A Multi-Population Genetic Algorithm for Adaptive QoS-Aware Service Composition in Fog-IoT Healthcare Environment

Idir Aoudia[1], Saber Benharzallah[2], Laid Kahloul[1], and Okba Kazar[1]
[1]LINFI Laboratory, Biskra University, Algeria
[2] LAMIE Laboratory**,** Computer Sciences department, Batna 2 University, Algeria

Abstract: *The growth of Internet of Thing (IoT) implies the availability of a very large number of services which may be similar or the same, managing the Quality of Service (QoS) helps to differentiate one service from another. The service composition provides the ability to perform complex activities by combining the functionality of several services within a single process. Very few works have presented an adaptive service composition solution managing QoS attributes, moreover in the field of healthcare, which is one of the most difficult and delicate as it concerns the precious human life.In this paper, we will present an adaptive QoS-Aware Service Composition Approach (P-MPGA) based on multi-population genetic algorithm in Fog-IoT healthcare environment. To enhance Cloud-IoT architecture, we introduce a Fog-IoT 5-layared architecture. Secondly, we implement a QoS-Aware Multi-Population Genetic Algorithm (P-MPGA), we considered 12 QoS dimensions, i.e., Availability (A), Cost (C), Documentation (D), Location (L), Memory Resources (M), Precision (P), Reliability (R), Response time (Rt), Reputation (Rp), Security (S), Service Classification (Sc), Success rate (Sr), Throughput (T). Our P-MPGA algorithm implements a smart selection method which allows us to select the right service. Also, P-MPGA implements a monitoring system that monitors services to manage dynamic change of IoT environments. Experimental results show the excellent results of P-MPGA in terms of execution time, average fitness values and execution time / best fitness value ratio despite the increase in population. P-MPGA can quickly achieve a composite service satisfying user's QoS needs, which makes it suitable for a large scale IoT environment.*

Keywords: *IoT, service composition, adaptability, context; QoS, Fog-IoT computing, Healthcare.*

Received February 20, 2021; *accepted March 7, 2021*
https://doi.org/10.34028/iajit/18/3A/10

1. Introduction

The Internet of Things (IoT) is an emerging technology which can affect the industrial sector, the environment, the social sector and especially the health sector. IoT consists to connect a large number of daily objects to the internet [5], each object has his own identity and offers functionalities in the form of a service. The service composition provides the ability to perform complex activities by combining the functionality of several services within a single process [5], and the composite services form a new service that can be reused in another composition in order to answer to the complicated user's demands.

For reasons of conductivities, failures, battery charge and others, the availability of these services is unpredictable [3]. This unpredictability of availability and the dynamic evolution of user needs, mean that the service composition must manage this dynamism and adapt to new configurations not provided for in the conception. Adaptive service composition means modifying the system to allow it to behave correctly in different contexts to ensure the availability of the services offered, in order to respond to a situation not expected during the design phase.In brief, an adaptive composition should include the following 4 goals:

1. Recover from unexpected situations so that the application continues the intended execution, or at least ends in a consistent state, despite the occurrence of a failure.
2. Exploit new emerging opportunities to improve the quality of the chosen solution at any stage of execution.
3. Prevent future changes and misconduct by taking corrective action early, because a late reaction (i.e., after faulty or quality-impairing services have been performed) may result in an inability to find a suitable recovery from this point, or a selected new solution of lower quality than could be achieved by reacting earlier to changes.
4. Keep triggered adaptations transparent to the end user without downtime, as an interruption in the performance of the composite service could be highly undesirable, especially in time-sensitive applications.

The IoT is becoming more and more popular in several applications, however nowadays, if we take for example the health sector, nearly 2 million people are losing their

lives around the world due to the backlog in healthcare emergency services (traffic problem, location, etc.,). The adaptability will play a very important role in this dynamic IoT environment where a large number of volatile services are available, it will provide the possibility for the application (or composite service) to constantly change in order to meet new contextual constraints.

Adaptation is important in IoT service composition, but to produce an optimal solution for any service composition a Quality of Service (QoS) management must be done. QoS attributes allow us to do the difference between each IoT services, so it's essential to evaluate them to avoid poor quality service composition. Many different IoT architectures have been proposed and the IoT-Cloud architecture was the most piratical method, but as the number of devices using the cloud grows, more problems appear. In our study case, the most of data must be processed in real time, this is due to the fact that most services do not have enough memory to keep all the collected data or they have to make the decision themselves (low unit of calculation). These problems encountered by the Cloud-IoT computing led to the development of the Fog-IoT architecture, the Fog represents a mediator between the IoT services and the cloud [1]. The Fog-IoT architecture was mainly introduced to enhance the architecture of Cloud-IoT systems [30] and it represents a decentralisation of Cloud-IoT for getting the better of Cloud-IoT architecture obstacles.

Considering factors such as QoS attributes, the selection of IoT services for service composition is reduced to a multi-objective optimization problem. The optimization is to made use of the resource in most efficient way. It means maximising or minimising some attribute through an objective function [17], Heuristic algorithms such as Genetic Algorithm (GA), Ant Colony Optimization (ACO), and Particle Swarm Optimization (PSO) are adopted to find the optimal composition of the IoT service.

According to our study entitled (Service composition approaches for internet of things: a review) [5], where we evaluated the most popular service composition approaches in the IoT based on several criteria, we have seen that very few works have presented an adaptive service composition solution managing QoS attributes, moreover in the field of healthcare, which is one of the most difficult and delicate as it concerns the precious human life.

In Dar *et al*. [10, 9], an adaptive service composition has been proposed for an Assisted Living System (ALS) [25], the system is supposed to support people with chronic illness or those who need it for constant medical monitoring (e.g., the elderly) so that they can continue to live independently at home. The proposed architecture uses an event management mechanism to propagate the context information of IoT devices, a dynamic service replacement function is used in case of service failure making the system adaptative and a potential candidate for IoT environments where frequent service failures are observed, however, the latency due to the events managing increases linearly with the increase in the number of sensors. Also, there is no managing of QoS and their monitoring system only reports hardware failures and does not detect incorrect sensor data or software failures.

In [33], a QoS-Aware Selection of IoT services is proposed for Electro Cardio Graphy (ECG) monitoring system. The authors considered IoT framework as a composition of three major components: things, computing, and communication, QoS attributes are associated with each of the three components. The selection framework evaluates the relative importance of the QoS criteria to rank the IoT services. However, only the selection part of service composition was presented, no adaptability framework was illustrated.

Outside the healthcare field, other adaptive approaches have been proposed in [6, 12, 16, 19, 20, 21, 37, 38, 39, 41].

In [38], a novel Petri net-based service composition algorithm has been proposed. To manage dynamic change of the environments, the authors proposed a monitoring algorithm to monitor the IoT system in a cheapest way. A QoS evaluation is used, but only concern three quality properties, i.e., reliability, response time and cost. Experimental results have proved the soundness and correctness of the algorithms but there was no evaluation in real-world service system in IoT.

In [20, 39], a middleware approach for IoT service composition has been proposed. The authors use a monitoring system with only a few surveillance resources to monitor and ensure operations and the robustness of the system. Each component service is represented by an agent, which is responsible for maintaining the QoS information (price, time, reliability, reputation degree, availability) of each service. Experiments evaluated the correctness and robustness of the models and algorithms, but there was no evaluation in real-world service system in IoT.

In [21], a probabilistic service composition approach has been proposed. To monitor the system and make this approach adaptative, the authors use alternative candidate in case of failure (when the probability of success of the selected service is not very high). Also, during the service selection, the Authors describe and analyse QoS attributes (reliability and cost). However, this approach is centralized and does not deal with the constraint of real time.

In [19], an adaptive approach for service composition in IoT has been proposed. The proposed selection mechanism is based on an optimization technique to detect situations where certain compositional requirements are not met or certain services become unavailable or when failures/ exceptions occur. This method tries to reduce the need for human intervention

in reconfiguration of the composition. However, the monitoring system is centralized and use a list of unavailable services. The author also manages some QoS attributes (response time, reliability, availability, location, battery level and reputation) but it's extremely demanding and demands an intensive computing.

In [12], an adaptative service composition approach for IoT is proposed. The generation of the composition scheme is partially achieved at runtime using abstract services provided at design time, this allows flexibility and adaptability without having to build a service mix from scratch. However, no monitoring system for identification and resolution has been proposed. Also, no specific QoS management was cited in this work.

In [37], an event-aware service composition approach for IoT is proposed. The authors present a monitoring system with an override mechanism based on an event strategy, in this approach, changes detection is made at the discovery level to save more time and energy during selection and to ensure that the services will run successfully, some changes may occur at run time. Bayesian learning is used to update dynamic QoS (availability, response time and response time) and automatic service re-composition in case of a service failure. However, there was no evaluation in real-world service system in IoT.

In [6], an adaptive service composition approach for IoT is proposed. The approach effectively manages services changes during runtime, an adaptation is performed as soon as possible and in parallel with the execution process, thus reducing downtime, increasing the chances of successful recovery and providing the most optimal solution based on the current state of the environment. However, no specific QoS management was cited in this current version of this work.

In [16], novel multi-objective service composition for IoT is proposed. The authors instead of optimization a single object, they take maximization of QoS and minimization of cost as two objects. To solve this complex optimization problem, a multi-objective artificial bee colony algorithm is used. However, the proposed method is designed for the cloud services composition, and no adaptability framework was illustrated.

In [41], a hybrid service selection method for IoT was proposed. The authors use optimization approach and QoS attributes to find the best candidate service, the IoT service selection problem is transformed into a single-objective optimization problem adopting a simple weighting method. Experimental shows that the proposed algorithm can satisfy user's needs, however, energy consumption is not considered in the service selection and no adaptability framework was illustrated.

In comparison with the above works, in our previous work [4], we identified some QoS attributes associated with IoT components that best quantifies and analyses the services offered by the IoT's service providers. Also, IoT-cloud architecture problems led us to use the 5-layared architecture implemented on a Fog-IoT system. We believe that this important aspect of service selection and architecture framework should be necessarily defined before designing a service composition algorithm.

Here in this paper, we have identified few more QoS attributes that describes service selection along with those in the previous work. Like in our previous work [4], our main objective is to propose an approach for the adaptive composition of services, user-centric, which adapts to the different situations in the IoT environment and most importantly, this approach can handle our ambulance emergency study case. Hence, the major contribution of this work, which makes it novel, can be summarized as follows:

1. Considering factors such as QoS attributes, the selection of IoT services for service composition is reduced to a multi-objective optimization problem, thus; to solve this problem, a Parallel Multi-Population Genetic Algorithm (P-MPGA) is proposed.
2. P-MPGA algorithm implements a smart selection method which allows us to select the right service.
3. P-MPGA also implements a monitoring system that monitors services to manage dynamic change of IoT environments.
4. Lastly, experimental evaluation is done to verify the robustness of the proposed framework by comparing it to the traditional GA [35] and the Genetic Algorithm (MGA) proposed in [22].

The remainder of this paper is organized as follows: in section 2 we presented our layered architecture based on the Fog-IoT concept, the QoS model used, and the P-MPGA solving algorithm respectively. Section 3 analyses and discusses the experimental results. Finally, conclusions and future work are given in section 4.

2. System Model

In this section, we present our five-layered architecture based on the Fog-IoT concept, the QoS model used, and the P-MPGA solving algorithm.

2.1. System Five-Layered Architecture

There is no single and general agreement on IoT architecture that is agreed in the research world [7]. Many different architectures have been proposed and according to some researchers the IoT architecture has three layers [24, 29, 40], but some researchers favour the four-layer architecture [11]. The authors believe that due to the development of IoT, the three-layer architecture is basic and cannot meet application requirements.

The four-layer architecture has played an important role in the development of IoT, but due to another challenge in IoT regarding security and privacy, the five-layer architecture [23, 31] has also been proposed.

This architecture has three layers like the previous architectures whose names are perception layer, transport layer and application layer. It also has two additional layers. The names of these new proposed layers are Processing layer and Business layer. It is considered that this architecture has the ability to meet the requirements of IoT, it also has the ability to secure IoT applications. Below are the 5 layers of this architecture:

1. "Business" layer: Manages the entire system, including applications as well as user privacy. (Management & monitoring)
2. Application: Responsible for providing services specific to the application for the user (Interface).
3. Processing: Store, analyse and process the huge amount of data using databases etc... according to user requirements.
4. Transport: Transfer of sensor data between the different layers via networks such as wireless, 3G, LAN, Bluetooth, RFid and NFC
5. Perception: Detect and collect information on the environment (by sensors)

The hierarchy of all proposed layer architectures of the Internet of Things (IoT) is illustrated in Figure 1, which shows the IoT layer architectures consisting of three layers, four layers and five layers respectively. We also consider that this 5-layer architecture can meet the requirements of IoT and satisfy a maximum of the criteria cited in our study entitled "Service composition approaches for internet of things: a review" [5].

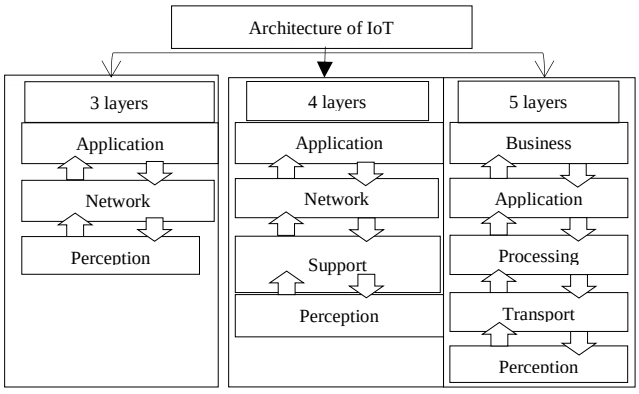

Figure 1. IoT layered architectures.

In our work we are most interested in the processing layer, this layer is also called the middleware layer. It collects the information sent from the transport layer, performing processing on the collected information. It's responsible for removing extra information that doesn't make sense and extracts useful information. However, it also removes the problem of big data in IoT, or a large amount of information is received, which can affect the performance of IoT.

In general, information is sent from local storage to cloud storage where all objects send the collected information Figure 2. Finally, using the information gathered, appropriate action is taken. It is not required that the action always be performed using this information, but we can also remotely manage and control objects and machines and use the information to keep records for future use.

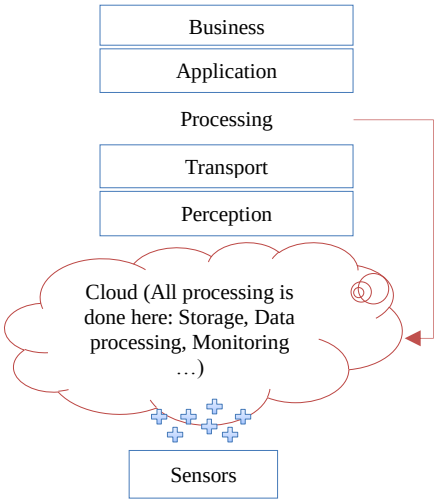

Figure 2. The relation between processing layer and the cloud.

The IoT-Cloud architecture is a practical method for a few devices. But as the number of devices using the cloud grows, so does the use of its bandwidth, latency, and that also involves:

- Don't fit in systems that must receive immediate and real-time actions.
- There will be latency during data transfers.
- Risk of overloading the Cloud (number of growing objects...).
- Requires a high bandwidth (too much data to transmit).
- Requires always to be connected to the internet.
- Cloud security issue (Privacy).
- Problem of scalability.

The problems encountered by the IoT-Cloud architecture led to the development of the fog to overcome these obstacles in the best possible way [25]. Fog computing is a layer that resides between the cloud service and local devices. Fog nodes will be distributed geographically to provide services to their required local devices. Each node will be able to perform calculation tasks and will be able to provide services related to the data collected in the area under their control. In summary, Fog Computing represents a decentralization of cloud computing Figure 2.

As we said in section 1, the healthcare field is one of the most difficult and delicate because it concerns the lives of people. IoT with Cloud Computing has improved the quality of life for patients; however, this architecture is often too reductive and unsuitable for many emerging healthcare applications with critical requirements. Fog Computing may be the solution to the problem [26], it allows low and predictable response times, which can often mean the difference between life

and death for patients; it ensures that at least the most critical part of the overall service is always available to the patient, also in the presence of hostile environments with intermittent or no network connectivity to the Cloud; it protects sensitive health-related data by storing it locally rather than sending it to the cloud via the internet.

The use of Fog computing and the defragmentation of the Transport layer in 4 sub-layers (security, storage, pre-processing and monitoring) as shown in Figure 3 allows us to have the following advantages:

- Work directly on local networks so it's faster (low latency).
- No need to consult the cloud and be connected to the internet.
- Real-time interactions (almost).
- Mobility.
- Distribution and decentralization.
- Localization.
- Security (Reduce the risk of attacks).
- System Smart and efficient (effective).
- Only necessary information is sent to the Cloud for analysis.

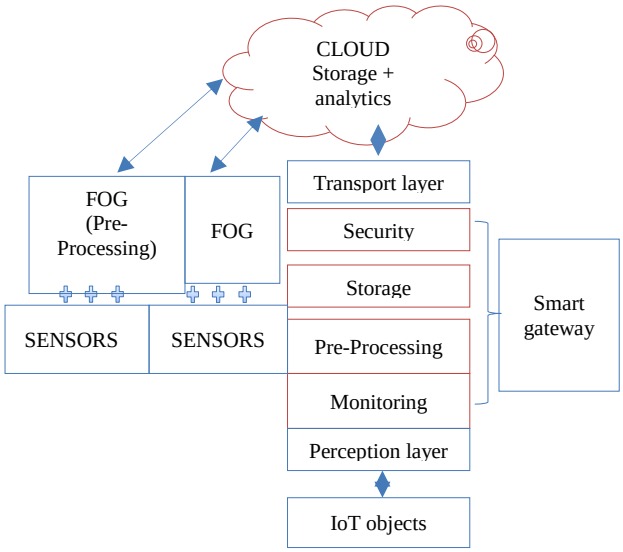

Figure 3. Cloud-Fog-IoT architecture.

2.2. QoS Model

Each object in the IoT can provide a number of specific services. Some IoT services may be similar or the same. However, QoS performance may be different from each other. Therefore, assessing the QoS helps to differentiate one service from another. Thus, it helps the service requesters in recognizing the best IoT service for its application.

To address IoT services separately, the exploration of this three IoT's components; the thing, the communication and the compute is needed, with their respective quality-of-service metrics [32]. The key attributes of QoS on the IoT can be dynamic or static [36]. Services in the IoT are linked to the physical world, so the geographic location information of devices affects user satisfaction, without forgetting that this dynamic environment, which affects the availability of these services [5]. Providing an acceptable level of QoS is an important issue in Fog-IoT [14] and for service selection of Fog-IoT-based healthcare system, we have taken twelve QoS attributes that have the most impact on this study case, and these attributes can be modelled as follows:

1. Availability (A): it is a percentage of time, and indicates when the service is available.
2. Cost (C): The cost that the user needs to pay for acquisition of the service.
3. Documentation (D): Measure of documentation (i.e., description tags)
4. Location/Range (L): The distance between the service and the destination specified by user.
5. Precision (P): The ability of the sensors to measure the deviation obtained in the output when the same signal or data is measured repeatedly under similar condition [8].
6. Reliability (R): The rate of that a service request is completed successfully.
7. Response time (Rt): The average time between when submitting a request from the user and accepting the service response.
8. Reputation (Rp): The average score of multi-evaluation of the service by the use of any scoring system.
9. Security (S): Represents the security level ensured by a service (authentication, encryption, etc.). Security allows protecting users against illegal forbidden access [15].
10. Service Classification (SC): The service classification represents various levels of service offering qualities. There are four service classifications: Platinum (High quality), Gold, Silver and Bronze (Low quality)
11. Success Rate (SR): Amount of response / number of request messages (%)
12. Throughput (T): Total Number of invocations for a given period of time (invokes/second).

The difference of QoS performance between each IoT information service is the key to service selection and composition, so it's essential to make a quantitative evaluation to the QoS performance of each service [18]. We assume that a composite service consists of (n) abstract services, denoted as: CS= {S1, S2,..., Sn}. For each abstract service Si, it owns several candidate concrete services, denoted as: Si= {Si1, Si2,..., Sim} (m represents the number of candidate services of Si).
There are four structural models in service composition Figure 4, i.e.,:

1. Sequence.
2. Parallel.
3. Loop.

4. Selection [42].

The aggregation functions [22] of composite service for the thirteen QoS attributes are formulated in Table 1 respectively.

Table 1. Aggregation function of QoS properties based on the W3C working group pattern.

	QoS Attribute	Sequential	Loop	Parallel	Selection
f_1	**Availability (A)**	$A_{cs} = \prod_{i=1}^{n} \Delta_{S_{ij} \in S_i} A_{ij}$	$A_{cs} = \prod_{i=1}^{n} \Delta_{S_{ij} \in S_i} A_{ij}$	$A_{cs} = \prod_{i=1}^{n} \Delta_{S_{ij} \in S_i} A_{ij}$	$A_{cs} = \prod_{i=1}^{n} \Delta_{S_{ij} \in S_i} p_{ij} * A_{ij}$
f_2	**Cost (C)**	$C_{cs} = \sum_{i=1}^{n} \Delta_{S_{ij} \in S_i} c_{ij}$	$C_{cs} = \sum_{i=1}^{n} \Delta_{S_{ij} \in S_i} c_{ij}$	$C_{cs} = \sum_{i=1}^{n} \Delta_{S_{ij} \in S_i} c_{ij}$	$C_{cs} = \sum_{i=1}^{n} \Delta_{S_{ij} \in S_i} p_{ij} * c_{ij}$
f_3	**Documentation (D)**	$D_{cs} = \prod_{i=1}^{n} \Delta_{S_{ij} \in S_i} D_{ij}$	$D_{cs} = \prod_{i=1}^{n} \Delta_{S_{ij} \in S_i} D_{ij}$	$D_{cs} = \prod_{i=1}^{n} \Delta_{S_{ij} \in S_i} D_{ij}$	$D_{cs} = \prod_{i=1}^{n} \Delta_{S_{ij} \in S_i} p_{ij} * D_{ij}$
f_4	**Location (L)**	$L_{cs} = \sum_{i=1}^{n} \Delta_{S_{ij} \in S_i} L_{ij}$	$L_{cs} = \sum_{i=1}^{n} \Delta_{S_{ij} \in S_i} L_{ij}$	$L_{cs} = \sum_{i=1}^{n} \Delta_{S_{ij} \in S_i} L_{ij}$	$L_{cs} = \sum_{i=1}^{n} \Delta_{S_{ij} \in S_i} p_{ij} * L_{ij}$
f_5	**Precision (P)**	$P_{cs} = \prod_{i=1}^{n} \Delta_{S_{ij} \in S_i} P_{ij}$	$P_{cs} = \prod_{i=1}^{n} \Delta_{S_{ij} \in S_i} P_{ij}$	$P_{cs} = \prod_{i=1}^{n} \Delta_{S_{ij} \in S_i} P_{ij}$	$P_{cs} = \prod_{i=1}^{n} \Delta_{S_{ij} \in S_i} p_{ij} * P_{ij}$
f_6	**Reliability (R)**	$r_{cs} = \prod_{i=1}^{n} \Delta_{S_{ij} \in S_i} r_{ij}$	$r_{cs} = \prod_{i=1}^{n} \Delta_{S_{ij} \in S_i} r_{ij}$	$r_{cs} = \prod_{i=1}^{n} \Delta_{S_{ij} \in S_i} r_{ij}$	$r_{cs} = \prod_{i=1}^{n} \Delta_{S_{ij} \in S_i} p_{ij} * r_{ij}$
f_7	**Response time (Rt)**	$Rt_{cs} = \sum_{i=1}^{n} \Delta_{S_{ij} \in S_i} t_{ij}$	$Rt_{cs} = \sum_{i=1}^{n} \Delta_{S_{ij} \in S_i} t_{ij}$	$Rt_{cs} = \max(t_{ij})$	$Rt_{cs} = \sum_{i=1}^{n} \Delta_{S_{ij} \in S_i} p_{ij} * t_{ij}$
f_8	**Reputation (Rp)**	$Rp_{cs} = \prod_{i=1}^{n} \Delta_{S_{ij} \in S_i} Rp_{ij}$	$Rp_{cs} = \prod_{i=1}^{n} \Delta_{S_{ij} \in S_i} Rp_{ij}$	$Rp_{cs} = \prod_{i=1}^{n} \Delta_{S_{ij} \in S_i} Rp_{ij}$	$Rp_{cs} = \prod_{i=1}^{n} \Delta_{S_{ij} \in S_i} p_{ij} * Rp_{ij}$
f_9	**Security (S)**	$S_{cs} = \min(S_{ij})$	$S_{cs} = \min(S_{ij})$	$S_{cs} = \min(S_{ij})$	$S_{cs} = \min(S_{ij})$
f_{10}	**Service Classification (Sc)**	$Sc_{cs} = \prod_{i=1}^{n} \Delta_{S_{ij} \in S_i} Sc_{ij}$	$Sc_{cs} = \prod_{i=1}^{n} \Delta_{S_{ij} \in S_i} Sc_{ij}$	$Sc_{cs} = \prod_{i=1}^{n} \Delta_{S_{ij} \in S_i} Sc_{ij}$	$Sc_{cs} = \prod_{i=1}^{n} \Delta_{S_{ij} \in S_i} p_{ij} * Sc_{ij}$
f_{11}	**Success rate (Sr)**	$Sr_{cs} = \prod_{i=1}^{n} \Delta_{S_{ij} \in S_i} Sr_{ij}$	$Sr_{cs} = \prod_{i=1}^{n} \Delta_{S_{ij} \in S_i} Sr_{ij}$	$Sr_{cs} = \prod_{i=1}^{n} \Delta_{S_{ij} \in S_i} Sr_{ij}$	$Sr_{cs} = \prod_{i=1}^{n} \Delta_{S_{ij} \in S_i} p_{ij} * Sr_{ij}$
f_{12}	**Throughput (T)**	$T_{cs} = \prod_{i=1}^{n} \Delta_{S_{ij} \in S_i} P_{ij}$	$T_{cs} = \prod_{i=1}^{n} \Delta_{S_{ij} \in S_i} T_{ij}$	$T_{cs} = \prod_{i=1}^{n} \Delta_{S_{ij} \in S_i} T_{ij}$	$T_{cs} = \prod_{i=1}^{n} \Delta_{S_{ij} \in S_i} p_{ij} * T_{ij}$

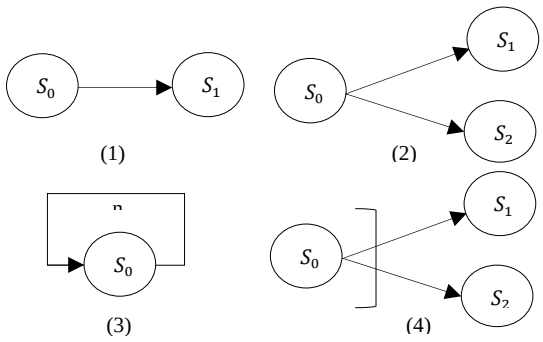

Figure 4. Structural models in service composition (sequence, parallel, loop and selection).

The problem is to find an optimal composition of the services, with low cost, high availability, documentation, precision, reliability, reputation, percentage, Service classification, success rate, throughput and security rate in less time considering the location.

Consequently, by giving equal weight to each QoS property described in Table 1; the service composition problem can be merged into a single objective function (1) as follows.

$$f = \text{maximise}(w_1 f_1 + w_3 f_3 + w_4 f_4 + w_5 f_5 + w_6 f_6 + w_8 f_8 + w_9 f_9 + w_{10} f_{10} + w_{11} f_{11} + w_{12} f_{12} - w_2 f_2 - w_7 f_7) \quad (1)$$

Herein, w_i is the weight of each QoS property, such that:

$$\sum_{i=1}^{12} w_i = 1$$

2.3. Genetic Algorithm

Considering factors such as space and time constraints, energy efficiency, and configurability of IoT services, the selection of IoT services for service composition is reduced to a multi-objective, multi-constrains optimization problem. The optimization is to made use of the resource in most efficient way. It means maximising or minimising some attribute through an

objective function [17], Heuristic algorithms such as Genetic Algorithm (GA), ACO, and Particle PSO are adopted to find the optimal composition of the IoT service.

Some global optimization algorithms are commonly used today, such as Enumeration algorithm, greedy algorithm, backtracking method, dynamic programming, and genetic algorithms [28] etc…, However, the temporal and spatial efficiency of the enumeration method is relatively low, a greedy algorithm cannot guarantee the acquisition of an optimal solution, the time cost of the backtracking method and of the dynamic programming increases exponentially depending on the magnitude of the problem. Compared to the above algorithms, GA can help achieve an optimal or near-optimal solution at a relatively low computational cost. As long as the fitness function is modelled mathematically, GA can be used to solve a large-scale optimization problem.

GA is the relatively wise choice to solve the problem of overall optimization of the composition of the service. The GA is a heuristic algorithm used to search for the optimal solution by simulating the natural process [35], in the traditional GA [13], the stop criterion of the algorithm is taken as being N= Nmax, N being the number of iterations. However, for some near real-time application scenarios in the context of the Internet of Things, there is a need to speed up the information service composition process. To handle this, the preservation of elitism is used to improve the efficiency of the algorithm. Otherwise, population similarity is used as additional termination conditions of the algorithm. It is quite possible to create individuals at random. And this method brings a very useful concept in genetic algorithms: diversity. The more the individuals of the initial population are different from each other, the more we will have the chance to find there, not the perfect solution, but what to manufacture the best possible solutions

2.4. MGA Framework

The MGA [34] divides individuals into multiple groups or sub-populations based on fitness values. Individuals from the same community have the opportunity to pair. If a person produces very well-adjusted fitness, the person migrates from their original group to the appropriate group with a higher fitness value, and vice versa. Thus, all individuals in the population enjoy equal opportunities, regardless of their physical condition or high physical condition. This allows MGA to maintain the diversity of the population. Also, MGA is easy to be paralyzed because the entire population is already divided into several sub-populations Figure 5.

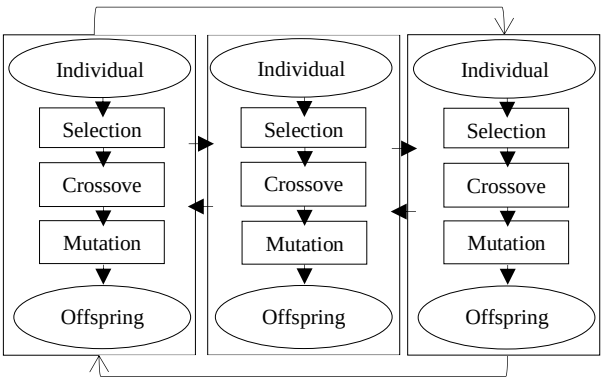

Figure 5. MGA's framework.

2.5. Fitness Function

In Darwin's model, individuals with the best characteristics have a better chance of surviving and reproducing. To determine this survivability, we will use a mathematical function called the fitness function or the objective function (2).

$$f = w_1 f_1 + w_3 f_3 + w_4 f_4 + w_5 f_5 + w_6 f_6 + w_8 f_8 + w_9 f_9 + w_{10} f_{10} + w_{11} f_{11} + w_{12} f_{12} - w_2 f_2 - w_7 f_7)$$

Herein, w_i is the weight, and f_i is the aggregation of each QoS property cited in Table 1, such that:

$$1 > w_i > 0 \; and \sum_{i=1}^{12} w_i = 1 \tag{2}$$

2.6. Chromosome Encoding and Genetic Operators

The encoding defines the representation of the variables of the problem to their manipulation by the evolutionary algorithm, thus each chromosome represents a possible service composition. We assuming that all services have been numbered with integers, the chromosome is represented by an array whose size is equal to the number of tasks, and the value of each position in the vector indicates the order of candidate service for the task. For example, in our study case; the process of solution encoding is shown in Figure 6. Here (3, 5, 1, 2, 4, 7, 6) indicates there are 7 tasks in the composite service (based on our composite service for the hospital emergency Figure 7).

3	5	1	2	4	7	6

Figure 6. Chromosome encoding.

The selection operator is the most important operators, its objective is to find the best individuals for the selection of the new population and the reproduction, i.e., selecting the individuals for crossover and mutation. We used the roulette wheel which consists of the distribution of the selection conversation proportionally to the fitness value.

$$\varphi(x_i) = \frac{\boldsymbol{f(x_i)}}{\sum_{i=1}^{n} \boldsymbol{f(x_i)}} \tag{3}$$

such as *f* is fitness value, and n is population size for each individual, we calculate $\varphi(x_i)$, $Som = \sum_{i=1}^{n} \varphi_i$

r is a real: 0 <r <Som; S=0; while (S < r) S=S+ $\varphi(x_i)$; x=x+1; end (4)

The individual who has been chosen several times, has the chance to participate in the following population or to the one-point crossover, it has the same principle as the casino Biased Roulette Wheel, this one-point crossover with the respect of these two rules (5) will guarantee us the selection of the right service.

Offspring_1=begin_father+end_mother
Offspring_2=begin_mother+end_father (5)

Random mutation has been applied to maintain diversity of the individuals, by changing the values of the randomly selected, the selected service must be in the same service class as the one changed. Also, amonitoring method is used to monitor each point (gene) of each chromosome, each point (gene) represents a given service, if a service become unavailable or when failures / exceptions occur, this service is instantly replaced with another one of his service class. This will allow us to recover from the unexpected situation so that the application continues its execution without no interruption. To save time of execution, the idea is to exploit the loops like the fitness function one for example.

Algorithm 1: our multi-population genetic algorithm (P-MPGA)

Input	*Available IoT services*
Output	*An approximately optimal chromosome*
//GP	*Global population*
//P	*The population of current generation*
//N	*The number of individuals (global)*
//m	*The number of sub-populations*
//n	*The number of individuals of each sub-population*
//t	*The number of current generations*
//T	*The maximum number of generations*
//p_m, p_c	*The probability of mutation and crossover*
Step 1	*Divide (GP) to sub-population (m) according to (N)*
Step 2	*initialise each population randomly*
Step 3	*t=0*
Step 4	*for i = 1 to m in parallel*
Step 5	*while (t<=T) and not 'stopping criterion' do*
Step 6	*Fitness (P_i) using (2)*
Step 7	*Sort (P_i)*
Step 8	*Improved_Select(P_i)*
Step 9	*Improved_Crossover(P_i) according to p_c*
Step 10	*Improved_Mutation(P_i) according to p_m*
Step 11	*t = t+1*
Step 12	*end while*
Step 13	*end for*

Algorithm 2: our Monitoring algorithm

Input	*Individual (x_i)*
Output	*A safe Individual (x_i)*
//p	*A point of (p_j) or the gene*
//Sc	*Service class*
Step 1	*for each (p_j)*
Step 2	*If (p_j) is not safe*
Step 2	*Replace (p_j) with another service picked up from same (Sc_i)*
Step 3	*End*
Step 4	*End*

Algorithm 3: our improved selection algorithm (improved_select)

Input	*Population (P_i)*
Output	*Population (P_i) with all roulette wheel value calculated for each individual*
//P	*The population of current generation*
// x	*Individual of (P_i)*
Step 1	*for each individual*
Step 2	*calculate $\varphi(x_i)$ using (3)*
Step 2	*Monitoring (x_i)*
Step 3	*End*
Step 4	*calculate Som using (4)*
Step 5	*r =random (0, Som)*
Step 6	*S=0*
Step 7	*while (S < r)*
Step 8	*S=S+ $\varphi(x_i)$*
Step 9	*x=x+1*
Step 10	*end*
Step 11	*Migrate best individual ()*

Algorithm 4: our improved crossover algorithm (improved_crossover)

Input	*Population (P_i)*
Output	*2 new individuals*
//P	*The population of current generation*
// x	*Individual of (P_i)*
//po	*The one point*
// x_father	*Individual father for the crossover*
// x_mother	*Individual mother for the crossover*
//b_father	*The begin of the father according to po*
//e_father	*The end of the father according to po*
//b_mother	*The begin of the mother according to po*
//e_mother	*The end of the mother according to po*
//os_1	*Offspring 1*
//os_2	*Offspring 2*
Step 1	*Choose the two-best induvial for x_father&x_mather respectively*
Step 2	*Choose a random one point*
Step 3	*os_1= b_father + e_mother*
Step 4	*os_2=b_mother + e_father*

Algorithm 5: our improved mutation algorithm (improved_mutation)

Input	*Individual (x_i)*
Output	*Individual (x_i) with a mutation for the new generation*
//p	*The random point (gene) for the mutation*
//Sc	*Service class of (x_i)*
// x_service	*New service from (Sc_i)*
Step 1	*po = random (1,7)*
Step 2	*x_service= service picked up from (Sc_i)*
Step 3	*Replace p with x_service*

3. Experimental Setup and Evaluation

In this section, we evaluate our propped Multi- (P-MPGA)(Algorithm 1) by comparing it to the traditional (GA) [35] and the (MGA) proposed in [22]. We run P-

MGA, GA and MGA 10 times, and use the averaged values for evaluation.

3.1. Experience Setup

Algorithms were implemented with Java using PC soft Windev23 IDE, and ran on an Intel i7-6820HQ 2.71Ghz, SDD, 16 GB RAM, laptop PC with Window 10.

To illustrate the idea of our service composition, an example of the ambulance emergency is used. We suppose that our system is equipped with different devices, including sensors and camera, all the experiments are based on the process presented in Figure 7, which represent an example of composite service for our study case. It contains about seven services and each can be a composite service itself, like the path pickup or the automatic control of traffic lights on the ambulance route, which consists to let the ambulance pass by giving the green light. Table 2 lists the meaning of each one of them.

Our experiments are based on a publicly available Web service dataset QWS Dataset (2.0) [2], we extend the size of QWS dataset randomly. The number of services is finally extended to 10 000 and 12 QoS attributes are selected for our experiments, i.e., Availability (A), Cost (C), Documentation (D), Location (L), Memory Resources (M), Precision (P), Reliability (R), Response time (Rt), Reputation (Rp), Security (S), Service Classification (SC), Success rate (Sr), Throughput (T), they are bound to the 2^{nd}, 8^{th}, 9^{th},6^{th},5^{th}, 12^{th},1^{st},7^{th}, 10^{th}, 11^{th}, 4^{th}, 3^{th}, fields of QWS Dataset respectively.

Table 2. Service classes (Sc).

Sc Id	Sc Name and Description
Sc1	Pick up one hospital
Sc2	Pick up one ambulance
Sc3	Send patient information to the hospital
Sc4	Pick up ambulance's best path to hospital
Sc5	Traffic lights: Facilitate road traffic by using traffic lights
Sc6	Notify public cars: Facilitate road traffic by notify public cars
Sc7	Notify the authorities: Facilitate road traffic by notify the authorities like the police
Initialization and the end of the process	
Init	Emergency reported
End	The patient is arrived to the chosen hospital

A Service Class (SC) for IoT service is a tuple (*nm*, *dsc*, *op*)

Where:

- *nm* is the name of the service.
- *dsc is the text description of the service.*
- *op is an operation of the service.* Here, *nm*, *dsc* and *op* are the same as those specified for services.

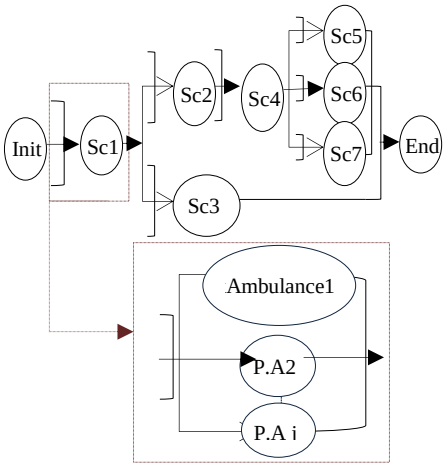

Figure 7. A composite service for the hospital emergency.

Here in Figure 7, each place Sci like in Sc1, is a service selection between services "Pick up one ambulance" in class selection structural model (Sc1 Figure5).

Table 3. P-MPGA parameter.

Number of available services	10 000
Population size	5 000
Number of generations	2 000
Selection method	roulette wheel
Crossover type	one-point
Crossover probability	90%
Mutation type	Random pick from a specific Sc
Mutation probability	10%
Number of iterations	40

3.2. Experience Results

In this section, we present the results obtained by our P-MPGA compared with GA and MGA, we used same genetic operator, parameters (shows in Table 3) and fitness function for all algorithms, we also used four sub-population for P-MPGA. Figure 8 shows that with P-MPGA the execution time is significantly reduced compared with GA & MGA. Figure 9 shows that P-MPGA perform better than GA&MGA, P-MPGA converges better and to maximal fitness value. Table 4 represents the optimal fitness value obtained after 40 iteration with the run time, despite the increase in number of services (5000,10000,15000 & 20000), P-MPGA shows a better ratio (execution time/optimal fitness value). In Figure 10, despite the increase in population, P-MPGA still obtains a good optimal fitness value, what makes our algorithm scalable.

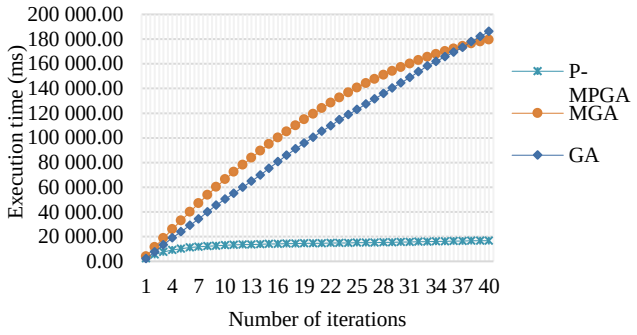

Figure 8. Comparison between P-MPGA, MGA & GA (execution time).

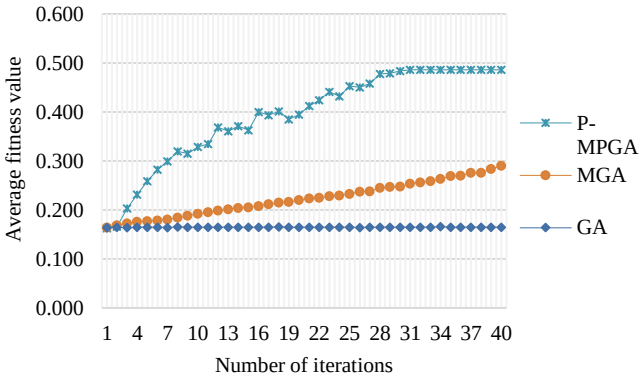

Figure 9. Comparison of fitness values obtained by P-MPGA, MGA and GA.

Table 4. Ratio (Execution time/Best fitness value) after 40 iterations.

Method	Best fitness value	Execution Time (ms)	Ratio
With 5 000 Services			
MGA	0,5555995702744	179590,0000	323236,3911
GA	0,5517482757568	172156,0000	312019,0992
P-MPGA	0,4922352731228	16775,7500	34080,7555
With 10 000 Services			
MGA	0,5178905129433	191 823,0000	370392,9599
GA	0,5178905129433	173699,0000	335397,1460
P-MPGA	0,5178905129433	16 752,2500	32347,0880
With 15 000 Services			
MGA	0,4989299476147	219 689,0000	440320,3316
GA	0,4989299476147	195192,0000	391221,2545
P-MPGA	0,4989299476147	18 655,7500	37391,5218
With 20 000 Services			
MGA	0,6055026650429	234 021,0000	386490,4541
GA	0,5641102790833	201543,0000	357275,8864
P-MPGA	0,5641102790833	18 946,2500	33586,0747

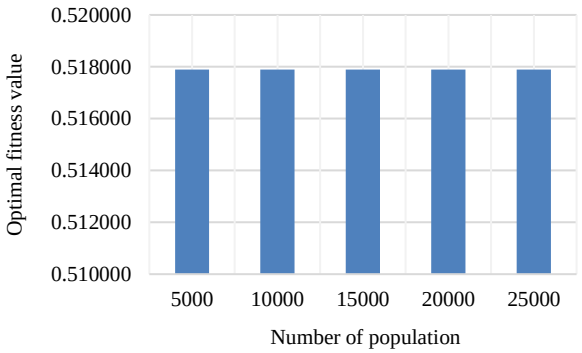

Figure 10. Optimal fitness values obtained by P-MPGA according to the number of populations.

4. Conclusions and Perspectives

In this paper, we presented an adaptive QoS-aware service composition approach based on multi-population genetic algorithm in Fog-IoT healthcare environment. IoT-cloud architecture problems led us to use the 5-layared architecture implemented on a Fog-IoT computing system especially the Processing layer. Our work was focus on this Processing layer where we divided it into four sub-layers (security, storage, pre-processing and monitoring), it allows us to have promising advantages. Secondly, we implemented a QoS-aware multi-population genetic algorithm (P-MPGA), and we considered12 QoS attributes, i.e., (A), (C), (D), (L), (M), (P), (R), (Rt), (Rp), S (S), (Sc), (Sr), Throughput (T). P-MPGA implements a smart selection method which allows us to always select the right service. Also, a function is used for monitoring services to manage dynamic change of IoT environments.

Experimental results show the excellent results of P-MPGA in terms of execution time, average fitness values and execution time / best fitness value ratio despite the increase in population. P-MPGA can quickly achieve a composite service satisfying user's QoS needs, which makes it suitable for a large scale IoT environment. As a future work, we will try to focus more on the monitoring system, considering the energy consumption of the framework. And evaluate of our model in real-world service system (ambulance emergency), which is an important study case towards human life.

References

[1] Abou-Tair D., Büchsenstein S., and Khalifeh A., "A Fog Computing-Based Framework for Privacy Preserving Iot Environments,"*in Proceedings of the International Arab Journal of Information Technology*, vol. 17, no. 3,pp. 306-314, 2020.

[2] Al-Masri E. and Mahmoud Q., "Investigating Web Services on The World Wide Web," *in Proceeding of the 17th International Conference on World Wide Web*,Beijing, pp. 795-804, 2008.

[3] Aoudia I., Benharzallah S., Kahloul L., and Kazar O., "A Comparative Analysis of Iot Service Composition Approaches," *The International Arab Conference on Information Technology Yassmine Hammamet*, Yassmine Hammamet, pp. 1-7, 2017.

[4] Aoudia I., Benharzallah S., Kahloul L., and Kazar O., "QoS-Aware Service Composition in Fog-Iot Computing Using Multi-Population Genetic Algorithm," *The International Arab Conference on Information Technology*, 6th of October City, pp. 1-9, 2020.

[5] Aoudia I., Benharzallah S., Kahloul L., and Kazar O., "Service Composition Approaches for Internet of Things : A Review," *International Journal of Communication Networks and Distributed Systemsm*, vol. 23, no. 2, pp.194-230,2019.

[6] Barakat L., Miles S., and Luck M., "Adaptive

Composition in Dynamic Service Environments," *Future Generation Computer Systems*, vol. 80, pp. 215-228, 2018.

[7] Burhan M., Rehman R., Khan B., and Kim B., "IoT Elements, Layered Architectures and Security Issues: A Comprehensive Survey," *Sensors*,vol. 18, no. pp. 2-37,2018.

[8] Chalmers D. and Sloman M., "A Survey of Quality of Service in Mobile Computing Environments," *IEEE Communications Surveys and Tutorials*, vol. 2, no. 2, pp. 2-10, 1999.

[9] Dar K., Taherkordi A., Baraki H., Eliassen F., and Geihs K., "A Resource Oriented Integration Architecture for The Internet of Things: A Business Process Perspective," *Pervasive and Mobile Computing*, vol. 20, pp. 145-159, 2015.

[10] Dar K., Taherkordi A., Rouvoy R., and Eliassen F., "Adaptable Service Composition for Very-Large-Scale Internet of Things Systems," *in Proceedings of the 8th Middleware Doctoral Symposium*,New York, pp. 1-2, 2011.

[11] Darwish D., "Improved Layered Architecture for Internet of Things," *International Journal of Computing Academic Research*, vol. 4, no. 4, pp. 214-223, 2015.

[12] Fki E., Tazi S., and Drira K., "Automated and Flexible Composition Based on Abstract Services for A Better Adaptation to User Intentions," *Future Generation Computer Systems*, vol. 68, pp. 376-390, 2017.

[13] Goldberg D., *Genetic Algorithms in Search, Optimization, and Machine Learning*, Choice Reviews Online, 1989.

[14] Haghi Kashani M., Rahmani A. and Jafari Navimipour, "Quality of Service-Aware Approaches in Fog Computing," *International Journal of Communication Systems*, vol. 33, no. 8, pp. 1-34, 2020.

[15] Houhamdi Z. and Athamena B., "Identity Identification and Management in The Internet of Things," *The International Arab Journal of Information Technology*, vol. 17, no. 4A, pp. 645-654,2020.

[16] Huo Y., Qiu P., Zhai J., Fan D., and Peng H., "Multi-Objective Service Composition Model Based on Cost-Effective Optimization," *Applied Intelligence*, vol. 48, no. 3, pp. 651-669, 2018.

[17] Kashyap N., Kumari A., and Chhikara R., "Service Composition in IoT Using Genetic Algorithm and Particle Swarm Optimization," *Open Computer Science*, vol. 10, no. 1, pp. 56-64, 2020.

[18] Kashyap N. and Kumari C., "Hyper-Heuristic Approach for Service Composition in Internet of Things," *Electronic Government*, vol. 14, no. 4, pp. 321-339, 2018.

[19] Kouicem A., Chibani A., Tari A., Amirat Y., and Tari Z., "Dynamic Services Selection Approach for The Composition of Complex Services in The Web of Objects," *IEEE World Forum on Internet of Things*, pp. 298-303, 2014.

[20] Li B., Yang R., and Hu Y., "An Experimental Study for Intelligent Logistics: A Middleware Approach," *Chinese Journal of Electronics*, vol. 25, no. 3, pp. 561-569, 2016.

[21] Li L., JinZ., Li G., Zheng L., and Wei Q., "Modeling And Analyzing The Reliability and Cost of Service Composition in The Iot: A Probabilistic Approach," *in Proceedings IEEE 19th International Conference on Web Services*, Honolulu, pp. 584-591, 2012.

[22] Li Q., Dou R., Chen F., and Nan G.,"A Qos-Oriented Web Service Composition Approach Based on Multi-Population Genetic Algorithm for Internet of Things," *International Journal of Computational Intelligence Systems*, vol. 7, no. 2, pp. 26-34,2014.

[23] Madakam S., Ramaswamy R., and Tripathi S., "Internet of Things (IoT): A Literature Review," *Journal of Computer and Communications*, vol. 3, no. 5, pp. 164-173, 2015.

[24] Mashal I., Alsaryrah O., Chung T., Yang C., Kuo W., and Agrawal D., "Choices for Interaction With Things on Internet and Underlying Issues," *Ad Hoc Networks*, vol. 28, pp. 68-90, 2015.

[25] NEXOF-RA. Deliverable D10.1: Requirements Report. IST-FP7-216446, 2009.

[26] Oteafy S. and Hassanein H., "Iot in The Fog: A Roadmap for Data-Centric Iot Development," *IEEE Communications Magazine*, vol. 56, no. 3, pp. 157-163, 2018.

[27] Puliafito C., Mingozzi E., LongoF., Puliafito A., and Rana O., "Fog Computing for The Internet of Things: A Survey," *ACM Transactions on Internet Technology*, vol. 19, no. 2, pp. 1-41, 2019.

[28] Qiufen W. and Liang D., "A Heuristic Genetic Algorithm for Solving 0-1 Knapsack Problem," *Computer Applications and Software*,vol. 30, no. 2, pp. 33-37, 2013.

[29] Said O. and Masud M., "Towards Internet of Things: Survey and Future Vision," *International Journal of Computer Networks*, vol. 5, no. 1, pp. 1-17, 2013.

[30] Sarha A., *Fog Computing as Solution for IoT-Based Agricultural Applications*, IGI Global, 2021.

[31] Sethi P. and Sarangi S., "Internet of Things: Architectures, Protocols, and Applications," *Journal of Electrical and Computer Engineering*, vol. 2017, pp. 1-25, 2017.

[32] Singh M. and Baranwal G., "Quality of Service (Qos) in Internet Of Things," *in Proceedings 3rd International Conference on Internet of Things: Smart Innovation and Usages*, Bhimtal, pp. 1-6,2018.

[33] Singh M., Baranwal G., and Tripathi A., "QoS-

Aware Selection of IoT-Based Service," *Arabian Journal for Science and Engineering*, vol. 45, pp. 1-18, 2020.

[34] Sathya S. and Radhika M., "Convergence of Nomadic Genetic Algorithm on Benchmark Mathematical Functions," *Applied Soft Computing Journal*, vol. 13, no. 5, pp. 2759-2766, 2013.

[35] Sun M., Shi Z., Chen S., Zhou Z., and Duan Y., "Energy-Efficient Composition of Configurable Internet of Things Services," *IEEE Access*, vol. 5, pp. 25609-25622, 2017.

[36] Temglit N., Chibani A., Djouani K., and Nacer M., "Distributed Approach for QoS Service Selection in Web of Objects," *Procedia Computer Science*, vol. 83, pp. 1170-1175, 2016.

[37] Yachir A., Amirat Y., Chibani A., and Badache N., "Event-Aware Framework for Dynamic Services Discovery and Selection in the Context of Ambient Intelligence and Internet of Things," *IEEE Transactions on Automation Science and Engineering*, vol. 13, no. 1, pp. 85-102.

[38] Yang R., Li B., and Cheng C., "A Petri Net-Based Approach to Service Composition and Monitoring in The IOT," *in of Proceedings Asia-Pacific Services Computing Conference*, Fuzhou, pp. 16-22, 2014.

[39] Yang R., Li B., and Cheng C., "Adaptable Service Composition for Intelligent Logistics: A Middleware Approach," *in Proceedings of the International Conference on Cloud Computing and Big Data*, Wuhan, pp.75-82, 2014.

[40] Yun M. and Yuxin B., "Research on The Architecture and Key Technology of Internet of Things (Iot) Applied on Smart Grid," *in Proceedings of the International Conference on Advances in Energy Engineering*, Beijing, pp. 69-72, 2010.

[41] Zhang X., Geng J., Ma J., Liu H., Niu S., and Mao W., "A Hybrid Service Selection Optimization Algorithm in Internet of Things," *Eurasip Journal on Wireless Communications and Networking*, vol. 8, pp. 8593 -85949, 2020.

[42] Zhao X., Song B., Huang P., Wen Z., Weng J., and Fan Y., "An Improved Discrete Immune Optimization Algorithm Based on PSO for Qos-Driven Web Service Composition," *Applied Soft Computing Journal*, vol. 12, no. 8, pp. 2208-2216, 2012.

Idir Aoudia is a Ph. D student at LINFI Laboratory Biskra University 07000, Obtains his master 2 SIM degree in 2015 from Batna University (Algeria). His current research interests, services composition and Internet of things.

Saber Benharzallahis a professor and researcher in the computer science department of Batna 2 University (Algeria). Received his Ph. D degree in 2010 from the Biskra University (Algeria). Prof. Benharzallah is currently director of laboratory LAMIE (Batna 2 University). His research interests include Internet of things, service-oriented architecture, and context aware systems.

Laid Kahloul received the Ph. D degree in computer software and theory from the Computer Science Department, Biskra University, Biskra, Algeria, in 2012. He is currently a Professor with the Computer Science Department, Biskra University, Algeria. His current research interests include Petri nets, High Level Petri Nets, and Software Engineering.

Okba Kazar is a Professor in the Computer Science Department of Biskra he helped to create the laboratory LINFI at the University of Biskra. He is the author of numerous publications, member of international conference program committees and the "editorial board" for various magazines. His research interests are artificial intelligence, multi-agent systems, web applications and information systems.