UDC 004.058.2

M. B. Shkliaruk

COMPARATIVE ANALYSIS OF THE COMPLEXITY OF TWO ALGORITHMS FOR SOLVING A SINGLE TASK

Vinnytsia National Technical University

Анотація

Проведено порівняння двох однакових за складністю алгоритмів вирішення конкретного завдання. Визначено, що час виконання алгоритму залежить не тільки від його складності.

Ключові слова: алгоритм, складність, складність алгоритму.

Abstract

Two algorithms with the same complexity for solving a particular task have been compared. It has been determined that the execution time of an algorithm depends not only on its complexity.

Key words: algorithm, complexity, the complexity of the algorithm.

Introduction

The complexity of an algorithm is a quantitative characteristic reflecting the resources consumed by the algorithm during its execution [1, 2]. Algorithm complexity is typically assessed based on execution time or memory usage. In both cases, complexity depends on the size of the input data: an array of 100 elements will be processed faster than a similar one with 1000 elements. This is not about the exact computation time, which depends on the processor, data type, programming language, and so on. Complexity is evaluated when striving to extend the size of the input data to infinity.

However, two different algorithms with the same complexity may have different execution times. Therefore, a comparative analysis of the complexity of two algorithms for solving a single task is a relevant task.

Description of Algorithms

The task for which the proposed algorithms were developed involved finding the largest and smallest quadrilaterals by area using specified points as vertices. The program results for 200 random points are shown in Figure 1, and for both algorithms, they are identical.

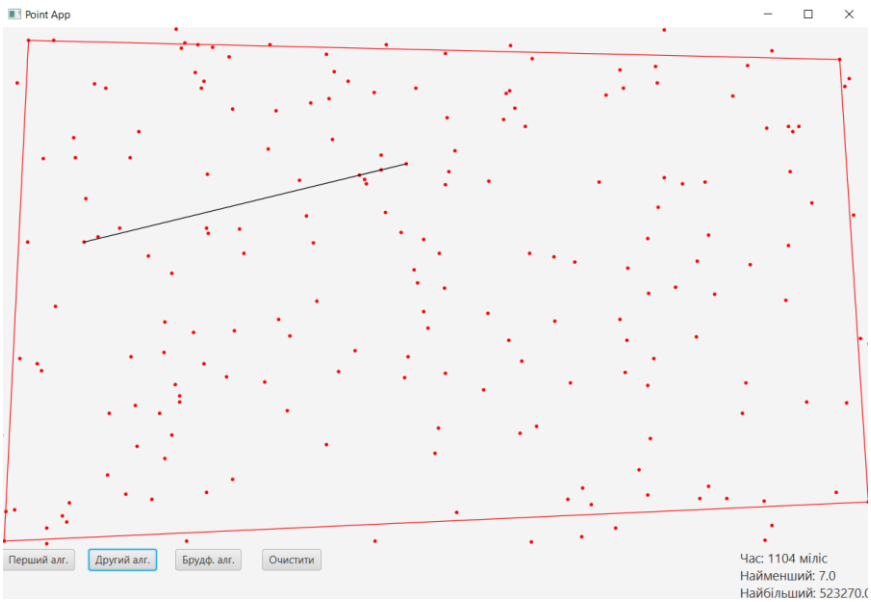

Figure 1 – Largest and smallest quadrilaterals by area

Both algorithms were created using four nested for loops, hence they had a complexity of $O(n^4)$. However, the execution time of each of them for the same number of points differs significantly. Let's consider what influenced the comparison results.

```
for (int i = 0; i < points.size(); i++) {
    for (int j = 0; j < points.size(); j++) {
        if (i!=j) {
            for (int k = 0; k < points.size(); k++) {
                if (i != k && j != k) {
                    for (int n = 0; n < points.size(); n++) {
                        if (i != n && j != n && k != n) {
                            //обчислення
                        }
                    }
                }
            }
        }
    }
}
```

Figure 2 – Code of the first algorithm

The first algorithm, in addition to loops, also employed intelligent iteration with the pre-exclusion of loops for points that were equal to each other (Figure 2). Therefore, the number of iterations through the array of points was somewhat reduced.

The second algorithm used the same iteration principle but in a different way: iterations in each subsequent nested loop started from a point that was one index larger than the current point of the previous loop. However, to achieve this, it was necessary to add a check for the areas of three quadrilaterals created from the same points but in different combinations (Figure 3). This way, combinations such as 1-2-3-4, 2-3-4-1, 3-4-1-2, 4-1-2-3 were excluded.

```
for (int i = 0; i < points.size(); i++) {
    for (int j = i + 1; j < points.size(); j++) {
        for (int k = j + 1; k < points.size(); k++) {
            for (int n = k + 1; n < points.size(); n++) {
                double square1 = getSquare(points.get(i), points.get(j), points.get(k), points.get(n));
                double square2 = getSquare(points.get(i), points.get(j), points.get(n), points.get(k));
                double square3 = getSquare(points.get(i), points.get(k), points.get(j), points.get(n));
                //обчислення
            }
        }
    }

}
```

Figure 3 – Code of the second algorithm

The results of comparing time expenditures depending on the number of specified points are depicted in Table 1.

Table 1 – Comparison of the results of the first and second algorithms

	10 points	50 points	100 points	200 points
Algorithm 1	10 ms	105 ms	1040 ms	13000 ms
Algorithm 2	8 ms	60 ms	300 ms	1400 ms

It is immediately apparent from the table that the first algorithm significantly lags in execution time compared to the second. However, they have the same complexity and differ only in their 'intelligent iterations', which allow filtering out unnecessary iterations and reducing time and memory device costs.

Conclusions

The assessment of algorithm complexity is a crucial stage in algorithm design and analysis. Theoretical evaluations, such as determining time and space complexity in terms of "Big-O notation," provide a theoretical framework for comparing algorithms and predicting their efficiency as input data sizes increase.

However, real-world programs may interact with various factors that are challenging or impossible to account for in theoretical models. These factors may include specific computations within the algorithm, the architecture of a particular computer, properties of specific input data, and other aspects of real-world algorithm usage.

Therefore, for a comprehensive assessment of algorithm complexity, it is essential to conduct experimental analysis, which involves measuring execution time on real input data. The combination of theoretical evaluation and experimental analysis provides a more complete picture of an algorithm's efficiency and suitability for practical tasks.

REFERENCES

1. Wikimedia contributors. "Algorithm – Wikipedia." Electronic resource. Accessed at: https://en.wikipedia.org/wiki/Algorithm.
2. Savchuk V. "Big O: Algorithm Complexity." The Code. Electronic resource. Accessed at: https://www.the-code.com.ua/en/algorithm-complexity/.

Шклярук Марія Богданівна – студентка групи 2ПІ-22б, факультет інформаційних технологій та комп'ютерної інженерії, Вінницький національний технічний університет, Вінниця, e-mail: shkliiaruk.mariia@gmail.com

Науковий керівник: **Мельник Марина Борисівна**, викладач англійської мови, кафедра іноземних мов, Вінницький національний технічний університет. E-mail: melnykmary1@gmail.com

Maria Bogdanivna Shklyaruk - student of group 2PI-22b, Faculty of Information Technologies and Computer Engineering, Vinnytsia National Technical University, Vinnytsia, e-mail: shkliiaruk.mariia@gmail.com

Scientific supervisor Melnyk Maryna Borysivna – teacher of English, Department of the Foreign Languages, Vinnytsia National Technical University. E-mail: melnykmary1@gmail.com