Trading 4.0: An Online Peer-to-Peer Money Lending Platform

Christian Malakani, Antoine Bagula, Olasupo Ajayi, and Hloniphani Maluleke

Department of Computer Science, University of the Western Cape,
Cape Town, South Africa
3965400@myuwc.ac.za

Abstract. Online money lending platforms also referred to as peer-to-peer lending platforms are a mean of lending and borrowing money between private individuals mostly via an online web-based platform. Transactions are often done without the direct arbitration of financial institutions, such as banks or other financial agencies. In 2005 the first money lending platform called Zopa was created in the United Kingdom, and several other have since followed. The exponential growth of Internet has been instrumental to the widespread adoption of peer-to-peer lending, as transactions can now be carried from anywhere via the Internet. However, despite the comfort, security of transactions and fraud prevention is still a challenge. The objective of this paper is to develop a Cloud-based money lending platform using both direct and reverse auction mechanisms based on blockchain technology to secure transactions between lenders and borrowers. The developed prototype uses knapsack and perfect sum algorithms in both direct and reverse auctioning mode, to determine the least compromising match between loan requests and offers. After rigorous testing, obtained results showed that indeed the system can meet user requirements in a secure manner.

Keywords: Auctioning · Blockchain · Cloud computing · Peer-to-peer lending · Knapsack · Perfect sum algorithm

1 Introduction

Peer-to-peer lending (P2PL) can be defined as the process of lending and borrowing money between private individuals via an online platform, without the direct involvement of financial institutions. Banks and other financial agencies (such as credit bureaus) still play vital roles, as they act as formal depositary institutions, providing bank accounts where funds are deposited and as confirmation authority for financial information provided by the user of the platform [1]. Peer-to-peer lending has been in existence for centuries and dates to Friendly Societies in Britain around the 1640s [1]. As with modern peer-to-peer lending, the members of these Friendly Societies were mostly working-class who provided each

Y. H. Sheikh et al. (Eds.): AFRICOMM 2021, LNICST 443, pp. 149–169, 2022.
https://doi.org/10.1007/978-3-031-06374-9_10

other with mutual financial support [2]. The concept of informal or semi-formal mutually beneficial societies or co-operative societies has existed for years. Only in recent times have they evolved into the Internet base platforms in existence today. While the banking system can be very restrictive, with stringent policies and requirements that must be met before loans are granted, P2PL platforms are less demanding, hence more attractive to customers. Despite their popularity and the ease of obtaining needed loans, an inherent shortcoming of P2PL is securing transactions. In the absence of regulatory bodies or well-established policies, fraud and related offenses are common. This aim of this paper is to address this security challenge by proffering a secure solution for P2PL transactions. The specific contributions of our work are to develop a Cloud-based P2PL system based on two auctioning mechanisms for financial transactions between lenders and borrowers; and to use blockchain technology to ensure the secure of such transactions. The rest of this paper is organized as follows: a review of related works is presented in Sect. 2, while our methodology and system architectures are discussed in Sect. 3. In Sect. 4 an illustrative case study is presented, while Sects. 5 and 6 respectively detail the implementation and testing processes of the Trading 4.0 platform. Finally, Sect. 7 concludes the paper and gives insight for future works.

2 Literature Review

The continuous growth of online communities has led to a new way of managing loan in the financial industry. The emergence of online P2PL platforms has moved the concept of personal loan into the web, and practically eliminated the need for third party financial institutions (such as banks and credit bureaus) mediating between lenders and borrowers. Websites such as www.prosper.com [3], offer an online platform where private borrowers and lenders can initiate a loan process between USD 2,000 and 40,000. These platforms generally force the borrower to specify the motivation behind their loan request and to provide details of their financial situation, such as income or credit history. Lenders then select the loan request that meet their requirement with the appropriate interest rate according to the information provided by the borrowers [2]. For borrowers, the online P2PL platform is a means of obtaining loans with better terms without than those offered by banks. On the lenders' side, the platform can be seen as a way to invest money. The platform makes revenues by charging service fees from realised transactions. The awareness of online P2PL rose in 2006 with the introduction of the first online peer-to-peer platform called Zopa in the United Kingdom [2]. Several other types of money lending platforms have subsequently been developed, many of which are reviewed in [4–6]. Online P2PL platforms can be divided fundamentally into two types: commercial and non-commercial [5]. In commercial platforms the lenders make profits from the interest rate of the loans they provide, and the platform make profits by charging service fees from realised transactions. In non-commercial or charity-based platforms, the main concept is not to make profits but to provide support and help people to accomplish their project without being constrained financially. Online P2PL platforms can also vary according to the way the interest rate of the borrower is fixed. Platforms such as [3] use an auction mechanism where the borrowers publish their request and set the maximum interest rate they wish to pay. The request is subsequently published on the platform for a limited period, during which lenders bid by specifying the minimum interest rate they wish to take and the amount of money they wish to provide [6]. Other platforms such as the German smava.de [7] set the interest rate based on some financial and demographic characteristics of the borrower. Borrowers and lenders are the key participants in all lending platform and many studies, including [8] have focused on these two participants to discuss the determinants that are necessary for the success of a lending activity. Lenders typically seek to invest money and gain the best possible return on investment. They do this by selecting the loan request that best suits their requirement by reviewing the information provided by the borrower. The borrower on the other hand seeks liquidity. Unfortunately, borrowers sometime hide certain information that can prevent them from getting the requested fund. Some platforms require their borrowers to supply financial records that have been checked by external organisations (banks or other credit bureaus), to enable lenders make decision based on accurate information. Other platforms request their borrowers to provide demographic information such as race, location, gender, etc., or social information such as family members, photos, hobbies, which cannot be validated. In a bid to determine the most relevant borrow information, Michael Klafft in [9] revealed that the credit rating of the borrower has the largest impact, and this is followed by debt-to-income ratio and house ownership. Information such as existence of a verified bank account had lower impact.

Auctioning is a powerful means for allocating commodities and services to the highest-value bidders. It originated from the Latin word "auctus", which means increasing [10]. With a concise chronological history of auctions from ancient times to recent times done by Zajicek in [11]. Auctioning has long since been a means of selling items, particularly goods of considerable valuation, with many interested buyers. Auctioning has been applied to numerous fields and for sales of diverse items. The application has grown further with the advent of the Internet, which has catalysed the development of numerous online public auctioning websites. With regards to borrowing and lending, [11] reviewed numerous bidding strategies employed in bidders in small loan auctions carried out on prosper.com. The authors in [12] developed a model to predict the outcome of a loan request as well as the probability that a loanee would repay a loan. The predictive model was based on data from an online loan auctioning platform. In [13], the authors studied the impact past experience has on bidders when trying to obtain loan from online peer-to-peer bidding platforms. Using data from renrendai.com, the authors were able to draw certain inferences, including the fact that gender and level of education influence the success of bids. Using the data from the same China based renrendai.com platform, Caglayan et al. in [14], showed evidence of herding by lenders in online peer-to-peer platforms.

The Blockchain is a distributed database wherein "blocks" are used to represent transactions. Each block in a chain of blocks (or blockchain) holds openly accessible data about (or replicas of transactions of) other blocks. Though open, the content of each block cannot be unilaterally altered without the "knowledge" of the other blocks [15]. It is for these reason that blockchain is often considered as an open, transparent, distributed but secure platform. The term blockchain has been attributed to "Satoshi Nakamoto" in the article "Bitcoin: A Peer-to-Peer Electronic Cash System" [16]. In the last decade, blockchain has witnessed an unfathomable growth and has continued to attract interest of academic researchers and cooperate organizations alike [17]. It has since been applied to solving problems in the food and medical supply chain, logistics, finance, voting and other domains [18,19]. Its application in smart contracts is of relevance to this work, as we consider transactions between lenders and borrows a sort of pseudo-contract. In this application domain, [20] considered smart contracts using blockchains, while [21] considered the insurance and energy domains. By leveraging on these various concepts, this paper seeks to develop a secure online peer-to-peer money lending platform. Our proposed methodology for achieving this, is discussed in the next section.

3 Methodology

This section describes the methodology of our secure online P2PL platform.

3.1 System Architecture

The software architecture is a 3 layered structure depicted in Fig. 1, with the layers described as follows:

- Lower layer: includes the end user's device(s) and the applications required to access the system. Examples smart phones, and web browsers.
- The Middle layer is the transactions layer and is responsible for performing the backend auctioning operations. For this work, two auctioning models (direct and reverse auctioning) are considered using two algorithms (0/1 knapsack and Perfect Sum).
- The Top layer is the blockchain layer, where transactions are recorded and linked using cryptography.

3.2 The Auction System

A core component of our system is the auctioning mechanism. For this work, both the direct and reverse auction mechanisms are considered to determine which yields the best compromise for lenders and borrowers. In direct auction, borrowers can view all the active lending offers and can bids on any given offer by specifying the amount of money they are willing to borrow and the maximum

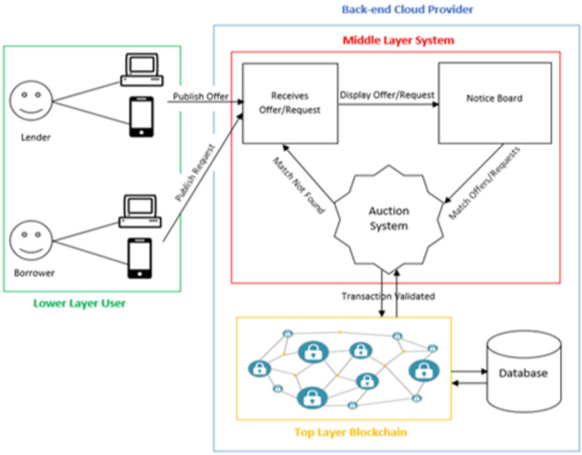


Fig. 1. System architecture

interest rate they can pay. In reverse auction, loan requests and corresponding conditions are displayed on the platform. Lenders can then decide where to place their bids by specifying the amount of money and the minimum interest rate they are willing to offer. A good compromise between lenders and borrowers in an auction can be defined according to the expectation of the party who initiated the auction. Matching requests to offers (or vice versa) in a way that maximizes profit (least compromise), can be considered a "combinatory optimization problem", hence we considered two algorithms to find the optimal solution. These algorithms are 0/1Knapsack and Perfect Sum Algorithm (PSA), both of which are described as follows.

0/1 Knapsack Algorithm. The 0/1 Knapsack Problem is one of the paradigmatic problems in combinatorial optimization. It is one in which a set of items (I1, I2, I3 … In) with given values (V1, V2, V3 … Vn) and weights (W1, W2, W3 … Wn) are available and the aim is to select a subset of the items to maximize the total profit without exceeding a known knapsack capacity (W) [24]. In applying 0/1 knapsack to the direct auctioning model, we considered the loan offers as knapsacks and the borrowers' requests and interest rates as items and their corresponding values. The roles are reversed for the reverse auctioning model. Our implementation of 0/1 knapsack algorithm is depicted in Fig. 2.

Perfect Sum Algorithm (PSA). Given a set S of numbers and a target K, PSA seeks to select all subsets of the given set whose sum equal to the given target [25]. In our application to the auction system, the set S corresponds to the available bid(s), while the targets are the auctioned amounts. The auction system thus seeks to identify all subsets that sum up to the target, it then selects the subset that offers the best interest rate. If no subset is equal to the target, no result is returned. Our implementation of the PSA is shown in Fig. 3.

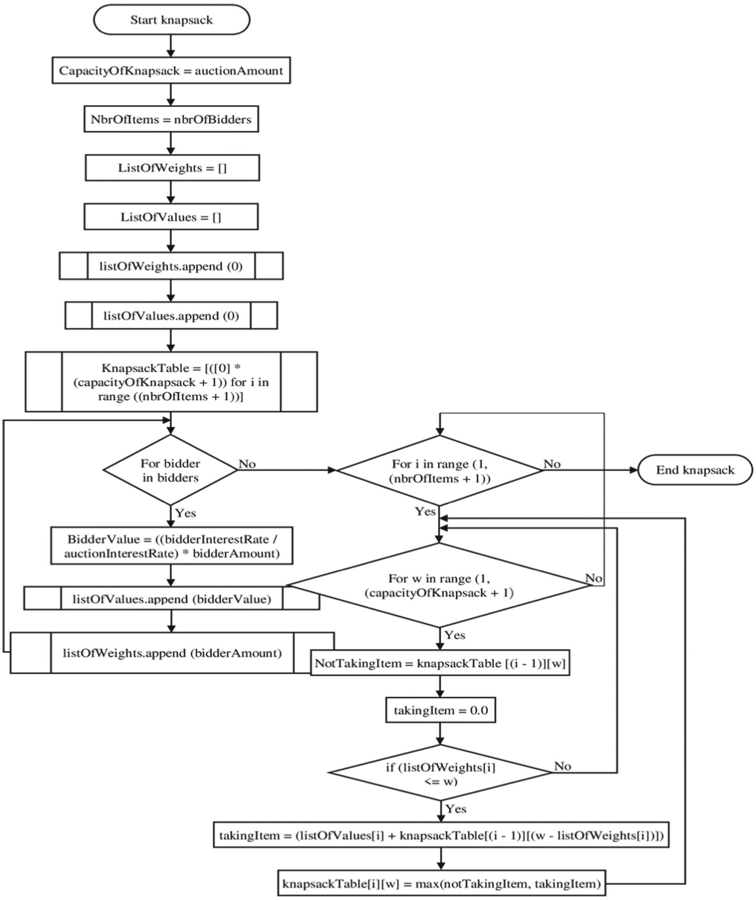

Start knapsack
CapacityOfKnapsack = auctionAmount
NbrOfItems = nbrOfBidders
ListOfWeights = []
ListOfValues = []
listOfWeights.append (0)
listOfValues.append (0)
KnapsackTable = [([0] * (capacityOfKnapsack + 1)) for i in range ((nbrOfItems + 1))]
For bidder in bidders
No
Yes
BidderValue = ((bidderInterestRate / auctionInterestRate) * bidderAmount)
listOfValues.append (bidderValue)
listOfWeights.append (bidderAmount)
For i in range (1, (nbrOfItems + 1))
No
End knapsack
Yes
For w in range (1, (capacityOfKnapsack + 1)
No
Yes
NotTakingItem = knapsackTable [(i - 1)][w]
takingItem = 0.0
if (listOfWeights[i] <= w)
No
Yes
takingItem = (listOfValues[i] + knapsackTable[(i - 1)][(w - listOfWeights[i])])
knapsackTable[i][w] = max(notTakingItem, takingItem)

Fig. 2. 0/1 Knapsack flowchart

3.3 Blockchain and Security

Blockchain is a sequence of blocks, which holds a complete list of transaction records [22]. A Blockchain is a growing list of records, called blocks, which are linked using cryptography. Each block contains a cryptographic hash of the previous block, a timestamp and transaction data. The cryptographic hash of a block is produced from the other field contained in the block which are: Block number, Transaction data, Timestamp and Cryptographic hash of the previous block.

A blockchain is resistant to modification of the data, as it is made up of a chain of data blocks, with each block aware of the information in other blocks. To illustrate this resistance to modification, assume a chain of three blocks 1, 2 and 3. At the onset, each block has information on the other blocks through a hash value. If someone changes the data in block number 2, the hash value of

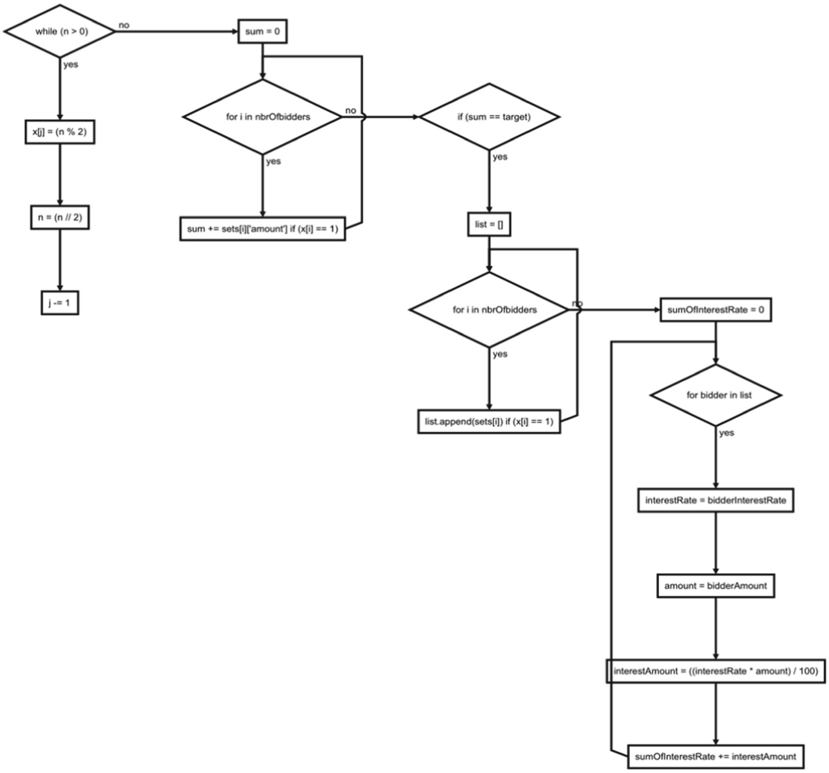


Fig. 3. Flowchart depiction of PSA implementation

block number 2 will also change. Due to this change, the hash value of block 2 contained in block number 3 will no longer be the same as the new hash value of block number 2. In such a case of disparity in hash values, the entire blockchain becomes invalid. Blockchain is not completely impervious to attacks, but for an attack to be successful, there would be a need to change the hash value on every block in the chain. This is particularly difficult if there are numerous blocks in the chain. Furthermore, blockchains often have validation software which check the integrity of the blockchain regularly and at short intervals. By so doing, an attacker has limited time to modify the entire blockchain, hence easy to detect. Beyond this, blockchain also use consensus protocol to ensure security, especially in P2P networks, such as our proposed P2PL platform. The next subsections describe the use of P2P network and consensus protocol to ensure security.

Distributed P2P Network. In securing a distributed P2P Network, the blockchain is replicated among several computer nodes. Transactions can then be sent to any node within the network. Upon successfully completing the transaction, all the nodes "talk" to each other to compare and synchronize the copy of their blockchain [22]. A consensus protocol maintains the communication among the nodes and makes sure that they have a unified version of the blockchain [23]. If an attacker changes the entire data of a blockchain in one node, the corrupted blockchain will be replaced by the correct blockchains available in most nodes when the consensus software verifies the validity of all the blockchains. In this study we used a network composed of 3 computer nodes to illustrate and implement the blockchain. Each node in this network, maintains a copy of the blockchain as illustrated in Fig. 4. The valid transactions after the validation of the consensus protocol will be inserted in the database.

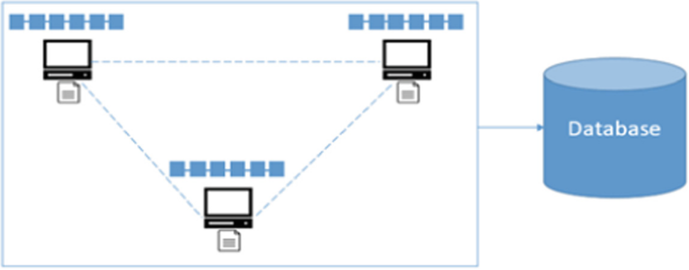


Fig. 4. P2P blockchain model

Consensus Protocol. The consensus protocol is part of the blockchain system. The role of the consensus protocol is to maintain the communication among the nodes and ensure a unified and synchronize version of the blockchain exists across all nodes [23]. The consensus protocol runs in every node to verify and validate its blockchain. It does this by comparing hash values. Subsequently, it compares the blockchain in each node with that of other nodes, to update them or resolve conflicts. We implemented these processes in Python and Figs. 5, 6 show the logical flow of our implementation. Upon completing the verification processes, a new block (containing block number, transaction data, timestamp, and cryptographic hash of the previous block) is created. This is then inserted into blockchain via any of the nodes.

4 Illustrative Case Study

Suppose we have the following records in an auction:

- Borrower: Amount: \$300; Interest Rate: 10% = \$30.
- Lenders bids:
  1. Amount: \$20; Interest Rate: 5% = \$1
  2. Amount: \$280; Interest Rate: 10% = \$28
  3. Amount: \$295; Interest Rate: 3% = \$8.85
  4. Amount: \$300; Interest Rate 15% = \$45

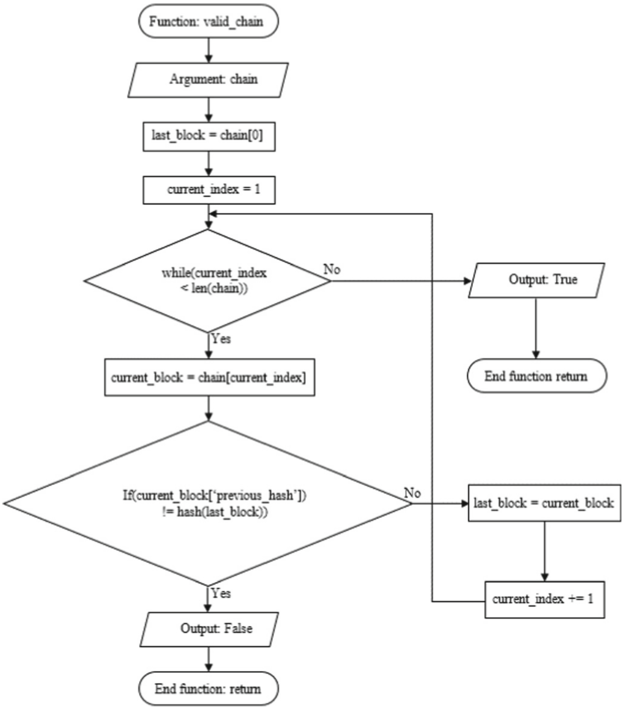


Fig. 5. Consensus protocol - hash value verification

In this auction the best compromise depends on the expectation of the borrower. One borrower can find that the best compromise is to select the Bids 1 and 2 as they produce the full amount that the borrower is requesting for, that is \$300, and with a lower interest value of \$29 (\$1 + \$28) compared to the \$30 they proposed to pay back. Another borrower can find that the best compromise is the third bid. Though it does not provide the full amount requested, the borrower will pay back an interest rate of only \$8.85 which is much lower and more favourable. The borrower might feel happy with the \$5 compromise on the amount requested, that is \$295 instead of the \$300 being requested for. The 0/1 Knapsack algorithm does not always choose the bid(s) that provide the full amount of the auction, but it will choose the bid(s) that maximize the profit. In essence it will select the bid(s) that offer the best combination of amount and interest rate. Table 1 shows the output of the knapsack algorithm for this illustrative example.

In applying PSA to the illustrative example, the set would be made up of {bid1, bid2, bid3, bid4}. The auction system then identifies the subsets {bid1, bid2} and {bid4}, as they sum up to the auction amount (\$300). The subset that will be selected for the auction would thus be {bid1, bid2} because it provides the smaller interest rate (\$1 + \$28 = \$29). This is also less than or equal to the

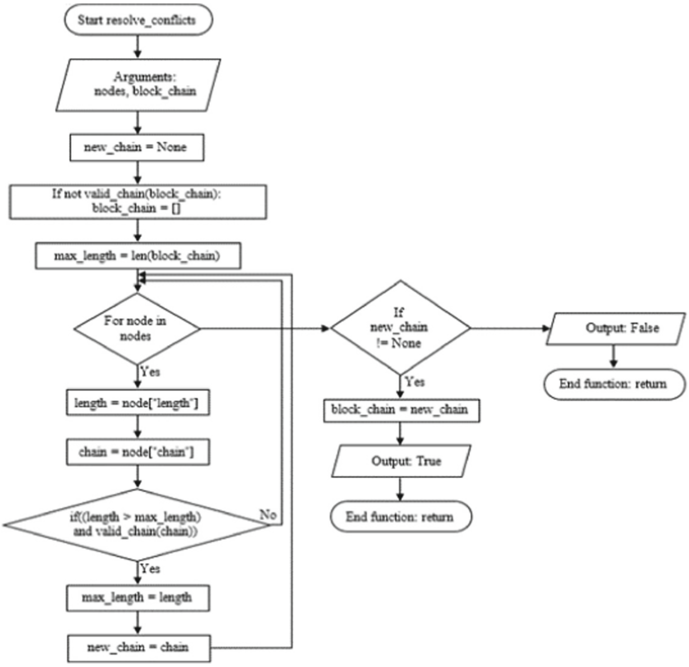


Fig. 6. Consensus protocol - blockchain verification

Table 1. Application of Knapsack algorithm in Auctioning

	Direct Auction	Reverse Auction
Items (Bidders#)	$N = 4$	$N = 4$
Knapsack Capacity (Auction amount)	Lender Amount = \$300	Borrower amount = \$300
Value	$\frac{BidderInterestRate}{AuctionInterestRate} * BidderAmount$ $v_1 = \frac{5}{10} * 20 = 10$ $v_2 = \frac{10}{10} * 280 = 280$ $v_3 = \frac{3}{10} * 295 = 88.5$ $v_4 = \frac{15}{10} * 300 = 450$	$\frac{AuctionInterestRate}{BidderInterestRate} * BidderAmount$ $v_1 = \frac{10}{5} * 20 = 10$ $v_2 = \frac{10}{10} * 280 = 280$ $v_3 = \frac{10}{3} * 295 = 983.3$ $v_4 = \frac{10}{15} * 300 = 200$
Weights (Bidder Amounts)	W1 = 20, W2 = 280, W3 = 295, W4 = 300	W1 = 20, W2 = 280, W3 = 295, W4 = 300
Result and Observation	The algorithm selects bid #4, as it produces the best profit for the lender with the highest interest rate.	Bids 1 & 2 together seem to be a good deal as they provide the full amount requested by the borrower (\$300) while paying \$29 (\$1 +\$28) as interest, which is less that the \$30 that the borrower proposed in the auction. The algorithm will however select Bid #3 because it provides the best profit for the borrower as the borrower will pay back an interest rate of \$8.85 which is much lower yet he receives \$295, which is close the amount requested (\$300).

interest rate proposed by the borrower (\$30), while the second subset provides an interest rate which is higher (\$45). Table 2 shows a comparison of the knapsack and PSA for two bidding scenarios using both reverse and direct auctioning.

Table 2. Comparison of results of knapsack and perfect sum algorithms

	Knapsack	**PSA**
Reverse Auction: Borrower: Amt.=$300; Rate: 10% =$30 Lenders bids: 1. Amt.: $20; Rate: 5% = $1 2. Amt.: $280; Rate: 10% = $28 3. Amt.: $295; Rate: 3% = $8.85 4. Amt.: $300; Rate 15% = $45	Bid(s) selected: 3. Amt.: $295; Rate: 3% = $8.85	Bid(s) selected: 1. Amt.: $20; Rate: 5% = $1 2. Amt.: $280; Rate: 10%=$28
Reverse Auction: Borrower: Amt.=$300; Rate: 10% =$30 Lenders bids: 1. Amt.: $295; Rate: 3% = $8.85 2. Amt.: $300; Rate 15% = $45	Bid(s) selected: 1. Amt.: $295; Rate: 3% = $8.85	No match found
Direct Auction: Lender: Amt.=$300; Rate: 10% =$30 Borrowers bids: 1. Amt.: $20; Rate: 5% = $1 2. Amt.: $280; Rate: 10% = $28 3. Amt.: $295; Rate: 3% = $8.85 4. Amt.: $300; Rate 15% = $45	Bid(s) selected: 4. Amt.: $300; Rate: 15% = $45	Bid(s) selected: 4. Amt.: $300; Rate: 15% = $45
Direct Auction: Lender: Amt.=$300; Rate: 10% =$30 Borrowers bids: 1. Amt.: $20; Rate: 15% = $3 2. Amt.: $280; Rate 10% = $28 3. Amt.: $290; Rate 30% = $87	Bid(s) selected: 3. Amt.: $290; Rate: 30% = $87	Bid(s) selected: 1. Amt.: $20; Rate: 15% = 3$ 2. Amt.: $280; Rate 10% = $28

5 Application Development

In implementing our proposed Trading 4.0 P2PL platform we developed a web application, which is described in this section.

5.1 Requirements

User Requirements. The user requirements specify the actions that the user must be able to perform. These include: ability to register users on the platform. Borrowers should be able to publish loan requests on the platform and specify the amount and maximum interest rate they are willing to pay, while lenders should be able to publish lending offers on the platform by specifying the amount and the minimum interest rate they are willing to accept. In direct auctioning, borrowers must be able to view all the active lending offers and related conditions, while in reverse auction mechanism, the loan request and the corresponding conditions should be visible to lenders. Finally, after the matching of demands and offers, the user that initiated the transaction must be able to fully view and validate the transaction(s).

Functional Requirements. The functional requirements focus on what the system is meant to do. For our secure Cloud-based P2PL system, the functional requirements include i) allowing lenders request for loans and borrowers provide funds through a bidding system; ii) select the best among multiple bids by using direct and reverse auctioning mechanism; iii) ensure security and traceability of transactions using blockchain technology.

Non-functional Requirements. Non-functional requirements focus on the conditions that determine the quality of a system. They focus on how the system works rather than what it does. They often include usability, including user interface and user experience; security, including authentication, and accountability; and maintainability, including customer support, software upgrades etc.

Activity Diagrams. Figure 7 illustrates the interaction between users and the system.

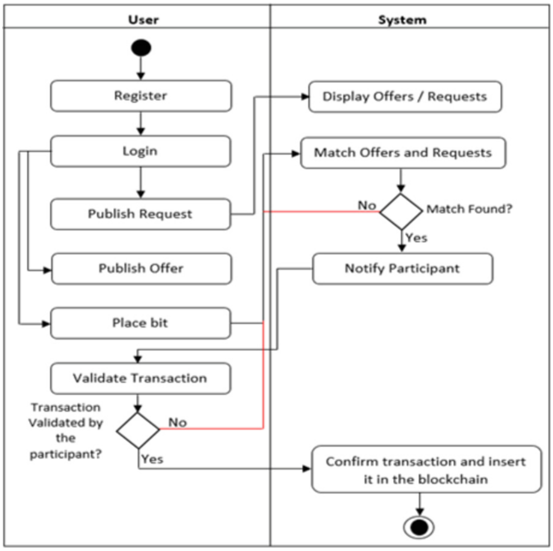

User
System
Register
Login
Publish Request
Publish Offer
Place bit
Validate Transaction
Transaction Validated by the participant?
No
Yes
Display Offers / Requests
Match Offers and Requests
No
Match Found?
Yes
Notify Participant
Confirm transaction and insert it in the blockchain

Fig. 7. Activity diagram of the proposed system

Functional Diagram. The main functionality of the trading platform can be split into 3 categories of sub-functions, and the relationship between them is depicted in Fig. 8. The first group consists of the functions performed by the users, including signing up, and logging into the platform, posting requests, offers, placing bid and validating transactions. The second group represents the functions performed by the auction system. It includes displaying published offers and requests, matching offers with requests, notifying participants and logging the transaction in a blockchain. The last group are performed by the blockchain system. Upon receiving the transaction, a block with a cryptographic hash value is created and inserted into the blockchain of one of the computer nodes, as described in Sect. 3.

5.2 Data Model Design

Due to the large number of users (lenders and borrows), the dynamic nature of the intended trading platform and the speed at which data is generated, data

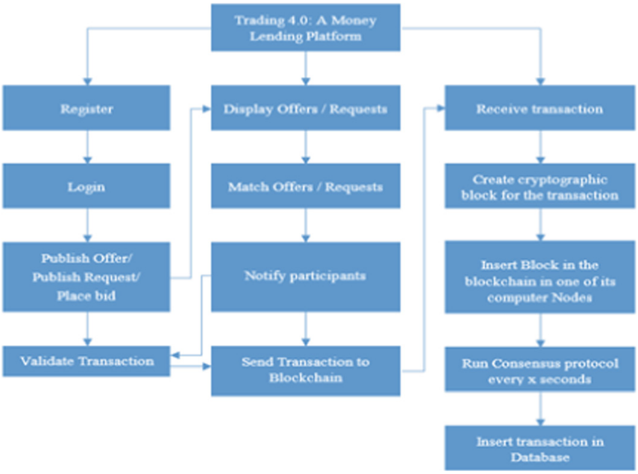

Trading 4.0: A Money Lending Platform
Register
Login
Publish Offer/ Publish Request/ Place bid
Validate Transaction
Display Offers / Requests
Match Offers / Requests
Notify participants
Send Transaction to Blockchain
Receive transaction
Create cryptographic block for the transaction
Insert Block in the blockchain in one of its computer Nodes
Run Consensus protocol every x seconds
Insert transaction in Database

Fig. 8. Functional decomposition

associated with the P2PL platform can be described as Big Data. Big Data is defined as large volume of unstructured data that are too complex for traditional relational data management software to process. Such data are often large in volume, in different formats and generated at a fast pace. Figure 9 shows the architecture of a Big Data orchestration, from data acquisition at the source to analytics and reporting to gain useful insight. In the context of our work, data is acquired in real-time and processed in a streaming manner. Due to enormous size of the data and amount of computation required, processing is carried out on a Cloud computing infrastructure. A Data model is a collection of concepts that can be used to describe the structure of a database [27]. In this work MongoDB (a Cloud based NoSQL database management tool) is used. The corresponding data model and relationship between entities is depicted in Fig. 10.

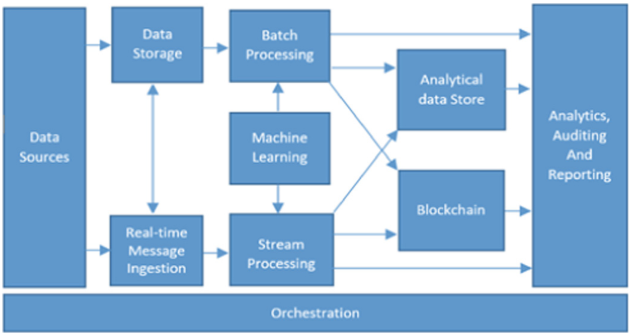


Fig. 9. A Big Data orchestration model [28].

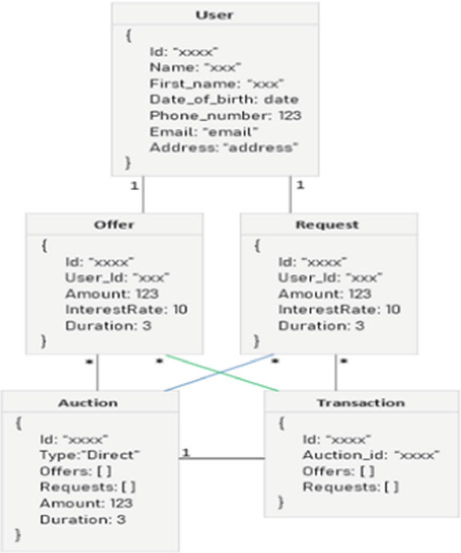


Fig. 10. Data Model design

5.3 Interface Design

The user can interact with the platform using a web browser or a mobile application. Snapshots of the user interfaces are given below.

1. **Publish a new offer or request.** Figure 11 shows the interface through which users can publish a new loan offer (lender) or request (borrower).
2. **General Dashboard.** Once requests and/or offers have been made, they are published by the system on a general dashboard. This is as illustrated in the snapshot in Fig. 12.
3. **Loan Details.** Figure 13 is a snapshot showing the details of an offer or request in an auction as well as the bids available.

5.4 Prototype

Our prototype system was developed using the following components:

Hardware. The front-end application was hosted on a computer running Windows 10, with 8 GB RAM, while and the back end which include the auction system, blockchain and database were hosted on a Cloud infrastructure. At least three different virtual machines are needed for the blockchain implementation. For this work, we used an Openstack deployment made up of 5 systems as the backend. Openstack was chosen because it is a free and opensource platform for creating testbed Cloud infrastructure. The systems had similar specifications and were configured as follows: 2 GB RAM, 4.0 GHz CPU, running windows OS.

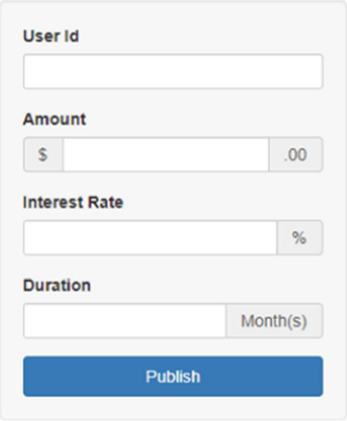
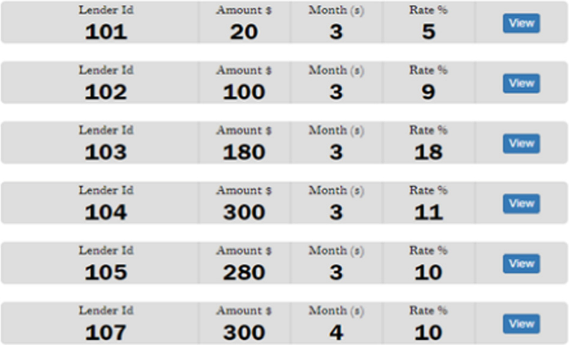


Fig. 11. New offer/request interface



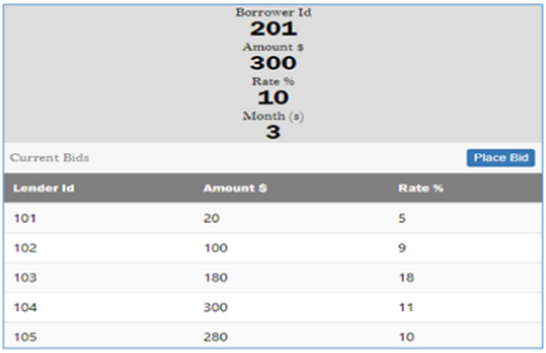
Lender Id	Amount $	Month (s)	Rate %	
101	20	3	5	View
102	100	3	9	View
103	180	3	18	View
104	300	3	11	View
105	280	3	10	View
107	300	4	10	View

Fig. 12. Published offers and request dashboard

The auction system was hosted on machine 1, the database on machine 2, while machines 3 to 5 ran the blockchain system.

Software. The auction system was developed using Python 3.7 and data in JSON format. “PyMongo 3.8.0” was used to create a MongoDB database client, while a web socket was used to maintain the communication between the front and back ends. The front-end was developed using C# and ASP.NET MVC 5. Libraries used included: “Bootstrap3.4.1” for designing the user interface; “Newtonsoft.JSON.12” for converting C# object to JSON format, “WebSocketSharp.1.0.3-rc11” was used to implement the web socket connection between the front and back-end. “Flask-1.1.1” was used to maintain the http communication between the auction system and the blockchain.

Functions, Methods and Classes. The main classes used in the application include: User class, Offer and Request classes, Auction class, Transaction class, Knapsack class (implements the knapsack algorithm to match offers and requests


Borrower Id
201
Amount $
300
Rate %
10
Month (s)
3

Current Bids

Lender Id	Amount $	Rate %
101	20	5
102	100	9
103	180	18
104	300	11
105	280	10

Fig. 13. Detailed view of loan offer/request

in an auction), PerfectSum class (implements the Perfect sum algorithm), and Blockchain class, which contains all the methods needed to implement the blockchain. These methods include: register_node, create_block, resolve_conflicts (implementation of the consensus protocol), and valid_chain (verifies integrity of the blockchain through hashing).

6 Testing

During the developmental stages, several tests were carried out to ensure functionality of the various subcomponents of the system. These limited testing were done mainly on the auction system and the blockchain. Dynamic tests were also conducted to ensure that the publishing of offers, and requests and place bids functioned as expected. The knapsack and Perfect Sum algorithms were tested to make sure that they produce the expected results while matching the offers and requests in an auction. Finally dynamic tests were done to test some functionalities of the blockchain to make sure that new blocks were successfully and inserted into the blockchain for each completed transaction. The rest of this section described the extensive tests carried out on the application.

6.1 Test Design

Functional Testing. Functional testing involves testing the main functions of the software. Each functional requirement can be considered as a test scenario and we carried our several tests for each, as summarized on Table 3.

API Testing. API testing involves testing application programming interfaces (API) directly to determine if they meet the expected functionalities. API testing was carried out on the blockchain using a Postman. Table 4 shows the design of the API testing.

Load Testing. Load testing focuses on the ability of a system to handle increasing levels of loads resulting from transaction requests generated by concurrent

Table 3. Test scenarios

Test scenario	Test cases
Publish loan request	Publish loan request with valid data Publish loan request with invalid data
Publish lending offer	Publish lending offer with valid data Publish lending offer with invalid data
Place bid	Place bid with valid data Place bid with invalid data
Match Offers & requests	Match a lending offer with several borrower bids Match a borrowing request with several offer bids
Blockchain	Insert a new transaction in the blockchain in any node Delete or change blockchain in any node and run consensus protocol

Table 4. API testing

Functionality	Path	Method	Body (form-data field)	Expected result
Insert a new transaction in the blockchain in any node	http://{node}/transaction/confirm	POST	Key: transaction; Value: {transaction value in JSON}	Code: 200 OK. Message: Transaction Confirmed. New transaction inserted in blockchain and Database
Get the blockchain from any node	http://{node}/chain	GET		Code: 200 OK. Display blockchain Data
Manually delete or change blockchain data from any node and Execute consensus protocol on the node	http://{node}/nodes/resolve	GET		Code: 200 OK. Blockchain data restored (execute the following url to check "http://{node}/chain")

users or processes. The result of the load test shown in Fig. 12 was performed on a system with 8 GB of RAM. The result can be improved by distributing the system across multiple servers and implementing a load balancer between the front-end and back-end. This can help balance the user requests load and increase scalability and redundancy of the system. These were out of scope of this project.

6.2 Test Execution

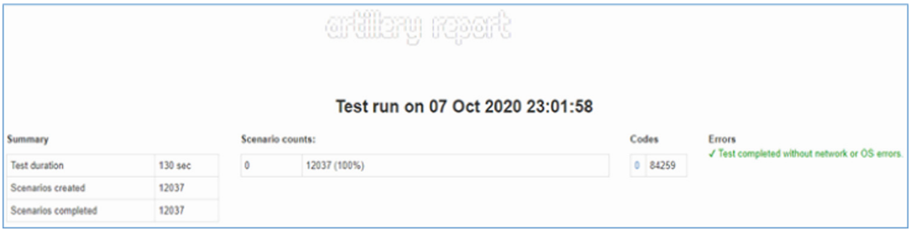
Test Case Execution. The test cases were executed using dynamic testing. Dynamic testing includes executing the software to see if the requirements are executed perfectly. The Table 5 shows some of the important tests carried out.

Load Testing Execution. The test was performed on Artillery.io, which is an advanced load testing toolkit [29]. The test was performed for a duration of 2 min, with a ramp up arrival rate (increase in users) from 50 to 150 virtual users over 2 min. The result of the tests is shown in Fig. 14.

Table 5. Test cases execution

Description	Execution steps
Match a lending offer with several borrower bids 1	1. Publish Offer: Amount \$300; Rate 10%; Duration 3 months 2. Bid Requests: Amount: \$20; Rate: 5% = \$1 Amount: \$280; Rate: 10% = \$28 Amount: \$295; Rate: 3% = \$8.85 Amount: \$300; Rate 15% = \$45 **EXPECTED RESULT/SELECTED BID(S)** Knapsack: Amount: \$300; Rate: 15% = \$45 PerfectSum: Amount: \$300; Rate: 15% = \$45
Match a lending offer with several borrower bids 2	1. Publish Offer: Amount \$300; Rate 10%; Duration 3 months 2. Bid Requests: Amount: \$20; Rate: 15% = \$3 Amount: \$280; Rate: 10% = \$28 Amount: \$290; Rate: 30% = \$87 **EXPECTED RESULT/SELECTED BID(S)** Knapsack: Amount: \$290; Rate: 30% = \$87 Perfect Sum: Amount: \$20; Rate: 15% = \$3 Amount: \$280; Rate: 10% = \$28
Match a borrowing request with several offer bids 1	1. Publish Offer: Amount \$300; Rate 10%; Duration 3 months 2. Bid Requests: Amount: \$20; Rate: 5% = \$1 Amount: \$280; Rate: 10% = \$28 Amount: \$295; Rate: 3% = \$8.85 Amount: \$300; Rate: 15% = \$45 **EXPECTED RESULT/SELECTED BID(S)** Knapsack: Amount: \$295; Rate: 3% = \$8.85 Perfect Sum: Amount: \$20; Rate: 5% = \$1 Amount: \$280; Rate: 10% = \$28
Match a borrowing request with several offer bids 2	1. Publish Offer: Amount \$300; Rate 10%; Duration 3 months 2. Bid Requests: Amount: \$295; Rate: 3% = \$8.85 Amount: \$300; Rate: 15% = \$45 **EXPECTED RESULT/SELECTED BID(S)** Knapsack: Amount: \$295; Rate: 3% = \$8.85 Perfect Sum: No match found

In summary, the load test ran for 130 s, with 12,037 virtual users. In total 84,259 transactions were carried out during the test duration. The system can handle load increase of up to 150 users and/or transactions per second and still meet its performance objectives without errors.



artillery report

Test run on 07 Oct 2020 23:01:58

Summary

Test duration	130 sec
Scenarios created	12037
Scenarios completed	12037

Scenario counts:

0	12037 (100%)

Codes

0	84259

Errors

✓ Test completed without network or OS errors.

Fig. 14. Test cases execution

7 Conclusion

Peer-to-peer lending (P2PL) platforms have come a long way from their early days in the 16th century, though they have maintained their primary use of providing monetary loan in a quick and convenient manner compared to other formal financial institutions. Over the years P2PL platforms have adopted the Internet and are avail customers the opportunity to lend money from the comfort of their house or via their Smart phones. However, with ubiquitous accessibly comes greater security risk, thus frauds are not uncommon in online P2PL systems. In this work we developed a secure money lending platform which uses reverse and direct auctioning to provide the best compromise for lenders. The system uses knapsack and perfect sum algorithms to determine the best match between loan requests and offers. Finally, a multi-nodal blockchain model was used to ensure security and traceability of transactions. Details of the developed prototype application were given, alongside results of requirement tests carried out. Obtained results show that indeed the system was able to offer users best "deal" with regards to lending money.

Though we have developed a functional system, there is still room for improvement. Our current system has a limit on the number of user requests it can handle, in future work a load balancer could be implemented to balance requests loads, increase scalability and redundancy of the system. Furthermore, this work only considered two auctioning mechanisms, more models and a mobile version could be considered in the future. Finally, the application of the auction mechanisms developed to support resource allocation in i) "Internet-of-Things in motion" as suggested in [30,31] and ii) multi-sink fog-based based Cyber physical IoT systems as suggested in [32,33] are other avenues for future work.

References

1. Omarini, A.E.: Peer-to-peer lending: business model analysis and the platform dilemma (2018). Milan
2. Bachmann, A., et al.: Online peer-to-peer lending-a literature review. J. Internet Banking Commer. **16**(2), 1 (2011)
3. www.Prosper.com. Accessed 07 Apr 2021

4. Frerichs, A., Schumann, M.: Peer to peer banking-state of the art. Universität Göttingen. Institut für Wirtschaftsinformatik. Arbeitsbericht (2008)
5. Ashta, A., Assadi, D.: An analysis of European online micro-lending websites. Cahiers du CEREN. **4**(29), 147–60 (2009)
6. Galloway, I.: Peer-to-peer lending and community development finance. Commun. Investments **21**(3), 19–23 (2009)
7. Collier, B.C., Hampshire, R.: Sending mixed signals: multilevel reputation effects in peer-to-peer lending markets. In: Proceedings of the 2010 ACM Conference on Computer Supported Cooperative Work, pp. 197–206, 6 February 2010
8. Freedman, S., Jin, G.Z.: Do social networks solve information problems for peer-to-peer lending. Evidence from Prosper.com **11**, 19 (2008)
9. Klafft, M.: Peer to peer lending: auctioning microcredits over the internet. In: Agarwal, A., Khurana, R. (eds.) Proceedings of the International Conference on Info Systems, Technology and Management. IMT, Dubai, 6 March 2008
10. Bhamare, M., Wagholi, B., Chame, A., et al.: Online auction system. IJSRD **7**(4), 18–21 (2019)
11. Zajicek, C.: The History of Auctions: from Ancient Greece to Online Houses. The Telegraph. https://www.telegraph.co.uk/art/online-auctions/history-of-auctions/. Accessed 16 June 2021
12. Puro, L., Teich, J.E., Wallenius, H., Wallenius, J.: Bidding strategies for real-life small loan auctions. Decis. Support Syst. **51**(1), 31–41 (2011)
13. Ceyhan, S., Shi, X., Leskovec, J.: Dynamics of bidding in a P2P lending service: effects of herding and predicting loan success. In: Proceedings of the 20th International Conference on WWW, pp. 547–556, 28 March 2011
14. Li, X., Jiang, X., Yang, Y.: Learning by P2P bidding. Asia-Pacific J. Account. Econ. **4**, 1–24 (2021)
15. Aslam, U., Manzoor, S., Babar, M.I.: Systematic literature review of blockchain in requisites of big data and its applications. Education **2015** (2012)
16. Nakamoto, S.: Bitcoin: a peer-to-peer electronic cash system. Manubot, November 2019
17. Anascavage, R., Davis, N.: Blockchain technology: a literature review, SSRN 3173406, 3 May 2018
18. Tasatanattakool, P., Techapanupreeda, C.: Blockchain: challenges and applications. In: International Conference on Information Networking (ICOIN), pp. 473–475. IEEE, 10 January 2018
19. Al-Jaroodi, J., Mohamed, N.: Industrial applications of blockchain. In: IEEE 9th Annual Computing, Communication Workshop and Conference (CCWC), pp. 0550–0555. IEEE, 7 January 2019
20. Luu, L., Chu, D.H., Olickel, H., Saxena, P., Hobor, A.: Making smart contracts smarter. In: Proceedings of the 2016 ACM SIGSAC on Computer and Communications Security, pp. 254–269, 24 October 2016
21. Cohn, A., West, T., Parker, C.: Smart after all: blockchain, smart contracts, parametric insurance, and smart energy grids. Georgetown Law Technol. Rev. **1**(2), 273–304 (2017)
22. Zheng, Z., Xie, S., Dai, H., Chen, X., Wang, H.: An overview of blockchain technology: architecture, consensus, and future trends. In: 2017 IEEE International Congress on Big Data (BigData Congress), pp. 557–564. IEEE, 25 June 2017
23. Mackenzie, B., Ferguson, R.I., Bellekens, X.: An assessment of blockchain consensus protocols for the Internet of Things. In: 2018 International Conference on Internet of Things, Embedded Systems and Communications (IINTEC), pp. 183–190. IEEE, 20 December 2018

24. Furini, F., Monaci, M., Traversi, E.: Exact approaches for the Knapsack problem with setups. Comput. Oper. Res. **1**(90), 208–220 (2018)
25. Perfect Sum. www.geeksforgeeks.org/perfect-sum-problem/?ref=leftbar-rightbar. Accessed 29 June 2020
26. Hulme, M.K., Wright, C.: Internet based social lending: past, present and future. Soc. Futures Observatory **11**, 1–15 (2006)
27. IDC.com. www.idc-online.com/technical-references/.../Data-Model.pdf. Accessed 29 June 2020
28. Pietersen, K., Bagula, A., Gaffor, Z., Jovanovic, N.: Big Data Infrastructure for Transboundary Aquifer Systems Analytics. Sustainable Water Partnership. https://www.swpwater.org/wp-content/.../20200420-UWC-TBA-data-storage-needs-1.pdf. Accessed 16 June 2021
29. www.Artillery.io. Accessed 30 June 2020
30. Ismail, A., Bagula, A., Tuyishimire, E.: Internet-of-things in motion: a UAV coalition model for remote sensing in smart cities. Sensors **18**(7), 2184 (2018)
31. Maluleke, H., Bagula, A., Ajayi, O.: Efficient airborne network clustering for 5G backhauling and fronthauling. In: 16th IEEE International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob), pp. 99–104 (2020)
32. Bagula, A., Erasmus, Z.: IoT emulation with Cooja. In: ICTP-IoT Workshop (2015)
33. Kun, M., Bagula, A., Nyirenda, C., Ajayi, O.: An IoT-based Fog computing model. Sensors **19**(12), 2783 (2019)