Computing, Performance and Communication Systems (2024)
Clausius Scientific Press, Canada

DOI: 10.23977/cpcs.2024.080101
ISSN 2371-8889 Vol. 8 Num. 1

Research on Parallel Algorithm Optimization Strategies in High Performance Computing

Yi Zhan, Cennie Wang

Chengdu University, Chengdu, Sichuan, 610106, China

Keywords: High Performance Computing; Parallel algorithms; Optimization strategy

Abstract: In the past decade, with the rapid growth of mobile internet, cloud computing, and big data technology, data has shown explosive growth in different fields. In the era of big data, people have more information to utilize, but the difficulty of obtaining effective information is also greater than before. Therefore, it is necessary to study parallel computing models and performance optimization for big data processing. Exploring the value behind big data using data processing techniques has become a current research focus in the field of data. Given the importance of parallel applications of artificial intelligence (AI) and big data, it is crucial to focus on analyzing the High Performance Computing (HPC) that integrates the two. The complexity and diversity of storage structures, computer architecture, as well as the large volume and complex data of big data processing problems, pose significant challenges for the application of high-performance computers in the field of big data processing. Big data not only provides AI with an increasingly rich set of training data, but also puts higher demands on the computing power of computer systems. Faced with the problems of large scale and complex computation of big data, this paper proposes a multi strategy parallel genetic algorithm (GA) based on machine learning (ML) for optimizing HPC.

1. Introduction

Due to the growth of information technology, people have entered the era of big data [1]. The extraction, processing, and analysis of massive data provide enormous potential application value for social production and life [2]. The arrival of the big data era has increasingly high requirements for the data processing capabilities of information systems, from throughput to response time, from scalability to fault tolerance, from ease of use to high cost-effectiveness. This undoubtedly brings serious challenges to the application of big data processing technology, and also determines that distributed parallel processing will become a necessary means to solve big data processing technology problems in the near future. The new era has become the standard for measuring the era of technology. In the use and application of big data technology, parallel computing models have been introduced, mainly to effectively process and classify large amounts of data, and optimize its performance [3].

While big data brings many conveniences, it also brings huge challenges. With the continuous reduction of data collection costs and the continuous increase in data scale, the share of big data processing in high-performance computing will continue to grow. It is particularly important to use high-performance computers to process complex and diverse large-scale datasets to improve computational efficiency [4]. In order to better utilize these data resources and find new opportunities in the challenges of the big data era, people need to further study parallel computer models and achieve model optimization, striving for the further growth of big data [5]. In large-scale applications, a lot of data needs to be completed within a certain time limit, and the speed and efficiency of data processing determine the value of the data. The vast amount of unknown knowledge and value hidden in big data, compared to smaller data, can bring enormous commercial value to society through data mining analysis, realize various high-value value-added services, and make people's lives more convenient and fast [6].

At present, optimizing and balancing the indicators of parallel computing models is the key to ensuring the successful application of distributed parallel computing in the field of big data. It should be noted that parallel computing models in China are still in their early stages. Therefore, it is necessary to explore parallel computing models and their performance optimization strategies for big data processing, in order to fully explore the value of big data and improve the efficiency of big data processing [7]. In the era of big data, performance optimization must be based on the characteristics and application scope of data to achieve a state of strong operability, in order to truly realize the advantages of big data. The rapid growth of AI technology is a new technological science that studies and develops the simulation, extension, and expansion of human intelligence. ML and data analysis are key technologies for transforming big data into useful knowledge, and research has shown that in many cases, the larger the data processed, the better the performance of ML models. Faced with the problems of large scale and complex computation of big data, this paper proposes a multi strategy parallel GA based on ML for optimizing HPC.

2. HPC for Big Data and AI

2.1. Parallel Processing Architecture for Big data and AI

With the increase in data volume rapidly, the drawbacks of long training time in big data mining and ML have constrained the application of large-scale AI algorithms [8]. Meanwhile, in recent years, HPC or supercomputing has shown an accelerating trend of growth, and parallel algorithms and platforms for big data and AI have quickly become hot topics in international scientific research and industry. The overall framework structure of the HPC platform for big data and AI is shown in Figure 1.

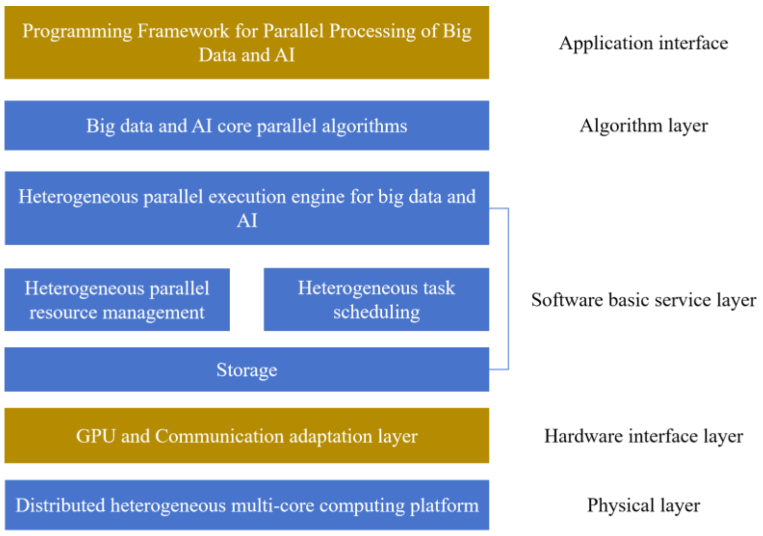
Programming Framework for Parallel Processing of Big Data and AI
Application interface
Big data and AI core parallel algorithms
Algorithm layer
Heterogeneous parallel execution engine for big data and AI
Heterogeneous parallel resource management
Heterogeneous task scheduling
Software basic service layer
Storage
GPU and Communication adaptation layer
Hardware interface layer
Distributed heterogeneous multi-core computing platform
Physical layer

Figure 1: Overall framework structure of HPC platform for big data and AI

Taking heterogeneous computing architecture as its research direction, this architecture meets the requirements of supercomputing and high-performance computing [9]. Due to its advantages of low cost, high performance, and reducing the difficulty of parallel programming, it has been widely used in accelerating algorithms. Heterogeneous multi core parallel processing architecture should consider universality in the system architecture, and work in combination with analysis frameworks and modeling in multimodal algorithms. Based on the characteristics of different ML algorithms, research efficient task mapping methods within and between computing nodes, in order to fully tap into the computing potential of computing devices within and between nodes, and improve computing efficiency. In specific work, it is necessary to grasp the timeliness and node nature of its problems, clarify the direction of integration between AI and big data applications, in order to facilitate the stable operation and functional utilization of parallel computing [10].

2.2. Application of ML in HPC

In ML, reinforcement learning is quite unique, as it continuously optimizes decisions based on feedback obtained through repeated interaction and trial and error with the environment. GA is a global search algorithm that simulates natural biological processes. This algorithm typically abstracts the problem space as a population and generates a new generation of population through genetic operator operations such as selection, crossover, and mutation on individual individuals in the population. After multiple iterations, the final population tends to converge. Due to its high efficiency, simplicity, and robustness in global parallel search, GA has been applied in many fields. When GA is applied to specific problems, parameters often need to be explored multiple times or given based on experience, so parameter settings are closely related to GA's generalization ability. Introducing parallel optimization can significantly accelerate algorithm running speed and reduce running time, especially on multi-core processors where parallel computing can fully utilize computing resources.

3. Multi Strategy Parallel GA Based on ML

3.1. GA and Big Data Cluster Analysis

Using parallel optimization to accelerate the evolution process of algorithms, introducing clustering algorithms to partition the population, ensuring the diversity and uniformity of sub populations. Set the collected network raw data as a $n*m$ form data matrix, which represents n data objects in the dataset, each containing m different attributes. A_{ig} represents the g th attribute of the i th data. This matrix can be represented as:

$$A_{m*n} = \begin{bmatrix} A_{11} \cdots A_{1g} \cdots A_{1m} \\ \cdots \ \cdots\cdots \ \cdots \ \cdots \\ A_{i1} \cdots A_{ig} \cdots A_{im} \\ \cdots \ \cdots\cdots \ \cdots \ \cdots \\ A_{n1} \cdots A_{ng} \cdots A_{nm} \end{bmatrix} \tag{1}$$

Based on this matrix, the similarity in network data can be measured, the attribution relationship between data can be evaluated, and clustering analysis of data can be completed.

The selection operation adopts a combination of roulette wheel strategy and elite retention strategy. The probability of an individual being selected is shown in equation (2).

$$p_i = \frac{f_i}{\sum f_i} \tag{2}$$

The stronger the adaptability of an individual, the greater the chance of survival. To accelerate population evolution and ensure that better individuals are not lost in other processes, an elite retention strategy is used to replace the worst fitness individual in offspring with the one with the highest fitness in the current generation.

Due to the need to differentiate tasks with smaller memory. Therefore, when applying for memory for a Task, use an indicator to distinguish whether it is a large memory requirement type or a small memory requirement type. Define the indicator as Mem_{avg}, as shown in the following equation:

$$Mem_{avg} = \sum_{i=1}^{l} Mem_{task}(i) / \sum_{i=1}^{k} f(i); l \in L, k \in K \tag{3}$$

In the formula, L represents the set of tasks that have not experienced overflow after all runs have ended; K represents the set of memory pool Tasks that have already been called in. Mem_{avg} represents the average memory usage of tasks that have not experienced overflow at the end of all runs, and when a newly called task has a larger memory request than Mem_{avg}, it is considered that the task is required for larger memory. On the contrary, it is considered a small memory requirement.

If the weights of each dimension are different during data processing, it is necessary to weight the attributes of each dimension to obtain the weighted data distance. The specific calculation formula is as follows:

$$d(a,b) = \sqrt{k_1(c_{a1}-c_{b1})^2 + k_2(c_{a2}-c_{b2})^2 + \cdots + k_n(c_{an}-c_{bn})^2} \tag{4}$$

In the above equation, k_1, k_2, k_n represents the weight of each dimension attribute.

3.2. Experimental Analysis

To demonstrate the design value of the algorithm proposed in this article, a simulation experiment is set up and its computational ability is verified. Meanwhile, traditional algorithms are selected as comparative cases. In the experiment, six computers with the same configuration were selected to form an experimental platform in the experimental network. The main control server and router were installed on the platform to achieve the functionality of the experimental platform.

As shown in Figure 2, the comparison of the write time required for memory allocation using the algorithm proposed in this paper and traditional algorithms after writing 1000 256kB data blocks in the storage system is shown. Traditional algorithms complete the sending of data block copies through the central node that sends the data. As redundancy increases, the amount of data written also increases, so the completion time also shows a linear increasing trend. When the algorithm in this article allocates copies through storage and forwarding, the system can evenly distribute the cost of automatically generating copies to the received copy service endpoint. The central node only needs to complete one copy transmission, so the required write time is less, indicating that the algorithm runs fast.

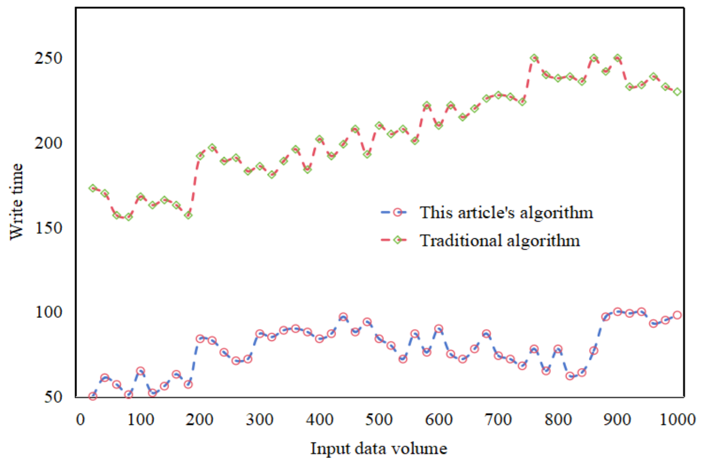

Figure 2: Writing time for different algorithms

Table 1 shows the clustering results of two algorithms on the Iris dataset. Due to the small size of the Iris dataset and the small difference in algorithm runtime, the accuracy of data clustering is mainly compared. Compared to traditional algorithms, our algorithm has increased the number of correct clusters from 137 to 148, with an accuracy improvement of 8.6%.

Table 1: Clustering performance of different algorithms

Algorithm	Number of clusters	Precision/%	Iterations
This article's algorithm	148	96.2	8
Traditional algorithm	137	87.6	5

Compared to traditional algorithms, the data point division of our algorithm is more reasonable, and the clustering results are closest to the actual classification of Iris data. The clustering accuracy is the highest among similar algorithms. In terms of iteration times, the algorithm in this article increases the corresponding number of iterations and computational complexity due to the need to initialize the cluster center; However, under small-scale data conditions, the increase in iteration times is very limited, and the additional parallel computing time can be almost negligible.

In this study, the parallel ability, data search ability, and clustering ability of our algorithm and traditional algorithms were studied through data clustering accuracy and algorithm running time. The experimental results showed that the effectiveness of our algorithm is significantly better than that of traditional algorithms.

4. Conclusions

With the growth of cloud computing and big data technology rapidly, data intensive computing has gradually become the mainstream computing mode. In order to better adapt to the needs of the future era of massive data, it is necessary to conduct in-depth analysis and exploration of computer parallel algorithms on the basis of big data, continuously optimize computer parallel algorithms, better handle related technologies, and optimize their performance indicators, providing important assistance for human life and work. The use of HPC systems, especially supercomputing systems, as computing platforms for big data and AI is gradually becoming a trend. As a key technology for high-performance and supercomputing, parallel algorithms are a necessary way to fully utilize resources and accelerate computing. It should be noted that current parallel algorithms still face growth bottlenecks, which are not conducive to the deep growth and application of big data. In the context of big data, designing more efficient parallel algorithms for specific application scenarios to complete data mining tasks and meet the needs of practical applications is the direction of future efforts. Faced with the problems of large scale and complex computation of big data, this paper proposes a ML based parallel GA for big data to optimize HPC. The experimental results show that the algorithm designed in this paper has good clustering performance and computational efficiency. The future work is to explore the possibility of implementing this algorithm on other big data computing platforms. In subsequent research, the experimental scope will be expanded to verify the effectiveness of this method and provide technical support for future big data processing processes.

References

[1] Li Kenli, Yang Wangdong, Chen Cen, et al. High-performance computing for artificial intelligence and big data [J]. Frontier of data and computing growth, 2020, 2(1):11.

[2] Yan Hua, Wang Yisheng, Wang Ruiqi, et al. GPU-based parallel computing method for reliability of large-scale multi-stage mission system [J]. Systems engineering and electronics, 2019, 41(1):8.

[3] Yang Fan, Gao Guojing, Zhang Yifeng. Research on memory optimization algorithm of parallel computing framework [J]. Information Technology, 2020, 44(8):5.

[4] Lan Fengchong, Li Jiwen, Chen Jiqing. DG-SLAM Algorithm for Complex Deep Learning and Parallel Computing in Dynamic Scenes [J]. Journal of Jilin University: Engineering Edition, 2021, 51(4):10.

[5] Zhang Wenjie, Jiang Liehui. Big data clustering algorithm based on Map Reduce parallel computing [J]. Computer Application Research, 2020(1):4.

[6] Yang Yi, Xiong Ying. Simulation of multi-database parallel scheduling algorithm based on cloud computing platform [J]. Computer Simulation, 2023, 40(6):459-462.

[7] Shi X, Yu X, Esmaeili-Falak M. Improved arithmetic optimization algorithm and its application to carbon fiber reinforced polymer-steel bond strength estimation [J]. Composite Structures, 2023, 306(5):116599.

[8] Néstor Rocchetti, Nesmachnow S, Tancredi G. High-performance computing simulations of self-gravity in astronomical agglomerates:[J]. Simulation, 2023, 99(3):263-289.

[9] Yuan Xuefeng, Zhou Hua, Zhao Qi, et al. Research on NMSFast and Optimization of Pyramid Template Matching Algorithm [J]. Logistics and Fuzziness, 2023, 13(4):3994-4003.

[10] Xu Peiyu, Zhang Zeqiang, Guan Chao. Modeling and hybrid teaching optimization algorithm for the balancing problem of man-machine parallel disassembly line [J]. Computer Integrated Manufacturing System, 2023, 29(7): 2175-2190.