Design of distributed timing task scheduling system for smart grid

WenSheng Tang\textsuperscript{1a*}, JinZhi Wang\textsuperscript{1b}, Tao Fan\textsuperscript{2c}

\textsuperscript{1}Marketing Department of State Grid Corporation of China, Beijing, 100031, China
\textsuperscript{2}Internet Department of State Grid Corporation of China, Beijing, 100031, China
\textsuperscript{a*}mufengqin0603@163.com

Abstract—With the rapid development of information technology and the growing scale of enterprise development, more and more enterprise application systems appear due to business scenarios, requiring the system to execute the specified business operations at the planned time, which is the demand of "timed tasks". The rapid growth of business volume makes the number of timed tasks also increase massively, and the importance of timed task application scenarios requires more and more strict guarantee of reliable triggering of tasks. In this paper, to address these problems, we design and implement a distributed system providing timed task scheduling service based on Quartz, an open source lightweight job scheduling framework, and realize business modules such as timed task control service, trigger execution, lock grabbing trigger, unified configuration management, task load balancing and log management. The system has been tested and put into operation, and has completed the requirements well and achieved the design goals.

1. Introduction

Grid dispatching is an important element of power supply enterprises in the management of grid operation and maintenance, power supply, transmission, substation, distribution and power safety management, and the overall quality and efficiency of grid dispatching work directly determines the level of power service capability of power supply enterprises. Efficient and scientific grid dispatching specification and implementation mode is an important guarantee for power supply enterprises to achieve good economic and social benefits. With the rapid development and popularization of smart grid, the informationization of grid dispatching operation has made rapid development in power supply enterprises all over China. The traditional microcontroller-based timing scheduling system uses the establishment of scheduling pairs to schedule according to priority. Not only is it difficult for the system to handle a large number of dispatch requests, but it also tends to cause scheduling system anomalies [3-5]. The scheduling system from the perspective of economic value constraint only considers the influence of a single economic variable and ignores the carrying capacity of grid scheduling, which is prone to the fact that some scheduling tasks are always in the waiting state and it is difficult to meet the actual scheduling demand when applied independently [6]. The scheduling algorithm using particle swarm and fuzzy logic has improved the scheduling efficiency of the grid to a certain extent, but there are limitations when applied to large smart grids. Therefore, in this paper, we will use Quartz, a lightweight job scheduling framework, to design a distributed timed task scheduling system for smart grids for the actual needs of task scheduling for current smart grid operations.
2. Design of Distributed Timed Task Scheduling System for Smart Grid

The overall system architecture consists of five main components, which are a scheduling cluster based on Quartz implementation, a cluster of executors responsible for executing tasks, a message queue RabbitMQ for connecting the scheduling cluster to the executor cluster and distributing tasks[7], a Zookeeper cluster for executor cluster management and as a Dubbo registry, and a MySQL database.

Scheduling node cluster: The task scheduling service in this system is based on Quartz, but with an improved working model. RabbitMQ: RabbitMQ is a bridge between task senders and task receivers, and the queues in RabbitMQ act as staging points for tasks, with each queue storing a different type of task. Executor clusters: Executor clusters are specific executors of tasks in the system, the number of nodes in the cluster can be added and removed dynamically according to specific needs, and the executor nodes can be deployed in a simple and fast way.

2.1. Scheduling system hardware design

The distributed timed task scheduling system for smart grid designed in this paper adopts a top-down design approach, adding FPGA chips in the message storage process to improve the stable reliability[8]. Figure 2 below shows the overall hardware framework schematic of the timed task scheduling system.

In this paper, the system uses DDR2 SDRAM as the cache area for data, while DDR2 read/write control is implemented with an FPGA master controller chip. Considering leaving a certain redundancy, MT47H128M16RT-25E is selected as the data cache chip in the design, whose capacity size is 2GB and the maximum data operation rate of the chip can reach 800Mbps. The hardware part of the distributed timing scheduling system communicates through PCIE bus and adopts independent
point-to-point communication links to transmit data so that it can work at a very high frequency, and each pair of differential lines has a data transfer rate of 2.5 Gbps to realize the transmission communication of scheduling commands.

2.2. Task load balancing scheduling module design
Assume that in the operation of smart grid, there are $n$ different task servers to process $m$ different task requests[9]. Let the task execution node set be $N = \{ p_1, p_2, \ldots, p_n \}$, and the energy consumption set per task time of each task execution node is $C = \{ c_1, c_2, \ldots, c_n \}$, where $c_i$ represents the energy consumption of node $p_i$. When scheduling tasks in power grid operation, the task scheduling objective function is established with the minimum total operating energy consumption of the system scheduling server as follows:

$$\min_{\lambda_i^s, \lambda_f^s} E = \sum_{n=1}^{N} \sum_{s=1}^{S_n} C_n^s T_i$$  \hspace{1cm} (1)

In the above equation, $C_n^s$ denotes the energy consumption of the $s$th server as it performs its task. $T_i$ is the task scheduling execution time for the $s$th server on the grid. $\lambda_n^s$ is the total number of tasks assigned to the $n$th task execution node by grid scheduling. $f_n^s$ is the frequency of operation of the $s$th task execution node of the $n$th server. The binary variable $Y_n^s$ indicates the activation status of a server. If a server is activated, the binary variable takes the value of 1, otherwise the binary variable takes the value of 0. In order to achieve load balancing among server nodes during the grid task scheduling process, the execution nodes avoid congestion in the task scheduling queue due to uneven task load, which affects the system task scheduling efficiency. Therefore, the following scheduling constraints are set in this study. (1) Load balancing constraints:

$$\lambda_m = \sum_{n=1}^{N} \sum_{s=1}^{S_n} \lambda_n^s$$  \hspace{1cm} (2)

In the above equation, $\lambda_m$ is the task flow transmission speed submitted by the task demand side during grid scheduling. The above formula indicates that the total tasks to be performed by all server nodes should be equal to the total tasks reaching all front-end nodes. (2) Scheduling service delay constraint:

$$\frac{Y_n^s}{rf_n^s} + \frac{\lambda_m^s \bar{x}_{n}^{s2}}{2 \left( 1 - \frac{\lambda_m^s}{rf_n^s} \right) (krf_n^s)^2} \leq T$$  \hspace{1cm} (3)

Where $\bar{x}_{n}^{s2}$ is the average size of the task. $r$ is a scaling factor. $k$ is the scaling factor, which can be considered as a known parameter that indicates the expectation of the task size. $T$ is the maximum delay allowed by the activated server.

(3) The operating frequency constraint of each server, i.e., the operating frequency of each server cannot exceed the maximum frequency $f_{max}^s$.

$$0 \leq f_n^s \leq f_{max}^s, Y_n^s \in \{0, 1\}$$  \hspace{1cm} (4)

After setting up the power grid task scheduling objectives and constraints in the Quartz framework, the genetic algorithm is used to solve the above minimization problem, and a scheduling strategy for different mission requirements is obtained.
2.3. Task scheduling and trigger management module design
Task creation table TskTbl, task readiness table TrdTbl, event request table ErqTbl and timed waiting table TmrTbl are four independent tables. Each task list is a 64 bit one-dimensional table, and each task in the system occupies a binary bit in this one-dimensional table. The specific task list is shown in table 1.

| Symbolic Name (of a thing) | Instructions |
|---------------------------|--------------|
| TskTbl[N] Task creation form | Flag that a task has been created, as a flag for operations associated with the task. \([n]=1: n\)th task was created; \([n]=0: n\)th task was not created/deleted. |
| TrdTbl[N] Mission Ready Form | Flag that a task is ready to be used by the task scheduler. \([n]=1: n\)th task is ready; \([n]=0: n\)th task is not ready. |
| ErqTbl[N] Event Request Form | Flag that a task has an event request, used by the task scheduler. \([n]=1: n\)th task has an event request; \([n]=0: n\)th task has no event request. |
| TmrTbl[N] Timed Waiting List | Flag that a task has timed waits, used by the task scheduler. \([n]=1: n\)th task has timed waits; \([n]=0: n\)th task has no timed waits. |

When managing the task trigger records, cold execution data separation is used to separate the instance data from the historical data in the trigger run, which can improve the efficiency of accessing the instance data of the running tasks.

2.4. Task Timing Trigger Module Design
The main implementation classes are Executor, SimplePool, ParallelPool, etc., where SimplePool and ParallelPool are the task execution pool classes corresponding to simple tasks and parallel computing tasks respectively, and there are ConcurrentHashMap data structures in these execution pool classes. executorUnitTable is used to cache the Job instances running in the current Client, each instance will correspond to an ExecutorUnit class. In addition, the Executor class is responsible for connecting the clientService (RPC service interface class) and the task execution pool class, and is responsible for forwarding the RPC requests from the Server side to the corresponding task execution pool class for business processing. The parallel computing task execution pool working timing diagram is shown in figure 3 below.

When the system performs timed task scheduling, it sets the task trigger timing and performs parallel task scheduling calculations according to the working timing diagram shown in figure 3 above.
2.5. Log management module design
The log management module is responsible for recording task execution logs, providing information update services for task execution records and task log query functions. The classes and descriptions related to the log management module are shown in table 2.

| Interface Kind | Instructions |
|----------------|--------------|
| updateJobRecord | This interface defines the remote call service |
| uploadLog(String) | This class provides a concrete implementation of the remote call service |

The task execution records are updated and logs are uploaded mainly through Dubbo, a distributed RPC framework. The scheduling node is a service provider and the execution machine is a service consumer. The LogRPCService interface defines the remote call service interface provided by the module. This completes the design of the distributed timed task scheduling system for smart grid.

3. System experiments

3.1. Experimental content
In this paper, we have designed a distributed timed task scheduling system for smart grid and after completing the system design, the online system is tested for the functionality of each module of the system.

3.2. Experimental results
Test the scheduling management module and task timing trigger module of the scheduling system using test data. Access to the simulated Server side was used as the scheduling request for each functional module to run. The execution of the test cases by each functional module of the system shows that all test cases pass and each functional module functions normally. In the integration test environment, during the integration test of each software functional module of the system, the business timing scheduling logic is normal and the test results are as expected. For high availability testing, the system is brought online into a small smart grid with different sizes and numbers of dispatch requests generated by the computer. The average response time of the scheduling system is tested for different scheduling task requests. Figure 4 below shows the response time test of the system.

![Fig.4 Scheduling system response time test](image)

Analysis of the information in the above figure shows that as the number of scheduling task requests in the smart grid increases, the scheduling response time of the system shows a growing trend, but the overall growth is small, and the maximum response time is only 2463 ms. It shows that the
system can respond to the scheduling requests in the process of grid operation in a timely manner and perform task scheduling processing quickly. That is, the distributed timed task scheduling system of smart grid designed in this paper has good operation performance, and the on-line test results show that the system works well.

4. Conclusion

Grid system automation is an important means to ensure the safety, stability, reliability and economic operation of the power system. The management of power grid's operation state is adjusted, changed and maintained, so that the power supply and distribution of power grid can be orderly implemented according to the requirements. On the basis of maintaining the overall stable and safe operation of the power grid, the quality and mode of power supply in the power supply area can be achieved. In this study, a smart grid distributed timing task scheduling system is designed. After the on-line test, the functions of each module are in good condition, which can meet the task scheduling requirements of smart grid. Since our current working environment is a local area network, the impact of network delay and communication is not considered in the scheduling algorithm. Therefore, further research will focus on the distributed task scheduling of the power automation system in the wide area network, and the determination of the node load after the network delay is added.

References

[1] Zhou Hailing. (2021). A multi-objective scheduling method for smart grid loads based on improved GSO algorithm. Journal of Physics: Conference Series, 1907(1), 012018.
[2] Ali Sajjad; Khan Imran; Jan Sadaqat; Hafeez Ghulam. (2021). An Optimization Based Power Usage Scheduling Strategy Using Photovoltaic-Battery System for Demand-Side Management in Smart Grid. Energies, 14(8), 2201-2201.
[3] Hao Liang; Sharad Sinha; Wei Zhang. (2018). Parallelizing Hardware Tasks on Multicontext FPGA With Efficient Placement and Scheduling Algorithms. IEEE Trans. on CAD of Integrated Circuits and Systems, 37(2), 350-363.
[4] Paul Suraj; Chatterjee Navonil; Ghosal Prasun. (2021). Dynamic task allocation and scheduling with contention-awareness for Network-on-Chip based multicore systems. Journal of Systems Architecture, 115(2),102020.
[5] Abbasi Sohaib Iftikhar; Kamal Shaharyar; Gochoo Munkhjargal; Jalal Ahmad; Kim Kibum. (2021). Affinity-Based Task Scheduling on Heterogeneous Multicore Systems Using CBS and QBICTM. Applied Sciences, 11(12), 5740-5740.
[6] Duan Jiandong; Liu Fan; Yang Yao; Jin Zhuanting. (2021). Flexible Dispatch for Integrated Power and Gas Systems Considering Power-to-Gas and Demand Response. Energies, 14(17), 5554-5554.
[7] Jo Haesung; Park Jaemin; Kim Insu. (2021). Environmentally Constrained Optimal Dispatch Method for Combined Cooling, Heating, and Power Systems Using Two-Stage Optimization. Energies, 14(14), 4135-4135.
[8] Zare Mohsen; Narimani Mohammad Rasoul; Malekpour Mostafa; Azizipanah-Abarghooe Rasoul; Terzija Vladimir. (2021). Reserve constrained dynamic economic dispatch in multi-area power systems: An improved fireworks algorithm. International Journal of Electrical Power and Energy Systems, 126(PA), 106579-.
[9] Dey Bishwajit; Raj Saurav; Mahapatra Sheila; MSheila Fausto Pedro Garcro. (2022). Optimal scheduling of distributed energy resources in microgrid systems based on electricity market pricing strategies by a novel hybrid optimization technique. International Journal of Electrical Power and Energy Systems, 1 34(10):107419.