Research on battery data compression method for electric vehicles

Zunyu Mei
College of Electrical and Electronics Engineering, Harbin University of Science and Technology, Harbin 150080, China
Email: meizunyu@hrbust.edu.cn

Xiaogang Wu*
College of Electrical and Electronics Engineering, Harbin University of Science and Technology, Harbin 150080, China
and
State Key Laboratory of Power Transmission Equipment & System Security and New Technology, Chongqing University, Chongqing 400044, China
Email: xgwu@hrbust.edu.cn
*Corresponding author

Xiaoyu Li and Chunbo Zhu
School of Electrical Engineering and Automation, Harbin Institute of Technology, Harbin 150001, China
Email: xiaoyu070220202@126.com
Email: zhuchunbo@hit.edu.cn

Abstract: The transmission and storage of electric vehicle battery management system (BMS) data are time consuming and infeasible, which reduces the efficiency of data transmission and storage. For this problem, this paper presents a method of battery data compression, the pre-compression is to reduce the amount of data and improve the data redundancy and repeatability, and a strategy was formulated that two universal compression methods, Huffman coding and run-length coding, were applied to different battery data. Experimental results show the method of battery data compression can obtain a good compressibility. The battery data compressibility of this paper was about 21%.

Keywords: battery data; data compression; pre-compression; Huffman coding; run-length coding.
1 Introduction

Battery management system (BMS) possesses more and more functions with the development of electric vehicles technology, the functions include: battery basic state parameters measurement, state-of-charge (SoC) or state-of-health (SoH) or internal resistance estimation, fault diagnosis, protection of battery charging and discharging, control of charging and discharging depth, control of batteries imbalance, battery thermal management, etc. The realisation of the functions is usually based on state parameters, for example, voltage of the battery, current, temperature, storage of battery basic state parameters is important particularly. But, it brings many difficulties to the increase store of the battery parameter data.

To reduce storage space and the transmission time, it is necessary to compress the large amounts of the data, availably. Of data compression, as a solution to the problem of mass storage and transmission, scholars are widely paying attention and research. A combination of discrete cosine transform (DCT) and fractal image compression techniques is proposed in order to compress the image efficiently (Rawat and Meher, 2013). DCT is employed to compress the colour image while the fractal image compression is employed to evade the repetitive compressions of analogous blocks. The given image is encoded by means of Huffman encoding technique. A compression method for multiple data is presented based on Huffman coding (Datta Das et al., 2014), applied to induction motor remote control system. The method is proved to be able to make a large amount of data compression effectively, and improve the security of data transmission. An improved method for compressing industrial computed tomography (CT) images is presented with the universal compression methods of Huffman coding.
Research on battery data compression method for electric vehicles

fusion (Jiang and Zeng, 2013), experimental results show that this method can obtain a
good compression ratio as well as keeping satisfactory quality of compressed images. He
et al. (2012) proposed a strategy of the global positioning system (GPS) data processing
by removing redundant information and shifting the combination codes according to the
characteristics of the GPS system for intelligent garment. Its compressed result is
analysed by comparing with direct data compression and Huffman coding. That paper’s
strategy takes into account both the speed and the effect of the compression, and has a
good performance in the compression ratio. Various universal lossless compression
algorithms are introduced and compared in detail (Stabno and Wrembel, 2007). Different
algorithms have different compression rates for different data, compression rate not only
depends on the algorithm itself, but also depends on the characteristics of the compressed
data (Han and Li, 2016; Ren, 2015; Zhang et al., 2015).

At present, there is less research on battery data compression method for electric
vehicles. This paper presents a method of battery data pre-compression and a strategy
between the two universal compression algorithms, Huffman coding and run-length
coding, according to the characteristics of the battery data. In the process of battery data
compression transmission, the efficiency is improved.

2 Battery data characteristics for electric vehicles

The datum is usually divided into three types: string type, numeric type and time data
type (Nourani and Tehranipour, 2005). There is a big difference between different types
of data, and the same type of data also has different characteristics. Researching on
characteristics of battery data is helpful to the data compression. In this paper, the battery
data for electric vehicles include the time data, the insulation resistance data, the current
data, the total voltage data, the SoC data, the battery voltage data and temperature data.

The time data are used to identify the collecting time of the data. BMS collects and
records the data with a fixed time cycle, normally, so the time data are arranged in the
form of an arithmetic sequence.

The battery state-of-charge reflects the remaining battery power, is the ratio of the
remaining power and the capacity of the fully charged state, which ranges from 0 to 1.
SoC accuracy is 1% recorded by the BMS. And, BMS records only number before the
percent sign, normally. The state-of-charge data are slowly changing data, profile of SoC
data is shown in Figure 1. The insulation resistance data has changed dramatically, which
profile is shown in Figure 2.

The current refers to current provided power for the electric vehicles. Current data are
violent and irregular. Positive value of the current indicates that the electric vehicle is in
charge or in braking energy recovery stage, the negative value indicates that the electric
vehicle is in the driving stage. The accuracy of the current data is 0.1. The current data
profile is shown in Figure 3.

Electric vehicles need to monitor many batteries in the battery pack, and voltage data
and temperature data account for more than 80% of the total amount of data, so the
compression of the voltage and the temperature are the key to getting better compression
ratio.
Whether the electric vehicle is in the state of charge, braking state, driving state, the voltage data curves have a consistent trend, and the difference between any two voltage values is close at different times. Any 10 sets of voltage data are selected in battery data and the voltage data profile is shown in Figure 4.

**Figure 1** The state of charge data profile

![State of Charge Data Profile](image1)

**Figure 2** The insulation resistance data profile

![Insulation Resistance Data Profile](image2)
There is small temperature difference between different positions in the battery pack at the same time. The temperature data curves also have a consistent trend at different times. The temperature data and the voltage data have similar characteristics. Any five sets of the temperature data are selected in battery data, the voltage data profile is shown in Figure 5.
3 Battery data compression method for electric vehicles

In this paper, the compression ratio is the ratio of the data size after compression to the size before. In the universal compression algorithm, the data compression ratio depends on the data redundancy and repeatability (Ma et al., 2015; Patauner et al., 2011; Wu and Chung, 2015).

Therefore, according to the characteristics of the time data, SOC data, the voltage data and the temperature data, this paper carries on the pre-compression in order to improve the data redundancy and repeatability, and reduce the amount of data, then compresses the pre-compression processed data and other data further. Of the two universal compression algorithms, Huffman coding and run-length coding, different algorithms are used to compress the different data.

When using the universal compression algorithms to compress the data, based on analysis of the battery data characteristics and research of two universal compression algorithms, run-length coding algorithm is used to compress the time data, the insulation resistance data, the current data, the temperature data; and Huffman coding algorithm is used to the total voltage data and the voltage data further in this paper.

3.1 Pre-compression process

Pre-compression process is as follows:

The total voltage data and the voltage data of the battery are processed as follows:

There is one column in the total voltage data, \( n \) columns, \( n \) cell voltages, in the voltage data.
For the total voltage data, the difference between the total voltage and the \( n \) cell voltages is recorded. For the voltage data, the second cell voltage data to the \( n \)th cell voltage subtract the first cell voltage data, and record differences, respectively, at all times, for first cell voltage data, the voltage at the start time is set to the voltage sign and recorded, the voltages of remaining time subtract the sign, and record the difference, respectively.

The temperature data of the battery are processed as follows:

There are \( m \) columns, \( m \) temperature points, in the total temperature data.

For the temperature data, the second temperature point data to the \( N \)th second temperature point data subtract the first temperature point data, and record differences, respectively, at all times.

For first temperature point data, the temperature at the start time is set to the temperature sign and recorded.

Temperatures of remaining time subtract the temperature sign, and record the differences, respectively.

Time data and SoC data are processed as follows: there is one column by the time data, one column by the SoC data. The datum at the start time is set to the datum sign and recorded. The data of the remaining time subtract data of the previous time, and record the differences, respectively.

### 3.2 Universal lossless compression algorithm

#### 3.2.1 Huffman coding algorithm

Lossless compression algorithm is divided into statistic compression algorithm and dictionary compression algorithm. Huffman coding algorithm is based on statistic compression algorithm, and most representative lossless data compression algorithm (Chlopkowski and Walkowiak, 2015; Das and Rao, 2012; Nakatsuka et al., 2013). This algorithm encodes data based on the probability of the character. The higher the probability of the character in the original data, the shorter the corresponding encoding (Bok et al., 2010; Kontoyiannis and Verdu, 2014). Therefore, a relatively small amount of encoding can represent the original data. Battery data are numeric character and have high redundancy, the number of the character is less, so Huffman coding is suitable for compression of battery data. Huffman coding algorithm process is shown in Figure 6.

**Figure 6** Huffman coding process

- Reading input data
- Statistics for character frequency
- Input character sorting
- Establish Huffman tree
- Encoding character
- Encoding input data
Among them, the Huffman tree and the character encoding implementation process are as follows:

Suppose character set \{d_1, d_2, d_n\} need to be encoded under binary. Before that, the Huffman tree needs to be established.

Each character \(d_i\) is considered as a junction, also is the leaves, arranged from high to low according to the weights value \(w_i\) of the character \(d_i\). Two of minimum weight value junctions combine into a new junction and the weight value with branches, the new weight value is the sum of the two junction weight value. And continue the arrangement and combination until the Huffman tree is obtained.

The left branch of the tree is set to 0 and the right branch is set to 1. Then, each character \(d_i\) has only one path from the root to the leaves. All of the tree branches on the path correspond to the binary code of the character \(d_i\) (EomDooSeop, 2012; Nalbantoglu et al., 2010).

To Huffman encoding algorithm, for the character set \{d_1, d_2, d_3, d_4, d_5\}, byte size of the character set is 100. The corresponding character probability is \{40\%, 30\%, 15\%, 10\%, 5\%\}. The characters are arranged in the order of probability from large to small, shown in Table 1, The Huffman tree is shown in Figure 7, encoding results are shown in Table 2.

\[\text{Figure 7} \quad \text{Huffman tree (see online version for colours)}\]

<table>
<thead>
<tr>
<th>1</th>
<th>40</th>
<th>30</th>
<th>15</th>
<th>10</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>d_1</td>
<td>d_2</td>
<td>d_3</td>
<td>d_4</td>
<td>d_5</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>40</td>
<td>30</td>
<td>15</td>
<td>15</td>
<td></td>
</tr>
<tr>
<td>d_1</td>
<td>d_2</td>
<td>d_3</td>
<td>d_4</td>
<td>d_5</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>40</td>
<td>30</td>
<td>30</td>
<td></td>
<td></td>
</tr>
<tr>
<td>d_1</td>
<td>d_2</td>
<td>d_3</td>
<td>d_4</td>
<td>d_5</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>60</td>
<td>40</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>d_1</td>
<td>d_2</td>
<td>d_3</td>
<td>d_4</td>
<td>d_5</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>100</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>d_1</td>
<td>d_2</td>
<td>d_3</td>
<td>d_4</td>
<td>d_5</td>
<td></td>
</tr>
</tbody>
</table>
Research on battery data compression method for electric vehicles

Table 1  
Arrangement of the character

<table>
<thead>
<tr>
<th>Character</th>
<th>$d_1$</th>
<th>$d_2$</th>
<th>$d_3$</th>
<th>$d_4$</th>
<th>$d_5$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>40</td>
<td>30</td>
<td>15</td>
<td>10</td>
<td>5</td>
</tr>
<tr>
<td>Frequency</td>
<td>40%</td>
<td>30%</td>
<td>15%</td>
<td>10%</td>
<td>5%</td>
</tr>
</tbody>
</table>

Table 2  
The character encoding results

<table>
<thead>
<tr>
<th>Character</th>
<th>$d_1$</th>
<th>$d_2$</th>
<th>$d_3$</th>
<th>$d_4$</th>
<th>$d_5$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Encoding results</td>
<td>1</td>
<td>00</td>
<td>010</td>
<td>0110</td>
<td>0111</td>
</tr>
</tbody>
</table>

Size of character after compression is 205 bits.

\[ 1 \times 40 + 2 \times 30 + 3 \times 15 + 4 \times 10 + 4 \times 5 = 205 \]

The size of the original character storage space is 800 bits. The character set is compressed.

3.2.2 Run-length coding algorithm

The principle of run-length coding algorithm is to use a character instead of more characters with the same value, and to reach purpose of compressing data (Jiang and Li, 2010; Lin and Chang, 2009). The data can be compressed by run-length coding when the data are repeated characters (Zolghadr-E-Asli and Alipour, 2004; Koff and Shulman, 2006). Data after pre-compression and the insulation resistance data are suitable to be compressed by run-length coding.

The realisation of run-length coding algorithm is: repeating character and time replace repeated parts of the original data. A string, aaaaaaabbcccc, examples illustrate, length of the character is 16. The original string is replaced with the string a8b6c2. Length of the string a8b6c2 is 6. As can be seen from the example, compression rate is lower with the repeatability higher. Schematic diagram of run-length coding is shown in Figure 8.

Figure 8  
Schematic diagram of run-length coding

The characteristics of run-length coding algorithm is: run-length coding algorithm is very precise, one of the symbols has an error, which affects the entire encoding sequence so that the run-length encoding cannot be restored to the original data (Huang et al., 2005; Zhou, 2009). The compression rate obtained by the run-length coding depends largely on the characteristics of the data itself. The greater the redundancy of the data, the lower the compression rate (Jalilian et al., 2009; Siao et al., 2014). On the contrary, the greater the compression rate. But there is a drawback to the run-length coding algorithm,
if the data are ‘ABCABCABC’, using this algorithm will increase the data, the data encode into 1A1B1C1A1B1C1A1B1C, the amount of data is greater, it cannot achieve the compression effect (Qian and Zhang, 2012).

4 Experiments and results analysis

4.1 Battery data compression experiments

Battery experimental data of this paper come from an experimental electric bus with 324 ternary batteries. Two batteries connected in parallel, then all batteries are connected in series. Battery data of one day are selected as the experimental sample. Battery data are collected and estimated by the BMS, including the time data, the insulation resistance data, the current data, the total voltage data, the voltage data including 162 voltage points, the temperature data including 54 temperature points, SoC data. BMS data acquisition cycle is one second. The data sample contains 49,064 s, 656,565 byte size of the data.

The run-length coding and Huffman coding algorithm carry out a series of experiments. Pre-compression can reduce the compression ratio, two coding algorithms compress and decompress the data with pre-compression or without pre-compression, two sets of results are obtained, shown in Tables 3 and 4.

<table>
<thead>
<tr>
<th>Data</th>
<th>The run-length coding</th>
<th>The pre-compression and run-length coding</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time</td>
<td>0.3333</td>
<td>0.0762</td>
</tr>
<tr>
<td>Total voltage</td>
<td>0.2292</td>
<td>0.0840</td>
</tr>
<tr>
<td>Voltage</td>
<td>0.3607</td>
<td>0.2176</td>
</tr>
<tr>
<td>Temperature</td>
<td>0.1749</td>
<td>0.1458</td>
</tr>
<tr>
<td>SoC</td>
<td>0.0097</td>
<td>0.0034</td>
</tr>
<tr>
<td>All of the data</td>
<td>0.3523</td>
<td>0.2134</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Data</th>
<th>The Huffman coding</th>
<th>The pre-compression and Huffman coding</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time</td>
<td>0.4345</td>
<td>0.1875</td>
</tr>
<tr>
<td>Total voltage</td>
<td>0.3696</td>
<td>0.0643</td>
</tr>
<tr>
<td>Voltage</td>
<td>0.3877</td>
<td>0.2115</td>
</tr>
<tr>
<td>Temperature</td>
<td>0.3321</td>
<td>0.2055</td>
</tr>
<tr>
<td>SoC</td>
<td>0.2994</td>
<td>0.1255</td>
</tr>
<tr>
<td>All of the data</td>
<td>0.3821</td>
<td>0.2109</td>
</tr>
</tbody>
</table>
To verify the reasonableness of selecting universal compression algorithm for different data, based on the above experiments, two kinds of universal compression algorithms are used to do the experiment of compression and decompression, including data of time, total voltage, SoC, voltage, temperature after pre-compression process and insulation resistance, current, respectively. Run-length encoding and Huffman coding compression ratios for different types of data are shown in Table 5.

<table>
<thead>
<tr>
<th>Compression ratios</th>
<th>Run-length</th>
<th>Huffman coding</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time(after pre-compression)</td>
<td>0.0762</td>
<td>0.1875</td>
</tr>
<tr>
<td>Insulation resistance</td>
<td>0.1143</td>
<td>0.3530</td>
</tr>
<tr>
<td>Current</td>
<td>0.2604</td>
<td>0.3575</td>
</tr>
<tr>
<td>Total voltage(after pre-compression)</td>
<td>0.0840</td>
<td>0.0643</td>
</tr>
<tr>
<td>SOC(after pre-compression)</td>
<td>0.0097</td>
<td>0.1255</td>
</tr>
<tr>
<td>Voltage(after pre-compression)</td>
<td>0.2176</td>
<td>0.2115</td>
</tr>
<tr>
<td>Temperature(after pre-compression)</td>
<td>0.1458</td>
<td>0.2055</td>
</tr>
</tbody>
</table>

In a series of experiments, it reaches the requirement of lossless compression by comparing the data of decompression and before compression. The size of battery data before compression is 60,629,566 bytes. The size of battery data after compression is 12,744,336 bytes. Total compression ratio is 0.2102.

4.2 Results analysis

The compression ratio is lower when pre-compression processing is used, as can be seen from Tables 3 and 4 results. After pre-compression, run-length coding algorithm can achieve lower compression ratio than Huffman coding algorithm for the time data, the insulation resistance data, the current data, SoC data and the temperature data. Huffman coding algorithm can achieve lower compression ratio than run length coding algorithm for the total voltage data and the voltage data.

5 Conclusions

This paper presents a method of pre-compression for battery data according to the characteristics of the universal compression algorithm and the battery data. The characteristics of the battery data are analysed in detail. Two universal compression algorithms are described in detail. The effectiveness of the pre-compression method is verified by experimental results, and 10% lower compression ratio can be obtained. Strategies of choosing two universal algorithms for compression different battery data after pre-compression are proposed, which the effectiveness is verified by experimental results. The compression ratio obtained by the proposed battery data compression method is 0.2102.
Acknowledgements

This work is supported by Visiting Scholarship of State Key Laboratory of Power Transmission Equipment & System Security and New Technology (Chongqing University) (2007DA10512716416); Science Funds for the Young Innovative Talents of HUST, No. 201503.

References


