Toward Optimal Service Composition in the Internet of Things via Cloud-Fog Integration and Improved Artificial Bee Colony Algorithm

Guixia Xiao*
Changde Vocational and Technical College, Modern Educational Technology Center,Hunan Changde, 415000 China
Faculty of Computer Science and Information Technology, Universiti Putra Malaysia, Selangor, Malaysia, 43400 MAS

***Abstract*—In the quest to delve deeper into the burgeoning realm of the service-oriented Internet of Things (IoT), the pressing challenge of smoothly integrating functionalities within smart objects emerges prominently. IoT devices, notorious for their resource constraints, often lean heavily on cloud infrastructures to function effectively. However, the emergence of fog computing offers a promising alternative, allowing the processing of IoT applications closer to the sensors and thereby slashing delays. This research develops a novel method for IoT service composition that leverages both fog and cloud computing, utilizing an enhanced version of the Artificial Bee Colony (ABC) algorithm to refine its convergence rate. The approach introduces a Dynamic Reduction (DR) mechanism designed to perturb dimensions innovatively. Traditionally, the ABC algorithm generates new solutions that closely mimic their parent solutions, which unfortunately slows down convergence. By initiating the process with significant dimension disparities among solutions and gradually reducing these disparities over successive iterations, this method strikes an optimal balance between exploration and exploitation through dynamic adjustment of dimension perturbation counts. Comparative analyses against contemporary methodologies reveal significant improvements: a 17% decrease in average energy consumption, a 10% boost in availability, an 8% enhancement in reliability, and a remarkable 23% reduction in average cost. Combining the strengths of fog and cloud computing with the refined ABC algorithm through the Dynamic Reduction mechanism significantly advances the efficiency and effectiveness of IoT service compositions.**

Keywords*—Internet of Things (IoT); fog computing; service composition; Artificial Bee Colony (ABC) Algorithm; Dynamic Reduction Mechanism

I. Introduction

The Internet of Things (IoT) represents an innovative technological framework enabling the interconnection of smart objects to facilitate collaboration, coordination, and communication, thereby facilitating the deployment of intelligent applications [1]. The IoT offers endless possibilities for data-driven decision-making and automation of processes. IoT has the potential to revolutionize both our professional and personal lives. With a vast network of devices and objects interconnected, the IoT facilitates seamless communication, data sharing, and intelligent decision-making, leading to significant advancements in numerous fields [2]. In the industrial environment, the IoT can enhance operational efficiency, automate processes, and improve productivity. IoT-enabled systems can optimize resource utilization, streamline workflows, and enable predictive maintenance, reducing downtime and enhancing productivity. IoT applications in industries such as manufacturing, logistics, and healthcare can also reduce costs, increase safety, and improve quality [3].

A myriad of conveniences and benefits can be derived from the IoT in our daily lives. A smart home equipped with IoT devices can control and automate various functions, including lighting, temperature, and security. Personalized healthcare management, early detection of health issues, and improved well-being are possible with connected wearable devices and health monitoring systems [4]. IoT-enabled smart cities can optimize energy usage, enhance transportation systems, and enable efficient urban planning, resulting in sustainable and livable cities. Presently, the count of interconnected smart objects exceeds eight billion, and this figure is expected to undergo substantial growth annually. Smart objects exhibit heterogeneity in their functionalities, communication prowess, and available resources. Typically, these objects grapple with constraints in resources, particularly when they are battery-powered devices such as wireless sensors and mobile phones, with limited computation and storage capacities [5].

The growing popularity of IoT has resulted in the rapid expansion of diverse IoT services across various domains. This includes areas such as home automation, healthcare, manufacturing, and agriculture. Concurrently, cloud computing adoption has spurred a shift towards Microservices Architecture (MSA) for composing services in cloud-native computing, particularly in the context of cloud application development [6]. Service composition involves a series of steps that include the provision of resources, resource allocation, deploying functions, and combining functions to create a complete service offering. Microservices, in this context, refer to compact functions that can be independently launched and expanded. They often employ distinct middleware stacks for their implementation. It is worth noting that there is a trend of migrating legacy services, originally built on monolithic architectures, to modular MSA to take advantage of technological advancements and expedite the development process. Containerization offers advantages over traditional Virtual Machines (VMs) in the cloud, particularly in terms of size and flexibility. Containers are lightweight, isolated environments that encapsulate individual microservices, allowing for efficient resource utilization and scalability [7].

IoT applications, characterized by their diverse requirements and operational constraints, necessitate the seamless integration of various smart objects while ensuring optimal energy utilization and Quality of Service (QoS) provisioning. However, the inherent resource limitations of individual IoT devices pose significant challenges in achieving efficient service composition [8]. Traditionally, IoT devices have relied on cloud infrastructures to surmount resource constraints, capitalizing on the extensive computational resources and storage capabilities offered by the cloud. Nonetheless, the reliance on distant cloud data centers introduces latency issues, particularly for applications requiring real-time or low-latency responses. In response to these challenges, fog computing has emerged as a promising paradigm, positioning computational resources closer to IoT devices at the network edge. This approach mitigates the latency drawbacks associated with conventional cloud-centric architectures, facilitating more efficient data processing and service execution.

The (IoT) is revolutionizing the way smart objects interconnect to facilitate intelligent applications across various domains, from industrial automation to personal healthcare and smart cities. As the number of interconnected devices surpasses 8 billion and continues to grow, the diversity in their functionalities and resource constraints, especially for battery-powered devices, poses significant challenges. Traditional reliance on cloud infrastructures to overcome these limitations introduces latency issues, particularly problematic for real-time applications. In this context, the study aims to address the integration of cloud and fog computing to enhance IoT service composition. By leveraging an improved Artificial Bee Colony (ABC) algorithm with a Dynamic Reduction (DR) strategy, the research seeks to optimize energy efficiency, ensure Quality of Service (QoS), and utilize resources effectively. The primary objective is to develop a cloud-fog-based service composition method that balances exploration and exploitation, reduces latency, and meets the diverse requirements of IoT applications.

The research provides the following key contributions:

- Innovative cloud-fog-based service composition: This paper introduces a novel approach to service composition in IoT applications, integrating cloud and fog computing to address the unique challenges posed by diverse IoT requirements and operational constraints.
- Enhanced ABC algorithm with DR strategy: The research incorporates the DR strategy within the ABC algorithm, offering a dynamic dimension perturbation mechanism that significantly improves the rate of convergence, thus enhancing the algorithm's efficacy in service composition.
- Balanced exploration and exploitation framework: The establishment of a balanced exploration and exploitation framework in service composition is achieved through the modulation of dimension perturbation counts. This ensures enhanced solution diversity, crucial for addressing the diverse needs of IoT applications.
- Mitigation of latency issues: By harnessing fog computing at the network edge, this research effectively mitigates latency issues associated with traditional cloud data centers. The proximity of computational resources facilitates efficient data processing and service execution, leading to substantial reductions in delays.

The reminder of this paper consists of as follows, Section II reviews the previous studies. Section III discuss about the methodology. Section IV presents the results and discussion. Finally, the paper concludes in Section V.

II. Literature Review

Previous research has explored various approaches to address service composition in IoT. Cloud computing has been extensively used to augment the capabilities of IoT devices, enabling scalable and on-demand access to resources. However, the inherent latency in cloud data centers may not always meet the stringent latency requirements of certain IoT applications. Fog computing, which operates at the network edge, has garnered significant attention due to its capability to process data in close proximity to the data source. This approach effectively reduces latency and minimizes bandwidth usage.

The paper in [9] have put forward a novel approach for the composition of IoT services, with a focus on both energy efficiency and QoS. The proposed approach adopts hierarchical optimization strategies as its underlying framework. In the first stage, the compromise ratio technique is used to filter out services that match the user's unique QoS criterion. After that, a relative dominance idea is used to find the best service for the composite service. The aim of this approach is to extend the lifespan of IoT devices and minimize energy consumption. The evaluation of relative dominance takes into account various factors, including the energy profile of the service, QoS characteristics, and user preferences. Results of the simulation provide evidence that the proposed algorithm surpasses alternatives, as indicated by improved performance metrics. These include enhanced optimality, decreased energy consumption, and reduced selection time.

The authors in study [10] have introduced a comprehensive framework that facilitates the development of interoperable, cost-effective, and customizable IoT prototypes. This framework is based on an architectural design that encapsulates any IoT component, whether hardware or operational logic, as an individual web service characterized by an array of transferable states. These IoT components may be easily linked into custom applications by providing a proper sequence of state transfers across web services. The paper establishes an architecture driven by a finite-state machine (FSM) model and presents a practical implementation of this architecture called the Hyper Sensor Markup Language (HSML). Furthermore, the paper delves into two real-world use cases and provides evaluations pertaining to their application within the proposed framework.

The study in [11] proposed a novel approach aiming to identify and share common functional components within IoT service compositions. The objective is to integrate and optimize concurrent requests, ensuring that the temporal dependencies of shared components are not violated and thereby improving resource utilization. This approach enables the composition of IoT services in the context of concurrent requests to be transformed into a constrained multi-objective optimization

problem, which can be effectively addressed using heuristic algorithms. The proposed technique has been extensively evaluated through experiments, comparing it with state-of-the-art algorithms. The results demonstrate the efficiency and performance of this approach, particularly when the number of IoT nodes is relatively large and their functionalities exhibit a high degree of overlap.

The paper in [12] presented a novel approach to address the service composition problem while improving QoS. Their approach combines a hidden Markov model (HMM) with an ant colony optimization (ACO) algorithm. The HMM predicts quality of service, with the emission and transition matrices being enhanced using the Viterbi algorithm. QoS estimation is performed using the ACO algorithm to identify a suitable path. The results of their study demonstrate the effectiveness of this approach in terms of various QoS metrics, including availability, response time, cost, reliability, and energy consumption. The suggested approach is further validated by comparing it with existing methods, which confirms its superiority.

In study [13], it introduced a hybrid approach, combining Artificial Neural Network (ANN) and Particle Swarm Optimization (PSO) algorithms, to enhance QoS factors in cloud-edge computing. In order to validate the accuracy and improve the success rate of candidate-composed services and QoS factors using the proposed hybrid algorithm, they have presented a formal verification method based on labeled transition systems. This verification method aims to verify critical Linear Temporal Logic (LTL) formulas. The experimental results demonstrate the exceptional performance of the proposed model, as evidenced by minimal verification time, efficient memory consumption, and the ability to guarantee critical specification rules specified by Linear Temporal Logic (LTL) formulas. Additionally, they have observed that the proposed model outperforms other service composition algorithms in terms of response time, availability, price, and fitness function value.

The authors in study [14] suggested a novel approach for QoS-aware service composition in the context of Fog-IoT computing, leveraging a multi-population genetic algorithm. In order to address the challenges associated with the architecture of IoT-cloud systems, they have adopted a five-layered architecture, with a particular emphasis on the transport layer within a Fog computing environment. The transport layer has been further divided into four sub-layers, namely security, storage, pre-processing, and monitoring, which offer promising advantages. In addition, the authors have implemented a multi-population genetic algorithm (MPGA) based on a QoS model, encompassing seven dimensions: cost, response time, reliability, reputation, location, security, and availability. The experimental results demonstrate the effectiveness of the MPGA in terms of fitness value and execution time, particularly when applied to a case study involving ambulance emergency services. These findings highlight the efficiency and suitability of the proposed approach for handling real-world scenarios.

The paper in [15] have proposed an optimization algorithm called PD3QND, which is based on deep reinforcement learning. PD3QND incorporates various techniques, including Deep Q-Network (DQN), noise networks, prioritized experience replay, double dueling architecture, and demonstration learning. Experimental results demonstrate that PD3QND outperforms heuristic algorithms and methods such as DQN in dynamic QoS environments within the manufacturing IoT domain. PD3QND effectively balances the trade-off between exploitation and exploration, adapting to changes in QoS requirements. It successfully addresses the cold start problem and exhibits robust and efficient search capabilities within the solution space. Moreover, PD3QND demonstrates faster convergence speed and greater adaptability, providing a promising approach for optimizing manufacturing IoT systems.

The study [16] introduced a framework for the composition of IoT services in fog-based IoT networks, using a multi-objective optimization methodology. The proposed solution utilizes the Non-dominated Sorting Genetic Algorithm II (NSGA-II) algorithm. In this framework, the cloud controller is responsible for distributing application requests to fog servers in real-time. When an application request is received, fog servers break it down into IoT service requests and then further split them into specific time intervals. The suggested approach optimizes each time frame independently, taking into account parameters such as QoS, energy consumption, and fairness. The experimental assessment findings provide evidence of the efficacy of the suggested strategy. It effectively maximizes energy efficiency and equity while maintaining quality of service standards, without any decline in performance. This framework offers a promising solution for efficient IoT service composition in fog-based IoT networks.

This paper introduced in [19] a novel method for service composition in IoT environments that prioritizes the QoS through a multi-objective fuzzy-based hybrid algorithm. The approach combines the strengths of fuzzy logic to handle uncertainties and multi-objective optimization to balance conflicting goals such as minimizing latency, maximizing throughput, and ensuring reliability. The proposed method enhances the flexibility and adaptability of IoT service compositions by dynamically adjusting to varying network conditions and service requirements. Critical evaluation highlights its significant contribution to improving service reliability and user satisfaction in IoT networks. However, the complexity of the hybrid algorithm and its computational overhead may pose challenges for implementation in resource-constrained IoT devices.

The IMBA paper [20] presented an innovative bat-inspired algorithm specifically designed to optimize resource allocation in IoT networks, particularly within the IoT-mist computing paradigm. This nature-inspired algorithm leverages the echolocation behavior of bats to efficiently search for optimal resource allocation solutions, thereby addressing the inherent constraints and dynamic nature of IoT environments. The critical strengths of the IMBA lie in its ability to adaptively balance exploration and exploitation, ensuring efficient utilization of computational resources and reducing latency. Notably, the algorithm demonstrates significant improvements in resource allocation efficiency and network performance. However, a key issue is the algorithm's potential sensitivity to parameter settings, which may require extensive tuning for different IoT scenarios. To strengthen its scientific foundation, future studies could investigate the robustness of IMBA across

diverse IoT applications and environments, and explore automated parameter tuning mechanisms to enhance its ease of deployment.

This paper in [21] addressed the critical challenge of efficient resource allocation for IoT requests within a hybrid fog–cloud environment. It proposes a resource allocation strategy that dynamically distributes IoT workloads between fog and cloud resources based on real-time analysis of resource availability, latency requirements, and energy consumption. The strategy aims to optimize performance by leveraging the proximity of fog computing to IoT devices while utilizing the extensive computational power of the cloud for more intensive tasks. Key contributions of this work include a substantial reduction in response times and energy consumption, which are crucial for latency-sensitive and resource-constrained IoT applications. The research effectively demonstrates the benefits of hybrid fog–cloud architectures in enhancing QoS for IoT services. Nevertheless, the proposed strategy's dependency on accurate real-time data and its complexity in managing hybrid environments might present practical challenges. Further research could focus on refining the allocation algorithms to handle larger-scale IoT deployments and ensuring robustness in dynamic and heterogeneous IoT ecosystems.

III. METHODOLOGY

The suggested service composition approach integrates cloud and fog computing within an IoT ecosystem to capitalize on their strengths. Positioned at the network periphery, the fog layer facilitates instantaneous processing and analysis utilizing compact, energy-efficient devices. These devices have the capability to execute intelligent processes by invoking the outcomes of their calculations. Conversely, the cloud system is comprised of robust servers housed in centralized facilities and is responsible for managing resource-intensive operations such as big data analytics and machine learning. The amalgamation of cloud and fog computing provides numerous advantages. On the other hand, the robust processing and storage features of the cloud layer enable the efficient management of large data volumes and execution of complex computations, thereby benefiting tasks that necessitate substantial resources. The real-time processing abilities of the fog layer effectively minimize latency and enhance response times, rendering it well-suited to time-sensitive applications. This model significantly optimizes data analysis and processing, resulting in notable improvements in scalability, effectiveness, and performance for organizations. Furthermore, it has the potential to yield better security measures and cost reductions. As illustrated in Fig. 1, the model follows a three-layered architecture, which encompasses the IoT, fog, and cloud layers. The IoT tier consists of sensors and intelligent units that together form the IoT environment. The fog layer functions as an intermediary between cloud services and IoT devices, where fog nodes efficiently receive and process requests. Depending on the immediate needs of applications, queries may be directly managed inside the fog layer or forwarded to the cloud layer for further analysis.

Within the domain of IoT service composition, IoT nodes are categorizable into two distinct classes: Abstract Services (ASs) and Concrete Services (CSs). Abstract services provide higher-level descriptions that encapsulate the functionalities provided by a group of concrete services. These abstract services offer a more generalized representation of the services available within the IoT system. On the other hand, concrete services refer to specific, invocable services offered by individual IoT components.

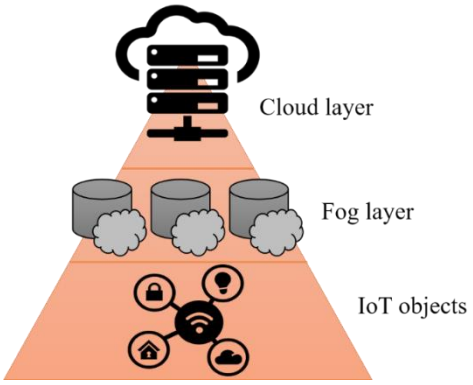

Fig. 1. System architecture.

Concrete services exhibit a dual nature, comprising functional attributes and non-functional aspects. Functional characteristics encapsulate the explicit functionalities that a service provides. These characteristics outline the core functionality and purpose of the service. On the other hand, non-functional features encompass various QoS factors associated with the service. These aspects include parameters such as energy, cost, reliability, and response time. Non-functional features provide important criteria for evaluating and selecting services based on their performance and operational characteristics. The composition of IoT services entails the creation of composite services through the interconnection of atomic services using diverse structural patterns. Within a composite service, various structural patterns can be employed to specify the interactions among atomic services. Six discernible forms of composition structure patterns include:

- Sequential: Atomic services are executed in a sequential order.
- AND split (Fork): The execution is split into multiple branches, and all branches are executed concurrently.
- XOR split (Conditional): The execution splits into multiple branches, but only one branch is selected and executed based on a condition.
- Loop: An atomic service or a set of atomic services is repeated until a specific condition is met.
- AND join (Merge): Multiple branches are joined together, and the execution continues after all branches have been completed.
- XOR join (Trigger): The execution waits for a condition to be satisfied before continuing.

In the mentioned context, the focus is on the sequential model of composition. Nevertheless, it is crucial to emphasize that alternative composition schemes possess the potential to be streamlined or converted into sequential schemes through established methodologies, as indicated in the reference. Fig. 2 provides a visual representation of how IoT services are

composed, illustrating the interconnection between atomic services. In the IoT, evaluating QoS parameters is crucial to differentiate between services and make informed decisions. The paper adopts a perspective on service composition that regards service sequences as workflows. QoS concerning IoT services refers to non-functional attributes, including reliability, availability, response time, and throughput. QoS values can be provided by service providers or determined by the users based on their specific requirements. Users often exhibit diverse preferences and requisites concerning factors such as packet loss, resource costs, reliability, and response time among other factors. The study concentrates on evaluating services based on four QoS properties as follows:

- Energy: This indicator assesses the energy efficiency and sustainability of a service by measuring the amount of energy it consumes throughout its operating period.
- Cost: This factor represents the monetary expenditure required for users to acquire the desired service, encompassing the financial dimension of utilizing the service.
- Reliability: Reliability is a measure of a service's capacity to operate with precision and consistency, without any faults or malfunctions, in order to achieve the desired results.
- Availability: This measure reflects how long a service remains available over a certain period, indicating the dependability and availability of the service for users.

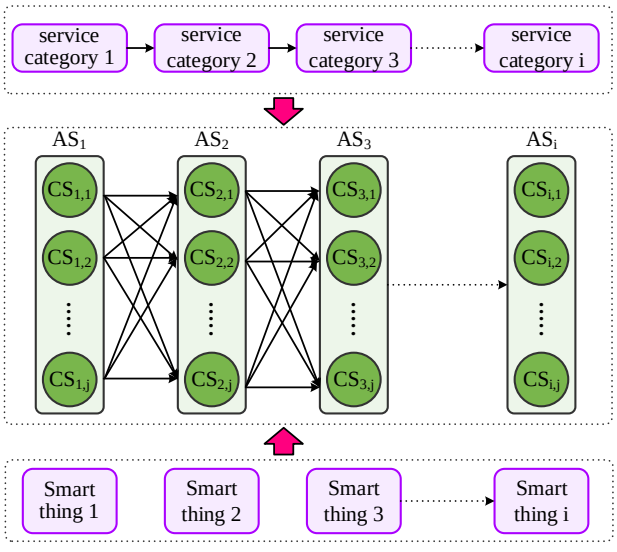

Fig. 2. The process of IoT service composition.

The method integrating cloud and fog computing for IoT service composition was chosen to capitalize on the complementary strengths of these paradigms, addressing the inherent limitations of IoT systems. This hybrid approach leverages the fog layer's ability to perform real-time processing and analysis at the network edge, significantly reducing latency and enhancing response times for time-sensitive applications. The cloud layer, with its robust processing and storage capabilities, efficiently handles resource-intensive tasks such as big data analytics and machine learning, which are beyond the capacity of individual IoT devices. This integration aims to optimize data processing and analysis, thereby improving scalability, effectiveness, and performance for a wide range of applications. Additionally, it enhances energy efficiency, reduces operational costs, and provides better security measures by distributing the computational load between the fog and cloud layers. The method was selected to meet the diverse and dynamic requirements of IoT environments, ensuring a balanced and efficient service composition that can adapt to varying user needs and network conditions.

Table I delineates distinct QoS aggregation functions employed for evaluating the suggested dynamic service composition model. These aggregation functions play a pivotal role in efficiently ascertaining the most favorable service composition aligned with users' desires and needs. This determination considers attributes encompassing availability, reliability, energy, and cost. The study utilizes the Simple Additive Weighting (SAW) technique to convert the combined QoS values, which have varying ranges and units, into a single global value.

TABLE I. QOS AGGREGATION FUNCTIONS FOR SERVICE COMPOSITION

Attribute	Function
Energy	$q_e(S) = \sum_{i=1}^{n} q_e(s_i)$
Cost	$q_c(S) = \sum_{i=1}^{n} q_c(s_i)$
Reliability	$q_r(S) = \sum_{i=1}^{n} q_r(s_i)$
Availability	$q_a(S) = \sum_{i=1}^{n} q_a(s_i)$

The study focuses on an objective function aimed at minimizing. It employs positive and negative normalization formulas, as indicated in Eq. (1) and Eq. (2), correspondingly Cs. Qi represents the ith attribute value for a particular concrete service, while $Qimax$ and $Qimin$ reflect the greatest and lowest values of the ith attribute across all the concrete services in the service candidate set. In order to assess the suitability of a certain solution, the study builds a fitness function based on Eq. (3). Each attribute of QoS inside an atomic service is weighted by W_i. The weights are bounded between the range of 0 and 1 ($0 \leq W \leq 1$), and the total sum of all weights $\sum 4i=1 W_i$ is equivalent to 1. Q_i denotes the cumulative attribute value of the solution that corresponds to the ith QoS attribute.

$$N_{Cs.Q^i} = \begin{cases} \dfrac{Q^i{}_{\max} - cs.Q^i}{Q^i{}_{\max} - Q^i{}_{\min}}, Q^i{}_{\max} \neq Q^i{}_{\min} \\ 1, Q^i{}_{\max} = Q^i{}_{\min} \end{cases} \quad (1)$$

$$N_{Cs.Q^i} = \begin{cases} \dfrac{cs.Q^i - Q^i\,\min}{Q^i{}_{\max} - Q^i{}_{\min}}, Q^i{}_{\max} \neq Q^i{}_{\min} \\ 1, Q^i{}_{\max} = Q^i{}_{\min} \end{cases} \quad (2)$$

$$Fitness = \sum_{i=1}^{4} W_i * Q_i \quad (3)$$

In the context of IoT service composition, the energy consumption of candidate services is an important factor that significantly affects the devices hosting those services. To facilitate the selection of services with better energy-saving effects, each candidate service is associated with an energy consumption parameter. The energy profile of a specific service represented as EproFile(Cs_{ij}), consists of several variables. One of the factors is the service's autonomy, denoted as SA(Cs_{ij}). The autonomy of a service is determined by the power capacity of the device that hosts the service. The calculation is performed using Eq. (4), where CE(Cs_{ij}) represents the current energy level of the battery-powered device housing the service Cs_{ij}, and ET(Cs_{ij}) represents the energy threshold of the battery-powered device capable of hosting the service Cs_{ij}.

$$SA(Cs_{ij}) = CE(Cs_{ij}) - ET(Cs_{ij}) \quad (4)$$

The energy consumption for operating a concrete service, represented as EC(Cs_{ij}), remains fixed and can be determined using Eq. (5). The equation defines RT(Cs_{ij}) as the mean duration of the service Cs_{ij}, and ECR(Cs_{ij}) as the rate at which energy is used.

$$EC(Cs_{ij}) = ECR(Cs_{ij}) \times RT(Cs_{ij}) \quad (5)$$

Thus, Eq. (6) is used to compute the energy profile for the service Cs_{ij}, considering the variables of autonomy and energy consumption.

$$E\ Pr\ o\ Fi(Cs_{ij}) = \frac{EC(Cs_{ij})}{SA(Cs_{ij})} \quad (6)$$

A low energy profile suggests that the IoT device running the service Cs_{ij} has a comparatively extended lifespan. Hence, Eq. (7) is used to compute the energy profile for composite services. Within this equation, the variable xi denotes the specific component chosen from the abstract service class, corresponding to the ith position.

$$CE\ Pr\ o\ Fi(x) = \sum_{i=1}^{n} E\ Pr\ o\ File(x^i) \quad (7)$$

The ABC algorithm is a popular optimization algorithm that draws inspiration from the foraging activity of bees. It employs the principles of labor division and knowledge sharing to address both continuous and discrete optimization issues. ABC is renowned for its straightforwardness, few control settings, and robust stability. The population in ABC consists of three distinct categories of bees: worker bees, observer bees, and scouts. These bees are linked to three exploration procedures: the employed bee stage, the onlooker bee stage, and the scout stage. The quantity of engaged bees is the same as the quantity of spectator bees.

The process begins by establishing an initial population of n solutions, denoted as $X_i = (x_{i,1}, x_{i,2}, ..., x_{i,D})$, where i ranges from 1 to n. In this context, n represents the size of the population, whereas D refers to the size of the dimension. During the employed bee stage, each individual bee is assigned the task of investigating the surrounding area of a particular solution. The employed bee associated with the i^{th} solution, X_i, develops a new solution, V_i, following a search method outlined in Eq. (8).

$$v_{i,j} = x_{i,j} + \varphi_{i,j}(x_{i,j} - x_{k,j}) \quad (8)$$

In Eq. (1), φ is a stochastic variable that takes on values uniformly distributed in the interval [-1, 1], x_k denotes a distinct solution chosen at random from the group, with the exception of the current solution X_i (where k is not equal to i), j is a number selected at random from the set of integers ranging from 1 to D, and D is the size of the dimension. The conventional ABC algorithm utilizes an elite selection technique to decide whether V_i or X_i is selected for the subsequent iteration. If V_i is superior to X_i, it supplants X_i in the population. Eq. (8) states that the disparities between V_i and X_i are only present in the j^{th} dimension. For the other D-1 dimensions, the values of V_i and X_i are identical. As a result, V_i and X_i exhibit a high degree of similarity, and the step size for the present search is minimal since it only investigates a single dimension. Consequently, the search process can see a decrease in speed.

During the observer bee stage of the ABC algorithm, the primary emphasis is placed on conducting an extensive search rather than examining the surroundings of all solutions inside the swarm. The solutions chosen in this phase are determined by their selection probabilities, which are computed using Eq. (9). The probability of selecting the i-th option, denoted as $probi$, is derived based on the fitness value of X_i, which is calculated using Eq. (10). Eq. (10) calculates the fitness value of the solution X_i, where $fVal_i$ represents the function value of X_i.

$$prob_i = \frac{fitness_i}{\sum_{i=1}^{SN} fitness_j} \quad (9)$$

$$fitness_i = \begin{cases} \frac{1}{1+fval}, iffval_i \geq 0 \\ 1 + |fval_i|, otherwise \end{cases} \quad (10)$$

The observer bees, like the worker bees, generate a new solution V_i using Eq. (8) and then evaluate its function value against X_i. If V_i is superior to X_i, it supplants X_i in the population for the subsequent iteration.

In the elite selection approach, if V_i is inferior to X_i, it signifies that the enhancement of X_i is seen as a failure. If V_i is superior to X_i, the enhancement is considered successful. A counter, denoted as $trail_i$, is used to monitor the number of failures for each solution in the population. If the value of $trail_i$ grows quite big, it indicates that X_i could have reached a local minimum and is unable to move away from it. In such instances, X_i is reset using Eq. (11).

$$x_j = Low_j + r_j.(Up_j - Low_j) \quad (11)$$

In Eq. (11), r_j is a random number within the range [0, 1], and [Low, Up] represents the definition domain of the problem.

A technique called dimensional perturbation with a DR approach is suggested to enhance the traditional ABC algorithm by addressing the problem of delayed convergence and improving exploitation. At first, the number of dimension perturbations is assigned a high value, which must be less than the dimension size (D). By increasing the number of dimension perturbations, it is possible to create greater disparities between

children and their parent solutions. This aids in expediting the search process and swiftly identifying superior options.

As the iterations continue, the frequency of dimension disturbances gradually reduces. The objective of reducing the number of dimension perturbations is to minimize the differences between offspring and their parent solutions, hence enhancing the identification of more precise solutions. Eq. (12) governs the dynamic updating of the number of dimension perturbations, which is indicated as DP(t).

$$DP(t) = (1 - \frac{t}{T_{max_0}}) \quad (12)$$

Eq. (12) defines T_{max} as the maximum number of repetitions, and D_0 as the beginning value for dimension perturbation. The suggested technique sets the value of D_0 as the product of λ and D, where λ is a parameter that falls within the range of (0,1). At the start of the procedure, at iteration 0, (0) is equivalent to $D0$. During the course of the iterations, $D(t)$ steadily diminishes from $D0$ to zero. However, if the value of $D(t)$ drops below 1, the number of dimension perturbations will be fewer than one, which is considered unacceptable. In order to prevent this scenario, a simple approach is used, as shown in Eq. (13).

$$DP(t) = \begin{cases} DP(t), if DP(t) \geq 1 \\ 1, otherwise \end{cases} \quad (13)$$

IV. Results and Discussion

The simulation was performed using a CPU core i5 2.5 GHz with 8GB RAM, and the programming language used was MATLAB R2020a. MATLAB is widely recognized as one of the best tools for simulating metaheuristic algorithms, and it is commonly employed in research papers. The QWS dataset was utilized, which consists of QoS measurements for 2507 service implementations. To deal with fluctuations in QoS values under dynamic IoT conditions of service delivery, a technique randomly updates the QoS status after each service iteration by multiplying each QoS value with a random integer between 0.9 and 1.1.

The assessment of the suggested approach comprises four essential quality of service metrics: cost, energy, reliability, and availability. The findings clearly indicate the exceptional efficacy of the suggested approach. The simulation experiments were performed using 10, 30, 50, 70, and 100 service classes, each representing a specific job, and a pool of 50 potential services. Fig. 3 presents a comparison of the energy parameter of the proposed technique with the methods specified in studies [9], [17], [18]. Fig. 4 and Fig. 5 depict the logarithm (base 10) of the attained outcomes for the availability and reliability metrics, correspondingly. The figures demonstrate that the suggested strategy produces good results in all three indicated parameters. Fig. 6 demonstrates that the cost parameter of the suggested technique is lower compared to other algorithms. As the quantity of requests grows, this parameter undergoes a substantial reduction. This may be credited to the efficient choice of services facilitated by the suggested algorithm.

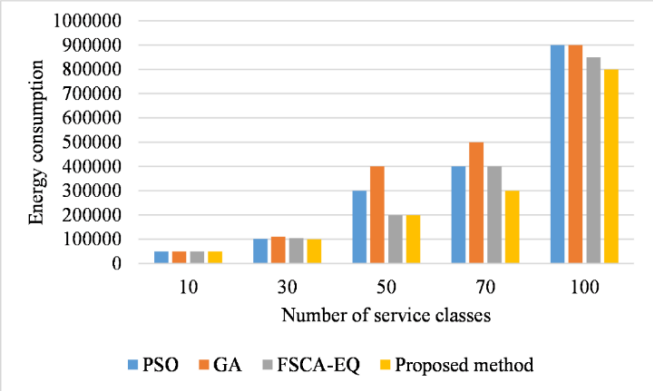

Fig. 3. Energy consumption comparison.

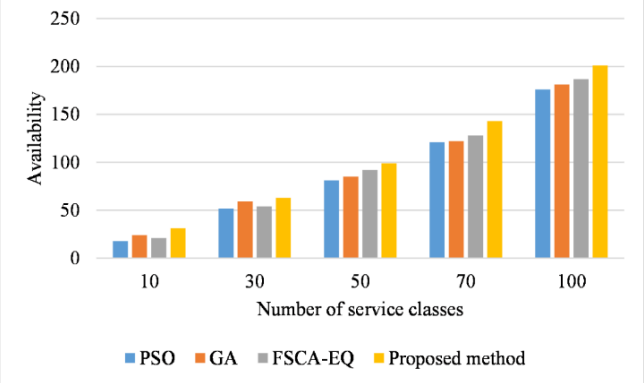

Fig. 4. Availability comparison.

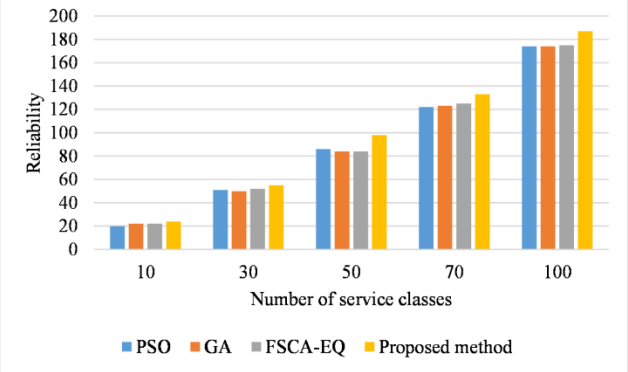

Fig. 5. Reliability comparison.

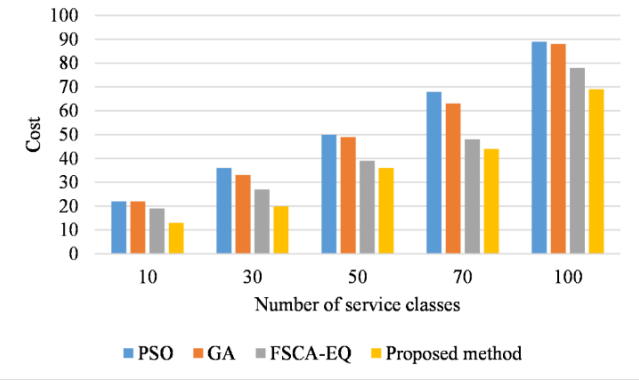

Fig. 6. Cost comparison.

The performance of the proposed method was assessed against benchmark algorithms considering four key parameters, namely variation time rate (Jitter), Packet Delivery Ratio (PDR), throughput, and average end-to-end delay. Jitter is the fluctuation in delay that occurs in the transmission of data packets between two nodes. Jitter is a prominent measurement that exerts a substantial impact on real-time applications. PDR is the proportion of successfully received data packets to the total number of data packets sent. Throughput denotes the rate at which data may be transmitted via a communication channel, measured as a ratio of data transferred to time. End-to-end delay refers to the average duration it takes for a data packet to reach its destination node, including the time required to compute its arrival time.

Fig. 7 depicts the rate of change of the end-to-end packet delay (jitter) for the proposed technique, as compared to the benchmark algorithms PSO, GA, and FSCA-EQ. This figure demonstrates that the curve of the suggested method constantly surpasses other state-of-the-art procedures. Our technique is regarded as an asymptotically optimum algorithm. Consequently, the outcomes are not excessively responsive to the original control values.

Fig. 8 illustrates the fluctuation of PDR for the suggested technique compared to other cutting-edge algorithms. Our solution clearly outperforms other methods in terms of providing a high PDR for data packets via the network.

Fig. 9 presents a comparison of approaches based on the overall throughput. Our approach achieved a significant improvement of around 53% and 75% compared to GA and FSCA-EQ, respectively. At first, the FSCA-EQ and GA are inherently parallel. This parallelism enables the identification of all potential options for achieving an ideal solution in several directions. Nevertheless, these strategies do not provide a universal solution for wireless network difficulties, particularly when they are time-related. The efficiency of FSCA-EQ and GA is contingent upon both the population size and the values of the input control parameters. Consequently, this has a negative impact on both the predicted operating time and the computational cost. This accounts for the decrease in the slope of the FSCA-EQ and GA curves when the service size is enlarged. Conversely, the proposed method curve has demonstrated a very high throughput rate as it remains unaffected by the change in population size.

According to Fig. 10, our solution surpasses previous methods by providing decreased latency as the arrival rate increases. It demonstrates a significant improvement of around 65%, 58%, and 71% compared to GA, FSCA-EQ, and PSO, respectively. The stability of the suggested approach was the defining characteristic of its curve, surpassing that of other benchmark algorithms. It should be noted that the behavior of the suggested technique remains mostly unchanged when the scale of the network is altered. The benchmark algorithms, namely FSCA-EQ, GA, and PSO, exhibit a linear trend where a rise in the service scale is accompanied by an increase in the delay time. In contrast, our approach exhibits consistent performance even when the scope of the services is expanded.

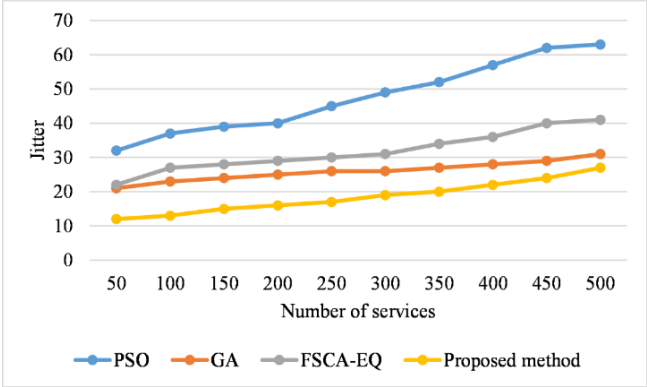

Fig. 7. Jitter comparison.

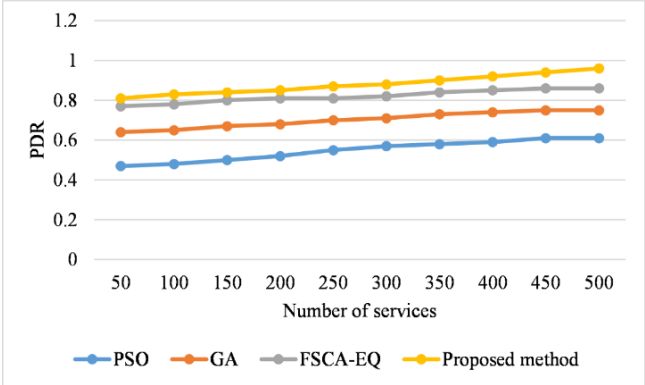

Fig. 8. PDR comparison.

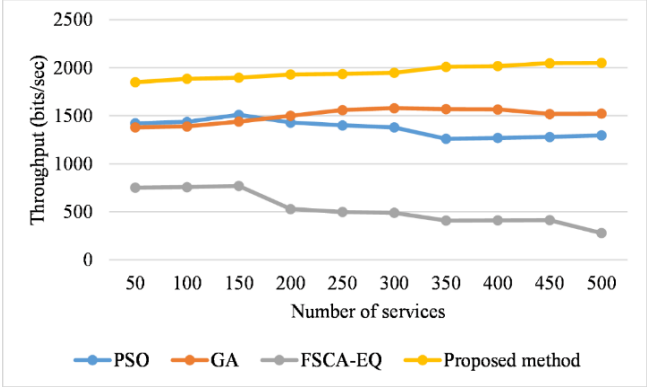

Fig. 9. Throughput comparison.

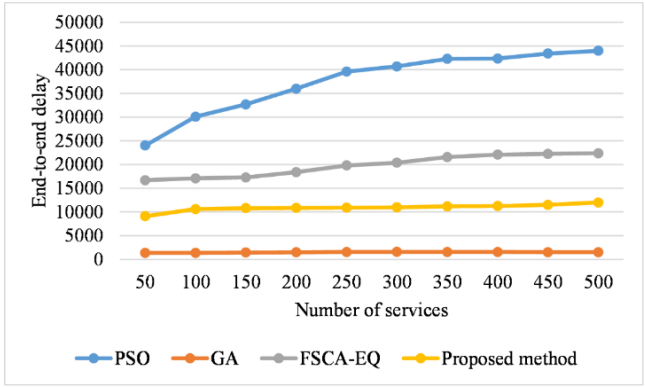

Fig. 10. End-to-end delay comparison.

The research findings highlight the exceptional efficacy and novelty of the proposed service composition approach, integrating cloud and fog computing within an IoT ecosystem. The assessment, which focused on four essential QoS metrics—cost, energy, reliability, and availability—demonstrated significant improvements across all parameters when compared to existing methods. Specifically, the proposed technique showed lower energy consumption and cost, which can be attributed to the efficient selection of services facilitated by the algorithm. The simulation experiments, spanning multiple service classes and requests, consistently indicated superior performance in availability and reliability metrics. Additionally, the proposed method outperformed benchmark algorithms in terms of jitter, Packet Delivery Ratio (PDR), throughput, and end-to-end delay. Notably, the technique achieved a substantial improvement in throughput (53% and 75% higher than GA and FSCA-EQ, respectively) and demonstrated remarkable stability and lower latency even as the service scale increased. This consistency and robustness in performance, unaffected by network scale changes, underscore the method's capability to effectively handle diverse and dynamic IoT environments, presenting a significant advancement in IoT service composition. The findings validate the proposed approach as a highly efficient and scalable solution, offering substantial improvements over state-of-the-art algorithms, and confirming its potential to enhance real-time processing, resource allocation, and overall QoS in IoT applications.

V. Conclusion

The proposed method enhances IoT service composition by building on the ABC algorithm, a nature-inspired optimization technique. The study introduces a Dynamic Reduction (DR) methodology to optimize the ABC algorithm, dynamically adjusting the number of dimension perturbations during solution generation. This approach effectively balances the trade-off between exploration and exploitation, fostering diversity in solutions during the initial phases and promoting convergence toward optimal solutions in later iterations. The experimental results highlight substantial improvements with the proposed algorithm: a 17% reduction in average energy consumption, and enhancements in availability and reliability by 10% and 8%, respectively. Additionally, a notable 23% reduction in average cost underscores the economic viability of this approach for QoS-aware service composition in IoT.

However, while these results are promising, there are limitations to consider. The complexity of the DR methodology may pose challenges in terms of computational overhead and implementation in resource-constrained IoT devices. Furthermore, the performance gains observed in controlled experimental settings may not fully translate to real-world environments with diverse and dynamic IoT applications.

Future work should focus on addressing these limitations by optimizing the computational efficiency of the DR methodology and validating its performance in varied real-world scenarios. Potential areas for improvement include exploring automated parameter tuning to enhance adaptability and investigating the integration of this approach with emerging edge computing paradigms. Additionally, expanding the scope of QoS metrics to include other critical factors such as security and user satisfaction could provide a more comprehensive evaluation of the proposed method's effectiveness. By addressing these areas, the robustness and applicability of the proposed service composition approach can be further strengthened, paving the way for more reliable and efficient IoT systems.

Acknowledgment

This work was supported by the Changde Vocational and Technical College Key Project (ZY2304).

References

[1] B. Pourghebleh and N. J. Navimipour, “Data aggregation mechanisms in the Internet of things: A systematic review of the literature and recommendations for future research,” Journal of Network and Computer Applications, vol. 97, pp. 23–34, 2017.

[2] B. Pourghebleh and V. Hayyolalam, “A comprehensive and systematic review of the load balancing mechanisms in the Internet of Things,” Cluster Comput, vol. 23, no. 2, pp. 641–661, 2020.

[3] B. Pourghebleh, V. Hayyolalam, and A. Aghaei Anvigh, “Service discovery in the Internet of Things: review of current trends and research challenges,” Wireless Networks, vol. 26, no. 7, pp. 5371–5391, 2020.

[4] F. Kamalov, B. Pourghebleh, M. Gheisari, Y. Liu, and S. Moussa, “Internet of medical things privacy and security: Challenges, solutions, and future trends from a new perspective,” Sustainability, vol. 15, no. 4, p. 3317, 2023.

[5] P. Kumar, R. Kumar, G. P. Gupta, R. Tripathi, A. Jolfaei, and A. K. M. N. Islam, “A blockchain-orchestrated deep learning approach for secure data transmission in IoT-enabled healthcare system,” J Parallel Distrib Comput, vol. 172, pp. 69–83, 2023.

[6] V. Hayyolalam, B. Pourghebleh, M. R. Chehrehzad, and A. A. Pourhaji Kazem, “Single-objective service composition methods in cloud manufacturing systems: Recent techniques, classification, and future trends,” Concurr Comput, vol. 34, no. 5, p. e6698, 2022.

[7] V. Hayyolalam, B. Pourghebleh, A. A. Pourhaji Kazem, and A. Ghaffari, “Exploring the state-of-the-art service composition approaches in cloud manufacturing systems to enhance upcoming techniques,” The International Journal of Advanced Manufacturing Technology, vol. 105, pp. 471–498, 2019.

[8] E. Teniente, “Iot semantic data integration through ontologies,” in 2022 IEEE International Conference on Services Computing (SCC), IEEE, 2022, pp. 357–358.

[9] Z. Chai, M. Du, and G. Song, “A fast energy-centered and QoS-aware service composition approach for Internet of Things,” Appl Soft Comput, vol. 100, p. 106914, 2021.

[10] R. Xiao, Z. Wu, and D. Wang, “A Finite-State-Machine model driven service composition architecture for internet of things rapid prototyping,” Future Generation Computer Systems, vol. 99, pp. 473–488, 2019.

[11] M. Sun, Z. Zhou, J. Wang, C. Du, and W. Gaaloul, “Energy-efficient IoT service composition for concurrent timed applications,” Future Generation Computer Systems, vol. 100, pp. 1017–1030, 2019.

[12] S. Sefati and N. J. Navimipour, “A qos-aware service composition mechanism in the internet of things using a hidden-markov-model-based optimization algorithm,” IEEE Internet Things J, vol. 8, no. 20, pp. 15620–15627, 2021.

[13] M. Hosseinzadeh et al., “A hybrid service selection and composition model for cloud-edge computing in the internet of things,” IEEE Access, vol. 8, pp. 85939–85949, 2020.

[14] I. Aoudia, S. Benharzallah, L. Kahloul, and O. Kazar, “QoS-aware service composition in Fog-IoT computing using multi-population genetic algorithm,” in 2020 21st International Arab Conference on Information Technology (ACIT), IEEE, 2020, pp. 1–9.

[15] Y. Chen, L. Cheng, and T. Wang, “Deep Reinforcement Learning for QoS-Aware IoT Service Composition: The PD3QND Approach,” in 2023 IEEE 14th International Conference on Software Engineering and Service Science (ICSESS), IEEE, 2023, pp. 38–41.

[16] M. Guzel and S. Ozdemir, “Fair and energy-aware IoT service composition under QoS constraints,” J Supercomput, vol. 78, no. 11, pp. 13427–13454, 2022.

[17] A. Naseri and N. Jafari Navimipour, “A new agent-based method for QoS-aware cloud service composition using particle swarm optimization algorithm,” J Ambient Intell Humaniz Comput, vol. 10, pp. 1851–1864, 2019.

[18] M. Chen, Q. Wang, W. Sun, X. Song, and N. Chu, “GA for QoS satisfaction degree optimal Web service composition selection model,” in 2019 6th International Conference on Behavioral, Economic and Socio-Cultural Computing (BESC), IEEE, 2019, pp. 1–4.

[19] Hamzei, Marzieh, Saeed Khandagh, and Nima Jafari Navimipour. "A quality-of-service-aware service composition method in the internet of things using a multi-objective fuzzy-based hybrid algorithm." Sensors 23, no. 16 (2023), 7233.

[20] Almudayni, Ziyad, Ben Soh, and Alice Li. "IMBA: IoT-Mist Bat-Inspired Algorithm for Optimising Resource Allocation in IoT Networks." Future Internet 16, no. 3 (2024), 93.

[21] Afzali, Mahboubeh, Amin Mohammad Vali Samani, and Hamid Reza Naji. "An efficient resource allocation of IoT requests in hybrid fog–cloud environment." The Journal of Supercomputing 80, no. 4 (2024), 4600-4624.