Journal of Computers Vol. 32 No. 5, 2021, pp. 128-141
doi:10.53106/199115992021103205010

Research on Remote Upgrade Method of Distributed Load Balancing for Power Monitoring Terminal

Yi-Qin Bao[1]*, Qiang Zhao[2], Hao Zheng[1], Hong-Tao Zhang[1],
Wen-Bin Xu[3], Zheng-Tang Sun[4]

[1] College of Information Engineering of Nanjing XiaoZhuang University, Nanjing 2111711, Jiangsu, China
baoyiqin@njxzc.edu.cn

[2] Department of Information Systems Schulich School of Business, Toronto 416647, Canada
Ryan.zhao@jmrex.com

[3] Nanjing Huazhu Industrial Intelligent Equipment Co., Ltd., Jiangsu 211175, China
392335241@qq.com

[4] School of Computer Science and Technology of Heilongjiang University, Harbin, China
1909095501@qq.com

Received 30 January 2021; Revised 20 April 2021; Accepted 23 April 2021

Abstract. With the increasing application of power monitoring terminal and the increasing demand for its function, it is necessary to upgrade the field embedded software remotely.The existing online upgrade schemes generally deal with the terminals one by one, and there are also distributed upgrade schemes. However, due to the large differences in the communication performance of terminals in different places, there is a problem of load imbalance, resulting in the low efficiency of the whole upgrade. Aiming at the efficiency of remote online upgrade of power monitoring terminal, a distributed load balancing remote upgrade method is proposed. The power monitoring terminal is upgraded at the same time through the cloud platform, and the genetic algorithm is used for load balancing processing, so as to improve the efficiency. The practical results show that this remote upgrade method improves the upgrade efficiency of the terminal by comparing a variety of upgrade schemes, compared with no load balancing method, the average time of upgrading a terminal is reduced by 36%, which achieves good application effect in practical application.

Keywords: remote upgrade, load balancing, genetic algorithm, power monitoring terminal, distributed system

1 Introduction

With the development of national power systems, it is crucial to fulfill the functionalities of real-time collection, dynamic monitoring, energy consumption analysis, cost accounting, performance evaluation and report release of the electricity. By doing so, we could refine energy management, promote energy saving and consumption reduction. Consequently, there is a growing demand for power monitoring and control. The power monitoring terminal is an online energy monitoring and analysis management system with the core purpose of energy saving and consumption reduction. It provides products, technologies, strategies, methods and information support for key energy users by comprehensively monitoring, diagnosing and analyzing the energy consumption and working conditions of energy-saving equipment, process equipment and key energy-consuming equipment. This could help us achieve energy saving and consumption reduction for the whole production line of key energy users [1].

* Corresponding Author

The power monitoring terminal is the central equipment of the power monitoring system, which communicates with the master station through a 4G wireless communication to realize remote monitoring [2]. However, as a monitoring device, its monitoring content will often change and adjust with the power supply department's needs, which requires remote in-application-programming (IAP) [3] upgrade of terminal software. The traditional solution is to have the manufacturer's technical staff update the program in person or via an in-system-programming (ISP). On top of that, the conventional solution also includes switching to the manufacturer's own upgrading platform for a remote upgrade. Both methods suffer from low upgrading efficiency and poor reliability. In some abnormal conditions, in-person maintenance is also required, especially for a large number of geographically dispersed equipment upgrades, which has a hefty workload and complicated system management. In this case, we need an efficient distributed upgrade platform to improve the efficiency of the remote upgrade.

At present, IAP has the following relevant research. Jiang et al. [4] proposed the IAP upgrade and remote upgrade technology of ARM7. Chao et al. [5] realized the on-site remote upgrade of embedded wireless mobile devices. Jiang et al. [6] discovered the online upgrade method triggered by software in 2012. Zhang et al. [7] realized the IAP upgrade technology of the Cortex M3 chips. Liu et al. [8] implemented the CAN upgrade technology for DSP chips in 2016. In 2018, hang et al. [9] made a systematic summary on the upgrade of embedded devices. Mirfakhraie et al. [10] studied the FTOA for car ECU upgrading protocol in 2019. Zhan [11] proposed the upgrade scheme of Automotive ECU based on UDS protocol. Lo et al. [12] studied the security upgrade of IoT equipment based on MQTT. Cian [13] proposed a wireless upgrade scheme based on the Lora communication. In 2020, Chen et al. [14] proposed the postal upgrade scheme of embedded equipment. Zhao et al. [15] proposed a multi-objective optimization method for the software upgrade.

The above research is mainly concentrated on upgrade technology, implementation method and the upgrade scheme of embedded software, without considering the load balance and upgrading efficiency of a distributed upgrade. In this paper, according to the demand of this kind of application, we will propose a load-balancing method based on a genetic algorithm for the power monitoring terminal. The remote upgrade system is realized by this method. The comparison with other upgrade systems shows that the technique is feasible and can improve upgrading efficiency.

This paper introduces the overall architecture of the remote upgrade system, analyzes the programming file (in HEX format), extends the communication protocol, uses the breakpoint continuation technology [16], and establishes the mathematical model of the genetic algorithm. Such an architecture makes the upgrading process load-balance, which improves the upgrading efficiency.

The contributions and innovations of this paper can be summarized as follows:

(1) The architecture of a distributed upgrade platform.

(2) Programming file (in HEX format) structure, communication protocol and breakpoint continuation of transmission.

(3) Using the mathematical model of the genetic algorithm to solve the load balancing of the upgrade task.

The rest of the paper is organized as follows. The second section introduces the remote upgrade system architecture and software structure. The third section analyzes the upgrade file structure, communication protocol and breakpoint continuation method. The fourth section proposes the load balancing method of the genetic algorithm. The fifth section compares and analyzes the experimental results. The sixth section summarizes the conclusion.

2 Remote Upgrade System

2.1 System Architecture

The power monitoring system mainly includes power monitoring terminals, 4G communications, and cloud platforms. The cloud platform comprises multiple communication servers, databases and WEB meter reading management, forming a distributed system [17], as shown in Fig. 1. The power monitoring terminal connects with the communication server of cloud platform through 4G wireless network and realizes data communication. Web meter reading management realizes remote monitoring and remote upgrade through cloud platform. However, as the number of power monitoring terminals connected to the server increases, reaching tens of thousands, it is necessary to adopt load balancing [18] technology in the system. The connection between the power monitoring terminal and the server is dynamic, accomplished by a task scheduling [19] program.

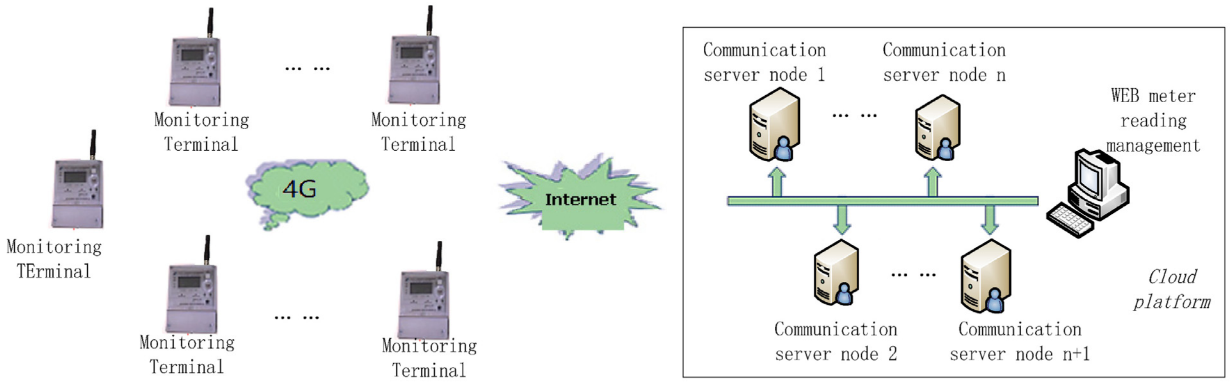
Monitoring Terminal
Monitoring Terminal
Monitoring TErminal
4G
Monitoring Terminal
Monitoring Terminal
Internet
Communication server node 1
Communication server node n
WEB meter reading management
Communication server node 2
Communication server node n+1
Cloud platform

Fig. 1. Architecture of electric energy monitoring system

Remote power monitoring and upgrading are fulfilled by exchanging data via connection to some specific sockets of the communication server. The distributed remote upgrade method developed in this article will significantly simplify and accelerate the upgrading process. More precisely, when you upgrade, you just need to upload the voltage monitoring terminal CPU upgrade file (in HEX format) into the distributed system. The system will automatically parse the HEX file and pack it into multiple frames according to the communication protocol. It is then distributed and sent to the communication server, where the task manager performs scheduling and forwards it to the power monitoring terminal to realize a remote upgrade.

2.2 Upgrade Platform Software Structure

As shown in Fig. 2, the system software can be divided into two components: the communication server and the application platform for the master station. The communication server interacts with the master station, receiving the messages sent by the master station. After the communication server finishes parsing the protocol, the communication server establishes the connection with the voltage monitoring terminal through communication task scheduling and communication interface. Application platform of master station includes remote monitoring data analysis, remote upgrade data control, breakpoint continuation, log management, abnormal data analysis and other modules.

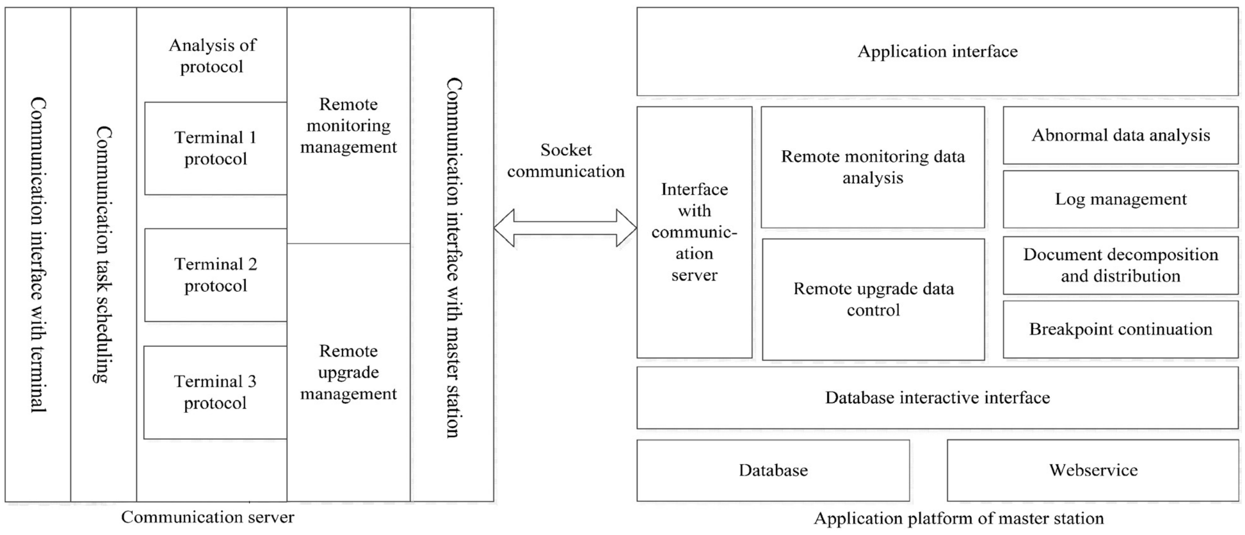
Communication interface with terminal
Communication task scheduling
Analysis of protocol
Terminal 1 protocol
Terminal 2 protocol
Terminal 3 protocol
Remote monitoring management
Remote upgrade management
Communication interface with master station
Socket communication
Application interface
Interface with communic-ation server
Remote monitoring data analysis
Remote upgrade data control
Abnormal data analysis
Log management
Document decomposition and distribution
Breakpoint continuation
Database interactive interface
Database
Webservice
Communication server
Application platform of master station

Fig. 2. Upgrade platform software structure

Since the voltage monitoring terminals produced by different manufacturers run on the same platform, the communication protocol and the upgrade platform must be unified. When the application platform of the master station needs upgrades, it first needs to analyze the files to be upgraded from different manufacturers [20] and follows the unified protocol. Later, it sends the corresponding files to the communication servers through the file decomposition and distribution module and the remote upgrade data control module. Lastly, by leveraging the distributed breakpoint resumable transmission, our method solves the problem of network reliability and multi-terminal simultaneous upgrade. In terms of security, it ensures the integrity of the upgrade file because it passes the security demonstration and the message verification.

On the interface of the application platform, users can see the upgrade status of each power monitoring terminal in the database. They can also view the data and quality analysis results by accessing the Web Service interface, which is very convenient for data analyzing and logs management.

3 Upgrade and Communication

3.1 Upgrade File Resolution

The power monitoring terminal adopts LPC2478 of ARM7TDMI-S and uses Keil simulation and development tools to generate the HEX file, which can be opened with Notepad. The format of each line is shown in Table 1:

Table 1. Hex file format

LEN	ADDR	TYPE	DATA	CS

LEN: the number of data bytes in the record, 1 byte.
ADDR: starting address, 2 bytes.
TYPE: record type, 1 byte, 04 refers to the beginning of hex file, 00 refers to the data record of hex file, 01 refers to the end of hex file.
DATA: HEX File data, LEN bytes.
CS: is the check sum of this line, 1 byte.
For example: a HEX file

```
:020000040000FA
:1020000018F09FE518F09FE518F09FE510F09FE5A8
:102010000CF09FE5846FA0B80CF09FE50CF09FE5F5
:102020002C21000034200000382000009420000003
:0CBF9000000000000040000040EC1F00001A
:00000001FF
```

(1) File header
:020000040000FA

02	is the number of data bytes in the record
0000	this field is always 0000
04	is the record type, which refers to the beginning of the file
0000	is the address of the segment
FA	is the checksum of the record

(2) File data
:1020000018F09FE518F09FE518F09FE510F09FE5A8

10	is the number of bytes recorded in this row
2000	is the starting address of data in memory (EPROM address to be generated)
00	record type 00 refers to a data record

18F0-9FE5 is hex file data

A8	is the checksum of the record

(3) End of document
:00000001FF

00	is the number of data bytes in the record
0000	this field is always 0000
01	record type 01, refers to the end of the document
FF	is the checksum of the record

3.2 Processor Flash Architecture

The structure of the processor Flash is shown in Fig. 3. The program to be written into the power monitoring terminal consists of two programs: boot.hex and main.hex. The boot.hex is the startup program. The start address starts from 0x0000 when compiling. The primary functionality is to look at the startup flag (4 bytes) when powering on. If all are 0xAA, enter the main program main.hex, if all are 0x55, the new main.hex program transmitted via 4G will overwrite the previous main.hex. The main.hex is the main program, which will be upgraded. The start address is from 0x2000 during compilation (if the upgrade function is not required, the start address starts from 0x0, and boot.hex is not needed), which mainly realizes the function of the power monitoring terminal.

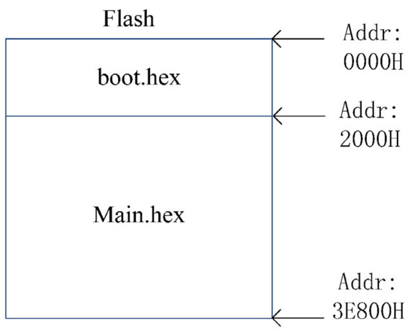

Fig. 3. Flash memory structure of processor

In order to perform the remote IAP upgrade, two hex files (boot.hex and main.hex) shall be merged using the file format mentioned in section 2.2 into a single file dy-main.hex before the terminal leaves the factory. The merge format is as follows:
Boot header
Boot body address starts from 0
Main body address starts from 0x2000
Main footer
dy- main.hex It is a power monitoring terminal program with IAP function. It needs to be written into LPC2478 in advance. Remote upgrade process is essentially transferring the main.hex file through the master station.

3.3 Communication Protocol

The remote upgrade relies on the standard protocol "Power Load Management System Data Transmission Protocol". The protocol uses the 6.2.4FT1.2 asynchronous transmission [21] format of GB/T18657.1. The protocol also uses a communication protocol similar to "session". More concretely, the master station is responsible for collecting and informing the device with the specified address code (usually called "terminal") to prepare to receive data. Then, the master station sends data frames containing different control codes to the terminal and waits for its answer. The data frame format specified by the protocol is shown in Table 2, where 68H is the starting character, A is the address field, C indicates the source and relationship between the messages before and after. Meanwhile, it is also responsible for controlling the message. L indicates that the length of the data field is a double word, and 2 Ls are set for comparison to enhance reliability. Lastly, CS is the sum of all bytes modulo 256 from the frame starting character to the check code to ensure the correctness of the transmitted data.

The link address field of the frame format contains administrative location code + terminal address (power monitoring terminal address), which is used as the unique identification of distributed data transmission and reception. User data contains all user type data, which is identified by function code and data unit identification to distinguish different types of user data, as shown in Table 3.

The main application function code AFN used in remote upgrade is shown in Table 4. When AFN = 0Fh, the data unit adopts F1 file transmission data format as shown in Table 5.

Table 2. Frame structure

Data Field	Identification
Beginning character	68H
Length	L
Length	L
Beginning characte	68H
Control	C
Address field	A
Link data	Data
Check Sum	CS
End of character	16H

Table 3. Link customer data frame format

Data field	ID
Application level function code	AFN
Frame sequence	SEQ
Data Unit ID (1-N)	DA, DT.
Data Unit (1-N)	UNIT

Table 4. Application Level function code definition

AFN	Meaning
00H	ACK/NACK
06H	ID AUTHENTICATION
0FH	FILE TRANSFER

Table 5. File transfer mode

Data content	Data format	Bytes
File id	BIN	1
File attribute	BIN	1
File code	BIN	1
Total fragment num	BIN	1
Segment i identification	BIN	1
Segment i data length Lf	BIN	2

3.4 File Transfer and Breakpoint Continuation

3.4.1 File Transfer

(1) The master station reads in main.hex, following the format in 3.1, up to the total number of rows RowNum. Then, it calculates the total number of messages TotalNum and packs 16 rows into a package.

(2) Convert the ASCII code characters of each line of data into hexadecimal code (a pack of 16 lines following the format of Table 5), and the data file stores 16 lines of data, which sums up to a total of 16*16 = 256 bytes, where the last line may be less than 256 bytes.

(3) Send data messages according to the file communication protocol mentioned in section 3.2.

(4) Add 2 bytes after the last frame of the data packet, which serves as the checksum CS of the entire file data.

(5) The terminal receives all messages and calculates the checksum CS1 of all message data. If CS is equal to CS1, the file transmission is correct, the device restarts, and enters the upgrade process.

3.4.2 File Breakpoint Continuation

Since the data transmission channel is wireless, there will inevitably be signal interference and packet loss. All upgrade processes use a breakpoint resumable transmission process, as shown in Fig. 4.

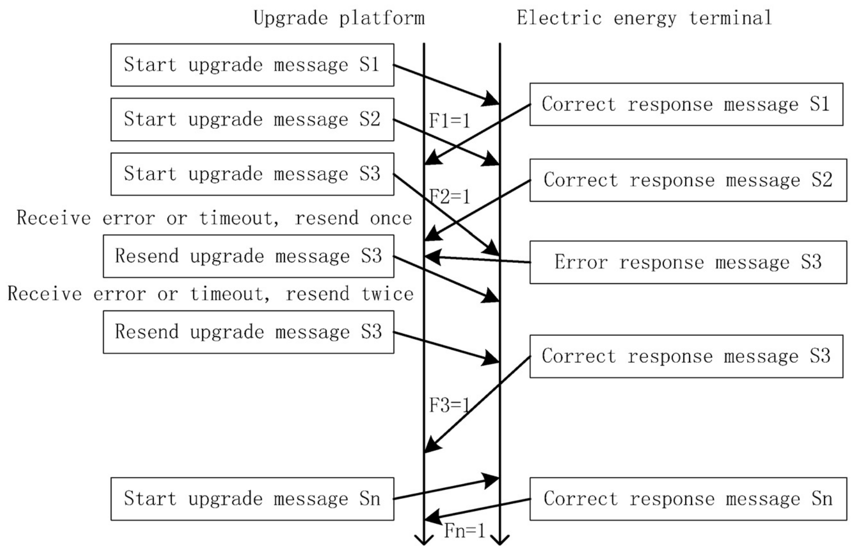
Upgrade platform
Electric energy terminal
Start upgrade message S1
Start upgrade message S2
Start upgrade message S3
Receive error or timeout, resend once
Resend upgrade message S3
Receive error or timeout, resend twice
Resend upgrade message S3
Start upgrade message Sn
Correct response message S1
Correct response message S2
Error response message S3
Correct response message S3
Correct response message Sn
F1=1
F2=1
F3=1
Fn=1

Fig. 4. Flow chart of file transfer and breakpoint continuation

Realization process: There are a total of n messages to transmit a file; the messages are from S1-Sn, the message transmission flag is from F1-Fn, and the initial flag Fi = 0 (i = 1~n). The upgrade platform starts by sending messages from S1, and for any message Si, if the correct response message is received, set the flag Fi = 1 and send the following message; if the error response message is received or the reception timeout (10 seconds), then retransmit. For the same message, if the correct response message cannot be obtained after retransmission up to three times, the following message will continue to be transmitted, and the flag Fi = 0 will be set. When the last message Sn is sent, the upgrade platform checks whether all the message flags Fi (i = 1~n) is 1. If any of Fi is 0, Si is a breakpoint. Repeat the above process to resume transmission with breakpoints until all the messages are marked as 1. Finally, compare the checksum and update the upgrade program if it is correct.

4 Load Balancing Method Based on Genetic Algorithm

In order to improve the efficiency of network upgrades, optimization of scheduling algorithms is needed. Li et al. [22] conducted a comparative study on related optimization algorithms. For the problems of network load balancing and traffic scheduling, Agarwal et al. [23] designed a dynamic management algorithm. For the SDN deployment scheme of network traffic, Zhang et al. [24] use genetic algorithms to solve the energy-saving problem. Based on the above research, this paper designs a genetic algorithm load balancing method.

Suppose there are N upgrade tasks (upgrade tasks of the power monitoring terminal), which are allocated to M virtual servers on the cloud platform for processing. The load balancer is used to allocate and schedule the length (upgrade file) of each task and the size of each virtual server. The processing speed is known. Please give a way to assign the tasks so that the total processing time of all tasks is the shortest. We are mapping this practical problem into a mathematical model of the genetic algorithm.

4.1 Mathematical Model of Genetic Algorithm

4.1.1 Definitions

(1) Length of upgrade task

The length of the upgrade task is represented by Tasks,

For example: Tasks = 200k bytes means that the length of the upgrade file is 200kb.

(2) Processing speed of energy meter and server node

The processing speed of the server node is represented by the array nodes, nodes [j] (j = 1~M), where j is the node number, and nodes [j] represents the processing speed of node j.

For example, M = 10, nodes = [10, 12, 15, 20, 10, 8, 8, 8, 5, 8] nodes [0] indicates that the processing speed of the server is 10kb/s.

(3) Task processing time array

After the task tasks and node nodes are determined, the task processing time assigned by all tasks to all nodes can be determined, which is represented by the array timeMatrix.

timeMatrix[i] represents the time required to assign tasks to node i for processing. It is calculated by the following formula:

timeMatrix [i] = tasks/nodes [i](i = 1~M)

For example: timeMatrix = [20.0, 16.7, 13.3, 10.0, 20.0, 25.0, 25.0, 25.0, 40.0, 25.0].

(4) Chromosome

Each chromosome is a solution for assigning all tasks to the server. Each evolution will produce N chromosomes. Each chromosome is a feasible solution to the current problem. The feasible solution is composed of multiple elements. Each element is called a gene. A chromosome array is used to record the feasible solution in each evolution process of the algorithm. The chromosome group is represented by: Chromosome [i] (i = 1~N).

A chromosome is a one-bit array, the subscript of the one-bit array represents the number of the task, and the value of the array represents the number of the node. Then the meaning of chromosome [i] = j is: Assign task i to node j.

For example: chromosome [0] = 2 means task 0 is assigned to node 2.

(5) Fitness

In the genetic algorithm, fitness function plays the role of God, which judges the adaptability of each chromosome, retains chromosomes with high fitness, and eliminates chromosomes with poor fitness. Then when the algorithm is implemented, an array of fitness is needed to record the fitness of the current N chromosomes (N tasks), expressed as adaptability [i] (i = 1 ... N)

The subscript of the adaptability array represents the number of the chromosome, and adaptability [i] represents the fitness of the chromosome number i.

In the example of remote upgrade load balancing scheduling, the total execution time of N upgrade tasks is used as the criterion for fitness evaluation. When all tasks are assigned, the maximum value of chromosome genes is fitness, that is, the total time to perform tasks. The shorter the total duration, the better the fitness and vice versa.

(6) Select operation

We can see that in each evolution process mentioned above, the probability of each chromosome being selected in the next evolution needs to be calculated according to the fitness. The calculation formula is shown in formula 1:

$$\text{selectionProbability } [i] = \text{adaptability } [i] / \sum_{i=1}^{N} \text{adaptability } [i] \tag{1}$$

(7) Cross operation

Crossover operations are using multiple points to cross the chromosomes. Firstly, the location of multiple crossing points is randomly generated, and then the two chromosomes exchange some gene codes at crossing points, thus forming the sub individuals. See Formula (2) - (4) for cross formula.

$$T = \text{newpop } [i].\text{chrom } [j] \tag{2}$$

$$\text{newpop } [i].\text{chrom } [j] = \text{newpop } [i+1].\text{chrom } [j] \tag{3}$$

$$\text{newpop}[i+1].\text{chrom } [j] = T \tag{4}$$

(8) Mutation operation

For each chromosome in the population, the gene values of some genes will be randomly mutated with some specific probability.

Finally, a total of 100 evolutions will be carried out through the above algorithm, and 100 chromosomes will be generated for each evolution. The abscissa in the figure represents the number of evolutions, and the ordinate represents the task execution time.

4.1.2 Mathematical Model Algorithm

According to the mathematical model of genetic algorithm, the design algorithm flow chart is shown in Fig. 5.

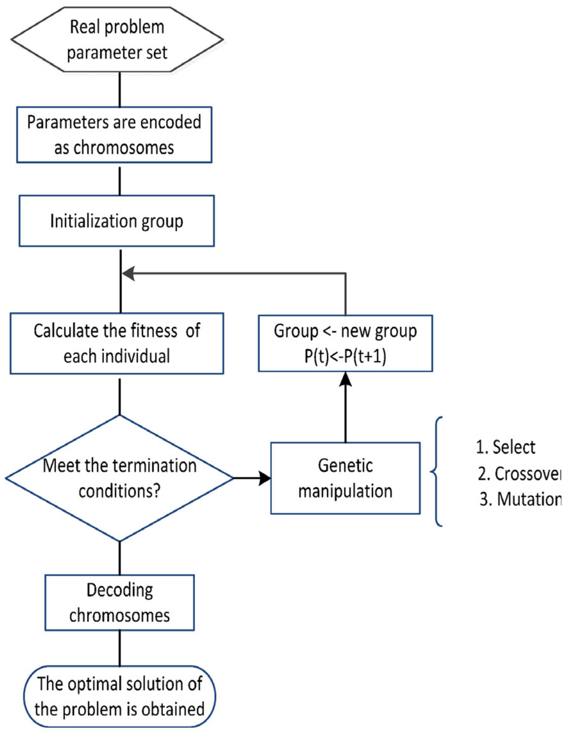
Real problem parameter set
Parameters are encoded as chromosomes
Initialization group
Calculate the fitness of each individual
Group <- new group P(t)<-P(t+1)
Meet the termination conditions?
Genetic manipulation
1. Select
2. Crossover
3. Mutation
Decoding chromosomes
The optimal solution of the problem is obtained

Fig. 5. Flow chart of genetic algorithm

Algorithm 1 describes the execution process of the genetic algorithm. Firstly, feed in the graph, total deployment cost and the maximum number of hereditary times, and use a random method to construct the initial population (lines 1-3); then, calculate the initial population fitness corresponding to all individuals using the fitness function (line 4); then we will start the iterative process, where each iteration performs selection, crossover, and mutation operations (lines 7-18); finally, the fitness of the lately generated individual is calculated (line 12).

Algorithm 1. genetic algorithm

```
Input: graph G=(V, E), total deployment cost is H, maximum genetic times is S.
Output: deployed server node collection P($P \subseteq V$).
1.  determine the range of nodes that can be upgraded
2.  number of randomly generated nodes M
3.  initialize the population
4.  calculate the fitness of all individuals in the initial population
5.  gen = 0
6.  While gen < S do
7.      Selection
8.      Crossover
9.      If $\sum_{i=1}^{m} h(v_i) \leq$ H then
10.       Mutation
11.       If $\sum_{i=1}^{m} h(v_i) \leq$ H then
12.           Calculate the population fitness of new individuals
13.       else
```

```
14.             Mutation again
15.         EndIf
16.     else
17.         Crossover again
18.     EndIf
19  EndWhile
20. Store individuals in P
Return  P
```

4.2 Running Results of Mathematical Model

Run algorithm 1: GeneticAlgorithm (sizepop, vardim, bound, MAXGEN, params), where sizepop is the population size, vardim is the variable dimension (the number of terminals to be upgraded), bound is the variable boundary (the number of servers), MAXGEN is the maximum number of inheritance, and params is [crossover rate, mutation rate, alpha]. Two ways to simulate running:

(1) The task of specifying the size of the upgrade file is run in simulation. Setting: Tasks = 200kb, running ga = GeneticAlgorithm (100, 100, [1, 10], 1500, [0.8, 0.2, 0.5]), the results are as follows:

(2) The load balancing scheduling results, as shown in Fig. 6, gradually converge with the increase of genetic times.

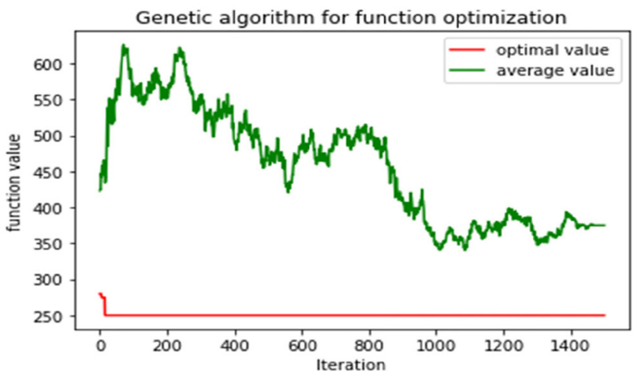

Fig. 6. Load balancing scheduling result curve based on genetic algorithm

(3) Load balancing node allocation results

(a) Best fitness: 250.0

(b) Best chromosome:

Chromosome = [8, 9, 1, 2, 4, 5, 6, 5, 5, 3, 3, 5, 2, 7, 8, 3, 10, 4, 5, 4, 7, 10, 1, 2, 3, 3, 4, 7, 8, 7, 7, 7, 2, 10, 4, 6, 4, 2, 6, 3, 9, 2, 3, 7, 1, 2, 5, 8, 6, 2, 6, 2, 10, 10, 4, 4, 6, 8, 3, 1, 10, 10, 10, 8, 7, 2, 1, 3, 2, 6, 6, 5, 5, 3, 5, 2, 9, 7, 9, 8, 3, 1, 3, 6, 6, 3, 3, 4, 4, 4, 5, 3, 2, 9, 2, 8, 8, 2, 7, 8]

(c) The results of node assignment are shown in Table 6.

Table 6. Number of tasks assigned by nodes

Node identifier	Quantity (Piece)
1	6
2	15
3	15
4	11
5	10
6	10
7	10
8	10
9	5
10	8

(d) Load balancing distribution histogram, as shown in Fig. 7.

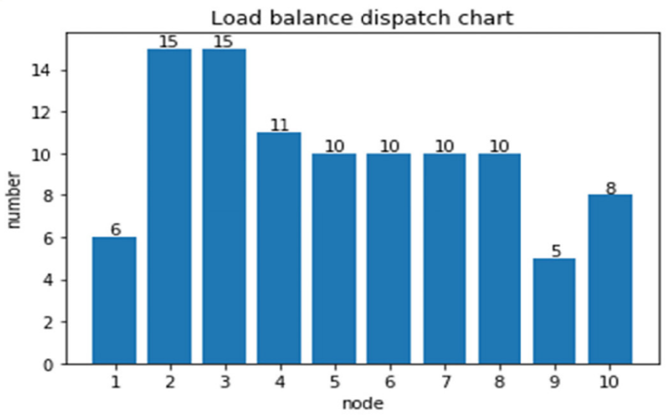

Fig. 7. Load balancing histogram

The task of changing the size of the upgrade file is simulated. The file size ranges from 50 to 500 kb, and the results are shown in Table 7.

Table 7. Upgrade schedule with file size

Upgrade file size (kBytes)	Time (Min)
50	62.5
100	120
150	187.5
200	250
250	325
300	360
350	420
400	480
450	540
500	625

The data in Table 7 is simulated as a curve, as shown in Fig. 8. As the size of the upgrade file increases, the upgrade time increases linearly.

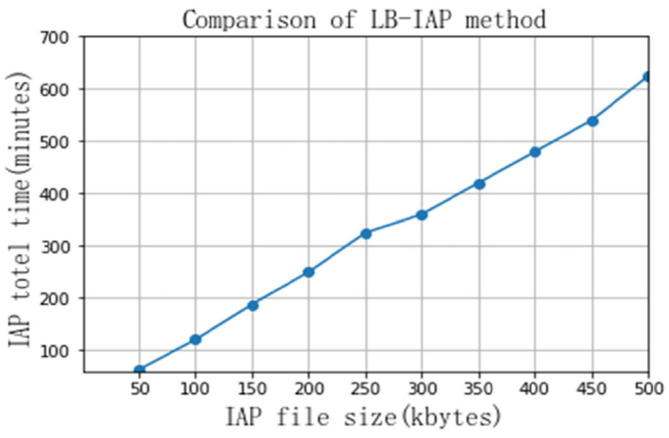

Fig. 8. Upgrade time curve with file size

5 Testing and Comparison of the System

For the remote upgrade function of the electric energy monitoring terminal, three schemes are used for testing and comparison. The first method is the standard IAP upgrade method (CM-IAP), which completes the upgrade tasks one by one; the second method is the distributed upgrade method (DT-IAP), where the terminals are all upgraded at the same time; the third method, the load balancing upgrade method (LB-TAP), is to upgrade the terminals at the same time through the genetic algorithm. There are two test comparison schemes: the real-time test of the specified upgrade file size and the simulation test with different upgrade file sizes.

(1) Real-time test of the specified upgrade file size. The comparison of results is shown in Table 8.

Table 8. Comparison of test results of remote upgrade methods

Upgrade method	Terminal Number (Pieces)	IAP file size (kBytes)	Lost message rate (%)	Success rate (%)	Time consuming (Min)
CM-IAP	100	210.78	3.8%	100	1998
DT-IAP	100	210.78	3.5%	100	396
LB-IAP	100	210.78	3.4%	100	247

From the test results, we can see that the load balancing upgrade method (LB-TAP) takes the least time to upgrade 100 terminals, which only lasts for 247 minutes. On average, it takes 2.47 minutes to upgrade one terminal, which has the highest efficiency. The common IAP upgrade method (CM- IAP) has the longest upgrade time, with an average of 19.98 minutes, and the efficiency is the lowest. Because both DT_IAP and LB-IAP use distributed upgrades, the upgrade time is significantly reduced compared to CM-IAP, the average time to upgrade a terminal is reduced from 3.96 minutes to 2.47 minutes, which is reduced by 36%. Moreover, since LB-IAP uses a genetic algorithm-based load balancing method, it further decreases the upgrade time compared with DT-IAP. Although all of the three upgrade methods have packets lost during the wireless upgrade, their success rate all reaches 100% due to resumable breakpoint transmission. In conclusion, the analysis shows: LB-TAP is the most effective upgrade method.

(2) In the simulation test with different upgrade file sizes, the file size ranges from 50 to 500 kb, and the comparison of the results is shown in Fig. 9.

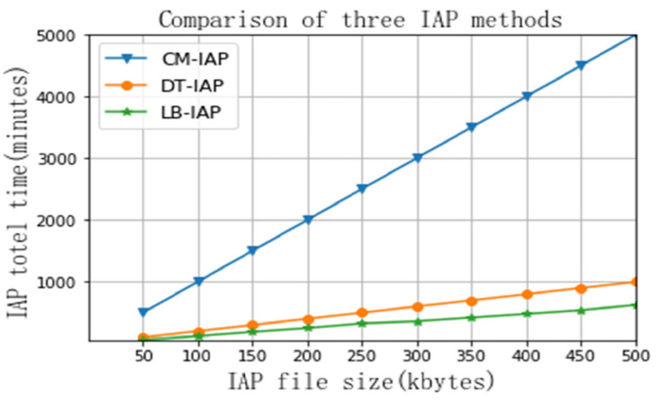

Fig. 9. Comparison chart of upgrade time with upgrade file size

From the visualization of the test results, we can intuitively see that the upgrade time shows a linearly increasing trend for all of the three upgrade methods. Furthermore, among all three upgrade methods, the LB-IAP curve has the smallest slope, indicating this upgrade method has the same effect despite file size. In other words, this result illustrates the wide range and versatility of this method.

6 Conclusions

This paper proposed a load balancing upgrade method for remote online upgrade of electric energy monitoring terminals to improve efficiency. Through the introduction of the remote upgrade software platform, the analysis of the upgrade file, the realization of the communication protocol, the adoption of the breakpoint resumable transmission technology, and the load balancing processing based on the genetic algorithm, a distributed remote upgrade for the power monitoring terminal is designed. The experiments and comparison results show that this load balancing upgrade method improves the upgraded efficiency, the average time to upgrade a terminal is reduced by 36%. Although this IAP upgrade method can be used for reference in other embedded systems, further analysis and research are needed to achieve software definability and versatility.

The follow-up work will be on the versatility of the upgrade method. With the continuous development of smart grids, there will be a large number of terminals that need to be upgraded. How to use a unified upgrade platform to upgrade all terminals, such as through software definable methods, and if the versatility of the method can be achieved, has substantial economic and practical value.

Acknowledgements

This study was supported by the Natural Science Foundation Project of China (61976118), the Natural Science Foundation Project of Jiangsu Province (BK20180142), and the key topics of the "13th five-year plan" for Education Science in Jiangsu Province (B-b /2020/01/18).

References

[1] M. Alibakhshikenari, I. Huynen, A Comprehensive Survey of Metamaterial Transmission-Line Based Antennas: Design, Challenges, and Applications, IEEE Access 8(2020) 144778-144808.

[2] M. Alibakhshikenari, F. Babaeian, B. Virdee, S. Assa, K. Kumar, A Comprehensive Survey on Various Decoupling Mechanisms with Focus on Metamaterial and Metasurface Principles Applicable to SAR and MIMO Antenna Systems, IEEE Access 8(2020) 192965-193004.

[3] J.-W. Tian, J. Liu, X.-X. Liu, W.-H. Qi, Design and implementation of remote upgrade platform for power consumption information acquisition terminal, Computer applications and software 31(1)(2014) 330-333.

[4] X.-M Jiang, X.-H. Li, Z.-R. Ren, M. Yao, IAP online upgrade and teleupgrade resolvent based on ARM, Journal of Computer Applications 28(2)(2008) 519-521.

[5] F. Chao, B.-G. Yu, Open Remote Field Upgrade of Embedded Wireless Mobile Device, CEA 43(1)(2007) 3-3.

[6] J.-C. Jiang, Z.-S. Wang, H.-Z. Feng, T. Liu, Design and implementation of IAP on-line upgrading technology based on software trigger, Journal of Computer Applications 32(6)(2012) 1721-1723.

[7] W.-J. Zhang, Y.-M. Nan, Design and implementation of IAP techniques based on STM32F103VB, Journal of Computer Applications 29(10)(2009) 2820-2822.

[8] R. Liu, A method of remote update for DSP system based on CAN bus, Information Technology and Network Security, 35(13)(2016) 71-72,75.

[9] W.-L. Hang, G.-R. Zhan, Design of online software updating for embedded devices, Electronic Design Engineering, (14)(2018) 167-171.

[10] T. Mirfakhraie, G. Vitor, K. Grogan, Applicable Protocol for Updating Firmware of Automotive HVAC Electronic Control Units (ECUs) Over the Air, in: Proc. 2018 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE

Smart Data (SmartData), 2018.

[11] K.-X・Zhan, Auto ECU Upgrade Scheme Base On UDS Protocol, Computer Applications and Software 36(1)(2019) 191-196, 203.

[12] N.-W. Lo, S.-H. Hsu, A Secure IoT Firmware Update Framework Based on MQTT Protocol, in: Proc. International Conference on Information Systems Architecture and Technology, 2019.

[13] G. Cian, Efficient Firmware Update Transmission for LoRa Low Power Wide Area Technology, Dublin２:Trinity College Dublin, 2019・

[14] Y. Chen, Postal upgrade scheme for embedded devices, Computer science 47(6a)(2020) 648-651.

[15] S.-H. Zhao, Z.-L. Ren, H. Jiang, Multi objective optimization method for software upgrade problem, Computer science 047 (006)(2020) 16-23.

[16] H. Dai, Design and implementation of file timing transmission software based on FTP, Computer applications and software 30(1)(2013) 332-333.

[17] Z. Wang, A distributed component model with OpenMP thread Affinity Support, Computer applications and software 30(3)(2013) 203-206.

[18] S. Jin, Q. Li, Dynamic load balancing scheme based on multi-objective genetic algorithm, Computer engineering and science 35(12)(2013) 102-106.

[19] F. Qiao, Grid task scheduling algorithm with fuzzy multi-objective, Computer engineering and science 36(9)(2014) 1644-1649.

[20] X. Hao, Q. Zong, Q.-X. Li, R. Xu, Hypersonic vehicle real time simulation platform based on dSPACE, Computer applications and software 31(2)(2014) 52-54.

[21] C.-B. Xu, Real time analysis of IEEE-1394b optical bus interconnection, Science, technology and engineering 12(21)(2012) 5347-5351.

[22] Y.-L. Li, S.-Q. Wang, Q.-R. Chen, X.-G. Wang, Comparative Study of Several New Swarm Intelligence Optimization Algorithms, CEA 56(22)(2020) 1-12.

[23] S. Agarwal, M. Kodialam, T. Lakshman, Traffic engineering in software defined networks, in: Proceedings of IEEE INFOCOM, 2013.

[24] J. Zhang, H. Wang, S.-T. Luo, A hybrid software based on genetic algorithm to define network routing energy saving algorithm, Computer science 47(06)(2020) 242-247.