Indonesian Journal of Electrical Engineering and Computer Science
Vol. 2, No. 2, May 2016, pp. 268 ~ 274
DOI: 10.11591/ijeecs.v2.i2.pp268-274

Arithmatic Coding Based Approach for Power System Parameter Data Compression

Subhra J. Sarkar*[1], Nabendu Kr. Sarkar[2], Trishayan Dutta[3], Panchalika Dey[4], Aindrila Mukherjee[5]
[1,3-5]Dept. of EE, Techno India Batanagar, B7-360 / New, Ward No. 30, Putkhali, Maheshtala, Kolkata – 700141, West Bengal, India
[2]Dept. of EE, Haldia Institute of Technology, I.C.A.R.E Complex, H.I.T Campus, Hatiberia, PO HIT, District Midnapore (E), Haldia, West Bengal 721657, India
*Corresponding author, e-mail: subhro89@gmail.com1, nsarkares@rediffmail.com, trishayanduttakol@gmail.com, panchalika.dey95@gmail.com, aindrilamukherjee24@gmail.com

Abstract
For stable power system operation, various system parameters like voltage, current, frequency, active and reactive power etc. are monitored at a regular basis. These data are either stored in the database of the system or transmitted to the monitoring station through SCADA. If these data can be compressed by suitable data compression techniques, there will be reduced memory requirement as well as lower energy consumption for transmitting the data. In this paper, an algorithm based on Arithmetic Coding is developed for compressing and decompressing such parameters in MATLAB environment. The compression ratio of the algorithm clearly indicates the effectiveness of the algorithm.

Keywords: *Data Compression, Arithmetic Coding, Compressed data, Power System, Parameter monitoring*

1. Introduction

In modern power system, a number of alternators connected in same or other plants operate in parallel to meet the load demand of the system. Grid implies the system comprising of connecting alternators of all power plants in parallel which can be achieved after synchronizing them with each other or with the bus bar. The voltage, frequency, phase angle and phase sequence of incoming alternator and bus bar should be same. Due to some sudden transients or some faults in the system, the system tends to become unstable resulting deviation from standard voltage and frequency value. So, continuous monitoring of the system voltage and frequency is very important. The continuous monitoring of the system parameters is done by employing suitable highly sophisticated communication system built in within the power system and using a system called SCADA (Supervisory Control and Data Acquisition) [1-7].

SCADA system is used for supervision and control of remote field devices. The desired power system parameters are collected at remote end using SCADA using some suitable meters, sensors, process equipment etc. These collected data readings are then transmit to the control centre with suitable wired communication channel including power line, telegraphic cable, etc. or through or wireless communication. SCADA has a wide application in the industry. It is extremely popular in electricity utilities as it enables the remote operation & control of substations and generating stations. Processing of the collected data might require some opening and closing of circuit breakers or valves particularly when system parameters exceed the predefined thresholds. In SCADA architecture, the role of Remote terminal units (RTUs) are to connect sensors in the system, transmit the acquired data to the supervisory system and receiving instructions from supervisory system. On the other hand, Programmable logic controllers (PLCs) have sensors connected to it but do not have any inbuilt telemetry hardware. Thus PLCs can replace RTUs due to its economy, versatility, flexibility and configurability. The communication system between field devices and control centre might be wired or wireless. Control centre comprises of Human-Machine Interface (HMI), Data Historian, control/data acquisition server, communication router etc. HMI presents the processed data to human operator for monitoring and interaction. Data Historian has various data, events and alarms in a

Received October 11, 2015; Revised February 20, 2016; Accepted March 9, 2016

database which is HMI accessible. Both HMI and Historian are the clients for data acquisition server which allows them to access any data from field devices using suitable protocol [8].

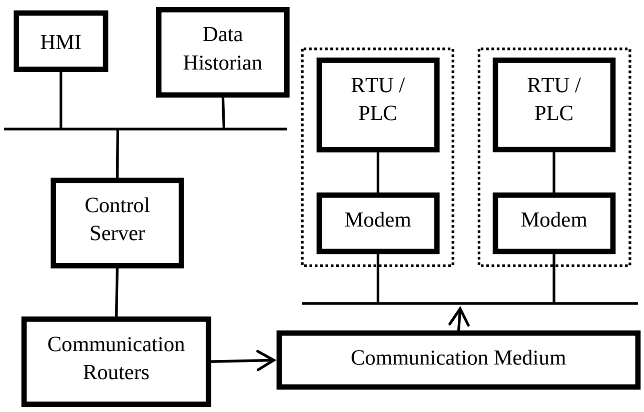

Figure 1. SCADA Architecture

In [9], comparison of performance of various lossless and lossy power system related data compression is available. There are few works for compressing power system data using wavelet transform [10-13]. Power system data compression and its implementation will be of greater significance while approaching towards smart grid [14, 15]. In the proposed method, an approach to reduce the number of bits of the power system monitoring information developed in MATLAB environment and tested offline. At the encoding end, power system parameters say voltage, frequency and load power factor are compressed to form a character string using arithmetic coding based approach. At the decoding end, the compressed character string is decompressed to obtain the necessary parameters. As there is a reduction in the number of bits to be transmitted or storage, there must be a subsequent reduction in energy required for transmission and memory requirement. Besides that, there is an inherent encryption in the proposed algorithm which provides data security.

2. Data Compression

Data compression is the process of reducing the number of bits (or bytes) in order to have easier storage or transmission of the data. Data compression can also lead towards the data encryption thereby providing the necessary data security. In other words, it is also be defined as the recombination the bits (or bytes) for making the smaller and compact form of data by eliminating the identical data bits or continuously recurring data. At the point of data restoration, some decompression method must be developed for decoding the compressed data so that the original data can be extracted. For the applications dealing bulk information, data compression is extremely important as significant saving in storage is observed while storing the data. Besides that, there is a reduction in time required for data transfer and energy requirement for data transmission [1, 18-23].

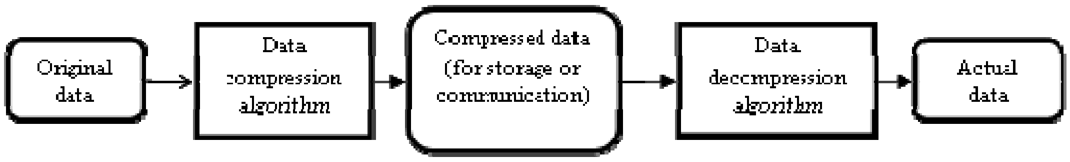
Original data
Data compression algorithm
Compressed data (for storage or communication)
Data decompression algorithm
Actual data

Figure 2. Block Diagram of Data Compression System

Data compression might be either lossy or lossless data compression techniques. In lossy data compression, some data which are not of much importance are eliminated from the file to be compressed and thus there is a reduction of file size. But it is obvious that the decoded file will not be an exact replica of the actual file. Applications of this compression includes compressing multimedia files like audio, video, images etc. especially for applications such as streaming media and internet telephony. Transform coding, Karhunen- Loeve Transform (KLT) coding, wavelet based coding etc. are few of the available lossy data compression techniques. On the other hand, there is no loss of data in lossless data compression, and all the information can be reconstructed form the compressed data and can be restored. So, this compression can be used for reducing the size of important data (ZIP file), medical images etc. The approach of compression is similar in most of the lossless compression techniques. Initially a statistical model for the input data is generated which is then utilized to map input data to bit sequences in such a way that the most probable (frequently encountered) data will produce shorter output than improbable data in the subsequent steps. Shannon-Fano algorithm, Huffman algorithm, Arithmetic Coding etc. are the few commonly known lossless data compression techniques [1, 16- 21].

Basic arithmetic coding is a lossless data compression technique where a data string is encoded in form of a fractional value lying in between 0 and 1. In this method, based on the probability of the content of source message the interval is narrowed successively [18, 19]. Considering some source message comprising of symbol set S comprising of characters {a, b, c, d, s, k} with their respective probability of occurrence {0.3, 0.2, 0.1, 0.2, 0.1, 0.1}. Based on this information, the range for a particular symbol can be obtained as given in table 1 [1, 19, 20]. Basic Arithmetic Coding Algorithm for encoding and decoding a particular symbol string is discussed in the subsequent steps [1, 16].

Table 1. Probability distribution table for symbol set S

Sl. No.	Character	Probability	Cumulative probability	Range	
				r_low	r_hi
1	a	0.3	0.3	0	0.3
2	b	0.2	0.5	0.3	0.5
3	c	0.1	0.6	0.5	0.6
4	d	0.2	0.8	0.6	0.8
5	s	0.1	0.9	0.8	0.9
6	k	0.1	1	0.9	1

2.1. Algorithm at Encoding End

Input: Symbol string (S)

Output: Binary string bin of float number (num)

```
STEP 1: Calculate the length (l) of S.
STEP 2: Initialize variables min = 0, max = 1and r = 1.
STEP 3: Set i = 1.
STEP 4: Repeat steps 5 - 9 until i≤l+1.
STEP 5: x= $i^{th}$ character of S.
STEP 6: Corresponds to x, obtain r_low and r_hi.
STEP 7: Update min = min + r * r_low and max = min + r * r_hi.
STEP 8: r = max – min.
STEP 9: i = i + 1.
STEP 10: End of the loop.
STEP 11: Obtain a number num (min <num< max) with minimum binary string length.
STEP 12: bin = Binary equivalent of num
STEP 13: End.
```

Consider, for a symbol string {b, a, c, k} having 4 characters, which is to be encoded by the arithmetic coding algorithm. The execution results for the four iterations are illustrated in table 2. The output of the algorithm will be the binary string (.) 0101011 corresponds to the float number, num = 0.3359375 lying between 0.3354 and 0.336 having minimum binary string length (7 bits in this case) [1, 19].

Table 2. Iteration Steps in Arithmetic Coding for string ‘back’

Iteration No.	Character (x)	min	max	r
1	b	0.3	0.5	0.2
2	a	0.3	0.36	0.06
3	c	0.33	0.336	0.006
4	k	0.3354	0.336	0.0006

2.2. Algorithm at Decoding End

Input: Binary string (bin); Actual string length (l)
Output: Actual string (arr)

```
STEP 1: Calculate float number (num) corresponding to the binary string.
STEP 2: Define a null array arr.
STEP 3: Set i = 1.
STEP 4: Repeat steps until i≤ (l+1).
STEP 5: Determine the range within which num lies(corresponding to some character x).
STEP 6: arr (i) = x.
STEP 7: lo = r_low (x), hi= r_hi (x) and r = hi -lo.
STEP 8: num = (num – lo) / r.
STEP 9: i = i + 1.
STEP 10: End of the loop.
STEP 11: Array arr is the encoded string.
STEP 12: End.
```

For binary string being encoded bin:= 010101 where l = 4, the process of execution of the algorithm to extract the encoded string is as given in table 3.

Table 3. Iteration Steps to obtain the encoded string

Iteration No. (i)	Value (num)	arr(i) = x	lo	hi	r
1	0.3359375	b	0.3	0.5	0.2
2	0.1796875	a	0	0.3	0.3
3	0.598958333	c	0.5	0.6	0.1
4	0.989583333	k	0.9	1.0	0.1

3. Proposed Arithmetic Coding based Algorithm

In the MATLAB based algorithm, power system parameters (say voltage, frequency and power factor) is compressed to form a character string at the encoding end. At the decoding end, the power system parameters are extracted from the compressed string by arithmetic decompression based algorithm. The data symbol set D comprises of the characters between 0-9.If the probability of each character is assumed to be equal, the probability distribution table for D is given in table 4.

Table 4. Probability distribution table for data symbol set D

Sl. No.	Character	Probability	Cumulative probability	Range	
				r_low	r_hi
1	0	0.1	0.1	0	0.1
2	1		0.2	0.1	0.2
3	2		0.3	0.2	0.3
4	3		0.4	0.3	0.4
5	4		0.5	0.4	0.5
6	5		0.6	0.5	0.6
7	6		0.7	0.6	0.7
8	7		0.8	0.7	0.8
9	8		0.9	0.8	0.9
10	9		1	0.9	1

3.1. Algorithm at Encoding End

Input: System voltage (v) in kV; System frequency (f) in Hz; Load power factor (pf)

Output: Character array (char)

```
STEP 1: Convert v, f and pf decimal string vstr, fstr, pfstr.
STEP 2: pfv:= Concatenate (pfstr, fstr, vstr).
STEP 3: arith:= Binary array corresponds to the arithmetic coding of pfv.
STEP 4: num:= 12- Length (pfv).
STEP 6: binum:= Binary equivalent of num.
STEP 7: nwari:= Concatenate (binum, arith).
STEP 8: len:= Array length (nwari).
STEP 9: num:= len%7.
STEP 10: zero:= Zero array having num number of elements.
STEP 11: nwar:= Concatenate (zero, wari).
STEP 12: count:= Array length (nwar) / 7.
STEP 13: spar:= Split nwar in (7 X count) array.
STEP 14: Define null array char and deci of size (1 X 7).
STEP 15: Initialize i:= 1 and repeat steps 16-18 until i≤ count.
STEP 16: deci (i):= Decimal equivalent of nwar (7 X i).
STEP 17: asci:= ASCII character corresponds to deci (i).
STEP 18: char(i):= asci. i:= i+1.
STEP 19: End of the loop.
STEP 20: Int:= ASCII character corresponds to len.
STEP 21: com:= Concatenate (Int, char).
STEP 22: End.
```

3.2. Algorithm at Encoding End

Input: Character array (char)

Output: System voltage (v) in kV; System frequency (f) in Hz; Load power factor (pf)

```
STEP 1: len:= Array length (combo).
STEP 2: actlen:= ASCII value corresponds to the character combo (1).
STEP 3: numzero:= 7*(len-1) – actlen.
STEP 4: nwar:= New array of size (1X(len-1)) formed from combo after removing combo(1).
STEP 5: null:= Null array. Initialize i:= 1 and repeat steps 6-8 until i≤ (len-1).
STEP 6: p:= nwar(i).
STEP 7: bin:= (1X7) binary array corresponds to binary value of p.
STEP 8: null:= Concatenate (null, bin). i:= i+1.
STEP 9: End of the loop
STEP 10: nwbin:= New binary array formed from bin after removing numzero number of zeros
from the beginning.
STEP 11: strsz(1X2):= Array containing first two bits of nwbin.
STEP 12: num:= Decimal equivalent of strsz.
STEP 12: modbin:= Modified binary array formed from nwbin after removing strsz.
STEP 11: str:= Character string corresponds to the arithmetic decoding of modbin using actlen.
STEP 12: Using num, split str to obtain v, f and pf.
STEP 13: End.
```

4. Result and Analysis

Effectiveness of any compression algorithm can be determined by the value of compression ratio, i.e. ratio of the size of uncompressed data to that of compressed data. Higher will be the compression ratio, better will be the compression. High compression ratio also implies reduced memory and energy requirement for data storage and data transmission respectively. The performance of the algorithm in terms compression ratio as well as execution time required at encoding and decoding end is given in table 5.

Table 5. Compression ratio and execution time required for various voltage levels

	Input parameters					Compression result			
Sl. No	Voltage (in kV)	Frequency (in Hz)	pf	No. of integer	Actual data size (in bytes)	Compressed data size (in bytes)	Compression ratio	Encoding time	Decoding time
1	0.41	49.95	0.965	9	226	98	2.306	0.423	0.139
2	3.31	50.01	0.931	10	226	98	2.306	0.438	0.121
3	11.02	50.02	0.802	11	242	114	2.123	0.428	0.137
4	220.06	50.01	0.875	12	258	114	2.263	0.425	0.129

From table 5, it is clear that the compression ratio for this arithmetic coding based algorithm is above 2 for any possible input. As the length of encoded binary array is dependent on the content of input array length but not on the length of the input array, it is quite possible that at similar voltage levels with slightly different inputs, there is a variation of compressed string length and thus there is some variation in the compression ratio. Few such examples are given in table 6.

Table 6. Variation of number of characters in compressed string with inputs

	Input parameters			
Sl. No.	Voltage (in kV)	Frequency (in Hz)	Power factor	Number of characters in compressed string
1	1.13	49.98	0.675	7
2	1.11	50.01	0.758	6
3	33.12	49.96	0.752	6
4	33.12	50.01	0.798	7

5. Conclusions

From the above discussions, it is clear that there is a significant reduction in size of the input data without loss of any sigle bit. This implies a reduced memory requirement for storing bulk volume of such input data. There is an inherent encryption in the algorithm, thereby can deal with data security problems successfully. Reduced amount of data transfer also results lower energy requirements for communication purposes. Though the above results are obtained by executing the algorithm offline, this algorithm can also be implemented for online testing where the encoding and decoding programs are executed simultaneously. The variation of total execution time (in sec.) with total number of decimal numbers in the input parameters is given in figure 3.

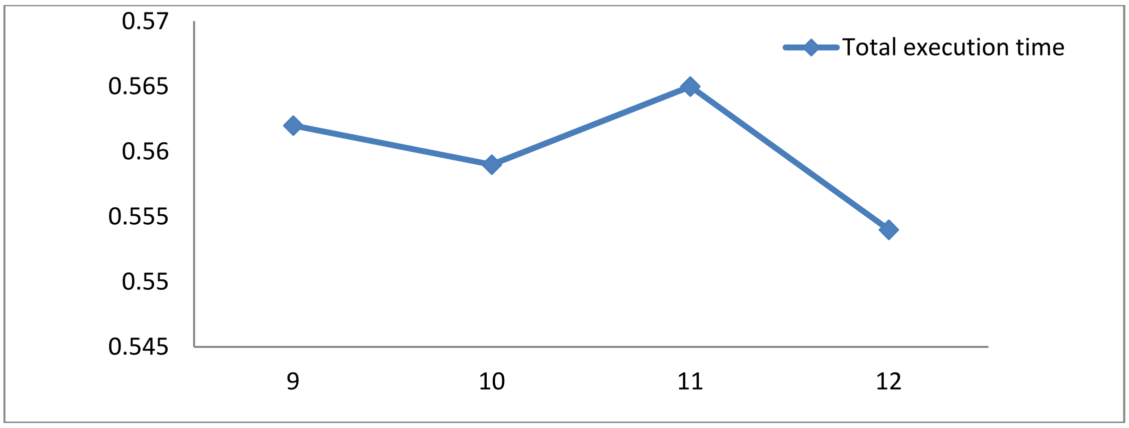

Figure 3. Variation of total execution time with number of integers

It is clear from figure 3 that the execution time is independent of the number of the intergers in the input parameters. The algorithm in present form can deal with three power system parameters due to software limitations. But if the limitation can de taken care of, it is possible to compress more number of parameters to improve the system performance. There will be improvement in the compression ratio further, if probability distribution can be prepared by consulting the real time data values. This algorithm can also be implemented with other lossless data compression techniques to compare and improve the performance, if possible. The work can also be extended for compressing large multi-dimensional data array containing different power system parameters monitored at different time instances.

References

[1] SJ Sarkar, B Das, T Dutta, P Dey, A. Mukherjee, *An Alternative Voltage and Frequency Monitoring Scheme for SCADA based Communication in Power System using Data Compression*, International Conference and Workshop on Computing and Communication (IEMCON), Vancouver. 2015: 1- 7.

[2] JB. Gupta. *Power System Analysis*. 1st Edition, S. K. Kataria & Sons, New Delhi. 2011.

[3] Leonard L Grigsby. *Power System Stability & Control*. 3rd Edition, CRC Press, Boca Raton. 2012.

[4] AE Fitzgerald, Charles Kingsley, Stephen. D Umans. *Electric Machinery*. 6th Edition, New York. 2003.

[5] CL Wadhwa*. Electrical Power Systems*. 6th ed., New Age International Publishers, New Delhi. 2014.

[6] John J Grainger, William D Stevenson Jr. *Power System Analysis*. International Edition, McGraw Hill Inc., Singapore. 1994.

[7] BM Weedy, BJ Cory, N Jenkins, JB Ekanayake, G Strbac. *Electric Power System*. 5th Edition, Wiley, West Sussex. 2012.

[8] Nicoleta ARGHIRA, Daniela HOSSU, Ioana FĂGĂRĂŞAN, Sergiu Stelian ILIESCU, Daniel Răzvan COSTIANU. Modern SCADA Philosophy in Power System Operation – A Survey. *U.P.B. Sci. Bull., Series C*. 2011; 73(2).

[9] Sarasij Das, M.S. Thesis (Engineering). Power System Data Compression for Archiving. Indian Institute of Science. 2007.

[10] Shang L, Jaeger J, Krebs R. *Efficiency Analysis of data compression of power system transients using wavelet transform*. Power Tech Conference Proceedings, IEEE Bologna. 2003; 4.

[11] Gabriel Găşpăresc, *Power Quality Data Compression*, Available. [Online] http://cdn.intechopen.com/pdfs/44193/InTechPower_quality_data_compression.pdf.

[12] Jiaxin Ning, Jianhui Wang, Wenzhong Gao, Cong Liu. A Wavelet based data compression technique for smart grid. *IEEE Transactions on Smart Grid*. 2011; 2(1): 212-218.

[13] G. Panda, PK Dash, AK Pradhan, SK Meher. Data Compression of Power Quality Events Using the Slantlet Transform. *IEEE Transactions on Power Delivery*. 2002; 17(2): 662-667.

[14] K Nithiyananthan, V Ramachandran. Effective Data Compression Model for On-line Power System Applications. *International Journal of Electrical Energy*. 2014; 2(2).

[15] Michel P Tcheou, Lisandro Lovisolo, Moises V Ribeiro, Eduardo AB da Silva, Marco AM Rodrigues, João MT Romano, Paulo SR Diniz. The Compression of Electric Signal Waveforms for Smart Grids: State of the Art and Future Trends. *IEEE Transactions on Smart Grid*. 2014; 5(1): 291-301.

[16] Tribeni Prasad Banerjee, Amit Konar, Ajith Abraham. CAM based High-speed Compressed Data Communication System Development using FPGA, [Online] Available: http://www.academia.edu/2374082/CAM_based_Highspeed_Compressed_Data_Communication_System_Development_using_FPGA.

[17] Difference between Lossless & Lossy Data Compression, [Online] Available: http:// www.rfwireless-world.com/Terminology/lossless-data-compression-vs-lossy-data-compression.html.

[18] Theory of Data Compression. [Online]. Available: http://www. data-compression .com/theory.shtml.

[19] Ze-Nian Li, Mark S Drew, Jiangchuan Liu. *Fundamentals of Multimedia*. 2nd Edition, Springer. 2014.

[20] Claudio Iombo, MS. Thesis, Predictive Data Compression using Adaptive Arithmetic Coding. Louisiana State University & Agricultural and Mechanical College. August 2007.

[21] ZSR Kodituwakku, US Amarasinghe. Comparisons of Lossless Data Compression Algorithms for Text Data. *Indian Journal of Computer Science and Engineering*. 1(4): 406 -425.

[22] Rupinder Singh Brar, Bikramjeet Singh. A Survey on Different Compression Techniques & Bit Reduction Algorithms for Compression of Text / Lossless Data. *International Journal of Advanced Research in Computer Science & Software Engineering*. 2013; 3(3): 579-582.

[23] Chang, Chia-Hui Wang, Ching-Chia Hsieh. Data Compression for Energy Efficient Communication on Ubiquitous Sensor Network. *Tamkang Journal of Science and Engineering*. 2011; 14(3): 345-354.