Traces: Inkling Blockchain for Distributed Storage in Constrained IIoT Environments

Riya Tapwal, Pallav Kumar Deb, Sudip Misra, *Senior Member, IEEE*, Surjya Kanta Pal

***Abstract*—Storing data from Industrial Internet of Things (IIoT) sensors in blockchain (BC) for monitoring the applications leads to management issues like bloating. The crux of this work is generating traces (the part of industrial data) using an ARIMA model and storing only the metadata over the network, resulting in reduced delay and managed data. We determine the size of the traces for storing on the S&G blocks (blocks that store traces along with their metadata) by considering principal parameters such as training time, block size, and error. In general, S&G consists of three phases: 1) Categorizing the data into groups based on their sampling rates, 2) Storing the trace of data and metadata into the blocks, and 3) Retrieving the entire data. We demonstrate the feasibility of S&G with errors and regret in the range of 0.07 to 0.10 and 0.20 to 0.25, respectively, using the appropriate ARIMA model.**

***Index Terms*—Blockchain, Industrial Internet of Things, ARIMA model, Storage-as-a-Service, Big data.**

I. Introduction

The sensors in IIoT environments generate sizeable and heterogeneous data with varying sampling rates, variability, data type, and volume from multiple ranges of applications (force, power, temperature, and others), leading to data management and security challenges. Since blockchain (BC) is one of the solutions that implicitly offers data immutability, transparency, and security in addition to a distributed architecture [1], it is often an ideal platform for storing industrial data [2]. However, IIoT sensors generate a massive amount of time-series big data, and storing them on the BC, especially on distributed resource-constrained devices, leads to the bloating problem [3]. Further, the utilization of external storage devices or synergy of BC with the cloud vanishes the various benefits of BC, such as decentralization, distributability, and immutability. Apart from this, it also results in the extra overhead of offloading data to the cloud and fetching from the same, which is not suitable for industries that require real-time processing. In such scenarios, an optimized content storage mechanism on distributed resource-constrained BC deployments for facilitating Storage-as-a-Service (SaaS) is beneficial.

In this work, we propose Store and Generate (S&G) blocks, a method for addressing the challenges mentioned earlier. The crux of this work is the ability of an ARIMA model [4] to generate sequences from a given time-series data. We strategically select portions (or traces) of the industrial data for storing on the blocks of a BC along with the metadata for training the model (using the same trace). The same is repeated for the next batch of sensor data. We use metadata for generating back the sensor data and concatenating it with the traces before presenting it on the screen on request from the end-users. We determine the minimal size of the traces by considering the training time, block size, and error (after generation). In summary, S&G does not require the need to store the sensor data over distributed resource-constrained BC deployments in its entirety and only stores the traces and the necessary metadata, which reduces the overall storage size.

Example scenario: Consider a factory floor consisting of an accelerometer operating with a sampling rate of 48 KHz and a power sensor with a sampling rate of 12 KHz. Indeed, the data rate is very high as well as there is variation in the attributes of the data. In such a scenario, the conventional blockchain does not suffice the storage requirement of the industry as the massive data creates the problem of bloating. However, S&G blocks only store the traces and respective metadata and prove beneficial for the industry with massive data production. S&G blocks allows the factory to manage the data by storing only the traces of data in the BC and generating back the time series data on the demand of the end-users.

A. Motivation

BCs are a popular choice in industrial applications for secured and reliable storage [2]. However, the IIoT sensors generate sizeable data, and storing them on a BC, especially on resource-constrained devices, leads to the bloating problem [3]. Further, the synergy of cloud or fog with BC adds an extra overhead of offloading the data that is not desirable in industries that require real-time processing and vanishes the main benefits (decentralization and distribution) of BC. Distributed storage of these data for monitoring and historical analysis while exploiting the features of a BC is beneficial to facilitate Storage-as-a-Service in industries. This acts as a motive for us to develop the proposed S&G method. We aim to reduce the storage size by pushing only the necessary traces and metadata from the trained model.

B. Contribution

We propose a method for optimizing the size of BC as a platform for storing and retrieving time-series data from IIoT sensors. Towards achieving this, the following are the specific set of contributions in this work:

R. Tapwal, P. K. Deb, and S. Misra are with the Department of Computer Science and Engineering, Indian Institute of Technology Kharagpur, India. e-mail: tapwalriya@gmail.com, (pallv.deb,sudipm)@iitkgp.ac.in

S. K. Pal is with the Department of Mechanical Engineering, Indian Institute of Technology Kharagpur, India. e-mail:skpal@mech.iitkgp.ernet.in

- ***S&G blocks:*** We propose S&G blocks for storing only a trace of the IIoT sensor data on distributed resource-constrained BC devices and training an ARIMA-based model for generating the rest of the sequence on demand. This results in better management of industrial data as we do not need to store the same in its entirety.
- ***Data Categorization:*** We categorize the data based on the type and sampling rate. We use K-Means to categorize data into various groups. Further, we store different types of data in separate transactions inside the blocks for generating separate time series for variant data.
- ***Algorithmic Independence:*** S&G blocks is independent of consensus as well as the machine learning models and is flexible to use any consensus as well as machine learning algorithms. In this paper, as a proof of concept, we utilize Proof of Stake as a consensus mechanism and the ARIMA model for the generation of data.
- ***Robustness:*** We demonstrate the feasibility of the S&G method using different sensors. In particular, we use data from sensors such as accelerometer and torque with varying sampling rates from open-source datasets.
- ***Optimization:*** We formulate a linear programming-based method for minimizing the trace sizes by considering the training time, block size, and error (after generation).
- ***Evaluation:*** Through extensive experiments and deployment on resource-constrained devices such as a network of Raspberry Pis, we demonstrate the feasibility and the efficiency of S&G in comparison to conventional BC.

II. Related Work

Xu *et al.* [5] proposed a section BC to solve the problem of oversized BC and considered equal-sized blocks. In section BC, nodes can change their location to receive more remuneration, and the data can be distributed worldwide. Zerka *et al.* [6] proposed a BC-based platform using sequential distributed learning. The proposed C-Distrim (Chained Distributed Machine Learning) to predict the data. The integration with BC improved transparency and trust as compared to the centralized approach. Further, Kantesariya *et al.* [7] proposed opti-shard for the validation of hierarchical BC. This architecture distributed the transactions among various non-overlapped and disjointed shards. They also proposed a mechanism to choose good shards and identify faulty shards. Singh *et al.* [8] proposed deep learning-based BC for software-defined network to provide seamless data transfer. Purva *et al.* [4] used ARIMA to predict the level of the sea. They did parameter tuning to obtain the parameters of the ARIMA model. The model gave the highest correlation coefficient and the lowest root means square value. Lie *et al.* [10] proposed a groupchain which is suitable for fog computing. The integration of BC with fog computing to service IoT-oriented solutions enhanced scalability. Later, Aunsri *et al.* [14] reduced large dimensional data into the small matrix by using the random compression method. Gangwar *et al.* [15] utilized the ARIMA model for predicting the time series and further proposed a network configuration methodology that can accommodate the variable load.

Table I: Comparison of S&G blocks with various consensus algorithms by considering centralized (C) and decentralized (D), categorization of transactions, block size optimization (BO), and data prediction.

Scheme	C/D	Categorization of transactions	BO	Data Prediction
Ali *et al.* [9]	D	No	No	No
Kantesariya *et al.* [7]	D	No	Yes	No
Lei *et al.* [10]	D	No	Yes	No
Wang *et al.* [11]	D	No	No	No
Taghavi *et al.* [12]	D	No	No	No
Chowdhury *et al.* [13]	D	No	No	Yes
S&G-blocks (proposed)	D	Yes	Yes	Yes

A. Synthesis

BCs are not general-purpose databases and hence, suffer from various challenges such as BC bloating and its unmanageable size. The BC offered by various researchers do not consider these problems, as shown in Table I. This paper proposes a BC that stores only a segment of data. The S&G model is responsible for generating the remaining data. In this paper, we perform segregation of data having similar sampling rates and further store metadata in the blocks. S&G model generates data by using the metadata present in the block.

III. System Model

A. Network Architecture

As shown in Fig. 1, we consider a set of sensors, $S = \{s_1, s_2, s_3, \ldots, s_m\}$ which sense the data produced by various IIoT devices. The sensors forward the data to the system, which categorizes it into groups so that the S&G blocks can generate a separate time series for different sensor's data. After the categorization of data, BC stores data of sensors in variant transactions according to the traits of the data. Further, the S&G blocks use this data to train the machine learning model and generate time series for various sensors. S&G blocks can optimize the size of the blocks by considering three different scenarios: 1) Taking real-time data for a few seconds, storing it in BC, and using it to predict future data using a machine learning model for the next few seconds. 2) Storing complete data in the BC and optimizing the BC by deleting some data and generating the same by using the ARIMA. 3) Taking data in batches and implementing S&G blocks to optimize the size of data. The performance of S&G blocks is the same in all three scenarios. The optimal size of the block, training time of the machine learning model, and the error in predicting data are the same in all three scenarios.

B. S&G blocks

We consider various types of sensors deployed in an industry that sense the data generated by IIoT devices. The IIoT devices generate a massive amount of data and are difficult to manage. To manage this, we first group the data into various clusters. Further, we consider a BC whose block does not contain normal transactions; rather, the transactions are segregated into various groups where each group of the transaction contains

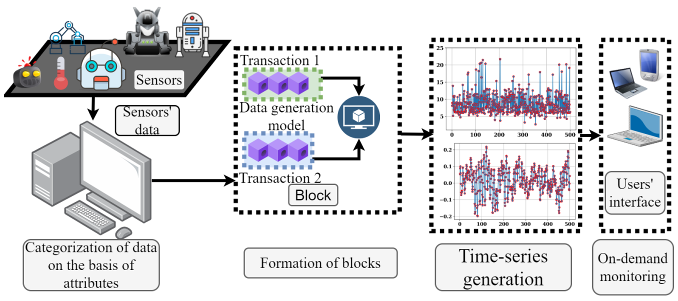


Figure 1: Overview of S&G model.

the data of one cluster. The block contains the data generated by the sensors for a few seconds, and the ARIMA model present in the block generates data for the next few seconds. We repeat the same for the next batch of data. In this manner, the model generates a time series of complete data. The user interface uses time series for monitoring various tasks. Further, the symbols used in S&G blocks are given in Table II.

C. Analytical Framework

We aim to minimize the overall size of the blocks, error, time, and the data points required to train the ARIMA model.

1) Minimizing the number of data points: In ARIMA, the maximum number of data points requires to train the model is dependent on the values of p and q. Hence, for a particular type of sensor S_i with sampling rate $\mho$, the number of data points on which the prediction depends is n and is given by $n = max(p, q)$. The aim is to reduce n for every type of sensor as given in Equation 1.

$$N = min \sum_{i=1}^{m} n_i \tag{1}$$

where m is the number of sensors.

2) Minimizing time: The number of inputs for every sensor varies according to the sampling rate $\mho$. We take input from sensor S for T seconds. Hence, the number of inputs $(I), \forall i \in S$ is given as $I = \mho_i \times T$. Further, we calculate the training time as $\tau = \sum_{j=1}^{m} \sum_{k=1}^{\kappa} \sum_{i=1}^{I} t_{ikj}$ by taking input from the sensor, t_{ikj} is the time required to train the model by considering only a single data point of i^{th} sensor, where k is the number of sensors of a particular type and m is the types of sensors available. The aim is to reduce the value of time for training the model. The training time is directly proportional to the number of data points and is represented as $\tau \alpha N$. The training time for the model with constant k_1 is given as:

$$\tau = k_1 \times N \tag{2}$$

3) Minimizing the size of the block: Let space utilized by single data point i be s_i, $\mathcal{A}$ is the space taken by the ARIMA model, and C is the space required by the other components of the block. The size of the block is given by $\mathcal{B} = \sum_{j=1}^{m} \sum_{k=1}^{\kappa} \sum_{i=1}^{\mho \times T} s_{ikj} + \mathcal{A} + C$. Also, the total number of data points in a block is given as $\eta = \mho \times T \times \kappa \times m$. Further, the blocksize is directly proportional to the number of data points present in the same and is represented as:

$$\mathcal{B} = \eta \times s_i + \mathcal{A} + C \tag{3}$$

Table II: Symbol table.

Symbols	Description
S	Set of sensors
n	Data points of single sensor
p	Lag observations in ARIMA model
q	Size of moving average window
τ	Training time
t_{ikj}	Training time of single data point
k	Same type of sensors
m	Total number of sensors
N	Data points in a single block
A	Space occupied by ARIMA model
B	Block size
C	Space occupied by other components of a block
η	Data points in a block
Y_t	Original data
χ_t	Predicted data
f_t	Loss function
R_t	Regret
β	Balancing factor
J_u	Objective function
ζ	Utilization function
α	Bearable error
κ	Types of sensors
$\mho$	Sampling rate

The aim is to reduce the block size to a minimum such that:

$$min\left(\sum_{j=1}^{m} \sum_{k=1}^{\kappa} \sum_{i=1}^{\mho \times T} s_{ikj} + \mathcal{A} + C\right) \tag{4}$$

where s_{ikj} is the space utilize by the data point.

4) Minimizing the error in predicting the time series: The error in the ARIMA model is given as:

$$\epsilon_t = \epsilon_{t-1} + \epsilon_{t-2} + \cdots + \epsilon_{t-q} \tag{5}$$

Now, if the number of the data points on which the prediction depends increases, the error in the prediction decreases (the error decreases with constant k_2 ($\forall k_2 > 0$)) and is given as:

$$\epsilon_t = \frac{k_2}{\eta} \tag{6}$$

Further, the loss function $\forall p, q <= n$ with $Y_t(p, q)$ as the original data, and $\chi_t(p, q)$ as the predicted data is given as $f_t(p, q) = l_t(Y_t(p, q), \chi_t(p, q))$. The definition of loss function is given as $l_t(x, y) = x - y$. Further, the regret of the model should be minimum and is given as:

$$R_t(p, q) = l_t(Y_t(p, q), \chi_t(p, q)) - min(l_t(Y_t(p, q), \chi_t(p, q))) \tag{7}$$

where (p, q) is the order of the ARIMA model representing the number of auto-regressive and moving average terms, respectively.There is a trade-off between the error and the number of data points. Therefore, we need a balancing factor β whose value lies between (0,1] such that $\beta_b^S + \beta_b^\epsilon = 1$. The objective function $\forall b \in B$ becomes:

$$J_u = \beta_b^S \mathcal{B} + \beta_b^\epsilon \epsilon \tag{8}$$

$$J_u = \beta_b^S (\eta \times s_i + \mathcal{A} + C) + \left(\beta_b^\epsilon \times \frac{k_2}{\eta}\right) \tag{9}$$

Equation 9 represents that if $\beta_b^S > \beta_b^\epsilon$, giving more preference to accuracy and if $\beta_b^S < \beta_b^\epsilon$ giving more preference to less storage space. Further, $\beta_b^S \times \mathcal{B} \leq S_{max}$, $\tau <= \tau_{max}$ and $\beta_b^\epsilon \times \epsilon \leq \epsilon_{max}$ where S_{max} is the maximum possible size of the block and ϵ_{max} is the error in predicting the next value when only a single data point trains the ARIMA model. We calculate the values of ϵ_{max}, S_{max}, and τ_{max} using Algorithm 1. The algorithm runs at most for η number of times. The complexity of the algorithm is $O(\eta)$.

Algorithm 1: Algorithm for calculating the maximum block size, maximum time, and minimum error.

```
Input: Heterogeneous data sensed by sensors
Output: Maximum block size, maximum training time
  and minimum error possible
Initialize:
$S_{min} = \mathcal{A} + C$
$S_0 = S_{min}$, i=1, t=1;
Procedure:
while true do
    S$_i = S_{i-1} + \sum_{s\epsilon S} \sum_{k=1}^{\kappa} \sum_{i=1}^{\mho \times T} s_{ik}$
    $\tau_i = \tau_{i-1} + \sum_{s\epsilon S} \sum_{k=1}^{\kappa} \sum_{i=1}^{\mho \times T} t_{ik}$
    $i = i + 1$
    $t = t + 1$
    if $\tau_i - \tau_{i-1} < \tau_{thresh}$ then
        break;
    end
end
$S_{max} = S_i$, $T_{max} = T_i$, $n = \sum_{s\epsilon S} i$;
$\epsilon_{min} = \epsilon_{t-1} + \epsilon_{t-2} \cdots + \epsilon_{t-n}$
```

For a given space utilization decision (ζ) and bearable error decision (α), we define system utility as the weighted sum of all blocks data prediction utilities $\forall b\epsilon B$.

$$U(\zeta, \alpha) = \sum_{b\epsilon B} J_u \tag{10}$$

We formulate the space allocation to a block and bearable error decision as to the minimization problem $\forall b\epsilon B$ and $\forall s\epsilon S$, which is given as $min_{\zeta,\alpha} U(\zeta, \alpha)$.

Theorem 1. *The utility function $U(\zeta, \alpha)$ is convex.* □

Proof. We prove that $U(\zeta, \alpha)$ is convex by demonstrating that $\frac{\partial^2 U(\zeta,\alpha)}{\partial \eta^2} > 0$. On solving, we obtain the following expression:

$$\frac{\partial^2 U(\zeta, \alpha)}{\partial \eta^2} = \frac{2 \times \beta_b^\epsilon \times k_2}{\eta^3} \tag{11}$$

In Equation 7, the value of β_b^ϵ lies between (0,1), k_2 is the positive constant, and η is the number of data points, which are always positive. Under these conditions the expression $\frac{2\times\beta_b^\epsilon\times k_2}{\eta^3} > 0$, which proves that utility function is convex. □

In this work, we adopt the Karush Kuhn Tucker (KKT) method for finding the optimal block size such that the error in predicting the data and the number of data points in a single block become optimal. Since KKT typically helps in maximizing the functions, we modify the expression in Equation 8 as:

$$V(\zeta, \alpha) = -\beta_b^S \mathcal{B} - \beta_b^\epsilon \epsilon \tag{12}$$

We obtain the Langrangian for Equation 12 as:

$$\begin{aligned} L(\zeta, \alpha) = &\sum_{b\epsilon B} (-\beta_b^S \mathcal{B} - \beta_b^\epsilon \epsilon) + \mu_1 (S_{max} - \beta_b^S \times \mathcal{B}) \\ &+ \mu_2 (\tau_{max} - \beta_b^\epsilon \times \epsilon) \end{aligned} \tag{13}$$

The conditions for finding the optimal expressions are:

$$\Delta_{\mathcal{B}} L = 0, \Delta_\epsilon L = 0, \mu_1, \mu_2 >= 0 \tag{14}$$

where μ_1 and μ_2 are the non-negative Langrangian multipliers. Finally, we obtain the optimal values for $\mathcal{B}$ as:

$$\mathcal{B} = \frac{S_{max}}{\beta_b^S} \tag{15}$$

where $S_{max} = \mathcal{A} + C + \sum_{s\epsilon S} \sum_{k=1}^{\kappa} \sum_{i=1}^{\mho \times \tau_{max}} s_i$. Further, we obtain the optimal error (ϵ) and the optimal number of data points (η) as:

$$\epsilon = \frac{\epsilon_{max}}{\beta_b^\epsilon} \tag{16}$$

$$\eta = \frac{k_2 \times \beta_b^\epsilon}{\epsilon_{max}} \tag{17}$$

We obtain the values of ϵ_{max} and S_{max} from Algorithm 1.

The S&G blocks calculates the optimal block size by applying Lagrange Method. Indeed, if the size of the block becomes optimal, consequently, the training time, data generation time, and error become optimal. This makes the S&G blocks an optimal method for storing industrial data.

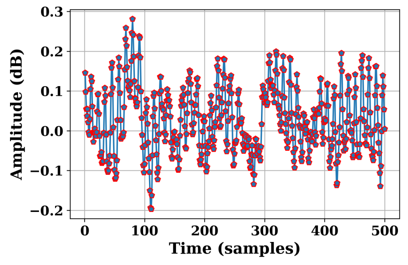
IV. Performance Evaluation

A. Experiment Setup

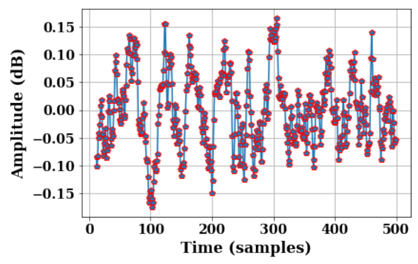
We use Python 3 and three different types of data as proof of the concept for executing our routines. Type 1 data is the data of the accelerometer with a sampling rate of 48000 samples/second, Type 2 data is the data of the accelerometer with a sampling rate of 12000 samples/second, and Type 3 data is the data of torque sensors. Further, we use a modified BC (S&G) that categorizes the transactions and uses a data prediction model to generate a time series of each type of data. We add the ARIMA model to each block so that it can utilize the metadata in the same to produce the next portion of data.

B. Deployment Architecture

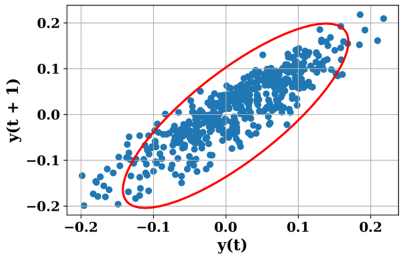
We generate a client-server architecture for implementing the S&G blocks. The client program continuously sends the sensors' data to the BC (server). By using the linear optimization technique, we calculate the optimum number of data points required to generate the time series such that the size of the block and the error in predicting data are minimum. By performing experiments, we observe that 3000 data points are a suitable number for storing in the block. By using these



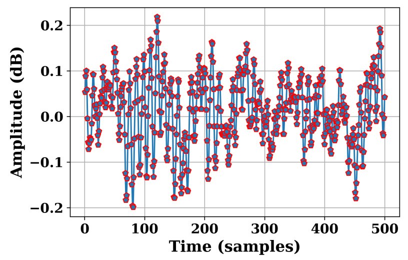
(a) Time series (Accelerometer 12KHz).



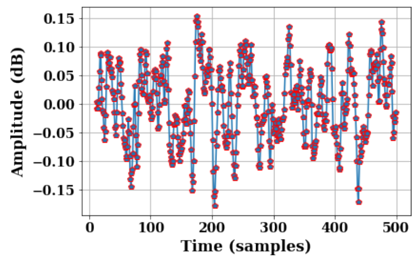
(b) Seasonality (Accelerometer 12KHz).



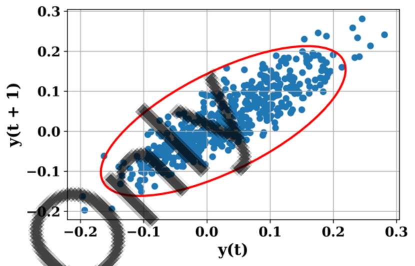
(c) Lag plot (Accelerometer 12KHz).



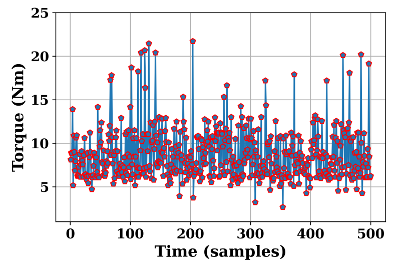
(d) Time series (Accelerometer 48KHz).



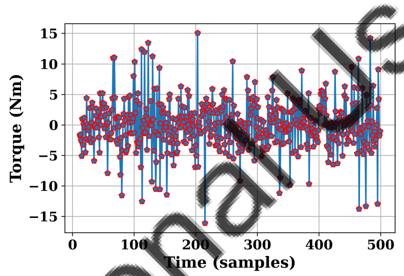
(e) Seasonality (Accelerometer 48KHz).



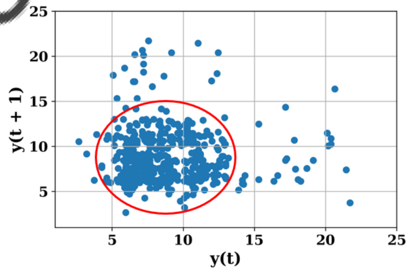
(f) Lag plot (Accelerometer 48KHz).



(g) Time series (Torque 2MHz).



(h) Seasonality (Torque 2MHz).



(i) Lag plot (Torque 2MHz).

Figure 2: Data analysis of three different sensors.

data points, we generate the next 3000 data points using the ARIMA model. We repeat the same for the next batch of the sensor data and store it in the next block of the BC.

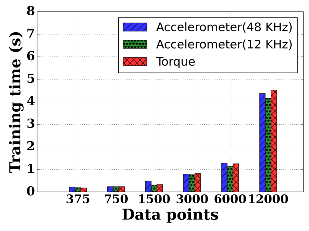
C. Data Analysis

Fig. 2 represents the data produced by various sensors. As proof of concept, we study the data by taking 500 data points. Fig. 2(a) represents the data of the Accelerometer with a 12KHz sampling rate. This figure depicts how the acceleration varies with time. We observe that most of the values of acceleration lie between -0.2 to 0.2 dB. Further, Fig. 2(b) represents the seasonality of the data. We calculate the seasonality by considering 12 lag points and observe that there is moderate seasonality in the plot. Fig. 2(c) is the lag plot of the Accelerometer with a 12KHz sampling rate. The lag plot is elliptical linear with a positive slope. This concludes that the data has moderate randomness with positive auto-correlation. Similarly, Fig. 2(d) represents the data of the Accelerometer with a 48KHz sampling rate. From Fig. 2(e) and Fig. 2(f), we conclude that the properties of the Accelerometer with a 48KHz sampling rate is similar to the Accelerometer with 12KHz. Further, Fig. 2(g) represents the data of the Torque sensor. The value of torque lies in the range of 5Nm to 25Nm. Fig. 2(h), represents the seasonality of the data whereas Fig. 2(i), represent the lag plot of the data. The lag plot is circular with some outliers. From this figure, we can conclude that the data has moderate randomness with no auto-correlation.

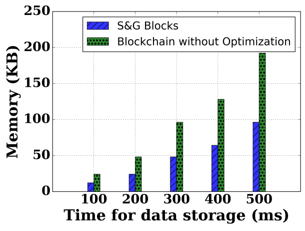
D. Results

1) Relation between training time and data points: We explore the training time against the data points by performing 30 iterations. We observe from Fig.3(a) that with the increase in the number of data points, the training time increases. This is because S&G blocks has to feed more data points to the ARIMA model and take their features. We also observe that the training time increases by 50% when we increase data points from 3000 to 12000. From these observations, we imply that with the increase in the block size (the data points stored in S&G blocks), the training time increases. We also conclude that for real-time scenarios, it is better to utilize S&G blocks as it optimizes the block size such that the training time is diminutive, which is acceptable.

2) Relation between memory and data points: We record the memory consumption by the block against the number of data points. We perform 30 iterations to present our observation. Fig. 3(b) depicts that with the increase in the number of data points, the memory requirement of the block also increases. Further, we also observe that the memory requirement of S&G blocks is almost half as compared to the BC without data optimization. The memory requirement decreases by 49.9%. Further, we also observe that by doubling the data points, the memory requirements also get approximately double. This is because more data points require more space. From these observations, we imply that with the increase in the block size (the data points stored in S&G blocks), the memory requirement increases. We also conclude that for resource-constrained scenarios, it is better to utilize S&G blocks as it optimizes the block size such that the memory requirement is diminutive, which is acceptable.



(a) Training time.



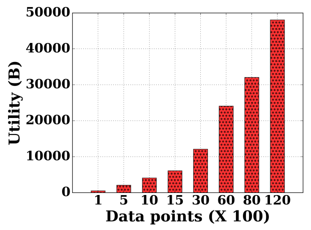
(b) Memory requirement.

Figure 3: Training time and memory requirement.

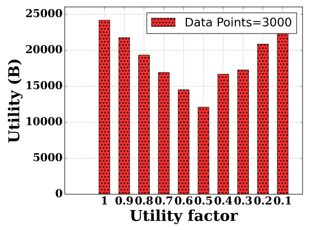
3) Relation between utility with data points: We record the value of utility as mentioned in Eq. 10 with a balancing factor of 0.5 for varying numbers of data points. Indeed, the utility is the summation of error and the block size; we represent the error in terms of space and calculate the utility of S&G blocks. We represent the error as space occupied by the number of erroneous data points. To present our observations, we perform 40 iterations and observe that the value of utility increases with the increase in data points, as shown in Fig. 4(a). This is because the block size factor increases with the increase in data points. We imply that the utility is minimum when the number of data points is minimum.

4) Relation between utility and utility factor: From Fig. 4(b), we explore that if the balancing factor increases more than 0.5, then we are giving more preference to accuracy in predicting the data. But with the decrease in balancing factor, we are trying to optimize the memory requirement. The value of utility decreases with the decrease in the value of the balancing factor to 0.5 balancing factor; after that, it again starts increasing. This is because, with the decrease in the balancing factor, the space factor reduces; however, the error increases. We imply that at the optimal balancing factor for 3000 data points, the utility is around 13000.

5) Data predictions: S&G blocks predict the time-series data, which is further used for data analysis. S&G blocks use a machine learning model for data prediction. As proof of concept, we use the ARIMA model with order (3,1,2) for data prediction. Fig. 5 represents the various types of predicted data by the ARIMA model. From these figures, we can deduce that the model predicts the data with negligible error for different types of data. Fig. 5(a) represents the actual and predicted



(a) Utility with data points.



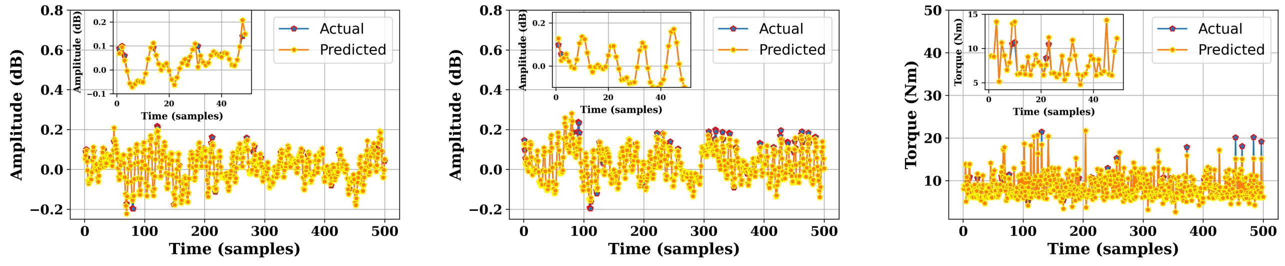
(b) Utility with utility factor.

Figure 4: Utility of S&G blocks.

data of the Accelerometer with a 12KHz sampling rate. We conclude that the predicted data mostly overlapped the actual data, and predicted values of acceleration lie between -0.2dB to 0.2dB, and the error in predicting data is 0.001. Similarly, Fig. 5(b) represents the actual and predicted data of the Accelerometer with a 48KHz sampling rate. We conclude that the predicted values of acceleration lie between -0.2dB to 0.25dB, and the error in predicting data is 0.004. Further, Fig. 5(b) represents the actual and predicted data of the Torque sensor. The actual and predicted values of torque lie between 5Nm to 25Nm, and the error in predicting data is 0.007. We imply that the error in predicting data increases with the number of outliers in data. This is because the model cannot predict the outliers accurately and result in errors. Further, we also imply that S&G model predicts the data with high accuracy without any loss of data.

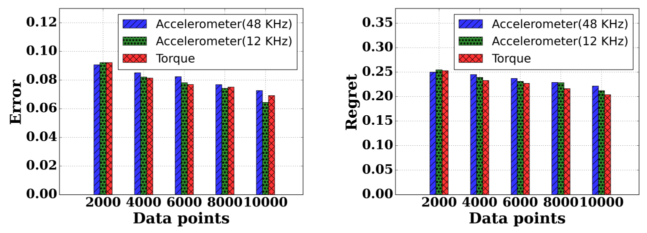
6) Relation between error and data points: We record the errors in predicting the data points by considering the various number of data points for training the S&G model. We perform 40 iterations of experiments to calculate the error for three different types of sensors, i.e., accelerometer with 48000 samples/second, accelerometer with 12000 samples/second, and torque sensor with 2M samples/second. From Fig. 6(a), we observe that with the increase in data points, the error in predicting time series decreases for each type of sensor. Specifically, the error for accelerometer with 48000 samples/second, accelerometer with 12000 samples/second, and torque sensor with 2M samples/second is 0.0727 dB, 0.06441 dB, and 0.06912 Nm, respectively. Further, we imply that the error in predicting the time series is inversely proportional to the number of data points used to train the S&G model. Indeed, S&G blocks optimally stores the data points in the BC which results in diminutive error and further generates data on the demand of the user with no loss.

7) Relation between regret and data points: We explore the regret of the S&G blocks by changing the number of data points and performing 40 iterations to get our observations. We observe from Fig. 6(b) that with the increase in data points, the regret of the S&G blocks decreases. This is because the S&G blocks perform better if the data for training is large. Specifically, the regret for accelerometer with 48000 samples/second, accelerometer with 12000 samples/second, and torque sensor is 0.2216 dB, 0.2121 dB, and 0.2041 Nm, respectively. Also, by varying the order of the ARIMA model used in S&G blocks, regret varies. We imply that the regret



(a) Predictions (Accelerometer 12KHz). (b) Predictions (Accelerometer 48KHz). (c) Predictions (Torque 2MHz).

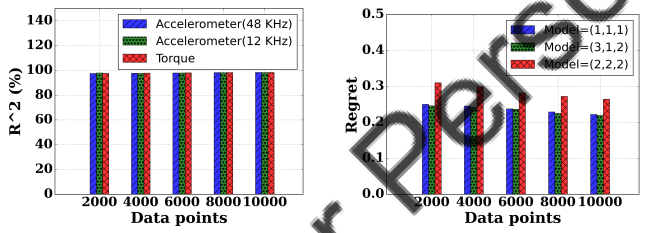
Figure 5: Data predictions by ARIMA model.



(a) Error. (b) Regret.

Figure 6: Error and regret in predicting data.

reduces by 26.66% with the increase in data points. Indeed, S&G blocks optimally stores the data points in the BC which results in diminutive regret and further generates data on the demand of the user with no loss.



(a) R^2 value for varying data points. (b) Regret for different ARIMA models with varying data points.

Figure 7: R^2 value and regret for varying data points.

8) Relation between R^2 and data points: We record the R squared value in predicting the data points by considering the various number of data points for training the S&G model. We perform 40 iterations to present our observations. We calculate the R squared value for accelerometer with 48000 samples/second, accelerometer with 12000 samples/second, and torque sensor with 2M samples/second. From Fig. 7(a), we observe that with the increase in data points, the R squared value in predicting time series increases for each type of sensor. Specifically, for 10000 data points, R squared value for accelerometer with 48000 samples/second, accelerometer with 12000 samples/second, and torque sensor is 98.3%, 98.2%, and 98.3% respectively. Further, we imply that the R squared value in predicting the time series is directly proportional to the number of data points used to train the S&G model. Apart from this, from Fig. 7(b), we conclude that the selection of an appropriate order reduces regret. This is because the inappropriate selection of the order either leads to under-fitting or over-fitting of data and by 24% approximately when changes order from (2,2,2) to (3,1,2) at 3000 data points.

V. Conclusion

In this paper, we proposed – S&G model – for efficiently managing the massive data sensed by a myriad of sensors by storing only a trace of the IIoT sensor data over the distributed resource-constrained BC devices. The proposed scheme subsists of three stages: 1) Grouping of data based on their characteristics, 2) storing the batch of data in the corresponding transaction, and 3) generating a subsequent batch of data using the data prediction model. We stored only a few data points in the block and further generated the next few data points using the S&G model. The proposed model reduced the need for storing whole data in the BC and made the same more manageable. The S&G model generated a time series of data for a few seconds, and further, we again stored the next batch of the data in the subsequent block. Finally, we used the stored as well as the generated data for monitoring various industrial tasks.

In this work, we ceased our research in optimizing the data. We abstained from making BC operations faster. We refrained from comparing the ARIMA model with other models and focused on the data-centric selection of the transaction and generating time series. In the extended work, we include the same and study the effects of the changing prediction model. Apart from this, the limitation of S&G blocks is that it shows unexpected behavior if the time series shows abrupt variation. Further, S&G blocks adds additional time complexity to BC as it utilizes the ARIMA model to predict the data. In our future work, we will also try to resolve this issue.

VI. Acknowledgement

The first author of the paper is awarded by the Prime Minister Research Fellowship for implementing the work reported in this paper.

References

[1] C. Qiu, F. R. Yu, H. Yao, C. Jiang, F. Xu, and C. Zhao, "Blockchain-Based Software-Defined Industrial Internet of Things: A Dueling Deep Q-Learning Approach," *Internet of Things Journal*, vol. 6, no. 3, pp. 4627–4639, 2018.

[2] W. Liang, Y. Fan, K. C. Li, D. Zhang, and J. L. Gaudiot, "Secure Data Storage and Recovery in Industrial Blockchain Network Environments," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 10, pp. 6543–6552, 2020.

[3] I.-T. Chou, H.-H. Su, Y.-L. Hsueh, and C.-W. Hsueh, "BC-Store: A Scalable Design for Blockchain Storage," in *proceedings of the* 2^{nd} *International Electronics Communication Conference*, 2020, pp. 33–38.

[4] Y. K. Purba, D. Saepudin, and D. Adytia, "Prediction of Sea Level by Using Autoregressive Integrated Moving Average (ARIMA): Case Study in Tanjung Intan Harbour Cilacap, Indonesia," in *proceedings of the* 8^{th} *International Conference on Information and Communication Technology (ICoICT)*. IEEE, 2020, pp. 1–5.

[5] Y. Xu, "Section-Blockchain: A Storage Reduced Blockchain Protocol, The Foundation of an Autotrophic Decentralized Storage Architecture," in *proceedings of the* 23^{rd} *International Conference on Engineering of Complex Computer Systems (ICECCS)*. IEEE, 2018, pp. 115–125.

[6] F. Zerka, V. Urovi, A. Vaidyanathan, S. Barakat, R. T. Leijenaar, S. Walsh, H. Gabrani-Juma, B. Miraglio, H. C. Woodruff, and M. Dumontier, "Blockchain for Privacy Preserving and Trustworthy Distributed Machine Learning in Multicentric Medical Imaging (C-DistriM)," *IEEE Access*, vol. 8, pp. 183 939–183 951, 2020.

[7] S. Kantesariya and D. Goswami, "Determining Optimal Shard Size in a Hierarchical Blockchain Architecture," in *proceedings of the International Conference on Blockchain and Cryptocurrency (ICBC)*. IEEE, 2020, pp. 1–3.

[8] M. Singh, G. S. Aujla, A. Singh, N. Kumar, and S. Garg, "Deep-Learning-Based Blockchain Framework for Secure Software-Defined Industrial Networks," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 1, pp. 606–616, 2021.

[9] S. Ali, G. Wang, B. White, and R. L. Cottrell, "A Blockchain-Based Decentralized Data Storage And Access Framework For Pinger," in *proceedings of the* 17^{th} *International Conference on Trust, Security and Privacy in Computing and Communications/*12^{th} *International Conference on Big Data Science and Engineering (TrustCom/BigDataSE)*. IEEE, 2018, pp. 1303–1308.

[10] K. Lei, M. Du, J. Huang, and T. Jin, "Groupchain: Towards a Scalable Public Blockchain in Fog Computing of IoT Services Computing," *Transactions on Services Computing*, vol. 13, no. 2, pp. 252–262, 2020.

[11] S.-Y. Wang, Y.-J. Hsu, and S.-J. Hsiao, "Integrating Blockchain Technology for Data Collection and Analysis in Wireless Sensor Networks with an Innovative Implementation," in *International Symposium on Computer, Consumer and Control (IS3C)*. IEEE, 2018, pp. 149–152.

[12] M. Taghavi, J. Bentahar, H. Otrok, and K. Bakhtiyari, "A Blockchain-Based Model for Cloud Service Quality Monitoring," *IEEE Transactions on Services Computing*, vol. 13, no. 2, pp. 276–288, 2020.

[13] M. R. Chowdhury, S. Tripathi, and S. De, "Adaptive Multivariate Data Compression in Smart Metering Internet of Things," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 2, pp. 1287–1297, 2021.

[14] N. Aunsri and P. Taveeapiradeecharoen, "A Time-Varying Bayesian Compressed Vector Autoregression for Macroeconomic Forecasting," *IEEE Access*, vol. 8, pp. 192 777–192 786, 2020.

[15] P. Gangwar, A. Mallick, S. Chakrabarti, and S. N. Singh, "Short-Term Forecasting-Based Network Reconfiguration for Unbalanced Distribution Systems With Distributed Generators," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 7, pp. 4378–4389, 2020.

Riya Tapwal is a Ph.D. Research Scholar in the Department of Computer Science and Engineering, Indian Institute of Technology Kharagpur, India. She has completed her M.Tech degree in Mobile Computing from the National Institute of Technology, Hamirpur, India, in 2020. Prior to that, she received the B.Tech degree in Computer Science and Engineering in 2018 from University Institute of Information Technology, Himachal Pradesh University, Shimla, India. The current research interests of Riya include Wireless Networks, Fog Computing, Industrial Internet of Things, and BC.

Pallav Kr. Deb is a Ph.D. Research Scholar in the Department of Computer Science and Engineering, Indian Institute of Technology Kharagpur, India. He received his M.Tech degree in Information Technology from Tezpur University, India, in 2017. Prior to that, he has completed the B. Tech degree in Computer Science from the Gauhati University, India, in 2014. The current research interests of Mr. Deb include UAV swarms, THz Communications, Internet of Things, Cloud Computing, Fog Computing, and Wireless Body Area Networks. His detailed profile can be accessed at https://pallvdeb.github.io

Sudip Misra (SM'11) is a Professor at IIT Kharagpur. He received his Ph.D. degree from Carleton University, Ottawa, Canada. Prof. Misra is the author of over 350 scholarly research papers. He has won several national and international awards, including the IEEE ComSoc Asia Pacific Young Researcher Award during IEEE GLOBECOM 2012, the INSA NASI Fellow Award, the Young Scientist Award (National Academy of Sciences, India), Young Systems Scientist Award (Systems Society of India), and Young Engineers Award (Institution of Engineers, India). He has also been serving as the Associate Editor of the IEEE Transactions on Mobile Computing, the IEEE Transactions on Vehicular Technology, the IEEE Transactions on Sustainable Computing, the IEEE Systems Journal, and the International Journal of Communication Systems. Dr. Misra has 11 books published by Springer, Wiley, and World Scientific. For more details, please visit http://cse.iitkgp.ac.in/∼smisra.

Surjya Kanta Pal completed his graduation from Government Engineering College, Jalpaiguri, West Bengal in 1991, M.Tech from IIT Kanpur in 1993, and Ph.D. from IIT Kharagpur in 1999, in Mechanical Engineering. Professor Pal has done 3 years of Postdoctoral research at The University of Sheffield, UK. His current research interests include Friction Stir Welding, Industry 4.0, Modelling, and simulation of manufacturing processes. He started teaching at IIT Guwahati from July 2002 to July 2004 and then joined IIT Kharagpur, where he is presently working as a Professor in the Department of Mechanical Engineering. Professor Pal has published 264 research articles which include 156 International Journal Papers, 15 International Book Chapters, and 90 Conference articles. He has also filed 11 patents, out of which five are in the area of Industry 4.0.