

Edge Computing Framework and Resource Scheduling Algorithm based on Smart Grid

Xiaoyi Yu^{1,*}, Youchan Zhu¹, Jinlei Qin¹

¹School of Control and Computer Engineering, North China Electric Power University, Baoding 071003, China.

*Corresponding author Email: 2192221003@ncepu.edu.cn

Abstract

With the rapid development of smart grids and the Internet of Things, the number of smart applications in the grid has increased sharply, and the number of mobile terminals has increased, resulting in an increasing load on the cloud infrastructure, so edge computing has emerged. Edge computing uses infrastructure closer to the user side to process data at the edge of the network, which not only improves the response speed of the system, but also reduces the cost for system transmission. Based on edge computing, this paper first proposes an edge computing architecture suitable for smart grids, and then based on this framework, proposes a GAFCM-based maximum and minimum resource scheduling algorithm (GF-MaxMin) for edge computing resource scheduling. Finally, a simulation experiment is performed to compare and analyze with the other two algorithms, and the effectiveness of the algorithm is verified by the two evaluation indicators of average execution time and user satisfaction.

Keywords

Smart Grid; Cloud Computing; Edge Computing; Resource Scheduling.

1. Introduction

The storage and processing of data in smart grids are generally carried out in cloud computing. Although cloud data centers have strong computing and storage capabilities [1], with the rapid development of the Internet of Things, smart grid data has exploded. Cloud computing brings some problems and challenges [2]. In this context, edge computing emerged as a new computing model, enabling data to be processed in a timely and effective manner near the source, which not only improves the response speed of the system, but also reduces the demand for system transmission [3].

At present, the edge computing framework proposed by domestic and foreign research is often general [4, 5], but for the edge layer in the smart grid, when the edge definition is too small, the edge layer still needs to be configured with a large number of edge nodes. The cloud pressure has not been reduced; when the edge definition is too large, the edge node will process a lot of calculations, which will easily cause information blockage at the node and cannot guarantee the real-time characteristics of the edge node. Therefore, in view of the application characteristics of the smart grid itself, it is necessary to propose an edge computing framework adapted to the smart grid.

Common basic resource scheduling algorithms include MaxMin algorithm [6] and artificial intelligence algorithms such as genetic algorithm, simulated annealing algorithm, particle swarm algorithm, and ant colony algorithm [7-10]. The object of most resource scheduling algorithms is all resources [11, 12]. Resource scheduling is carried out from all resources, and the search range is large, which leads to large other costs such as time and energy consumption.

And without considering the characteristics of the resource itself, it is not possible to provide services on demand.

This article first designs an edge computing framework based on the smart grid environment. The framework is divided into terminal layer, edge layer and cloud layer. In addition, the edge layer is further divided into first-level edge layer and Secondary edge layer. The first-level edge layer meets the task requirements of the medium-voltage platform area, and the second-level edge layer meets the task requirements of the low-pressure platform area. Then based on this framework, an edge computing resource scheduling algorithm is proposed, namely the maximum minimum resource scheduling algorithm based on GAFCM (GF-MaxMin). The algorithm first uses the fuzzy C-means clustering algorithm optimized based on genetic algorithm (GAFCM) to aggregate resources. Classes, which divide resources into computing, bandwidth, and storage. Then for tasks in the smart grid, search for resources in the same type of resource clustering, and allocate resources through the MaxMin algorithm. Finally, the Matlab simulation platform is used to compare the algorithm with the other two algorithms to verify the feasibility and effectiveness of the algorithm.

2. Edge Computing Framework of Smart Grid

The article proposes a layered architecture of smart grid based on edge computing. The article proposed consists of three parts: terminal equipment layer, edge layer and cloud layer. In order to further alleviate the pressure of the clouds, the edge layer is divided into a first-level edge layer and a second-level edge layer. The tasks of the middle-pressure platform area are handled by the first-level edge layer, and the tasks of the low-pressure platform area are handled by the second-level edge layer. Such a double-layer edge structure can not only better share the pressure of the clouds, but also not easily cause information blockage at the nodes, and the real-time performance is stronger. The architecture diagram is shown in Figure 1.

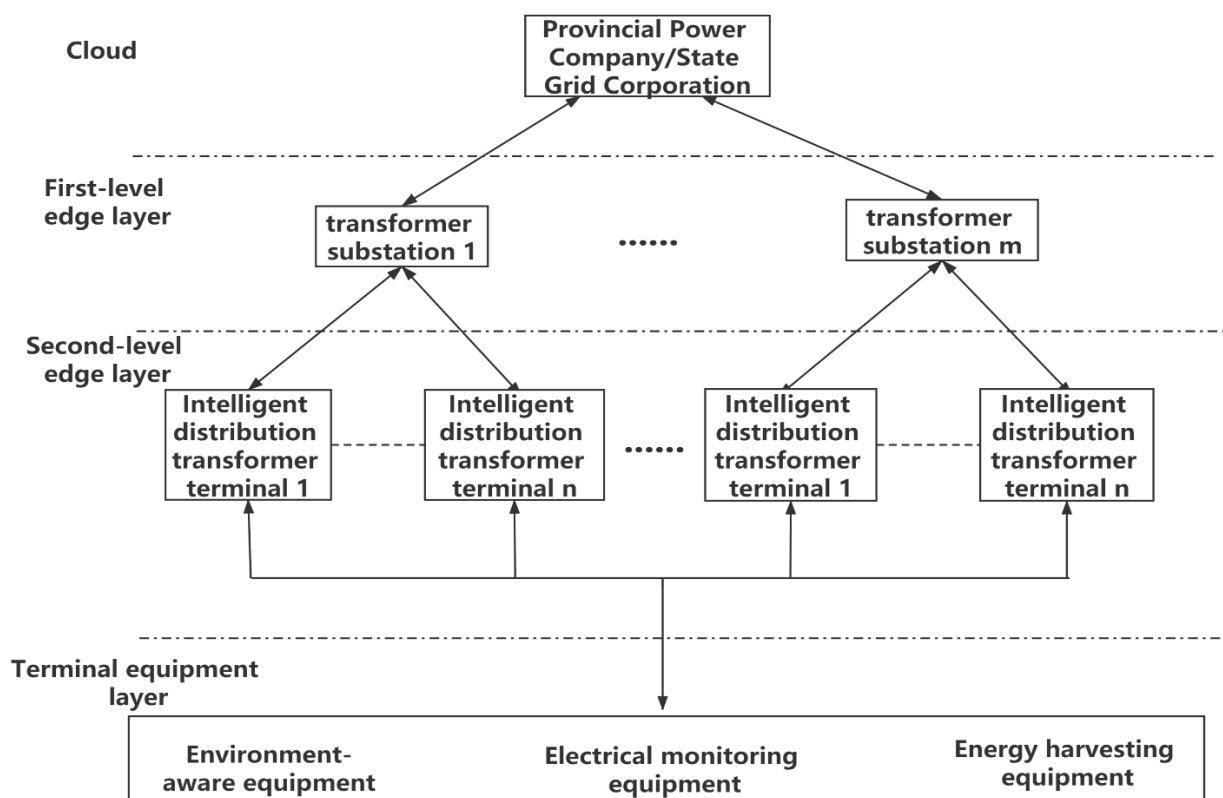


Fig. 1 Edge computing framework of smart grid

The terminal layer is the data source of the entire architecture. It mainly consists of three types of equipment: electrical monitoring equipment, environmental perception equipment, and energy collection equipment. These devices monitor, sense, collect, and control the operating status, operating environment, and users' electricity consumption information of electrical equipment in the smart grid through existing technologies.

The cloud layer is the layer with the largest amount of tasks in the entire architecture and requires optimal servers. The cloud layer is generally deployed in the provincial power company or the state grid corporation. In addition to processing edge upload tasks, the cloud layer also needs to schedule and allocate edge nodes, and more importantly, monitor, manage, and control the entire smart grid to ensure safe and stable operation.

The edge layer is located between the terminal layer and the cloud layer. It first receives data from the terminal. The data that cannot be processed is further uploaded to the cloud layer. The processed result of the cloud layer is then sent to the terminal through the edge layer. Edge tasks include three categories: prediction and early warning, classification and clustering, and demand response. Prediction and early warning are used to analyze big data to predict the future electricity consumption of users such as load and electricity prices; classification and clustering are to provide more accurate feature information and classify terminal user data; finally, different users can be formulated according to the prediction results. Demand response. The construction of edge nodes is generally built through intelligent distribution terminals such as DTU, FTU, TTU, but the computing power is limited and can only handle some simple tasks, so the secondary edge layer can be deployed in the intelligent distribution terminal, responsible for the low-voltage power distribution station area. When the task is difficult and the number of tasks is large, the first-level edge layer can be deployed in the substation, so as to meet the needs of users while further alleviating the pressure on the cloud layer.

3. Resource Scheduling Algorithm

In this paper, the resource set in the edge node of the intelligent smart grid is represented by $R = \{r_1, r_2, \dots, r_p\}$, where $p = |R|$ is the number of resources. The resource in the edge node is represented by a three-dimensional vector $r_i = \{r_c, r_b, r_m\}$, and the meaning of each attribute is as follows: 1) r_c represents the calculation size of the resource; 2) r_b represents the bandwidth size of the resource; 3) r_m represents the storage size of the resource. The three attribute vectors of the resource determine the performance of the resource, so the comprehensive performance of the resource is:

$$r_f = \sqrt{\frac{a(r_c)^2 + b(r_b)^2 + c(r_m)^2}{a + b + c}} \quad (1)$$

Among them, a , b , and c respectively represent the empirical coefficients of resource calculation size, bandwidth size, and storage size.

In this paper, the task set at the terminal equipment of the intelligent smart grid is represented by $T = \{t_1, t_2, \dots, t_q\}$, where $q = |T|$ is the number of tasks. The tasks in the task set are further represented by a five-dimensional vector $t_j = \{t_c, t_b, t_m, t_d, t_{type}\}$, and the meaning of each attribute is as follows: 1) t_c represents the task's computing demand for resources; 2) t_b represents the task's bandwidth demand for resources; 3) t_m represents the task's storage of resources Requirements; 4) t_d represents the size of the task; 5) t_{type} represents the task category. When $t_{type} = 1$, the task is computationally sensitive. When $t_{type} = 2$, the task is bandwidth-sensitive. When $t_{type} = 3$, Indicating that the task is a storage-sensitive task.

MaxMin algorithm is a heuristic resource scheduling algorithm. The idea of this algorithm is to map large tasks to fast servers. First, calculate the expected completion time of the task to the server, and assign the server with the largest expected completion time to the corresponding

server. Task, delete the task after the assignment, and then update the ready time of the server, repeat the above operation until all tasks are mapped to the server.

However, the object of MaxMin algorithm is all resources. With the explosive growth of terminal equipment in smart grid, the number of edge nodes, namely the number of resources, also increases exponentially. Every time resources are allocated to tasks, all resources need to be searched. The search range is Larger, leading to other overheads such as time and energy consumption. Therefore, it is proposed that the fuzzy C-means clustering algorithm (FCM) is used to divide and cluster the resources, which are divided into computing, bandwidth and storage resources. According to the task category, the tasks are matched to the corresponding resource clusters, and the tasks are scheduled appropriately in the clustering. Resources. Although the FCM algorithm is simple and easy to understand, it still has some problems. It is sensitive to the initial value and has a local optimum. GA algorithm has the characteristics of global optimization. Therefore, this paper proposes to introduce GA algorithm when FCM generates initial clustering centers to solve the problem of initial value sensitive and easy to converge to local extremum.

In summary, this article proposes a GAFCM based maximum and minimum resource scheduling algorithm (GF-MaxMin) under the framework of the smart grid. The flowchart is shown in Figure 2 below. The algorithm first performs data standardization and normalization processing on resources to avoid uneven impact on the clustering results due to different attribute value dimensions, and then uses GA optimized FCM clustering algorithm to cluster resources, which are divided into computational and bandwidth types and storage resources, and then use MaxMin algorithm to allocate resources in the corresponding resource clusters for tasks.

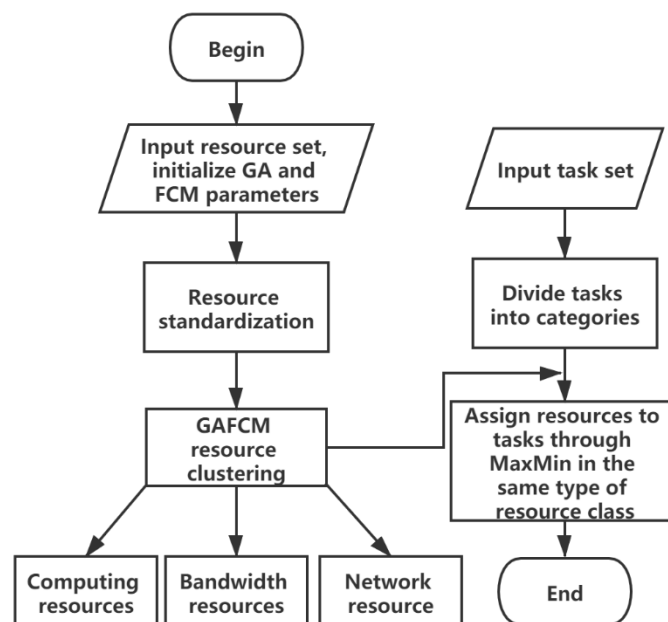


Fig. 2 GF-MaxMin resource scheduling algorithm

4. Simulation

4.1. Experimental Design

The Matlab simulation platform is used to experimentally verify the algorithm of this paper. The parameter settings of the algorithm in the paper are shown in Table 1 below. The attribute characteristics of resources, tasks and the number of tasks are randomly generated within the specified range.

Table 1. Parameter Settings

Test parameters	Numerical value
Number of resource nodes	100
Maximum number of tasks	1000, the number of tasks is random
Resource node computing power/(Mbit·ms ⁻¹)	[100,1000]
Resource node bandwidth capability/(Mbit·ms ⁻¹)	[100,1000]
Resource node storage capacity/M	[512,5120]
Task computing requirements/(Mbit·ms ⁻¹)	[100,1000]
Task bandwidth requirements/(Mbit·ms ⁻¹)	[100,1000]
Task storage requirements/M	[512,5120]
Task length/M	[50000,250000]

The algorithm in the paper is compared with Gale Shapely algorithm and Berger algorithm. The performance indicators for comparison are average execution time and user satisfaction. The average execution time is the sum of the transmission time for the task to be transmitted to the edge node and the calculation time for the task to be processed on the node. User satisfaction is determined by the ratio between the expected demand of the task and the actual resource obtained, so the calculation formula for user satisfaction is as follows:

$$U_{sat} = \ln(\sum \frac{\alpha * t_{jc}}{r_{ic}} + \frac{\beta * t_{jb}}{r_{ib}} + \frac{\gamma * t_{jm}}{r_{im}}) \quad (2)$$

Among them, α , β , and γ are the empirical coefficients of task calculation, bandwidth, and storage resources requirements, t_{jc} , t_{jb} , and t_{jm} are the calculation requirements, bandwidth requirements, and storage requirements of task j , respectively, and r_{ic} , r_{ib} , and r_{im} are respectively The computing power, bandwidth capacity, and storage capacity of resource i allocated by j .

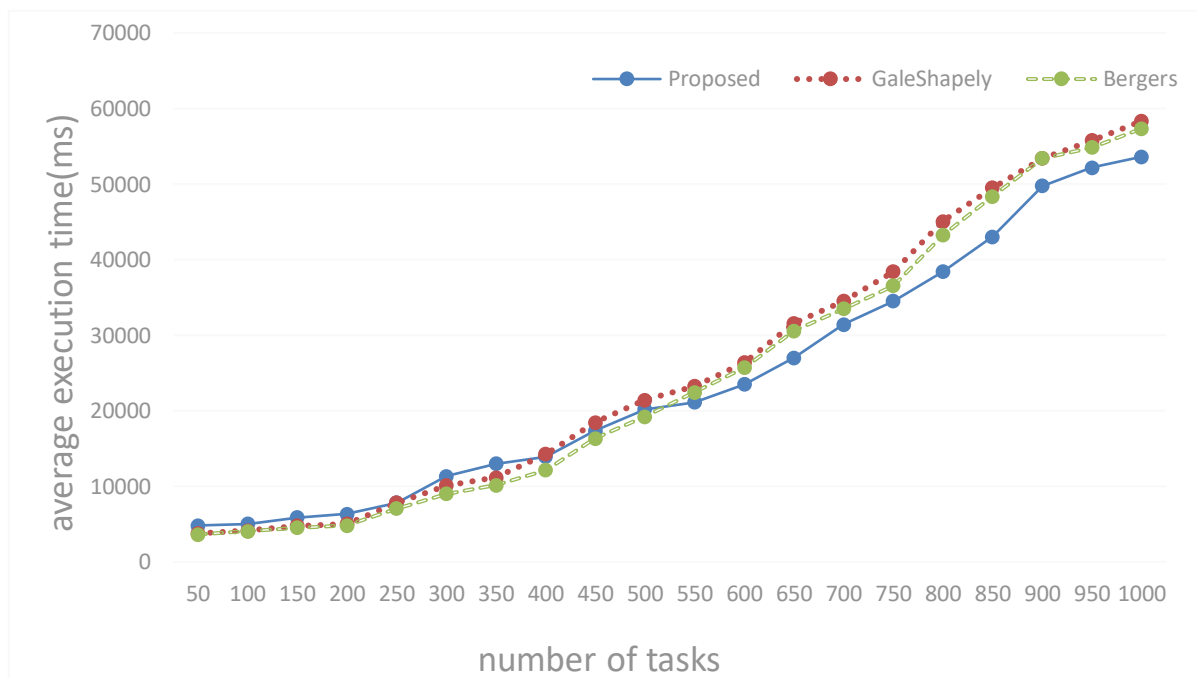


Fig. 3 Average execution time comparison

4.2. Simulation Results

As shown in Figure 3, when the amount of tasks is small, the average execution time of the GF-MaxMin algorithm is longer than that of the other two algorithms, but as the amount of tasks increases, the average execution time of the algorithm is relatively small. This is because the

algorithm needs to update the cluster center and the degree of membership during clustering. The advantage of clustering is not obvious, so the time is longer. However, when the amount of tasks is large, the time spent on clustering is relatively small, and it is almost negligible. Allocating resources in the corresponding resource class can reduce the resource traversal time, so the average execution time is relatively small.

As shown in Figure 4 below, the user satisfaction of the GF-MaxMin algorithm has always been higher than that of the other two algorithms. This is because the algorithm clusters resources and allocates the same type of resources to tasks, taking into account the task's demand for resources.

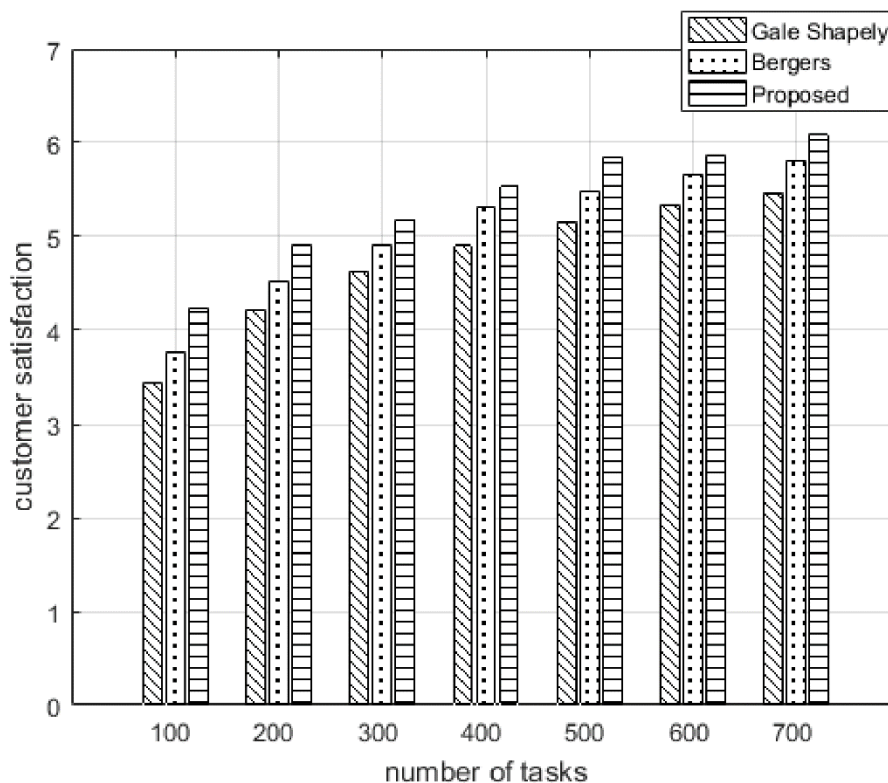


Fig. 4 User satisfaction comparison

5. Summary

This paper first proposes an edge computing architecture adapted to the smart grid. The framework is divided into terminal layer, edge layer and cloud layer. The edge layer is further divided into a primary edge layer and a secondary edge layer. The first-level edge layer meets the task requirements of the medium-voltage platform area, and the second-level edge layer meets the task requirements of the low-pressure platform area. Then based on this framework, an edge computing resource scheduling algorithm is proposed. The algorithm uses genetic algorithm to optimize fuzzy clustering to clusters resources, and allocates appropriate resources through MaxMin algorithm in the corresponding resource clustering according to the task category. Finally, through experiments and analysis, it can be seen that when the task volume is large, the new algorithm ensures a high user satisfaction while ensuring a short average execution time.

References

- [1] Zhuang W, Huang L. Overview of Cloud Computing Resource Allocation and Management Technology [C]// 2019 6th International Conference on Systems and Informatics (ICSAI). 2019.

- [2] Albataineh H, Nijim M, Bollampall D. The Design of a Novel Smart Home Control System using Smart Grid Based on Edge and Cloud Computing[C]// 2020 IEEE 8th International Conference on Smart Energy Grid Engineering (SEGE). IEEE, 2020.
- [3] Cao K, Liu Y, Meng G, et al. An Overview on Edge Computing Research[J]. IEEE Access, 2020, PP(99):1-1.
- [4] Hu P, Chen W. Software-Defined Edge Computing (SDEC): Principles, Open System Architecture and Challenges [C]// 2019 IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computing, Scalable Computing & Communications, Cloud & Big Data Computing, Internet of People and Smart City Innovation (SmartWorld/SCALCOM/UIC/ATC/CBDCom/IOP/SCI). IEEE, 2019.
- [5] Amjad A, Rabby F, Sadia S, et al. Cognitive Edge Computing based resource allocation framework for Internet of Things[C]// Second International Conference on Fog & Mobile Edge Computing. IEEE, 2017.
- [6] Derakhshan M, Bateni Z. Optimization of tasks in cloud computing based on MAX-MIN, MIN-MIN and priority [C]// 2018:45-50.
- [7] Gao Y G, Song D Y. A New Improved Genetic Algorithms and its Property Analysis [C]// Third International Conference on Genetic & Evolutionary Computing. IEEE, 2009.
- [8] Xin-Hua T, Xu C, Zhi-Feng F. A Multi-objective Genetic Algorithm Based on Simulated Annealing[J]. IEEE, 2012.
- [9] Sun X, Hu P, Li Y, et al. An Improved Algorithm for Multi-Swarm Particle Swarm Optimization Based on Clustering Algorithm [C]// 2018 IEEE 4th International Conference on Computer and Communications (ICCC). IEEE, 2018.
- [10] Dai Y, Lou Y, Lu X. A Task Scheduling Algorithm Based on Genetic Algorithm and Ant Colony Optimization Algorithm with Multi-QoS Constraints in Cloud Computing [C]// International Conference on Intelligent Human-machine Systems & Cybernetics. IEEE, 2015:428-431.
- [11] Pan S, Qiao J, Jing J, et al. Distributed Resource Scheduling Algorithm Based on Hybrid Genetic Algorithm [C]// 2017 International Conference on Computing Intelligence and Information System (CIIS). IEEE Computer Society, 2017.
- [12] Chen N, Fang X, Xin W. A cloud computing resource scheduling scheme based on estimation of distribution algorithm. IEEE, 2015.