J Inf Process Syst, Vol.17, No.6, pp.1035~1043, December 2021
https://doi.org/10.3745/JIPS.04.0230

ISSN 1976-913X (Print)
ISSN 2092-805X (Electronic)

JOURNAL OF INFORMATION PROCESSING SYSTEMS JIPS

Study on Data Processing of the IOT Sensor Network Based on a Hadoop Cloud Platform and a TWLGA Scheduling Algorithm

Guoyu Li* and Kang Yang*

Abstract

An Internet of Things (IOT) sensor network is an effective solution for monitoring environmental conditions. However, IOT sensor networks generate massive data such that the abilities of massive data storage, processing, and query become technical challenges. To solve the problem, a Hadoop cloud platform is proposed. Using the time and workload genetic algorithm (TWLGA), the data processing platform enables the work of one node to be shared with other nodes, which not only raises efficiency of one single node but also provides the compatibility support to reduce the possible risk of software and hardware. In this experiment, a Hadoop cluster platform with TWLGA scheduling algorithm is developed, and the performance of the platform is tested. The results show that the Hadoop cloud platform is suitable for big data processing requirements of IOT sensor networks.

Keywords

Cloud Computing, Hadoop, Internet of Things, Sensor Network

1. Introduction

Sensors in Internet of Things (IOT) networks constantly sense (i.e., monitor) physical conditions in the environmental for relevant movements (or changes) and respond based on preprogrammed rules [1-3]. Hence, the sensor network is the basis of the IOT. With the expansion of the IOT sensor network, various intelligent sensors have been used in many applications, making high-speed data acquisition and processing the key issues for IOT sensor networks. As single nodes contain a large number of sensors working simultaneously, real-time processing of generated mass data is highly required. Further, the managed node removes redundant data based on judgment. If all work is performed by one single node, this node will be abnormally busy and fail to respond in time, leading to the paralysis of the entire network. Cloud computing for IOT networks have been developed to enable the processing of massive amounts of IOT data in real time [4,5]. Cloud computing is a practical method to manage large amounts of resources: it divides computing processes into many subroutines and distributes these subroutines to many spare servers across the cloud computer network. After calculations and analysis, the results are transported to the end user. Cloud computing is characterized by mass data storage, high-speed data

Manuscript received August 11, 2021; first revision September 29, 2021; accepted October 10, 2021.
Corresponding Author: Guoyu Li (lgy@hdc.edu.cn)
* School of Information Engineering, Handan University, Handan, China (lgy@hdc.edu.cn, kyang_hd@126.com)

analysis, and real-time processing [6,7]. Hadoop is a distributed computing framework, which can run applications on a large number of low-cost hardware devices in a cluster. The Hadoop-based framework is used for big data collection, processing, and storage in many applications, such as air pollution monitoring, fault detection, and disaster management [8,9]. In this study, fiber Bragg grating and traditional sensors form a sensor network to collect temperature data continuously. Temperature data are continuously collected, stored, and accumulated by computing nodes and finally form different sizes of data packets. To enhance the ability of real-time data processing of the sensor network, a cloud-computing-based data processing platform for sensor networks is investigated. Furthermore, the performance of the Hadoop cluster platform with time and workload genetic algorithm (TWLGA) is studied.

2. IOT Sensor Network Based on Cloud Computing

In an IOT sensor network, when a sensor node collects and processes massive amounts of data in real time, the node will become busy and fail to respond in real time. To avoid node paralysis, cloud computing technology is used to distribute data from one busy node to idle nodes. These nodes share resources with one managed node. In this manner, data processing is not concentrated in a few busy nodes, but is distributed over many idle nodes. All the calculated results are transported back to the managed node. Therefore, cloud computing technology not only enhances the real-time processing ability of the nodes but also ensures high-speed acquisition and analysis of massive data. As the resources are calculated by a series of idle nodes, rather than a few busy nodes, cloud computing technology reduces the risk of computing workload. Therefore, a cloud-computing-based IOT sensor network is more reliable, manageable, and flexible.

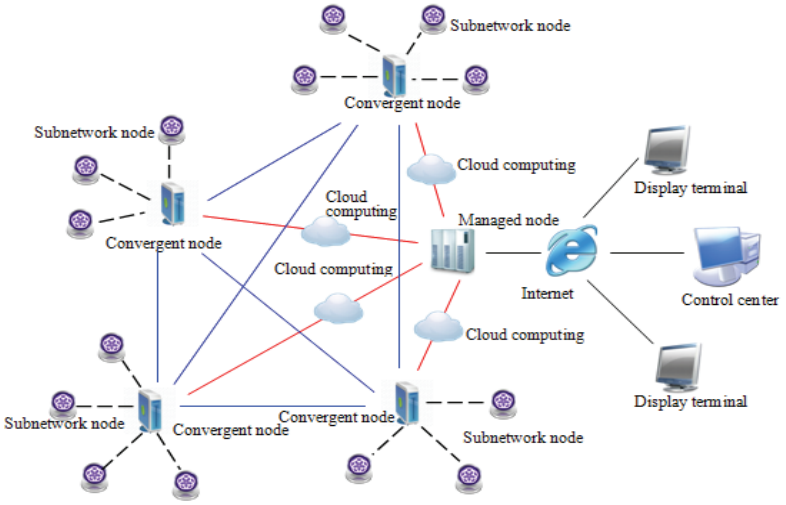

Fig. 1. IOT sensor network based on cloud computing.

The data processing of the IOT sensor network is constantly utilized by using the theoretical model of cloud computing. The overall technical schematic diagram of an IOT sensor network based on cloud

computing is shown as Fig. 1. In the figure, many subnetworks constitute the IOT sensor network, and each subnetwork contains a series of sensors. The sensing elements of the subnetworks monitor the variation of parameters. The convergent nodes are connected each other, and each convergent node connects to many subnetworks. The convergent nodes are all connected to the managed nodes using the cloud computing technology. Based on information of the managed node, data resources are calculated by a series of idle nodes, rather than the busy nodes. The managed node is controlled from a control center through commands over the Internet. A display terminal is used to display analysis results of the IOT sensor network.

3. Time and Workload Genetic Algorithm

Generally, a Hadoop cluster comprises computing hardware and software, and the scheduler does the assignment of specific tasks. Hadoop uses the first in, first out (FIFO) scheduling algorithm by default. The FIFO scheduling algorithm is a queue-form algorithm, which is relatively simple and cannot meet complex requirements. In our experiment, the Hadoop cluster is built with computers of different configurations, computing power, and workload. Therefore, some tasks might be improperly assigned to nodes with heavy workload. To address this problem, in TWLGA we propose considering the dual constraints of time and workload to enable clusters to meet the requirements of short task running time and more reasonable assignment [10,11]. The TWLGA task scheduling includes chromosome coding [12], node workload [13], fitness function [14], and crossing and variation [15].

3.1 Chromosome Coding

Data processing adopts the indirect encoding of tasks and required resources. First, the number of task slices and the resource number of corresponding task slices are determined, and then the chromosome coding is performed. Second, the chromosomes are decoded, and the task-resource tables are retrieved. Third, the expected time to complete (ETC) matrix can be used to calculate the time in which each resource completes all the tasks assigned to it. Assuming that the number of subtasks assigned to resource i are M_i, the time required for resource i to complete all subtasks is

$$EachResourceTime = \sum_{j=1}^{n} Time(i,j), i \in [1, M], \tag{1}$$

where n represents the number of subtasks of each resource and $Time(i,j)$ represents the time to execute the jth task of resource i. In cloud computing, the tasks are all computed in parallel, so the one with longest running time of the all resources can be thought as the final completion time of the task.

$$JobFinalTime(J) = max\big(EachResouceTime(i)\big) \tag{2}$$

3.2 Node Workload

In Hadoop clusters, the three major categories of computing roles are client computer, master node, and slave nodes. The role of a client computer is to load data to the cluster and submit it to the MapReduce programming model. The master node has the two key functions: storing large amounts of data and

running parallel computations on all the data. The slave nodes make up the vast majority of computers and do all the work of storing data and running computations. Each slave node runs both a data node and task tracker daemon that communicates with and receives instructions from the master node. The workload of each slave node is affected by a number of factors, including CPU usage, memory usage, and network resources. In Hadoop clusters, only four factors are usually considered. The workload of the nodes is defined using the following formula:

$$WorkLoad(i) = W_{cpu} \times \mu_{cpu} + W_{me} \times \mu_{me} + W_{disk} \times \mu_{disk} + W_{nr} \times \mu_{nr}, \quad (3)$$

where W_{cpu}, W_{me}, W_{disk}, and W_{nr} represent the proportion of CPU, memory, disk, and network resources of the nodes in the overall performance, respectively. Further, μ_{cpu}, μ_{me}, μ_{disk}, and μ_{nr} represent the usage of the CPU, memory, disk, and network resources, respectively.

In the experiment, the master computer and slave computers collect and store data from the sensor network. However, some slave nodes have heavy workloads, while others are relatively idle. Therefore, the master computer shares the workload of busy nodes with idle nodes, which not only raises efficiency of each single node but also provides the compatibility support to reduce the possible risk of software and hardware.

3.3 Fitness Function

Assuming that the initial population is P, the number of computational resources is R, and the number of sub-tasks is N, the random coding of the chromosomes is generated by random initialization. That is, a total of P chromosomes, whose length is R, and the number gene is randomly selected in $[1, R]$.

The selection fitness function not only directly determines the quality of the algorithm results but also directly affects the speed or the task running time and appropriateness of node allocation. From this analysis, it is well known that fitness based on GA considering the running time is $JobFinalTime(J)$, so the fitness function can be established as

$$P_{time} = \frac{1}{JobFinalTime(J)} \quad (4)$$

However, in the above fitness function, only the task running time is considered, while the workload of the assigned node itself is not considered. Thus, by considering the workload factor, the fitness function is improved as follows:

$$Optimum(J) = \frac{1}{JobFinalTime(J) \times (1 + WorkLoad(i))}. \quad (5)$$

In this manner, the workload and the task scheduling are all considered, so the possibility of assigning tasks to nodes with heavy workloads is avoided.

3.4 Crossing and Variation

The function of crossover is to generate different individuals through crossover transformation of the genes, which is the basis for the entire algorithm. The mutation operation can maintain and improve the diversity of species to improve the local search ability. The TWLGA improves the probability formula

of the cross variation so that they can achieve adaptive adjustment. The improved probability formula is as follows:

$$P_c = \frac{P_{c1}-P_{c2}}{P_{c1}} \times \frac{f_0-f'}{f_0-\bar{f}}, \tag{6}$$

$$P_m = \frac{P_{m1}-P_{m2}}{P_{m1}} \times \frac{f_0-f'}{f_0-\bar{f}}, \tag{7}$$

where f is the fitness of the mutant individual, f_0 is the maximum fitness of the population, f' is the maximum fitness of the crossover individual, and $\bar{f}$ is the average fitness of the population. After the improvement of the sum calculation formula P_c and P_m, it can complete the task scheduling, reduce the total time of task execution, and considers the workload of the nodes. This greatly improves the calculation efficiency.

4. Experimental Test and Results

Our Hadoop cluster platform was developed to test the data processing performance based on cloud computing with the TWLGA. The experimental setup is shown in Fig. 2. The Hadoop cluster is composed of five computers: one acts as the master computer, and the others act as slave computers. The names of the five computers were changed before Hadoop software installation to master, slave1, slave2, slave3, and slave4 in the catalog /etc/hostname of each computer. Then, the hostname and the IP address were added to the configuration file in the catalog /etc/hosts, so each node computer can be recognized and can access the others.

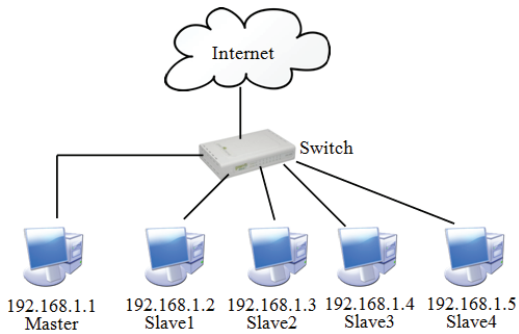

Fig. 2. Network topology of the Hadoop cluster.

The operating system of each node computer is Ubuntu 12.04.3, and the version of Hadoop software is 1.0.2. Hbase was used as the database for the Hadoop cluster, with its worksheets used to store sensor data. The version of Hbase is 0.94.10. The Tomcat web server was used in the cluster and was installed on the master computer, which is deployed by MyEclipse software.

Table 1 shows the format of source data collected from fiber Bragg grating sensors; the data represents year, month, day, observation time (hour and minute), wavelength. The temperature is calculated from wavelength data. There are a large amount of data, and the format is complex. According to the demand,

users only care about the year, month, time, and corresponding temperature, so they only need to extract part of the data. As thousands of data files exist, the advantage of Hadoop in processing large data files can be utilized. Hadoop's processing efficiency of small files is quite low, so it is not advisable to process small files directly. In our experiment, the file merging function of Hadoop was used to merge all files containing data from the same year. Therefore, a single large file was created for each year, and the useful data is extracted by MapReduce, which takes full advantage of Hadoop's ability to handle large files.

Table 1. Source data format

Year	Month	Day	Hour	Minute	Wavelength
2021	03	01	08	30	154574
2021	03	01	08	30	154577
2021	03	01	08	31	154575
2021	03	01	08	31	154579
…	…	…	…	…	…
2021	03	01	11	20	154593
2021	03	01	11	20	154593

4.1 Hadoop I/O Performance Test

As enhancing the response speed and processing power is the main challenges for the data processing system, the performance test of Hadoop I/O is necessary. The reading and writing speed tests of the Hadoop I/O were carried out by hadoop-1.0.2.jar. Meanwhile, the reading and writing speed of the Hadoop Distributed File System (HDFS) were tested through the MapReduce task, so the 10 512M files were read and written to test the HDFS I/O performance. The experimental test results are shown in Table 2.

Table 2. HDFS I/O performance test

Symbol	Read	Write
Number of files	10	10
Total Mbytes processed	5120	5120
Throughput (Mb/s)	17.72	3.49
Average I/O rate (Mb/s)	114.20	4.14
I/O rate standard deviation	120.74	1.92
Test exec time (s)	147.08	313.39

From Table 2, the writing speed of the Hadoop cluster is 4.14 Mb/s. However, the reading speed is 114.20 Mb/s, and the reading speed is about 30 times faster than the writing speed. Hence, the Hadoop cluster was mainly used for the reading operation, especially that it is suitable for one time writing and reading multiple times.

4.2 MapReduce Performance Test

MapReduce is an important programming model for large-scale data parallel and distributed application. Meanwhile, Hadoop is an associated implementation of MapReduce with open source. The

MapReduce performance test files are five txt files, and the sizes are 160 Mb, 320 Mb, 640 Mb, 1.3 Gb, and 2.6 Gb. The five files are placed and counted the one node, two nodes, and three nodes. The experimental test results are shown in Table. 3.

Table 3. MapReduce performance test (unit: second)

Symbol	160 Mb	320 Mb	640 Mb	1.3 Gb	2.6 Gb
1 node	29	64	408	548	1213
2 nodes	38	90	359	487	1054
3 nodes	55	116	352	453	901

From Table 3, the results show that, in the case of small amount of data, the more nodes it has, the slower the calculation speed. As the amount of data increases, the multi-nodes system embodies the superiority. When dealing with small amount of data, the MapReduce must read the data from all of the nodes, so the time spent on network transmission is also crucial. Once large-scale data need to be processed, the role of MapReduce is crucial, which fully demonstrates that MapReduce is suitable for processing the big data.

4.3 HBase Performance Test

To test the HBase performance, the general performance testing tool from Yahoo is used. The writing time, data throughput, and the reading time test are carried out separately when the threads are 1, 50, 100, 1000, and 5000. The test results are shown in Figs. 3–5.

Figs. 3 and 4 show the HBase performance test when writing data, and the writing time and data throughput are changing inversely as the number of threads increases. Fig. 5 shows the HBase performance test when reading data, the reading time decreases exponentially and then flattens out with the increase of threads.

From Figs. 3–5, whether writing or reading data, the time of processing data does not increase in the case of the increasing number of threads, the throughput is within the range of 2,300 to 2,900, and there is no significant change in overall. Thus, the HBase still has fast processing speed under the condition of the high concurrency.

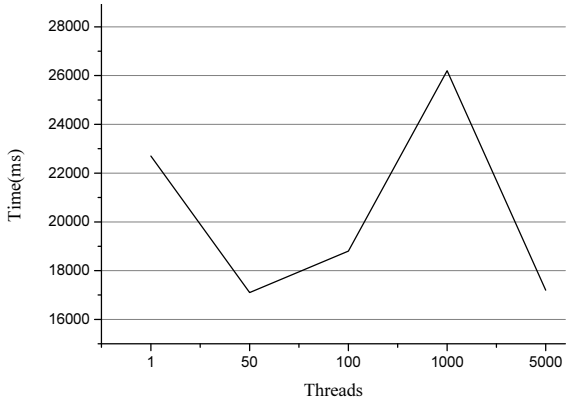

Fig. 3. Writing time change with different threads.

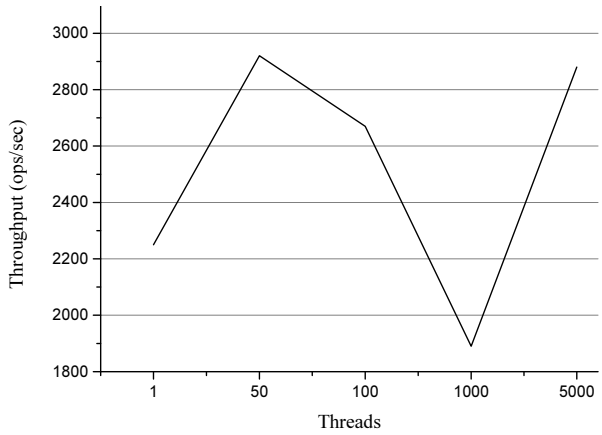

Fig. 4. Throughput change with different threads.

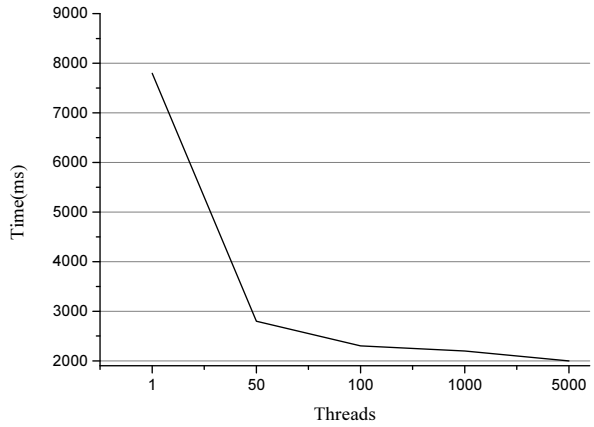

Fig. 5. Reading time change with different threads.

5. Conclusion

In this paper, the data processing of an IOT sensor network based on Hadoop cloud platform and TWLGA scheduling algorithm was proposed. To improve the platform performance, cloud computing technology with TWLGA was adopted to process massive amounts of data and avoid paralysis of the IOT sensor network. The workloads of single busy nodes were shared to idle nodes. Thus, the efficiency of each node was enhanced, and the possible risk of network paralysis was reduced. Finally, a Hadoop cluster platform was built, and the performance of the platform was tested. The results show that the Hadoop cluster platform is suitable for big data processing of IOT sensor networks.

Acknowledgement

This paper is supported by the project of National Natural Science Foundation of China (No. 62175055) and the Research Fund of Handan University (No. 16215).

References

[1] A. C. Sarma and J. Girao, "Identities in the future Internet of Things," *Wireless Personal Communications*, vol. 49, no. 3, pp. 353-363, 2009.

[2] D. Bandyopadhyay and J. Sen, "Internet of things: applications and challenges in technology and standardization," *Wireless Personal Communications*, vol. 58, no. 1, pp. 49-69, 2011.

[3] R. Roman, C. Alcaraz, J. Lopez, and N. Sklavos, "Key management systems for sensor networks in the context of the Internet of Things," *Computers & Electrical Engineering*, vol. 37, no. 2, pp. 147-159, 2011.

[4] L. Wang, G. Von Laszewski, A. Younge, X. He, M. Kunze, J. Tao, and C. Fu, "Cloud computing: a perspective study," *New Generation Computing*, vol. 28, no. 2, pp, 137-146, 2010.

[5] J. Yan, X. Wang, Q. Gan, S. Li, and D. Huang, "Secure and efficient big data deduplication in fog computing," *Soft Computing*, vol. 24, no. 8, pp. 5671-5682, 2020.

[6] P. Liu, *Cloud Computing*. Beijing, China: Electronic Industry Press, 2010.

[7] H. Yedidsion, S. Ashur, A. Banik, P. Carmi, M. J. Katz, and M. Segal, "Sensor network topology design and analysis for efficient data gathering by a mobile mule," *Algorithmica*, vol. 82, no. 10, pp. 2784-2808, 2020.

[8] E. Suganya and C. Rajan, "An AdaBoost-modified classifier using particle swarm optimization and stochastic diffusion search in wireless IoT networks," *Wireless Networks*, vol. 27, no. 4, pp. 2287-2299, 2021.

[9] K. Preethi and R. Tamilarasan, "Monitoring of air pollution to establish optimal less polluted path by utilizing wireless sensor network," *Journal of Ambient Intelligence and Humanized Computing*, vol. 12, no. 6, pp. 6375-6386, 2021.

[10] P. K. Yadav, A. Aggarwal, and M. P. Singh, "Workload analysis in a grid computing environment: a genetic approach," *International Journal of Computer Applications*, vol. 93, no. 16, pp. 26-29, 2014.

[11] H. Aziza and S. Krichen, "A hybrid genetic algorithm for scientific workflow scheduling in cloud environment," *Neural Computing & Applications*, vol. 32, no. 18, pp. 15263-15278, 2020.

[12] A. Tuncer and M. Yildirim, "Chromosome coding methods in genetic algorithm for path planning of mobile robots," in *Computer and Information Sciences II*. London, UK: Springer, 2011, pp. 377-383.

[13] L. Li, M. Guo, L. Ma, H. Mao, and Q. Guan, "Online workload allocation via fog-fog-cloud cooperation to reduce IoT task service delay," *Sensors*, vol. 19, no. 18, article no. 3830, 2019. https://doi.org/10.3390/s19183830

[14] S. A. B. Shah, M. Rashid, and M. Arif, "Estimating WCET using prediction models to compute fitness function of a genetic algorithm," *Real-Time Systems*, vol. 56, pp. 28-63, 2020.

[15] Z. Zhou, F. Li, H. Zhu, H. Xie, J. H. Abawajy, and M. U. Chowdhury, "An improved genetic algorithm using greedy strategy toward task scheduling optimization in cloud environments," *Neural Computing and Applications*, vol. 32, no. 6, pp. 1531-1541, 2020.

Guoyu Li **https://orcid.org/0000-0003-1735-0536**

He received B.S. degree in physics from Hebei Normal University in 2001, M.S. degrees in microelectronics and solid-state electronics from Hebei University of Technology in 2004, and Ph.D. degree in optics from Nankai University in 2007. He is currently a professor in the School of Information Engineering, Handan University, Handan, China. His current research interests include sensor network and IoT.

Kang Yang **https://orcid.org/0000-0002-0452-7868**

She received B.S. degree in electronics science and technology from Hebei University of Technology in 2007, M.S. degree in communication and information systems from Hebei University of Technology in 2010, and Ph.D. degree in optics from the Nankai University in 2019. She is currently an associate professor in the School of Information Engineering, Handan University, Handan, China. Her research interests include communication and communication network.