Proceedings of the 8th Faculty of Science International Conference (FOSIC 2025), Delta State University, Abraka, Nigeria. 12th – 14th November, 2025. Pp. 232 - 242

MULTI-OBJECTIVE COOPERATIVE PARTICLE SWARM OPTIMIZATION RESOURCE SCHEMING TECHNIQUE IN VEHICULAR CLOUD INFRASTRUCTURE AS A SERVICE PLATFORM

Egheneji A. W.[1] and Edje E. A.[2]

[1]Computer Science Department, Delta State University, Abraka, Nigeria

[2]Computer Science Department, Delta State University, Abraka, Nigeria

ABSTRACT

The increase in vehicles in cities poses considerable social issues and obstacles. As automobiles and their devices generate more data, Vehicular Ad Hoc Networks (VANET) can help enhance network performance. VANETs provide connectivity between vehicles and infrastructure, facilitating the exchange of information and the sharing of resources. To support VANETs, Vehicular Cloud Computing (VCC) leverages cloud concepts in this environment. Vehicles in the Vehicular Cloud processing (VCC) network frequently seek resources such as processing power, bandwidth, and storage, which they (vehicles) are unable to process on their own due to resource limitations. They seek these services, which are sometimes provided, sometimes blocked because the resource is already in use by another vehicle, and sometimes rejected owing to a shortage of available resources. In the same circumstance, some resources may remain idle simply because no proper technique was employed to allocate these resources to the cars, causing a challenge in VCC. This study introduces the Cooperative Particle Swarm Optimization (CPSO) Algorithm, an enhanced variant of Particle Swarm Optimization (PSO) resource allocation mechanism for vehicular clouds. The technique employs metaheuristics to optimize search and allocate resources in a vehicular cloud. A fog-based paradigm to help with the allocation process was established. The CPSO was compared to four different algorithms: MARIA, GREEDY, FRACTAL, and WORST. During the comparison process, we consider the number of blocked, attended, and denied services, as well as throughput. Simulation results indicate that the CPSO outperformed other techniques in all four performance aspects: blocking fewer, attending more, rejecting fewer services and increasing throughput.

INTRODUCTION

People and companies have required access to computer resources such as servers, storage, databases, networking, software, and analytics over the internet rather than relying on local infrastructure or personal devices (Obidike *et al* 2025). With the rapid innovations of storage and powerful computational processing technologies, as well as the achievements of the Internet have made computing resources to be affordable at reduced pricing and more available than ever before (Edje, 2020). These resources are provided by cloud service providers and are typically hosted in remote data centers. All these constitute to Cloud computing. Cloud computing has become an active area of research over the last decade to date, (Edje and Muhammad, 2020).

Our lives nowadays experience sudden, exponential changes in technologies, including the ones involved with transportation, which directly impacts social and economic aspects of human life (Quessada *et al*., 2021). The advancement of technology in the Automotive Sector is well perceived in recent improvements in the safety and experience of passengers traveling in road vehicles (Ribeiro *et al*., 2023). With each passing year, there is a significant increase in the number of vehicles around the world. As a result, the number of connected vehicles circulating on the streets among us also grows, sharing more data than ever before (Meneguette *and* Marques, 2022).

Vehicular Ad-Hoc Network (VANET) are a collection of vehicles connecting by wireless networks and provide services such as traffic management and transportation by applying information and communication technologies(Kaleibar *and* Abbaspour, 2020). In the last years, the Intelligent transportation system (ITS) involves the Vehicular Ad-Hoc Network (VANET) to facilitate data exchange among vehicle (Ezzidani *et al*., 2021). Vehicles will be highly connected with the aid of ubiquitous wireless networks (Liu *et al*., 2024). Many modern smart vehicles are connected to the cloud in Vehicular Cloud Computing (VCC) to offer various services, such as information, storage, cooperation, computation, and infotainment as a service (Pande *et al*., 2021).

The transportation industry has also encountered new development opportunities, presenting a promising prospect for the collaborative development of an intelligent transportation system that integrates "human vehicle-road-cloud" (Li *et al*., 2024). This explosion of new applications has, nonetheless, brought new challenges, where efficient and effective allocation of computational resources for the fulfillment of application requirements is at the crux of them all (Ribeiro *et al*., 2022). Due to limited storage and computational capabilities such a huge amount of multimedia-related data cannot be processed on the standalone onboard devices (Siddiqi *et al*., 2020).

Recently, several researches have considered the way to offload the tasks of vehicles to vehicle nodes (VNs) with more computing resources than the vehicle's local devices (Zhang *et al*., 2023). To assist the vehicular cloud in the management of available resources and offer a broader range of services, without impacting the network and the user experience the paradigm of Fog is explored (Pereira *et al*., 2021). The study has a number of resource allocation models that have been put out for VCs. The vast majority of these models are based on a set of methods, being the following among the most widely adopted: greedy algorithms, meta-heuristics, combinatorial optimization, multi-objective optimization, dynamic programming and reinforcement learning (Ribeiro *et al*., 2023). In this study, we are proposing a resource allocation in vehicular cloud computing network based on Cooperative Particle Swarm Optimization (CPSO) algorithm. This technique will tend to tackle the problems of resource allocation in vehicular cloud computing.

MATERIALS AND METHODS

Analysis of the developed model

Cooperative Particle Swarm Optimization (CPSO) is a population-based metaheuristic algorithm that extends traditional Particle Swarm Optimization (PSO) by integrating the concept of multi-swarm collaboration. Van den Bergh and Engelbrecht devised CPSO to improve PSO's ability to handle complicated, high-dimensional, and multimodal optimization problems, which frequently trap traditional PSO in local optima due to premature convergence. The core idea behind CPSO is to divide the high-dimensional search space into smaller, more manageable subcomponents, which are then optimized collaboratively utilizing numerous sub-swarms. Each sub-swarm optimizes a specific subcomponent of the overall solution vector, allowing for a divide-and-conquer strategy to the optimization issue.

In classical PSO, each particle represents a potential solution in a multidimensional search space and modifies its position in response to its own and its neighbors' experiences. While this strategy is effective for low-dimensional problems, it loses diversity and prematurely converges in high-dimensional landscapes. This convergence frequently traps particles in local optima, particularly in problems with several peaks and valleys in the fitness landscape. CPSO overcomes this limitation by introducing a cooperative coevolutionary strategy that enables each subcomponent of the solution vector to be optimized independently while cooperating with the rest of the system.

CPSO operates by breaking the overall solution vector into smaller sub-vectors, each representing a sub-swarm. For example, a solution vector with dimensionality D can be divided into K subcomponents, each of which may have one or more dimensions. Each sub-swarm has a population of particles that investigate and exploit their own subcomponent of the solution. The fitness of each particle in a sub-swarm is evaluated by merging it with the best-known solutions from other sub-swarms, resulting in a complete solution vector known as the context. This context vector functions as a cooperative framework, allowing each sub-swarm to assess the impact of its changes within the context of the overall solution.

The shared context vector ensures that sub-swarms cooperate. For each particle in a sub-swarm, the context vector is created by replacing the relevant piece of the vector with the particle's own position and using the best-known positions from the other sub-swarms for the remaining components. The resulting complete solution is then assessed using the problem's fitness function. If the new solution improves on the particle's personal best or the sub-swarm's overall best, appropriate updates are applied. This cooperative technique ensures that each sub-swarm optimizes its component in relation to the global solution space, resulting in better coordinated and effective search behavior.

CPSO has various advantages over normal PSO. By breaking down the problem, it minimizes the dimensionality of the search space that each particle must explore, resulting in faster convergence within each sub-swarm. The algorithm's cooperative character contributes to overall population variety, lowering the risk of premature convergence and allowing the algorithm to more effectively escape from local minima. Furthermore, CPSO is modular by design and can be efficiently parallelized, making it suited for large-scale and distributed optimization issues.

In the domain of cloud computing, particularly vehicle cloud computing (VCC), CPSO has demonstrated significant potential for resolving resource allocation issues. VCC environments are distinguished by dynamic, mobile, and heterogeneous computing nodes, making efficient resource allocation important to performance. CPSO can be used to efficiently allocate computational jobs (cloudlets) to virtual machines (VMs), control bandwidth distribution, and even assign storage resources in such networks. In this situation, each sub-swarm can be allocated to optimize a specific resource type or task segment, and their collaboration guarantees that the total system performance is optimal.

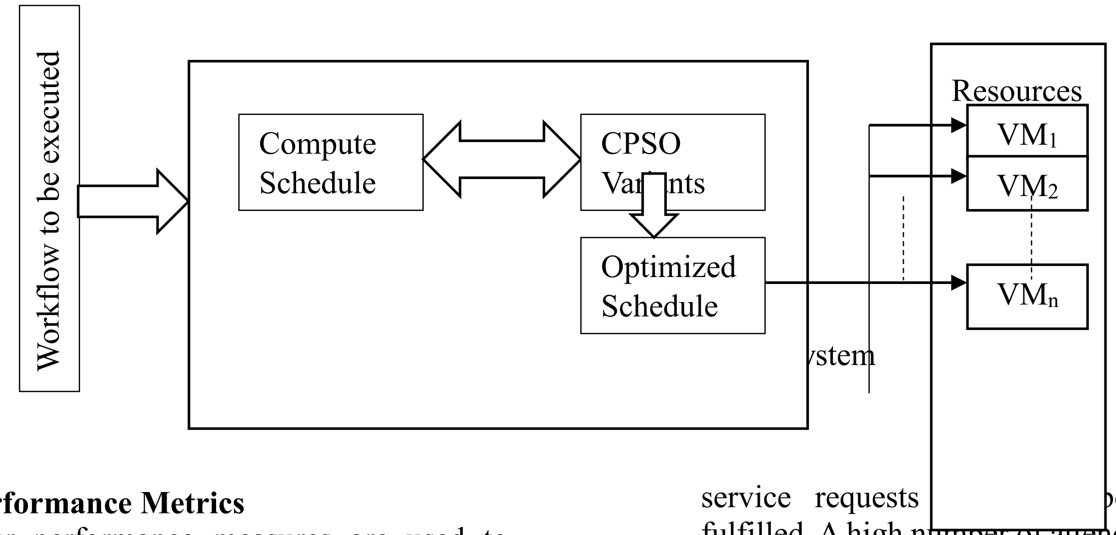
From a theoretical standpoint, CPSO is an example of cooperative coevolution, a broader category of algorithms inspired by biological evolution in which many populations evolve in tandem. CPSO demonstrates how cooperation among specialized organisms can result in emergent problem-solving capabilities, a notion shared by many natural and social systems. CPSO's search space decomposition and collaborative fitness evaluation mechanism make it a reliable and scalable solution for a variety of optimization challenges.

Cooperative Particle Swarm Optimization is a considerable improvement over classical PSO, providing better performance in high-dimensional and difficult optimization situations. Its capacity to partition the search area, coordinate many swarms, and retain solution variety makes it ideal for real-world applications like work scheduling, resource allocation, and service optimization in vehicular cloud systems. As computational challenges increase in size and complexity, the principles underlying

CPSO—modularity, cooperation, and contextual evaluation—are likely to remain central to the development of next-generation optimization algorithms.

Architecture of the developed System

System architecture is a high-level plan or idea that shows how a system is put together, what parts it has, how they work together, and how they relate to each other. It shows how hardware, software, and people work together to reach certain goals, making sure that the system meets business needs, is easy to manage, and works quickly and safely. Figure 3.5 shows the architecture of the CPSO resource allocator.



Performance Metrics

Four performance measures are used to assess the completed work. They are: Accepted Services, refused services, blocked services, and throughput.

1. **Accepted Services:** This is a metric that represents the number of service requests that have been fulfilled. A high number of attended services indicates that the allocation policy is efficient in optimizing resource allocation, considering the evaluated interval. Accepted service can be calculated as seen in equation 3.1 below

$$Accepted\ service = \frac{N_{acc}}{N_{req}} \qquad (3.1)$$

   Where N_{acc} is the number of successfully executed requests and N_{req} is the total number of service requests

2. **Refused Services:** These are the services that were prevented from allocating their resources in all VCs, due to the insufficient resources of the VCs to provide such services. The services where the evaluated algorithm is unable to allocate the necessary resources needed for the determined service in the cloud. Refused service can be calculated as seen in equation 3.2 below

$$Refused\ Service = \frac{N_{ref}}{N_{req}} \qquad (3.2)$$

   Where N_{ref} is the number of Refused Requests and N_{req} is the total number of service requests

3. **Blocked Services:** These correspond to the number of times that VC cannot attend a service due to a lack of resources. It is a metric that computes the number of times a VC refuses a service request. Blocked service can be calculated as seen in equation 3.3 below

$$Blocked\ Service = \frac{N_{blk}}{N_{req}} \qquad (3.3)$$

Where N_{blk} is the number of blocked requests and N_{req} is the total number of service requests

4. **Throughput:** This refers to the total number of tasks accomplished within a given execution time, which can be calculated as

$$Throughput = \frac{\sum M_{I,J}}{Execution\ time} \tag{3.4}$$

Where $\Sigma M_{i,j}$ is the number of successfully completed task

Experimental Setup

For this experiment, the Manhattan district in New York, USA (See map in Figure 3.6), was considered. For the district's four RSUs (Road Side Units) to be connected to one another and be able to interact over a 5G network, they are positioned at key locations. An Edge Cloud (EC) is placed on each RSU and is in charge of overseeing the distribution of network computing resources. Different quantities of cars are taken into consideration for the simulation, which is generated based on the simulation time in each scenario, which is 2400, 4800, and 7200 seconds. As a result, 381 vehicles, 778 vehicles, and 1175 vehicles were produced in each simulation. Additionally, it is divided into time slots with 480 seconds each, making them 5, 10, and 15 slots respectively for a specific simulation duration.

The Pearson III distribution, which is regarded as an advanced gamma pattern that can imitate vehicle entrances and exits in a heterogeneous manner, was employed to simulate vehicle entry and exit in a heterogeneous manner. All computational resources are assumed to be 100% shared by each EC (EC = [100, 100, 100, 100]). Processing speed, bandwidth, memory, and storage capacity are the shared resources. Every vehicle service has the same computational resources, and consumption figures are produced at random within the interval [1, 6], taking into account low-demand services like multimedia, entertainment, security, and text messaging, among others. During the simulation, artificial data is produced. 33 runs of each simulation were conducted, and a 95% CI was used. The simulation was developed in Python programming language.

Table 3.1: Workload Parameters

Workloads	Parameters	Value
	File Size	Memory
Google Cloud Trace Log	5GB	50GB

Table 3.2: Simulation Setup

Parameters	**Values**
Urban Scenery	City of Manhanttan
Number of Vehicles	381 – 778 - 1175
Time per slot	480 seconds
Time Slots	5 – 10 - 15
Simulation time	2400 – 4800 – 7200 seconds
Resource values	Synthetic and random of [1,6]
Vehicle entry and exit	Pearson III distribution

Number of runs	33
Confidence interval	95%
Programming Language	Python programming language
Simulation environment	Microsoft visual studio 2022

RESULTS

The study developed a system that implements the Cooperative Particle Swan Optimization Algorithm, which is an enhanced version of the PSO Algorithm for resource allocation and optimization in vehicular cloud computing. This study tends to develop a model that resolves the problem of slow and premature convergence, especially in complex and multi-modal environments. The swans which divided into sub problems helps to prevent premature convergence and enhances the adaptability of the algorithm to changes. Figure 4.1 shows the flowchart of the CPSO algorithm.

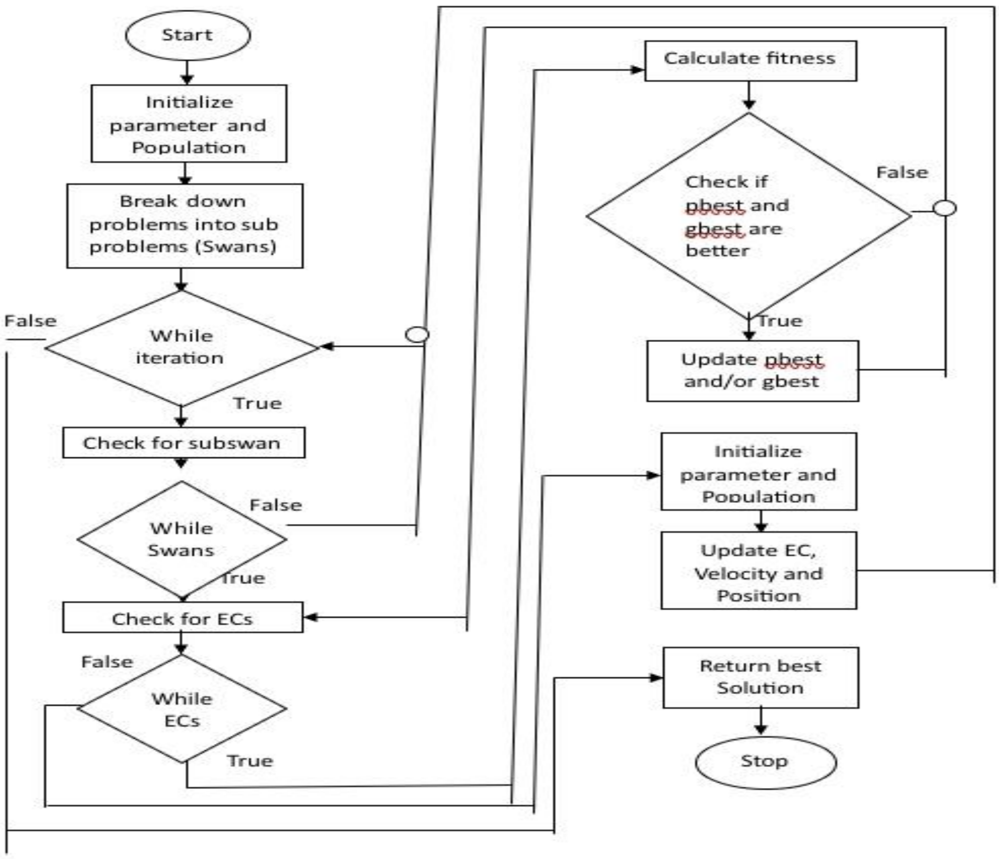


Figure 4.1: Flowchart of the developed model

System Algorithm

Algorithm 2: CPSO

1. Start
2. Break down problem into subproblems (Swan)
3. For each Swan
4. **for** each particle $i = 1, ..., S$ **do**

5. Initialize the particle's position with a uniformly distributed random vector: $\mathbf{x}_i \sim U(\mathbf{b}_{lo}, \mathbf{b}_{up})$
6. Initialize the particle's best known position to its initial position: $\mathbf{p}_i \leftarrow \mathbf{x}_i$
7. **if** $f(\mathbf{p}_i) < f(\mathbf{g})$ **then**
8. update the swarm's best known position: $\mathbf{g} \leftarrow \mathbf{p}_i$
9. Initialize the particle's velocity: $\mathbf{v}_i \sim U(-|\mathbf{b}_{up}-\mathbf{b}_{lo}|, |\mathbf{b}_{up}-\mathbf{b}_{lo}|)$
10. **while** a termination criterion is not met **do**:
11. **for** each particle $i = 1, ..., S$ **do**
12. **for** each dimension $d = 1, ..., n$ **do**
13. Pick random numbers: $r_p, r_g \sim U(0,1)$
14. Update the particle's velocity: $\mathbf{v}_{i,d} \leftarrow w\, \mathbf{v}_{i,d} + \varphi_p\, r_p\, (\mathbf{p}_{i,d}-\mathbf{x}_{i,d}) + \varphi_g\, r_g\, (\mathbf{g}_d-\mathbf{x}_{i,d})$
15. Update the particle's position: $\mathbf{x}_i \leftarrow \mathbf{x}_i + \mathbf{v}_i$
16. **if** $f(\mathbf{x}_i) < f(\mathbf{p}_i)$ **then**
17. Update the particle's best known position: $\mathbf{p}_i \leftarrow \mathbf{x}_i$
18. **if** $f(\mathbf{p}_i) < f(\mathbf{g})$ **then**
19. Update the swarm's best known position: $\mathbf{g} \leftarrow \mathbf{p}_i$
20. Stop

DISCUSSIONS

In this section, we showed the results gotten after our simulations. The developed algorithm is experimented on 381, 778, and 1175 vehicles in simulation time 2400, 4800, and 7200 seconds. The total number of services requested by the vehicles can be seen in Table 4.1. The 4 metrics (Accepted services, blocked services, refused services, and throughput) are used to compare with MARIA, FRACTAL, Greedy, and Worst algorithms.

Accepted Services

Looking at Figure 4.3, we present the evaluation of the number of services accepted. This metric represents the services of the vehicles that we managed to allocate in a cloud computing service. In the first configuration, with 371 vehicles in the scenario, CPSO performed better, accepting an average of 346 services, followed by MARIA with 301, FRACTAL with 288, Greedy with 280 and Worst with 279. For the second configuration with 778 vehicles, CSPO accepted 454 services, MARIA with 495, FRACTAL with 489, Greedy and Worst are tied with 359 though the CPSO has less number of accepted services than MARIA and FRACTAL but it can be seen that it accepted 58.38% of the total number of services requested by the vehicles at that simulation time. Same applies for configuration with 1175 vehicles where CPSO accepted 543, MARIA with 699, FRACTAL with 696, Greedy with 412 and Worst with 413. CPSO performed in terms of the percentage of services accepted.

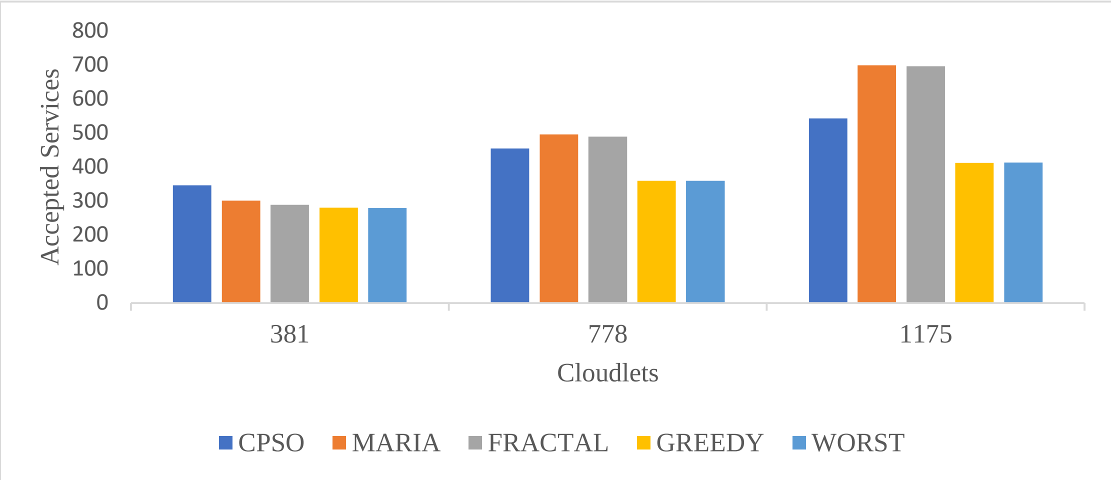


Figure 4.1: Comparison of Accepted Services

Blocked Services

Figure 4.4 illustrates the average number of blocked services in vehicular clouds. Blocked services occur when a vehicular cloud attempts to allocate a service but lacks the necessary resources. In the configuration of 381 vehicles, CPSO blocked services the least, with an average of 8 blocks. While MARIA, FRACTAL, Greedy and Worst has 436, 570, 736, and 810 blocks respectively. For 778 vehicle configurations, CPSO also blocked lesser services with 299 blocks while MARIA, FRACTAL, Greedy and Worst has 1434, 1542, 2132, and 2252 blocks. Then with the last configuration of 1175 vehicles, The CPSO has 604 blocks, MARIA with 2402, FRACTAL with 2485, Greedy and Worst tied with 4724.

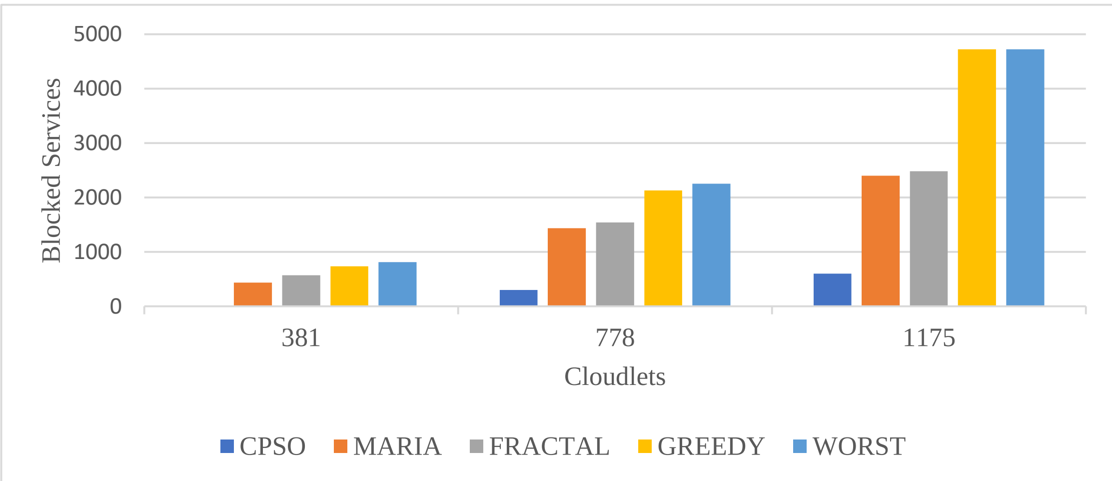


Figure 4.2: Comparison for Blocked Services

Refused Services

Figure 4.5 shows the results for refused services, which were not assigned in any of the VCs due to limited resources. In the initial configuration of 381 vehicles, the CPSO denied an average of 17 services.

While MARIA, FRACTAL, Greedy, and Worst denied 80, 86, 101, and 102 services respectively. For 778 vehicle configurations, the CPSO denied an average of 25 services. While MARIA, FRACTAL, Greedy, and Worst denied 281, 282, 419, 419 services. For 1175 vehicle configurations, the CPSO denied an average of 29 services. While MARIA, FRACTAL, Greedy, and Worst denied 472, 473, 763, 761 services.

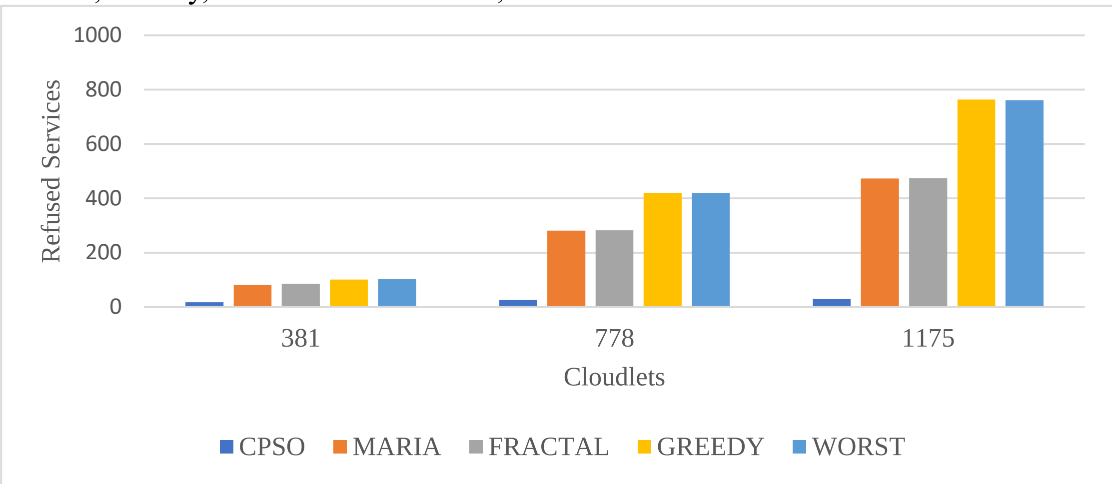


Figure 4.3: Comparison of Refused Services

Throughput

This metric depicts the accepted service in VC per unit time. For the first configuration of 381 vehicles, the CPSO has 0.14 services per unit time, MARIA has 0.13 while FRACTAL, Greedy and Worst has 0.12 each. For 778 vehicle configuration CPSO has 0.09, MARIA and FRACTAL with 0.1, while Greedy and Worst is tied with 0.07. For 1175 vehicle configuration CPSO has 0.08, MARIA and FRACTAL with 0.1, while Greedy and Worst is tied with 0.06.

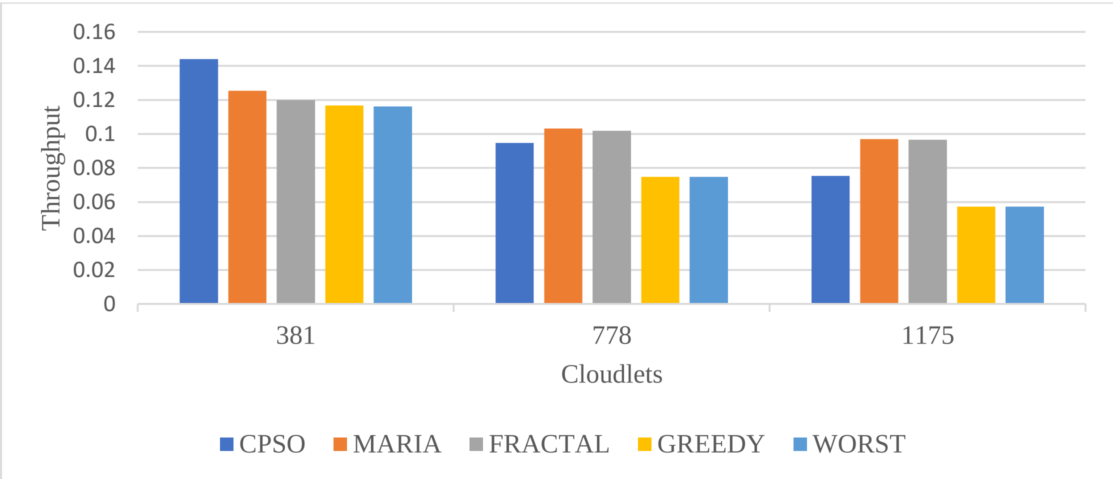


Figure 4.4: Comparison for Throughput

CONCLUSION

As smart transportation systems, vehicle numbers, and technology evolve, smart cities face new obstacles to improve their services. One of the issues is improving vehicle service requests for faster and more efficient operations. Cloud computing enables speedier service delivery. However, limited computational resources necessitate optimizing resource allocation. This paper introduces the CPSO, an improved version of the bio-inspired PSO algorithm that optimizes resource allocation in VANETs. The CPSO are easily adaptable to various environments and circumstances.

ACKNOWLEDGEMENTS

We thank the Department of Computer Science of the Faculty of Science in Delta State University Abraka. We are profoundly thankful to family members, friends and other colleagues who contributed in any way to the successful completion of this project. Your support is deeply appreciated. Finally, all the glory I give back to God Almighty for making it possible for all the above mentioned persons to be able to make their different inputs.

REFRENCES

Edje E. A. (2021) “Enhanced non-parametric sequence learning shceme for internet of things sensory data in cloud infrastructure” *Doctoral dissertation, Universiti Teknologi Malaysia*

Edje E. A. (2019) “A survey on the utilization of cloud computing services for academic learning ” *Journal of the Nigerian Association of Mathematical Physics, Vol. 49 No. 1*

Edje E. A., and Muhammad. L. S. (2020) "Cloud Computing Enabled Data Center Infrastructure Development and Deployment by IT Firms." *International Journal of Computing and Digital Systems*, 9(1), 37+

Ezzidani, A., Ouammou, A., Hanini, M., and Tahar, A. Ben. (2021). A SMDP Approach to Evaluate the Performance of a Vehicular Cloud Computing System with Prioritize Requests. *Mathematical Modelling of Engineering Problems*, *8*(6), 928–936. https://doi.org/10.18280/mmep.080612

Kaleibar, F. J., and Abbaspour, M. (2020). An approach to model the optimal service provisioning in vehicular cloud networks. *2020 11th International Conference on Information and Knowledge Technology, IKT 2020*, 62–66. https://doi.org/10.1109/IKT51791.2020.9345638

Kaleibar, F. J., and St-Hilaire, M. (2024a). A Connectivity-aware Method for Infrastructure-less Vehicular Cloud Service Discovery. *2024 IEEE 10th World Forum on Internet of Things, WF-IoT 2024*, 619–624. https://doi.org/10.1109/WF-IoT62078.2024.10811226

Kaleibar, F. J., and St-Hilaire, M. (2024b). SLA-Based Service Provisioning Optimization in Vehicular Cloud Networks Using Fuzzy Logic. *IEEE Access*, *12*, 101727–101744. https://doi.org/10.1109/ACCESS.2024.3430826

Liu, X., Zheng, J., Zhang, M., Li, Y., Wang, R., and He, Y. (2024). A Game-Based Computing Resource Allocation Scheme of Edge Server in Vehicular Edge Computing Networks Considering Diverse Task Offloading Modes. *Sensors*, *24*(1). https://doi.org/10.3390/s24010069

Meneguette, R. I., and Marques, H. A. P. (2022). A Game Theory-Based Vehicle Cloud Resource Allocation Mechanism. *Revista Eletrônica de Iniciação …, June*. https://sol.sbc.org.br/journals/index.php/reic/article/view/2281%0Ahttps://sol.sbc.org.br/journals/index.php/reic/article/download/2281/1940

Obidike, C. (2025). *MULTI SCHEMES FOR DYNAMIC TASK SCHEDULING IN CLOUD*

COMPUTING. *June*. https://doi.org/10.4314/sa.v24i2.9

Pande, S. K., Panda, S. K., Das, S., Sahoo, K. S., Luhach, A. K., Jhanjhi, N. Z., Alroobaea, R., and Sivanesan, S. (2021). A Resource Management Algorithm for Virtual Machine Migration in Vehicular Cloud Computing. *Computers, Materials and Continua*, *67*(2), 2647–2663. https://doi.org/10.32604/cmc.2021.015026

Quessada, M. S., Lieira, D. D., Pereira, R. S., De Grande, R. E., and Meneguette, R. I. (2021). A Bat Bio-inspired Mechanism for Resource Allocation in Vehicular Clouds. Proceedings - 17th Annual International Conference on Distributed Computing in Sensor Systems, DCOS 2021, 197–204. https://doi.org/10.1109/DCOSS52077.2021.00042

Ribeiro, A., da Costa, J. B. D., Filho, G. P. R., Villas, L. A., Guidoni, D. L., Sampaio, S., and Meneguette, R. I. (2023). HARMONIC: Shapley values in market games for resource allocation in vehicular clouds. *Ad Hoc Networks*, *149*(May), 103224. https://doi.org/10.1016/j.adhoc.2023.103224

Ribeiro, A., Filho, G. P. R., Guidoni, D. L., De Grande, R. E., Sampaio, S., and Meneguette, R. I. (2022). A Shapley Value-based Strategy for Resource Allocation in Vehicular Clouds. *Proceedings - IEEE Global Communications Conference, GLOBECOM*, 5801–5806. https://doi.org/10.1109/GLOBECOM48099.2022.10001300

Siddiqi, M. H., Alruwaili, M., Ali, A., Haider, S. F., Ali, F., and Iqbal, M. (2020). Dynamic priority-based efficient resource allocation and computing framework for vehicular multimedia cloud computing. *IEEE Access*, *8*(April), 81080–81089. https://doi.org/10.1109/ACCESS.2020.2990915

Zhang, K., Yang, J., and Lin, Z. (2023). Computation Offloading and Resource Allocation Based on Game Theory in Symmetric MEC-Enabled Vehicular Networks. *Symmetry*, *15*(6). https://doi.org/10.3390/sym15061241