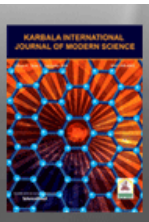

Karbala International Journal of Modern Science

Volume 7 | Issue 4 Article 5

An Online E-voting System based on an Adaptive Ledger with Singular Value Decomposition Technique

Rihab Habeeb Sahib
Department of Software, College of Information Technology, University of Babylon,
rihab.sahib@student.uobabylon.edu.iq

Prof. Dr. Eman Salih Al-Shamery
Department of Software, College of Information Technology, University of Babylon

Recommended Citation

Sahib, Rihab Habeeb and Al-Shamery, Prof. Dr. Eman Salih (2021) "An Online E-voting System based on an Adaptive Ledger with Singular Value Decomposition Technique," *Karbala International Journal of Modern Science*: Vol. 7 : Iss. 4 , Article 5.
Available at: https://doi.org/10.33640/2405-609X.3156

An Online E-voting System based on an Adaptive Ledger with Singular Value Decomposition Technique

Abstract

Regular E-voting systems for elections may count the votes in less time,less cost,save the privacy of citizens,but still considered risky as votes can be tampered.E-voting systems based on a network distributed ledger show fast results,more trusted,save privacy,cannot be tampered,and distributed in which no central organization controls the system.This paper illustrate an e-voting system to solve the challenge of a massive ledger that is distributed among network-nodes using a data reduction technique as a security-matching-tool,singular value decomposition(SVD) that handle a copy of election results in another form and matched with the SQL-database results to announce a successful election-event representing a transparency-powerful-secured-system

Keywords

Election, E-voting, Singular value decomposition, Distributed ledger, hash function, Cloud, Network

1. Introduction

The security of elections is an important feature for the national security in every country, as it is a matter of national security [1]. The first voting system was based on paper and pen. Intending to decrease the costs related to this event, the traditional elections are replaced with an electronic system. This is considered an important issue to limit manipulation and having the voting system verifiable and traceable, Electronic voting systems offer advantages over manual voting [2].

The online voting system does not work well enough due to the lack of legitimacy of votes. Hence, to construct an application that is difficult to attack and data cannot be manipulated the "blockchain technology" is used, which is sometimes referred to as "distributed ledger technology" (DLT) in an E-voting application. Blockchain stores data in a decentralized way and is considered a revolutionary technology for many future applications [3].

Blockchain has been used for the first time in transactions of Bitcoin database systems [4]. It is distributed, transparent and an unchangeable ledger [5]. It consists of blocks that are linked to one another in sequence order. Each block is linked to the previous block by containing the previous hash code for that block except for the first block that will have a hash value of zeros [6]. In the case of trying to attack the system, the database owned by attackers who want to hack the system is different from the database owned by concerned users. By that, the existing database on the users' nodes is not valid. As the data corresponding to the results on each node is distributed under the blockchain permission protocol [7]. That was the first use of blockchain technology before using it in applications like e-voting systems.

Since blockchain for large communities can be a difficult and slow process due to the block miming mechanism, we introduce a web application for an e-voting system that meets the properties needed for a voting process based on an adaptive distributed ledger with the singular value decomposition technique (SVD) as a dimensionality reduction tool and a matching tool that is used to create an immediate copy of the results and store the outputs of SVD in distributed network node using cloud services. Also, we introduce a full example of the voting procedure till the end to illustrate the system step by step.

Section 2 is a literature review of related works compared to our work with tables illustrating the features of each e-voting system. Section 3 shows theoretical concepts of techniques and methods that are employed in our e-voting system. Section 4 explains the phases and stages of the proposed system. An example of how the SVD technique is used as a dimensionality reduction method is shown in Section 5. The web application and interfaces of the e-voting system are shown in Section 6 and show how the SVD technique is used to match the results of the ledger and SQL database results which is our main contribution followed by the conclusions in Section 7.

2. Literature review

In reference [8], Yasmine (2013) proposed an internet voting system in which the system generates a unique password (10 digits) for each voter. On Election Day, voters will enter their information and password on the voters' registration form. If the information is correct; the voters enter the form where they can select a specific political group or candidate to vote for, else if any incorrect information were added, the user will not be allowed to vote. Only an eligible voter can cast a vote once and the whole process is controlled by the administrators of the central organization. The system is controlled by a central organization and all data and results are stored locally.

In our proposed system, the voters enter their information during the day of the election event and are allowed to vote only once. The database of voters and candidates is stored in a cloud service with security features. Also, the file (ledger) that contains the votes is distributed among several network nodes to ensure that the system is decentralized and no particular party controls the ledger of results using the SVD data mining technique aiming to use it as a matching tool to check the transparency of the results and as a dimensionality reduction tool.

In reference [9], Aakash (2020) proposed an online voting system in which the back-end server takes care of authenticating the users and maintaining necessary details using the XAMPP server. The user interface at the server's end enables creating the election on behalf of the users. Also, the admin can log in to his/her account and can manage the whole voting process by adding a new election, generating an ID for the user, verifying the users, and generating results.

https://doi.org/10.33640/2405-609X.3156

Our proposed system handles the whole system dynamically in real-time. The administrator starts the system by opening a connection to the storage services. Every time a group of voters votes (composing a block), the SVD technique is applied on each block to generate another copy of the immediate results in another form and distribute the ledger of results in separated network nodes. The admin has no control over the system as it is running. Also, there will be a copy of the results that are then matched with the SQL database of results to rate the success and transparency of the system.

Authors in Ref. [10] proposed an e-voting system using an Android platform that allows the voter to cast the vote with no need to reach the polling booth. The application has an authentication process to avoid any attempt of fraud using the one-time password (OTP) that will be given to the voter after he\she login by giving the name, ID, and location to vote, then the voter must enter the number of mobile for generating the OTP that will be submitted to consider the vote as valid. The firebase server is used to authenticate the number of mobiles and sends back an OTP to the voter.

After the end of the voting process, the results will be shown within seconds. All the counts for the casted votes are encrypted using the Advanced Encryption Standard 256 (AES256) algorithm and saved in the database to avoid any control by other than the administrator that will decrypt the results then announce them.

Our proposed system is a web application that is suitable for all citizens using any device connected to the internet. The results can be viewed at any time and a group of votes is structured in blocks secured by the hash function SHA 256. Also, the whole system has its data and files encrypted using a cloud encryption service and firewall as a service that secures the network. Our system can check the results whether they are manipulated or not by employing SVD as a matching tool that checks the transparency of the election event by converting the results immediately into another form distributed in separated servers and match them at the end with the SQL database of results.

In reference [11], Ahmed (2017) proposed a design for an electronic voting system that mainly covers four requirements, authentication where only eligible citizens are allowed to vote, anonymity where the voter has to remain anonymous, accuracy in which every vote is unique and counted, and the verifiability to make sure all votes are counted correctly.

In [11], the candidate will be the first transaction added to the block, containing the name of the candidate and considered the base block, in which votes for that candidate are placed on top of the base node and will not be counted as a vote. Each vote will get recorded and the blockchain will be updated. Security is ensured by having the previous voter's information in the block and since all blocks are connected, it would be easy to find out if any of the blocks is compromised. Each vote is sent to a specific candidate's node, and the nodes then add the vote to the blockchain. To ensure decentralization, the voting system will have a node in each area where the election is held. Limitations in this system are the ability to change and cast a vote by a hacker using malicious software already installed on the voter's device.

Our proposed system implements the mentioned requirements where only eligible citizens are allowed to vote depending on the valid ID saved in the database that will return the information of the citizen ensuring his\her personal information. The voter remains anonymous as the whole block containing the IDs of voters will be hashed using SHA256. Every vote is unique and counted in a ledger that will be distributed by using the SVD technique. Since blockchain is considered a risk for e-voting systems, Our proposed system takes advantage of the distributed ledger technology as a suitable mechanism for an e-voting system.

Authors (2020) in Ref. [12], proposed an online e-voting system in which casting a ballot and saving information is done using a cloud server for consuming time and vote from anywhere through an online web-based voting system. Using a cloud over the conventional electronic casting a ballot framework will advance into another idea of incorporated approach for casting a ballot framework for better precision and less weakness of the votes in the decision. By using a cloud there is the capacity to offer to cast a ballot framework to voters both in the nation and outside through a web-based casting a ballot. The voter utilizes the username and secret key for login and cast a ballot, the outcome can be seen simply after the end date.

Our proposed system uses the Azure cloud to save a database that contains citizens' information and files reducing the size of the general ledger we are dealing with based on the SVD technique. At the same time, speeding up the search needed for each voter to enter the system using his\her information, which is implemented by the cloud engine. Also, using the cloud services to deal with the distributed files that are extracted from SVD.

Authors (2020) in Ref. [13], proposed a decentralized e-voting system using blockchain for elections to provide transparency and flexibility. The system

Table 1
Criteria of e-voting systems for the related works and the proposed system.

	Criteria features for the e-voting systems					
Reference and year	Transparency	Real-time	Trusted	Secured	Authentication	Anonymity
[8]: 2013	X	X	X	X	√	√
[9]: 2020	X	X	X	X	√	√
[10]: 2020	X	√	X	√	√	√
[11]: 2017	√	√	√	√	√	√
[12]: 2020	√	X	√	√	√	√
[13]: 2020	√	X	√	√	√	√
The proposed system	√	√	√	√	√	√

consists of three modules, the user validation model that uses biometric (fingerprint) and other information (name, gender, address) to verify the user. At the time of voting, all this information is hashed using the Message-Digest algorithm (MD5) and checked with the election database.

The second model is the dynamic ballot loading model, it relies on the residence location of citizens and loaded in the ballot. The voters will have to go to the nearest polling booth. The third model is the Acknowledgment model after casting their vote. In this model a transaction ID is created for each voter after casting the vote in a blockchain network, it is given as acknowledgment to the user. The Election Authority (EA) has the role of creating a vote, not allowing more than one vote for each voter, and pay the fees of casting a vote for the bitcoin address created within the backend. The results are published after tallying the votes which are made by EA.

The candidate must register with his\her information to create an ID for that candidate, this candidate ID will be used by the voter to vote for him\her. The voting fees for the address of bitcoin for the voters is zero as soon as the voter casts their vote, so there is no chance to repeat a vote more than once.

Our proposed system distributes the ledger of results based on SVD to separated storage nodes. And no need for the voter to go to the nearest ballot station, as our system is a web application that can be reached by any device. Since using blockchain can lead to some drawbacks like scalability and latency that are mentioned by Saker (2020), the system is applied to small-scale organizations, avoiding the extra throughput that is needed for the validation process of each block. Our system leverages the features of a distributed ledger and employs the SVD technique as a tool used for ensuring transparency in real-time.

Table 1 shows the criteria of e-voting systems features for the related work and our proposed e-voting system, each has its implementations that achieve the desired feature. Although, some features can have a lack of performance due to challenges that may face the systems such as, it is almost impossible to guarantee that any system can be fully secured due to the development of malicious software and technology.

Additionally, Table 2 illustrates the mechanisms, techniques, and functions used for each e-voting system including our proposed system.

3. Theoretical concepts

Our e-voting system employs several techniques and methods that should be introduced generally as important concepts as follows:

Table 2
Technologies, techniques, and functions for the e-voting systems for the related works and the proposed system.

	Technologies, techniques, and functions for the e-voting systems							
Reference and year	Smart phone App.	Web App.	Blockchain Tech.	DLT Tech.	Data Mining Tech.	Hash Function	Cryptography Functions	Cloud services and servers
[8]: 2013	X	√	X	X	X	X	X	X
[9]: 2020	X	√	X	X	X	X	X	√
[10]: 2020	√	X	X	X	X	X	√	√
[11]: 2017	X	√	√	√	X	√	X	X
[12]: 2020	X	√	X	X	X	X	√	√
[13]: 2020	X	X	√	√	X	√	X	X
The proposed system	X	√	X	√	√	√	X	√

3.1. Distributed Ledger Technology (DLT)

The term "Distributed Ledger Technology" (DLT) has been raised to describe blockchain in the finance industry. Sometimes, DLT and blockchain are used interchangeably. However, this is not correct, it is how the term has been used recently, especially in the finance field. DLT is now an active and growing area of research in many fields. From a financial sector point of view, DLTs are shared and used between known participants and permission blockchains. Generally, DLT is used as a shared database, with known and verified participants. They do not require mining to secure the ledger nor have a cryptocurrency [14]. The ledger is a database that stores the transactions, replicated, shared, and synchronized among the nodes of the blockchain network [15]. Our proposed system employs DLT by having a copy of the election results distributed among network nodes.

3.2. Singular value decomposition (SVD)

The singular value decomposition technique is a way to analyze a matrix that results in a low-dimensional representation of a high-dimensional matrix. It makes it easy to eliminate the less important parts of that representation to produce an approximate representation with any desired number of dimensions [16]. It is a statistical form of principal component analysis (PCA). SVD is considered one of the most widely used mathematical formalism/decomposition in data mining, machine learning, pattern recognition, artificial intelligence, and other fields. Mathematically, SVD can be seen as the best low-rank approximation to a rectangle matrix. The left and right singular vectors are mutually orthogonal and provide the orthogonal basis for row and column subspaces. When the data matrix is centered as in most statistical analyses, the singular vectors become eigenvectors of the covariance matrix and provide mutually uncorrelated/de-correlated subspaces which are much easier to use for statistical analysis [17].

SVD is an efficient and stable way of separating the structure into a collection of linearly distinct parts, each of which has its contribution to energy [18]. Any 2-dimensional matrix A of size $(m \times n)$ can be factorized into three matrices. Where m represents the rows, n represents the columns, and $m \geq n$ [19,20]. The result of multiplying the three matrices is approximately equal to the original matrix A, as explained in equations (1)–(3) [20,21].

$$A = USV^{T} \tag{1}$$

$$= \begin{bmatrix} u_{1,1} & \cdots & u_{1,n} \\ u_{2,1} & \cdots & u_{2,n} \\ \vdots & & \vdots \\ \vdots & \cdots & \vdots \\ u_{n,1} & & u_{n,n} \end{bmatrix} \begin{bmatrix} \sigma_{1,1} & 0 & \cdots & 0 \\ 0 & \sigma_{2,2} & \cdots & 0 \\ \vdots & & & \vdots \\ \vdots & & \cdots & \vdots \\ 0 & & & \sigma_{n,n} \end{bmatrix}$$

$$\times \begin{bmatrix} V_{1,1} & \cdots & V_{1,n} \\ V_{2,1} & \cdots & V_{2,n} \\ \vdots & & \vdots \\ \vdots & \cdots & \vdots \\ V_{n,1} & & V_{n,n} \end{bmatrix} \tag{2}$$

$$= \sum_{i=1}^{n} \sigma_i u_i v_i^{T} \tag{3}$$

The huge matrix can be represented as a multiplication of three small matrices such as shown in Fig. 1. This will be our main use of SVD in our system to decompose the ledger as small as possible without

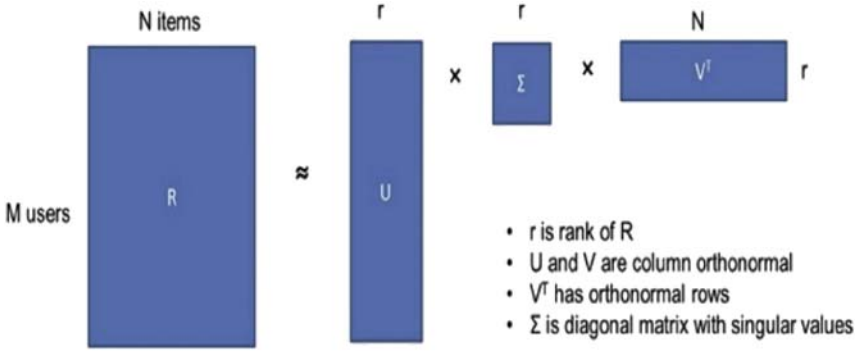

Fig. 1. Size reduction of SVD decomposition [17].

losing important data that will be represented as binary data defining 1 for a vote and 0 otherwise, which will be transformed into real numbers and distributed among three nodes in the network, in other words, each node is cloud storage for each matrix output of SVD.

The property of SVD to provide the closest rank-l approximation for a matrix X can be used in wide-ranging applications. By setting the small singular values to zero, we can obtain matrix approximations whose rank equals the number of remaining singular values. Very good approximations can often be obtained using only a small number of terms to decompose the matrix instead of working with data of large size [22].

Our e-voting system uses the SVD technique to deal with a matrix of size $m*n$ where m represents the candidates (rows) and n represents the voters (columns). Initially, the matrix is of zero values. Every time a voter votes for a candidate, the zero corresponding to the desired candidate is replaced by the value 1. SVD is applied to this matrix to extract its three outputs that are singular matrices (U, S, and V) as explained previously, and distribute them to the network nodes. By multiplying the three matrices we construct the ledger (matrix) of results that are matched at the end of the election event with the SQL database of results. This process occurs to each block of the transaction (votes) that is explained in detail at the end of section 7.

3.3. Hash functions

NIST "The National Institute of Standards and Technology" standard specifies the adoption of secure hash algorithms such as "SHA-1, SHA-224, SHA-256, SHA-384 and SHA-512" [23]. Hash Function algorithms are used to produce the message digest during data transmission. Therefore, it becomes an essential tool for embedded security in different applications. A hash function takes a variable-length message as an input resulting in an output of fixed length. It is a one-way function that makes it difficult to invert a hash value to a message input. Also, it is infeasible to find a message that gives the same hash value. Any simple change in any block will lead to an invalid block because each block is related to the previous block by the proposed hash function [24]. The hash value is generated using the "Secure Hash Algorithm" (SHA256) to generate an almost idiosyncratic fixed-size 256-bit hash. The SHA256 takes input for any size of plaintext and encrypts it to a 256-bit binary value, in a strictly one-way function. Fig. 2 shows the logic of SHA256 [25].

3.4. Microsoft Azure cloud service

Azure is one of the largest cloud computing platforms that offer many services in the cloud as shown in Fig. 3. Service providers can expand their services in areas where they have existing infrastructure and add new services without major infrastructure investments. Software engineers can use Azure to create, configure and manage web applications and web services without major capital expenditures and prepare the resources needed quickly and efficiently [26].

In Cloud we only need an Internet connection, to access the information stored from any geographic location, at any time, and also to post/transmit information if necessary. Social platforms, store and process big data in optimal response time [28].

Azure Web Sites are flexible, scalable, and secure platform components that can be used for building web applications that run client applications and web services. The second option for deploying web services on Azure is by using Azure Cloud Services. Azure Web Sites own an easily useable self-service portal with a gallery of different templates for web solutions that can be used without having to create and update virtual machines, install and configure Internet Information Server (IIS), etc. [29].

Azure Cloud Services offer an opportunity to set up and work in the cloud application level where Azure takes care of all details including security, load balancing, and monitoring. All these features in combination provide continuous availability of applications. In other words, Azure Cloud Services are containers of hosted applications and web services [30].

Our proposed system used Azure storage services to store the matrices of SVD (U, S, and V) as storage nodes in three separated servers around the world. The queue storage is used to hold transactions (votes) before they are managed by SVD. Also, The databases of the candidates and voters are secured within the azure SQL database service.

4. The proposed system

The proposed system as shown in Fig. 4 is based on an adaptive distributed ledger, by employing the singular value decomposition technique (SVD), in which a copy of the SQL database results is processed and handled by SVD. The output of SVD is three matrices (left singular matrix U, singular value matrix S, and right singular matrix V). These matrices store a copy of the results each time a block of transactions is created in three separated places and so on for all the following

blocks. At the end of elections, the last ledger performed by SVD is matched with the SQL database to ensure no manipulation occurs.

The system consists of four phases, each contains many stages that also contain many steps. The first phase is the preprocessing phase which includes preparing the lists of candidates and voters, preparing the distributed network nodes for storing and processing data, and creating the adaptive ledger that holds the results based on the SVD technique.

The second phase is the confirmation process where each citizen is allowed to vote based on the international ID card and cast a vote for only one time.

The third phase is the e-voting phase that includes adding transactions (votes) to the block, each block has an ID number that is the summation of all IDs for citizens in the block, a timestamp showing the time and date of creating that block, and secured by SHA256 hash function, also each block will contain the hash value of the previous block except for the first block that will be a value of zeros.

The SQL database that contains the results will be transformed to another form using SVD creating an adaptive SVD ledger that distributes its output (matrices of real numbers) in three separated servers.

Using SVD as a security tool for matching the results is the main contribution of our system. Additionally, SVD is known as a dimensionality reduction method that aims to deal with sparse data, which is useful in the case of having many zeros in the ledger.

The final phase is the resulting phase, in which the SQL database is matched with the retrieved SVD adaptive ledger to ensure the success of the election event.

During the event, each citizen has the right to view the results online for transparency without knowing who voted for which candidate.

The phases of the proposed system are explained as follow:

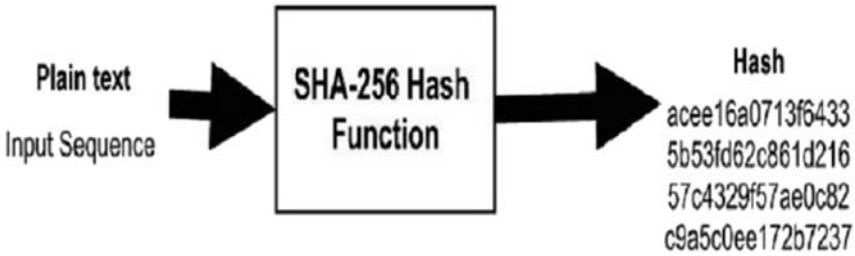

Fig. 2. Basic function of SHA256.

Fig. 3. Microsoft Azure cloud services [27].

4.1. Preprocessing phase

The preprocessing phase consists of the following stages:

4.1.1. Preparing the list of candidates and voters

The database of candidates includes the candidates' names, IDs, counts, and photos. While the database of voters includes the citizen's names, IDs, and state

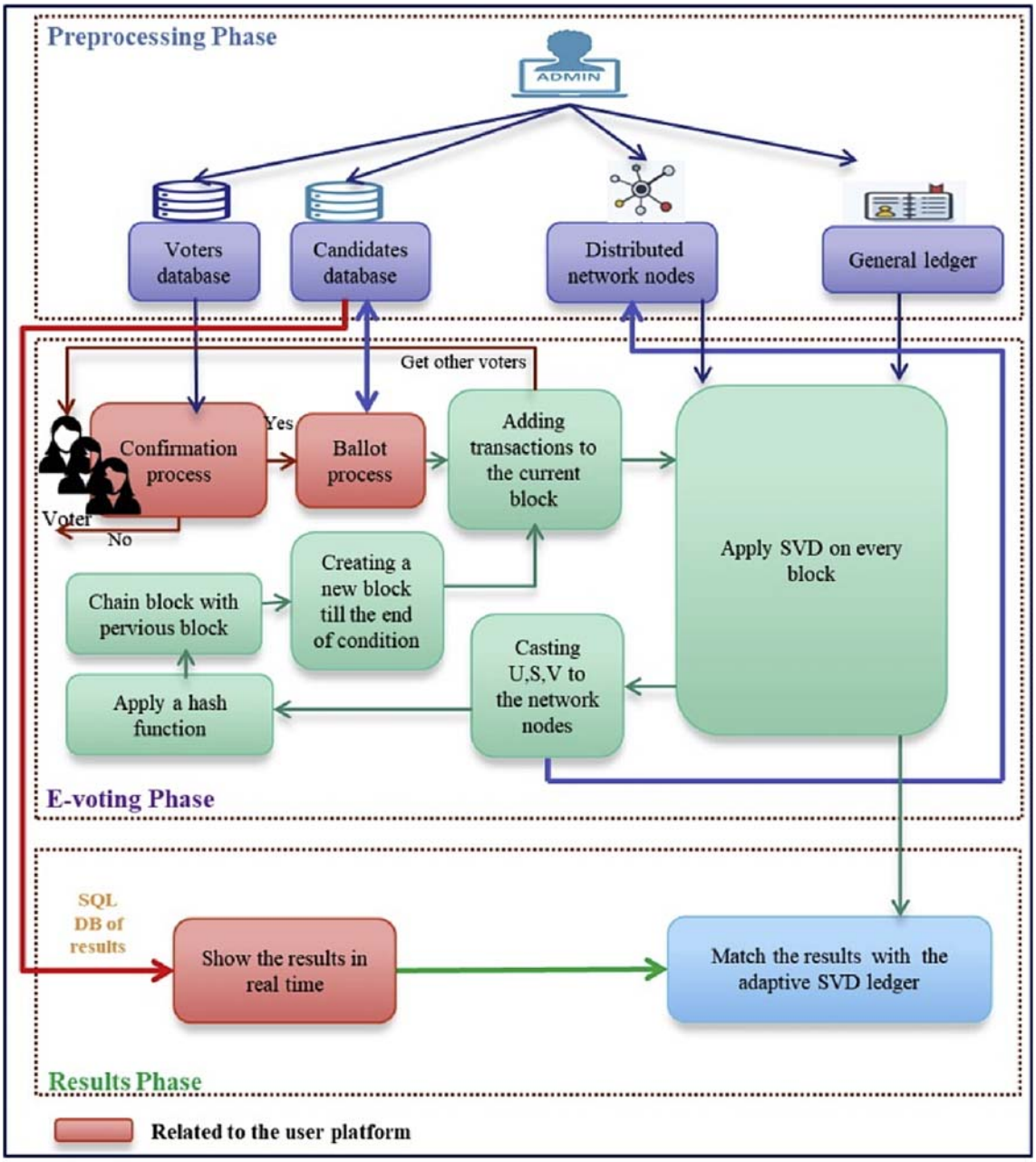

Fig. 4. The block diagram of the proposed e-voting system with the SVD technique.

which is indicated by 1 for a vote or zero otherwise. These databases are managed using a cloud SQL database service.

4.1.2. Building the distributed network nodes

Setting the distributed network nodes in three separated places, they are responsible for storing the outputs of SVD ledger receiving and retrieving them at the end of election to match the SVD adaptive ledger with the SQL DB of the results. Thus we will have three distributed nodes. E-voting phase explain in details how these nodes function.

4.1.3. Creating the general ledger

The general ledger initially contains values of zeros. It represents the number of votes for each candidate (rows refer to candidates and columns refer to voters). A citizen is allowed to vote only once. In other words, each column should contain many zeros and at most only a single 1 representing a vote. When a block is generated, counts for transactions (votes) in that block are added to the SQL database of results. At the same time, the general ledger is updated and the SVD technique is applied to this ledger to transform the results into another form, distribute them and finally retrieve them at the end of the event as a security tool to match the SVD adaptive ledger with the SQL database of results.

4.2. Confirmation process

The confirmation process includes checking the ID of the voter, ensuring that the voter votes only once and cannot change his/her vote based on a valid ID. The confirmation process flow chart is explained in detail as shown in Fig. 5.

4.3. E-voting phase

The E-voting phase is the part where the SVD data reduction technique is applied to the general ledger. In this phase the vote is added as a transaction in a block, retrieving data from the distributed storage nodes that contain U, S, and V to obtain the adaptive SVD ledger,

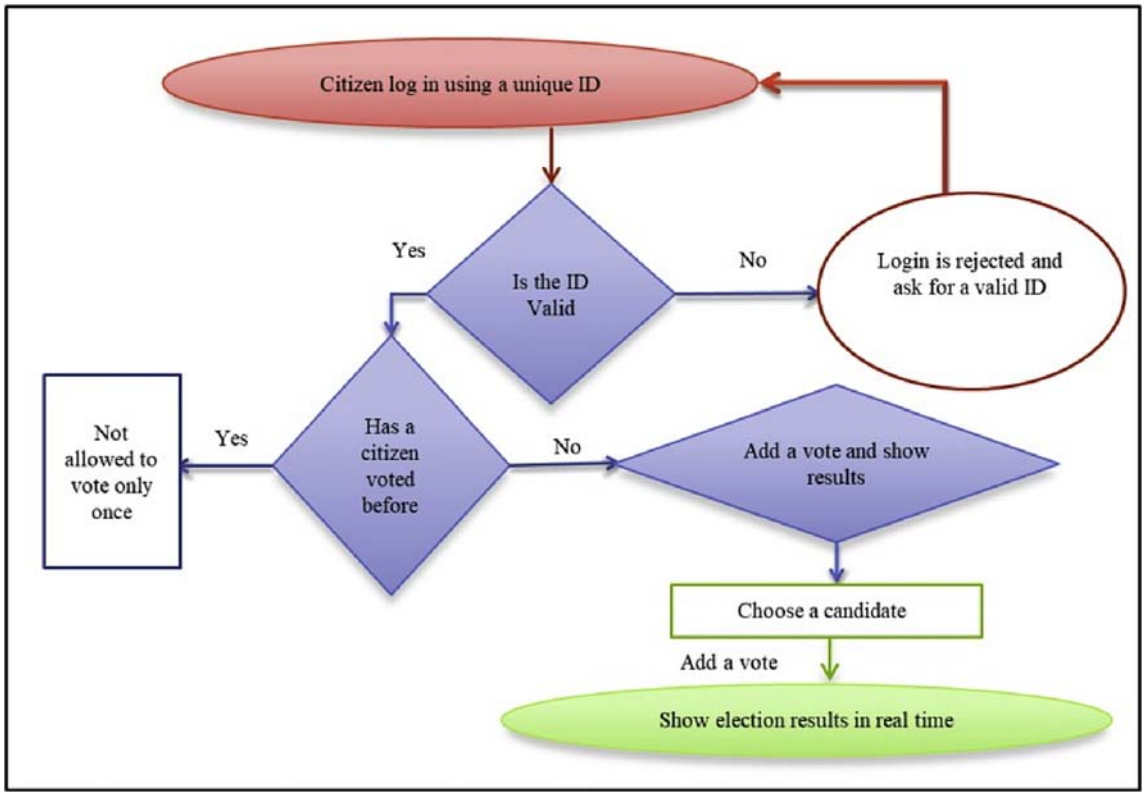

Fig. 5. Flow chart of the confirmation process.

multiply the coefficients of the matrices and validate the ledger, then updating the ledger and applying SVD decomposition to cast back the matrices U, S, and V^{-}.

The reason behind this process is to have another copy of the results in another form which is achieved by applying the SVD technique and distribute the outputs of SVD matrices in separated servers as storage nodes.

The E-voting phase consists of the following stages:

4.3.1. Ballot process

The ballot process is an interface for the list of candidates. The voter votes for the desired candidate. A counter is added within the database for each candidate.

4.3.2. Adding transaction of votes to the current block

Setting the distributed network nodes in three separate places, they are responsible for storing the outputs of the SVD ledger, receiving and retrieving data. At the end of the election, the SVD ledger is constructed by multiplying these three matrices (the resulting ledger is of binary data indicating values of 1s that represent votes for the desired candidate) and match the contents of the ledger against the SQL database of the results.

4.3.3. Retrieving the U, S, V from the network nodes

The system consists of three distributed network nodes. They represent storage nodes (one for the left matrix of SVD decomposition (U), one for representing the singular values of SVD decomposition (S), and the third one represent the right matrix of SVD decomposition (V^{-})). These matrices are retrieved to obtain the ledger as shown in Fig. 6. This process will save us space, time, and speed for retrieving and casting data in this form instead of the whole ledger. The ledger will always be distributed to the network nodes after each time a block is created.

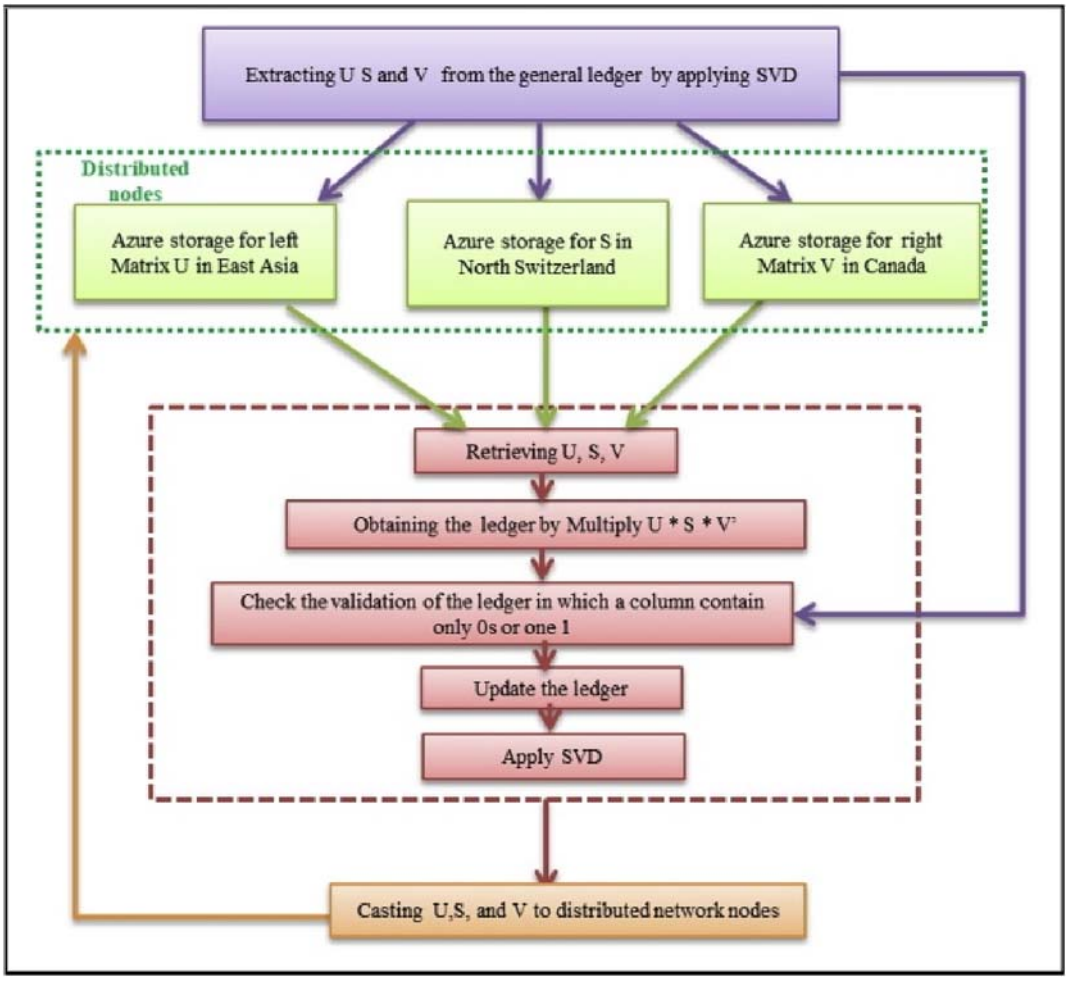

Fig. 6. A Block diagram showing the distributed network nodes with SVD.

Table 3
General ledger of size 8*10.

	V1	V2	V3	V4	V5	V6	V7	V8	V9	V10
C1	0	0	1	0	0	0	0	0	0	1
C2	0	1	0	0	0	0	0	1	0	0
C3	0	0	0	0	0	1	0	0	0	0
C4	0	0	0	0	0	0	0	0	0	0
C5	0	0	0	1	0	0	0	0	0	0
C6	0	0	0	0	0	0	0	0	0	0
C7	0	0	0	0	1	0	0	0	0	0
C8	1	0	0	0	0	0	1	0	1	0

V: Voter, C: Candidate.

4.3.4. Obtaining the ledger by multiplying U, S,V'

The adaptive ledger is obtained through the individual matrices, U, S, and V, by multiplying U, S, V′ in the main node where the ledger is checked whether or not it has been manipulated.

4.3.5. Ledger validation

This step is an important step where the ledger is checked to be validated. In other words, each column that represents a citizen in the general ledger must have at least only one 1 corresponding to only one candidate.

4.3.6. Updating the adaptive ledger

After checking the validation of the ledger, the ledger is updated in the main node to hold the current data in an acceptable form.

4.3.7. Applying the SVD decomposition

The SVD method is applied to produce the new three matrices U, S, and V instead of the whole ledger with sparse data, which will save us time and space.

4.3.8. Casting the U, S, V to the network nodes

After applying the SVD method on the updated ledger, the main node cast back the individual matrices U, S, and V to the distributed network nodes and repeat all the above steps until the election ends.

Table 4
The left matrix U for SVD applied on a ledger 8*10.

0	0	1	0	0	0	0	0
0	1	0	0	0	0	0	0
0	0	0	0	0	1	0	0
0	0	0	0	0	0	0	−1
0	0	0	1	0	0	0	0
0	0	0	0	0	0	1	0
0	0	0	0	−1	0	0	0
−1	0	0	0	0	0	0	0

4.3.9. Applying a hash function

The hash function SHA256 as explained in section 3.3 is used to chain the blocks together, making it more secure and more complicated against any attempt of manipulation. In which, each block is chained by the previous block by its hash value of SHA256 (32 bytes) adding more security to the system.

4.3.10. Chain the block with the previous block

After applying the hash function, the block is now officially added to the network.

4.3.11. Creating a new block as the current block till the end condition occurs

A new block is created as the first one, by adding votes as transactions. All the mentioned steps in the e-voting phase keep running until the end condition occurs which is reaching the time that is set by the administrator to end the election event.

4.4. Result phase

The final phase is the results phase that is a chart or a table suitable enough to show the results online (the results are shown in real-time to achieve transparency) and viewed by all citizens. The votes are fetched from the queue storage in the cloud and deleted as soon as they are grouped as transactions to form a block.

5. Experimental results

The proposed system solves the issue of dealing with a ledger of large size by applying the SVD dimensionality reduction technique that is employed also as mentioned previously as a security matching tool.

Our general ledger is of size 100*100000 (100 rows indicate the number of candidates and 100000 columns indicate the number of votes). Each time a block is

Table 5
The diagonal singular value matrix S for SVD applied on a ledger 8*10.

1.7321	0	0	0	0	0	0	0	0	0
0	1.4142	0	0	0	0	0	0	0	0
0	0	1.4124	0	0	0	0	0	0	0
0	0	0	1.0000	0	0	0	0	0	0
0	0	0	0	1.0000	0	0	0	0	0
0	0	0	0	0	1.0000	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

Table 6
The right matrix V for SVD applied on a ledger 8*10.

−0.5774	0	0	0	0	0	0	0.5164	−0.1225	−0.6205
0	0.7071	0	0	0	0	−0.7071	0	0	0
0	0	0.7071	0	0	0	0	−0.5477	−0.0866	−0.4387
0	0	0	1.0000	0	0	0	0	0	0
0	0	0	0	−1.0000	0	0	0	0	0
0	0	0	0	0	1.0000	0	0	0	0
−0.5774	0	0	0	0	0	0	−0.2582	−0.6325	0.4472
0	0.7071	0	0	0	0	0.7071	0	0	0
−0.5774	0	0	0	0	0	0	−0.2582	0.7550	0.1733
0.0000	0	0.7071	0	0	0	0	0.5477	0.0866	0.4387

Table 7
Retrieving the original matrix with no loss of information by applying U(:,1:r) *S(1:r, 1:r)*V(:,1:r)'.

0	0	1.00	0	0	0	0	0	0	1.00
0	1.00	0	0	0	0	0	1.00	0	0
0	0	0	0	0	1.00	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	1.00	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	1.00	0	0	0	0	0
1.00	0	−0.0000	0	0	0	1.00	0	1.00	−0.00

created the SQL database for results, stores the counts for candidates, and SVD is applied on the general ledger that is updated by replacing a zero value by 1 for each citizens' vote corresponding to the row for a certain candidate. This process mentioned previously aims to keep a copy of the immediate results in another form.

For simplicity we consider an 8*10 general ledger shown in Table 3 as an example to show how SVD works when we have values of ones and zeros, in which each column has at most the value 1 referring to

Fig. 7. First interface for voters to cast a vote or go to the result page.

Table 8
The SQL database for the first 30 voter.

No. of voter in SQL database	Voter Name	Voter ID
1	Hassan Ali Jiwad	100150162
2	Hussain Hamoodi Alawy	100150166
3	Rajja Mohammad Hassan	100150170
4	Tammar Khalid Hazim	100150174
5	May Abdul Raheem	100150178
6	Worood Falah Khalid	100150182
7	Dalal Raji Hussaien	100150186
8	Fakkar Thamer Salih	100150190
9	Kareem Naji Yaman	100150194
10	Shukriyia Abdul Elah	100150198
11	Rawnaq Sahib Hussien	100150202
12	Saber Abdul Rahman	100150206
13	Sabriya Ibrahim Ali	100150210
14	Mayada Abdul Aljaleel	100150214
15	Safa Hashem Abood	100150218
16	Zainab Gafil Bassam	100150222
17	Noor Abul Ritha	100150226
18	Dua'a Abul Ameer	100150230
19	Hawraa Mahdi Abd	100150234
20	Salih Thamer Hadi	100150238
21	Mihsen Mohammad Hadi	100150242
22	Hadi Gaffar Ali	100150246
23	Qasim Khalid Waleed	100150250
24	Karrar Muhammad	100150254
25	Ali Jiwad Hussain	100150258
26	Tammara Turkey Hassan	100150262
27	Bilal Jassim Ali	100150266
28	Thamer Jiwad Ali	100150270
29	Thay Abul Rahman	100150274
30	Saja Qasim Ali	100150278

a vote for the desired candidate, the following ledger is a copy of the SQL DB results, that SVD works on to transform the values into another form and distribute the outputs in three different places:

Table 3 shows that voter1 (V1) voted for candidate8 (C8), V2 voted for C2, and so on. The validation of the ledger is first done by checking the values in which a voter has the right to vote only once so each column should have at most one value of 1.

Table 10
The Queue of transactions that are saved in Microsoft Azure Storage.

	Serial No. of voters in SQL database	Voter_ID	Candidate_ID
1	21	100150242	5
2	5	100150178	7
3	23	100150250	4
4	10	100150198	7
5	7	100150186	1
6	13	100150210	1
7	29	100150274	1
8	16	100150222	10
9	28	100150270	4
10	19	100150234	3
11	20	100150238	10
12	1	100150162	2
13	15	100150218	1
14	17	100150226	1
15	9	100150194	3
16	4	100150174	3
17	25	100150258	10
18	30	100150170	3
19	3	100150278	3
20	11	100150202	5
21	2	100150166	2
22	22	100150246	4
23	6	100150182	7
24	8	100150190	1
25	12	100150206	10
26	24	100150254	2
27	14	100150214	4
28	27	100150266	1
29	18	100150230	10
30	26	100150262	10

As mentioned previously the distributed nodes will contain the outputs of SVD (U, S, and V matrices). The SVD is applied using Matlab as shown in Tables 4–6:

These matrices U, S, and V transformed the general ledger into coefficients of another form. Producing our SVD adaptive ledger which is distributed in three separate servers as storage nodes using cloud storage services.

Table 9
The SQL database for the first 10 candidate.

No. of Candidates in SQL DB	Candidate Name	Candidate ID	Counts	Photo of the candidate
1	Ali Mohammad	1	0	Images＼ p(1).png
2	Mohammad Hassan	2	0	Images＼p(2).png
3	Gufran Jasim	3	0	Images＼p(3).png
4	Ahmen Raheem	4	0	Images＼p(4).png
5	Ali Abdul Rahman	5	0	Images＼p(100).png
6	Ragda Abd Ali	6	0	Images＼p(5).png
7	Shaker Jewad Fadil	7	0	Images＼p(6).png
8	Shahla Abdul	8	0	Images＼p(7).png
9	Sa'ad Lateef Abd	9	0	Images＼p(8).png
10	Qasim Abdul Kareem	10	0	Images＼p(9).png

Show Results as a chart Go to the main page

ID	Name	Picture	N_voters	Percentage
1	Ali Mohammad Hussein	Images\can\p(1).png	7	23
2	Mohammad Hassan Ubed	Images\can\p(2).png	3	10
3	Gufran Jasim Kareem	Images\can\p(3).png	5	16
4	Ahmed Raheem Kareem	Images\can\p(4).png	4	13
5	Ali Abudul Rahman	Images\can\p(100).png	2	6
6	Ragda Abd Ali	Images\can\p(5).png	0	0
7	Shaker Jewad Fadil	Images\can\p(6).png	3	10
8	Shahla Abdul Kader	Images\can\p(7).png	0	0
9	Sa'ad Lateef Abd	Images\can\p(8).png	0	0
10	Qasim Abdul Kareem	Images\can\p(9).png	6	20
11	Rasool Mohammad	Images\can\p(10).png	0	0
12	Nuha Jabbar Nahi	Images\can\p(11).png	0	0
13	Kaffar Abdulla Naji	Images\can\p(12).png	0	0

Fig. 8. The result page corresponding to the table of votes.

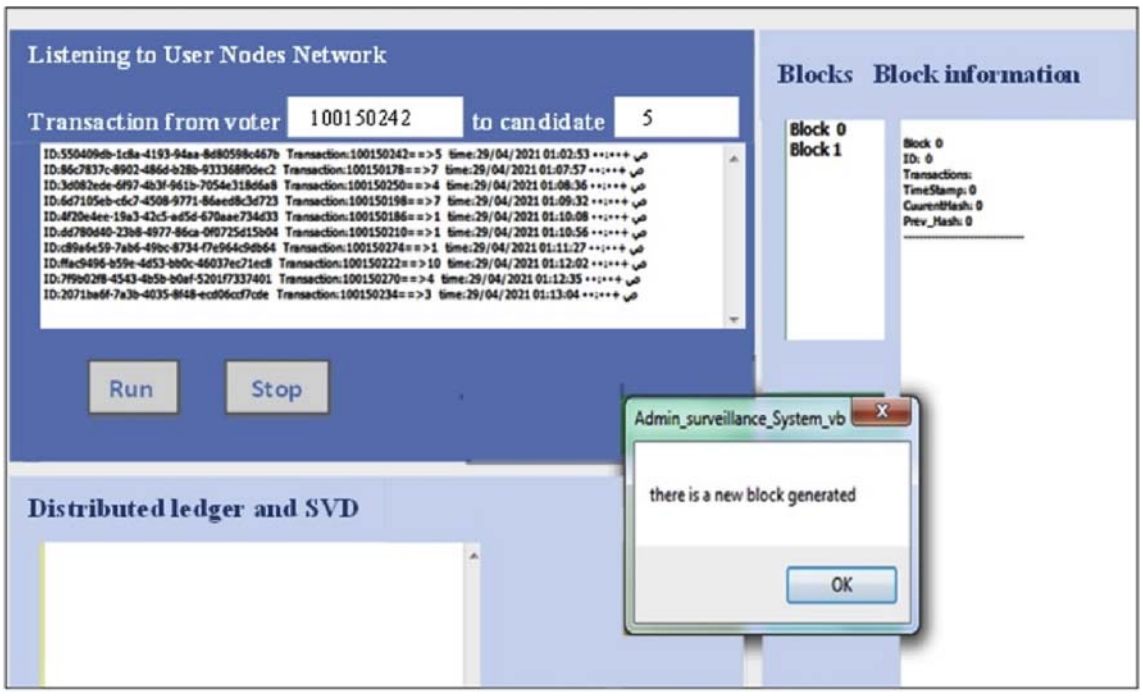

Fig. 9. The system announcing a new block after 10 transactions.

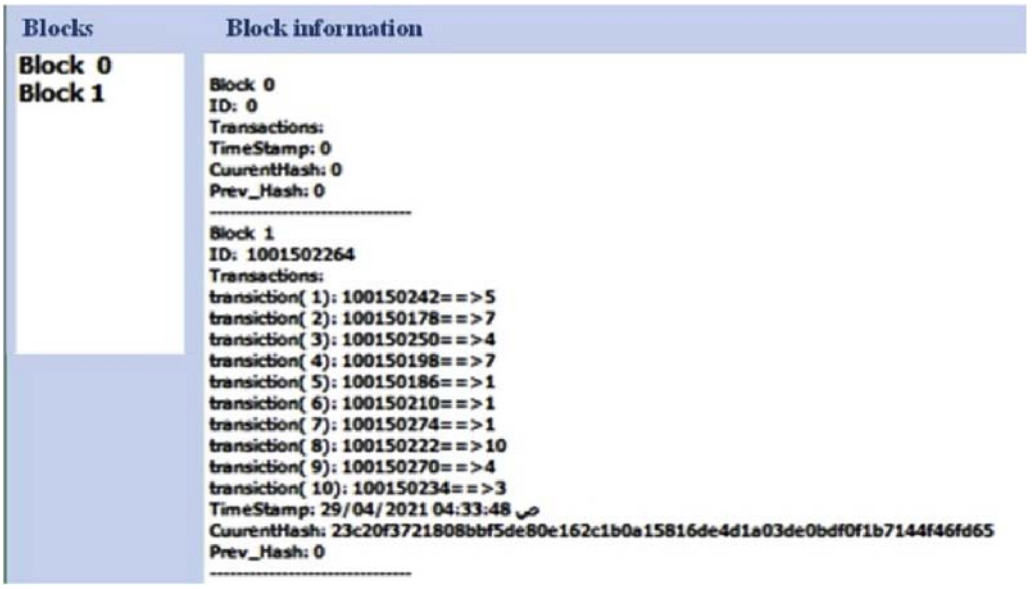

Fig. 10. The contents of one block corresponding to the example in Table 10.

Now we can deal with each matrix in a decomposed manner, as the main aim of SVD is to reduce dimensionality on the general matrix without losing information. This is done by choosing a suitable rank for the matrix depending on the singular values in matrix S (the diagonal matrix).

According to Table 5, we can deal with each matrix in less size (in a decomposed manner), by the rank *r* that is determined based on the importance rate of the non-zero diagonal values in matrix S. We can see in Table 5 that the number of non-zero values is 6, so the rank can be 6 or less especially if the binary matrix is of massive size, For example, the rank of *r* may result in the exact binary matrix we need, by applying the following:

Table 11
Transactions for block1 only.

	Serial No. of voters in the SQL database	Voter_ID	Candidate_ID
1	21	100150242	5
2	5	100150178	7
3	23	100150250	4
4	10	100150198	7
5	7	100150186	1
6	13	100150210	1
7	29	100150274	1
8	16	100150222	10
9	28	100150270	4
10	19	100150234	3

$$U*r + r*r + V*r$$

$$U(:, 1:r)*S(1:r, 1:r)*V(:, 1:r)'$$

Where we keep all the rows in U and columns from 1 to *r*. From S we keep the rows and columns from 1 to *r*. And from V transpose we keep all rows and 1to *r* columns.

So, instead of dealing with a ledger of size 8*10 which is of size 80, we deal with $8*r + r*r + 10*r$. For example, if $r = 3$, then the size of the ledger in this example is decomposed to size 63. By that, we used the SVD technique as a dimensionality reduction method.

Our e-voting system deals with a ledger of size $10^2*10^5 = 10^7$ which is a very large matrix. By having a low rank we can deal with decomposed matrices of less size.

Table 12
Transactions for block2 only.

	Serial No. of voters in the SQL database	Voter_ID	Candidate _ID
11	20	100150238	10
12	1	100150162	2
13	15	100150218	1
14	17	100150226	1
15	9	100150194	3
16	4	100150174	3
17	25	100150258	10
18	30	100150170	3
19	3	100150278	3
20	11	100150202	5

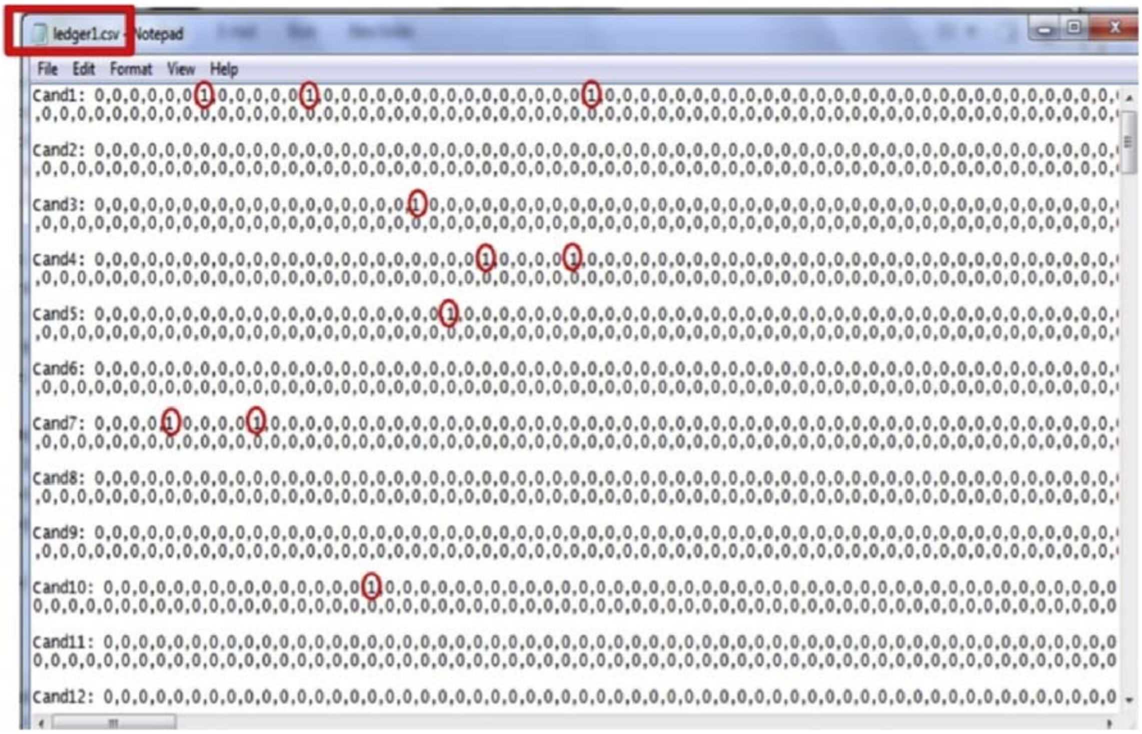

Fig. 11. Ledger1 after the whole process of SVD is applied for Block1.

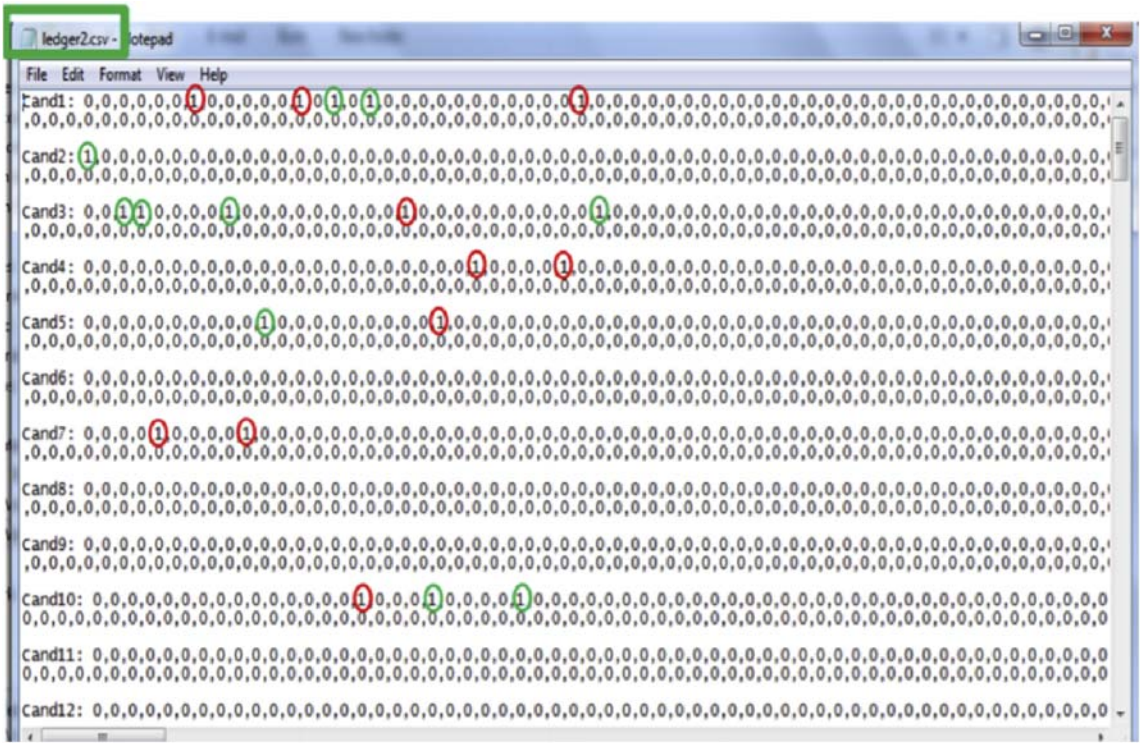

Fig. 12. Ledger2 after the whole process of SVD is applied for Block2.

Table 13
Transactions for block3 only.

	Serial No. of voters in the SQL database	Voter_ID	Candidate _ID
21	2	100150166	2
22	22	100150246	4
23	6	100150182	7
24	8	100150190	1
25	12	100150206	10
26	24	100150254	2
27	14	100150214	4
28	27	100150266	1
29	18	100150230	10
30	26	100150262	10

This result makes a huge difference compared with the original matrix. By retrieving the original matrix U*S*V′ we get the same ledger with no loss of information as shown in Table 7 which is similar to the original general ledger shown in Table 3.

Employing the SVD technique will not only aim to reduce the data dimensionality of the ledger but also speed up the process of dealing with the ledger in less amount of time.

Additionally, this ledger that is managed by SVD is used at the end of the election to be matched with the results as a matching tool to rate the transparency of the voting process.

6. Results and discussion

The proposed e-voting system consists of two main interfaces, the web application for the voter and another interface for the administrator. The first interface that appears to the user (voter) is shown in Fig. 7, where the voter chooses to cast a vote or view the results if he\she voted previously.

The voters enter their ID national card number to cast a vote that is allowed only once, otherwise, a message is shown rejecting the same ID or an invalid. After voting, the voter has the right to view the result as a table or a chart for transparency.

The interface of the administrator is where surveillance occurs. The administrator has no control over the system as the system works dynamically in real-time. However, the admin sets the number of transactions that are needed in a block as soon as the system begins to run.

The administrator views how the e-voting process occurs by showing which voter ID voted to which candidate ID. The administrator platform will keep receiving transactions from queue storage in a cloud according to the number of transactions that are set previously. Then a block is generated containing the ID

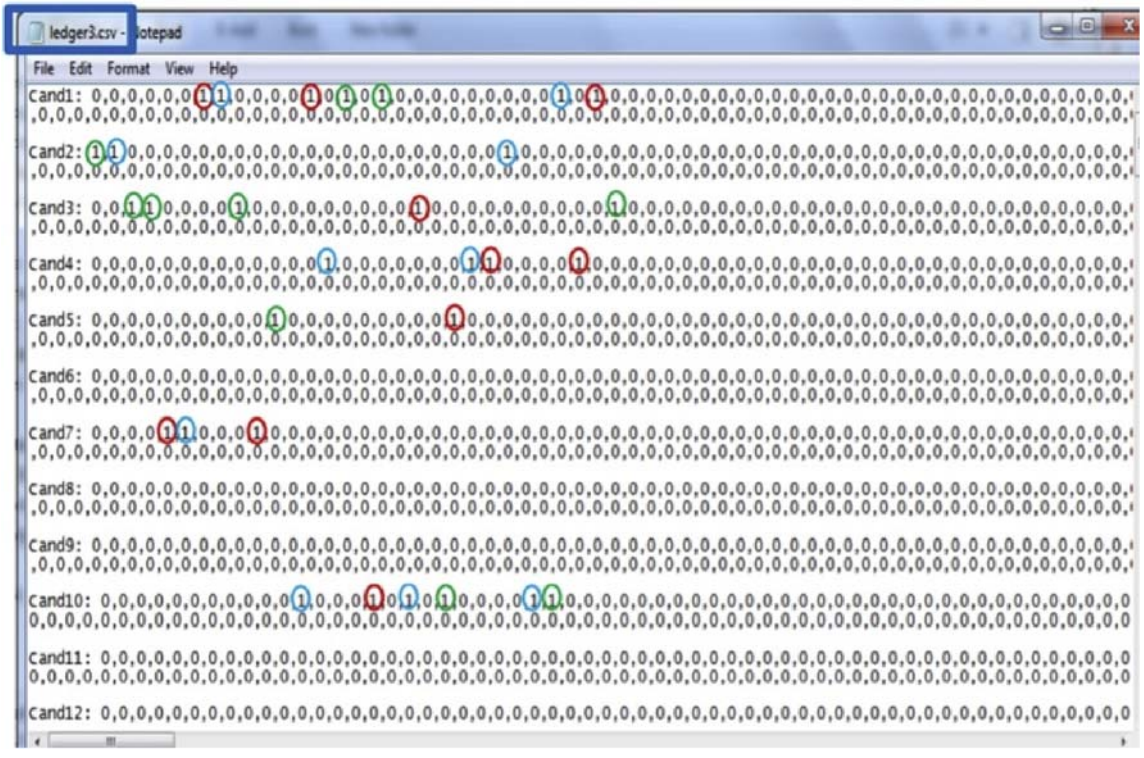

Fig. 13. Ledger3 after the whole process of SVD is applied for Block3.

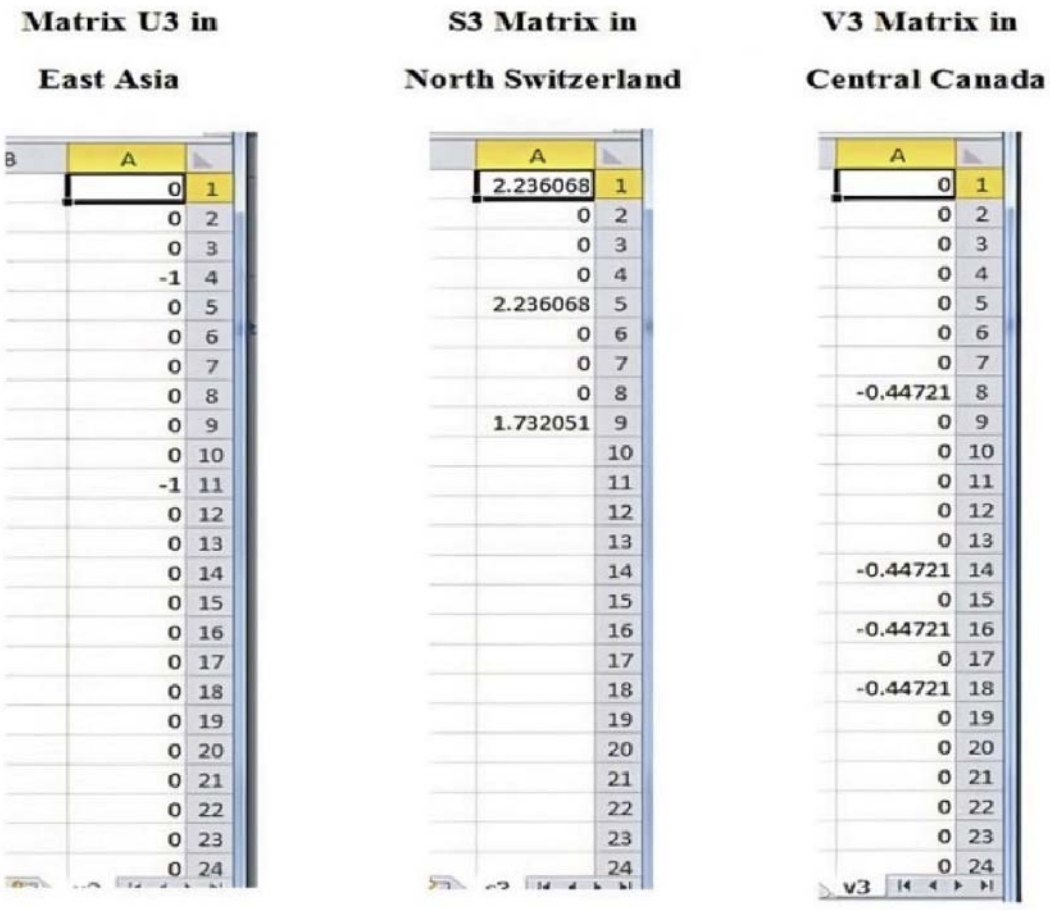

Fig. 14. The U3, S3, and V3 matrices for the ledger3 of block3.

of the block, transactions indicating which ID voted to which ID for candidates, a timestamp, a hash value, and the hash value of the previous block.

The following example of a real test done by volunteers illustrates how the SVD technique is employed as a tool to check the transparency of the system:

- **A Step by Step Example**

The transactions in each block are set to 10. The following tables, Tables 8 and 9 are part of the SQL database for the voters and candidates respectively. For simplicity, the voter DB shows the first 30 voters, and the candidate DB shows the first 10 candidates.

The following voters voted for the following candidates as shown in Table 10. For simplicity, we gave every 10 transactions in a single block a color. In this example, voter 21 who's ID is 100150242 voted for candidate number 5, and so on.

According to the 30 votes in Table 10, Fig. 8 shows the immediate results in real-time by clicking on the (view the results) button to achieve transparency, in which candidate 1 has seven votes, candidate 2 has three votes, and so on.

During the election event, the system will start listening to the voting processing fetching transactions from the queue storage in the cloud, and delete them from the queue as they are no longer necessary to be saved there. Fig. 9 shows the administrator interface.

When the number of transactions reaches 10, a message appears announcing that a new block is generated and name Block1. Each block consists of a block ID which is the summation of all voter IDs in the block, the transactions showing which ID voted to

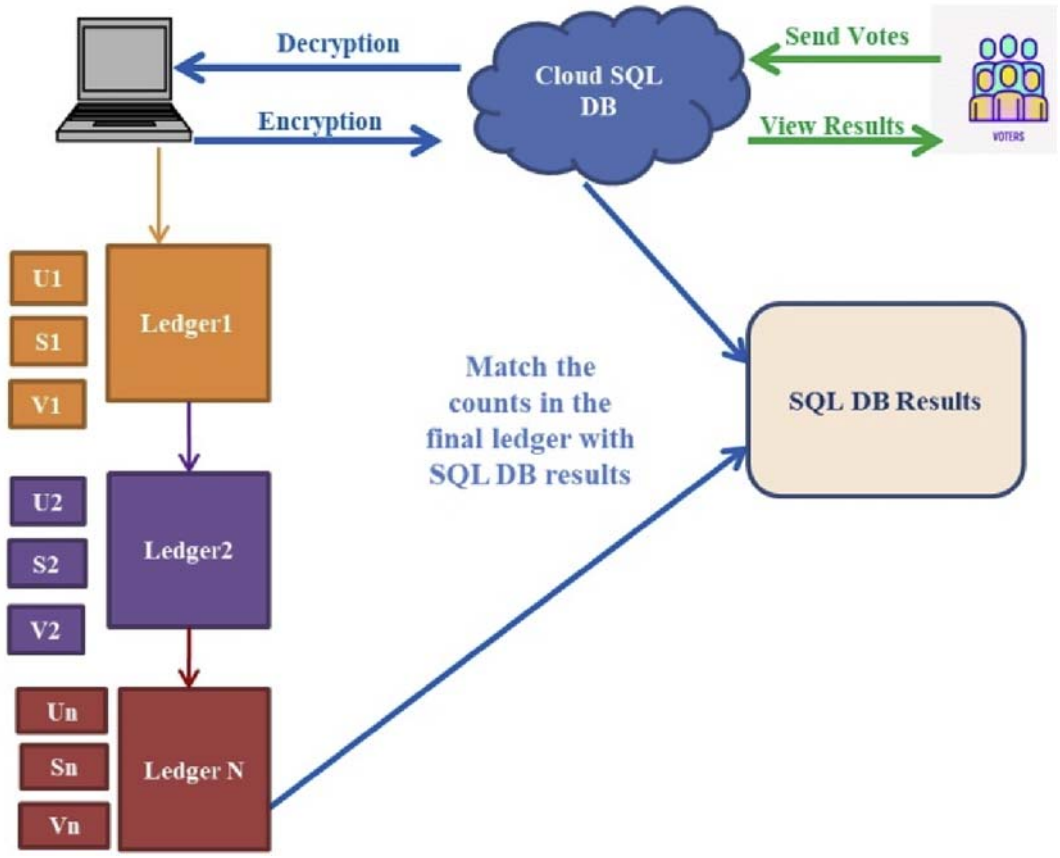

Fig. 15. The last SVD ledger matches its counts with the number of votes in the SQL database of results.

which candidate number, a timestamp showing the date and the time of creating a block, a hash value using the hash function (SHA256) and the hash value of the previous block which is 0 for the initial block. As shown in Fig. 10.

As soon as a block is generated, the matrices U, S, and V are downloaded (the left matrix U stored in East Asia through the Azure Microsoft storage account, the Singular value matrix S stored in North Switzerland, and the right matrix V stored in Central Canada). Initially, these matrices contain zero values they are downloaded to the system, read, and write every time a new block is created.

After reading the matrices, the ledger is constructed by multiplying U*S*V' producing the adaptive ledger that is used at the end of the event to match the results with the results in the SQL database.

Then, SVD decomposition is applied to the updated adaptive ledger to extract the newly updated matrices U, S, and V. (the ledger contain rows of candidates and columns of voters), each time a voter votes for a certain candidate the ledger is updated by replacing a 0 by 1 corresponding to the column of the voter with the row of the candidate.

The U, S, and V matrices are then written as files and cast back to the network nodes. All these steps are repeated on every block till the end of the event.

Each time a block is generated, an adaptive ledger and three matrices U, S, and V are created by SVD. In other words, Block1 will have (ledger1, U1, S1, and V1), Block2 will have the updated (ledger2, U2, S2, and V2), and so on. According to the voters in block1, as shown in Table 11, we find that the corresponding voters that voted for the corresponding candidates are circled in red as shown in Fig. 11. For example, the voters who voted for candidate number 1 have the serial numbers 7, 13, and 29 that are the number of columns in ledger 1, and so on.

For the next block (Block 2) in Table 12, we see that the voter with the serial number 20 (which is the number of the column in the ledger) has voted for candidate 10 and so on. The new update for the

adaptive ledger which is now ledger2 is circled in green as shown in Fig. 12. Again this ledger will have an updated copy of U2, S2, and V2.

The final ledger is ledger 3 for example, voter 2 with the serial number 2 has voted for candidate 2, and so on as shown in Table 13 (represent Block3) and circled in blue as shown in Fig. 13. Again, ledger 3 has an updated copy of U3, S3, and V3.

The U3, S3, and V3 are shown in Fig. 14 for the last ledger which is ledger3. The values get more complicated as long as the votes increase because, in SVD. Each value in each row is related to all values in that row, resulting in real complicated coefficients.

This last ledger has the counts for each candidate that is matched with the SQL database of results to rate the transparency and success of the election event, as shown in Fig. 15.

In our e-voting system, the SVD technique uses low-rank for large binary matrix, which showed a perfect decomposed matrix with no loss of results. For example, according to the rank = 3 for 100 candidates and 100000 voters, instead of downloading reading, and writing a matrix with a size of 10000000, we deal with size U*3 + 3*3 + V*3 which is 100* 3 + 9+ 100000*3 = 300309.

All these processes done by SVD allows us to employ this technique as a tool that retrieves the results in another forum where we produce a ledger of results that is saved and secured with the help of Microsoft Azure cloud services that is used to save the databases, results, and copies of U, S, and V in a distributed manner that is kept as a strong tool for matching the similarity of results ensuring a successful transparency election event.

7. Conclusions

Applying the SVD dimensionality reduction method as a security matching tool aims to keep another copy of results as soon as a block of votes is created and distribute the extracted three output matrices (U, S, and V) of SVD in three different areas (nodes) through the Azure cloud storage. Instead of having the ledger in one place, this process of applying SVD allows the ledger to be distributed in a way that cannot help an attacker control the results. Another benefit of employing the SVD technique is decreasing the size of the ledger as small as possible, as the SVD takes charge of this process by decomposing the matrices of U, S, and V and keeping the important data safe with no loss of information. Also, the system manages the security in different levels by checking the validation of the ledger where each voter votes only once for one candidate, and each block of transactions (votes) is secured by a hash function (SHA256). The SVD technique plays an important role in checking the transparency and success of the election event. At the end of the event, the last adaptive distributed ledger that is based on SVD is matched with the SQL database of results. Otherwise, in case of an attack occurs such as SQL injection, the final results of elections are based on the adaptive distributed ledger that is kept distributed in another form of data. All this work is considered a step forward leading to more promising systems dealing with the ledger based on the SVD data mining technique.

It is suggested to add biometric algorithms as future works within the confirmation process for authentication confirmation stages and extend the use of the singular value decomposition (SVD) technique to work on larger data or develop the technique for any suitable application such as surveys, healthcare applications indicating symptoms of disease, or as part of the E-government applications. Also, other data mining techniques can be suggested instead of SVD such as principal component analysis (PCA) or other methods. Due to the huge number of blocks that are created and processed by the SVD technique, the system is recommended for districts of cities instead of using it for citizens of a whole city. Also, another limitation is that the e-voting system needs an internet connection for all voters' devices to enter the e-voting web application.

Acknowledgments

We are thankful to all citizens who volunteered to vote as part of testing the system.

References

[1] F. Hjálmarsson, G.K. Hreiðarsson, Blockchain-based E-voting system, in: IEEE 11th Int. Conf. On Cloud Computing, San Farnsico, 2018, pp. 1–6.
[2] M. Ahmad, A.U. Rehman, N. Ayub, M.D. Alshehri, M.A. Khan, A. Hameed, H. Yetgin, Security, usability, and biometric authentication scheme for electronic voting using multiple keys, Int J Distribut Sens Net 16 (2020) 1–16.
[3] A. Indapwar, M. Chandak, A. Jain, E-voting system using Blockchain technology, Int. J. Adv. Trends Comput. Sci. Eng. 9 (2020) 2778.
[4] S. Nakamoto, Bitcoin: A Peer-to-Peer Electronic Cash System. http://www.Bitcoin.Org. (Accessed 15 March 2021). Accessed.
[5] D.A. Wijaya, Ebook Bitcoin Tingkat Lanjut, Indonesia, 2016.
[6] H. Watanabe, S. Fujimura, A. Nakadaira, Y. Miyazaki, A. Akutsu, J. Kishigami, Blockchain contract: a complete

consensus using blockchain, in: IEEE 4th Global Conf. Consum Electron, 2015, pp. 577−578.
[7] R. Hanifatunnisa, B. Rahardjo, Blockchain-based e-voting recording system design, in: 11th Int. Conf. On Telecommunication Systems Services and Applications, Lombok, Indonesia, 2017, pp. 1−6.
[8] M. Yasmine, Tabra, Internet voting system in Iraq, Int. J. Adv. Res. Comput. Sci. Software Eng. 3 (2013) 47−52.
[9] S. Aakash, Aashish, Akshit, and Sarthak, Online Voting System, SSRN electronic library, 2020, pp. 1−5.
[10] G. Kalaiyarasi, K. Balaji, T. Narmadha, V. Naveen, E-voting system in smart phone using mobile application, in: Int. Conf. On Advanced Computing and Communication Systems, Coimbatore, India, 2020, pp. 1466−1469.
[11] A. Ben Ayed, A conceptual secure blockchain-based electronic voting system, Int. J. Network Secur. Appl. 9 (2017) 5−9.
[12] R. Govindaraj, P. Kumaresan, K. Sree Harshika, Online voting system using cloud, in: Int. Conf. On Emerging Trends, Information Technology and Engineering Vellore, India, 2020, pp. 1−4.
[13] S. Sekar, C. Vigneshwar, J. Thiyagarajan, V.B. Soorya Narayanan, M. Vijay, Decentralized e-voting system using Blockchain, Int. J. Eng. Technol. 7 (2020) 312−324.
[14] I. Bashir, Mastering Blockchain, second ed., Packt Publishing Ltd, Birmingham, 2018.
[15] H. Abdullah, A.H. Ibrahim, Blockchain technology opportunities in kurdistan, applications, and challenges, Indonesian J. Electr. Engi. Comp. Sci. 18 (2020) 405−411.
[16] J. Leskovec, A. Rajaraman, J. Ullman, Handbook Mining of Massive Datasets, second ed., Cambridge University Press, Cambridge, 2014.
[17] A.A. Abdulkadhem, T.A. Al-Assadi, Geo-localization of video-based on proposed LBP-SVD method, Int. J. Civ. Eng. Technol. 10 (2019) 409.
[18] A.A. Abdulkadhem, T.A. Al-Assadi, Proposed a content-based image retrieval system based on the shape and texture features, Int. J. Innovat. Technol. Explor. Eng. 8 (2019) 2189.
[19] A.K. Bhandari, A. Kumar, P.K. Padhy, Enhancement of low contrast satellite images using discrete cosine transform and singular value decomposition, World Academy of Science, Eng. Technol. J 5 (2011) 20−26.
[20] K. Meenakshi, C.S. Rao, K.S. Prasad, A fast and robust hybrid watermarking scheme based on schur and SVD transform, Int. J. Res Engin. Technol. 3 (2014) 7−11.
[21] B. Madhu, G. Holi, An optimal and secure watermarking system using SWT-SVD and PSO, Indon. J. Electr. Engin. Comp. Sci. 18 (2020) 917−926.
[22] D.P. Berrar, W. Dubitzky, M. Granzow, Singular value decomposition and principal component analysis, in: A Practical Approach to Microarray Data Analysis, Springer, Kluwer: Norwell, 2003, pp. 91−109.
[23] National Institute of Standards and Technology, Secure Hash Standard, Federal Information Processing Standards Publication, Gaithersburg, 2015, pp. 40−180.
[24] S.B. Suhaili, T. Watanabe, Design of high-throughput SHA-256 hash function based on FPGA, in: 6th Int. Conf. On Electrical Engineering and Informatics, Langkawi, 2017, pp. 1−6.
[25] A.B. Ayed, A conceptual secure blockchain-based electronic voting system, Int. J. of Network Security and Its Applications 9 (2017) 5−9.
[26] E. Srbinovska, P. Mitrevski, Web services deployment in Microsoft azure cloud computing platform, Int. S. Con. Inf. Commun. Ener. Syst. Technol. 1 (2014) 75−78.
[27] Teleco advises businesses to move to the Cloud with MS Azure. https://teleco.ca/teleco-advises-business-to-move-to-the-cloud-with-ms-azure/. (Accessed 2 April 2021). Accessed.
[28] G. Carutasu, M.A. Botezatu, C. Botezatu, M. Pirnau, Cloud computing and windows azure, in: Int. Conf. 8th Edition Electronics, Computers and Artificial Intelligence, Ploiesti, 2016, pp. 7−12.
[29] T. Dykstra, R. Anderson, M. Wasson, Building Real World Cloud Apps with Windows Azure, Microsoft, USA, 2014.
[30] R. Jennings, Cloud Computing with the Windows Azure Platform, Wrox, Indianapolis, 2009.