Compression Scheme for RFID Equipment

Yair Wiseman
Computer Science Department
Holon Institue of Technology
Holon, Israel
wiseman@cs.biu.ac.il

***Abstract*— EPC tags for RFID were designed to accommodate a diversity of many potential possibilities. The generously allocations for the each of the fields in the EPC tags caused these tags to be inefficient, because their size is too large. To facilitate shorter EPC tags, an effective compression scheme for EPC tags is suggested with the aim of transmission of fewer bits by RFID equipment.**

Keywords— Electronic Product Code (EPC); RFID; Compression; Arithmetic Coding

I. INTRODUCTION

The aim of Electronic Product Code (EPC) is suggesting a unanimous identifier offers a sole code for every item anywhere in the world [1]. EPC has a defined format [2], which is freely available. Actually, the main objective of EPC tag data standard is an efficient binary format fitting for storing an EPC identifier effectively within RFID tags. The EPC is intended to meet the needs of a range of productions, while guaranteeing matchlessness for every single object tag.

Since 2003 Walmart has requested to put EPC tag labels on some of its products [3]. Some branches of the US government like The Department of Defense and The Department of Homeland Security also began to use EPC tags more than a decade ago for various purposes like warehouse supply chain, tracking armaments, smart borders etc. [4]. These wide use have created growth within the RFID industry and have lowered the price tag for RFID deployment [5].

EPCs are not solely intended for utilization along with RFID equipment [6]. EPC data can as a matter of fact to be composed based on reading of optical data producers like various linear bar code versions; however EPC was used to assign a unique identifier to any individual object, whereas bar codes usually only categorize the manufacturer and class of products.

Low-cost passive RFID tags typically have restricted memory size available for the entire EPC codes. In addition, the communication bandwidth is very limited. The passive RFID tags has no power source and they produce the electronic waves using the interrogating radio waves sent be the transmitter. These attributes have instigated studies facilitating techniques enabling reduce of the data encoded by the RFID technology [7,8].

II. COMPRESSION OF EPC CODES

We suggest compressing the EPC data using an entropy encoder. The idea of manipulating the EPC data is typically common for cryptography [9]. We would like to take the same concept of manipulating the EPC data, but for data compression.

In other applications, when the correlation between the bits is not strong, Arithmetic Coding is usually selected. Sometimes Huffman Codes [10] are also selected because of its robustness against errors [11]; however, The Arithmetic Coding has a better compression ratio [12].

In [13] Shannon proved that an optimal coding encodes every event with a probability P, by $-\log_2 P$ bits. Huffman Codes round the sizes of the codewords to an integer number of bits; therefore, Huffman Codes are almost always not optimal, unless $-\log_2 P$ is an integer number for every probability P for any codeword in the item set. Such a distribution is called a Dyadic Distribution [14] and it is not suitable for almost all of the applications. Arithmetic Coding successfully deal with this disadvantage but can be quite complex and slow.

The use of Arithmetic Coding in embedded systems has been suggested a long ago in [15]. There have been a lot of studies to reduce the complexity of Arithmetic Coding implementation. This complexity damaged the time and power performance of Arithmetic Coding. Some of those studies can be found in [16].

Unlike, Huffman Codes that make use of a bit string for each item in the original file, the Arithmetic Coding technique improves on this by using fractions of bits as codewords [17], where one bit can be "owned" by several items. Therefore, a codeword can be represented by a non-integer numbers of bits (e.g. by 5.3 bits). The Arithmetic Coding technique is described by the following pseudocode:

Let L be a set of items.

Each item i in L has a probability Pi within [0,1], such that:

$$\sum_{i \in L} P_i = 1$$

Each item is represented by the interval:

$$[\sum_{j<i} P_j\ ,\ \sum_{j\le i} P_j\ )$$

Repeat until EOF:

The current interval is divided into sub-intervals according to the items' probabilities.

Replace the current interval by the sub-interval of the items that were read.

Write into the compressed file the shortest binary fraction available in the current interval.

III. Adapting of Arithmetic Coding for EPC Compression

EPC common version is 96 bits long and contains several segments:

- Header – Identifies the length, type, structure, version, and generation of the EPC (8 bits).
- EPC Manager Number – Identifies the business, the company or the manufacturer of a line of items for consumption (28 bits).
- Object Class – Identifies a class of items (24 bits).
- Serial Number – Identifies the instance (36 bits).

E.g. the EPC data for a specific Diet Coca Cola can of 330ml identifier is

Header is 0011 0000 for Serialized Global Trade Item Number (SGTIN-96)

EPC Manager Number is 0000 0000 0000 0000 1010 1000 1001 for Coca Cola Company.

Object Class is 0000 0000 0000 0001 1011 1111 for Diet Coca Cola can of 330ml.

Serial Number is 0000 0000 0000 0000 0110 1011 0011 1010 0111 for a specific individual 330ml can of Diet Coca Cola.

So the entire EPC for this specific 330ml can of Diet Coca Cola is:

0011 0000 0000 0000 0000 0000 1010 1000 1001 0000 0000 0000 0001 1011 1111 0000 0000 0000 0000 0110 1011 0011 1010 0111.

There are 28 bits for the company/manufacturer i.e. there are 2^{28} (=268435456) potential companies/manufacturers. Obviously, many of the potential combinations are not in use. The same inefficiency in the codes also occurs in the Object Class. There are 2^{24} (=16777216) potential objects, but most of the companies/manufacturers do not have so many products. The serial number which has 2^{36} (=68719476736) potential individual objects and it is also almost always too much, but in this field there is also another attribute. Many of the companies/manufacturers begin a new series of numbers because of a new year, a new manager or some out of the blue reasons.

Because of these many unused numbers, the code strings of RFID tags have several characteristics:

- There are many unused code strings. The components of the EPC are derived from existing codes (on bar codes, in databases, etc). They are selected from 0…0 to 1…1, but most of them are unexploited.
- The unused codes are more frequent and in addition many of them tend to cluster rather than randomly scatter in the range.

There are many compression techniques know how to effectively handle data with such characteristics e.g. JPEG [18] or MP3 [19], so we will adapt their way of behaving. They split the codes into pairs of:

1. The Recurring sequence.
2. The rest of the number.

Then, they give a special codeword (if they use Huffman codes) or a special interval (if they use Arithmetic Coding) for each of the possible combinations.

According to this method, we analyzed the probabilities of the possible EPC data and we build tables of intervals for the Arithmetic Coding scheme so as to facilitate such a compression.

IV. Results

We used data from a retail corporation selling a diversity of products to build our dataset. A separate table has been built for each of the EPC segments.

The tables are ideal for the current use of EPC codes; however when more companies/manufacturers join the RFID market, more codes will be assigned and the tables will be less accurate; therefore the compression can be less efficient. In spite of this, we are still sure that no compression is less efficient than a less accurate compression and furthermore, a comprehensive change in the tables can be done in the future if needed.

The current number of companies having an EPC numbers for RFID is some tens of thousands which means 13 bits of the EPC number are actually unneeded and just 15 bits are required for containing the significant/valuable data.

The retail corporation we have tested has just 951 suppliers out of the tens of thousands companies that have EPC numbers; therefore we have referred to the other suppliers as "very rare" and the compression was built according to this assumption. Obviously, a wider dataset can yield more accurate compression.

Most of the vendors do not sell many kinds of items to retail stores. Figure 1 shows how many items are sold by the suppliers of the retail corporation. It can be straightforwardly seen that most of the suppliers sell no more than eight products. Eight products can be distinguished by merely $\log_2 8$ bits i.e. 3 bits.

Fig. 1. Number of products per supplier

EPC gives 24 bits for the item description which mean 16777216 kinds of items; however, usually just much fewer bits are required, so any encoder compression will make advantage of such an attribute and the Arithmetic Coding is no exception.

The last bits of the EPC tag are a serial number for each specific item. This field contains 36 bits for 68719476736 potential items of the very same product. Again, this is much more than is required to almost all of the items.

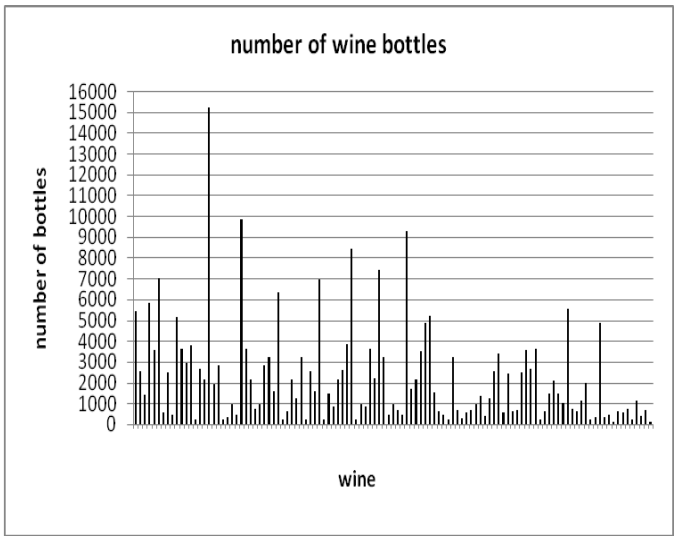

Fig. 2. Number of wine bottles sold in one year.

We check some types of wine bottles in size of 750ml. There were 113 different wines in this category. Figure 2 shows the number of wine bottles for each Object Class sold in one year (2014). As can be seen in this figure, most of the wines need only just few bits. Even if we take the most sold wines (15248 items) and we take just the half of the bits suggested by EPC – 18 bits, we will have 262144 combinations which are enough for more than 17 years of selling of this item.

What is more, the Arithmetic Coding is an entropy encoder, so when the entropy is very low because most of the numbers needed to be compressed are similar; the compression will work at its best.

Taken as a whole, the 96 bits of the EPC tag were compressed in average to 27.48 bits which are 28.625% of the original size.

In particular, we omitted the header because we used only SGTIN-96. The EPC Manager Number was reduced from 28 bits to 9.37 bits.

The Arithmetic Coding stipulates that the average code length to each item will be in average the entropy of the given data calculated by this formula:

$$\sum_{i=1}^{n} P_i Log(P_i)$$

Where n is the number of the entries to be compressed and P_i is the probability of each of these entries.

When applying the Arithmetic Coding to the data shown in Figure 1, this formula will give a result of 6.33 bits in average for each entry i.e. the Object Class was reduced from 24 bits to 6.33 bits.

And only remaining the Serial Number was reduced from 38 bits to 11.78 bits.

The reduction ratio of each of the EPC tag segments is shown in Figure 3.

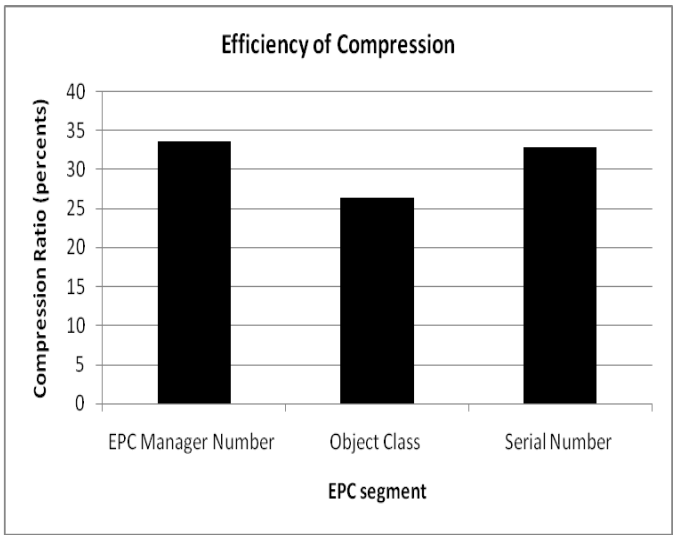

Fig. 3. Compression ratio in percents of each of EPC tag segments

V. Conclusions and Future work

The power and the communication bandwidth of passive EPC tags are somewhat limited [20]. In order to confront this challenge two main approaches have been suggested:

- Constructing passive tags that are capable to consume less energy.
- Facilitating smaller codes with less bits for the EPC tags

In this paper we focused in the second option. We succeed to efficiently compress EPC tags. This compression can reduce the power consumption of EPC tags and make them even cheaper which is certainly essential for a more widespread use in numerous industries.

The results are encouraging. EPC tags were compressed to 28.625% of their original size. This reduction can make the passive RFID tags more effective.

In the future, we consider making our dataset more comprehensive. Also, we consider to make our implementation adaptive with the aim of adjusting the compression according to new manufacturers/companies and new products arriving at the market.

References

[1] D. C. Ranasinghe, M. Harrison, and P. H. Cole, "EPC network architecture." In Networked RFID Systems and Lightweight Cryptography, pp. 59-78, Springer Berlin Heidelberg, 2008.

[2] S. Sanjay, "How Inexpensive RFID Is Revolutionizing the Supply Chain: (Innovations Case Narrative: The Electronic Product Code)", Innovations, Vol. 7, No. 3, MIT Press, Cambridge, MA USA, pp. 35-52, 2012.

[3] R. Weinstein, "RFID: a Technical Overview and its Application to the Enterprise", IT professional, IEEE Computer Society, Vol. 7, no. 3, pp. 27-33, 2005.

[4] M.R. Rieback, B. Crispo and A. S. Tanenbaum, "The evolution of RFID security", Pervasive Computing, IEEE Computer Society, Vol. 1, pp. 62-69, 2006.

[5] M. O. Majekodunmi, "Utilizing automatic identification tracking systems to compile operational field and structure data", MSc Thesis, University of Maryland, College Park, 2014.

[6] A. Milne, "The rise and success of the barcode: Some lessons for financial services", Journal of Banking Regulation, Vol. 14, no. 3, pp. 241-254, 2013.

[7] L. Wang, X. Li Da, B. Zhuming and X. Yingcheng, "Data cleaning for RFID and WSN integration", IEEE Transactions on Industrial Informatics, Vol. 10, No. 1, pp. 408-418, 2014.

[8] T. Hongsongkiat, P. Chongstitvatana, "AES implementation for RFID Tags: The hardware and software approaches" In Proceedings of IEEE International Computer Science and Engineering Conference (ICSEC), pp. 118-123, 2014.

[9] D. C. Ranasinghe, "Lightweight cryptography for low cost RFID", In Networked RFID Systems and Lightweight Cryptography, pp. 311-346, Springer Berlin Heidelberg, 2008.

[10] D. A. Huffman, "A method for the Construction of Minimum Redundancy Codes", In Proceedings of the Institute of Radio Engineers, Vol. 40, pp. 1098-1101, 1952.

[11] S. T. Klein and Y. Wiseman, "Parallel Huffman Decoding with Applications to JPEG Files", The Computer Journal, Oxford University Press, Swindon, UK, Vol. 46(5), pp. 487-497, 2003.

[12] Y. Wiseman, "A Pipeline Chip for Quasi Arithmetic Coding", IEICE Journal - Trans. Fundamentals, Tokyo, Japan, Vol. E84-A No.4, pp. 1034-1041, 2001.

[13] C. E. Shannon, "A Mathematical Theory of Communication, Bell System", Tech. Journal, vol. 27, pp. 398-403, 1948.

[14] C. Schmitz, and A. Schmeink, "Construction of Efficient Amplitude Phase Shift Keying Constellations." In Proceedings of SCC 2015 - 10th International ITG Conference on Systems, Communications and Coding, pp. 1-6., VDE, Feb. 2015.

[15] H. Lekatsas, J. Henkel and W. Wolf, "Arithmetic Coding for Low Power Embedded System Design", In Proceedings of Data Compression Conference, Snowbird Utah, pp. 430-439, 2000.

[16] M. Tao, M. Hempel, D. Peng, and H. Sharif. "A survey of energy-efficient compression and communication techniques for multimedia in resource constrained systems", IEEE Communications Surveys & Tutorials, Vol. 15, No. 3, pp. 963-972, 2013.

[17] P. G. Howard and J. S. Vitter, "Arithmetic Coding for Data Compression", In Proceedings of the IEEE, Vil. 82, No. 6, pp. 857-865, 1994.

[18] Y. Wiseman, "The still image lossy compression standard – JPEG", Encyclopedia of Information and Science Technology, Third Edition, Vol. 1, Chapter 28, pp. 295-305, 2014.

[19] K. Brandenburg, "MP3 and AAC explained" In proceedings of 17th International Conference of High-Quality Audio Coding, Audio Engineering Society, 1999.

[20] W. Trappe, R. Howard, R. S. Moore, "Low-Energy Security: Limits and Opportunities in the Internet of Things", IEEE Security & Privacy, Vol.13, No. 1, pp. 14-21, Jan.-Feb. 2015.

[21] Y. Wiseman and I. Gefner, "Conjugation Based Compression for Hebrew Texts", ACM Transactions on Asian Language Information Processing, Vol .6(1), article no. 4, 2007.

[22] Y. Wiseman and A. Barkai, "Smaller Flight Data Recorders", Journal of Aviation Technology and Engineering, Vol. 2(2), pp. 45-55, 2013.

[23] M. Reuven and Y. Wiseman, "Medium-Term Scheduler as a Solution for the Thrashing Effect", The Computer Journal, Oxford University Press, Swindon, UK, Vol. 49(3), pp. 297-309, 2006.

[24] Y. Wiseman, "Take a Picture of Your Tire!", Proc. IEEE Conference on Vehicular Electronics and Safety (IEEE ICVES-2010) Qingdao, ShanDong, China, pp. 151-156, 2010.

[25] Y. Wiseman, Joel Isaacson and Eliad Lubovsky, "Eliminating the Threat of Kernel Stack Overflows", Proc. IEEE Conference on Information Reuse and Integration (IEEE IRI-2008), Las Vegas, Nevada, pp. 116-121, 2008.

[26] Y. Wiseman, "Burrows-Wheeler Based JPEG", Data Science Journal, Vol. 6, pp. 19-27, 2007.