The Journal of Supercomputing (2024) 80:17790–17822
https://doi.org/10.1007/s11227-024-06151-4

Service placement in fog–cloud computing environments: a comprehensive literature review

Fatemeh Sarkohaki[1] · **Mohsen Sharifi**[1]

Accepted: 14 April 2024 / Published online: 2 May 2024

Abstract
With the rapid expansion of the Internet of Things and the surge in the volume of data exchanged in it, cloud computing became more significant. To face the challenges of the cloud, the idea of fog computing was formed. The heterogeneity of nodes, distribution, and limitation of their resources in fog computing in turn led to the formation of the service placement problem. In service placement, we are looking for the mapping of the requested services to the available nodes so that a set of Quality-of-Service objectives are satisfied. Since the problem is NP-hard, various methods have been proposed to solve it, each of which has its advantages and shortcomings. In this survey paper, while reviewing the most prominent state-of-the-art service placement methods by presenting a taxonomy based on their optimization strategy, the advantages, disadvantages, and applications of each category of methods are discussed. Consequently, recommendations for future works are presented.

Keywords Internet of things · Cloud computing · Fog computing · Service placement · VANET

1 Introduction

With the ever-increasing progress of computing, wireless infrastructure, and technologies, the Internet of Things (IoT) model has emerged and caused a substantial digital transformation in everyday life. In this model, smart objects can include mobile phones, vehicles, home appliances, sensors, actuators, or any other embedded device. These devices are connected through modern communication network infrastructures to exchange data related to the real and virtual worlds. Therefore, IoT expands human-to-human communication to human-to-machine and

✉ Mohsen Sharifi
msharifi@iust.ac.ir

[1] School of Computer Engineering, Iran University of Science and Technology, Tehran, Iran

machine-to-machine communications [1, 2]. This technology is expected to pave the way for a large number of applications in areas such as security [3–6], environment [7–12], healthcare [13–17], industry [18–20], sports [21, 22], e-commerce [23, 24], agriculture [25–27], and transportation [28–31], which will lead to a dramatic improvement in the quality of daily human life.

While most people know the IoT with large-scale sensors, health monitors and self-driving cars, the application of this technology is much more than these [32]. The automobile and transportation sectors are one of the major beneficiaries of IoT. Since the sensors and actuators in the IoT provide the possibility of monitoring and controlling the surrounding environment, so smart vehicles can be connected to each other using these sensors and actuators and create a special VANET. In such a network, the exchange of information between vehicles and roadside equipment informs drivers of dangerous conditions and impending accidents, preventing probable road accidents [33].

Meanwhile, with the expansion of the use of IoT, a large amount of data is produced, most of which has a short lifespan and mainly requires fast processing in order to provide desired services. Service means a repetitive activity with a specific result that is independent and separate and may be composed of other services [34]. For example, in the VANET and Unmanned Aerial Vehicles (UAV), there are intensive and delay-sensitive services such as choosing the shortest route or notifying road accidents [35–37]. Cloud computing centers were candidate to process this huge amount of data, wherein a set of configurable computing resources (for example, networks, servers, storage resources, and services) are provided. By using cloud computing technology, individuals and organizations can pay only the costs related to their cloud resources and store information and process their services in the cloud without providing the necessary infrastructure facilities [33].

Many IoT applications have real-time data processing requirement but experience high latency when interacting with centralized cloud servers [38]. This problem is due to the large distance between IoT devices and cloud servers. In addition, the massive growth of IoT devices has resulted in the cloud being faced with a huge amount of data to process. Therefore, the current cloud infrastructure suffers from the problem of reduced bandwidth, increased response time, high latency and network congestion [39]. Hence, as shown in Fig. 1 [40], a new 3-layer model with a fog computing middle layer has evolved to overcome these limitations. The fog computing layer, similar to the cloud computing layer, provides end users with data, computing resources, storage resources, etc., albeit on a smaller scale [41]. Any device such as a controller, smart gateway, switch, router, or an embedded server with processing, network and storage capabilities can be used as a fog node.

Fog computing is a processing paradigm that creates a platform for distributed and latency-aware applications. This paradigm is very critical for very dynamic networks such as VANET [42, 43]. Due to having processing capacities close to the user, fog computing is considered a promising model for the placement of requested services by the users of the IoT and VANET [44, 45]. However, since the end devices are heterogeneous in these networks and their requested services are highly diverse, and these services are mainly sensitive to delay and response time, so to reduce the delay and send quick responses to users requires

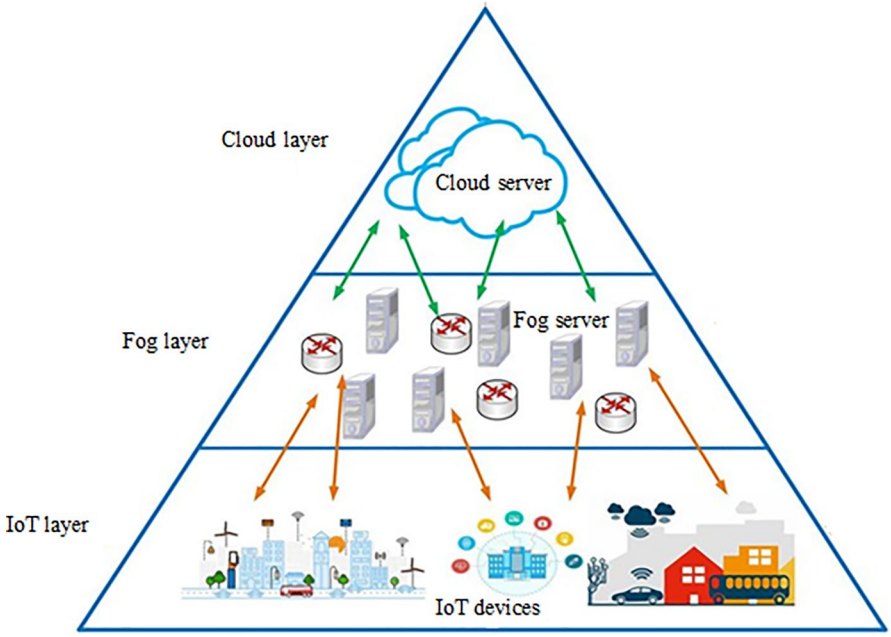

Fig. 1 Three-layer architecture of fog–cloud computing [40]

a mechanism to coordinate and place the requested services in the available resources in fog computing in an optimal way [46].

Service placement itself is a challenging process. There are two main challenges in this field. First, the user's equipment should choose which services are in which order and with what prioritization to load and send to the cloud computing infrastructure in a way that is cost-effective for him or her. The second challenge is deciding which computing nodes to assign each of these services. The latter one is more complex, due to the heterogeneity and distribution of fog nodes and hence, more research has been done on it.

So far, many methods have been presented to solve the problem of placing IoT services in the fog–cloud computing environment. Based on the optimization strategies, these methods can be classified into seven major categories including exact solutions, approximate solutions, heuristic and meta-heuristic-based, machine learning-based, game theory-based, neural networks-based algorithms and other methods. In this paper, the newest and most prominent methods of each of these categories are reviewed and the advantages, disadvantages, and applications of each method are examined. The most important contributions of this paper are as follows:

- Providing a comprehensive overview of the service placement problem and the available methods to solve it.
- Presenting a taxonomy of service placement methods into seven major categories based on their optimization strategy.

Table 1 List of acronyms

Notation	Description
ACO	Ant colony optimization
AIoT	Artificial intelligence of things
DAG	Directed acyclic graphs
DDQN	Double deep Q-network
DRL	Deep reinforcement learning
GA	Genetic algorithm
GAT	Graph attention networks
GCN	Graph convolutional network
GNN	Graph neural networks
IIoT	Industrial internet of things
ILP	Integer linear programming
INLP	Integer nonlinear programming
IoT	Internet of things
LLC	Limited look-ahead control
MDP	Markov decision process
MEC	Mobile edge computing
MINLP	Mixed integer nonlinear programming
PSO	Particle swarm optimization
QoS	Quality-of-service
RL	Reinforcement learning
SA	Simulated annealing
UAV	Unmanned aerial vehicles
VANET	Vehicular Ad hoc network
VM	Virtual machine

- Presenting a wide-ranging comparison of existing service placement methods in terms of application area, optimization criteria, type of virtualization, optimization approach, etc., taking into account the advantages and disadvantages of each method.
- Providing discussions, conclusions and prospects for future works in the field of service placement in fog computing environments.

Table 1 represents the list of acronyms used in this paper. Figure 2 illustrates the process of the current study on service placement methods. The remainder of this paper is as follows; Sect. 2 describes the service placement problem in more detail. In Sect. 3, the service placement methods are reviewed and organized into seven major categories. In Sect. 4, a comprehensive discussion of the studied methods is presented. Section 5 recommends directions for future works. Finally, the conclusion of the current research is presented in Sect. 6.

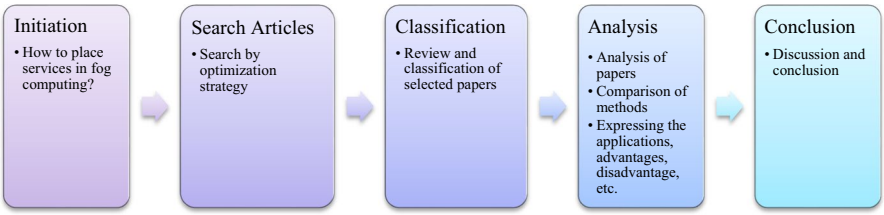

Fig. 2 Process of the current study

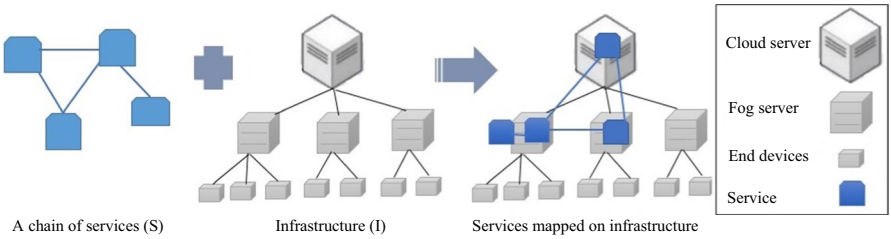

Fig. 3 Service placement problem

2 The service placement problem in fog computing

The problem of service placement in fog computing can be defined as follows. Suppose that set I includes an infrastructure of cloud servers, fog nodes and end devices, such that this set contains information including capacity (CPU, RAM, storage, etc.) and links between nodes (information such as latency and bandwidth of each link) for cloud servers, fog servers and end devices. Also, suppose that S is the set of services created by user applications, which contains information about service components of applications and connectors between them. A connection is a relationship between two components of a service that allows them to call each other and exchange data. Then, the goal of service placement in fog computing is to find the mapping of services (set S) on infrastructure nodes (set I), so that a set R of requirements, including the resources required by the services (in terms of CPU, RAM, storage, etc.) as well as connections (e.g., bandwidth) is met (Fig. 3). If there are several locations in the existing infrastructure that fulfill all the service requirements, then among these locations, according to a set O of objectives, the best location should be selected. These goals are different according to the type of services and application and can include goals such as minimizing delay, reducing response time or overall energy consumption. There may be other requirements in the placement of the services. For example, placing some services in certain infrastructure nodes may be prohibited for security reasons.

With the expansion of the IoT and considering the variety of services and applications available, finding a solution to choose a suitable place from the existing infrastructure to respond to the services requested by users in a way that meets the requirements of the services and preferably optimizes the given goals has become

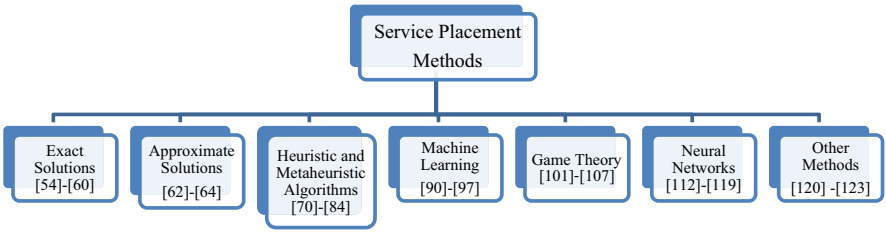

Fig. 4 Classification of investigated service placement methods

very essential. This challenge has high complexity due to the hierarchical, distributed, and heterogeneous structure of fog computing and also due to the different characteristics of IoT devices and user expectations that vary from one user to another and from one program to another. The service placement in fog is an NP-hard problem. This is proven by Teng et al. in [47]. Their proof is based on the reduction from the knapsack problem, which is NP-hard. So far, many methods have been introduced to place the service in the fog, and each of these methods has tried to optimize a number of goals. In the continuation of this paper, a number of the most prominent state-of-the-art methods of service placement in the fog have been studied.

3 Service placement literature review

Due to the huge amount of data generated by devices in the Internet of Things network and the increase in the number of their services, a large number of researchers have provided numerous solutions to the problem of placing the services received from the applications in these networks [44]. For instance, with the expansion of VANET technology and the need for real-time response to service requests created in these networks, the need to use the fog environment for these networks has also increased, and in line with that, new methods for optimum placement of service in these networks are being introduced. The purpose of service placement in the fog environment is to assign services to fog nodes in such a way that one or more specific criteria are optimized [48]. The most important of these criteria include energy consumption, priority, delay, throughput and cost. Service placement techniques try to maximize (or minimize) the values of these criteria based on their proposed design and system performance [49]. In this paper, while introducing, reviewing and comparing the latest available methods of service placement in IoT, MEC[1] and VANET networks, a taxonomy of these methods based on the optimization strategies used by them is introduced. The presented taxonomy includes seven categories of service placement methods: (1) exact solutions, (2) approximate solutions, (3) heuristic and meta-heuristic-based algorithms, (4) machine learning-based techniques

[1] Mobile Edge Computing.

(5) game theory-based (6) neural networks-based techniques and (7) other methods. Figure 4 illustrates this taxonomy.

3.1 Service placement using exact solutions

An exact solution is often calculated using a mathematical solver or by performing exhaustive research (counting all solutions). The use of mathematical programming is a solution that is often used to solve optimization problems by formulating them along with constraints and objective functions to solve complex problems such as service placement. In these solutions, then, the solution range of the objective function is explored with the main goal of maximizing or minimizing its value, guaranteeing the return of the optimal solution. Nevertheless, it is important to pay attention to the fact that the examination of the entire solution space suffers from a very high time complexity, and these methods are not very suitable for solving the complex problem of service placement in the fog environment [50]. Different types of mathematical programming models such as linear programming [51], nonlinear programming [52] and mixed linear programming [53] have been studied in fog calculations, some of them are introduced below.

In [54], a service placement architecture based on Integer Linear Programming (ILP) is introduced for IoT applications. This architecture aims to minimize communication delays by continuously adapting services and migrating them based on network conditions and user status. While successful in reducing hops and service migration, it does not consider service priority or energy consumption. Tinini et al. [55] introduced an ILP model in a hybrid architecture called Cloud/Fog RAN to support 5G traffic in mobile networks, effectively reducing energy consumption however potentially reducing network coverage. Gong's method [56], which is based on ILP model and deployment of virtual machines, focuses on minimizing access delay for mobile users' requests, improving edge server efficiency, and service placement optimization. However, it only targets delay reduction without considering other optimization criteria.

Kim et al. [57] designed a fog computing infrastructure called Fog Portal, which optimizes service placement based on user participation. Their approach transforms the service placement problem into a mixed integer nonlinear programming (MINLP) problem and aims to minimize energy consumption, yet it overlooks resource usage cost and service delay. Yala et al. [58] proposed a framework based on Mobile Edge Computing architecture for placing services in the edge network, focusing on achieving the minimum delay in service placement in fog nodes. This scheme uses an optimal placement algorithm based on counting. It introduces an almost optimal placement algorithm based on divide and conquer to achieve minimum data traffic by distributing copies of virtual machines of applications in the edge network. Daneshfar et al. [59], in addition to proposing a fog infrastructure, formulated the service placement problem as an ILP model. Their research aimed to minimize the total cost of placing services on fog nodes, considering service and server costs, user service time budget, and resource availability. Their approach ensures cost-effective service placement while meeting the availability criterion for

each user. Donassolo et al. [60] tackled the placement of microservices produced by IoT applications as an ILP problem and used a divide-and-conquer strategy to minimize costs while meeting resource and network needs.

Overall, the research efforts highlighted above underscore exact solutions, mostly leveraging ILP models, for optimizing service placement in fog and edge computing environments. While these approaches demonstrate improvements in reducing communication delays, energy consumption, and total cost, some limitations exist, such as overlooking certain optimization criteria and failing to consider factors like service priority and energy consumption of fog devices.

3.2 Service placement using approximate solutions

Approximate algorithms are efficient algorithms that provide the possibility of calculating sub-optimal solutions of NP-hard optimization problems. These algorithms provide sub-optimal solutions along with a coefficient to calculate the degree of approximation of the real solution. Also, approximate algorithms have the ability to guarantee the existence of their solution within the stated error range [61]. In the following, a number of approximate solutions for service placement have been introduced.

Yu et al. [62] addressed the service placement problem in IoT applications, proving the NP-hardness of both single and multi-application scenarios and proposing a stochastic algorithm for non-parallelizable multi-applications. They demonstrated that parallelization can lead to a polynomial-time solution. Ouyang and Zhou [63] focused on optimizing service placement in mobile edge networks under long-term cost constraints, utilizing the Lyapunov optimization function and a distributed approximation scheme based on Markov approximation to address the unpredictability of user mobility. Their method, evaluated using the ONE simulator, demonstrates improvements in average delay and service migration in 5G application scenarios. Ning et al. [64] aimed to maximize system capabilities in terms of server storage and service execution delay by formulating the service placement problem in the edge network. They utilized the Lyapunov optimization and stochastic algorithms based on average approximation to enable dynamic service placement, along with a distributed Markov approximation algorithm. Their evaluations demonstrated improvements in service migration rate and algorithm execution time, particularly for data and tasks less than 60 MB.

Generally speaking, the approximate service placement methods are mainly based on the Markov approximation algorithm, address the NP-hardness challenges, and achieve improvements in parameters like average delay, service migration, and algorithm execution time in fog computing and MEC, although they do not guarantee an optimal solution.

3.3 Service placement using heuristic and meta-heuristic algorithms

The dynamic structure of the fog nodes makes the problem of service placement very complicated in terms of calculations and the detailed analysis of the entire

solution space is impractical. Also, using approximate solutions is not very effective. Therefore, heuristic and meta-heuristic approaches are often investigated for service placement problem. Heuristic approaches are a set of rules and methods that simplify the discovery of practical solutions for computationally complex problems and provide a means to obtain acceptable solutions in a reasonable time. Of course, these methods do not provide any guarantee for optimal performance. Meta-heuristic methods are generally inspired by nature [65–69]. Unlike heuristic approaches that are prone to the problem they do not guarantee to provide a solution, meta-heuristic methods try to improve the result in a reasonable time through an iterative search process for better solutions while trying to avoid getting stuck in local optima. In the following, a number of researches conducted for service placement using heuristic and meta-heuristic algorithms are introduced.

Hoseiny et al. [70] introduced a heuristic approach based on network priority and energy consumption for fog–cloud computing. They used a hybrid method based on task prioritization and Genetic Algorithm (GA) to find suitable computing nodes. These researchers considered IoT applications with a maximum of two services to be placed in fog and cloud. However, they did not consider criteria such as resource availability and service cost in the fog environment. Sarrafzade et al. [71] presented a GA-based approach with a penalty-based method to reduce delay and placement time and consider the proximity of application programs to users. The authors have evaluated their method in terms of delay criteria, network efficiency, energy consumption and cost. Maia et al. [72] addressed load distribution by proposing a service placement and dynamic load distribution strategy using prediction and Limited Look-ahead Control (LLC). In this work, service placement and load balancing are jointly formulated as a multi-objective performance criteria optimization problem and the GA is used to solve the formulated problem.

Khosroabadi et al. [73] presented a heuristic algorithm that uses clustering of fog devices and prioritization of requirement-sensitive services. The method has been evaluated in the iFogSim simulation environment using real data obtained from a test platform in terms of response time, delay and energy consumption criteria. Eyckerman et al. [74] introduced a service placement algorithm in fog for smart vehicle applications using distributed Ant Colony Optimization (ACO) algorithm. While the computational complexity is low, the use of resources and devices is maximized. One of the main disadvantages of [74] is that the scholars evaluated their method only in terms of execution time. Souza et al. [75] proposed algorithms for the placement of distributed services in the fog–cloud environment using service atomization and placement strategies such as First-Fit, Best-Fit, and Best-Fit with queue. However, energy consumption and cost are not considered in this multi-level fog computing environment. Kumar Apat et al. [76] compared deterministic and non-deterministic approaches and presented a meta-heuristic technique based on GA for multi-objective service placement, focusing on energy consumption, delay, and cost. Of course, this work suffers from a weakness in the detailed description of the proposed approach of its authors.

EPMOSO is introduced by Rong Ma [77] to place services in edge servers utilizing GA and Particle Swarm Optimization (PSO). EPMOSO also leverages a weighting method called Simple Additive Weighting (SAW) to determine the

most balanced location of the edge server. Hu et al. [78] combined GA and Social Spider Optimization to manage data density and reduce VM migration in their energy-aware service placement approach. Natesha and Guddeti [79] formulated the service placement problem in the fog computing environment as a multi-objective optimization problem to minimize cost, energy consumption, and service time for advanced services. They introduced a two-tier fog framework using Docker and containers, along with an elitism-based variant of GA. The most significant drawback of this method is its high time complexity. Natesha and Guddeti [80] extended their previous work using the combination of the elitism-based GA and PSO to minimize service costs and ensure the service quality of Industrial IoT applications.

Guerrero et al. [81] studied three evolutionary algorithms, including single-objective GA with weighted sum transformation (WSGA), non-dominated sorting genetic algorithm-II (NSGA-II) and multi-objective evolutionary algorithm based on decomposition (MOEA/D) for service placement in a fog environment. Although the implementations and evaluations done in [81] have shown that these methods cause the diversity of the solution space, they also increase the execution time. In [82], Shahryari et al. presented a task offloading scheme that optimizes offloading, placement in the appropriate fog node, and allocation of computing resources decisions. Their scheme is based on a combination of GA and particle swarm optimization for the trade-off between work completion time and energy consumption. The simulation results demonstrated the high convergence rate and the reduction in energy consumption in comparison with other compared methods, although the time complexity was high.

In 2024, Apat et al. [83] presented hybrid methods based on a combination of GA and simulated annealing (GA-SA) and a combination of GA and PSO (GA-PSO) to optimize energy, cost, and makespan while deploying IoT services in fog–cloud computing environments. Recently, Azizi et al. [84] devised a joint load distribution and service placement method that tries to minimize delay and cost. This heuristic method assigns services to fog nodes based on sorting the nodes considering their request rate and prioritizing services with their requested load.

The literature review has revealed that in the field of service placement using heuristic and meta-heuristic methods, GA and PSO algorithms are leveraged the most by scholars. Moreover, while determining a reasonable initial location is among the main challenges of the methods in this category, the state-of-the-art solutions emphasize optimizing various criteria such as energy consumption, delay, cost, and service time.

3.4 Service placement using machine learning methods

The next solution used by scholars to solve the service placement problem is the leverage of machine learning techniques [85, 86]. Among these techniques, Deep Reinforcement Learning (DRL) is the most widely used method that has been used for service placement in fog computing [87–89].

Zhan et al. [90] investigated the computational offloading scheduling problem in the automotive edge computing scenario. They leveraged Markov decision processes and deep reinforcement learning (DRL) to minimize cost, delay and energy consumption. The Reinforcement Learning (RL) used in their method was based on the advanced proximal policy optimization algorithm and a parameter-shared network architecture was used with a convolutional neural network to approximate both policy and value functions. Talpur [91] used a dynamic RL framework to optimize edge resource utilization and service delay in service deployment. This work also considers the mobility of the vehicles, the variety of demands and the dynamics of requests for different types of services. Lv et al. [92] improved task offloading for autonomous vehicles using DRL and Markov decision processes to consider task priorities and deadlines comprehensively.

The combination of DRL and ILP is another approach that researchers used in [93] to provide an algorithm for automatic service placement at the edge of the virtual network. This method provides the possibility of assigning automated driving services, especially self-driving cars, to shared resources on edge servers, considering constraints such as computing loads, network edge infrastructure, and deployment cost. Another study by Zhou et al. [94] focused on offloading tasks in VANET networks to fog environments, employing an optimization algorithm based on the Convex-Concave Procedure (CCP) to determine the requirements of the servers and a service placement mechanism based on adaptive learning to minimize network delay.

Furthermore, Nsouli et al. [95] introduced a service mesh architecture utilizing Kubernetes-based clustering and reinforcement learning to manage communication between microservices in vehicular fog infrastructure. The authors evaluated their method only in terms of the response time, although the results demonstrated the good performance of this method in terms of this single criterion. Wei et al. [96] formulated a task offloading and transmission power problem in inter-vehicle networks, addressing privacy concerns and task offloading priorities using a privacy-aware multi-agent DRL approach. Their method allows the offloading process to be carried out in such a way that each vehicle reaches its Nash equilibrium without losing the offloading priority. The evaluation of this method has been done using the real-world dataset in terms of cost criteria. Recently, Sharma and Thangaraj [97] introduced a novel DRL-based approach to minimize energy consumption and service execution time. Their method, called DDQN-PER, utilizes double deep Q-Network and prioritized experience replay to learn the optimal placement policy.

This subsection explored various state-of-the-art machine learning-based approaches for service placement in different edge and fog computing scenarios. Each showcased study improved performance in terms of cost, delay, energy consumption, resource utilization, and operating system efficiency through simulations and real-world dataset evaluations. These findings demonstrate the efficacy of machine learning techniques in enhancing service placement and scheduling in fog–cloud computing environments, even though the high time complexity of the agents' learning phase can be considered a weakness of these methods.

3.5 Service placement using game theory

Game theory studies mathematical models of strategic interactions between rational agents. Traditional methods of game theory deal with two-player zero-sum games in which the gains or losses of each participant are exactly balanced by the losses of the other participants [98]. Since game theory is mainly used as a modeling approach in the mathematical social sciences, it is another technique that has been used by researchers to solve the service placement problem in fog computing [99, 100].

Kayal and liebeherr [101] presented a game-theoretic approximation method for distributed service deployment, which was inspired by an iterative hybrid auction method. The main goal of their method was to optimize energy consumption and communication cost. The authors compared their method with heuristic methods only with numerical examples and did not provide a simulation or implementation of their algorithm. Sharma and Thangaraj [102] introduced a game theory-based service placement method based on decentralized matching in order to deploy IoT applications in cloud servers, improving energy consumption and run time. While this method is decentralized and avoids a centralized point of failure, the mobility of things is not taken into account. In [103], Shi et al. combined DRL and mean field game for task placement in edge computing, showing effective decision-making and reduced average delay.

Fairness-aware game theory is another method that has been used for service placement by Aloqaily et al. [104]. In this method, a cooperative distributed game model has been introduced to manage the placement of services in vehicular cloud computations. In simulations, this method has been evaluated in terms of delay and the number of executed services. Zafari et al. in [105] suggested that for service placement and resource-sharing, services first allocate available resources to their applications and share the remaining resources with other providers' applications. For this purpose, they introduced two game theory Pareto optimal allocation and Polyandrous–Polygamous matching based on Pareto optimal allocation algorithms. This method improves players' productivity and increases user satisfaction.

In [106], Xiao et al. used a method called heat-aware task offloading using game theory to offload requested tasks of vehicles. By heat, they meant traffic jams, and their goal was to accommodate the requests of vehicle users while driving in crowded "hot" places. The results have shown that this method has less delay and energy consumption compared to other evaluated methods. In 2022, Shabir et al. [107] a non-cooperative, distributed framework for task offloading, demonstrating efficient resource utilization in vehicle-generated tasks. In a non-cooperative game, nodes independently decide to offload tasks in a fully distributed manner, and no external entity is available to enforce agreements between vehicle nodes. The method was evaluated using Manhattan road traffic data, showing improvements in delay and cost criteria.

To sum up, the research efforts highlighted above utilize game theory techniques, mostly in combination with other methods, for optimizing service placement in fog computing. While these methods demonstrate improvements in reducing criteria such as delay and energy, they mainly suffer from high time complexity.

3.6 Service placement using neural networks

Another technique used by researchers to solve the service placement problem is neural networks. An artificial neural network is a collection of nerve cells or neurons that are connected in a specific architecture to solve specific problems and each of them performs simple calculations [108]. Graph Neural Networks (GNN) and Graph Attention Networks (GAT) are among the most recent instances of neural networks that are used in solving the service placement problem [109–111].

Zhong and He [112] proposed a three-layer cloud-edge-service architecture for offloading tasks in MEC with a combination of DRL and a modified GAT. Graph attention networks are neural network-based architectures that work on data with a graph structure. The simulation results showed improved utility compared to other methods, whereas important metrics such as energy consumption and computational complexity were not considered. Wu et al. in [113] introduced a method based on GNN and reinforcement learning for fine-grained task offloading in mobile network applications. This method uses a preprocessing network with scalable GNN capability for the efficient processing of Directed Acyclic Graphs (DAG). In this method, the scheduling is done through training based on the policy gradient algorithm under the random entry of applications. Although the simulation results indicated reduced task completion time and energy consumption, no analysis of its time complexity was provided.

Eyckerman et al. [114] introduced a method using multi-objective reinforcement learning and a trained neural network for service placement. The method reduces resource consumption and is applicable to scenarios with limited resources, whereas the time complexity was not mentioned. In [115], Zhang et al. presented an end-to-end offloading model using a Deep Graph Matching Algorithm (DGMA) based on graph neural networks. In this mechanism, two GNNs are used to place the service graph and the network graph, and the training process is performed based on the constructed training data. While the method showed a reduction in task delay and execution time, it did not consider essential network performance metrics. In 2022, He et al. [116] introduced an offloading method for services in inter-vehicle networks, based on deep deterministic policy gradient combined with GAT and operation branching, in which GAT limits the offloading destinations to the neighborhood and the operation branching finds the coordinates of the various branches. The evaluation results demonstrated good performance for small systems but not large and compact ones.

Tang et al. [117] considered the problem of offloading "dependent" tasks. In their proposed method, the decisions are modeled as a Markov Decision Process (MDP) to minimize the transfer cost and calculation cost. Furthermore, an algorithm based on DRL using a Graph Convolutional Network (GCN) has been used to show the MDP state space and accelerate decision-making in edge computing. While the experiments in the real-world environment showed a reduction in offloading cost for interdependent tasks, the priority of tasks has not been paid attention to. Sun et al. [118] presented a task offloading method based on the GNN and graph reinforcement learning, showing a reduction in task offloading delay, with increased computational complexity. In 2024, Liu

[119] leveraged the Takagi–Sugeno fuzzy neural network to propose a method for task offloading in the Internet of Vehicles, which aims to minimize time and energy consumption.

Overall, there are several state-of-the-art neural network-based methods for service placement and task offloading in MEC and fog computing environments. Each method demonstrates improvements in specific performance metrics, including delay and energy consumption, with some limitations, such as the lack of consideration for time complexity or suitability for certain system sizes.

3.7 Service placement using other methods

Other strategies have been introduced to solve the problem of service placement in the fog, which are not included in the previous six categories. For instance, in 2022, Tong et al. [120] presented a drone-equipped multi-scale collaborative fog computing system, which involves using mobile drones to provide communication and computing services for ground users. They formulated the communication between drones, task offloading, transmission power, computing resource allocation, and drone location optimization as a mixed INLP problem, solving it by a combination of generalized decomposition and convex approximation. The method was evaluated in terms of delay, energy consumption, and efficiency. Sarkar et al. [121] introduced a method for the dynamic placement of tasks considering deadlines, called Deadline-aware Dynamic Task Placement (DDTP). The method treated the fog computing environment as several clusters, employing a strategy based on task completion deadlines and incorporating an approach for migrating unexecuted tasks to other suitable fog nodes. The Bully algorithm was utilized to address the challenge of the single breaking point of the fog controller node. The method's evaluation considered the delay of tasks in the queue, overall completion time, reliability, and throughput.

Furthermore, Ayoubi et al. [122] introduced an independent IoT service placement method that operates based on a control cycle consisting of monitoring, analysis, decision-making, and execution stages. This method monitors available resources and the status of application services, prioritizes requested services according to deadlines, and applies the Pareto II evolutionary algorithm in the decision-making stage to optimize service placement. The method focuses on finding the location of services based on the criteria of minimum service delay and cost to achieve maximum server efficiency in the fog environment. Recently, Cao et al. [123] proposed a hybrid containerized service placement method, which is a combination of the submodular and convex optimization approaches that aim to minimize service response time. Their method is a greedy algorithm with a polynomial-time complexity.

In the following, a brief comparison of all the methods highlighted in the above taxonomy from different points of view is given in Tables 2 and 3, which can help to summarize and draw valuable insights and conclusions.

Table 2 Comparison of service placement methods in terms of their optimization strategy, advantages, and shortcomings

Ref	Optimization strategy	Evaluation environment	Target application	Advantages	Drawbacks
[54]	ILP	Implementation	Smart buildings, smart energy grids, and smart mobility	- Service migration reduction - Hop count reduction	- Not considering the service priorities - Not considering energy consumption - High time complexity
[55]	ILP	CPLEX Software	Smartphone users	Fog resource usage minimization	Limited network coverage
[56]	ILP	Analytical tool	Smartphone users	Delay reduction	Not considering metrics other than delay
[57]	MINLP	MATLAB	IoT applications	Energy consumption reduction	- Not considering delay - High computational complexity
[58]	Divide and conquer	Analytical tool	Smartphone users	Delay reduction	Not considering integrated and non-dividable apps
[59]	ILP	MATLAB and Gurobi Optimizer solver	Smartphone users	- Users' cost reduction - Server availability promotion	Simultaneous offloading of all services to several fog nodes
[60]	Divide and conquer using a central coordinator	FIT/IoT-LAB and Grid'5000	IoT applications	Cost reduction	Not considering energy consumption
[62]	Fully polynomial-time approximation	Simulation, C++	IoT applications	Delay and execution time reduction	Inexact random algorithm for non-parallelizable apps
[63]	Markov approximation	Simulator ONE	Smartphone users	- Service migration reduction - Delay reduction	Unrealistic assumptions in the service model
[64]	Markov approximation	Simulation and real-world scenario	Smartphone users	- Service migration reduction - Suitable for dense and scattered environments	Not suitable for big data
[70]	Hybrid approach based on GA and task prioritization	MATLAB	IoT applications	Considering service priority	Not considering availability and service cost

Table 2 (continued)

Ref	Optimization strategy	Evaluation environment	Target application	Advantages	Drawbacks
[71]	GA and penalty-based	iFogSim	IoT applications	Considering penalty and service priority	Not considering mobility
[72]	GA & LLC	Python	IoT applications	Dealing with placement and load balancing at the same time	Not considering energy consumption
[73]	Clustering-based heuristic	iFogSim	Smart home applications	Fog resources efficiency promotion	Not considering mobility
[74]	Distributed ACO	Testbed	Smart vehicles applications	- Low complexity - Resource usage minimization	- Not considering metrics but cost - Single point of failure
[75]	First-Fit, Best-fit, Best-fit-Queue, service atomization and parallelism	Laboratory implementation	IoT applications	Meeting the service deadlines	Not considering energy consumption and cost
[76]	GA	Simulator YAFS	IoT applications	- Energy consumption reduction - Delay reduction	Lack of accurate description of the proposed method
[77]	Hybrid GA-PSO	MATLAB	IoT applications	- Energy consumption reduction - Delay reduction - Efficiency promotion	High time complexity
[78]	Hybrid GA- Social spider optimization	Unknown simulator	IoT applications	Energy consumption reduction	- Not considering metrics but energy - Not considering delay in VM migration
[79]	Elitism-based GA	Laboratory implementation	IoT applications	- Energy consumption reduction - Cost reduction - Execution time reduction	High time complexity

Table 2 (continued)

Ref	Optimization strategy	Evaluation environment	Target application	Advantages	Drawbacks
[80]	Hybrid elitism-based GA and PSO	Testbed	Industrial IoT applications	- Energy consumption reduction - Cost reduction - Execution time reduction	- Not considering load balancing
[81]	WSGA NSGA-II MOEA/D	Laboratory implementation	IoT applications	- Delay reduction - Diverse solutions	High time complexity
[82]	Hybrid GA-PSO	Unknown Simulator	IoT applications	- Energy consumption reduction - Network overhead reduction	- High time complexity - Not considering mobility
[83]	GA, SA, and PSO	Simulator YAFS	IoT applications	- Energy consumption reduction - Cost reduction	Not considering service priorities
[84]	Prioritization and sort-based heuristic	Custom simulator using Java	IoT applications	- Cost reduction - Delay reduction	Not considering energy consumption and Virtualization
[90]	Markov decision-making process and DRL	Unknown simulator	Smart vehicles applications	- Energy consumption reduction - Cost reduction - Delay reduction	Considering a fixed number of resources and bandwidth for each vehicle
[91]	DRL	Simulator SUMO and MATLAB	Smart vehicles applications	- Delay reduction - Service migration reduction	High computational complexity
[92]	Markov decision-making process and DRL	Simulator SUMO	Self-driving vehicles applications	Considering service priorities	Not considering energy consumption
[93]	ILP and DRL	Unknown simulator	Self-driving vehicles applications	- Allocation time reduction - Execution time reduction	Not considering service deadlines
[94]	Convex-concave procedure and adaptive learning	Simulation in MATLAB	VANET applications	Considering diversity among resources and servers	High time complexity

Table 2 (continued)

Ref	Optimization strategy	Evaluation environment	Target application	Advantages	Drawbacks
[95]	Microservice placement using RL	Simulator Mininet–Wi-Fi	VANET applications	Placement of microservices based on their demand rate	Not considering energy and delay
[96]	Privacy-aware multi-agent DRL	Unknown simulator	VANET applications	Considering service priorities	Not considering energy and delay
[97]	DDQN and DRL	Simulation with Python	IoT applications	Considering service deadlines	Not considering cost and priority
[101]	Game-theoretic approximation inspired by an iterative hybrid auction method	Numerical analysis	IoT applications	Distributed algorithm	Not implementing the algorithm
[102]	Game theory and decentralized matching	Simulation with Python	IoT applications	- Considering the preferences of both fog nodes and services - Distributed algorithm	Not considering mobility
[103]	Hybrid DRL and mean field game	Unknown simulator	IoT applications	- Energy consumption reduction - Delay reduction	High cost of learning to make the right decision
[104]	Fairness-aware game theory	Simulator NS3 and CPLEX 9.0	VANET applications	Collaboration and interaction between requested services and providers	High time complexity
[105]	Game theory pareto optimal allocation and Polyandrous-Polygamous matching	Simulation in MATLAB	Smartphone users	Considering user expectations	Not considering delay and energy consumption
[106]	Game theory and DRL	Unknown simulator	VANET applications	Delay reduction	High time complexity
[107]	Non-cooperative self-executing game theory	Simulator SUMO	VANET applications	- Delay reduction - Resource utilization promotion	Not considering service deadlines

Table 2 (continued)

Ref	Optimization strategy	Evaluation environment	Target application	Advantages	Drawbacks
[112]	GAT and DRL	Unknown simulator	Smartphone users	- Simultaneous offloading of multiple tasks - Utility promotion	Not considering energy consumption
[113]	GNN and RL	Simulation with Python	Smartphone users	- Energy consumption reduction - Makespan time reduction	Not considering time complexity
[114]	Multi-objective RL and a trained neural network	Unknown simulator	IoT applications	- Energy consumption reduction - Delay reduction	Not considering time complexity
[115]	Deep graph matching based on GNN	Simulation with Python-based PyTorch	IoT applications	Suitable for real-time applications	Not considering network metrics
[116]	deep deterministic policy gradient and GAT	Simulation with Python	VANET applications	Suitable for small scale systems	Not suitable for large scale and dense systems
[117]	GCN and DRL	Real-world experiments	Smartphone users	- Considering task dependencies - Cost reduction	Not considering service priorities
[118]	GNN	Unknown simulator	Industrial IoT applications	- Delay reduction - Limiting the search space	High computational complexity
[119]	Fuzzy neural network	Simulation in MATLAB	Internet of vehicles applications	- Energy consumption reduction - Process time reduction	Not considering cost, priority, and virtualization
[120]	INLP, generalized decomposition and convex approximation	Simulation in MATLAB	IoT transportation applications	Offloading rate promotion	Needing to discover the location of the UAV in each round
[121]	Clustering and Bully algorithm	Unknown simulator	IoT applications	Considering service migration	Considering constant deadlines for various services
[122]	Control cycle and pareto II evolutionary algorithm	Simulator iFogSim	IoT applications	Fog nodes efficiency promotion	Not considering energy consumption

Table 2 (continued)

Ref	Optimization strategy	Evaluation environment	Target application	Advantages	Drawbacks
[123]	Submodular and convex optimization	Lab. implementation and simulation	IoT applications	- Low time complexity - Considering virtualization	Not considering critical objectives other than response time

Table 3 Comparison of service placement methods in terms of their optimization metrics and virtualization approach

Refs.	Application			Optimization metrics					Virtualization		
	IoT	MEC	VANET	Delay	Energy consumption	Cost	Priority	Other metrics	VM	container	None
[54]	✓							✓	✓		
[55]	✓				✓				✓		
[56]		✓		✓					✓		
[57]	✓				✓					✓	
[58]		✓		✓				✓	✓		
[59]		✓				✓			✓		
[60]	✓					✓				✓	
[62]	✓			✓					✓		
[63]		✓		✓		✓			✓		
[64]		✓				✓		✓	✓		
[70]	✓				✓			✓			✓
[71]	✓			✓	✓	✓		✓			✓
[72]	✓					✓	✓	✓			✓
[73]	✓			✓	✓			✓			✓
[74]			✓					✓			✓
[75]	✓			✓				✓	✓		
[76]	✓			✓	✓	✓			✓		
[77]	✓			✓	✓				✓		
[78]	✓				✓				✓		
[79]	✓				✓	✓		✓		✓	
[80]	✓				✓	✓				✓	
[81]	✓			✓				✓			✓

Table 3 (continued)

Refs.	Application			Optimization metrics					Virtualization		
	IoT	MEC	VANET	Delay	Energy consumption	Cost	Priority	Other metrics	VM	container	None
[82]	✓				✓			✓			✓
[83]	✓				✓	✓		✓	✓		
[84]	✓			✓		✓					✓
[90]			✓	✓	✓	✓					✓
[91]			✓	✓				✓			✓
[92]			✓				✓	✓			✓
[93]			✓	✓				✓	✓		
[94]			✓	✓				✓			✓
[95]			✓					✓		✓	
[96]			✓			✓					✓
[97]	✓			✓	✓			✓			✓
[101]	✓				✓	✓					✓
[102]	✓				✓			✓			✓
[103]		✓		✓	✓						✓
[104]			✓	✓				✓			✓
[105]		✓						✓			✓
[106]			✓	✓				✓			✓
[107]			✓	✓		✓		✓			✓
[112]		✓						✓			✓
[113]		✓		✓	✓						✓
[114]	✓			✓	✓						✓
[115]	✓							✓			✓

Table 3 (continued)

Refs.	Application			Optimization metrics					Virtualization		
	IoT	MEC	VANET	Delay	Energy consumption	Cost	Priority	Other metrics	VM	container	None
[116]			✓					✓			✓
[117]		✓				✓		✓			✓
[118]	✓			✓				✓	✓		
[119]			✓		✓			✓			✓
[120]	✓			✓	✓			✓			✓
[121]	✓			✓				✓			✓
[122]	✓			✓			✓	✓			✓
[123]	✓							✓		✓	

4 Discussion

According to the research conducted in this study, the service placement methods can be categorized into seven major classes including exact solutions, approximate solutions, heuristic and meta-heuristic-based, machine learning-based, game theory-based, neural networks-based algorithms and other methods, in terms of their optimization strategies.

As it is illustrated in Table 2, the main approaches in the exact solution category have been using the ILP method. Using this method, increases the computational complexity, and since these solutions cannot manage a large number of variables in a specific and reasonable period of time, the use of these approaches in the fog computing environment is not very desirable. According to Table 3, most of the solutions introduced in this category have solved the problem of service placement for IoT and MEC networks. Researchers have used analytical tools to evaluate their methods in this category and delay is the most important criterion that these methods have tried to reduce. Virtualization considered in most methods of this category is based on VM. In general, it can be stated that the use of methods based on exact solutions is suitable for small-sized problems, and they are not suitable for fog computations and large-scale networks.

Approximate methods can guarantee the existence of the solution within the stated error range. As can be seen in Tables 2 and 3, most of the service placement methods in this category are for IoT and MEC applications and are based on the Markov approximation algorithm. The main purpose of the methods of this category is to reduce delay and cost, and simulation has been used to evaluate the obtained results. Virtualization of these approaches has been mainly based on VM. The major drawback of these approaches is that they do not provide an optimal solution. The main application of these methods is when finding an acceptable solution in a relatively short period of time is more important than finding a costly optimal solution.

Exact solutions are not applicable to large-scale problems and approximate solutions do not provide optimal answers. Therefore, many researchers have gone to heuristic and meta-heuristic algorithms to solve the service placement problem. GA and PSO algorithms are among the most widely used algorithms exerted in the category of heuristic and meta-heuristic algorithms for service placement [70–84]. The methods of this category have the highest frequency in our taxonomy. The logic of using these algorithms is to optimize an initial valid location in an iterative process. Although this capability is the main advantage of this type of algorithm, in fog computing due to the heterogeneous structure, as well as in the VANET and IoT due to the rapid changes of nodes and delay-sensitive time limits of the services, it is not so easy to determine a valid initial location. As can be seen in Tables 2 and 3, most of the methods in this category have tried to reduce QoS criteria including delay, energy and cost. Most of these methods have proposed virtualization based on VM and used simulation to evaluate their performance.

Machine learning techniques are suitable for performing complex and repetitive tasks, and for this reason, researchers have used these techniques to offload and

deploy services. As seen in Table 2, the most widely used technique in this category is DRL [90–97]. The main reason for using this technique compared to supervised and unsupervised techniques is the continuous training of the operator to achieve the best results. As can be seen in Table 3, the methods performed with this technique are mainly introduced for the applications of VANET [90–96]. The reason for this is that in these networks, a large volume of services with different priorities are sent by mobile devices over short distances and they need fast processing. The reviewed approaches in this category have used simulation to evaluate their method and have mainly tried to reduce the criterion of service execution time. Of course, it is important to mention that the learning process of the agent in the reinforcement learning technique will be time-consuming for complex environments such as the VANET.

Game theory-based service placement methods are inspired by the analysis of the players' positions and their interaction with other players to choose the appropriate nodes for placing tasks and services. As can be seen in Table 2, most of these methods, in addition to game theory techniques, have used other complementary techniques such as reinforcement learning to discover the best answer. As can be seen in Table 3, the methods of this category have often been introduced for IoT applications and have used simulators to evaluate their results. These methods have tried to reduce criteria such as delay and energy consumption. Although the application of game theory techniques reduces the delay and the amount of energy consumed for the placement of services, it increases the time complexity.

Most state-of-the-art neural network-based service placement methods have leveraged GNN and GAT to select the appropriate nodes for offloading and placing tasks and services [112–118]. The use of neural networks such as graph neural networks can increase the speed of data classification and analysis and make it easier to choose the right node to place the service. As can be seen in Tables 2 and 3, these methods are often presented for VANET and Internet of Things applications and are mainly simulated using Python. However, it is worth mentioning that the use of neural networks has increased the computational complexity of this category of service placement methods.

Finally, there are methods that do not fit into any of the six main categories of service placement methods mentioned above. These methods have used approaches such as clustering [120], Bully algorithm [121], control cycle [122], convex optimization [123] for service placement. According to Tables 2 and 3, the mentioned methods have often been introduced to accommodate the services of the Internet of Things applications and have tried to place more tasks in less time. These methods have mainly taken into account the service deadline and used simulation for evaluation.

5 Recommendations and future works

After studying the research carried out in the field of service placement in fog computing, we found that there are a number of open issues in this field. So, future research can solve the existing shortcomings and provide more complete

solutions for this problem by addressing them. These challenges to be addressed by future works are as follows:

1. While most existing service placement methods consider user-requested services as equal significance, in applications such as VANET, emergency requested services such as ambulance, fire department, police should have a higher priority. Therefore, this should be taken into consideration in future works.
2. Most of the existing methods consider applications to include non-dependent services. This is despite the fact that the dependence of application services on each other is completely logical and common.
3. In many applications, including VANET or UAV, service-requesting users and even fog-level devices can be mobile. The future methods of service placement should consider this issue.
4. Privacy and security are always considered fundamental challenges in various information technology problems. While most of the existing service deployment methods consider QoS parameters for optimization, future methods should consider security, protecting the users' privacy, and preventing user information leakage.
5. It is quite probable that fog nodes will fail and become unavailable over time. This challenge has been ignored in most of the existing methods. A comprehensive and robust method for service placement in the future should be prepared for different scenarios of failure of fog nodes and take into account various issues raised in fault tolerance, including reliability, availability, and safety.
6. While most of the existing methods assume that user services are requested from one application, researchers in this field should research the development of methods for placing the services requested from several applications of users.
7. Due to the resource limitations in fog nodes, so far, the category of real-time services placement in applications such as online streaming games, video streaming, and augmented reality has not been addressed much. With the advancement of technology and infrastructure in this area, it is possible to consider the use of fog computing and service deployment for such applications in future works.
8. Although machine learning techniques and neural networks have recently attracted more attention in the field of fog computing and service placement, due to the unique capabilities of these techniques in learning and predicting the future behavior of users in the IoT network, it is possible to further apply these techniques in this field and present more valuable service placement methods.
9. According to our studies, the literature suffers from the lack of standard benchmarks for valid and consistent comparisons. Preparation and development of standard benchmarks and real-world and even synthetic datasets in various applications, in such a way that it is accepted by the general researchers of this field, can be valuable work for more valid evaluation and comparison of existing and future methods of service placement in the cloud-fog computing environment.

10. Creating a comprehensive and integrated simulation environment that includes repositories of implementations of existing methods available to everyone can be a great help in advancing studies and developing more efficient service placement methods.

6 Conclusion

Fog computing is a supplement to address the challenges facing cloud computing, including the distance of computing resources from end devices and as a result, the delay in responding. In this model, fog computing nodes (such as switches, routers, smart gateways) which have more limited resources than cloud data centers are placed in an intermediate layer between users' end devices and cloud servers. The heterogeneity and dispersion of the fog nodes have created challenges for this computing model, one of the most important of which is the problem of placing the services requested by users. Due to the significance of the problem, researchers have so far presented numerous methods to solve it. Each of these methods has considered aspects of this problem and tried to optimize some Quality-of-Service criteria. In this paper, the existing methods of service placement in fog computing are classified into seven main categories including exact and approximate solutions, heuristic and meta-heuristic-based, machine learning-based, game theory-based, neural networks-based algorithms and other methods, and the most prominent papers of each category were reviewed and compared. The results of our studies demonstrated that while exact solutions are not suitable for large-scale problems and approximate solutions do not guarantee the optimal answer, methods based on heuristic and meta-heuristic algorithms have received the most attention from researchers in this literature. However, these methods also face the challenge of determining the valid initial location due to the rapid changes of fog nodes and the sensitivity of services to delay. Machine learning techniques have recently received more attention from researchers in this literature due to their ability to learn and predict the future behavior of the network. The use of GNN is another widely used approach due to increasing the speed of data classification and analysis, which is placed in the category of neural network-based service placement methods. Considering the studies done, at the end of this paper, recommendations for future works, including considering the mobility of nodes in fog computing, priority and dependency of services, probability of node failure and fault tolerance, and creation of standard benchmarks and datasets and so on, were presented.

Author contributions Both authors in all stages of work including conceptualization, investigation, methodology, writing original draft, designing figures and tables, writing review, and editing have contributed equally.

Funding The authors did not receive support from any organization for the submitted work.

Data availability This manuscript is a review article, and no data or materials have been used.

Declarations

Competing interest The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Ethical approval Hereby, We Fatemeh Sarkohaki and Mohsen Sharifi consciously assure that for the manuscript "Service Placement in Fog–cloud Computing Environments: A Comprehensive Literature Review" the following is fulfilled: 1) This material is the original work of the authors, which has not been previously published elsewhere. 2) The paper is not currently being considered for publication elsewhere. 3) The paper reflects the authors' own research and analysis in a truthful and complete manner. 4) The paper properly credits the meaningful contributions of co-authors and co-researchers. 5) All sources used are properly disclosed (correct citation). 6) All authors have been personally and actively involved in substantial work leading to the paper and will take public responsibility for its content.

References

1. Qays MO et al (2023) Key communication technologies, applications, protocols and future guides for IoT-assisted smart grid systems: a review. Energy Rep 9:2440–2452
2. Moudgil V et al (2023) Integration of IoT in building energy infrastructure: a critical review on challenges and solutions. Renew Sustain Energy Rev 174:113121
3. Sharma VK et al (2022) An optimization-based machine learning technique for smart home security using 5G. Comput Electr Eng 104:108434
4. Philip SJ, Luu TJ, Carte T (2023) There's No place like home: Understanding users' intentions toward securing internet-of-things (IoT) smart home networks. Comput Hum Behav 139:107551
5. Khanpara P et al (2023) A context-aware internet of things-driven security scheme for smart homes. Secur Priv 6(1):e269
6. Zaminkar M, Sarkohaki F, Fotohi R (2021) A method based on encryption and node rating for securing the RPL protocol communications in the IoT ecosystem. Int J Commun Syst 34(3):e4693
7. Salehi-Amiri A et al (2022) Designing an effective two-stage, sustainable, and IoT based waste management system. Renew Sustain Energy Rev 157:112031
8. Salman MY, Hasar H (2023) Review on environmental aspects in smart city concept: water, waste, air pollution and transportation smart applications using IoT techniques. Sustain Cities Soc 94:104567
9. Hashemi-Amiri O et al (2023) An allocation-routing optimization model for integrated solid waste management. Exp Syst Appl 227:120364
10. Sridhar K et al (2023) A modular IOT sensing platform using hybrid learning ability for air quality prediction. Meas Sens 25:100609
11. Barthwal A (2023) A Markov chain–based IoT system for monitoring and analysis of urban air quality. Environ Monit Assess 195(1):235
12. Kumar M et al (2023) Quality assessment and monitoring of river water using IoT infrastructure. IEEE Internet Things J. https://doi.org/10.1109/JIOT.2023.3238123
13. Kumar P et al (2023) A blockchain-orchestrated deep learning approach for secure data transmission in IoT-enabled healthcare system. J Parallel Distrib Comput 172:69–83
14. Krishnamoorthy S, Dua A, Gupta S (2023) Role of emerging technologies in future IoT-driven healthcare 4.0 technologies: a survey, current challenges and future directions. J Ambient Intell Humaniz Comput 14(1):361–407
15. Rejeb A et al (2023) The Internet of Things (IoT) in healthcare: Taking stock and moving forward. Internet of Things 22:100721
16. Ahmed ST, Kumar V, Kim J (2023) AITel: eHealth augmented intelligence based telemedicine resource recommendation framework for iot devices in smart cities. IEEE Internet Things J. https://doi.org/10.1109/JIOT.2023.3243784
17. Cheikhrouhou O et al (2023) A lightweight blockchain and fog-enabled secure remote patient monitoring system. Internet of Things 22:100691
18. Khan AA et al (2023) The collaborative role of blockchain, artificial intelligence, and industrial internet of things in digitalization of small and medium-size enterprises. Sci Rep 13(1):1656

19. Rahman A et al (2023) Towards a blockchain-SDN-based secure architecture for cloud computing in smart industrial IoT. Digit Commun Netw 9(2):411–421
20. Huang J et al (2023) AoI-aware energy control and computation offloading for industrial IoT. Futur Gener Comput Syst 139:29–37
21. Karakaya A, Ulu A, Akleylek S (2022) GOALALERT: a novel real-time technical team alert approach using machine learning on an IoT-based system in sports. Microprocess Microsyst 93:104606
22. Liu L (2021) Construction of youth public sports service system based on embedded system and wireless IoT. Microprocess Microsyst 83:103984
23. Prajapati D et al (2022) Blockchain and IoT embedded sustainable virtual closed-loop supply chain in E-commerce towards the circular economy. Comput Ind Eng 172:108530
24. Kulkarni PM et al (2022) IOT data fusion framework for e-commerce. Meas Sens 24:100507
25. Boursianis AD et al (2022) Internet of things (IoT) and agricultural unmanned aerial vehicles (UAVs) in smart farming: a comprehensive review. Internet of Things 18:100187
26. Zeng H et al (2023) An IoT and Blockchain-based approach for the smart water management system in agriculture. Expert Syst 40(4):e12892
27. McCaig M, Rezania D, Dara R (2023) Framing the response to IoT in agriculture: a discourse analysis. Agric Syst 204:103557
28. Krishankumar R, Ecer F (2023) Selection of IoT service provider for sustainable transport using q-rung orthopair fuzzy CRADIS and unknown weights. Appl Soft Comput 132:109870
29. Jiang H et al (2023) RETRACTED ARTICLE: creating a ubiquitous learning environment using IoT in transportation. Soft Comput 27(2):1213–1213
30. Wu X et al (2023) A digital decision approach for scheduling process planning of shared bikes under internet of things environment. Appl Soft Comput 133:109934
31. Kuo Y-H, Leung JM, Yan Y (2023) Public transport for smart cities: recent innovations and future challenges. Eur J Oper Res 306(3):1001–1026
32. Quy VK et al (2022) Smart healthcare IoT applications based on fog computing: architecture, applications and challenges. Complex Intell Syst 8(5):3805–3815
33. Peixoto M et al (2023) FogJam: a fog service for detecting traffic congestion in a continuous data stream VANET. Ad Hoc Netw 140:103046
34. Tavousi F, Azizi S, Ghaderzadeh A (2022) A fuzzy approach for optimal placement of IoT applications in fog–cloud computing. Clust Comput 25:1–18
35. Sabuj SR et al (2022) Delay optimization in mobile edge computing: cognitive UAV-assisted eMBB and mMTC services. IEEE Trans Cognit Commun Netw 8(2):1019–1033
36. Kang H et al (2023) Cooperative UAV resource allocation and task offloading in hierarchical aerial computing systems: a MAPPO based approach. IEEE Internet Things J. https://doi.org/10.1109/JIOT.2023.3240173
37. He Y et al (2022) Trajectory optimization and channel allocation for delay sensitive secure transmission in UAV-relayed VANETs. IEEE Trans Veh Technol 71(4):4512–4517
38. Sadeghi-Niaraki A (2023) Internet of thing (IoT) review of review: bibliometric overview since its foundation. Futur Gener Comput Syst. https://doi.org/10.1016/j.future.2023.01.016
39. Rahimikhanghah A et al (2022) Resource scheduling methods in cloud and fog computing environments: a systematic literature review. Clust Comput. https://doi.org/10.1007/s10586-021-03467-1
40. Bonomi, F., et al. *Fog computing and its role in the internet of things*. in *Proceedings of the first edition of the MCC workshop on Mobile cloud computing*. 2012.
41. Kumar D, Annam S (2022) *Fog Computing Applications with Decentralized Computing Infrastructure—Systematic Review*. in *PROCEEDINGS OF THE SEVENTH INTERNATIONAL CONFERENCE ON MATHEMATICS AND COMPUTING: ICMC 2021*. 2022. Springer.
42. Songhorabadi M et al (2023) Fog computing approaches in IoT-enabled smart cities. J Netw Comput Appl 211:103557
43. Sethi V, Pal S (2023) FedDOVe: a federated deep Q-learning-based offloading for vehicular fog computing. Futur Gener Comput Syst 141:96–105
44. Hazra A et al (2023) Fog computing for next-generation internet of things: fundamental, state-of-the-art and research challenges. Comput Sci Rev 48:100549
45. Singh S, Vidyarthi D (2023) An integrated approach of ml-metaheuristics for secure service placement in fog–cloud ecosystem. Internet of Things 22:100817
46. Singh S, Vidyarthi D (2022) *QoS-Aware Service Placement for Fog Integrated Cloud Using Modified Neuro-Fuzzy Approach*. in *Soft Computing and Its Engineering Applications: 4th*

INTERNATIONAL CONFERENCE, icSoftComp 2022, Changa, Anand, India, December 9–10, 2022, Proceedings. 2023. Springer.
47. Teng M et al. (2020) *Priority based service placement strategy in heterogeneous mobile edge computing*. in *Algorithms and Architectures for Parallel Processing: 20th INTERNATIONAL CONFERENCE, ICA3PP 2020, New York City, NY, USA, October 2–4, 2020, Proceedings, Part I 20*. 2020. Springer.
48. Zare M, Sola YE, Hasanpour H (2023) Towards distributed and autonomous IoT service placement in fog computing using asynchronous advantage actor-critic algorithm. J King Saud Univ Comput Inf Sci 35(1):368–381
49. Das R, Inuwa MM (2023) *A review on fog computing: issues, characteristics, challenges, and potential applications.* Telematics and Informatics Reports, p. 100049.
50. Salaht FA, Desprez F, Lebre A (2020) An overview of service placement problem in fog and edge computing. ACM Comput Surv (CSUR) 53(3):1–35
51. Matoušek J, Gärtner B (2007) Understanding and using linear programming. Springer, Berlin
52. Kuhn HW, Tucker AW (2013) Nonlinear programming. Traces and emergence of nonlinear programming. Springer, Berlin, pp 247–258
53. Vielma JP (2015) Mixed integer linear programming formulation techniques. SIAM Rev 57(1):3–57
54. Velasquez K et al (2017) Service placement for latency reduction in the internet of things. Ann Telecommun 72:105–115
55. Tinini RI et al. (2017) *Optimal placement of virtualized BBU processing in hybrid cloud-fog RAN over TWDM-PON*. in *GLOBECOM 2017–2017 IEEE GLOBAL COMMUNICATIONS CONFERENCE*. IEEE.
56. Gong Y (2020) *Optimal edge server and service placement in mobile edge computing*. in *2020 IEEE 9th JOINT INTERNATIONAL INFORMATION TECHNOLOGY AND ARTIFICIAL INTELLIGENCE CONFERENCE (ITAIC)*. IEEE.
57. Kim W-S, Chung S-H (2018) User-participatory fog computing architecture and its management schemes for improving feasibility. IEEE Access 6:20262–20278
58. Yala L, Frangoudis PA, Ksentini A (2018) *Latency and availability driven VNF placement in a MEC-NFV environment*. in *2018 IEEE GLOBAL COMMUNICATIONS CONFERENCE (GLOBECOM)*. IEEE.
59. Daneshfar N et al. (2018) *Service allocation in a mobile fog infrastructure under availability and qos constraints*. in *2018 IEEE GLOBAL COMMUNICATIONS CONFERENCE (GLOBECOM)*. IEEE.
60. Donassolo B, et al. (2019) *Fog based framework for IoT service provisioning*. in *2019 16th IEEE ANNUAL CONSUMER COMMUNICATIONS & NETWORKING CONFERENCE (CCNC)*. IEEE.
61. Chen M et al (2013) Markov approximation for combinatorial network optimization. IEEE Trans Inf Theory 59(10):6301–6327
62. Yu R, Xue G, Zhang X (2018) *Application provisioning in fog computing-enabled internet-of-things: A network perspective*. in *IEEE INFOCOM 2018-IEEE CONFERENCE ON COMPUTER COMMUNICATIONS*. IEEE.
63. Ouyang T, Zhou Z, Chen X (2018) Follow me at the edge: mobility-aware dynamic service placement for mobile edge computing. IEEE J Sel Areas Commun 36(10):2333–2345
64. Ning Z et al (2020) Distributed and dynamic service placement in pervasive edge computing networks. IEEE Trans Parallel Distrib Syst 32(6):1277–1292
65. Jokar E, Mosleh M, Kheyrandish M (2022) Discovering community structure in social networks based on the synergy of label propagation and simulated annealing. Multimed Tools Appl 81(15):21449–21470
66. Mirjalili S, Mirjalili S (2019) *Genetic algorithm.* Evolutionary Algorithms and Neural Networks: Theory and Applications, p. 43–55.
67. Wang D, Tan D, Liu L (2018) Particle swarm optimization algorithm: an overview. Soft Comput 22:387–408
68. Blum C (2005) Ant colony optimization: Introduction and recent trends. Phys Life Rev 2(4):353–373
69. Jokar E, Mosleh M, Kheyrandish M (2022) GWBM: an algorithm based on grey wolf optimization and balanced modularity for community discovery in social networks. J Supercomput 78(5):7354–7377

70. Hoseiny F et al. (2021) *PGA: a priority-aware genetic algorithm for task scheduling in heterogeneous fog–cloud computing*. in *IEEE INFOCOM 2021-IEEE CONFERENCE ON COMPUTER COMMUNICATIONS WORKSHOPS (INFOCOM WKSHPS)*. IEEE.
71. Sarrafzade N, Entezari-Maleki R, Sousa L (2022) A genetic-based approach for service placement in fog computing. J Supercomput 78(8):10854–10875
72. Maia AM et al. (2020) *Dynamic service placement and load distribution in edge computing*. in *2020 16TH INTERNATIONAL CONFERENCE ON NETWORK AND SERVICE MANAGEMENT (CNSM)*. IEEE.
73. Khosroabadi F, Fotouhi-Ghazvini F, Fotouhi H (2021) Scatter: service placement in real-time fog-assisted iot networks. J Sens Actuator Netw 10(2):26
74. Eyckerman R et al (2020) Requirements for distributed task placement in the fog. Internet of Things 12:100237
75. Souza VB et al (2018) Towards a proper service placement in combined Fog-to-Cloud (F2C) architectures. Futur Gener Comput Syst 87:1–15
76. Apat HK et al. (2021) *A Nature-Inspired-Based Multi-objective Service Placement in Fog Computing Environment*, in *Intelligent Systems: Proceedings of ICMIB 2020*. Springer. p. 293–304.
77. Ma R (2021) Edge server placement for service offloading in internet of things. Secur Commun Netw 2021:1–16
78. Hu Y et al (2022) An energy-aware service placement strategy using hybrid meta-heuristic algorithm in iot environments. Clust Comput 26:1–7
79. Natesha B, Guddeti RMR (2021) Adopting elitism-based Genetic Algorithm for minimizing multi-objective problems of IoT service placement in fog computing environment. J Netw Comput Appl 178:102972
80. Natesha B, Guddeti RMR (2022) Meta-heuristic based hybrid service placement strategies for two-level fog computing architecture. J Netw Syst Manage 30(3):47
81. Guerrero C, Lera I, Juiz C (2019) Evaluation and efficiency comparison of evolutionary algorithms for service placement optimization in fog architectures. Futur Gener Comput Syst 97:131–144
82. Shahryari O-K et al (2021) Energy and task completion time trade-off for task offloading in fog-enabled IoT networks. Pervasive Mob Comput 74:101395
83. Apat HK et al (2024) A hybrid meta-heuristic algorithm for multi-objective IoT service placement in fog computing environments. Decis Anal J 10:100379
84. Azizi S et al (2024) DCSP: a delay and cost-aware service placement and load distribution algorithm for IoT-based fog networks. Comput Commun 215:9–20
85. Jordan MI, Mitchell TM (2015) Machine learning: Trends, perspectives, and prospects. Science 349(6245):255–260
86. Jokar E, Mosleh M, Kheyrandish M (2022) Overlapping community detection in complex networks using fuzzy theory, balanced link density, and label propagation. Expert Syst 39(5):e12921
87. Quadri C, Ceselli A, Rossi GP (2023) Multi-user edge service orchestration based on deep reinforcement learning. Comput Commun 203:30–47
88. Hao H et al (2023) Computing offloading with fairness guarantee: a deep reinforcement learning method. IEEE Trans Circuits Syst Video Technol. https://doi.org/10.1109/TCSVT.2023.3255229
89. Liu T et al (2022) Deep reinforcement learning based approach for online service placement and computation resource allocation in edge computing. IEEE Trans Mob Comput. https://doi.org/10.1109/TMC.2022.3148254
90. Zhan W et al (2020) Deep-reinforcement-learning-based offloading scheduling for vehicular edge computing. IEEE Internet Things J 7(6):5449–5465
91. Talpur A, Gurusamy M (2021) DRLD-SP: a deep-reinforcement-learning-based dynamic service placement in edge-enabled internet of vehicles. IEEE Internet Things J 9(8):6239–6251
92. Lv P et al (2022) Edge computing task offloading for environmental perception of autonomous vehicles in 6G networks. IEEE Trans Netw Sci Eng. https://doi.org/10.1109/TNSE.2022.3211193
93. Ibn-Khedher H et al (2022) Next-generation edge computing assisted autonomous driving based artificial intelligence algorithms. IEEE Access 10:53987–54001
94. Zhou Z et al (2019) Reliable task offloading for vehicular fog computing under information asymmetry and information uncertainty. IEEE Trans Veh Technol 68(9):8322–8335
95. Nsouli A, El-Hajj W, Mourad A (2023) Reinforcement learning based scheme for on-demand vehicular fog formation. Veh Commun 40:100571
96. Wei D et al (2022) Privacy-aware multiagent deep reinforcement learning for task offloading in VANET. IEEE Trans Intell Transp Syst. https://doi.org/10.1109/TITS.2022.3202196

97. Sharma A, Thangaraj V (2024) Intelligent service placement algorithm based on DDQN and prioritized experience replay in IoT-Fog computing environment. Internet of Things 25:101112
98. Tian Z et al (2019) Evaluating reputation management schemes of internet of vehicles based on evolutionary game theory. IEEE Trans Veh Technol 68(6):5971–5980
99. Sun Z et al (2023) BARGAIN-MATCH: a game theoretical approach for resource allocation and task offloading in vehicular edge computing networks. IEEE Trans Mob Comput. https://doi.org/10.1109/TMC.2023.3239339
100. Chen Y et al (2022) Qoe-aware decentralized task offloading and resource allocation for end-edge-cloud systems: a game-theoretical approach. IEEE Trans Mob Comput 23(1):769–784
101. Kayal P, Liebeherr J (2019) *Distributed service placement in fog computing: An iterative combinatorial auction approach*. in *2019 IEEE 39th INTERNATIONAL CONFERENCE ON DISTRIBUTED COMPUTING SYSTEMS (ICDCS)*. IEEE.
102. Sharma A, Thangaraj V (2022) DMAP: a decentralized matching game theory based optimized internet of things application placement in fog computing environment. Concurr Comput Pract Exp 34(23):e7189
103. Shi D et al (2020) Mean field game guided deep reinforcement learning for task placement in cooperative multiaccess edge computing. IEEE Internet Things J 7(10):9330–9340
104. Aloqaily MB, Kantarci, Mouftah HT (2017) *Fairness-aware game theoretic approach for service management in vehicular clouds*. in *2017 IEEE 86th VEHICULAR TECHNOLOGY CONFERENCE (VTC-Fall)*. IEEE.
105. Zafari F et al (2020) Let's share: a game-theoretic framework for resource sharing in mobile edge clouds. IEEE Trans Netw Serv Manage 18(2):2107–2122
106. Xiao Z et al (2019) Vehicular task offloading via heat-aware MEC cooperation using game-theoretic method. IEEE Internet Things J 7(3):2038–2052
107. Shabir B et al (2022) On collective intellect for task offloading in vehicular fog paradigm. IEEE Access 10:101445–101457
108. Krogh A (2008) What are artificial neural networks? Nat Biotechnol 26(2):195–197
109. Wu Z et al (2020) A comprehensive survey on graph neural networks. IEEE trans Neural Netw Learn Syst 32(1):4–24
110. Li Y, Liang S, Jiang Y (2023) Path reliability-based graph attention networks. Neural Netw 159:153–160
111. Veličković P (2023) Everything is connected: Graph neural networks. Curr Opin Struct Biol 79:102538
112. Zhong X and He Y (2021) *A Cybertwin-Driven Task Offloading Scheme Based on Deep Reinforcement Learning and Graph Attention Networks*. in *2021 13th INTERNATIONAL CONFERENCE ON WIRELESS COMMUNICATIONS AND SIGNAL PROCESSING (WCSP)*. IEEE.
113. Wu T et al. (2021) *A Scalable Computation Offloading Scheme for MEC Based on Graph Neural Networks*. in *2021 IEEE Globecom Workshops (GC Wkshps)*. IEEE.
114. Eyckerman R et al. (2022) *Application placement in fog environments using multi-objective reinforcement learning with maximum reward formulation*. in *NOMS 2022–2022 IEEE/IFIP network operations and management symposium*. IEEE.
115. Zhang J et al. (2022) *Fine-grained service offloading in B5G/6G collaborative edge computing based on graph neural networks*. in *ICC 2022-IEEE INTERNATIONAL CONFERENCE ON COMMUNICATIONS*. IEEE.
116. He Y et al (2022) A DDPG hybrid of graph attention network and action branching for multi-scale end-edge-cloud vehicular orchestrated task offloading. IEEE Wirel Commun. https://doi.org/10.1109/MWC.019.2100718
117. Tang Z et al. (2020) *Dependent task offloading for multiple jobs in edge computing*. in *2020 29th INTERNATIONAL CONFERENCE ON COMPUTER COMMUNICATIONS AND NETWORKS (ICCCN)*. IEEE.
118. Sun Z, Mo Y, Yu C (2021) Graph reinforcement learning based task offloading for multi-access edge computing. IEEE Internet Things J 10(4):3138–3150
119. Liu B (2024) Hybrid fuzzy neural network for joint task offloading in the internet of vehicles. J Grid Comput 22(1):10
120. Tong S et al (2022) Joint task offloading and resource allocation for fog-based intelligent transportation systems: a uav-enabled multi-hop collaboration paradigm. IEEE Trans Intell Transp Syst. https://doi.org/10.1109/TITS.2022.3163804

121. Sarkar I et al (2021) Dynamic task placement for deadline-aware IoT applications in federated fog networks. IEEE Internet Things J 9(2):1469–1478
122. Ayoubi M, Ramezanpour M, Khorsand R (2021) An autonomous IoT service placement methodology in fog computing. Softw Pract Exp 51(5):1097–1120
123. Cao T et al (2024) Walking on two legs: joint service placement and computation configuration for provisioning containerized services at edges. Comput Netw 239:110144