

## **Strategic application placement in fog computing through multi-objective meta-heuristic algorithm**

Chintansinh Parmar \*

Ankur Goswami †

Jinal Metha §

Jayesh Mevada ‡

*Department of Computer Engineering*

*Sankalchand Patel College of Engineering*

*Sankalchand Patel University*

*Visnagar 384315*

*Gujarat*

*India*

Ochchhav Patel ®

*Department of Computer Engineering*

*LDRP Institute of Technology & Research*

*Kadi Sarva Vishwavidyalaya*

*Gandhinagar 382015*

*Gujarat*

*India*

---

### **Abstract**

Fog computing has emerged as a promising paradigm for meeting the low-latency, high-bandwidth needs of Smart City applications close to data sources. However, existing placement strategies often struggle to jointly optimize timely task completion, efficient resource usage and reduced energy cost. This paper proposes a multi-objective PathFinder Algorithm (PFA) that balances makespan, Fog resource utilization and end-to-end latency in a unified framework. Extensive simulations in iFogSim2 use four baselines, Random, BF-SA, PEAPM and POAPM, evaluated on makespan, power consumption and task guarantee ratio

---

\* E-mail: [chintansinh191@gmail.com](mailto:chintansinh191@gmail.com) (Corresponding Author)

† E-mail: [ajgoswami.ce@gmail.com](mailto:ajgoswami.ce@gmail.com)

§ E-mail: [jinalmehta1314@gmail.com](mailto:jinalmehta1314@gmail.com)

‡ E-mail: [jmmevadace\\_spce@spu.ac.in](mailto:jmmevadace_spce@spu.ac.in)

® E-mail: [ochchhav\\_ce@ldrp.ac.in](mailto:ochchhav_ce@ldrp.ac.in)

(TGR). The results show that the PFA-based method outperforms the compared approaches across varying Fog network sizes, improving power conservation and overall performance.

*Subject Classification (2020):* 68U01, 68U35, 68Q01.

*Keywords:* Smart city, Fog computing, Application placement, Meta-heuristic optimization.

1. Introduction

The rapid evolution of Internet of Things (IoT) technologies has driven the emergence of fog computing [1], [2], [22], a paradigm that brings computation, storage and communication closer to IoT devices at the network edge [3]. This proximity reduces latency, conserves bandwidth and improves processing of the large data volumes generated by smart cities, healthcare and industrial automation. In real-time environments, fog computing still faces scalability, latency, resource-constraint and energy-consumption challenges, together with the Fog Application/Service Placement Problem (FAPP).

FAPP [5], [6] concerns how application services should be assigned to available fog nodes so as to improve systems, shorten response times, control data flow and keep data close to users despite each node’s fixed resource limits. Efficient deployment requires positioning services so that the CPU, memory and storage of each fog node are used to the fullest. Figure 1 shows two applications whose services are placed on four fog nodes. Fog nodes are physical or virtual devices at the network edge able to process, store and transmit data; services (ovals) are the tasks that make up an application; and arrows indicate dependencies between services, which are deployed across nodes according to criteria such as latency, node capacity and network traffic.

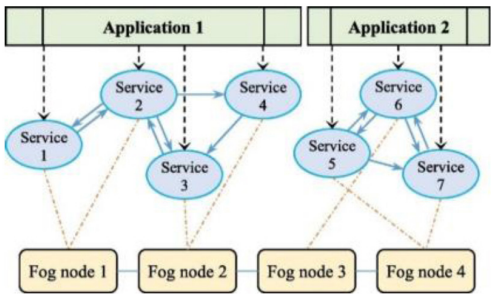


Figure 1

Service distribution across fog nodes for optimal application performance [7]

### 1.1 *Meta-heuristics for FAPP*

Optimization algorithms [8] find the best solution from a set of feasible ones with respect to a criterion. Meta-heuristics [9] are optimization algorithms designed for problems too hard to solve exactly in reasonable time; they escape local optima and explore the search space broadly. Common examples are Genetic Algorithms (GA) [10], Simulated Annealing (SA) [11], Ant Colony Optimization (ACO) [12] and Particle Swarm Optimization (PSO) [13]. For FAPP [5], meta-heuristics can find the best nodes for services while meeting several goals at once: low latency, high throughput and fair distribution of tasks. This study develops a new optimization approach for application placement in fog environments with the following objectives: (i) to study the placement problem and outline a meta-heuristic model that uses multiple optimization criteria; (ii) to design a meta-heuristic algorithm tailored to the multi-objective nature of the problem; and (iii) to validate the algorithm empirically against existing methods using several performance metrics.

## 2. Literature Review

Modern fog computing demands multi-objective optimization (MOO) to manage tasks and distribute resources across distributed systems, balancing opposing goals such as faster response, lower power demand and better resource use. Improved versions of Grey Wolf Optimization (GWO) [15] and PSO [16] address such problems, and nature-inspired algorithms handle multiple targets in unstable, changing digital environments [17]. Hybrid methods are also explored: combining GA and PSO gives good results for fog task scheduling in big-data projects [18], supported by GA-based service placement [10] and by genetic placement that exploits complex network features [20]. Alwabel and Swain [21] place application modules on fog and cloud platforms to meet service deadlines while reducing power use. Fog computing is poised to change how data-driven applications are deployed at the network edge, yet comprehensive solutions to its implementation challenges are still needed; multi-objective optimization with newer meta-heuristics is a promising direction for FAPP.

## 3. Proposed Work

The PathFinder Algorithm (PFA) uses three objectives, makespan, fog utilization and latency, because together they capture the core performance

trade-offs in fog environments. Makespan ensures the whole workload finishes in the shortest time; fog utilization measures how well distributed resources are used, minimizing idle capacity and improving cost-effectiveness; and latency reflects the real-time responsiveness of edge and IoT services.

### 3.1 *PathFinder Algorithm*

PFA iteratively refines a population of candidate solutions, each a mapping of application modules to fog devices, with the aims of (i) minimizing makespan, (ii) maximizing fog device utilization and (iii) minimizing end-to-end latency. A population of random or heuristic assignments is initialized, and each solution is evaluated by the makespan across all modules, the total or average device utilization, and the communication plus processing latency. In each iteration PFA ranks the solutions, identifies one or more best (often non-dominated) solutions, and partially moves every other solution toward them.

#### **Algorithm: Multi-objective PFA for the application placement problem**

**Input:** population size  $N$ , maximum iterations  $T_{\max}$ , coefficients  $\alpha, \beta, \gamma$  ( $\alpha$  and  $\gamma$  balance exploitation and exploration).

**Output:**  $X_{\text{best}}$  and  $F(X_{\text{best}})$ .

1. Initialize  $X(0) = \{X_1(0), \dots, X_N(0)\}$ , where each  $X_i(0)$  randomly assigns every module  $m_j \in M$  to a device  $d_k \in D$ .
2. Evaluate each  $X_i(0)$ :  $F(X_i(0)) = (f_1, f_2, f_3)$  (makespan, utilization, latency).
3. Combine  $f_1, f_2, f_3$  into a single  $\text{fitness}(X_i)$ ; set  $t = 0$ .
4. While  $t < T_{\max}$ : (a) rank  $X(t)$  by the multi-objective criteria; (b) let  $X_{\text{best}}(t)$  be the best solution; (c) update every solution,  $X_i(t+1) = \text{PathFinderUpdate}(X_i(t), X_{\text{best}}(t), \alpha, \beta, \gamma)$ , using Eq. (1), where  $R(t)$  is the random exploration operator; (d)  $t = t + 1$ .
5. End while; obtain the final solution, Eq. (2), and return  $X_{\text{best}}$  and  $F(X_{\text{best}})$ .

$$X_i(t+1) = X_i(t) + \alpha \cdot [X_{\text{best}}(t) - X_i(t)] + \gamma \cdot R(t) \quad (1)$$

$$X_{\text{best}} = \arg \min [\text{fitness}(X)] \quad (2)$$

3.2 Experimental Setup

PFA was implemented in iFogSim2, a toolkit for modeling and simulating fog and edge environments [18]. The number of fog nodes ranged from 20 to 60, reflecting small to moderately sized fog infrastructures [14], [18], and each simulation ran for 100–250 s of simulated time, capturing transient and steady-state behavior [18] (Table 1). IoT-to-fog latency was set to 10–15 ms and fog-to-cloud latency to 140–160 ms to mimic geographical distance and network delay [3], [4], [6]. Service modules had to complete within 350–750 ms, so the placement algorithm had to balance data location against available computing power [6], [19], [21]. IoT devices produced application traffic at one tuple every 3–7 s, keeping the system active and forcing placement and migration decisions [18].

**Table 1**  
System parameters used for the simulation

| Parameter                             | Value      |
|---------------------------------------|------------|
| Simulation environment                | iFogSim2   |
| Number of fog nodes                   | 20–60      |
| Simulation duration                   | 100–250 s  |
| IoT device to fog node latency        | 10–15 ms   |
| Fog node to cloud latency             | 140–160 ms |
| Application service delivery deadline | 350–750 ms |
| Application data receiving interval   | 3–7 s      |

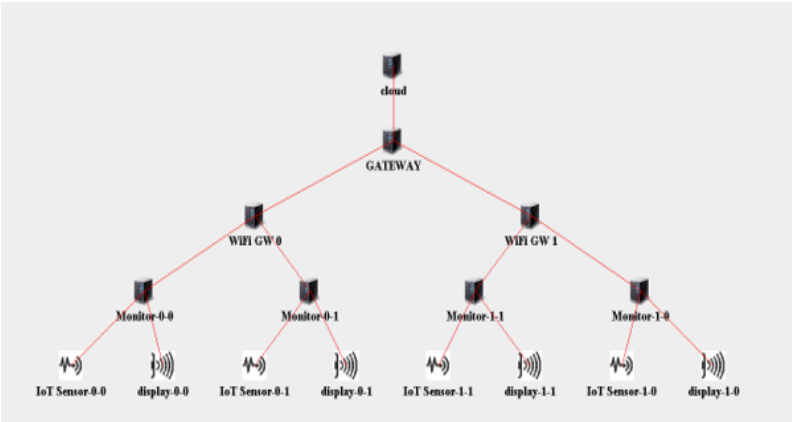


**Figure 2**  
Architecture of the proposed simulation

In the setup of Figure 2, application components (sensors, monitors and displays) are placed to optimize latency and bandwidth usage. Sensors send data to nearby WiFi gateways for initial processing; depending on the latency requirement, data is processed at the gateway or pushed to the cloud for heavier computation. Monitors and displays use the processed data. Distributing processing across network levels improves responsiveness and energy efficiency, and reducing the distance data travels for initial processing lowers the likelihood of bottlenecks.

#### 4. Results and Discussion

Five placement strategies, Random, BF-SA, PEAPM, POAPM and the proposed PFA, were benchmarked on three metrics: makespan, power consumption and task guarantee ratio (TGR), for 20–60 fog nodes.

##### 4.1 Makespan

Figure 3 shows that increasing the number of fog nodes from 20 to 60 lowers the makespan for all methods, since more nodes provide additional distributed computing resources [6], [14], [17]. Performance gaps remain between the methods, and the proposed approach consistently yields the lowest makespan, showing more effective use of fog resources.

##### 4.2 Power Consumption

Power consumption falls for all methods as nodes increase from 20 to 60, reflecting more balanced utilization and lower per-node load [7], [14], [17], [21]. As Figure 4 shows, the proposed algorithm consumes less

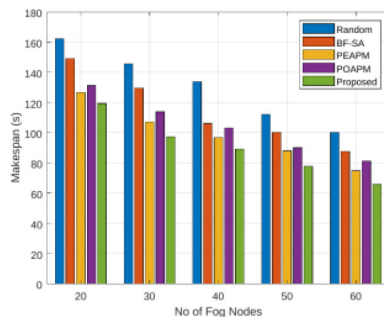


**Figure 3**

**Makespan versus number of fog nodes.**

energy than Random and BF-SA and also saves more than PEAPM and POAPM by reducing device overhead during module placement.

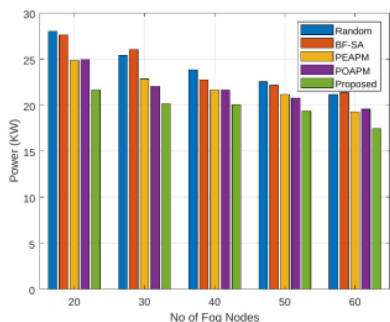


**Figure 4**

**Power (kW) versus number of fog nodes.**

#### 4.3 Task Guarantee Ratio (TGR)

Adding fog nodes raises the TGR of all models, meaning more tasks finish before their deadlines. As shown in Figure 5, the proposed approach keeps schedule performance strongest regardless of the number of nodes.

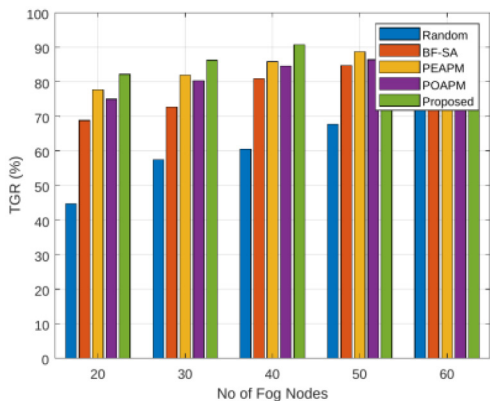


**Figure 5**

**Task guarantee ratio (TGR %) versus number of fog nodes.**

## 5. Conclusion and Future Scope

This work presented a multi-objective PFA for task placement in fog computing for smart city applications. By jointly optimizing makespan,

fog utilization and latency, the approach outperforms four baselines, Random, BF-SA, PEAPM and POAPM, in makespan, power consumption and TGR in iFogSim2 simulations, and it uses additional fog nodes more effectively than competing strategies. The present study assumes stable network connections throughout data transmission and does not consider IoT devices leaving the network or changes in data flow. Future work should adjust placement dynamically to live contextual changes and address the protection of sensitive data in data-intensive smart city systems.

## References

- [1] Z. Ahmadi, M. Haghi Kashani, M. Nikravan, and E. Mahdipour, "Fog-based healthcare systems: A systematic review," *Multimed. Tools Appl.*, vol. 80, no. 30, pp. 36361–36400 (Dec. 2021), doi: 10.1007/s11042-021-11227-x.
- [2] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli, "Fog computing and its role in the internet of things," in *MCC'12 – Proc. 1st ACM Mobile Cloud Computing Workshop* (2012), pp. 13–15, doi: 10.1145/2342509.2342513.
- [3] P. Hu, S. Dhelim, H. Ning, and T. Qiu, "Survey on fog computing: Architecture, key technologies, applications and open issues," *J. Netw. Comput. Appl.*, vol. 98, pp. 27–42 (2017), doi: 10.1016/j.jnca.2017.09.002.
- [4] P. P. Ray, "A survey of IoT cloud platforms," *Future Comput. Inform. J.*, vol. 1, no. 1–2, pp. 35–46 (2016).
- [5] H. K. Apat, R. Nayak, and B. Sahoo, "A comprehensive review on Internet of Things application placement in Fog computing environment," *Internet Things*, p. 100866 (2023).
- [6] R. Eyckerman, S. Mercelis, J. Marquez-Barja, and P. Hellinckx, "Requirements for distributed task placement in the fog," *Internet Things*, vol. 12, p. 100237 (2020).
- [7] B. Wu, X. Lv, W. Deyah Shamsi, and E. Gholami Dizicheh, "Optimal deploying IoT services on the fog computing: A metaheuristic-based multi-objective approach," *J. King Saud Univ. – Comput. Inf. Sci.*, vol. 34, no. 10, Part B, pp. 10010–10027 (Nov. 2022), doi: 10.1016/j.jksuci.2022.10.002.
- [8] M. N. A. Wahab, S. Nefti-Meziani, and A. Atyabi, "A comprehensive review of swarm optimization algorithms," *PLOS ONE*, vol. 10, no. 5, p. e0122827 (May 2015), doi: 10.1371/journal.pone.0122827.

- [9] S. Chattopadhyay, A. Marik, and R. Pramanik, "A brief overview of physics-inspired metaheuristic optimization techniques," *arXiv pre-print, arXiv:2201.12810* (2022), doi: 10.48550/arXiv.2201.12810.
- [10] C. Canali and R. Lancellotti, "GASP: Genetic algorithms for service placement in fog computing systems," *Algorithms*, vol. 12, no. 10 (2019), doi: 10.3390/a12100201.
- [11] R. A. Rutenbar, "Simulated annealing algorithms: An overview," *IEEE Circuits Devices Mag.*, vol. 5, no. 1, pp. 19–26 (1989).
- [12] J. del Sagrado, I. M. del Águila, and F. J. Orellana, "Multi-objective ant colony optimization for requirements selection," *Empir. Softw. Eng.*, vol. 20, no. 3, 577–610 (Jun. 2015), doi: 10.1007/s10664-013-9287-3.
- [13] T. M. Shami, A. A. El-Saleh, M. Alswaitti, Q. Al-Tashi, M. A. Summakieh, and S. Mirjalili, "Particle swarm optimization: A comprehensive survey," *IEEE Access*, vol. 10, pp. 10031–10061 (2022), doi: 10.1109/ACCESS.2022.3142859.
- [14] S. Smolka and Z. Á. Mann, "Evaluation of fog application placement algorithms: A survey," *Computing*, vol. 104, no. 6, pp. 1397–1423 (2022), doi: 10.1007/s00607-021-01031-8.
- [15] M. H. Nadimi-Shahraki, S. Taghian, and S. Mirjalili, "An improved grey wolf optimizer for solving engineering problems," *Expert Syst. Appl.*, vol. 166, p. 113917 (2021).
- [16] W. Abbes, Z. Kechaou, A. Hussain, A. M. Qahtani, O. Almutiry, H. Dhahri, and A. M. Alimi, "An enhanced binary particle swarm optimization (E-BPSO) algorithm for service placement in hybrid cloud platforms," *Neural Comput. Appl.*, vol. 35, no. 2, pp. 1343–1361 (2023).
- [17] F. A. Saif, R. Latip, Z. M. Hanapi, and K. Shafinah, "Multi-objective grey wolf optimizer algorithm for task scheduling in cloud-fog computing," *IEEE Access*, vol. 11, pp. 20635–20646 (2023), doi: 10.1109/ACCESS.2023.3241240.
- [18] M. Saad, R. I. Qureshi, and A. U. Rehman, "Task scheduling in fog computing: Parameters, simulators and open challenges," in *2023 Global Conf. Wireless and Optical Technologies (GCWOT)*, pp. 1–6 (Jan. 2023), doi: 10.1109/GCWOT57803.2023.10064652.
- [19] J. L. Herrera, J. Galán-Jiménez, J. Berrocal, and J. M. Murillo, "Optimizing the response time in SDN-fog environments for time-strict IoT

- applications," *IEEE Internet Things J.*, vol. 8, no. 23, pp. 17172–17185 (2021).
- [20] M. Azimzadeh, A. Rezaee, S. J. Jassbi, and M. Esnaashari, "Placement of IoT services in fog environment based on complex network features: A genetic-based approach," *Clust. Comput.*, vol. 25, no. 5, pp. 3423–3445 (Oct. 2022), doi: 10.1007/s10586-022-03571-w.
- [21] A. Alwabel and C. K. Swain, "Deadline and energy-aware application module placement in fog-cloud systems," *IEEE Access*, vol. 12, pp. 5284–5294 (2024), doi: 10.1109/ACCESS.2024.3350171.
- [22] H. Jain and M. Saxena, "A review on edge computing with data analysis and IoT techniques," *J. Inf. Optim. Sci.*, vol. 45, no. 4, pp. 947–957 (2024), doi: 10.47974/JIOS-1618.

*Received June, 2026*