A Cross-Layer Trust-based Consensus Protocol for Peer-to-Peer Energy Trading Using Fuzzy Logic

Mohammad Jabed M. Chowdhury, Muhammad Usman, Md Sadek Ferdous, Niaz Chowdhury, Anam Ibna Harun, Umme Sumaya Jannat, Kamanashis Biswas

***Abstract*—Peer-to-Peer (P2P) energy trading platforms are being actively designed, tested and operated by engineers, power distribution companies and prosumers. The assurance of the accountability of the conduct of different stakeholders through a robust trust management mechanism is imperative in such platforms. The usage of blockchain, as an underlying technology, can ensure numerous properties such as immutability, transparency and traceable execution of transactions, in addition to ensuring trust establishment among different entities of the system. Few blockchain-based decentralised energy trading platforms have been designed in the literature to build trust about the platform and among prosumers. However, none of these proposals have considered human-in-the-loop in the trust establishment process. Moreover, these solutions has considered trust only at a particular layer of blockchain, such as at the application or consensus layer. To bridge this gap, this paper presents a novel cross-layer trust-based consensus protocol that considers human-in-the-loop and employs fuzzy logic to address the issue of vagueness of trust values by offering human interpretable trust level. The experiment results demonstrate the efficiency and effectiveness of our proposed protocol in comparison to established consensus mechanisms. The analysis also shows the protocol is immune against selfish mining, 51% and Sybil attacks.**

***Index Terms*—Blockchain, Cross-Layer, Energy Trading, Fuzzy Logic, Trust.**

I. Introduction

Smart homes, as part of smart cities, are typically equipped with renewable energy resources that can produce their own electricity [1]. These smart homes are also capable of supplying excessive energy to microgrids. Peer-to-Peer (P2P) energy trading allows the sale of private or community-owned energy generators and storage sites to sell their energy directly to consumers, and the people who own/invest in the energy generation sell their energy at a rate that is privately agreed upon between the producer and consumer. This eliminates the middle man, being the "Energy Retailer", providing a cheaper and fairer energy price to the consumer [2].

In a distributed P2P trading environment, neither a producer nor a consumer of the electricity can be taken as a trustworthy entity. The trust establishment on the trading platform is, therefore, pivotal for its adoption in the consumer market and a blockchain-driven P2P mechanism can help in this regard. However, existing blockchain-driven solutions mainly focus on the reputation-based consensus [3] or system architecture [4] to establish trust. The underlying mechanisms have primarily considered the behaviour of miner nodes [5], [6] in the system and the opinions of producers (sellers) and consumers (buyers) are not considered in the trust establishment and evaluation processes. These studies have considered trust constituent factors only at blockchain consensus or application layers and completely overlooked several important aspects that could be utilised for trust establishment and evaluation purposes. Therefore, there is a need for a more holistic (cross-layer) trust establishment and evaluation mechanism that considers multiple aspects at different layers for different entities. Moreover, the system should monitor the behaviour of crucial parties in the system and ultimately provide rewards or punishments based on their behaviour. Furthermore, trust computation usually results into some numeric values which often could be ambiguous and subjective to human interpretation. Human beings are more comfortable with qualitative trust measures than some numeric values [7].

Considering all these factors, we propose a cross-layer trust-based consensus protocol suitable for a blockchain-empowered P2P energy trading platform. The proposed protocol utilises the notion of trust in a holistic manner across multiple layers of the blockchain system and computes scores for different entities accordingly. Then, fuzzy logic is utilised to construct a qualitative scaling to avoid any ambiguity [8]. Following are the key contributions of this work:

- A cross-layer trust-induced consensus protocol and architecture are proposed to realise a blockchain-empowered P2P energy trading platform.
- Leveraging the *human-in-the-loop* factor to compute trust. To realise this concept in a novel way, we mainly explored ratings of traders, time spent by a trader in the network, total coins traded and trust score of the trading platform on the basis of both human and service experiences.
- Implementation of the proposed protocol within a simulated environment is carried out to highlight its applicabil-

M. J. M. Chowdhury is with Department of Computer Secience and Information Technology, La Trobe University, VIC 3083, Australia. (e-mail: m.chowdhury@latrobe.edu.au).

M. Usman is with the Faculty of Computing, Engineering and Science, University of South Wales, Pontypridd CF37 1DL, U.K. (e-mail:muhammad.usman@southwales.ac.uk).

M. S. Ferdous is with the Department of Computer Science and Engineering, Shahjalal University of Science and Technology, Sylhet, 3114, Bangladesh and Imperial College Business School, Imperial College London, London, SW7 2AZ, U.K. (e-mail: m.ferdous@imperial.ac.uk)

N. Chowdhury is with Knowledge Media Institute, Open University, Milton Keynes, MK7 6AA, UK. (email: niaz.chowdhury@open.ac.uk)

Anam. I. Harun is with the Department of Computer Science and Engineering, Shahjalal University of Science and Technology, Sylhet, 3114, Bangladesh. (e-mail: anamibnaharun@gmail.com)

U. S. Jannat is with bKash, Dhaka, 1212, Bangladesh. (e-mail: tanjan.sj@gmail.com)

K. Biswas is with Faculty of Law and Business, Australian Catholic University, QLD 4014, Australia and School of ICT, Griffith University, QLD, 4222. (e-mail: Kamanashis.Biswas@acu.edu.au, k.biswas@griffith.edu.au)

Manuscript received August 23, 2020.

ity and ef ciency in comparison to well known consensus mechanisms. The results demonstrate the promising performance of the proposed consensus protocol in terms of elapsed time and memory and CPU consumption as compared to two well-known consensus protocols.

We have analysed the works related to blockchain-based energy trading in Section II. The high level overview of the system and proposed cross-layer trust architecture are elucidated in Section III. In Section IV, we have discussed our trust model, which is followed by evaluation and discussion in Section V and Section VI, respectively. Finally, the conclusion is drawn in Section VII.

II. Related Work

Multiple disciplines in energy trading, various notions of computer, engineering and mathematical sciences, economics, and other relevant disciplines join forces together to make it a success. Blockchain is one of the latest technologies used in the development of smart grids and energy trading platform [9], [2]. Amongst the earlier proposals, Hassan et al. described DEAL, a blockchain-based auction platform for microgrid energy trading. They kept the costly proof-of-work as the underlying consensus mechanism and focused extensively on privacy issues [10]. The proposal tried to aid trust using blockchain technology but did not have a separate in-house algorithm to achieve that goal. On the other hand, Gai et al. proposed a consortium blockchain-based energy trading platform for smart grids [11]. They have several algorithms to deliver a privacy-preserving trading platform but always stayed above the consensus layer, and their functionalities are con ned within the application or meta-application layer (a description of the layers is presented later in Section III).

Lu et al. proposed a distributed energy trading scheme supported by blockchain technology and software-de ned network (SDN) [12]. It introduces a concept called manager node that helps to complete the transactions. Despite using a blockchain, the scheme seems less decentralised and emphasised on achieving data security and privacy-preserving transactions using a distributed hash table (DHT) and bloom lter data structure. It is not clear from the proposal how the system reaches consensus and manages the trust, or if they are at all considered. Wang et al. moved a step forward by proposing an improved architecture of a blockchain-based crowdsourced energy system operating over a peer-to-peer network [5]. They used Hyperledger Fabric to implement their prototype and proposed a comprehensive design of an incentive-based crowdsourcing platform for energy trading. Dorri et al. took a big leap forward and came up with a new private blockchain platform managed by a consortium of energy producers, consumers, prosumers, and distribution companies [13].

More recently, Ali et al. and Nguyen et al. proposed peer-to-peer energy trading platforms using blockchain technology to enable distributed and secured energy trading for localities [14], [1]. Guan et al. further emphasised on the security concern and presented an IoT-based blockchain-enabled platform for secure and ef cient energy trading [15].

Fig. 1: Proposed cross-layer trust architecture

III. System Model and Cross-Layer Trust Architecture

In this section, we present a high-level system overview, proposed cross-layer architecture and underlying consensus mechanism. There are several components in a blockchain system whose functionalities range from collecting transactions, propagating blocks, mining, achieving consensus and maintaining the ledger for its underlying crypto-currencies, and so on. These components can be grouped together according to their functionalities using different layers similar to the well-known TCP/IP layer. The motivation of a layered design is its easier maintenance due to modular nature. Towards this aim, Ferdous et al. [16] introduced four layers, namely network, consensus, application, and meta-application layers. In this paper, we extend the four layer architecture into cross-layer trust architecture, illustrated in Fig. 1. This cross-layer architecture is particularly designed for a blockchain-based energy trading platform, which essentially provides a marketplace for energy sellers and buyers. However, this can be generalised for other applications. In an energy trading system, energy sellers (producers) with solar panels mounted on their roof-tops are connected to the system. This enables them to offer their (additional) energy to buyers (consumers). The solar panels are connected to the local electrical substation, which supplies energy to buyers. The unsold energy goes to the main power grid. A buyer can directly buy energy from a seller using our energy trading platform.

The cornerstone of the proposed system is a trust-based cross-layer consensus mechanism which relies on the trust valuations of different entities at different layers. Speci cally, trust valuations are presented as numerical trust scores (as well

as fuzzy logic) for different entities at the meta-application, application and consensus layers. The central component of our consensus mechanism is the utilisation of a crypto-currency called Trust Coin which is envisioned as a stable crypto-currency (colloquially known as *stablecoin*) and is pegged to the respective fiat currency of the country in which it is deployed. This trust coin is essential for anybody to participate in the system as well in the consensus mechanism. Entities can buy and sell trust coins from the designated exchanges. The buyers and sellers' scores contributed to the application layer in the calculation of the trust coin. In the same way, the validator's trust score also contributes to the application. The values of the upper and lower layers are contributing to the application layer (See Fig. 1). Therefore, we have call this a cross-layer protocol. Next, we present the internal mechanisms of the consensus algorithm at different layers.

Another key characteristic of our proposed consensus mechanisms is to integrate the concept of human-in-the-loop in the trust computation process. A number of features such as information quality, ease of dealing, transaction safety, peak cost, non-peak cost and satisfaction realise human experience at the meta-application layer. These factors have direct impact on the computation of trust score, which is then utilised to choose validators in the consensus layer. Therefore, as per our proposed mechanism, human satisfaction is imperative for better performance of the energy trading system.

Meta-application layer: In the context of our application, different entities, i.e., sellers and buyers, would participate in this layer using the marketplace. These entities would utilise a web-based UI (user interface) to create a wallet to simulate their digital identities by creating a private-public key pair. Any entity can access the marketplace using the UI to buy and sell electricity. Our system has two types of transactions: i) coin transactions and ii) trust transactions. Once a buyer accepts an offer, a coin transaction is created in which a certain amount of trust coin from the buyer to the seller is transferred. This transaction is propagated in the network which is then included in a block. Once this block is added in the blockchain, implying the finality of the coin transaction, the buyer and seller belonging to the respective transaction will need to rate each other as well as the system (marketplace platform). Furthermore, they also need to rate their experience of the received service. All these form the basis for a trust transaction which, similar to a coin transaction, is propagated within the network and consequently added in a block and then to blockchain. Once this happens, the respective trust scores of the buyer and the seller as well as the platform, are updated using the process described at the application layer.

Application layer: This layer includes crypto-currencies and smart-contract features of the blockchain. In our context, this layer is responsible for the smart-contract which is used to store the trust score of every entity within the system. As soon as a block is added in the chain, all trust transactions are extracted from the block within the smart-contract. Then the trust scores of every involved entity from the extracted transactions are calculated based on pre-defined formulas outlined in the subsequent section. The UI in the meta-application layer just interacts with the smart-contract to query for the trust score for each entity. Thus, the smart-contract acts as an internal Oracle of trust scores for different entities within the system. In addition, the trust coin is calculated and rewarded to the validator who creates the block.

Consensus layer: In our system, the proposed consensus algorithm is based on the concept of Proof-of-Stake (PoS) [17] and Proof-of-Burn (PoB) [18] consensus algorithms. In a PoS, an entity needs to deposit a certain amount of crypto-currency for the corresponding blockchain to be a validator (block creator in a PoS system) and selected based on their deposit. The validator is rewarded for creating a block. However, when he is found to misbehave, the deposit gets confiscated. In PoB, a miner needs to burn a certain amount of crypto-currency to join in the block creation process and get rewarded. In our algorithm, a certain number of nodes, called *validators*, participate in the consensus process. A node can be a validator after satisfying two conditions: i) it must stake a certain amount of unrecoverable trust coin in the smart-contract and ii) its trust score is above a minimum threshold value. Every validator is assigned a base trust score when it deposits the required amount of trust coin in the smart-contract. This base trust score increases over time as the validator creates blocks. A block is generated in every minute if there is at least a single transaction. If there is no transaction within a minute, no block is generated. We define an epoch as a collection of ten blocks validated by ten validators. The smart-contract selects and ranks top ten validators, as per their deposited trust coins, from the list of validators for the next epoch. These validators are selected for validating ten subsequent blocks in a random fashion. The deposits from the validators are not retrievable once they have been selected. Thus, creating the notion of coin burn as in PoB. Once the fth block in the current epoch is generated, the smart-contract then selects the next ten validators, for the next epoch, from the set of other validators excluding the validators of the current epoch and this process continues. A validator is rewarded with a certain amount of trust coins for validating a block and its trust score is updated following a specific formula. This formula to calculate the trust score for the validator relies on a number of factors passed from the upper two layers, namely, meta-application and application layers. Thus, creating the notion of a cross layer consensus mechanism. If any validator is found to misbehave, it is heavily penalised by reducing its trust score. If a validator misbehaves a certain number of times, its trust value will be dynamically deducted below the required threshold, thereby, implicitly withdrawing itself from the list of validators.

IV. Trust Model

This section elucidates the mathematical and algorithmic description of our proposed cross-layer trust-based consensus protocol.

A. Meta-application Layer

The energy buyers and sellers rate each other on the basis of their trading experiences. The energy trading is rated, at the meta-application layer, against a number of evaluation factors,

TABLE I: Meta Layer Features.

Category	Notation	De nition	Scale
Rating of trader		–	0-5
Time spent by a trader in the network		–	0-5
Total coins traded		–	0-5
Human Experience	h_1	Information quality	0-1
Human Experience	h_2	Ease of dealing	0-1
Human Experience	h_3	Transaction safety	0-1
Human Experience	h_4	Peak cost	0-1
Human Experience	h_5	Non-peak cost	0-1
Human Experience		Satisfaction	0-1
Service Experience	s_1	Availability	0-1
Service Experience	s_2	Response time	0-1
Service Experience	s_3	Latency	0-1
Service Experience	s_4	Throughput	0-1
Service Experience	s_5	Reliability	0-1
Service Experience		Satisfaction	0-1

denoted as a tuple $< ; ; ; ; >$, where is the rating of the trader, denotes the total time spent by a trader in the network, represents the total coins traded by a trader, shows human experience, i.e., satisfaction of energy trading service and is the service experience of the energy trading service. It is pertinent to observe that the trust score of traders is derived from , , and whereas the trust score for the platform depends on and . The summary of features for this layer is shown in Table I.

Trust score of traders: Traders are buyers and sellers in the system whose trust scores are derived after each transaction. The trader's just score will build trust about individual buyer or seller in the system. A trader's trust score depends on the following factors:

Rating of traders (): The traders rate each other as per the Likert scale, where 1 denotes the lowest and 5 represents the highest information quality of the particular aspect [19]. The rating of a trader is de ned as $= f\ _1;\ _2;\ _3; \ldots\ldots\ _n g$, where $_1$; $_2$, ... represent the rating derived after corresponding transactions 1, 2, and so on.

Total time spent (): This captures the total time spent by the traders. The more time spent by the traders in the system, the more trusted they are. The value ofis calculated by subtracting the registration time (of trader) in the system from the current time, resulting in an unbounded value. We de ne a normalisation function (given in (1), where is the unbounded value) to make it bounded within the 0 to 5 range. We have chosen this range due to its wider usage in e-commerce [19]. The normalisation function is formulated by extending the Hyperbolic Tangent function [20].

$$= \frac{5}{1 + e^{\ 0:5x}} \tag{1}$$

Total coins traded (): This captures the value for total trust coins traded by the traders. This is also an unbounded value and we use (1) to make it a bounded value (0-5 range).

Trust score of the platform: The platform's trust will build trust about the platform. The trust score of the platform is derived on the basis of and which are discussed next.

Human experience (): Let be represented as a set, $= f h_1; h_2; :::; h_n g$, where h_i and $0 \quad h_i \quad n$. In this case $i = 5$, where h_1 represents information quality, h_2 shows ease of dealing, h_3 is transaction safety, h_4 denotes the peak cost and h_5 derives the non-peak cost. The features h_1 to h_5 are evaluated as follows.

The information quality, h_1, is derived on the basis of four aspects, namely, (i) accuracy, (ii) relevance, (iii) interpretability, and (iv) accessibility. Each aspect is evaluated by the energy buyer, through a user interface of the application using likert scale. The normalised value of h_1 is derived as (2).

$$h_1 = \frac{\sum_{j=1}^{4} IQ_j}{20} \tag{2}$$

where IQ_j denotes the j_{th} information quality aspect and $h_1 \ 2 \ [0; 1]$. The ease of dealing, h_2, is computed on the basis of aspects, namely, (i) time to complete a deal and (ii) deal system usability. The both aspects are based on the perception of energy buyer. We have used 5 points Likert scale to derive the value of TS (Transaction Safety), PC (Peak Cost) and NC (Non-Peak Cost). Then we divide each of the scores by 5 to normalise. Thus, the normalised value of h_2 can be derived as (3)

$$h_2 = \frac{\sum_{k=1}^{2} ED_k}{10} \tag{3}$$

where ED_k denotes the k_{th} ease of dealing aspect and $h_2 \ 2 \ [0; 1]$. The normalised value of transaction safety, h_3, can be derived as $\frac{TS}{5}$, where TS is computed on the basis of the perception of energy buyer regarding nancial transaction safety.

The normalised value of the peak energy cost, h_4, is computed as $\frac{PC}{5}$, where PC is derived on the basis of satisfaction about the peak energy cost by the buyer. Similarly, the normalised value of the non-peak energy cost, h_5, is computed as $\frac{NC}{5}$, where NC is derived on the basis of satisfaction about the non-peak energy cost by the buyer, and $h_4 \ 2 \ [0; 1]$ and $h_5 \ 2 \ [0; 1]$. The collective trust score of is derived as (4).

$$= \frac{\sum_{i=1}^{n} h_i}{n} \tag{4}$$

Service experience (): Let be represented as a set, $= f s_1; s_2; :::; s_m g$, where s_i and $0 \quad s_i \quad m$. In this case $i = 5$, where s_1 represents availability, s_2 shows response time, s_3 is latency, s_4 denotes the throughput and s_5 shows reliability. The value of is derived as (5).

$$= \frac{\sum_{i=1}^{m} s_i}{m} \tag{5}$$

Trust score calculation: To calculate the trust scores for the traders and the platform, Algorithm 1 and Algorithm 2 are utilised with the notations presented in Table II.

In particular, Algorithm 1 is executed to calculate the trust score of a trader $u()$ once the corresponding coin and trust transactions are transferred and added in the block, implying the nality of the transactions. For this, at rst t_u and t_u:Score are measured and then $u\ t_u$ and $u: t_u$:Score (the total coin traded for u and its score) are calculated (line 4 to 9 of Algorithm 1). Finally, the new trust score of u, denoted with NTS_u^t, is computed by using the weighted arithmetic average

TABLE II: Notations used in Meta-application layer.

Notation	Description
$_u^t$	Rating for trader u in t transaction
$_u^t$	Total time spent by u up to t transaction
$_u^t$:Score	Total time spent score for u
p: $_u^t$	Total coin traded for u before t transaction
u: $_u^t$	Updated total coin traded for u after t transaction
$_u^t$	Total coin traded for u in t transaction
u: $_u^t$:Score	Total coin traded score for u after t transaction
^{t}	Human experience score in t
^{t}	Service quality score in t
FN ^{t}	Number of feedback before t
NTS $_u^t$	New trust score for u after t transaction
PTS $_u^t$	Previous trust score for u before t transaction
PTSP ^{t}	Previous trust score for the platform before t
NTSP ^{t}	New trust score for the platform after t
RT $_u$	Registration time for u

formula which utilises the number of old transactions by u and their corresponding trust scores as well the calculated trust score of the current transaction (line 10 to 12 of Algorithm 1).

Similarly, The trust score for the platform is computed using Algorithm 2 which leverages the weighted arithmetic average formula that depends on the number and the old scores and the current human experience ^{t} and current service quality ^{t} (line 4 to 5 of Algorithm 2).

B. Trust Membership Function

Trust values are shown as fuzzy numbers, namely, high, moderate, and low. The fuzzy number low has normalised value range $[a; c]$. The fuzzy numbers medium and high, on the other hand, have normalised value ranges $[b; e]$ and $[d; f]$, respectively, where $a \quad b \quad c \quad d \quad e \quad f$. The values of these parameters, however in practice, are defined by the electricity trading consortium and other entities of the community.

Let us consider the meta-application layer trust as Universe of Discourse, representing trust value as: “Low”, “Medium”, and “High”. The corresponding trust membership functions are defined below and a generalised depiction is shown in Fig. 2.

For low, if $t > c$, the trust value is zero. If $b \quad t \quad c$, then the corresponding value is calculated using equation of line segment $\frac{c \quad t}{c \quad b}$. Finally, if $t < b$ the corresponding function value is 1. The function for low trust is given as (6).

Fig. 2: Trust membership function

Algorithm 1: Trust score calculation for u after t

```
1  Input:  _u^t ; RT_u ;  _u^t ; PTS_u^t
2  Output: NTS_u^t //  . New trust score for  u
3  Start
4      currentTime = time();
5       _u^t = currentTime  RT_u ;
6       _u^t :Score = 5 / (1 + e^{0:5 _u^t});
7      p: _u^t = totalCoinTraded(u);
8      u: _u^t = p: _u^t +  _u^t ;
9      u: _u^t :Score = 5 / (1 + e^{0:5(u: _u^t)});
10     temp = ( _u^t +  _u^t :Score + u: _u^t :Score) / 3;
11     oldTransList = listOfTrans(u);
12     NTS_u^t = (|oldTransList|  PTS_u^t + temp) / (|oldTransList| + 1);
13     function totalCoinTraded(u)
14         totalCoin := NULL;
15         transList = listOfTrans(u);
16         while l 2 transList do
17             Extract traded coin c from l;
18             totalCoin += c;
19         end
20         return totalCoin;
21     function listOfTrans(u)
22         LIST := NULL;
23         Build L, all transactions minus t from the
             blockchain;
24         while l 2 L do
25             if l involves u then
26                 LIST = LIST [ l;
27         end
28         return LIST;
```

$$M_l^t(T) = \begin{cases} 0; & t > c \\ \frac{c \quad t}{c \quad b}; & b \quad t \quad c \\ 1; & t < b \end{cases} \tag{6}$$

For medium trust value, if $t < b$, then the corresponding function value is zero. If the value of medium trust lies in the range: $b \quad t \quad c$, then the corresponding function value is derived using the relevant segment of (7), i.e. $\frac{t \quad b}{c \quad b}$. The value for trust is 1 for $c \quad t \quad d$. The value of trust $d \quad t \quad e$ is defined as a line segment between points d and e. Next, for $t \quad e$, the value for medium trust is zero. The function for medium trust is given below.

$$M_m^t(T) = \begin{cases} 0; & (t \quad b) \text{or} (t \quad e) \\ \frac{t \quad b}{c \quad b}; & b \quad t \quad c \\ 1; & c \quad t \quad d \\ \frac{e \quad t}{e \quad d}; & d \quad t \quad e \end{cases} \tag{7}$$

For “High”, if $t < d$, the corresponding high trust function value is zero. If the value is in the range $d: \quad t \quad e$, then the trust value is derived using $\frac{t \quad d}{e \quad d}$ in (8). If $t \quad e$, then the

corresponding trust value is 1. The function for high trust is given below.

$$M_h^t(T) = \begin{cases} 0; & t \quad d \\ \frac{t \quad d}{e \quad d}; & d \quad t \quad e \\ 1; & t \quad e \end{cases} \tag{8}$$

The trust membership function is used to generate qualitative scaling (e.g., low, medium and high) from the quantitative scores calculated by Algorithm 1 and 2 for better human interpretation.

C. Application Layer

At this layer, the value of trust is derived on the basis of crypto-currency, namely trust coin. The smart-contract maintains the trust score for individual trader and validator in the system. The trust coin computation model aims to reward honest behaviour of the validator. The model enables both energy traders and validators to participate in the trust reward model. The reward, proposed in terms of transaction fees (by the buyers), is weighted against the trust score of traders in the network. This ensures the minimum (or no) influence of the less trusted nodes. The trust coin is an imperative economic incentive for the validators to continue validating. We explain it below using the notations presented in Table III.

Algorithm 2: Calculating platform trust score after t

1 Input: $\quad^t; \quad^t; PTSP^t; FN^t$
2 Output: $NTSP^t$ // . New trust score for platform
3 Start
4 $\quad temp = \frac{^t + \quad^t}{2};$
5 $\quad NTSP^t = \frac{FN^t \quad PTSP^t + temp}{FN^t + 1};$

1) Trust coin attributes

The proposed model is based on multiple constituent elements which contribute to the computation of the awarded trust coin. These elements have different evaluation scale ranges. We normalise them, on a scale ranging from 0 to 5, to compute the amount of coins rewarded for creating a block. The structure of the constituent elements is summarised below.

Block reward (R_b): It is a fixed amount of coins (5 coins) rewarded to the validator who creates a block b. It is similar to block reward concept in Bitcoin [21].

Trust Fee: Trust fee is the amount of coin the buyer is happy to pay if a validator includes this particular transaction in a block. It is also similar to the implicit transaction fees in Bitcoin [21]. To calculate the total trust fee for a block (denoted with $_b$), trust fee for each transaction is multiplied by the trust score of the buyer of the particular transaction (refer to Equation (9)).

$$_b = \sum_{i=1}^{n} tf_i \quad ts_i \tag{9}$$

TABLE III: Notations used in application layer

Notation	Description
$_b$	Total money transacted for block b
$_b$	Trust fees for block b
$_b$	Total calculated trust coin for block b
mt_i	Money transacted for transaction i
tf_i	Trust fees for transaction i
ts_i	Trust score of the buyer for transaction i
R_b	Block reward for block b
TL_b	Transaction list in the current block b
TS_u	Trust score for trader u
TT	$\lvert TL_b \rvert$, number of total transactions in b

where the total number of transactions n is in block b.

Trust Score of Buyer: Every buyer in the platform has his/her own trust score. Details of the trust score calculation is discussed in Section IV-A. These scores are multiplied by the trust fees they offer in the transaction.

Total coin Transacted: Each transaction contains an amount of coin that is transferred between the buyer and seller's account. The validator can include one to many transactions in one block. We want to encourage the validators to include as many transactions as possible (within the rage of blocksize, 4 megabyte). The trust coin for a block, denoted with $_b$, will be proportionate to the amount of total money transacted in that block and is calculated using (10).

$$_b = \sum_{i=1}^{n} ct_i \tag{10}$$

The transaction amount (denoted with $_b$) is then normalised using Equation 11.

$$!_b = \frac{5}{1 + e^{\quad 0:5 \quad _b}} \tag{11}$$

Total Coin: The total coin for a particular block is calculated based on (12) where tv_b denotes the trust value for block b and is explained in Section IV-D.

$$_b = R_b + \quad _b + !_b + tv_b \tag{12}$$

We have used Algorithm 3 to compute the trust coin for a particular block b. This algorithm essentially uses (9) (line 7 to 11 in Algorithm 3), (11) (line 13 to 17) and (12) (line 18) to calculate the total trust coin for a block.

D. Consensus Layer

In the consensus layer, the consensus process is handled as discussed in Section III. In this section, we explore how trust score is calculated once a block is validated by a validator. The trust score depends on a number of factors which are presented below using the notations presented in Table IV.

Trust score given by other validators: Once a block is added by a selected validator, the block is propagated in the network which must be validated as well by other validators in the network. Then, the other validators will provide trust values for the validator which has proposed the block. The validator trust score is used to filter out the untrusted or less trusted validator out of the consensus process. This trust value will be calculated automatically based on the following parameters.

Algorithm 3: Trust coin calculation for block b

```
1  Input: TL_b; tv_b
2  Output: _b // . Trust coin for b
3  Start
4      R_b = 5;
5       _b = 0;
6       _b = 0;
7      while i 2 TL_b do
8          buyer = i:buyer();
9          ts_i = trustScoreVal(buyer);
10         tf_i = i:trustFee();
11          _b += tf_i  ts_i;
12     end
13     while i 2 TL_b do
14         mt_i = i:amount();
15          _b += mt_i;
16     end
17     !_b = 5 / (1 + e^(0:5 _b));
18      _b = R_b +  _b + !_b + tv_b;
19     function trustScoreVal(v)
20         trustScore = NULL;
21         Retrievetrust Score for v from the
             smart-contract;
22         return trustScore;
```

TABLE IV: Notations used in consensus layer

Notation	Description
t_v	Total time spent by validator v up to t
t_v:Score	Total time spent score for v
tv_b	Calculated trust value for b
NTS_v	New average trust score for v
PTS_v	Previous average trust score for v
PB_v	Set of proposed blocks by v
RT_v	Registration time for v
TC_b	Total coin traded in b
TC_b:Score	Total coin traded score in b
TT :Score	Total transaction score in b
V	Set of validators
v	$v \in V$, Current validator
V'	$V \setminus v$, Set of other validators

Total number of transactions: It is the total number of transactions included in the proposed block as per (9).

Total Money Transacted: It is the total amount of money transacted in all transactions within the proposed block, calculated as per (10) and (11).

Time spent in the network: It is the total time the selected validator has spent in the network, denoted with t_v. The time is then normalised using Equation 13.

$$t_v\text{:Score} = \frac{5}{1 + e^{0:5\ t_v}} \tag{13}$$

Trust score calculation: The trust score for validator v is calculated using Algorithm 4. In the algorithm, at first, t_v and t_v:score are computed (line 5 and 6). Afterwards, the total coin for the particular block and the number of transactions are determined and used to compute the trust score (tv_b) for this block (line 7 to 16). Finally, this trust score is weighted against the trust scores of all blocks proposed by this particular validator (line 17). Finally, if a validator is found to act maliciously, (14) is used to compute the penalty (where NB represents the number of malicious acts) and (15) is used to update the trust score of the validator.

$$\text{penalty} = \frac{2}{1 + e^{(\ 0:1\ NTS_v\ \ NB\ )}} \tag{14}$$

$$NTS_v = PTS_v\ \ \text{penatly} \tag{15}$$

Algorithm 4: Trust score calculation for validator v

```
1  Input: RT_v; V'; PTS_v; TL_b
2  Output: NTS_v // . New trust score for v
3  Start
4      currentTime = time();
5       _v = currentTime  RT_v;
6       _v:Score = 5 / (1 + e^(0:5  _v));
7      TC_b = totalCoinTraded(TL_b);
8      TC_b:Score = 5 / (1 + e^(0:5TC_b));
9      TT = |TL_b|;
10     TT :Score = 5 / (1 + e^(0:5TT));
11     avgTrust = ( _v:Score + TC_b:Score + TT :Score) / 3;
12     temp = 0;
13     while i 2 V' do
14         temp += (avgTrust  trustScoreVal(i)) / 5;
15     end
16     tv_b = temp / |V'|;
17     NTS_v = (PTS_v  |PB_v| + tv_b) / (|PB_v| + 1);
18     function totalCoinTraded(TL_b)
19         totalCoin := NULL;
20         while l 2 TL_b do
21             Extract traded coin c from l;
22             totalCoin += c;
23         end
24         return totalCoin;
25     function trustScoreVal(v)
26         trustScore = NULL;
27         Retrievetrust Score for v from the
             smart-contract;
28         return trustScore;
```

V. Performance Evaluation

To test our proposed algorithms' feasibility and performance, we have utilised a blockchain simulator; namely, SimBlock [22] is a purpose-built blockchain simulator that can simulate blockchain functionalities such as block generation, block propagation, node management and different consensus algorithms such as PoW and PoS. Therefore, at first, we have added the capability of transactions, both for coin and

trust transactions and then developed our proposed consensus algorithm in SimBlock. This has allowed us to evaluate and compare three consensus algorithms, PoW, PoS and ours, against the backdrop of the same network configurations. Another issue is that SimBlock does not have any smart-contract and meta-applications features. Therefore, we have simulated these functionalities within SimBlock.

At the start of the simulation, each user (buyer/seller) was allocated 100 units of trust coins. Similarly, each validator was allocated between 2000 to 5000 units of trust coins with an assigned base trust score of 3. We evaluated time taken to complete one round of simulation (elapsed time) with these configurations, the memory and CPU usage for each round and dynamic updates in trust scores for different entities as more transactions and blocks were generated in the system. The elapsed time for each simulation helped us identify the consensus algorithm that took the lowest time to complete in a single round of simulation.

Ideally, it is a better indicator of the performance if TPS (transaction per second) for each algorithm can be computed. Unfortunately, SimBlock creates transactions and blocks in a fixed rate which needs to be setup as a configuration parameter for any simulation. Hence, TPS is not a useful performance indicator for SimBlock. That is why we have opted to use elapsed time instead. On the other hand, the memory and CPU usage provide an indication of the relative resource consumption for each consensus algorithm. Finally, the trust score update provides a visual analogy of how trust scores for different entities increase or decrease as they interact within the system.

We simulated between 250 to 8000 nodes for each consensus algorithm and recorded the elapsed time, memory consumption and trust score updates. The experiments were carried out in a PC with a Ubuntu 20.10 OS, having Intel Core-i7 2.50GHz CPU, 8GB DDR4 RAM, 1TB Hard disk and Intel HD 520 GPU. 50 iterations of simulations were carried out. The average results are presented and discussed in the subsequent subsections.

A. Elapsed time

The graph of elapsed time for PoW, PoS and the proposed (set of) algorithm is presented in Fig. 3a. The elapsed time starts to increase as the number of nodes increases. The rate of increase of the elapsed time is almost exponential for PoW and PoS, taking between 863 ms and 32394 ms for 250 nodes and 8000 nodes for PoW and 854 ms and 28820 ms for PoS respectively. For our proposed algorithm, the rate of increase is not as exponential as PoW and PoS. Nevertheless, it starts to increase substantially from 832 ms to 17958 ms for 250 and 8000 nodes, respectively. Another observation is that PoW has the highest elapsed time in all node ranges. Our algorithm has the lowest time and PoS has moderate time consumption. In comparison, the elapsed time for PoW for 250 nodes is 863 ms, whereas for our algorithm is 832 ms, a reduction of around 4%. Similarly, the elapsed times for PoW and our algorithm for 8000 nodes are 32394 ms and 17958 ms, reducing around 45%. The reduced elapsed time for our proposed algorithm is, as discussed earlier, a comparative indication of which algorithm performs faster. In this regard, the performance of our algorithm is substantially better than PoW and PoS, particularly when the number of participating nodes is large.

B. Memory Consumption

The comparative memory consumption percentage for PoW, PoS and the proposed algorithm is presented in Figure 3b. Even though the relative memory consumption is mostly similar, our consensus algorithm consumed the lowest memory among all algorithms with PoW consuming the highest in all node ranges. For example, the percentages of memory consumption between PoW and our algorithms for 250 nodes are 40.46% and 39.88% and for 8000 nodes are 46.16% and 43.8%, respectively. Thus, the graph shows that the proposed algorithm is more efficient in memory consumption than PoW and PoS. One might wonder about the high load of memory consumption (around 40%) for all algorithms. This is because of SimBlock being written in Java consumes high memory during the execution period [23]. Even so, this graph is useful to illustrate the comparative performance in terms of memory consumption between these consensus algorithms. CPU consumption has a similar effect; thus, its description is excluded for brevity.

C. Trust Score

We also studied the evolution of trust score for different validators following Algorithm 4 as different blocks were validated. We simulated some misbehaving validators in our experiment to observe the effect on their trust scores. The result of this experiment with three validators is plotted in Figure 4, where the trust score for each validator is updated after every 5 blocks. All validators start with a base trust score of 3 which increases as per the algorithm when a selected validator proposes a block. If a validator misbehaves (Validator 1 in block no. 45 and Validator 2 in block no. 35 and 50) its trust score is reduced drastically.

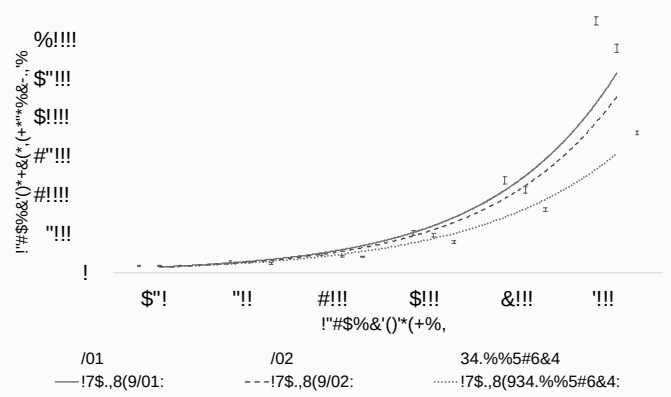
D. Complexity Analysis

In this section, we analyse the time complexities for all algorithms.

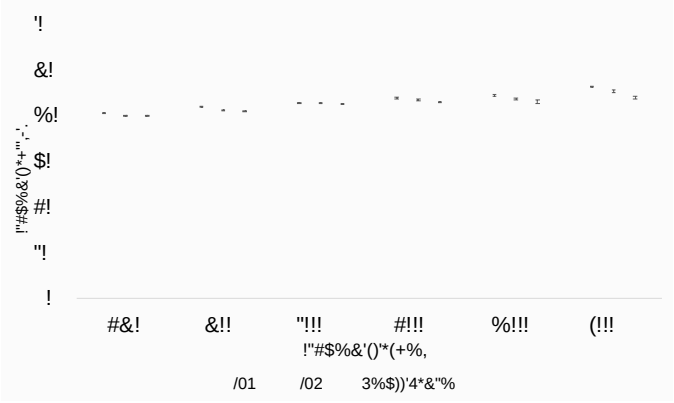
Proposition 1. The time complexity for (i) Algorithm 1, (ii) Algorithm 3 and (iii) Algorithm 4 is $O(n)$.

Proof. (i). Algorithm 1 calculates the trust score for a trader u after a transaction t at the meta-application layer. This trust score (denoted with $INTS_u^t$) depends on a number of factors such as total time spent score by u (denoted with $_u^t$:Score), total coin traded for u before t transaction (denoted with $_u^t$), total coin traded score for u after t transaction (denoted with u: $_u^t$:Score). It takes constant time to calculate all these factors, except the total coin traded $_u^t$) which requires the complexity of $O(n)$ for building the list of transactions involving u (functions totalCoinTraded, Line 13 to 20 in Algorithm 1 and listOfTranst, Line 21 to 28 in Algorithm 1). Therefore, the total time complexity for Algorithm 1 can be expressed as $O(n)$.

(ii). Similarly, Algorithm 3 calculates the trust coin $_b$) for block b. $_b$ is calculated using Equation 12 where it depends on the block reward R_b, total fees for b ($_b$), total money



(a) Elapsed time



(b) Memory Consumption

Fig. 3: Performance comparison among PoW, PoS and Proposed Algorithm

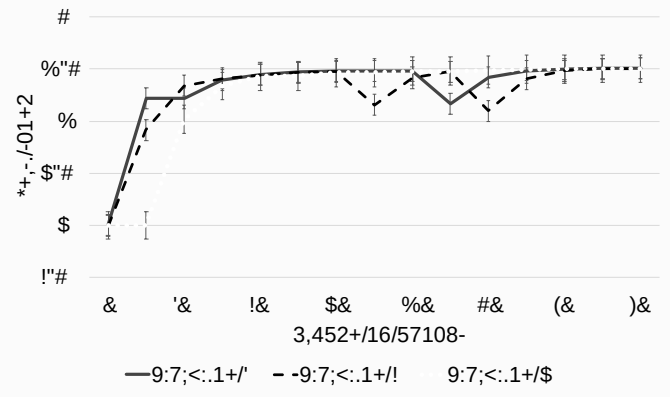


Fig. 4: Trust score updating for validators

transacted in b ($!_b$) and trust value for b (tv_b). Among these, the calculations of $_b$ and $!_b$ require iterative steps (line 7 to 11 and line 13 to 17 in Algorithm 3), thus requiring the complexity of $2{:}O(n)$. Other factors can be calculated in constant time complexity of $O(1)$. Therefore we conclude that the total time complexity for Algorithm 3 is $O(n)$.

(iii). Lastly, Algorithm 4 calculates the trust score for a validator v. This trust score (denoted with NTS_v) depends on the previous average trust score for v (denoted with PTS_v), set of proposed blocks by v (denoted with PB_v) and the calculated trust value for the current block b (denoted with tv_b). Among these, the calculations of PB_v and tv_b require iterative steps, thus resulting in the complexity of $2{:}O(n)$ whereas PTS_v) can be calculated in constant time $O(1)$. Therefore, we conclude that the total time complexity for Algorithm 4 is $O(n)$. □

Proposition 2. *The time complexity for Algorithm 2 is* $O(1)$.

Proof. Algorithm 2 calculates the platform trust score after a transaction t. This trust score (denoted with $NTSP^t$) depends on the human experience score in t (denoted with ^{t}), the service quality score in t (^{t}), the previous trust score for the platform before t ($PTSP^t$) and the number of feedback before t (FN^t). All these are provided as inputs in Algorithm 2 and the algorithm calculates $NTSP^t$ in constant time (line 4 and 5 in Algorithm 2). Therefore, we conclude that the time complexity for Algorithm 2 is $O(1)$. □

VI. Discussion

This section describes the design decision (Section VI-A) and threat analysis related to our trust model (Section VI-B).

A. Design Decision

We took the following design decisions to ensure a reliable trust score and trust coin calculation.

Two types of transaction: we have used two types of transactions, namely, *coin transaction* and *trust transaction*. A coin transaction encodes a trading event where the buyer pays the seller using trust coins. Trust transactions are used to calculate different entities' trust score and trust coins as rewards for validators. We differentiated these transactions because trust calculations are based on the previous coin transactions and linked differently.

Consensus choice: To avoid energy wastage, we have chosen a combination of PoS and PoB consensus mechanisms [17], [18]. PoS does not require complex mathematical calculations to create new blocks. In addition, the validator is chosen randomly and only one validator proposes the next block rather than all the validators go for a race.

Fairness: The validator of the proposed block will get the block reward and other rewards to create the next block as *trust coins*. As the validators need to keep the trust coin as *"stake"*, the greater the amount, the higher is the *"stake"*. That means the rich validators might get an unfair advantage, as their probability will increase over the time because of the block rewards. To make the competition fair we have proposed to burn the *"stake coin"* of the validator who will be chosen randomly to propose the next block [18].

Human interpretable scaling: At the meta-application layer, we have used fuzzy-logic to come up with more user-friendly trust scaling. Rather than the numerical values (0-5), we have scaled them as low, medium and high trust, which changes in a dynamic way.

Punishment for bad behaviour: To ensure that only trusted validators participate in the consensus process, we set a minimum threshold of trust score for the validators. All the

validators will be boot-strapped with base trust score, so that they can participate. If they act honestly, their trust score will increase. However, if they act maliciously/dishonestly, their trust score will decrease, and eventually, they will be out of the consensus process.

B. Threat Analysis

Below we discuss a few common security threats related to blockchain consensus and how our proposed model is immune to these attacks.

Selfish Mining: In selfish mining attack, an antagonist selectively disclosed mined blocks to waste computational resources of honest participants. Our proposed consensus is immune to this attack as the block creator (validator) is chosen randomly and there is no computational race in the algorithm.

51% attack: In 51% attack, the maximum of the miners collaborate together for dishonest behaviour [24]. However, in our proposal, there is a trust threshold to participate in the consensus process. Therefore, it will be difficult to convince highly trusted validators for collusion.

Sybil Attack: In this attack, the antagonist attacks the reputation mechanism of a network service by generating a large number of pseudonymous identities to influence the decision making within the network. However, our trust model is more biased towards validators with higher trust score. Therefore, it is difficult for the *"fake validators"* to be picked as the block creator. In addition, for the bootstrapping, the validators need to invest money to get the minimum threshold trust score to participate in the consensus. Therefore, according to crypto-economics [25], it will be an unrealistic option for the bad actors.

VII. Conclusion

Peer-to-Peer energy trading is a subject of interest amongst the scientific community for at least past one and half-decade. The climate change movement and rapid improvements in sensors, wireless networks and blockchain technologies paved the path towards a suitable platform for Peer-to-Peer energy trading. However, building trust on these platforms is challenging. In this paper, we adopted a holistic cross-layer approach to ingrain the trust in all aspects of a blockchain-based energy trading system. We also considered different human-driven aspects to compute the trust values of the end users. We considered common security threats to blockchain-based systems while designing our solution. Finally, we benchmarked our system with two popular blockchain consensus mechanisms, namely, PoW and PoS. The experiments show that the proposed protocol can ensure better trust without adding significant overhead in comparison to other mechanisms. The protocol achieves 26–58% reduction in elapsed time and also slightly reduces memory consumption (more than 2%). In future, we would like to deploy and study our proposed consensus protocol in real-life setting using an existing blockchain system.

References

[1] D. Nguyen and T. Ishihara, "Distributed peer-to-peer energy trading for residential fuel cell combined heat and power systems," *Elsevier International Journal of Electrical Power and Energy*, 2020.

[2] Y. Zhou, J. Wu, C. Long, and W. Ming, "State-of-the-art analysis and perspectives for peer-to-peer energy trading," *Elsevier Engineering*, 2020.

[3] M. T. de Oliveira, L. H. Reis, D. S. Medeiros, R. C. Carrano, S. D. Olabarriaga, and D. M. Mattos, "Blockchain reputation-based consensus: A scalable and resilient mechanism for distributed mistrusting applications," *Computer Networks*, p. 107367, 2020.

[4] T. Morstyn, N. Farrell, S. J. Darby, and M. D. McCulloch, "Using peer-to-peer energy-trading platforms to incentivize prosumers to form federated power plants," *Nature Energy*, vol. 3, no. 2, pp. 94–101, 2018.

[5] S. Wang, A. F. Taha, J. Wang, K. Kvaternik, and A. Hahn, "Energy crowdsourcing and peer-to-peer energy trading in blockchain-enabled smart grids," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 2019.

[6] S. Biswas, K. Sharif, F. Li, S. Maharjan, S. P. Mohanty, and Y. Wang, "Pobt: A lightweight consensus algorithm for scalable iot business blockchain," *IEEE Internet of Things Journal*, vol. 7, no. 3, pp. 2343–2355, 2019.

[7] T. Liu, "How trust pursuing businesses play in an asymmetric power network?" *IEEE Transactions on Engineering Management*, vol. 67, no. 1, pp. 18–29, 2020.

[8] M. Usman, V. Muthukkumarasamy, and X.-W. Wu, "Mobile agent-based cross-layer anomaly detection in smart home sensor networks using fuzzy logic," *IEEE Transactions on Consumer Electronics*, vol. 61, no. 2, pp. 197–205, 2015.

[9] W. Hou, L. Guo, and Z. Ning, "Local electricity storage for blockchain-based energy trading in industrial internet of things," *IEEE Transactions on Industrial Informatics*, 2019.

[10] M. Hassan, M. Rehmani, and J. Chen, "Deal: Differentially private auction for blockchain based microgrids energy trading," *IEEE Transactions on Services Computing*, 2019.

[11] K. Gai, L. Wu, Y. Zhu, M. Qiu, and M. Shen, "Privacy-preserving energy trading using consortium blockchain in smart grid," *IEEE Transactions on Industrial Informatics*, 2019.

[12] X. Lu, L. Shi, Z. Chen, x. Fan, Z. Guan, and X. Du, "Blockchain-based distributed energy trading in energy internet: An sdn approach," *IEEE Access*, 2019.

[13] 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*, 2019.

[14] F. Ali, M. Aloqaily, and O. O. Alfandi, O, "Cyberphysical blockchain-enabled peer-to-peer energy trading," *IEEE Computer*, 2020.

[15] Z. Guan, X. Lu, N. Wang, J. Wu, and X. Du, "Towards secure and efficient energy trading in iiot-enabled energy internet: A blockchain approach," *Elsevier Future Generation*, 2020.

[16] M. S. Ferdous, M. J. M. Chowdhury, M. A. Hoque, and A. Colman, "Blockchain consensus algorithms: A survey," *arXiv preprint arXiv:2001.07091*, 2020.

[17] J. Yang, A. Paudel, and H. B. Gooi, "Compensation for power loss by a proof-of-stake consortium blockchain microgrid," *IEEE Transactions on Industrial Informatics*, 2020.

[18] K. Karantias, A. Kiayias, and D. Zindros, "Proof-of-burn," in *International Conference on Financial Cryptography and Data Security*. Springer, 2020, pp. 523–540.

[19] A. J. Flanagin, M. J. Metzger, R. Pure, A. Markov, and E. Hartsell, "Mitigating risk in ecommerce transactions: perceptions of information credibility and the role of user-generated ratings in product quality and purchase intention," *Electronic Commerce Research*, vol. 14, no. 1, pp. 1–23, 2014.

[20] F. Xiao, Y. Honma, and T. Kono, "A simple algebraic interface capturing scheme using hyperbolic tangent function," *International Journal for Numerical Methods in Fluids*, vol. 48, no. 9, pp. 1023–1040, 2005.

[21] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," Manubot, Tech. Rep., 2019.

[22] Y. Aoki, K. Otsuki, T. Kaneko, R. Banno, and K. Shudo, "Simblock: A blockchain network simulator," in *IEEE INFOCOM 2019-IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*. IEEE, 2019, pp. 325–329.

[23] L. Prechelt, "An empirical comparison of seven programming languages," *Computer*, vol. 33, no. 10, pp. 23–29, 2000.

[24] G. Liang, S. R. Weller, F. Luo, J. Zhao, and Z. Y. Dong, "Distributed blockchain-based data protection framework for modern power systems against cyber attacks," *IEEE Transactions on Smart Grid*, vol. 10, no. 3, pp. 3162–3173, 2018.

[25] K. Iyer and C. Dannen, "Crypto-economics and game theory," in *Building Games with Ethereum Smart Contracts*. Springer, 2018, pp. 129–141.