UDC 004

V.O. Denysiuk
A.A. Polishchyk

SOFTWARE IMPLEMENTATION AND RESEARCH OF QUICK SORTING OF DATA ARRAYS BY OPEN MPI

Vinnytsia National Technical University

Анотація

Матеріали присвячені розробці та тестуванню програмної реалізації для сортування масиву елементів на основі паралельного швидкого сортування з використанням Open MPI. Програмна реалізація дозволяє отримати відсортовану послідовність елементів масиву та виміряти час виконання сортування для подальших досліджень.

Ключові слова: *алгоритм, програма, Open MPI, паралельний алгоритм, швидке сортування*

Abstract

The materials are devoted to the development and testing of a software implementation for sorting an array of elements based on parallel quicksort by Open MPI. The software implementation allows you to get a sorted sequence of array elements and measure the sorting execution time for further research.

Keywords: *algorithm, program, Open MPI, parallel algorithm, quick sorting*.

Introduction

The relevance is that the task of sorting arrays is one of the most important, because its purpose is to facilitate further processing of certain data and search tasks. Sorting is an integral part of working with almost any type of information, which ensures its classification and analysis [1].

The time complexity of sorting algorithms and the amount of memory used for sorting significantly affect the efficiency of computer data processing [1]. The subject of the work is a parallel algorithm for fast sorting of data arrays using Open MPI [2].

In contrast to the traditional sequential algorithm, this algorithm can be simultaneously executed on many computing devices, followed by combining the obtained results to obtain the correct overall result. The purpose of the work is to develop a software product that is ready for use.

Mathematical modeling of the parallel quick sorting algorithm

The quick sort algorithm can be implemented both in an array and in a doubly linked list. Quicksort is a comparison-based algorithm and is not stable. The running time of the sorting algorithm depends on the balance that characterizes the partition. Balance, in turn, depends on which element is chosen as a reference (relative to which element the division is performed) [3].

The parallel quicksort algorithm is optimized as follows. Instead of doubling the number of processes at each step, the approach uses n number of processes throughout the algorithm to find the reference element and reorder the list.

All these processes are performed simultaneously at each step of sorting the lists. A parallel algorithm model is developed by considering the data partitioning strategy and the processing method and applying an appropriate strategy to reduce interaction.

Open MPI and the following C++ libraries were used to perform parallel actions: <omp.h>, <iostream>, <chrono>, <time.h>, <random> [4].

Testing the parallel quicksort algorithm

Two array sorting algorithms were tested for n values from 10 to 1,000,000. Each of the array elements varies from 1 to 999 (Table 1).

The parallel execution time is O(logn). The total time complexity is θ(nlogn). The developed algorithm can work on parallel processors and runs much faster for large n.

Table 1
Time to sort an array

Elements in the Array	Quicksort (ms)	Parallel Quicksort (ms)
10	0,000 656	0,000 656
100	0,068 56	0,069265
1 000	0,719 220	0,708635
10 000	81,135 8	7,2634
100 000	285,707 0	28,1468
1 000 000	3 279,630 5	153,3212

Conclusions

The result of the work is a ready-made parallel algorithm for fast sorting of data arrays by Open MPI. Testing confirmed the correctness of theoretical studies. In the future, it is worth investigating the algorithm for sorting multidimensional data arrays.

In such a sorting algorithm, it is necessary to add an algorithm for dividing the array into sublists. For further testing of algorithms for sorting data arrays, it is necessary to create a program with the ability to generate a multidimensional array with random values by size.

REFERENCES

1. С. В. Коляденко, В. О. Денисюк, Н. П. Юрчук. Дискретний аналіз. Частина1. Навчальний посібник. Вінниця: ВНАУ, 2019.
2. Open MPI documentation. URL: https://www.open-mpi.org/
3. Parallel Quick Sort. URL: https://iq.opengenus.org/parallel-quicksort/
4. C++. URL: https://en.wikipedia.org/wiki/C%2B%2B

Денисюк Валерій Олександрович, к.т.н., доцент, доцент кафедри комп'ютерних наук, Вінницький національний технічний університет, Вінниця, Україна, e-mail: vad64@i.ua.

Поліщук Анатолій Андрійович, студент, група 2КН-22м, факультет інтелектуальних інформаційних технологій та автоматизації, Вінницький національний технічний університет, Вінниця, Україна.

Denysiuk Valerii Olexandrovich, PhD, assistant professor of Computer Sciences Department, Vinnytsia National Agrarian University, Vinnytsia, Ukraine, e-mail: vad64@i.ua.

Polishchyk Anatoly Andriyovich, student, Faculty of Intellectual Information Technologies and Automation, Vinnytsia National Agrarian University, Vinnytsia, Ukraine.