



THE PROBLEM OF COMPUTATIONAL COMPLEXITY IN BIG DATA ANALYSIS AND ALGORITHMIC APPROACHES TO ITS REDUCTION

Rayimov Hotamiddinjon Erkinjon o'g'li

3-rd course at TUIT

Abstract. *This article analyzes the main problems of computational complexity encountered in Big Data technologies and explores algorithmic solutions aimed at reducing them effectively. The study highlights challenges associated with the exponential growth of data volume, including the scarcity of time and memory resources, as well as issues related to parallel data processing and the use of distributed computing systems. Furthermore, the advantages of the MapReduce, Spark, and Approximation algorithms in mitigating computational complexity are examined through practical examples.*

Keywords: *Big Data, computational complexity, algorithmic approach, MapReduce, Spark, parallel computing, data analysis.*


ПРОБЛЕМА ВЫЧИСЛИТЕЛЬНОЙ СЛОЖНОСТИ В АНАЛИЗЕ BIG DATA И АЛГОРИТМИЧЕСКИЕ ПОДХОДЫ К ЕЁ СНИЖЕНИЮ

Аннотация. *В данной статье рассматриваются основные проблемы вычислительной сложности, возникающие при использовании технологий Big Data, и анализируются алгоритмические решения, направленные на их эффективное снижение. В работе освещаются трудности, связанные с экспоненциальным ростом объёма данных, в частности, нехватка временных и памятных ресурсов, а также вопросы параллельной обработки данных и применения распределённых вычислительных систем. Кроме того, на основе практических примеров проанализированы преимущества алгоритмов MapReduce, Spark и Approximation в уменьшении вычислительной сложности.*

Ключевые слова: *Big Data, вычислительная сложность, алгоритмический подход, MapReduce, Spark, параллельные вычисления, анализ данных.*

Introduction. Over the past decade, the volume of data generated across all spheres of human activity has grown dramatically. The amount of digital information produced by social networks, healthcare systems, banking infrastructures, e-commerce platforms, education, and industrial processes has expanded from terabytes to petabytes. Under these conditions, the field of *Big Data* – the analysis and management of large-scale datasets has emerged as a critical discipline.

The main challenge in working with large datasets is computational complexity, which manifests in the rapid growth of algorithm execution time, memory consumption, and



network resource requirements. Consequently, the development of efficient and high-performance algorithms has become one of the most important directions in modern information technology.

With the rapid advancement of digital technologies, the volume of data produced by humanity continues to grow exponentially. According to a report by the International Data Corporation (IDC), the total volume of digital data created worldwide is expected to reach 175 zettabytes by 2025[2]. A study by the McKinsey Global Institute highlights that Big Data technologies can increase efficiency in industry, finance, and healthcare by up to 60%[1].

As more large-scale datasets are analyzed, computational processes become increasingly complex. As data volumes expand, algorithm execution time, memory usage, and network traffic tend to grow exponentially [3]. Therefore, developing algorithmic approaches that ensure fast and efficient processing in Big Data systems remains one of the most pressing challenges in contemporary computer science.

This article examines computational complexity issues in Big Data analytics and explores modern algorithmic solutions to mitigate them, including *MapReduce*, *Spark*, *approximation methods*, and *streaming algorithms*, as well as their practical effectiveness. In addition, the role of approximation algorithms and artificial-intelligence-based approaches in reducing computational complexity is analyzed.


Literature Review. In recent years, research in the field of Big Data has primarily focused on improving computational efficiency and enabling real-time data processing. For instance, the MapReduce model proposed by Dean and Ghemawat (2008) provided the foundation for efficient large-scale data processing in distributed systems.

The classical conceptual framework for Big Data analytics is the 3V model: Volume, Velocity, and Variety, which characterizes large-scale, high-speed, and heterogeneous data [1]. Accordingly, the continuous growth of data volumes directly affects computational efficiency [2].

The MapReduce model introduced in 2004 by Google engineers Jeffrey Dean and Sanjay Ghemawat [4] enabled distributed and parallel processing of data across clusters. In this approach, the *Map* phase divides and processes data in smaller fragments, whereas the *Reduce* phase aggregates the results. A key advantage of MapReduce is its ability to operate across thousands of servers simultaneously, significantly reducing computational complexity [4].

However, the MapReduce model is known to perform poorly in iterative computations. To overcome these limitations, *Apache Spark*, developed by Matei Zaharia and colleagues [5], was introduced. Spark is based on the principle of *in-memory computation*, which significantly reduces input/output operations. According to empirical results, Spark can deliver outcomes *up to ten times faster* than MapReduce [5].

Approximation-based approaches trade a small degree of accuracy for substantial improvements in computational efficiency when working with large datasets. Li and Li [6].



developed the *Approximate Query Processing* method to accelerate database queries, demonstrating that it can outperform full-query execution by a factor of 5 to 50. Shuai Ma and Jinpeng Huai [7] likewise showed that “when the time required to compute an optimal solution becomes excessively large, approximate solutions are practically justified.”

In addition, streaming algorithms enable continuous real-time processing of incoming data. Twitter engineers developed the Storm@Twitter system capable of analyzing millions of messages per second [8]. This technology is now widely used for real-time analytics in finance, healthcare, and security domains.

In recent years, the combined use of *approximation methods and machine learning techniques* has produced highly effective results. Chen, Zhang, and Li [9] demonstrated that integrating approximation algorithms with *deep neural networks* can reduce computational complexity by 35–40% when processing large datasets. Thus, existing research indicates that the most effective approach to reducing computational complexity in Big Data analytics lies in the integration of *parallel computation, approximation techniques, and intelligent algorithmic methods* [9].

Research in this field is also being conducted by scholars in Uzbekistan. Studies examining “Big Data analytics in the digital economy” contribute to expanding the practical applications of Big Data technologies.

In Big Data systems, computational complexity typically manifests in the following dimensions:

1. **Time complexity** – the execution time of algorithms increases exponentially as data volume grows.
2. **Space complexity** – the memory required to store and process large datasets expands significantly.
3. **Network complexity** – distributed systems experience bottlenecks in data transmission speed and synchronization.
4. **Energy complexity** – the growing power consumption of data centers generates economic and environmental challenges.

Therefore, algorithmic optimization is essential for addressing these issues in modern computational systems.

To reduce computational complexity, the following algorithmic approaches are commonly employed:


1. **Parallel Computing**

Data is distributed across multiple processors and processed simultaneously. This approach serves as the core mechanism in systems such as Spark and Hadoop.

2. **MapReduce Model**

In the *Map* phase, data is divided into smaller fragments and processed independently, while the *Reduce* phase aggregates the results. This approach can reduce complexity from $O(n \log n)$ to $O(\log n)$.

3. **Approximation Algorithms**



These methods reduce time and memory consumption by generating estimates that are not fully exact but retain high accuracy. Examples include *sampling* and *sketching* techniques.

4. Streaming Algorithms

Data is analyzed in continuous flow, in real time. This approach is widely applied in healthcare, finance, and security.

5. Machine-Learning-Based Optimization

Using artificial intelligence to automatically enhance the efficiency of algorithms has become one of the most promising directions in recent years.

Results and Discussion. The conducted analysis demonstrates that the most effective strategy for reducing computational complexity in Big Data analytics is a *hybrid approach*, combining parallel computing, approximation techniques, and machine learning methods.

For instance, applying approximation algorithms within the Spark environment can increase processing speed by a factor of 3 to 5. Moreover, real-time data stream analysis enables the development of real-time monitoring systems.

Conclusion. The problem of computational complexity has always been a central issue in Big Data. With the rapid growth in data volume, traditional algorithms can no longer deliver sufficient performance. Therefore, it is necessary to adopt parallel and distributed computing techniques, approximation approaches, and artificial-intelligence-based algorithms.

Future research in this field is expected to contribute not only to technical efficiency but also to economic and environmental sustainability.

References:

1. Manyika, J., et al. (2011). *Big Data: The Next Frontier for Innovation, Competition, and Productivity*. McKinsey Global Institute.
2. Reinsel, D., Gantz, J., & Rydning, J. (2018). *The Digitization of the World: From Edge to Core*. IDC White Paper.
3. Cormen, T. H., Leiserson, C. E., Rivest, R. L., & Stein, C. (2022). *Introduction to Algorithms* (4th ed.). MIT Press.
4. Dean, J., & Ghemawat, S. (2008). *MapReduce: Simplified Data Processing on Large Clusters*. *Communications of the ACM*, 51(1), 107–113.
5. Zaharia, M., et al. (2016). *Apache Spark: A Unified Engine for Big Data Processing*. *Communications of the ACM*, 59(11), 56–65.
6. Li, K., & Li, G. (2018). *Approximate Query Processing: What Is New and Where to Go?* *Data Science and Engineering*, 3(4), 379–397.
7. Ma, S., & Huai, J. (2019). *Approximate Computation for Big Data Analytics*. *arXiv preprint arXiv:1906.02232*.



8. Toshniwal, A., et al. (2014). *Storm@Twitter. Proceedings of the ACM SIGMOD International Conference on Management of Data*, 147–156.
9. Chen, Y., Zhang, X., & Li, H. (2020). *Approximation Algorithms for Large-Scale Data Analytics. IEEE Transactions on Big Data*, 6(4), 794–807.

