise on scalability and privacy in favor of trustless interactions and often do not support interoperability. In this work, we propose a Hypergraph-based Adaptive consoRtium Blockchain framework (HARB), which coordinates DERs through high-order relationships rather than P2P pairwise relationships. HARB is presented in a three-layered network architecture to address the aforementioned challenges. The bottom layer (Underlay) addresses the issue of scalability, by exploiting the rich representation ability of hypergraphs to describe complex relationships among DERs and end-users. We use the described complex relations to form scalable network clusters with intra- and inter-community energy trading relationships. The middle layer (Overlay) presents a blockchain service model to describe adaptive blockchain modules that support interoperability between the network clusters. To preserve privacy, we present a data tagging and anonymization model in the top layer (Contract), which attributes transactions to specific network clusters. To evaluate our framework, we modeled various IoT devices with different computing resource profiles to simulate a DET environment. The analyzed results have shown that our proposed framework can effectively improve the performance of blockchain-based DET systems.**

***Index Terms*—Internet of Things, hypergraphs, blockchain, clustering, notaries, distributed energy trading**

I. Introduction

THANKS to the rapid deployment of Distributed Energy Resources (DERs), (e.g solar PV, battery, electric vehicles, and control devices), along with advances in Information and Communication Technologies (e.g. Internet of Things (IoT), artificial intelligence and machine learning), passive consumers are now becoming prosumers [1]. A prosumer takes the role of a producer-and-consumer, e.g. a residential smart home equipped with a solar panel, as illustrated in Fig. 1. Prosumers proactively manage their energy production and consumption loads, and occasionally trade their excess energy production directly with their neighbors who take the role of consumers. As the number of prosumers proliferates, energy trading markets are transitioning from traditional hierarchical structures with centralized control into Distributed Energy Trading (DET) structures with no centralized control. Generally, distributed market structures are comprised of a variety of players who compete and cooperate at the same time, which necessitates the need for cooperation and distributed trust.

Recently, blockchain technology has emerged as a promising solution for coordinating distributed trust in the DET market, without relying on centralized authorities. A blockchain is a decentralized shared ledger that provides distributed trust, provenance, and immutability over a distributed P2P network, where a ledger is a repository of transactions on the distribution of assets, recorded in accounts [2]. Using blockchain, researchers in academia and the energy industry are proposing novel DET solutions [2]–[4], which can effectively implement visibility, traceability, and trust in the management of DERs. Although blockchain-based DET applications are promising, current approaches suffer from three key limitations: (i) lack of scalability [4], (ii) lack of user and data privacy [2], and (iii) lack of interoperability between co-existing blockchain systems [3].

The scalability issue manifests during transaction validation. To validate transactions, most blockchain platforms follow the order-execute architecture [5]. This means that the blockchain first orders bundled transactions to form a block and then the block is distributed to all nodes where all transactions are executed and validated sequentially, using a consensus protocol. Proof-of-Work (PoW) [6] and Proof-of-Stake [2] are the most common consensus protocols in the order-execute blockchain systems. These consensus mechanisms impose high computation and communication requirements for validating and disseminating blocks in the network so that no single node could take the network over to create a fork [6]. Additionally, DET networks are heterogeneous, meaning that they contain a variety of IoT devices with different functions such as sensors, actuators, and controllers [1]. However, it can be difficult for resource-constrained nodes with limited storage capacity and intermittent bandwidth to participate in order-execute blockchain systems.

The privacy issue manifests during transaction storage. Existing blockchain systems treat data as part of the transaction, which is replicated across the network nodes to provide high visibility and traceability [2]. In the DET context, transaction data can be used to provide valuable services such as load balancing, price prediction, optimum energy consumption scheduling, and auxiliary services (e.g, behavior modeling).

Samuel Karumba and Salil Kanhere are with the school of Computer Science and Engineering, UNSW, Asutralia. E-mail:{s.karumba, salil.kanhere}@unsw.edu.au

Raja Jurdak is with the school of Computer Science, QUT, Australia E-mail: r.jurdak@qut.edu.au

Subbu Sethuvenkatraman is with Energy Business Unit, CSIRO, Australia E-mail: subbu.sethuvenkatraman@csiro.au

However, since blockchain ledgers are immutable, replicating payment and energy usage records may leak sensitive information including identities, locations, and energy production and consumption patterns, raising privacy concerns especially in local neighborhoods. Privacy-related attacks include linking attacks where information recorded in the distributed ledgers is used to obtain privately identifiable information by linking it with other datasets [7].

Today there exists a wide spectrum of blockchain protocols and platforms for building DET networks, which has led to multiple operating systems with different methods of data and consensus management. However, these systems do not support interoperability even when their implementation is based on the same blockchain platform, which in turn has led to data and information silos [3]. In an attempt to provide interoperability, ad-hoc systems such as centralized digital notaries have been proposed [8]. A centralized digital notary is a Trusted Third Party (TTP) that verifies or authenticates blockchain transactions and participants to ensure trust between inter-operating networks [9], as illustrated in Fig. 1.

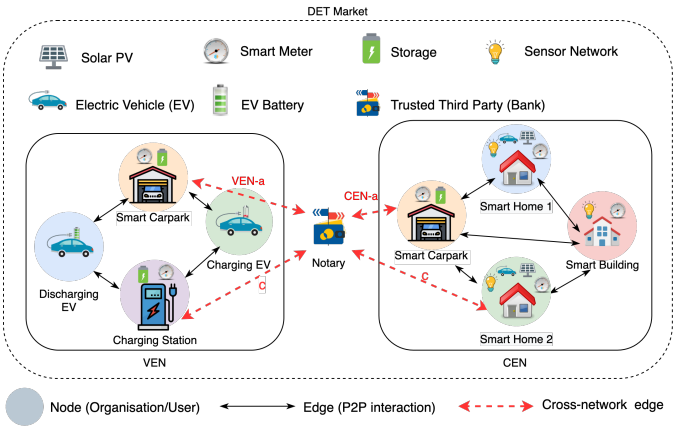


Fig. 1. An application of blockchain in Community-based Energy trading Network (CEN) and Vehicular Energy trading Network (VEN) use cases.

To understand the nature of the aforementioned blockchain challenges, let us consider two popular DET use cases where blockchain has been adopted, namely, Vehicular Energy Network (VEN) [10] and Community-based Energy trading Network (CEN) [11], illustrated in Fig. 1. In the CEN and VEN network structures, nodes (colored circles) represent participants (i.e. organizations or individual users) linked together in a P2P network model. In each network, multiple devices are collapsed into one node based on an organization or individual user ownership, to increase blockchain network scalability. Take for instance the Smart Home 1 node in Fig. 1 owned by a participant with Id: $User$-1, a device with high computing resources within the node (e.g Electric Vehicle (EV)) is chosen to represent the other resource-constrained IoT devices (i.e. Sensor nodes, Smart Meter, and Solar PV) on the blockchain network. However, since the resource-constrained IoT devices interact indirectly on the blockchain through the EV, the credibility of their transaction in the network cannot be guaranteed. Additionally, consider a blockchain participant node that interacts in both CEN and VEN (e.g. Smart Carpark), where each network implements a token system to trade energy assets. If this participant was to trade an energy asset, say **CEN-a** created in CEN within VEN then a digital notary is required to transfer the value of **CEN-a** into energy asset **VEN-a** that is tradeable within VEN. To validate such a transaction, the notary authenticates both the network participants and the energy assets through a lookup registry, which becomes a potential central point of failure. Therefore, there is a need for a privacy-preserving, scalable, and interoperable blockchain network design that provides better transaction visibility, distributed trust, and provenance in DET.

Fundamentally, P2P network structures use regular graph theory to model complex distributed systems [12]. However, regular graphs are limited by simple pairwise relationships which do not fully exploit contextual information (e.g. spatial, temporal, identity) and multi-dimensional interactions (e.g. human-thing, thing-thing), to completely describe the complex real-world systems under investigation. To address these P2P design limitations, we consider not only the devices taking part in the blockchain network but also other contextual information represented as complex multi-dimensional interaction (i.e. high-order relationships). We argue that a natural way of representing complex distributed systems, such as the DET ecosystem is to use a generalisation of graphs known as hypergraphs [13], as they can accurately model high-order relationships. In this paper, we present a Hypergraph-based Adaptive consoRtium Blockchain (HARB) framework optimised for dynamic distributed environments with heterogeneous transacting nodes. Our framework addresses the three aforementioned challenges of scalability, privacy and interoperability, as presented in the following key contributions:

i.) We propose a community discovery mechanism for network clustering derived using complex high-order relationships. We are motivated by the observation that, using the hypergraph clustering concept to cluster a distributed network into groups of nodes (sub-chains) has been found to increase the scalability of large networks [14].

ii.) We propose a blockchain network model with Adaptive Blockchain Modules (ABM), designed using the hypergraph transverse concept, described in section III-B. ABM extends the standard execute-order-validate blockchain architecture [15], to include a Distributed Notary module. Distributed notaries are highly reliable nodes selected from each cluster in the network for cross-chain interactions. This new module will provide interoperability between sub-chains to eliminate the central point of failure inherent in the centralized digital notary approach.

iii.) We propose a data Tagging and Anonymization Mecahism (dTAM) used to describe a privacy-preserving data-store model and a DET workflow engine, described in Section IV-C. dTAM is based on hypergraph coloring, Linked Ring Signature (LRSig.) [16], and Zero-knowledge Proofs (ZKP) [17] concepts.

iv.) Lastly, we model various IoT nodes with different com-

puting resource profiles to simulate a DET environment. We conduct extensive performance evaluations of HARB using Hyperledger Caliper, adjusting its key parameters while observing relevant metrics including latency, throughput, and resource utilization.

The rest of the paper is organized as follows. We review some of the related work in Section II. Section III gives a preliminary of our framework and formulate the research problems using hypergraph concepts. Section IV describes the technical details and design choices of our framework. We present our framework evaluation results and discussions in Section V. Finally, concluding remarks are given in Section VI and a highlight of the future directions.

II. State of the Art

In this section, we first review the relevant basics of the execute-order-validate blockchain architecture employed in Hyperledger Fabric. Next, we provide an overview of blockchain-based approaches for DET.

A. Execute-Order-Validate Architecture

The execute-order-validate blockchain architecture was first introduced in [18], through the private Hyperledger Fabric blockchain framework (HLF). HLF attempts to address scalability and privacy limitations presented by the order-executed blockchain architecture, using pluggable consensus and modularisation. Before HLF, virtually all blockchain systems had a hard-coded consensus protocol [18]. In a nutshell, a distributed HLF network separates nodes into the following four main modules: (i) Endorsers − execute and validate transactions, (ii) Orderers − establishes the total order of executed transactions, and (iii) Commiters − validate transactions and maintain the blockchain ledger (shared state). (iv) Channel − a shared state partition (sub-network) maintained by a subset of nodes in the network. However, although hyperledger is modular, its pluggable consensus is built over a P2P network protocol, which requires all nodes to validate and store transactions thus faces the issues of scalability, privacy, and interoperability.

B. Blockchain-based Distributed Energy Trading

The authors in [19] presented a detailed review on adopting blockchain in DET, where most of the nodes are inherently IoT devices. In general, blockchain is computationally expensive with limited scalability and is thus not suitable in the IoT context. In [20] Dorri proposed to address the scalability limitations of blockchain using a tiered Lightweight Scalable Blockchain (LSB), which is particularly optimized for the IoT environments. However, LSB employs a P2P overlay network to manage the blockchain. Pop et al. in [21], designed a scalable second-tier solution that combines blockchain with an off-chain real-time energy data storage. This approach reduces the frequency with which energy transactions are validated on the blockchain without losing the immutability benefits.

To solve privacy leakage, Gai et al. [7] presented a consortium blockchain using a noise-based privacy-preserving model to hide the trade distribution patterns that are exploited for linking attacks. This approach trades off data visibility for privacy by introducing dummy transaction accounts to mask privacy. In [22], the authors designed a blockchain-based system that leverages off-chain zero-knowledge proofs computation to preserve privacy. However, the interactions of resource-constrained IoT nodes within the blockchain are centrally managed by more resourced nodes. Additionally, Gao et al. [23] proposed a privacy-preserving payment mechanism for energy trading in VEN while anonymizing user information using LRSign.

The authors in [18] used HLF to partition (cluster) their network into channels, in an effort to address interoperability. For cross-channel trade between two channels, they used a third channel as a trusted channel to avoid double trading and support interoperability. Similarly, the authors in [3] present an architecture for interoperability between permissioned blockchain networks using relays for communications. However, the use of relays creates communication bottleneck, particularly for networks with high transaction volume and frequency such as the DET. To eliminate the use of trusted third parties in cross-chain trades, Dorri et al. in [24] proposed a secure private blockchain that ensures trading nodes commit to their obligations by introducing atomic meta-transactions.

Having discussed some of the recent work that attempts to addresses the aforementioned challenges, next we introduce the hypergraph notations and concepts adopted in our framework to address these challenges. To provide scalability, we identify naturally occurring collaboration groups using hypergraph clustering concepts to partition a distributed network into clusters (or sub-chains). For interoperability, we adopt hypergraph transversal to select distributed notaries such that at least one node from each chain is used for cross-chain transaction validation. Lastly, we propose the use of hypergraph coloring to represent distinct sub-chain data encoding format and ledger separation coupled with ZPK and LRSig to preserve data and transaction privacy.

III. Preliminaries

Here, we create an abstraction of a DET network in the form of a generalized hypergraph, as shown in Fig. 3. First, we introduce some of the basic notations and concepts of hypergraph that are relevant for modeling our framework. The bold-face uppercase letters denote a matrix.

A. Hypergraph Notations

A hypergraph is represented by $H = (V, E, w)$, where $V = \{v_1, v_2, \ldots, v_n\}$ is a set of n nodes, $E = \{e_1, e_2, \ldots, e_m\}$ is a set of m subsets (groups) of nodes referred to as hyperedges, such that $e_j \neq \phi$, and w is a weight function on the relationship that describes a hyperedge $\{e_j | \forall e \in E\}$, denoted as $w : E \rightarrow \mathbf{R}$. For example, given $E = \{l_1, l_2, \ldots, l_m\}$ is a set of m locations and r^{VVL} is a location interaction relationship (illustrated in Fig. 3), the weight $w(r^{v_i, v_j, l_k})$ is the frequency of interactions between nodes $v_i \wedge v_j \in V$ at location l_k (Charging Station). Two nodes $v_i, v_j \in V$ are adjacent if there is a hyperedge $e \in E$ that contains both v_i and v_j. A hypergraph H can be represented by an incidence matrix

$\mathbf{A} \in \mathbf{R}^{|V|\times|E|}$, of which each row in $n = |V|$ is associated with a node and each column in $m = |E|$ is associated with a hyperedge, where entries $a_{v,e} = 1$ if $v \in e$ and 0 otherwise. The degree of a node $v_i \in V$ is given by the number of hyperedges it belongs to. Similarly the degree of a hyperedge $e \in E$ is given by the cardinality $|e|$ of e. Fig. 3 show an example of a hypergraph representation of the P2P DET market in Fig. 1. Its detailed description is given in Section IV.

B. Hypergraph Concepts

1) Hypergraph Clustering: A clustered network is a hypergraph $H = (V, K)$, where $K = \{k_1, k_2, \ldots, k_z\}$ is a set of z hyperedges referred to as clusters. For a given cluster k, its sub-hypergraph is described as $H' = (V', E')$ where $V' \subseteq V$ and $E' \subseteq E$. For example, in Section IV clusters are illustrated by blocked line circles, such that CEN and VEN are independent clusters. Each cluster form an independent network with its distinct ledgers for network scalability.

2) Hypergraph Transversal: A transversal T of a hypergraph $H = (V, K)$ is a set $T \subseteq V$ with hyperedges K, such that T contains at least one node from every cluster k in K. More formally, $T \subseteq V$ is a transversal if and only if $\forall k \in K : T \cap K \neq \phi$. For example, given CEN and VEN are two intersecting clusters in K our notaries are given by selecting from a set of nodes at the intersection of CEN and VEN, given by $T = \{v_{,4}, v_{10}, v_{11}, v_{14}\}$ as illustrated in Fig. 4.

3) Hypergraph Coloring: A hypergraph coloring $\lambda(k)$ is a mapping of a cluster set K to colors C such that $\forall k \in K$ each cluster k has a distinct color. For a hypergraph $H = (V, K)$, where cluster $k_j \in K$ are hyperedges, we express the hyperedge coloring as a function $\lambda : K \longrightarrow C \quad | \quad \lambda(k_j) \neq \lambda(k_y)$ for any given pair of hyperedges. Moreover, for a node that is adjacent to more than one cluster its color $\lambda(v)$ is a color mixture of each cluster color $\lambda(k)$ given by:

$$\lambda(v) = \cup\{\lambda(e_j) \quad | \quad v \in e_j\} \tag{1}$$

We use each hyperedge coloring $\lambda(k)$ to represent different data encoding for the distinct cluster ledgers. Additionally, we use Equation (1) to tag each transaction indicating its visibility scope during execution, validation, and storage for privacy, as described in Section IV-C.

IV. Hypergraph-based Adaptive Consortium Blockchain

In this section we introduce the HARB framework as a three-layered network architecture for management of DERs in the DET market. In our framework we proposes the use of high-order relationships rather than simple pairwise relationships used in traditional P2P networks. Fig. 2 presents a conceptual view of the framework which comprises of three layers: underlay, overlay, and contract. Each layer describes various complex interactions between nodes at different levels of granularity based on the hypergraph concepts described in Section III-B to increase scalability, promote interoperability and preserve privacy in the DET ecosystem.

The underlay layer (physical network) is designed using complex high-order relationships described next. First, we articulate how complex high-order relations are derived by considering time, location, identity and context properties of interacting nodes. Second, we use high-order relationships for community discovery by interrogating the frequency of interactions among nodes. Dense interactions indicate the presence of intra-community relationships and sparse interactions indicate the presence of inter-community relationships. Lastly,

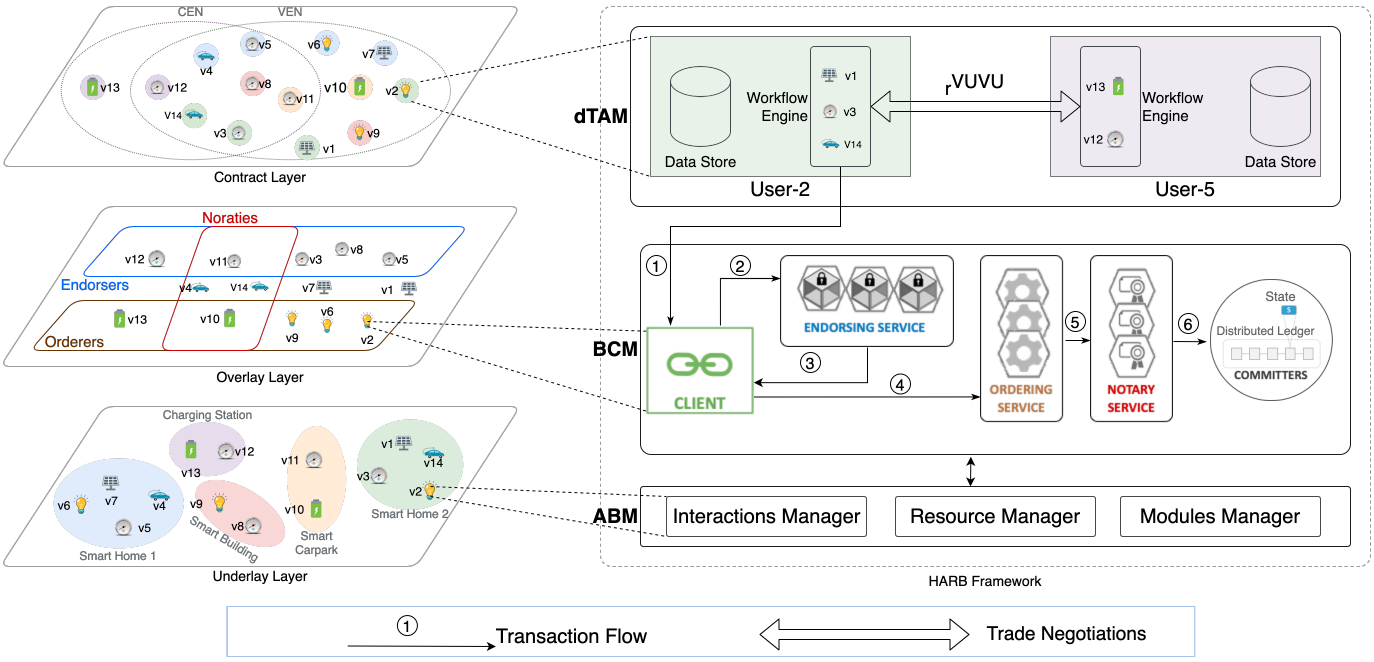


Fig. 2. High level overview of HARB Framework. On the left we have the three network layers and on the right we have an expanded view of the framework components.

we select one of the community-oriented relationships to cluster the distributed network into scalable sub-networks.

The overlay layer forms the Blockchain Network Model, which is described by an Adaptive Blockchain-modules Manager (ABM) and a Blockchain Client Manager (BCM), illustrated in Fig. 4. Nodes are adaptively grouped together by the ABM using the high-order interaction manager, nodes resource manager and the modules manager to form modules. Each module implements a blockchain service i.e., endorsing, ordering, validating and committing. The module manager in ABM assigns the role for each node based on its computing capability, reliability and reputation as shown in Table I. A node can be assigned multiple roles.

Finally, the contract layer represents the applications network. Applications (smart contracts) are used to define user requirements which are delivered using the blockchain services in the overlay layer. These required services include distributed trust, privacy, and value transfer. The contract layer uses the concepts of trade contracts and workflows implemented in a workflow engine, to fulfil DET agreements.

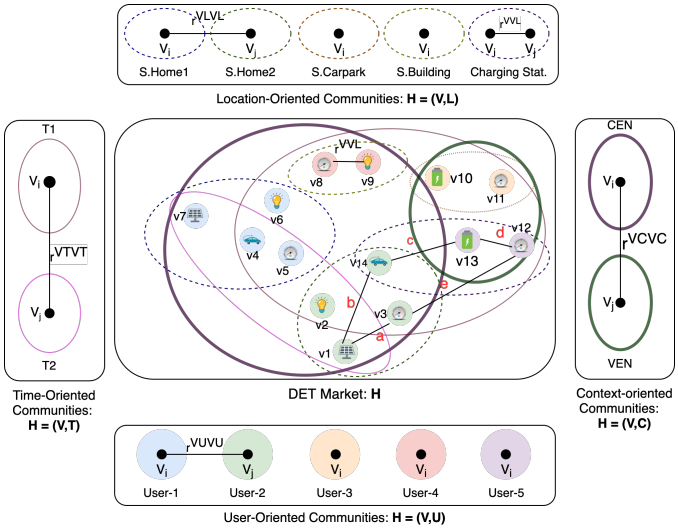


Fig. 3. An illustration of a hypergraph-based high-order relationship which fully exploits contextual information such as time, space (location) and identity (user/organisation) in a multi-multidimensional interaction (e.g. thing-thing-time, thing-thing-location, thing-thing-context).

A. Complex High-Order Relationships

Our goal here is to fully capture the contextual interaction information between interacting nodes. Extending the hypergraph models proposed in [13], [25], we describe complex interaction dimensions including: time, place, context and identity, which completely describe the relationships between nodes in a real-world DET ecosystem. Let $T = \{t_1, t_2, \ldots, t_\Delta\}$ be a set of Δ time periods when nodes interacts; $L = \{l_1, l_2, \ldots, l_d\}$ be a collection of d distributed geographical locations where nodes interacts; $C = \{c_1, c_2, \ldots, c_m\}$ be a set of m application contexts (e.g. a virtual power plant, microgrid, VEN or CEN); and Identity $U = \{u_1, u_2, \ldots, u_x\}$ be a set of x end-users or organisations who own and interact with the nodes. Therefore, given the multi-dimensions set $D = \{L, T, C, U\}$, the interactions of nodes in the DET space $H = (V, E)$ can be represent using hypergraphs as illustrated in Fig. 3, where E is varying based on the dimensions (i.e. $H = (V, L)$, $H = (V, T)$, $H = (V, C)$, and $H = (V, U)$).

Assuming that all interaction patterns P of any given pair of nodes $v_i \wedge v_j \in V$ in the DET ecosystem H are known, we derive high-order relationships R by finding all possible combinations of $y = |D|$ taken r at a time, given by:

$$ {}_yC_r = \binom{|D|}{r} \quad | \quad \forall r \in R : R \subseteq P \tag{2} $$

For example, if $r = 2$ then ${}_5C_2 = \{VV\}$ is a simple P2P relationship; if $r = 3$ then ${}_5C_3 = \{VVL, VVT, \ldots\}$ is an intra-community relationship; if $r = 4$ then ${}_5C_4 = \{VTVT, VLVL, \ldots\}$ is an inter-community relationship; and finally if $r = 5$ then ${}_5C_5 = \{VLTCU\}$ is status monitoring relationship for monitoring the state of nodes in the network (e.g. computing resources level). Note that, we eliminate any combination that does not match any known interaction patterns P in the DET ecosystem. We provide a detailed description for these relationships as follows:

1) Status relationships (R_{status}): This is a set of relationships $R_{status} = \{r^{VLTCU}\}$ that describe the attributes and state of a node in the network. A weight $w(r)|\forall r \in R_{status}$ is given to describe the frequency at which the status of a node is monitored, highlighted as follows:

- Node-Time-Location-Context-User relation (r^{VLTCU}): Represent the presence of interaction of a node v_i in location l_k at time t_δ owned by user u_α participating in energy trading context x. For example, we can use this relationship to monitor activities from IoT nodes such as sensors which actively monitor their environment.

2) Intra-community relationships (R_{intra}): This is a set of relationships $R_{intra} = \{r^{VVL}, r^{VVT}, r^{VVC}, r^{VVU}\}$ that describes when, where or whom a given pair of nodes have interacted with. A weight $w(r)|\forall r \in R_{intra}$ is given to indicate the frequency of each interaction. These relationships are highlighted as follows:

- Node-Node-Location relation (r^{VVL}): Is used to indicate the relationship where nodes $v_i \wedge v_j \in V$ interact at location l_k. Each new frequent location interaction forms an entry in the locations-oriented communities set L.
- Node-Node-Time relationship (r^{VVT}): Is used to indicate the case where nodes $v_i \wedge v_j \in V$ interact at a certain time t_δ. Frequent interactions for a given timestamp t_δ create an entry in the time-oriented communities set T.
- Node-Node-Context relation (r^{VVC}): Is used to indicate the relationship where nodes $v_i \wedge v_j \in V$ interact within the same context c_x (e.g CEN). Each unique context interaction creates a new community entry in the context-oriented communities set C.
- Node-Node-User (r^{VVU}): Represents a relationship where two interacting nodes $v_i \wedge v_j \in V$ belong to the same user or organisation u_α.

3) Inter-community relationships (R_{inter}: High frequency of interactions given by a certain intra-community relationships indicates the presence of a community. Therefore, inter-community relations refer to the set of relationships formed by interactions of nodes between identified communities. The weight $w(r)$ is the number of distinct communities created by each relationship. For example, the weight $w(r^{VLVL})$ in fig. 3 indicates that this relationship creates five location-oriented communities (i.e. $w(r^{VLVL}) = 5$). We summarise the derived community-oriented relationships as follows.

- Location-oriented relationship (r^{VLVL}): Indicates a relationship where a node $v_i \in V$ located in location $l_k \in L$ interacts with another node $v_j \in V$ located in location $l_w \in L$. This relationship can be used to cluster nodes into geographical regions.
- Time-oriented relationship r^{VVTT}: Indicates the relationship where a node $v_i \in V$ interacts with node $v_a \in V$ at time t_δ and then interacts with node $v_b \in V$ at time t_σ. Time-oriented clustering can be used for DERs scheduling in energy demand supply balancing. For example, during the time period $T2$ (daytime) energy is supplied from the Solar PVs $v1, v7$ and switched to backup energy storage systems $v10, v13$ during $T1$ (nightime) as illustrated in Fig. 3.
- Context-oriented relation (r^{VCVC}): Indicates a relationship where a node v_i in context c_x and another node v_j in context c_y interacts together to exchange value between contexts. Clusters formed by this relationship captures the network heterogeneity and can be used for interoperability.
- User-oriented relation r^{VUVU}: Indicates a relationship where a node $v_i \in V$ belonging to users $u_a \in U$ interacts with another node $v_j \in V$ belonging to users $u_b \in U$. Similarly, this relationship could be used to form clusters where end-users (or organisations) interacts through their nodes.

The community relationship with the highest modularity value is used to form the network clusters. A modularity function in [14] describes a good network clustering as the one with sparse inter-cluster interactions and dense intra-cluster interactions, given by:

$$Q = \frac{w(R_{inter})}{w(R_{intra})} > 0 \tag{3}$$

Where Q is the modularity value in the the range $0 < Q < 1$. We select the community relationship that has the best modularity measure of $Q \to 0$ (i.e. Q approaches zero), to form the clustered network $H = (V, K)$ (described in Section III-B), where K is a set of sub-networks. For example, in Fig. 3 the context-oriented relationship r^{VCVC} has been selected to create the network clusters indicated by the solid circled lines. In the next subsection the overlay network is discussed, which uses the selected clustering relationship together with the profile description of nodes to predict node behaviour and adaptively assign blockchain roles to the nodes.

B. Blockchain Network Model

Here we describe two core concepts that form our overlay network namely the Adaptive Blockchain-modules Manager (ABM) and the Blockchain Client Manager (BCM). A node $v_i \in V$ is the least divisible unit in the overlay network $H = (V, E)$, where E is a set of network clusters. We use ABM to describe a set of blockchain modules $M = \{endorsers, orderers, notaries, committers\}$ required to setup a blockchain network for each cluster $e_x \in E$. For value transfer between nodes $v_i \wedge v_j \in V$ in sub-chains and cross-chain, BCM describes a blockchain $clients$ module that exposes the blockchain services provided by the blockchain modules. Details of these two concepts are provided as follows.

1) Adaptive Blockchain-modules Manager (ABM): Since nodes in the DET ecosystem are largely heterogeneous IoT devices, the ABM adopts the standard Semantic Sensor Network Ontology[1] (SSNO) to annotate each node. SSNO is a service description framework that provides rich expressive description and well-defined semantics for IoT devices. For each node $v \in V$, v_i is expressed as a tuple $v_i = \langle V_{id}, V_{roles}, \langle V_{prop} \rangle \rangle$, where V_{id} is a unique identifier of v_i, $[V_{roles}]$ is a network role assigned to v_i (e.g. endorser, orderer etc.), and $\langle V_{prop} \rangle$ is a set of properties for v_i with each property expressed as a tuple of the form $V_{prop} = \langle Type, Prot, Res, \dots \rangle$, where $Type, Prot, Res$ refer to node type e.g (a sensor, computer or actuator), protocols (e.g HLF, Ethereum, HTTP etc.), and available resources respectively. To adaptively determine which network role can be assigned to node v_i, ABM continuously monitors its resource and reliability indices based on the metrics shown in Table I.

TABLE I
MINIMAL REQUIREMENTS FOR EACH BLOCKCHAIN ROLE.

Roles	CPU	Storage	Bandwidth	Reliability
Endorser	high	medium	medium	high
Orderer	medium	low	high	medium
Notary	high	hig	high	high
Committer	low	high	high	high

The resource indices $\lambda(v_i)$ and the reliability index $\Upsilon(v_i)$ are computed as follows.

- Resource Index: $\lambda(v_i)$ refers to the device capabilities, indicated by the computational elements such as CPU, memory, and bandwidth hosted on a particular node. ABM instruments a computing resource monitoring tool using the status relationship (r^{VLTCU}) to intercept each resource and harvest its current value. Considering the current tasks running on node v_i the resource index $\lambda(v_i)$ is given by:

$$\begin{aligned} \lambda(v_i) \quad &= \sum_{j=0}^{n} (\mu_j - S_j) \quad | \quad \forall \mu \in \Re, \\ &\mu \in \{CPU, memory, bandwidth\} \end{aligned} \tag{4}$$

[1] https://www.w3.org/TR/vocab-ssn/

Where $\Re$ is a set of the measured resource values for CPU,memory and bandwidth, n is the total number of measured resource values, μ_j is a measured resource value at index j from the resource monitoring tool, and S_j is the expected amount that the current running services will consume.

- Reliability index: Here, we consider reliability as a probabilistic rating in the range $[0, 1]$ and focus on how historical performance of a node affects its rating. Each time a node v_i is assigned a task, ABM evaluates its performance and assigns a probabilistic score t_i^k, which is added to its historical record of transactions performance scores $T(i) = \{t_i^1, t_i^2, \ldots, t_i^h\}$, where t_i^k is the level of task completion within $0 \leq t_i^k \leq 1$ and h is the n^{th} score. The score t_i^k is influenced by three factors h, k, d, s. (h) is the number of times the node has been selected for the same task. k is the number of time the node has completed the assigned task. (d) is the number of time the node has been selected for a task but didn't complete it due to resource depletion (e.g. low battery, insufficient memory, intermittent bandwidth etc.). And s is the completion status at the end of current task. To calculate a nodes reliability rating, we adopt the exponential averaging concept in [26], with a recency factor θ on its historical score as follows:

$$\Upsilon(v_i) = \begin{cases} \sum_{k=1}^{h} \theta^{(h-k)}.t_i^k, & h \neq 0 \\ 0, & h = 0 \end{cases} \tag{5}$$

where $\Upsilon(v_i)$ is the reliability of node v_i from the perspective of ABM, $\theta^{(h-k)}$ is the recency factor such that the weight $(0 < \theta < 1)$ is given to the most recent task performance scores.

In our framework, a module is an aggregate of nodes task interfaces. Nodes are aggregated together in a module, such that ABM provides the following blockchain services: execution, ordering, validation and commit (storage) through consensus. A role corresponds to a blockchain module as illustrated in Fig. 4.

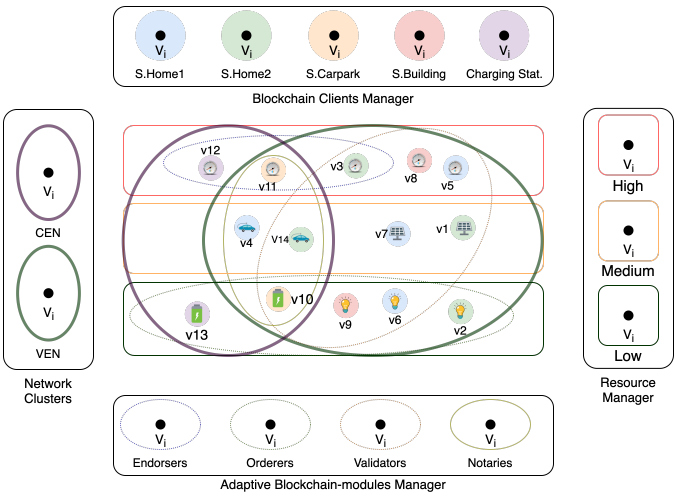


Fig. 4. BSM based Blockchain Modules

Recall that a given overlay network $H = (V, K)$ is created by either one of the community-oriented relationships described in Section IV-A, where K is a set of sub-networks. For our example in Fig. 4 a set of clusters $K = CEN, VEN$ are marked with solid colored circles. Suppose nodes in CEN and VEN were to exchange value, the inter-cluster interaction would be processed only by nodes within the two sub-networks. In each sub-network $k \in K$ only nodes $v_i \in V$ form modules for k. Each module $m \in M$ is determined by:

$$m = \sum_{i=0}^{|k|} (\eta \cdot \Upsilon(v_i) * \vartheta \cdot \lambda(v_i)) \quad | \quad \forall k \in K$$
$$m \in \{Endorser, Orderer, Committer\} \tag{6}$$

Where $M = \{Endorser, Orderer, Committer\}$ is a set of modules, η is a system level parameter for setting the required level of nodes reliability value $\Upsilon(v_i)$ in each module, and ϑ is a system level parameter for setting the required resource level for each computing resource of a node $\lambda(v_i)$.

However, not all nodes in the network can validate this interaction. ABM introduces an additional module referred to as the **notary**. We use the hypergraph transversal concept discussed in Section III-B to assign the notary role to a subset of nodes at the intersection of all clusters $k \in K$. For example, as show in Fig. 4 notaries are nodes with the highest degree in the main-network given by:

$$Tr = \bigcap_{i=0}^{n} d(i) \tag{7}$$

Where $d(v_i)$ is the degree of a node $v_i \in V$. Additionally, notaries are required to have high connectivity, processing power, storage and reliability as they validate all cross-chain interactions.

2) Blockchain Clients Manager (BCM): The overlay network layer presents the BCM, which interfaces between the blockchain service requests from the contract layer (discussed in Section IV-C) and the blockchain service responses from the blockchain modules. Any end-user node in the network can be assigned the blockchain client role as illustrated in Fig. 4. Specifically, a blockchain client describes how the blockchain service is processed for sub-chain and cross-chain network interactions as illustrated by the sequence diagram in Fig. 5.

A blockchain service is represented as a tuple of the form $S =< S_{id}, S_{in}, S_{out} >$, where S_{id}, S_{in}, S_{out} and S_{op} correspond to service identifier, input, output and operation, respectively. An end-user's blockchain client receives a transaction sequence map $T = \{Txn_1, Txn_2, \ldots, Txn_n\}$ (illustrated in Fig. 6) from dTAM in the contract layer. For each transaction $Txn_i \in T$, BCM uses the transaction tags (discussed in Section IV-C) to determine which nodes in the blockchain modules will process Txn_i as per the following steps:

1: The client initialises the blockchain service request by preparing a service payload S_{in} as a tuple of the form $S_{in} = \langle T_{req}, T_{op} \rangle$, where T_{req} is the blockchain transaction request, and $T_{op} = endorse()$.
2: Endorsers execute the T_{req} and return a set of responses $[T_{resp}]$ from each endorser and an endorsement status ε achieved through consensus.

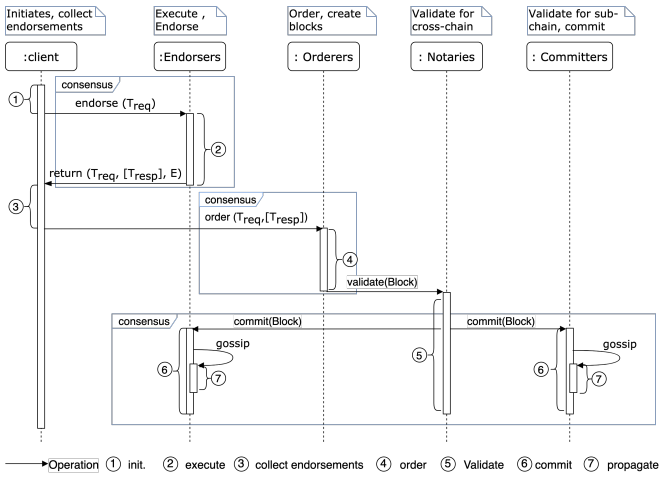


Fig. 5. A sequence diagram to describe how sub-chain and cross-chain transactions are processed.

3: If ε has status $valid$ the client prepares an ordering, validate and commit services request $S_{in} = \langle T_{req}, [T_{resp}], E, T_{op} \rangle$ and send it to the orderer module, where $T_{op} = [order(), validate(), commit()]$. Otherwise the client terminates the process with an error message $\perp$.
4: The orderer module determines the order of transactions through consensus to generate a block B. Once a block is generated, the orderer prepares a validation service request $S_{in} = \langle B, T_{op} \rangle$ which is sent to the notaries module for validation (i.e. $T_{op} = validate()$).
5: For each transaction $T_{req} \in B$ in the block, the notaries module only validates the cross-chain transactions determined by referencing to the transactions "Tag". If a cross-chain validation fail the client terminates the process with an error message $\perp$ from the notaries, otherwise the notaries forward the block to committer for sub-chain transaction validation.
6: Committers receive a block validation service request $S_{in} = \langle B, T_{op} \rangle$, where $T_{op} = validate()$ is for all sub-chain transactions in the block.
7: Since all nodes in the network are committers, only a few nodes are selected to validate and then propagate the block to all non-validating nodes using the gossip protocol. This is to increase the network throughput.

The client node repeats step 1 to 7 for all transaction $T_{xn} \in T$ and then prepare a blockchain service response S_{out} that is sent back to the contract layer. S_{out} contains either a success message or an error message $\perp$ if the process failed at any step.

In this subsection, we have provided a detailed description of the overlay network components used to implement the execute-order-validate architecture with cross-chain validators dubbed digital notaries. Each node with a Endorser, Notary, or Committer role must run a workflow smart contract to perform transaction execution and validation operations in the network. The workflow is described in the following subsection.

C. Data Tagging and Anonymization Model(dTAM)

In this subsection, we describe dTAM which forms the contract network denoted by $H = (V, U)$, illustrated as the contract layer in Fig. 2. The set V is a collection of all interacting nodes, and U is a subset of nodes grouped based on user-oriented relationships (i.e. ownership). For example, consider a DET contract between User-2 (owner of S.Home2) and User-5 (owner of the charging station). Their interactions can be represent through complex relationships a, b, c, d and e, illustrated in Fig. 3, capturing more information such as interaction time, location, context and the involved DERs for more visibility. dTAM encompasses a contract, a data-store and a workflow engine that forms the contract layer. A contract is processed through the use of smart contracts which manage the data-store and the workflow engine components in the contract network. Details of these three concepts are presented as follows.

1) Contract: A contract is described in dTAM as a sequence of multiple business rules used to record an agreement, a deal or a pact between trading parties (could be two or more). Each rule can be denoted as Complex Relationship Activity (CRA) of the form r^{VVU} or r^{VUVU} (discussed in Section IV-A) represented as:

$$interacts : A \longrightarrow B$$

Read as A interacts with B, where A is the initiator node and B is the follower node.

Every contract describes who the contracting parties are and what they are agreeing to. For instance, consider our energy exchange example above, where User-2 has agreed to exchange 20Kwh of energy with User-5. Our energy exchange contract will have five nodes $v1, v3, v14, v12$ and $v13$, and six business rules. Table II illustrates the respective CRAs for the complex relationships a, b, c, d and e in our DET contract. All rules are listed sequentially in the contract.

TABLE II
COMPLEX RELATIONSHIP ACTIVITIES

CRA	Initiator	Follower	Tag
$request\_20Kwh : v12 \rightarrow v3$	SM(v12)	SM(v3)	r^{VCVC}
$monitor\_20Kwh : v3 \rightarrow v1$	SM(v3)	Solar Pv (v1)	r^{VVC}
$charge\_20Kwh : v1 \rightarrow v14$	Solar PV (v1)	EV (v14)	v^{VVC}
$discharge\_20Kwh : v14 \rightarrow v13$	EV (14)	Storage (v13)	r^{VCVC}
$monitor\_20Kwh : v13 \rightarrow v12$	Storage (13)	SM(v12)	r^{VVC}
$pay\_500AUD : v12 \rightarrow v3$	SM(v12)	SM(v3)	r^{VCVC}

2) Data-store smart contract: dTAM describes the structure of a blockchain transaction as a tuple: $tx = \langle id, data, sign \rangle$, where id is the transaction identifier, $data$ is the record of a single unidirectional Complex Relationship Activity (CRA), and $sign$ is the transaction authentication signature generated by hashing the $data$. To enhance privacy and reduce the memory footprint of the transactions, we use Zero-Knowledge-Proofs (ZKP) [17] coupled with Linkable Ring Signatures (LRSig) [16] for user anonymisation. A LRSig is a 1-out-of-n signature scheme composed of three algorithms: keys generation $G(x)$, transaction signing $S(x)$, and transaction validation $V(x)$. Consider the transaction $data$ based on the

first CRA in Table II, where $v12$ initiates the requests of 20 Kwh of energy from $v3$, denoted by:

$$data = \{request\_20Kwh : v12 \rightarrow v3\}$$

To keep the amount of energy to be transferred private, dTAM generates a ZKP secret $\sigma = 20$ kwh, such that σ lies in the range $[0, u^x]$, where u^x discrete log is hard. Transactions validators discussed in Section IV-B can use ZKP to prove the knowledge of σ without revealing any private information such as the energy units transferred or the amount paid. Further, to anonymise transaction participants in the $data$, dTAM first invokes:

$$(sk, pk) \leftarrow G(1^k) \tag{8}$$

which takes a security parameter k for each node $v_i \in V$ as input and outputs their private sk and public pk keys. Second, the initiator node $v12$ signs the $data$ by invoking $S(x)$:

$$\rho \leftarrow S\left(1^k, sk, L, data\right) \tag{9}$$

which takes the initiators $v12$'s secret parameter k, private key sk, a list L of n public keys generated by G, and transaction $data$, to produce an anonymous data signature ρ. Any validator node v_i whose public key pk is in L can validate the anonymous signature ρ without the transaction participants identities being revealed. The validator nodes invoke $V(x)$:

$$1/0 \leftarrow V\left(1^k, L, \rho\right) \tag{10}$$

which takes as input the validator's secret parameter k, a list L of n public keys, signature ρ, and outputs 1 or 0 if valid or invalid respectively. Therefore, the structure of the transaction is modified to $tx = \langle id, \hat{\rho} \rangle$, where σ is the privacy preserving and anatomised transaction transaction data. Notice that, we do not need to add $data$ as part of the transaction, thus reducing the memory footprint on blockchain. The raw transaction data is stored in the off-ledger data-store.

3) Workflow engine smart contract: A workflow is described as a sequence of atomic meta-transactions in [24], such that all transactions in the workflow must be executed in the order they appear and each transaction must be valid. We extend this workflow to generate an executable representation of a contract encoded using notations such as JSON or XML, illustrated in Fig. 6. If one of the transaction in the execution sequence is invalid, previously executed transactions are rolled back. The workflow engine smart contract implements the process of generating a workflow from a contract in an algorithm, illustrated by Algorithm 1.

The steps of the workflow are described as follows:

1) Using the contract $Contr.$ as input and a set of nodes V as input, first we create a list L of n public keys $pk \quad | \quad \forall v_i \in L$.
2) Analyse all business rules in the contract into a list A of CRAs.
3) For each contract activities in A, create a transaction tx, then sign and anonymise as described by the data-store smart contract.
4) Using Equation (1) we tag each transaction tx based on the clustering relationship (e.g. context-oriented relationship). For example, given transaction data $request\_20Kwh : v12 \rightarrow v3$, its transaction tag will be given by $tag \leftarrow \lambda(v_{12}) \cap \lambda(v_3)$. The transaction tags is a cross-chain if $\lambda(v_{12}) \neq \lambda(v_3)$ relationship and a sub-chain interaction otherwise.
5) Once the transaction has been tagged we can drop $data$ from the blockchain transaction.
6) Finally, we collect all created transactions in an ordered set Tx. The algorithm output is a transaction workflow map $T = \{Tx_1, Tx_2, \ldots, Tx_n\}$, illustrated in Fig. 6.

Algorithm 1 Workflow Transaction Tagging Algorithm

Input: $Contr., V$
Output: $T = \{Txn_1, Txn_2, \ldots, Txn_n\}$

for all $v_i \in V$ **do**
  $(sk, pk) \leftarrow G(1^k)$
  $L \leftarrow append(pk)$
end for
Create a list A of all contract activities
for all $a \in A$ **do**
  $tx < id, data > \leftarrow createTransaction(a)$
  $tx < id, data, secret > \leftarrow generateZKP(tx)$
  $tx < id, data, \rho > \leftarrow signAndAnonymise(tx)$
  $tx < id, data, \rho, tag) \leftarrow tagTrasaction(tx)$
  $tx < id, \rho > \leftarrow dropData(tx)$
  $Tx \leftarrow append(tx)$
end for
return Tx

Fig. 6. Workflow Transaction Dependency Map

After generating the workflow transaction map as indicated in the process above, the contract layer invokes the overlay layer and submits the workflow transactions map as a blockchain service input S_{in} request. The blockchain client manager in turn prepares each transaction as an operation request OP_{in} sent to the blockchain modules. The workflow engine receives S_{out} with T_{resp} after successful completion

or an error $\perp$ message if processing failed.

V. Evaluation And Discussion

In this section, we present a Proof-of-Concept (PoC) implementation, quantitative performance evaluation, and a qualitative discussion of the HARB framework. To facilitate the examination of the proposed HARB elements as specified in Section IV, we adopted an experimental design for our PoC. This is because the realization of the proposed framework requires the blending of the components defined herein with existing blockchain technology such as Hyperledger Fabric or Ethereum.

A. Implementation

We implemented a PoC for our framework with the clustering functionality based on Hyperledger Fabric v2.0 due to its extendable modular design. Particularly, we extended its network modules to include a notary service component in addition to endorsers, committers, and orderer services. Additionally, we generalized the HLF's original P2P protocol in the blockchain service module using hypergraphs to capture high-order relations between nodes. At the application layer, we developed a workflow manager and a data-store manager smart contracts written in GoLang[2], one of the supported object-oriented programming languages in Fabric.

We defined a private HARB framework network to simulate a real-world DET market with 14 nodes as shown in Figure 3. Each distinct node was modeled to indicate different embedded resource capabilities, location, and operating times as shown in Table III. For resource capabilities, we assume that all Smart Meter nodes $\{v_3, v_5, v_8, v_{11}, v_{12}\}$, EV nodes $\{v_4, v_{14}$, Energy Batteries nodes $\{v_{10}, v_{13}\}$, Solar PVs nodes $\{v_1, v_7\}$,and Sensor Networks nodes $\{v_2, v_6, v_9\}$ are based on the specifications of Raspberry Pi 4, Beaglebone Black, Intel Edison, Intel Galileo, and Arduino Yum embedded-computer-boards, respectively[3]. We used high-order nodes relations and internal relations described in section IV-A to cluster nodes into subgroups. We used HLF channels to represent each cluster. In a Docker Compose[4] configuration files, we modeled the blockchain network components as described by ABM. We executed the docker-compose configuration file on a 3.70GHz Intel(R) Xeon(R) 12, Linux Server (Ubuntu) with 62GB RAM Memory, to create the blockchain network managed by BCM. Finally, we deployed the workflow smart contract and used Hyperledger Caliper[5] v3.0 to evaluate our blockchain network. Caliper is a benchmark tool for evaluating blockchains, allowing users to measure the performance of a blockchain model using metrics such as latency, throughput, CPU time, and memory usage.

TABLE III
Distinct Nodes Profile Information.

Category	CPU (MHz)	Storage (GB)	Connectivity	Mobile	Time
Smart Meters	700	2	Gigabits	False	Day, Night
Electric Vehicles	1000	4	Gigabits	True	Day, Night
Energy Batteries	500	2	Megabits	False	Night
Solar PVs	400	1	Megabits	False	Day
Sensor Networks	16	1	Megabits	False	Day, Night

[2]https://golang.org/
[3]https://learn.adafruit.com/embedded-linux-board-comparison
[4]https://docs.docker.com/compose/
[5]https://hyperledger.github.io/caliper/

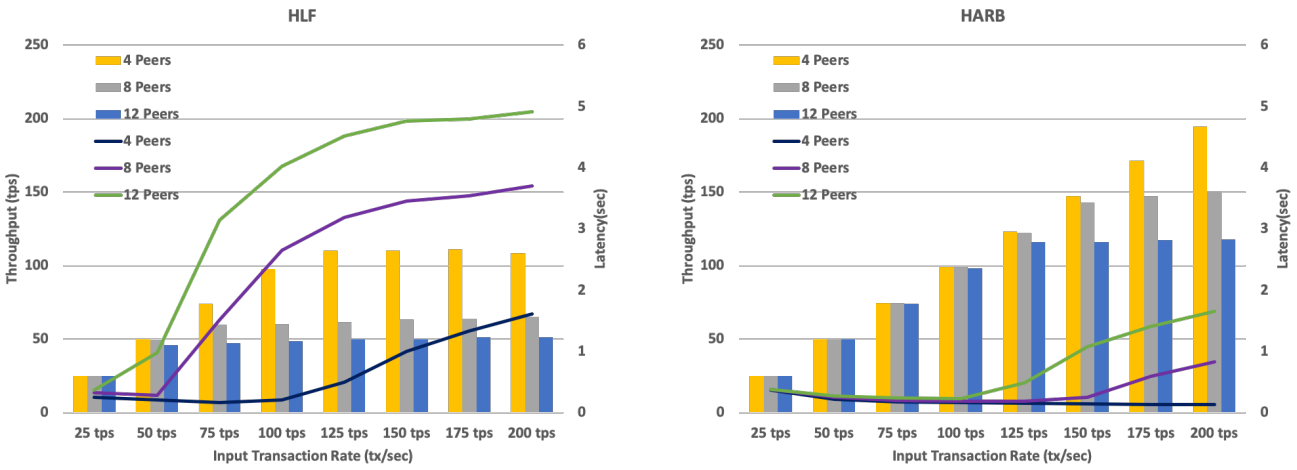
B. Performance Evaluation

Performance for executing transactions in the contract workflow map shown in Figure 6 was tested. First, to benchmark our framework, we considered HLF with its original P2P protocol as the baseline and compared its performance against the HARB framework. Reports produced by Caliper include throughput − measured in transactions per second (tps), transaction latency, and resource utilization. For both systems, we setup 14 Caliper clients, one for each participant node. To perform a Maximum Load Test, we generated test loads from the 14 clients with a send rate starting from 25 transactions per second (tx/sec) to 200 tx/sec incremented at 25 tx/sec. To determine the scalability of the framework, we repeated the experiment three times where in the first round we started with 4 nodes and 4 clients and incrementally added 4 nodes and 4 clients in each round. At the end of each round, an average of the throughput and latency is calculated.

Figure 7 shows a comparison of latency and throughput between the baseline and HARB networks for the three configuration rounds. The results show that for the given range of transactions, HARB's throughput scales linearly and is on average 2.3 times greater than the baseline before reaching the saturation point. We also noted that increasing the number of nodes in the baseline significantly degrades throughput compared to our framework. This is because the transaction ordering and consensus load increase with the increase in the number of nodes. Transaction latency experiences a steep increase after throughput saturation points (49tps for 12 nodes, 52tps for 8 nodes, and 102tps for 4 nodes) in the baseline. This is due to the congestion caused by the rapid growth in the number of transactions waiting in the execution and validation queues, which affects the commit latency. Similarly in our framework, transaction latency increases after throughput saturation points (104tps for 12 nodes and 150tps for 8 nodes). However, with 4 nodes, the throughput did not saturate within the 25 to 200 tx/sec test range. We also note that in our framework, the increase in latency is more gradual compared to the baseline. This is because our framework balances the execution and validation load by having only specific nodes within-participant clusters perform the tasks rather than all nodes in the network.

C. Discussions

In this section, we discuss how privacy is preserved from the perspective of a linking attack to prevent private information leakage, where an adversary can use brute-force algorithms to unmask encrypted data in the ledger. Further, we highlight some benefits of using distributed notaries as opposed to centralized digital notaries for interoperability between blockchain networks.



(a) Baseline

(b) HARB

Fig. 7. Performance Evaluation

1) Linking attack: In our framework, adversaries are nodes V_{adv} who can perform linking attacks. We enhance the privacy against linking attacks in two levels: (i) Reducing the amount of data included in a transaction. Recall from Section IV-C that dTAM employs anonymized ZKP proofs as transaction data, which introduces transaction data-level privacy. (ii) Reducing the number of nodes that execute or validate a transaction. dTAM describes a transaction tagging algorithm to mark a transaction visibility boundary. For instance, as shown in Table II each transaction is tagged based on the scope of the interacting node (e.g nodes $v12$ and $v3$ are from different location-oriented clusters owned by different users but within the same Time-Oriented Cluster). Therefore, even if an adversary node V_{adv} gets hold of one transaction, it is hard for it to obtain all the other transactions in the workflow map. Additionally, even with access to all transactions V_{adv}, transaction data is protected by the anonymized ZKP proofs as noted above.

2) Interoperability: Our framework presented decentralized notarization to address the issue of the trusted centralized notary, in that no single entity is trusted to securely store the data in question. In our approach, notaries N are part of the network such $\forall v_i \in N : v_i \in V$. We select notaries from a set of nodes with the highest degree of incidence as they already contain multiple ledgers within. Additionally, cross-chain validation in distributed notaries is through a consensus, which is guaranteed to be reliable based on the high reliability of the selected nodes.

Having designed, implemented, and discussed the proposed HARB framework in a PoC, next we summarise our work and discuss future directions.

VI. Conclusion

In this paper, we proposed an adaptive consortium blockchain framework based on hypergraph concepts to address the issues of scalability, interoperability, and privacy in blockchain-based DET networks. The framework groups nodes into communities by observing their inter-community and intra-community pattern of interactions based on locations, timestamps, contexts, and identities. For network scalability, we select the interaction pattern that has dense intra-community relationships and sparse inter-community relationships to cluster the DET network into sub-networks. We implemented our framework in a PoC and compared our result with the Hyperledger Fabric framework as our baseline, which is similarly based on the modular execute-order-validate blockchain architecture to evaluate scalability. Although clustering a network achieves scalability, multiple networks introduce the issue of interoperability for value transfer. Our model addressed this challenge using an adaptive blockchain model which proposes the concept of a distributed notary service for cross-network transaction validation. Additionally, our framework preserves privacy in intra- and inter-cluster interaction through the data tagging and anonymization model. Lastly, we performed qualitative interoperability and privacy analysis to justify our design choices.

Our work here presents a proof-of-concept of HARB in a simulated environment. An interesting direction for future work is to implement the HARB framework in a real-world DET ecosystem to further validate its performance in an uncontrolled environment.

References

[1] J. Abdella and K. Shuaib, "Peer to peer distributed energy trading in smart grids: A survey," *Energies*, vol. 11, no. 6, p. 1560, 2018.

[2] A. Dorri, A. Hill, S. Kanhere, R. Jurdak, F. Luo, and Z. Y. Dong, "Peer-to-peer energytrade: A distributed private energy trading platform," *ICBC 2019 - IEEE International Conference on Blockchain and Cryptocurrency*, pp. 61–64, 2019.

[3] E. Abebe, D. Behl, C. Govindarajan, Y. Hu, D. Karunamoorthy, P. Novotny, V. Pandit, V. Ramakrishna, and C. Vecchiola, "Enabling Enterprise Blockchain Interoperability with Trusted Data Transfer (industry track)," *Middleware Industry 2019 - Proceedings of the 2019 20th International Middleware Conference Industrial Track, Part of Middleware 2019*, pp. 29–35, 2019. [Online]. Available: http://arxiv.org/abs/1911.01064

[4] F. Blom and H. Farahmand, "On the scalability of blockchain-supported local energy markets," *2018 International Conference on Smart Energy Systems and Technologies, SEST 2018 - Proceedings*, 2018.

[5] E. Androulaki, Y. Manevich, S. Muralidharan, C. Murthy, B. Nguyen, M. Sethi, G. Singh, K. Smith, A. Sorniotti, C. Stathakopoulou, M. Vukolić, A. Barger, S. W. Cocco, J. Yellick, V. Bortnikov, C. Cachin, K. Christidis, A. De Caro, D. Enyeart, C. Ferris, and G. Laventman, "Hyperledger fabric," pp. 1–15, 2018.

[6] G. Wood, "Ethereum: a secure decentralised generalised transaction ledger," *Ethereum Project Yellow Paper*, pp. 1–32, 2014.

[7] K. K. Gai, Y. L. Wu, L. H. Zhu, M. K. Qiu, and M. Shen, "Privacy-preserving energy trading using consortium blockchain in smart grid," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 6, pp. 3548–3558, 2019. [Online]. Available: https://ieeexplore.ieee.org/ielx7/9424/8736410/08613816.pdf?tp=&arnumber=8613816&isnumber=8736410&ref=https://dx.doi.org/10.1109/tii.2019.2893433

[8] V. Buterin, "Chain Interoperability [white paper]," 2016. [Online]. Available: http://www.r3cev.com/s/Chain-Interoperability-8g6f.pdf

[9] S. Thompson, "The preservation of digital signatures on the blockchain," *See Also: the University of British Columbia iSchool Student Journal*, vol. 3, no. Spring, 2017.

[10] Y. Wang, Z. Su, and N. Zhang, "Bsis: Blockchain-based secure incentive scheme for energy delivery in vehicular energy network," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 6, pp. 3620–3631, 2019.

[11] L. M. Camarinha-Matos, "Collaborative smart grids – A survey on trends," *Renewable and Sustainable Energy Reviews*, vol. 65, pp. 283–294, 2016. [Online]. Available: http://dx.doi.org/10.1016/j.rser.2016.06.093

[12] C. Cooper, M. Dyer, and C. Greenhill, "Sampling regular graphs and a peer-to-peer network," *Combinatorics Probability and Computing*, vol. 16, no. 4, pp. 557–593, 2007.

[13] J. Jung, S. Chun, and K. H. Lee, "Hypergraph-based overlay network model for the Internet of Things," *IEEE World Forum on Internet of Things, WF-IoT 2015 - Proceedings*, pp. 104–109, 2015.

[14] M. E. Newman, "Modularity and community structure in networks," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 103, no. 23, pp. 8577–8582, 2006.

[15] E. Androulaki, A. Barger, V. Bortnikov, S. Muralidharan, C. Cachin, K. Christidis, A. De Caro, D. Enyeart, C. Murthy, C. Ferris, G. Laventman, Y. Manevich, B. Nguyen, M. Sethi, G. Singh, K. Smith, A. Sorniotti, C. Stathakopoulou, M. Vukolić, S. W. Cocco, and J. Yellick, "Hyperledger Fabric: A Distributed Operating System for Permissioned Blockchains," *Proceedings of the 13th EuroSys Conference, EuroSys 2018*, vol. 2018-Janua, no. 1, 2018.

[16] K. Soska, A. Kwon, N. C. Cmu, and S. Devadas, "Beaver: A Decentralized Anonymous Marketplace with Secure Reputation," *IACR Cryptology ePrint Archive*, p. 464, 2016.

[17] R. Mercer, "Privacy on the Blockchain: Unique Ring Signatures," 2016. [Online]. Available: http://arxiv.org/abs/1612.01188

[18] E. Androulaki, C. Cachin, A. De Caro, and E. Kokoris-Kogias, "Channels: Horizontal scaling and confidentiality on permissioned blockchains," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 11098 LNCS, pp. 111–131, 2018.

[19] A. Shrestha, R. Bishwokarma, A. Chapagain, S. Banjara, S. Aryal, B. Mali, R. Thapa, D. Bista, B. P. Hayes, A. Papadakis, and P. Korba, "Peer-to-Peer Energy Trading in Micro/Mini-Grids for Local Energy Communities: A Review and Case Study of Nepal," *IEEE Access*, vol. 7, pp. 131 911–131 928, 2019.

[20] A. Dorri, S. S. Kanhere, R. Jurdak, and P. Gauravaram, "LSB: A Lightweight Scalable Blockchain for IoT security and anonymity," *Journal of Parallel and Distributed Computing*, vol. 134, pp. 180–197, 2019.

[21] C. Pop, M. Antal, T. Cioara, I. Anghel, D. Sera, I. Salomie, G. Raveduto, D. Ziu, V. Croce, and M. Bertoncini, "Blockchain-Based Scalable and Tamper-Evident Solution for Registering Energy Data," *Sensors*, vol. 19, no. 14, p. 3033, 2019. [Online]. Available: https://www.mdpi.com/1424-8220/19/14/3033

[22] J. Eberhardt, M. Peise, D.-h. Kim, and S. Tai, "Privacy-Preserving Netting in Local Energy Grids," no. Icbc, 2020.

[23] F. Gao, L. Zhu, M. Shen, K. Sharif, Z. Wan, and K. Ren, "A Blockchain-Based Privacy-Preserving Payment Mechanism for Vehicle-to-Grid Networks," *IEEE Network*, vol. 32, no. 6, pp. 184–192, 2018.

[24] A. Dorri, F. Luo, S. S. Kanhere, R. Jurdak, and Z. Y. Dong, "SPB: A Secure Private Blockchain-Based Solution for Distributed Energy Trading," *IEEE Communications Magazine*, vol. 57, no. 7, pp. 120–126, 2019. [Online]. Available: https://ieeexplore.ieee.org/document/8767089/

[25] L. Yao, Q. Z. Sheng, N. J. Falkner, and A. H. Ngu, "ThingsNavi: Finding most-related things via multi-dimensional modeling of human-thing interactions," *MobiQuitous 2014 - 11th International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services*, pp. 20–29, 2014.

[26] C. Jia, L. Xie, X. Gan, W. Liu, and Z. Han, "Overall Peer Consulting Distribution," vol. 42, no. 1, pp. 164–177, 2012.

Samuel Karumba is a Ph.D. student at UNSW and a postgraduate research intern at CSIRO, Australia. He received his MSc. in Computer Science from Strathmore University, in Kenya. Before commencing his Ph.D., Samuel worked as a Research Software Engineer at IBM Research Africa for 4 and a half years, researching and developing blockchain-based technological solutions for Financial Inclusion, Agriculture and Supply Chain. His research interests are Blockchain technologies, Distributed Systems and the Internet of Things (IoT). Samuel is currently researching on blockchian-based decentralised management of distributed energy resources for future smart grids.

Salil S Kanhere received his M.S. degree and Ph.D. degree from Drexel University in Philadelphia. He is a Professor of Computer Science and Engineering at UNSW Sydney, Australia. His research interests include Internet of Things, blockchain, pervasive computing, cybersecurity and applied machine learning. He is a Senior Member of the IEEE and ACM, a Humboldt Research Fellow and an ACM Distinguished Speaker. He serves as the Editor in Chief of the Ad Hoc Networks journal and as Associate Editor of IEEE Transactions on Network and Service Management, Computer Communications and Pervasive and Mobile Computing. He has served on the organising committee of many IEEE/ACM international conferences including PerCom, CPS-IOT Week, MobiSys, WoWMoM, MSWiM, ICBC.

Raja Jurdak is a Professor of Distributed Systems and Chair in Applied Data Sciences at Queensland University of Technology, and Director of the Trusted Networks Lab. He received the PhD in information and computer science from the University of California, Irvine. He previously established and led the Distributed Sensing Systems Group at CSIRO's Data61, where he maintains a visiting scientist role. His research interests include trust, mobility and energy-efficiency in networks. Prof. Jurdak has published over 170 peer-reviewed publications, including two authored books most recently on blockchain in cyberphysical systems in 2020. He serves on the editorial board of Ad Hoc Networks, and on the organising and technical program committees of top international conferences, including Percom, ICBC, IPSN, WoWMoM, and ICDCS. He is a conjoint professor with the University of New South Wales, and a senior member of the IEEE.

Subbu Sethuvenkatraman is a senior researcher with CSIRO's energy business unit. He has over 15 years of applied research experience in development and implementation of low carbon technologies for distributed generation, built environment applications. His current research is focused on ways to use advances in digital technologies to reduce built environment carbon foot print.