An Efficient Genetic Algorithm for Service Placement in Fog Computing

Dihia Bendjenahi, Chadia Moumeni and Malika Bessedik
Higher National School of Computer Science, Algiers, Algeria
{*jd_bendjenahi, c_moumeni, m_bessedik*}*@esi.dz*

Keywords: Fog Computing, IoT, Service Placement Problem, Meta-heuristic Optimization Approach, Resource Allocation.

Abstract: The FSPP (Fog Service Placement Problem) involves the allocation of fog and cloud resources to IoT applications while meeting specific application requirements, including deadlines and minimizing response time. This paper addresses the FSPP in heterogeneous fog-cloud computing environments using a genetic algorithm (GA) approach. Our proposed GA aims to jointly maximize total resource utilization while respecting application deadlines. The paper presents a detailed system model, problem formulation, and the proposed GA methodology. Experimental results demonstrate the effectiveness of the GA approach in optimizing resource allocation and meeting Quality of Service (QoS) requirements when compared to the first-fit heuristic and to the random approach.

1 INTRODUCTION

In a rapidly evolving world shaped by technological advancements, inter-connected devices have seamlessly integrated into various domains, including industry, education, healthcare, manufacturing, and more, the IoT (Internet of Things) consequently emerged as a dynamic and swiftly advancing field, gaining popularity and demonstrating effectiveness in providing many services to its users (Apat et al., 2023).

During the initial stages of IoT development, cloud computing served as the primary processing layer, due to the limitations of computing capabilities within IoT devices, which did not allow it to perform all the operations required by IoT applications (Ali, 2015).

Cloud computing is an environment that offers different computing resources to users with a pay-as-you-go model. It consists of globally distributed data centers, offering an economically viable alternative to investing in physical infrastructure (Ahmad et al., 2017). However, the evolution of IoT applications towards increased criticality and complexity revealed a need for real-time processing and rapid responses. Cloud computing architectures which are centralized in nature, struggled to satisfy the real-time demand of IoT applications (Minh et al., 2017). This challenge pushed researchers to introduce an innovative computing architecture known as fog computing.

Fog computing was introduced by Cisco in 2012 (Bonomi et al., 2012) as an intermediate layer between end users and the cloud where the adoption of fog computing for IoT applications significantly improved the system's performance, since processing devices, or fog nodes, are now geographically proximate to IoT devices. However, the limitations of fog devices necessitated continued reliance on cloud computing for complex and resource-intensive tasks. This introduced a crucial challenge, known as the FSPP (Fog Service Placement Problem) (Skarlat et al., 2017). The FSPP involves the allocation of fog and cloud resources to IoT applications while meeting specific application requirements, including deadlines, minimizing response time, and managing energy consumption. The complexity of this task arises from the heterogeneous nature of fog devices, each with varying processing capabilities, and the diverse computing resource demands of IoT applications. The main challenges of service placement in the fog computing landscape are to improve the performance of IoT applications as well as develop more efficient and optimized algorithms for mapping application modules or services to fog nodes (Salaht et al., 2021). However, these challenges may require significant research efforts and may not have straightforward solutions.

Due to the complexity of fog computing infrastructures, and the varying demands of IoT applications in terms of computing resources and delay sen-

Bendjenahi, D., Moumeni, C. and Bessedik, M.
An Efficient Genetic Algorithm for Service Placement in Fog Computing.
DOI: 10.5220/0013158900003890

In *Proceedings of the 17th International Conference on Agents and Artificial Intelligence (ICAART 2025) - Volume 3*, pages 547-554
ISBN: 978-989-758-737-5; ISSN: 2184-433X

sitivity, the service placement problem in fog computing is considered an NP-complete problem (Liu et al., 2022), this means that it's difficult to solve it to optimality in a decent time with limited computing resources. Meta-heuristics can be employed to solve the FSPP due to their ability to efficiently explore the solution space and converge to high-quality solutions for optimal placements in a shorter amount of time. Different methods have been employed to solve this problem such as Genetic Algorithms and Particle Swarm Optimization algorithms are considered an interesting choice for this problem, as they proved their efficiency in solving a variety of optimization problems.

Therefore, In this paper, we address the FSPP problem in heterogeneous fog-cloud computing environments. We aim to jointly maximize the total resource utilization while considering service deadlines which helps to respect the delay-sensitivity of applications and to efficiently utilize fog resources.

The rest of the paper is organized as follows; The next section addresses the related literature. We present system modeling, including system architecture, resource and application models in Section 3. Then, we provide the problem formulation, in Section 4. The proposed approach is introduced in Section 5. In Section 6, we evaluate the performance of the proposed method. Finally, we conclude our study in Section 7.

2 RELATED WORK

In the following, we review some of the studies that have been conducted on this still-challenging issue.

The article (Skarlat et al., 2017) presents a framework for fog computing comprising colonies of fog, each with a primary node and fog cells dedicated to task execution, along with middle-ware interfacing between the fog and the cloud. A genetic algorithm is employed to solve the FSPP, which aims to optimize service allocation across resources within a fog computing setup, prioritizing placing services on fog nodes rather than using the cloud.

Aladwani (Natesha and Guddeti, 2021) developed a task scheduling algorithm named Tasks Classifications and Virtual Machines Categorization (TCVC) for scheduling IoT healthcare tasks in fog computing based on tasks importance by using the MAX-MIN algorithm.

Natesha and Guddeti (Hassan et al., 2020) addressed the FSPP with two-level resources using docker and containers. The problem is formulated as a multi-objective optimization problem that aims to minimize the cost, energy consumption, and service time by proposing an elitism-based genetic algorithm (EGA) for preserving a set of best solutions between the algorithm generations, avoiding altering them with crossover or mutation operations.

The authors in (Djemai et al., 2019) addressed the problem of service placement in fog–cloud environments by formulating it as a mixed integer linear programming (MILP) problem with the objective of achieving high QoS, low energy consumption, and maximum resource usage. The authors classified the IoT services into critical and normal categories. Then, they proposed two heuristic-based algorithms to solve the problem. The first algorithm aims to provide high QoS for critical services while the second focuses on the energy efficiency of the fog environment.

The research paper (Santoyo-González and Cervelló-Pastor, 2018) proposes an infrastructure and applications model for IoT, as well as an optimization strategy that takes into account the system's energy consumption, for the placement of energy-efficient IoT services on a fog computing infrastructure. The proposed approach uses discrete particle swarm optimization algorithm (DPSO) to optimize the placement of IoT services taking into account energy consumption and application requirements. The article presents simulation results that demonstrate the effectiveness of the proposed DPSO approach in the realization of an energy efficient IoT service placement on a fog infrastructure.

The authors of (Canali and Lancellotti, 2019) stress the need to optimize infrastructure placement service to minimize delays in the service access layer. The authors models the FSPP in a Fog Computing/NFV environment as a linear programming problem in mixed integer format. Then, they propose a heuristic solution taking into account requirements of the 5G mobile network.

Authors in (Jamil et al., 2019) addressed the use of genetic algorithms for solving the FSPP. They proposed a heuristic based on genetic algorithms to provide a scalable solution for mapping sensors over fog nodes.

Resource utilization is one of the critical objectives in fog computing. Therefore, most of the studied works focus on providing an efficient placement that maximizes resource usage. However, the challenges persist in efficiently distributing resources and prioritizing services in fog-cloud computing environments. As a result, we propose a service placement approach that efficiently allocates the cloud-fog resources while considering the different requirements of IoT applications.

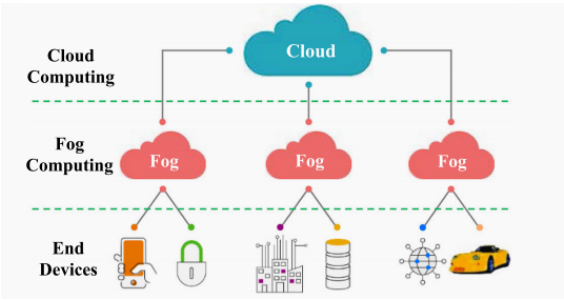

Figure 1: Fog Computing Architecture (Lin et al., 2020).

3 SYSTEM MODEL

3.1 System Architecture

In this section we present the used architecture of fog computing.

The fog computing architecture consists of three main layers, the top one is the cloud layer, the intermediate layer consists of fog devices or fog nodes, while the bottom layer is the IoT layer that includes IoT devices. Figure 1 shows the used architecture of fog computing.

It is advisable to keep all the given values because any text or material outside the aforementioned margins will not be printed.

3.1.1 IoT Layer

This layer consists of sensors and actuators. Sensors like cameras, heartbeat sensors, humidity sensors, temperature sensors, GPS sensors, etc, collect raw data from the external environment, convert it to signals, and transmit them to fog nodes for further processing. After processing, fog nodes send the results back to actuators that work as controllers to perform the necessary actions accordingly (Dokeroglu et al., 2019).

3.1.2 Fog Layer

The fog layer loosely integrates both cloud and IoT layers associated with the fog, thereby enabling independent evolution and a high degree of interaction between multiple layers (Dokeroglu et al., 2019). The fog layer is composed of heterogeneous fog devices having limited computing, storage, and networking capabilities e.g. routers, switches, proxy servers, and cellular base stations (Dokeroglu et al., 2019).

3.1.3 Cloud Layer

The cloud gets data from networking devices for long term behavior and data analysis and returns the results back to fog devices for further necessary actions (Dokeroglu et al., 2019).

4 PROBLEM FORMULATION

We assume that the services of each application can be placed and processed by any of its accessible fog nodes if it has the necessary computing capabilities and each node can offload its workload to the cloud if necessary.

As a result, in our model, we focus on maximizing resource usage as the objective of the FSP problem while satisfying the problem constraints which are described in 4.4.

Let x_{ij} be a binary variable that denotes whether an IoT service S_i is placed on the fog node F_j. It is formulated as follows:

$$x_{ij} = \begin{cases} 1 & \text{if service } S_i \text{ is placed on the fog node } F_j \\ 0 & \text{otherwise} \end{cases}$$

4.1 Application Model

We represent IoT applications as a set of services that are submitted by different IoT devices to fog nodes for execution. Each application A_k consists of a set of services S_i.

Each service S_i is characterized by its computing requirements; CPU demand (in million instructions per second – MIPS), memory demand (in megabyte – MB), storage demand (in megabyte – MB), its deadline (in milliseconds – ms), and size (in million instructions – MI) as follows:

$$S_i = \{ S_i^{\text{CPU}}, S_i^{\text{RAM}}, S_i^{\text{STOR}}, S_i^{\text{DL}}, S_i^{\text{Size}} \}$$

4.2 Resource Model

The fog nodes are represented as a non-directed graph G = (F, E), where F represents the set of fog nodes , and E is the set of links between fog nodes. Every fog resource has the following features:

$$F_j = \{ CPU_j, RAM_j, STOR_j \}$$

Where CPU represents the processing power, which is the number of instructions that can be executed per second CPU_j, the memory size RAM_j, the permanent storage size $STOR_j$ available.

4.3 Resource Utilization

Resource usage represents the ratio between the resources consumed by the served applications and the total available resources, i.e. the percentage of the usage of resources by IoT services in the fog system, which is defined as follows:

$$RU_{F_j} = \sum_{j=1}^{m} \sum_{i=1}^{n} \frac{x_{ij} \cdot RC_{S_i}}{R_{F_j}} \quad (1)$$

Where RC_{S_i} represents the resource demand of the IoT service S_i, m is the number of fog nodes, n is the number of services, and R_{F_j} is the resource capacity of the fog node F_j.

Besides, P_{S_i} indicates the execution priority of S_i. Therefore, we classify services based on their deadline (DL) as shown in equation 2.

$$P_{S_i} = \frac{1}{S_i^{\mathrm{DL}}} \quad (2)$$

4.4 Objective Function

We formulate the FSPP model for optimizing fog nodes resource usage using 7 while satisfying a set of constraints as follows:

$$RS_i \leq S_i^{\mathrm{DL}}, \forall i \in [1,n], \forall F_j \in FN \quad (3)$$

Where RS_i as denoted in equation 3 is the response time for the service S_i, which is calculated as follows:

$$RS_i = \sum_{S_i \in A_k} \mathrm{latency}_j^{\mathrm{proc}} + \mathrm{latency}^{\mathrm{Com}}$$

$$\mathrm{latency}_i^{\mathrm{proc}} = \frac{S_i^{\mathrm{Size}}}{\mathrm{CPU}^j}$$

$$\mathrm{latency}^{\mathrm{Com}} = \frac{\mathrm{data}_{jk}^{\mathrm{size}}}{\mathrm{BW}_{jk}}$$

BW_{jk} in Mb/s is link bandwidth.

And for each fog node:

$$\sum_{\forall S_i} S_i^{\mathrm{CPU}} \cdot x_{ij} \leq CPU_j \quad (4)$$

$$\sum_{\forall S_i} S_i^{\mathrm{RAM}} \cdot x_{ij} \leq RAM_j \quad (5)$$

$$\sum_{\forall S_i} S_i^{\mathrm{STOR}} \cdot x_{ij} \leq STOR_j \quad (6)$$

Equations 4 through 6 denote that no fog node should exceed its available computing and storage resources.

The objective function for our FSPP is presented in Eq 7, where the aim is to maximize fog node utilization. Therefore, we compute the average resource utilization across all fog nodes as illustrated in the equation. Hence, providing an efficient resource distribution that reduces cloud costs and latencies (in terms of cloud computing and communication resources).

$$Maximize : OF = \frac{1}{m} \cdot \sum_{j=1}^{m} ([RU_{F_j} < (1-y)]) \quad (7)$$

Where a resource utilization value of 1 represents using 100% of the resources available on the fog node. y is a constant that allows us to define the percentage of resources used for tasks other than service placement, like monitoring and system updates, so if the value of y is 0.1, 10% of the fog nodes resources would be left unused for system related tasks.

5 AN EFFICIENT GENETIC ALGORITHM FOR SERVICE PLACEMENT IN FOG COMPUTING

We present a GA-based approach to address the FSP problem by proposing strategy that efficiently allocates tasks to fog nodes while optimizing resource utilization and meeting Quality of Service (QoS) requirements (e.g. application deadline) as illustrated in Figure 2.

A genetic algorithm aims to generate a global optima solution through the mechanisms of selection, crossover, and mutation, a set of candidate solutions are maintained in each iteration (Das et al., 2018; Xue et al., 2021). This procedure is repeated until the termination criterion is satisfied (Xue et al., 2021).

Next we introduce the different components of our approach.

5.1 Solution Representation

Each chromosome represents a potential solution, it consists of several genes, indicating the allocation of tasks to fog nodes in our case. Every chromosome has a fitness value, which is obtained by measuring the quality of the solution represented by the chromosome. A solution is represented by a list in our context, where the index of the list is the ID of the task, and the element at that index represents the ID of the fog node executing the task.

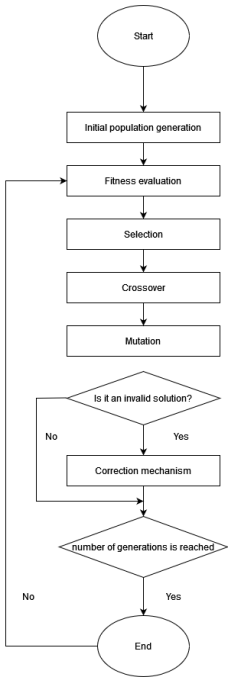

Figure 2: GA Diagram.

5.2 Initial Population

The population is initially generated in a random manner and all of the services are mapped to random fog nodes after prioritizing the IoT applications based on their deadline.

5.3 Fitness Evaluation

Fitness values are calculated using equation 7. We calculate the fitness score of a solution based on the percentage of fog nodes where the resource usage exceeded the available resources, and the number of fog nodes used in the solution. For each fog node that does not exceed its available resources while allocating services, we increment the fitness value by 1. The better the solution, the bigger fitness value, which helps ensure that only feasible and good quality solutions are chosen.

Note that fitness values undergo normalization by dividing them by the number of fog nodes present in the architecture. This process guarantees that fitness values fall within the range of 0 to 1.

5.4 Crossover Operation

We used uniform crossover, where two parents are randomly selected from the current population, and for each gene of the chromosome, a random gene is chosen either from the first or the second parent (with 0.5 selection probability for both), until we get our new offspring solution.

5.5 Mutation Operation

Random mutation was used, where each gene is mutated with a small probability, to explore neighboring solutions and introduce new genetic material into the population. The exploration rate is controlled by the mutation rate parameter, to help balance between exploration and exploitation.

5.6 Selection Operation

For each generation, the parent chromosomes are replaced by their offspring to form the next generation. Given that the parent selection in the crossover operation is random, some parents can have multiple offspring individuals, where others can have none.

5.7 Correction Mechanism

In case of invalid solutions, i.e. solutions violating resource constraints, a correction mechanism is applied to adjust task allocations and ensure solutions feasibility.

The correction mechanism verifies whether fog nodes exceed their available resources during the allocation of applications services, if a fog node is found to be executing more services than it can handle, the system isolates the services that are causing the violation, and attempts to find alternative execution nodes with enough available resources for each service. If a service requires more resources than any fog node can accommodate, it will be placed on the cloud as shown in Figure 3, which ensures the efficient usage of fog nodes and reduces the delay.

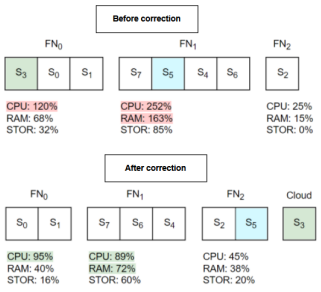

Figure 3: Example of the process of correcting an invalid solution.

Due to that GA seeks to explore a broad variety of potential solutions, services are assigned to fog nodes at random, which may result in resource capacity exceeding capacity, particularly in the early

phases of the optimization process. To guarantee that the solution satisfies resource limitations, a correction mechanism is implemented for all invalid solutions, whereby overloaded services are offloaded to the cloud. This guarantees that a solution gets modified to become viable in later iterations, even if the initial solution is invalid.

5.8 Top Solutions Preservation

A list containing the best solutions is maintained, storing the best solution from each generation. By the end of the evolution process, the best solution (with the highest fitness value) is selected from that list as the final solution.

In summary, the proposed genetic algorithm-based methodology offers a promising approach to address the fog service placement problem, enabling efficient resource allocation and QoS optimization in fog computing environments.

6 PERFORMANCE EVALUATION

6.1 Experimental Setup

To evaluate the effectiveness of our proposed method, we conducted a series of experiments using different parameters, and compared our genetic algorithm to a First-Fit heuristic, where each task is assigned to the first fog node that has enough resources to execute it (Brent, 1989), and a random method, which selects a random solution from the solution space, i.e. it chooses a random execution node for each service.

The datasets used in these experiments are detailed in 1.

Table 1: Datasets Information.

Test number	Fog nodes number	Tasks number
1	10	20
2	15	30
3	25	75

The fog nodes and services details are given in tables 2 and 3.

Table 2: Fog Nodes Characteristics.

ID	from 0 to (number of fog nodes - 1)
CPU_j	[500-3000] (MIPS)
RAM_j	[256-2048] (MB)
$STOR_j$	[256-2048] (MB)

The Java programming language (JAVASE-17) was used in the experimentation process. Eclipse was used as an IDE (2022-09 version) running on Windows 10.

Table 3: IoT Services Characteristics.

ID	from 0 to (number of tasks - 1)
S_i^{CPU}	[80-300] (MIPS)
S_i^{RAM}	[100-1024] (MB)
S_i^{STOR}	[0-512] (MB)
S_i^{DL}	[10-30] (s)

The used parameters of our genetic algorithm are: number of generations is 10, population size is 25 and mutation rate is 100.

6.2 Experimentation Results

The following results represents the averages of 100 executions. We selected different datasets where the allocation outcome can vary.

For the first set of experiments, the allocation process was relatively easy, and this is confirmed with the high accuracy of the obtained solutions, both with the genetic approach and the first fit algorithm. The second and third sets of experiments had a more challenging allocation process due to the specifications of the fog nodes and services.

The time was measured in milliseconds (ms), and the fitness value is between 0 and 1. During each run of the algorithms, we shuffle the datasets used, to help us obtain more accurate and less biased results. The detailed values are shown in the graphs given below in figures 4 through 8.

Figure 4 and 5 show the average execution times and average fitness values for each method, respectively. Whereas figures 6, 7 and 8 show the fitness value of each algorithm execution for the three methods.

Table 4: Genetic Algorithm Results.

NUM	GA_Exec_Time	GA_Fitness
1	15.756929	0.897
2	13.229277	0.336
3	31.985196	0.2696

Table 5: First Fit Heuristic Results.

NUM	FF_Exec_Time	FF_Fitness
1	0.214907	0.814
2	0.477475	0.282
3	0.49348	0.212

In order to further improve the proposed genetic algorithm, we experimented with different parameters and methods used for each step of the algorithm, the details are provided in the table 7.

Table 6: Random Algorithm Results.

NUM	RNDM_Exec_Time	RNDM_Fitness
1	0.017884	0.406
2	0.024984	0.046
3	0.053224	0.0459

Table 7: Methods Testing Details.

Step	Method Tested
Selection	- by rank - tournament - random
Crossover	- one-point - two-points - uniform
Replacement	- offspring replace parents with a weak fitness value - best N individuals (from offspring and parents) - replace parents by offspring

The figures 4 to 6 present the detailed evaluation results from the 100 runs of the algorithm with each dataset presented before. The execution times and fitness values are compared for the three algorithms at hand, which are the genetic algorithm, the first-fit heuristic, and the random solution generator.

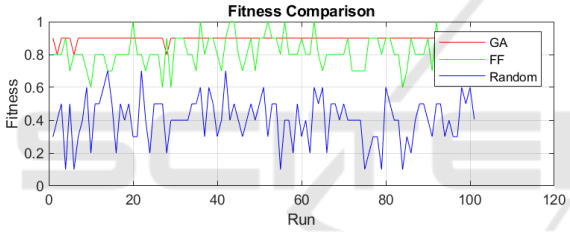

Figure 4: Fitness values for the first set of experiments.

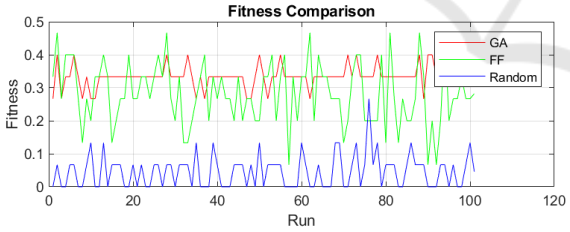

Figure 5: Fitness values for the second set of experiments.

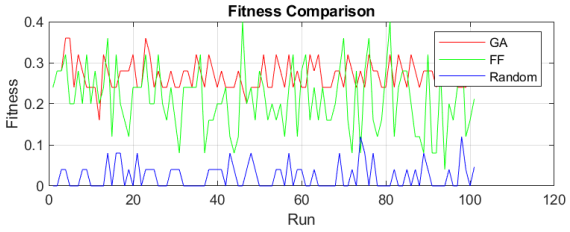

Figure 6: Fitness values for the third set of experiments.

6.3 Results Discussion

While the first fit algorithm initially seems promising due to its ability to yield favorable results within a reasonable time frame, our experiments revealed a significant drawback, where we observed that shuffling the dataset impacts the accuracy of its solutions, making it less suitable for the dynamic nature of the fog service placement problem. In contrast, the genetic algorithm remained unaffected by dataset shuffling, where each placement round introduces new data inputs, and the genetic algorithm emerges as a more robust and versatile solution.

Moreover, the parameters of the genetic algorithm, including the number of generations, mutation rate, and population size, were fixed through iterative experimentation with various values. Each adjustment was methodically tested to isolate its impact on solution quality.

Among the selection methods tested, tournament selection outperformed ranking and random selections. This superiority can be attributed to its effectiveness in exploration during the algorithm's progression. Consequently, it was integrated into the final algorithm, wherein each parent's selection probability is determined by its fitness.

In terms of crossover methods, the uniform crossover demonstrated superior performance compared to one-point and two-point crossovers. Its ability to produce more accurate results led to its selection for the final genetic algorithm.

Regarding replacement strategies, replacing parents with their offspring emerged as the most effective approach. This method facilitated extensive exploration of the solution space, thereby mitigating the risk of getting trapped in local optima. Consequently, it was chosen over alternatives such as replacing parents with better offspring or selecting the N best individuals.

These refined methodologies culminated in the development of our final genetic algorithm, which exhibited commendable performance compared to both random solutions and those generated by the first-fit heuristic.

In our specific scenario, incorporating gateways and cloud data-centers wasn't necessary, as our objective was to allocate all services exclusively to the fog nodes. Therefore, the complexity of managing offloaded services to other components of the infrastructure was not a factor in our analysis.

7 CONCLUSIONS

In this paper, we studied the service placement problem in fog computing by proposing a genetic algorithm that efficiently utilizes fog resources while meeting IoT applications' deadlines to optimize re-

source utilization and maximize the number of accepted services. We evaluated our proposed approach against a random solution and a first-fit heuristic-generated solution by using different datasets. Although the genetic algorithm scored better accuracy than both the random and the first-fit approach in most of the experiments, there is still room for improvement.

Our next research plan is to consider other objectives such as delay, energy consumption, and so on, which makes the model able to quantify the quality of a given solution with more precision, as well as using a fog computing simulator rather than a programming language alone, to be able to simulate a complete fog computing environment.

REFERENCES

Ahmad, U., Mohan, H., and Bakht, H. (2017). Cloud computing - a comprehensive definition. *Journal of Computing and Management Studies*.

Ali, Z. H. (2015). Internet of things (iot): Definitions, challenges and recent research directions. *International Journal of Computer Applications*, 128.

Apat, H. K., Nayak, R., and Sahoo, B. (2023). A comprehensive review on internet of things application placement in fog computing environment. *Internet of Things*, 23:100866.

Bonomi, R., Milito, J., Zhu, J., and Addepalli, S. (2012). Fog computing and its role in the internet of things. In *Proceedings of the first edition of the MCC workshop on Mobile cloud computing*.

Brent, R. P. (1989). Efficient implementation of the first-fit strategy for dynamic storage allocation. *ACM Transactions on Programming Languages and Systems*, 11:388–403.

Canali, C. and Lancellotti, R. (2019). Gasp: Genetic algorithms for service placement in fog computing systems. *Algorithms*, 12:201.

Das, A. K., Sengupta, S., and Bhattacharyya, S. (2018). A group incremental feature selection for classification using rough set theory based genetic algorithm. *Applied Soft Computing*, 65:400–411.

Djemai, P., Stolf, T., Monteil, T., and Pierson, J.-M. (2019). A discrete particle swarm optimization approach for energy-efficient iot services placement over fog infrastructures. In *2019 18th International Symposium on Parallel and Distributed Computing (ISPDC)*.

Dokeroglu, T., Sevinc, E., Kucukyilmaz, T., and Cosar, A. (2019). A survey on new generation metaheuristic algorithms. *Computers and Industrial Engineering*, 137:106040.

Hassan, O., Azizi, S., and Shojafar, M. (2020). Priority, network and energy-aware placement of iot-based application services in fog-cloud environments. *IET Communications*, 14(13):2117–2129.

Jamil, B. et al. (2019). A job scheduling algorithm for delay and performance optimization in fog computing. *Concurrency and Computation: Practice and Experience*, 32(7).

Lin, C.-C., Deng, D.-J., Suwatcharachaitiwong, S., and Li, Y.-S. (2020). Dynamic weighted fog computing device placement using a bat-inspired algorithm with dynamic local search selection. *Mobile Networks and Applications*, 25:1805–1815.

Liu, C., Wang, J., Zhou, L., and Rezaeipanah, A. (2022). Solving the multi-objective problem of iot service placement in fog computing using cuckoo search algorithm. *Neural Processing Letters*, 54(3):1823–1854.

Minh, Q. T., Nguyen, D. T., Le, A. V., Nguyen, H. D., and Truong, A. (2017). Toward service placement on fog computing landscape. In *2017 4th NAFOSTED Conference on Information and Computer Science*, pages 291–296, Hanoi. IEEE.

Natesha, V. and Guddeti, R. M. (2021). Adopting elitism-based genetic algorithm for minimizing multi-objective problems of iot service placement in fog computing environment. *Journal of Network and Computer Applications*, 178:102972.

Salaht, F. A., Desprez, F., and Lebre, A. (2021). An overview of service placement problem in fog and edge computing. *ACM Comput. Surv.*, 53:1–35.

Santoyo-González and Cervelló-Pastor, C. (2018). Latency-aware cost optimization of the service infrastructure placement in 5g networks. *Journal of Network and Computer Applications*, 114:29–37.

Skarlat, M., Nardelli, S., Schulte, M., Borkowski, M., and Leitner, P. (2017). Optimized iot service placement in the fog. *Service Oriented Computing and Applications*, 11(4):427–443.

Xue, Y., Zhu, H., Liang, J., and Słowik, A. (2021). Adaptive crossover operator based multi-objective binary genetic algorithm for feature selection in classification. *Knowledge-Based Systems*, 227:107218.