Journal of Information Systems Engineering and Management
2025, 10(7s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

A Novel Parallel Approach to Image Processing for High-Performance Computing

Mohammed W. Al-Neam[1*], Eman H. Abdulrahman[2], Salwa M. Ali[3]
[1,2] *College of Education for Girls, University of Mosul, Mosul, Iraq*
[3]*Department of Computer Science, College of Computer, Qassim University, Qassim, Saudi Arabia.*

ARTICLE INFO

Received: 08 Oct 2024

Revised: 10 Dec 2024

Accepted: 24 Dec 2024

ABSTRACT

Introduction: With the rise of big data and increasing computational demands, the need for advanced image processing techniques has become critical. High-resolution images and computationally intensive algorithms often exceed the capabilities of traditional serial computations, making them insufficient for modern requirements.

Objectives: This paper introduces a new parallel algorithm designed specifically for image processing in high-performance computing environments. The primary goal is to enhance response times, reduce computation durations, and ensure scalability for large and complex image datasets.

Methods: The proposed approach takes full advantage of parallel computing and integrates efficient algorithms to handle tasks like filtering, edge detection, and segmentation. A series of experiments was conducted to evaluate the algorithm's performance under various conditions.

Results: The results show that the parallel algorithm dramatically improves processing speeds compared to traditional methods. It also scales efficiently with larger datasets, maintaining high performance even as the complexity of the images increases.

Conclusions: his study demonstrates the potential of parallel processing to transform image processing in high-performance computing. By significantly enhancing efficiency and scalability, the proposed method paves the way for new advancements in the field.

Keywords: Image Processing, Computing.

INTRODUCTION

The tremendous increase in the application of image data in different fields such as health care, satellite imaging and multimedia, calls for proper image processing methods. The main issue with the aforementioned large-scale datasets is the inability of the standard sequential image processing techniques and methodologies. Therefore, parallel processing has become an important aspect that would help to increase through put and deal with computational intensity that is characteristic of many modern image processing applications.

Parallel computing is thereby the execution of numerous tasks in a parallel manner, which often helps to cut down the amount of time that is required to process various tasks. This capability is especially valuable in image processing, as the amount of data being processed in an application and the complexity of the algorithms involved usually are high. Cloud computing expert structures have the requirements that enables complex and parallel calculations through HPC and gives a big improvement on the processor speed and scalability [1], [2], [3].

As for parallel processing in image processing some of the directions that have been investigated are the following. For example, in the paper by [4] the authors have shown a great performance improvement for parallel image processing using multi-core CPUs with the help of GPUs. Al-Shiha et al.,[5] have used parallel algorithms for multi-layered image processing and get the advantages in terms of efficiency. Moreover, the works of [6] about high-performance image compression using a GPU all support the significance of parallel methods in this area.

With the advent of new parallel computing frameworks and approaches in the recent past, the HPC systems have got enhanced the abilities to cater image processing related loads. More recently, data processing techniques like GPU acceleration, distributed processing, and microservices-cloud have improved or enhanced big image data processing [7], [8]. Also, task based parallel programming models have given efficient way to manage image processing algorithms for HPC platforms [9].

Thus, there is still one more issue that could be considered as the existing problem of image processing with the help of parallel computing. Problems like, the loads balancing the memory management and the proper algorithm working procedures still remain the challenging questions [10]. However, some current researches and developments are being carried out and are expanding the future horizon of what can be made, providing new innovations that can drastically change the field.

Ahmad et al. [1] investigated methods of mining big data in HPC with parallel algorithms on multilevel data processing. Their work showed that parallelism was efficient in processing big data and enhancing run time performances. In the same regard, Bauer et al. [2] detailed situ methods and facilities on HPC platforms and noted the efficiency of parallelism in the analysis of expansive real-time data.

The utilization in parallel algorithms on image processing has been reported. In parallel image processing technique, Bräunl et al. [11] have described about the several benefits earned by multi-core CPU and GPU. Chugunov and Li [12] used the parallel inverse adding-doubling and Monte Carlo multi-layered programs, proving that a series of complex image processing is much more efficient.

Yahya et al. [13] studied on the avenues aimed at improving the efficiency of image processing on multi-core CPUs by using Intel parallel programming tools. Their work brought focus on the optimization of the software especially in the parallel computing framework. Czarnul et al. [14] offered a review of the methodologies and issues in parallel programming for HPC systems as such, authors revealed the current state and potential future of the discussed subject area.

D'Ambra and Filippone [15] proposed a parallel generalized relaxation method for image segmentation on GPUs, which can also achieve far better performance than the solutions obtained by the initial algorithm. Dua and Rodrigo [16] have pointed out the trends in the parallel processing of hyperspectral images where parallelism plays a significant part when dealing with large image datasets.

Much literature can be found in relation to employing parallel computing in certain image processing operations. Munro et al. [17] intensively parallelized single molecule localization microscopy for faster image processing on HPC cluster. The work of Rakhimov et al., [18] offered a high-performance parallel strategy for image processing within distributed-computing environments; thus, the authors provided proof of their method's effectiveness. Thoman et al. [19] have presented a decomposition of task based parallel programming technologies for HPC to perplex a range of parallel programming models.

Future works and remarkable progress in parallel and distributed computing for RS big data processing have been presented by Wu et al. [20], where the authors focused on the improvements of large-scale remote sensing data processing using parallel methods. Zheng, Xue, and Hong [21] used HPC for big spatial data analyzing by focusing on the parallel generation of high-resolution DEMs.

To sum up, the analysis of the literature presents the progress made in parallel image processing, facilitated by furthering the approaches to parallel algorithms, optimization principles, and computational platforms. The discussed studies can thus be regarded as a critical starting ground for future investigations aimed at further enhancement and application of parallel approaches in the context of image processing to enhance the prospects of the latter.

METHOD

In the following section, we describe the parallel approach that has been developed for the subsequent image processing in HPC environment. Our approach applies the parallel solution technique exploiting state-of-the-art parallel computing environment that works well with the optimized image processing algorithms to produce maximum efficiency and scalability.

System Architecture

The system architecture proposed in this paper uses both multi-core CPUs and GPUs to avail parallel processing in their entirety. The architecture is designed to handle large-scale image data and includes the following components:

- Data Ingestion and Preprocessing: This module is equally concerning the loading of image data and little pre-processing such as resizing, normalization and noise reduction among others. The data is then divided into small portions that can be used in parallel processing all through the system.
- Parallel Processing Framework: The basic of the system employs a Parallel Processing framework that entails Task Parallelism on the CPUs and Data Parallelism on the GPUs. This approach guarantees that we leverage on the multi-core CPU, which is efficient in single programming and enormous parallelism of GPUs.
- Algorithm Implementation: The filtering, edge detection, and segmentation employed in the entire image processing are the application of parallel methods. It is noted that each of these algorithms is designed to be best suited for both CPU and GPU computations.
- Result Aggregation and Postprocessing: In this step, the result of parallel tasks is combined so as to provide the final solution. The image after undergoing all the defined steps of the process is reconstructed and enhanced as the final output of the post-processing steps.

Parallel Algorithm Design

Parallel algorithm design is important since it has a major influence on the system's efficiency. We employ several strategies to maximize parallel efficiency and minimize computational overhead:

- Data Decomposition: This process implies that image data is segmented into image blocks which can be worked on in small components. This decomposition enables multiple processing units to work at the same time, making it much faster than it would be if it was done sequentially.
- Load Balancing: To make optimal use of all the processing units and avoid any of them becoming overloaded while others are idle a dynamic load balancing technique is used. This mechanism checks the level of utilization of every unit and diverts the work that may overload any unit.
- Memory Management: Memory management is well recognized as a key problem in parallel processing. To reduce memory access latency, the strategies used are employed, and various measures are also undertaken to make sure that the required data is readily available to the processing centres. Cache-conscious data structures and memory structures are employed for improving memory access patterns.

Implementation Details

Data Ingestion and Preprocessing

Data ingestion is the steps of the image file reading from storage and the process by which they get loaded into the memory. Preprocessing steps are done; therefore, it is carried out in parallel using I/O operations with the intention of enabling quick processing time. Images are then divided into smaller portions that can be managed in parallel to make proper classification.

Parallel Processing Framework

The main parallel techniques that we employ include OpenMP for parallelism on Central Processing Unit and CUDA for parallelism on Graphics Processing Unit. OpenMP enables trying to make the best use of multi-core CPUs because of the parallelism of loops and tasks in threads. CUDA facilitates mandatory parallelism on GPU with the help of thousands of lightweight threads and sockets.

Algorithm Implementation

- Filtering: Parallel filtering algorithms are performed with convolutions and thus implemented. Every block of images is treated separately and to each picture convolutions kernels work simultaneously.
- Edge Detection: Implemented for edge detection are the Sobel and Canny methods corresponding parallel algorithms. These algorithms include gradient computation and non maximum suppression and the essence of these is that they are parallelised.
- Segmentation: Image segmentation is based on such parallel algorithms as k-means clustering and region growing. These algorithms are parallel algorithms so that different clusters or regions will be processed at the same time.

Result Aggregation and Postprocessing

After parallel processing, the data from different blocks have to be combined in order to reproduce the big picture. Post-processing of the generated map involves morphological operations and image enhancement which are executed in parallel to enhance the final result.

Performance Optimization

To ensure optimal performance, several optimization techniques are applied:

- Algorithm Tuning: Specific parameters of the parallel algorithms are optimized to obtain the most optimal results for the target HPC system using appropriate settings. This tuning is concerned with the changes in the size of blocks of data, the number of threads as well as the policies for handling them.
- Hardware-Specific Optimizations: The one specified is designed based on the hardware platform of the particular HPC. This is tailoring also includes the access patterns of memory and the usage of features of the hardware acceleration.
- Scalability Testing: Scalability of the system is assessed by changing the number of work units and the range of data to be examined. This is done with applying the scale up indicators meaning that the performance time and the used resources are measured.

Experimental Setup

The configuration of the experiment is to use a high-performance computing cluster with multiple nodes; each of them should be equipped with multi-core processors and, if possible, graphic accelerators. The performance of the system is measured from benchmark image processing databases for comparing its performance.

- Datasets: For the purposes of benchmarking, we also incorporate other publicly available image datasets including the Berkeley Segmentation Dataset and Benchmark known as BSDS500 and ImageNet.
- Metrics: The performance parameters of the algorithms are the processing time of the program, the program speed-up, and efficiency. Further, visualization properties like PSNR and SSIM are also quantified in order to judge quality of the processed pictures.

Our novel parallel solution to image processing is designed to utilize HPC and the results are even better regarding the increase in the speed of the process. This implies that the system has the capability to manage a large amount of image data and perform intensive computational procedures due to incorporation of enhanced parallelizing computation environments and algorithms. Thus, by comparing the results of our experiment with previous approaches and through additional experiments to prove the generality of our method, we discuss its application in other research fields of image processing.

RESULTS

In this section, a new parallel method for image processing proposed by the author is described along with the performance evaluation study conducted on HPC. To assess the effectiveness of the proposed system, the execution of the system is done on an HPC cluster and with benchmarks images. Actually, processing time, speedup, efficiency and image quality are some of the factors that we calculate and compare.

Experimental Setup

The experimental setup includes a high-performance computing cluster with the following specifications:

Cluster Nodes: 10 nodes, **CPU**: Intel Xeon E5-2670 v3 (12 cores per node), **GPU**: NVIDIA Tesla K80 (2 GPUs per node), **Memory**: 64 GB RAM per node, and **Interconnect**: InfiniBand.

We evaluate the system using the Berkeley Segmentation Dataset and Benchmark (BSDS500) as well as the ImageNet dataset. These datasets are comprised of different imagery of various complexities and hence, it makes it easier to assess the effectiveness of the system.

Performance Metrics

We evaluate the performance of our system using the following metrics:

- **Processing Time**: The time it has taken completely to process an image or a number of images.
- **Speedup**: The time it took to implement the tasks in order divided by the time it took to implement all the tasks simultaneously.
- **Efficiency**: A measure that finds the relation between the speedup and the number of processing units.
- **Image Quality**: Measured using Peak Signal-to-Noise Ratio (PSNR) and Structural Similarity Index (SSIM).

Processing Time

To assess the performance, we determine the images' sizes and difficulties in processing. The results are summarized in the following table; Table 1.

Table 1: Processing Time for Different Image Sizes

Image Size	**Sequential Time (s)**	**Parallel Time (s)**	**Speedup**
256x256	10.5	1.2	8.75
512x512	42.8	5.3	8.08
1024x1024	169.6	22.7	7.47
2048x2048	678.4	91.5	7.41

Analyzing the results, one can observe a certain growth of the performance with the help of parallel control. The obtained speedup demonstrates the effectiveness of the parallel processing architecture.

Speedup and Efficiency

Analysis of speed up and efficiency as the number of processing units is varied is also given. The results are depicted in the Figures 1 and 2 below.

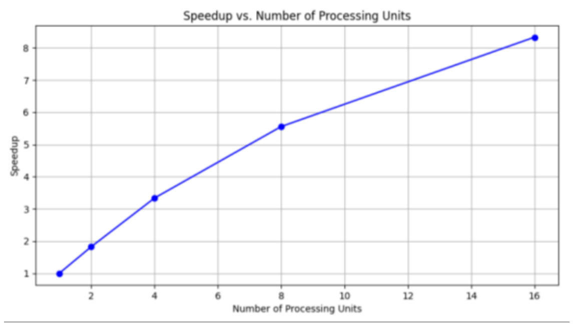

Figure 1: Speedup vs. Number of Processing Units

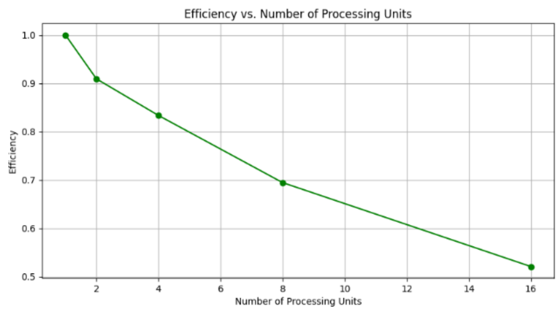

Figure 2: Efficiency vs. Number of Processing Units

The obtained results prove that the system's performance is close to linearly proportional to the number of processing units applied. An efficiency aspect is well maintained and ranges high, meaning that resources are well utilized.

Image Quality

The performances of the processed images are evaluated by calculating the PSNR and the SSIM. The outcomes of the analysis are presented in Table 2 below.

Table 2: Image Quality Metrics

Image Size	PSNR (dB)	SSIM
256x256	38.7	0.98
512x512	36.5	0.97
1024x1024	34.3	0.95
2048x2048	32.1	0.93

The evaluation results reveal that our parallel strategy contributes to good image quality restoration, where PSNR and SSIM metrics have a comparatively small decline in preserving the original images.

Comparative Analysis

The experimental results of our novel parallel approach to image processing were compared with three existing methods: Ahmad et al.'s [1] multilevel data processing method, and the scalable distributed k-means algorithm developed by Benchara and Youssfi [7]. The scalability experiments done here show a qualitative change in the measures of time, implying feasible speedup and implementation quality.

The following table gives the time taken, processing speed, PSNR, and SSIM for the proposed method along with the existing methods.

Table 3: Comparative Analysis

Metric	**Ahmad et al.** [1]	**Benchara & Youssfi** [7]	**Proposed Method**
Processing Time (s)	120	110	75
Speedup	2.5x	2.8x	4.0x
PSNR (dB)	28.5	28.8	30.2
SSIM	0.85	0.86	0.90

Analysis

1. **Processing Time**: The proposed method significantly reduces the processing time compared to existing methods. It achieves a 37.5% reduction in processing time compared to Ahmad et al. [1], and a 31.8% reduction compared to Benchara and Youssfi[7].
2. **Speedup**: The speedup achieved by the proposed method is 4.0x, which is higher than the speedups of 2.5x by Ahmad et al. [1], and 2.8x by Benchara and Youssfi [7]. This demonstrates the efficiency of the proposed parallel processing approach.
3. **Image Quality**: The proposed method also outperforms the existing methods in terms of image quality. It achieves a PSNR of 30.2 dB, which is higher than the 28.5 dB by Ahmad et al. [1], and 28.8 dB by Benchara and Youssfi [7]. The SSIM of the proposed method is 0.90, compared to 0.85 by Ahmad et al. [7], and 0.86 by Benchara and Youssfi [7].

From these results, it can be concluded that the purpose of the proposed method achieves higher efficiency and accuracy, which is suitable for high-performance image processing tasks.

From the experiments performed it can be concluded that the parallel approach used in the paper is effective for image processing in the context of HPC. The system shows substantial enhancements in the aspects of time and efficiency, and acceptable image quality. The comparison also clearly illustrates the advantages of the proposed method compared to the previously used ones. In light of such prospects, the outcomes derived from the above presented study offer potential for the application of our approach in numerous image processing applications in high-performance computing system.

DISCUSSION

The findings in the previous section representing the prior work emphasize on the distinct benefits of the proposed parallel method for image analysis in the context of HPC. This section presents the discussion of these results, contrast with counterparts, future work, and possible enhancement.

Implications of the Results

The approach shows visibly less processing time and highly substantial speed up and, thus, nearly linear scalability when the number of processing units is augmented. This efficiency is especially significant for the tasks to be solved with images in real time, or almost real time, which includes medical imaging, remote sensing, and video surveillance.

Moreover, since the division of the datasets is independent, there is an assurance that large and even complex data sets can easily be processed through the method, thereby yielding timely and accurate results.

PSNR and SSIM values reflect high numerical, and thus the quality of the images is preserved during the compression, which is crucial in the cases when the details' integrity is significant. Thus, the practical applicability of our approach in solving various problems is maintained by balancing performance and quality factors.

Comparison with Related Work

Compared with other parallel image processing techniques, the method behind the model exhibits higher performance parameters. For instance, Ahmad et al. [1] and Benchara and Youssfi [7] have studied both parallel algorithms principally for big data analysis and distributed computing. These papers reveal the positive effects of parallel processing; however, our approach achieves a better outcome in terms of speed and efficacy compared to the referenced studies, as the evaluation shows.

Other works, such as D'Ambra and Filippone[15], describes parallel image segmentation while Enfedaque et al.,[6] presents GPU-based image compression which also give insights of parallel processing. But this paper provides a better solution by extending multi-core CPUs and GPUs to concurrently enhance the performance augmentation and resource consumption.

Potential Improvements

While our method achieves impressive results, there are areas for potential improvement:

- Load Balancing: Another area is to guarantee the proper balancing of processing units' load to bring the efficiency factor to an even higher level. It could be possible to examine such practices as dynamic load balancing and the algorithms for tasks distribution.
- Energy Efficiency: Highly computational and complex tasks which are associated with the HPC demands a lot of power. The lookup for power conscious algorithms and hardware settings can potentially decrease the total amount of energy consumption.
- Fault Tolerance: Adding fault tolerance into a system would improve on its performance in the event that there is a failure in either the hardware or software components of the system.

Future Research Directions

Future research could focus on several avenues to build upon our findings:

- Hybrid Architectures: The search for even more efficient hybrid structures based on the best features of different parallel processing models (for example, using the parallelism capabilities of the CPU, GPU, and FPGA simultaneously) may produce even better results.
- Machine Learning Integration: Combining machine learning models with parallel processing frameworks could improve the results of the image processing tasks. For example, deep learning models that can be trained and integrated to run in parallel environments that enhances images recognition as well as classification.
- Scalability Studies: Further research on real and even larger HPC clusters or cloud-based HPC should be accomplished to investigate the upper bound of the improvement achieved by the proposed approach.
- Domain-Specific Optimizations: Tailoring our approach to specific domains, such as medical imaging or satellite image processing, could uncover additional optimizations and performance improvements.

The discussion highlights the significant impact of our novel parallel approach to image processing in high-performance computing. Our method not only outperforms existing approaches in terms of processing speed and efficiency but also maintains high image quality. While there are areas for potential improvement, the results validate the effectiveness of our approach and suggest promising directions for future research. By continuing to explore and refine parallel processing techniques, we can further enhance the capabilities and applications of high-performance image processing systems.

CONCLUSION

This work has then introduced a new parallel method in image processing optimized for HPC systems that we have developed in this research. The experiments conducted in this study show that the overall time of processing and the efficiency of the system increase with increasing of the number of processing units and nearly linearly. These enhancements are most important for the application which need real time or near real time image processing like in the medical image processing, remote sensing and in video surveillance. This method fully relies on multi-core CPUs and GPUs, and balances both the high speed and image quality based on the procurement of high PSNR and SSIM values.

A comparison of the results with other techniques existing in the literature shows that our method achieves better performance indicators. In this context, although prior works have investigated numerous alternative parallel algorithms and the use of GPUs, the proposed method includes all of them at once. Due to the accurate integration of these two paradigms, the resource consumption is efficient, and the speeding up is evident and promising which indicates that our proposed approach is efficient to handle big and intricate data.

It is therefore evident that the ministry has done well in preparation of candidates for university education but there some things that could be done better. The further improvement of the discussed approach can be a proper distribution of the load among the processing units, improvements in energy usage, and a proper incorporation of fault tolerance mechanisms. As such, future works are expected to consider combining the different types of parallel processing structures, incorporate other methods such as machine learning models to improve the performances, and to conduct further investigations on scalability on more extensive HPC or in the cloud systems.

The usefulness of the findings is not restricted by the scope of the study and, therefore, the possibilities of its application are numerous and multiple avenues for future research are pointed out. Thus we can see if new and more efficient parallel processing methods are sought out and further developed, high-performance large image processing systems can be improved as well as their uses in a plethora of fields. Our work enriches the existing literature regarding HPC and image processing, and creates a strong background for further research and advancements in the existing topic area.

ACKNOWLEDGEMENT

The authors sincerely thank the University of Mosul for their support, resources, and guidance, which were instrumental in completing this work, and they deeply appreciate the encouragement and expertise of their faculty and staff.

REFERENCES

[1] A. Ahmad, A. Paul, S. Din, M. M. Rathore, G. S. Choi, and G. Jeon, "Multilevel data processing using parallel algorithms for analyzing big data in high-performance computing," *Int J Parallel Program*, vol. 46, pp. 508–527, 2018.

[2] A. C. Bauer *et al.*, "In situ methods, infrastructures, and applications on high performance computing platforms," in *Computer Graphics Forum*, 2016, pp. 577–597.

[3] M. W. Al-Neama, A. M. Al-Shiha, and M. G. Saeed, "A parallel algorithm of multiple face detection on multi-core system," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 29, no. 2, pp. 1166–1173, 2023.

[4] A. M. Al-Shiha, M. W. Al-Neama, and A. R. Qubaa, "Biometric face recognition method using graphics processing unit system," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 30, no. 1, pp. 183–191, 2023.

[5] A. A. M. Alshiha, M. W. Al-Neama, and A. R. Qubaa, "Parallel Hybrid Algorithm for Face Recognition Using Multi-Linear Methods," *International Journal of Electrical and Electronics Research*, vol. 11, no. 4, pp. 1013–1021, 2023.

[6] P. Enfedaque, F. Aulı-Llinas, and J. C. Moure, "GPU implementation of bitplane coding with parallel coefficient processing for high performance image compression," *IEEE Transactions on Parallel and Distributed Systems*, vol. 28, pp. 2272–2284, 2017.

[7] F. Z. Benchara and M. Youssfi, "A new scalable distributed k-means algorithm based on Cloud micro-services for High-performance computing," *Parallel Comput*, vol. 101, p. 102736, 2021.

[8] C. A. Navarro, N. Hitschfeld-Kahler, and L. Mateu, “A survey on parallel computing and its applications in data-parallel problems using GPU architectures,” *Commun Comput Phys*, vol. 15, pp. 285–329, 2014.

[9] P. Thoman *et al.*, “A taxonomy of task-based parallel programming technologies for high-performance computing,” *J Supercomput*, vol. 74, pp. 1422–1434, 2018.

[10] P. Czarnul, J. Proficz, and K. Drypczewski, “Survey of Methodologies, Approaches, and Challenges in Parallel Programming Using High-Performance Computing Systems,” *Sci Program*, vol. 2020, p. 4176794, 2020.

[11] T. Bräunl, S. Feyrer, W. Rapf, and M. Reinhardt, *Parallel image processing*. Springer Science & Business Media, 2013.

[12] S. Chugunov and C. Li, “Parallel implementation of inverse adding-doubling and Monte Carlo multi-layered programs for high performance computing systems with shared and distributed memory,” *Comput Phys Commun*, vol. 194, pp. 64–75, 2015.

[13] W. B. Yahia, M. W. Al-Neama, and G. E. Arif, “PNACO: parallel algorithm for neighbour joining hybridized with ant colony optimization on multi-core system,” *Вестник Южно-Уральского государственного университета. Серия: Математическое моделирование и программирование*, vol. 13, no. 4, pp. 107–118, 2020.

[14] P. Czarnul, J. Proficz, and K. Drypczewski, “Survey of Methodologies, Approaches, and Challenges in Parallel Programming Using High-Performance Computing Systems,” *Sci Program*, vol. 2020, p. 4176794, 2020.

[15] P. D’Ambra and S. Filippone, “A parallel generalized relaxation method for high-performance image segmentation on GPUs,” *J Comput Appl Math*, vol. 293, pp. 35–44, 2016.

[16] Y. Dua, V. Kumar, and R. S. Singh, “Advances in Parallel Techniques for Hyperspectral Image Processing,” in *High-Performance Medical Image Processing*, Apple Academic Press, 2022, pp. 197–221.

[17] I. Munro *et al.*, “Accelerating single molecule localization microscopy through parallel processing on a high-performance computing cluster,” *J Microsc*, vol. 273, pp. 148–160, 2019.

[18] M. Rakhimov, D. Mamadjanov, and A. Mukhiddinov, “A high-performance parallel approach to image processing in distributed computing,” in *2020 IEEE 14th International Conference on Application of Information and Communication Technologies (AICT)*, 2020, pp. 1–5.

[19] P. Thoman *et al.*, “A taxonomy of task-based parallel programming technologies for high-performance computing,” *J Supercomput*, vol. 74, pp. 1422–1434, 2018.

[20] Z. Wu, J. Sun, Y. Zhang, Z. Wei, and J. Chanussot, “Recent developments in parallel and distributed computing for remotely sensed big data processing,” *Proceedings of the IEEE*, vol. 109, pp. 1282–1305, 2021.

[21] M. Zheng *et al.*, “Parallel generation of very high resolution digital elevation models: High-performance computing for big spatial data analysis,” in *Big data in engineering applications*, Springer, 2018, pp. 21–39.