RFID Data Reliability Optimiser Based on Two Dimensions Bloom Filter

Siti Salwani Yaacob, Hairulnizam Mahdin, Shahreen Kasim
Faculty of Computer Science and Information Technology,
Universiti Tun Hussein Onn Malaysia, Parit Raja, Batu Pahat, Johor, Malaysia.
hairuln@uthm.edu.my

***Abstract*—Radio frequency identification (RFID) is a flexible deployment technology that has been adopted in many applications especially in supply chain management. RFID system used radio waves to perform wireless interaction to detect and read data from the tagged object. However, RFID data streams contain a lot of false positive and duplicate readings. Both types of readings need to be removes to ensure reliability of information produced from the data streams. In this paper, a single approach, which based on Bloom filter was proposed to remove both dirty data from the RFID data streams. The noise and duplicate data filtering algorithm was constructed based on bloom filter. There are two bloom filters in one algorithm where each filter holds function either to remove noise data and to recognize data as correct reading from duplicate data reading. Experimental results show that our proposed approach outperformed other existing approaches in terms of data reliability.**

***Index Terms*—RFID; Bloom Filter; False Positive Reading; Noise; Duplicate Reading; Redundancy.**

I. INTRODUCTION

Radio Frequency Identification (RFID) technologies has been broadening applied in many applications such as supply chain management [1], healthcare management system[2], public transport system [3], and library management system [4]. RFID is a technology that uses radio waves to transfer detecting information between reader and tagged object from a distance without line of sight.

A typical RFID system consist of tag, reader, middleware and application [5]. A tag is a package that can be attached to the physical object. While reader, also known as interrogator will communicate with the tag by transmitting radio waves. The tag then sends the radio signal back to the reader and sends to the server for further analysis and processing of data. The RFID data generally in the form of reader_id, tag_id and timestamp [6].

As the tag is unique, the readers are able to detect the information of RFID tag items from a certain location at different times. While for RFID data are generated quickly and automatically [7], it can be used for real-time monitoring [7] or accumulated for object tracking [8]. Since the advancement of RFID technology has been broadened in many applications, RFID system still suffers from several conflicts that prevent it being implemented by the industry. The crucial part by implementing RFID system is to deal with the flood of data generated by the reader [9].

For example, the Coca-Cola Company produced more than a billion bottles per day. An effective retail in-store logistics at Coca-Cola Company is necessary to ensure high product availability at minimum operating cost [10]. The unreliable data reading such as noise, duplicate reading that were produced by RFID reader has become the primary factor limiting the widespread adoption of this technology. In that case, it is compulsory to filter the original data to maintain its reliability of data reading for business process. This is because, a small decrease of effective read rate will reduce the accuracy and reliability of further data process [11]. Therefore, it is necessary to implement filtering technique to provide quality RFID data.

RFID system generates massive amount of data which often contains errors and unimportant readings [12]. The errors can be classified into three types of error readings that caused by hardware in RFID system [13]; noise reading, missed reading and duplicate reading. Noise data or false positive reading happens when reader detects a tag, but it is not in the reader's field [14]. This happens because of tag collisions [15] and reader collisions [16]. Reader collision occurs when the signals from two or more readers overlap. Tag collision occurs when many tags are present in a small area. Although the occurrence of noise reading is low [17], noise reading can mislead important business decision-making. Missed reading or false negative happens when the tag is considered to be absent when it is present in the reader vicinity. However, missed reading can be solved by multi-reading periodically [6]. While data redundancy or duplicate readings occurred when the similar RFID data readings generates repeatedly due to multiple readings cycle and multiple readers implemented to cover specific area [18]. The duplicate readings problem is recognized as a serious issue in RFID and sensor networks [19]. Therefore, it is necessary to implement filtering technique to provide quality RFID data.

II. RELATED WORKS

Previous research on RFID data filtering, most of the approaches treated duplicate reading issue and noise reading issue in separate problem.

A. Window Based Approach

One of the approaches that used to filter data is by implementing window based approach. There are two types of window-based approach discovered; sliding window and landmark window [20]. Figure 1 portrayed the sliding window and landmark window. Sliding window is a window with certain, size that moves with specific time. While landmark window is a window that move with time.

Bai et al., [17] has conducted a research that focuses on reducing noises, false positive readings, false positive readings and duplicate readings. To overcome noise problems, they have proposed a technique where any tag

below the threshold value will be discard. Else, if the number of the readings with the same tag EPC values appears equal to or above threshold, then the EPC value is not noise and need to be forward for further processing. While in the duplicate elimination process, they proposed a technique by keeping a sliding window of size exist another reading in the window with the same key, then it issue (max_distance) in time from the previous reading with the same key. Then this reading is considered a new reading.

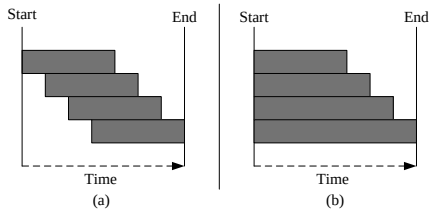

Figure 1: (a) Sliding window, and (b) Landmark Window

Tyagi, Ansari and Khan [21] proposed dynamic threshold sliding-window based approach (DTSW) to reduce false positive reading and false negative reading by adding time scheduling on threshold value. This means after a period of time, the data will recognize as a tag or else it will consider as a noise. They also proposed a technique to inspect data format of RFID and associate values such as header information by introducing CheckEPCHeader(). After recognizing data as a tag, CheckEPCHeader() will inspects all tags either it is a real tag or it is a noise.

Mahdin & Abawajy [22] proposed an approach denoise and duplicate elimination algorithm (DDSW) to filter noise and duplicate readings in one algorithm that make one filter. This approach used number of occurrence per time as the basis of filtering data in the data stream. Thus, eliminating one of the filters reduces the time required to filter duplicate readings. Then the author focuses their works to filter duplicate reading filtering in RFID data stream [23]. However, this approach has low false positive rates, which illustrates the improved correctness of the filtering process. It is more efficiency in terms of time and memory usage.

K. Hu et al., [6] has proposed HTB algorithm as a solution to filter noise data in RFID data streams based on sliding window approach. They solve the problem of sliding window when the size of data getting bigger, the RFID reading has not been outputted until expiring in sliding window. Although the research focuses on filtering on false positive reading, they are actually filtering the duplicate reading by applying time tolerance threshold in hash table technique.

B. Bloom Filter Approach

Another approach that used to filter data in RFID is Bloom filter approach [24]. A Bloom filter is a space-efficient probabilistic data structure that tells either the data is in the set or not.

Figure 2 shows a basic structure design of bloom filter. It represents data in its bit array of size m using k number of hash functions. Whenever the data has been hash, all bits in array that are initially set to 0 will be substitute to 1. The basic bloom filter supports two operations: test and add [25].

Based on Figure 3, the picture visualizes how a bloom filter operates. The bloom filter simply adds data such Tag X and Tag Y in the bloom filter. To test if an item is stored in the filter, again we feed it to the same k hash function. If one or more of these bits is not set then the queried element is definitely not present in the filter. As in Figure 3, if any of the bits are 0, for example Tag Z, then the string definitely does not exist in the filter. If all of the bits are 1, there is probability that the string exists in the filter. Generally, bloom-filter has been used to filter duplicate data. Removal and deletion is not allowed in normal bloom filter. This is for the reason that a single counter in bloom-filter can be hashed number of times by different data. Turning counter to 0 will disturb other data that is not involved in the deletion.

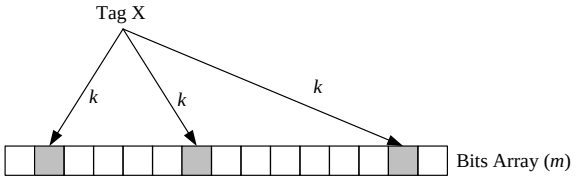

Figure 2: Basic Structure of Bloom Filter

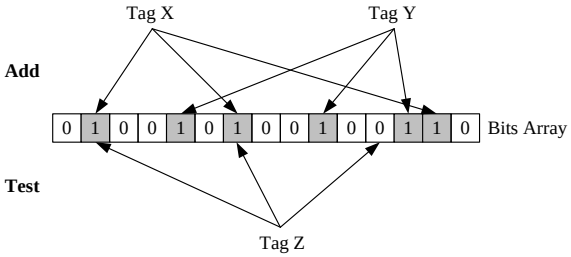

Figure 3: The Operation of Bloom Filter

Previously research shows that bloom filter has been extended to allow deletion and implemented many research in order to filter duplicate readings and noise in RFID data streams. Deng & Rafiei [26] has proposed an algorithm (DSBF) using bloom filter approach to eliminate duplicate data in the data stream applications. The bits in the regular bloom filter are change into cells consisting one or more bits. In order to eliminate old data, each cell is set to the maximum value and decrement the values of randomly selected cells whenever data arrives. However, this approach is still produce false positive errors and false negative errors.

Mahdin & Abawajy [27] has proposed an approach to filter duplicate readings in RFID based on Bloom filter. The proposed approach stores the information such as time of tag detected in the filter units to compare which reader a tag belongs to. However, the algorithm might complicate the selection reading cycle and the time of clearing data in the filter. Thus, the algorithm may delete the true reading and cannot be used in the filter.

Lee & Chung [28] has extends original bloom filter to support sliding window and proposed time bloom filter algorithm (TIBF) in order to detect duplicate data reading on RFID data streams. As the process of filtering duplicates were takes place at the server side, a lot of bandwidth wasted during transferring the duplicates. Hence, three algorithms; bloom filter, time bloom filter and time interval bloom filter were proposed to eliminate each duplicates data arrive. Time interval bloom filter were used in fault detection and elimination and this algorithm need more space than the time bloom filter. In this research, the time interval bloom filter need more space compare to time bloom filter. Time bloom filter depends on time information to check whether the data is duplicates or not. Even though it does not create false

negative errors, the major problem of this technique is bottleneck will occurs, as the data has to pass through this module in the server side.

C. Decision Support Model Approach

Decision support model is a structured process where it consists of several parts of manageable processes to filter RFID data stream. For examples, Majority voting approach and smooth filtering approach are examples of decision support model where each technique consists of several parts of process to filter data. Majority voting is an approach where it is efficiently dealing when both readers agree or disagree with each other in identifying the tag.

Tu & Piramuthu [29] believed that by using majority-voting technique would reduce low false read rates in supply chain. The purpose of majority voting is to detect false negative reading by identifying the presence of data reading in the data stream. They proposed Three Tag-Two Readers Model (TTTR) where all three tags are embedded in their object of interest. Each reader will identify either the presence of the object of interest either it is present or absent. There are several rules that have to be followed in order to detect the presence of the object of interest. However, even the purpose of this model is to reduce false read rates, it does not perform well as in the result there is slight increase false negative error by correcting the true- negative error.

C. M. Wu et al., [30] has proposed data filtering strategy using cluster based approach (CBA) to filter noise reading and duplicate reading. Figure 2.13 shows the cluster architecture. Cluster is a framework where RFID readers were grouped into several clusters according to its location of tagged objects. In this research, the readers were grouped in order to assist data cleaning process. This research, they focus on removing noise reading and duplicate reading by applying many techniques in order to decide whether the reading is duplicate reading or noise reading. The sliding window technique was applied to detect duplicate readings and noise readings. They also applied hash method to minimize search time of the sliding window. Besides, the majority voting approach was also applied to detect false positive readings between RFID readers.

III. FILTERING DESIGN

The strategy to remove duplicate readings and noise readings in RFID is by implementing the threshold value concept as [6], [21]. Reading that did not pass the threshold value is considered as noise. While the reading that is more than the threshold value is considered as duplicate.

The proposed algorithm is based on Bloom filter that consists of two Bloom filters; duplicate Bloom filter (DBF) and noise Bloom filter (NBF). There two types of Bloom filters considered as Two Dimension Bloom Filter (2BF).

According to [23], the size of hash function k is set to 7 and the size of m is set to 9 times bigger than the number of data n to get the best results. Threshold value was implemented in NBF to filter noise readings in the data stream. In this research, the threshold value is set to 7.Based on Figure 4, the algorithm consists of duplicate bloom filter (DBF) and noise bloom filter (NBF). The input for DNA is the tag identification reading (READING). In step 1-4, the algorithm checks the time to remove all readings in the filters. When the time is met, the counter position in both filters will be clear and reset to zero. Next, in step 5, as the reader receives tag identification readings for each tag, then the READING is sent to the filter. For steps 6-19, each incoming READING will be hashed and its condition is checked in the DBF. If the READING is not the DBF, this shows that the reading is not the correct reading. Hence increase the counter position of hashed reading in the NBF. If the READING is already in the DBF, then state that the READING as duplicate and filter next READING. While in step 20-32, if the counter position in the NBF is more or equal to the threshold value, this means that the READING is not noise and the READING is a true reading. Then all the hashed counter positions in NBF are copied, and mapped to DBF.

```
INPUT: READING
BEGIN

1:   IF (Time == True) THEN  //clearing counter when the time
     comes
2:       NBF[] = {0}
3:       DBF[]={0}
4:   ENDIF
5:   FOREACH (incoming READING) DO
6:       FOR(i=1<k)
7:           position = hash[i](READING)
8:           CounterNum[i]=position;
9:               IF (DBF[position] ==0) AND (NBF[position]==0)
10:                  THEN
11:                      DBF[position] =1
12:                      NBF[position] =1
13:              ELSEIF (DBF[position] ==0) AND (NBF[position]>0)
14:                  THEN
15:                      NBF[position] = NBF[position] + 1
16:                  ELSE
17:                      OUTPUT READING as DUPLICATE
18:              ENDIF
19:   EXIT(FOR) // go back to step 1 – new reading
20:      FOR(i=1<k)
21:          position=CounterNum[i]
22:          IF(NBF[position]>THRESHOLD
23:              NotNoise++
24:          ENDIF
25:      ENDFOR
26:          IF (NotNoise>=(k/2)) //if half or more counter have
     count over threshold
27:              OUTPUT READING IS CORRECT
28:          ENDIF
29:       FOR(i=1<k)  //copy NBF value to DBF value
30:          position=CounterNum[i]
31:          DBF(position)=1
32:       ENDFOR
33:  END FOR
```

Figure 4: Two Dimension Bloom Filter Algorithm (2BF)

IV. EXPERIMENT DESIGN

In this research the simulation of algorithms was developed using C++. While the RFID data was generated using Poisson distribution as in [31], Poisson distribution gives the random number of independent events occurring in a fixed time [32].

Two data sets were created and each set of data contains several samples. Each sample contains 10 tags and each tag will repeat for 10 cycles. The first set is focused different arrival rates. A set of data with a different noise ratio was created in the second set data. For the first set of data, the arrival rate for each sample is set of 5 readings per second, 10 readings per second, 15 readings per second, 20 readings per second and 50 readings per second. In this sample data is set to 10 % noise rate. In the next sample data, the arrival rate is

set to 50 readings per second. The noise ratios applied are 10%, 20%, 30%, 40% and 50% in each sample. The 2BF algorithm was compared with HTB algorithm [6], DTSW algorithm [21] and CBA algorithm [29] for performance analysis.

V. Results and Analysis

A. Different Arrival Rates

Table 1 and Figure 5 shows the results of time taken to process data under different arrival rates. 2BF and HTB took the least time to filter RFID readings. While, DTSW and CBA took longer time to process the data. This is because it needs to go through along the windows to read the readings that become bigger with the increasing of arrival reading for every time new incoming readings arrived. Unlike 2BF and HTB, these algorithms do not have to go through along the window to check duplicate readings and false positive readings. The function of hashing the RFID data and checking its existence in the filter is a constant operation.

Table 1
Time Execution under Different Arrival Rate

Arrival Rate	2BF	HTB	DTSW	CBA
5	0.042 s	0.11 s	0.02 s	0.02 s
10	0.093 s	0.02 s	0.09 s	0.08 s
15	0.181 s	0.03 s	0.25 s	0.26 s
20	0.266 s	0.04 s	0.52 s	0.54 s
50	0.945 s	0.01 s	4.03 s	4.05 s

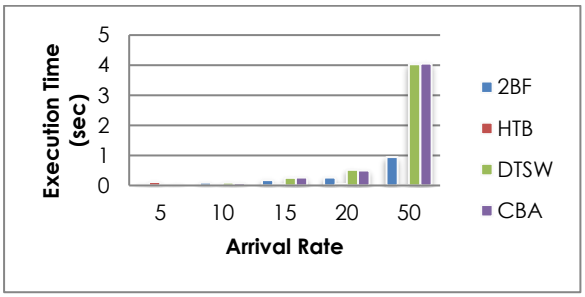

Figure 5: Processing time under different arrival rate

Table 2 and Figure 6 shows the result of the successful duplicates data filtered under different arrival reading rates. The RFID data sets with 10 % noise ratio does not affect the performance of algorithm to filter duplicates data. This means the RFID data with different arrival rates does not affect the performance of the algorithms to filter duplicates data.

Table 2
Duplicate Data Filtered under Different Arrival Rate

Arrival Rate	2BF	HTB	DTSW	CBA
5	100 %	98.7 %	98.9 %	99.5 %
10	100 %	99.6 %	99.4 %	99.7 %
15	99.9 %	99.8 %	99.6 %	99.7 %
20	99.98 %	99.8 %	99.7 %	99.9 %
50	99.9 %	99.9 %	99.8 %	99.95%

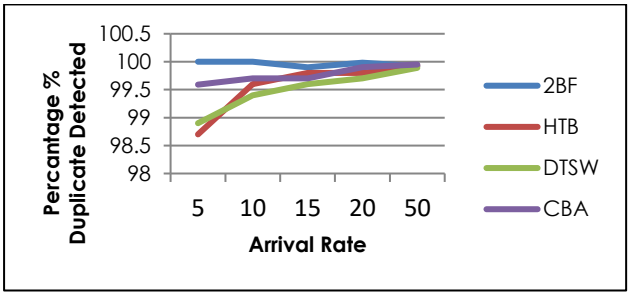

Figure 6: Duplicate Data Detected with Different Arrival Rate

Table 3 and Figure 7 shows the result of successful noise data filtered under different arrival reading rates. 2BF performs better than HTB. HTB produced small false positive rate after filter process. But both algorithms filtered data nearly 100%. The 10% noise data does affect DTSW and CBA. However, DTSW and CBA produce small false positive rates when the arrival reading is smaller.

Table 3
Noise Data Filtered with Different Arrival Rate

Arrival Rates	2BF	HTB	DTSW	CBA
5	100 %	98.9 %	91.58 %	97 %
10	100 %	99.15 %	95.37 %	98.2 %
15	100 %	99.59 %	97.21 %	98.9 %
20	100 %	99.69 %	97.94 %	99.2 %
50	99.9 %	99.78 %	99.05 %	99.6 %

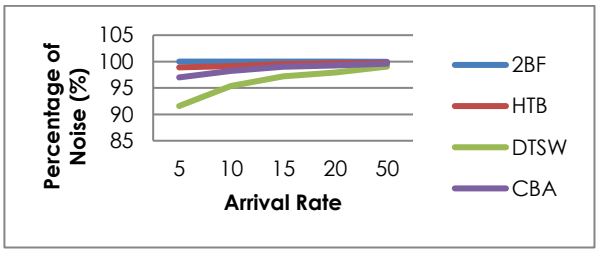

Figure 7: Percentage of Noise Detected Data under Different Arrival Rate

B. Different Noise Ratio

In this section, the arrival-reading rate is set to 50. Table 4 and Figure 8 shows the result of time taken to process data under different noise ratio. As in Table 4, the HTB also took the least time to filter data especially when the noise rates getting higher compared to 2BF. HTB used hashing method to reduce the searching time in window. While DTSW and CBA took longer time to filter false positive reading and duplicate readings. This is because DTSW has to go through along window to read the data before filter process.

Table 4
Time Executions with Different Noise Ratio

Noise (%)	2BF	HTB	DTSW	CBA
10	0.74 s	0.07 s	4.5 s	4.19 s
20	2.42 s	0.12 s	16.42 s	16.36 s
30	3.21 s	0.12 s	20.27 s	20.18 s
40	3.5 s	0.11 s	23.19 s	23.00 s
50	3.84 s	0.11 s	24.56 s	24.45 s

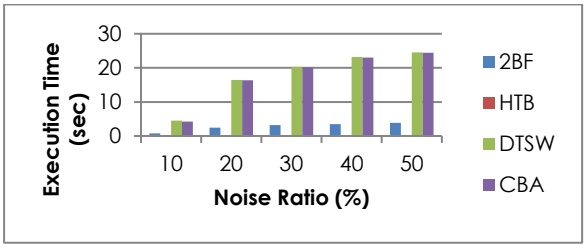

Figure 8: Processing Time with different Noise Ratio

Table 5 and Figure 9 shows the data filtered under different noise ratio. The proposed approach 2BF filtered duplicate data 100%. HTB has filtered data nearly 100%. There is slight decrease in DTSW and CBA where the algorithms unable to filter duplicate readings correctly in large arrival rate with higher noise ratio. The weakness of using sliding window approach (DTSW), is that the size cannot be large enough to filter data correctly. When the readings are scattered, there are readings that unable to be compared with other RFID data reading as it need to be done to filter duplicate reading. This left some reading in the window.

Table 5
Duplicate Data Filtered under Different Noise Ratio

Noise (%)	2BF	HTB	DTSW	CBA
10	100	99.93 %	99.89 %	99.95 %
20	100	99.95 %	89.93 %	90 %
30	100	99.94 %	75.62 %	76 %
40	100	99.92 %	56.5 %	57 %
50	100	99.9 %	30 %	30 %

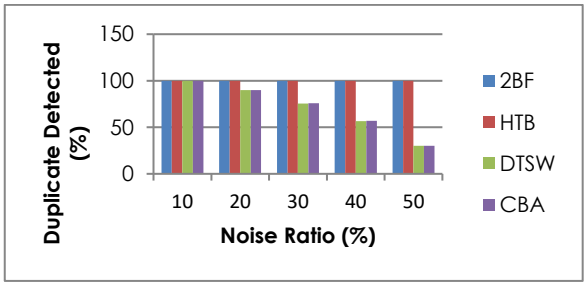

Figure 9: Percentage of Duplicate Detected Data under Different Noise Ratio

Table 6 and Figure 10 shows the data filtered under different noise ratio. 2BF and HTB filtered noise data nearly 100%. The performance of DTSW and CBA are decreasing when the noise ratio are higher. This shows that these algorithms effects on higher noise ratio and higher arrival rates. The size of sliding window needs to be large enough to go through the reading. As the window is large, the process to compare data to filter noise become complicated and this process might leave some reading in the window. Hence, DTSW and CBA are not suitable to deals with high volume reading per second.

Table 6
Noise Data Filtered under Different Noise Ratio

Noise (%)	2BF	HTB	DTSW	CBA
10	99.96 %	99.78 %	99.05 %	99.6 %
20	99.95 %	99.9 %	59.65 %	59.85 %
30	99.96 %	99.93 %	43.1 %	43.23 %
40	100 %	99.95 %	34.82 %	40 %
50	100 %	99.96 %	29.84 %	30 %

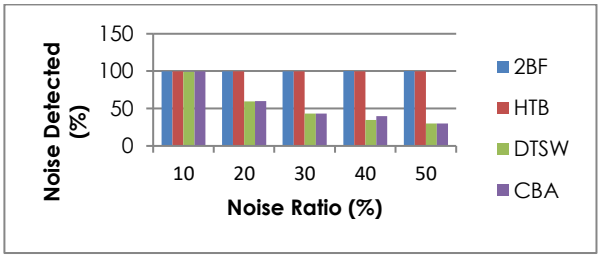

Figure 10: Percentage of Noise Detected Data under Different Noise Ratio

C. Analysis Summary

Table 7 shows the comprehensive summary on the performance of the algorithms. The time execution of HTB algorithm is faster compared to the 2BF algorithm. This is because each incoming new reading will be hashed in the table. While 2BF needs to be compared with noise bloom filter and duplicate bloom filter to identify false positive readings and duplicate readings. Even though HTB algorithm took the least time to process data, the 2BF algorithm completely filters duplicate readings and false positive readings. For 2BF, each type of reading has its corresponding filter. This is the reason that makes the 2BF more efficient.

Unlike HTB, every new incoming reading will be hash in the table. The hash table will update and count the incoming reading. When the time interval of incoming reading is greater than the time tolerance threshold value, the reading will be mark as duplicate reading. Else, it is false positive reading.

For DTSW algorithm and CBA algorithm has used sliding window approach to filter duplicate readings and false positive readings. These algorithms took longer time to process the data. For DTSW, it has to go through along the window that becomes bigger with the increasing of reading, every time new readings came. While CBA has to check whether the reading is in the sliding window and to check whether the reading has been outputted. The reading then has to go through noise checking process.

Table 7
Performance Evaluations of the Algorithms

	Performance Evaluation			
	Accuracy		Efficiency	
	Different Arrival Rates	Different Noise Ratio	Different Arrival Rates	Different Noise Ratio
2BF	Complete	Complete	More Efficient	More Efficient
HTB	Almost	Almost	Efficient	Efficient
DTSW	Almost	Not Accurate	Efficient	Less Efficient
CBA	Almost	Not Accurate	Efficient	Less Efficient

The weakness of using sliding window approach is that the size cannot be large enough to filter data correctly. When the readings are scattered, there are readings that are unable to be compared with other readings especially when arrival reading rate is higher with higher noise ratio. This will left some reading in the window. The performance result of DTSW and CBA from the experiment with different noise ratio clearly shows that both algorithms are not suitable to perform data filter especially with higher arrival reading rate and noise ratio.

VI. CONCLUSION

The 2BF is developed to filter noised readings and duplicates readings in the data stream. Even though the possibilities of practical implementations for Two Dimensions Bloom Filter has been shown, there are still some essences to be explored for the reliability of proposed algorithm. Therefore, recommendations regarding further development of this research work are needed. Comparisons with other techniques that can incorporate knowledge in the form of constraints on different set data is suggested so that the data can be better analysed.

ACKNOWLEDGEMENT

The authors would like to thank Universiti Tun Hussein Onn Malaysia (UTHM) for the support given to this research and also providing financial assistance under the Exploratory Research Grant Scheme (ERGS) grant no E054.

REFERENCES

[1] Yan P., Yang W., Tan B., and Yu B. 2015. RFID Solution to Improving Ammunition Supply Chain Management. LISS 2013, *Springer*. 1163–1168.

[2] Hu L., Ong D. M., Zhu X., Liu Q., and Song E.. 2014. Enabling RFID Technology for Healthcare: Application, Architecture, and Challenges, Telecommunication Systems. 1–13.

[3] Frick J. 2014. Improving Transport and Accessibility through New Communication Technologies. Advances in Production Management Systems. Innovative and Knowledge-Based Production Management in a Global-Local World, *Springer*, 2014. 572–578.

[4] Kushal K. S., Kadal H. M., Chetan S., and others. 2012. Design and Implementation of a RFID Based Prototype SmArt LibrARY (SALARY) System Using Wireless Sensor Networks. Advances in Computer Science, Engineering & Applications, *Springer*, 2012. 499–505.

[5] Kour R., Karim R., Parida A., and Kumar U. 2014. Applications of Radio Frequency Identification (RFID) Technology with eMaintenance Cloud for Railway System. International Journal of System Assurance Engineering and Management. 5(1): 99–106.

[6] Hu K., Li L., and Lu Z. 2013. A Cleaning Method of Noise Data in RFID Data Streams. *3rd International Conference on Consumer Electronics, Communications and Networks (CECNet)*, 2013. 1–4.

[7] Zhong R. Y., Huang G. Q., Dai Q. Y., and Zhang T. 2014. Mining SOTs and Dispatching Rules From RFID-Enabled Real-Time Shopfloor Production Data. *Journal of Intelligent Manufacturing*. 25(4): 825–843.

[8] Wang L., Gu T., Xie H., Tao X., Lu J., and Huang Y. 2014. A Wearable RFID System for Real-Time Activity Recognition Using Radio Patterns. Mobile and Ubiquitous Systems: Computing, Networking, and Services, *Springer*. 370–383.

[9] Kwon K., Kang D., Yoon Y., Sohn J.-S., and Chung I.-J. 2014. A real time process management system using RFID data mining. *Comput. Ind.* 65(4): 721–732.

[10] Metzger C., Thiesse F., Gershwin S., and Fleisch E. 2013. The impact of false-negative reads on the performance of RFID-based shelf inventory control policies. *Comput. Oper. Res.*, 40(7): 1864–1873.

[11] Liu H., Liu K., Gong W., Liu Y., and Chen L. 2014. Wonder: Efficient Tag Identification for Large-Scale RFID Systems. *2014 IEEE International Conference on Distributed Computing in Sensor Systems (DCOSS)*, 2014. 27–134.

[12] Liu Y., Fawang H. A. N., TJun. A. N., and ZHANG. H. 2015. An Efficient RFID Data Cleaning Method Based on Wavelet Density Estimation. *Journal of Digital Information Management*. 13(1): 11.

[13] Leema A. A. and Hemalatha M. 2013. Proposed Prediction Algorithms Based on Hybrid Approach to Deal with Anomalies of RFID Data in Healthcare. Egyptian Informatics Journal. 14(2): 135–145.

[14] Ma J., Sheng Q. Z., Xie D., Chuah J. M., and Qin Y. 2014. *Efficiently Managing Uncertain Data in RFID Sensor Networks.* World Wide Web. 1–26.

[15] Fritz G., Beroulle V., Nguyen M. D., Hély D.. 2011. RFID System On-line Testing based on the evaluation of the Tags Read-Error-Rate. *Journal of Electronic Testing*. 27(3): 267–276.

[16] Zhang L., Gandino F., Ferrero R., Montrucchio B., and Rebaudengo M. 2013. Trade-off Between Maximum Cardinality of Collision Sets and Accuracy of RFID Reader-to-Reader Collision Detection. *EURASIP Journal on Embedded Systems*. 2013(1): 1–14.

[17] Bai Y., Wang F., and Liu P. 2006. Efficiently Filtering RFID Data Streams. CleanDB Workshop. 50–57.

[18] Mahdin H.. 2014. A Review on Bloom Filter Based Approaches for RFID Data Cleaning. *Proceedings of the First International Conference on Advanced Data and Information Engineering (DaEng-2013)*, Springer Singapore. 79–86.

[19] Vahdati F., Javidan R., and Farrahi A.. 2010. A New Method for Data Redundancy Reduction in RFID Middleware. in *2010 5th International Symposium on Telecommunications (IST)*, 2010, 175–180.

[20] Rui W., Guoqiong, L. and Guoqiang D. 2014. Filtering Redundant RFID Data Based on Sliding Windows. Management of e-Commerce and e-Government (ICMeCG), International Conference 2014. 187–191.

[21] Tyagi S., Ansari A. Q., and Khan M. A. 2010. Dynamic Threshold Based Sliding-Window Filtering Technique for RFID Data. Advance Computing Conference (IACC), 2010 IEEE 2nd International, 2010. 115–120.

[22] Mahdin H. and Abawajy J. 2009. An Approach to Filtering RFID Data Streams. Parallel Architectures, Algorithms, and Networks. International Symposium on, Los Alamitos, CA, USA. 742–746.

[23] Mahdin H. and Abawajy J. 2010. An Approach to Filtering Duplicate RFID Data Streams. U- and E-Service, *Science and Technology*. *Springer Berlin Heidelberg*. 124: 125–133.

[24] Bloom B. H. 1970. Space/time Trade-Offs in Hash Coding With Allowable Errors. *Communications of the ACM*. 13(7): 422–426

[25] Dillinger P. C. and Manolios P. 2004. Bloom Filters in Probabilistic Verification. *Formal Methods in Computer-Aided Design*. 367–381.

[26] Deng F. and Rafiei D. 2006. Approximately Detecting Duplicates for Streaming Data Using Stable Bloom Filters. in *Proceedings ACM SIGMOD International Conference on Management of Data*, New York, USA. 25–36.

[27] Mahdin H. and Abawajy J. 2011. An Approach for Removing Redundant Data from RFID Data Streams. *Sensors*. 11(12): 9863–987.

[28] Lee C.-H. and Chung C.-W. 2011. An Approximate Duplicate Elimination in RFID Data Streams. *Data & Knowledge Engineering*. 70(12): 1070–1087.

[29] Tu Y.-J. and Piramuthu S. 2011. A Decision-Support Model For Filtering RFID Read Data In Supply Chains. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*. 41(2): 268–273.

[30] Wu C.-M., Cheng S.-S., and Chang R.-S. 2013. A Data Filtering Strategy Using Cluster Architecture In Radio Frequency Identification System. *International Journal of Radio Frequency Identification Technology and Applications*. 4(2): 149–161.

[31] Sarac A., Absi N., and Dauzere -Peres S. 2015. Impacts of RFID technologies on supply chains: a simulation study of a three-level supply chain subject to shrinkage and delivery errors. *European Journal of Industrial Engineering*. 9(1): 27–52.

[32] Forbes C., Evans M., Hastings N., and Peacock B. 2011. Statistical distributions. *John Wiley & Sons*.