Array 8 (2020) 100041

Contents lists available at ScienceDirect

Array

journal homepage: www.elsevier.com/journals/array/2590-0056/open-access-journal

Quality of service-aware service selection algorithms for the internet of things environment: A review paper

Aghabi N. Abosaif [a,*], Haitham S. Hamza [b]

[a] Department of Computer Science, Sudan University of Science and Technology, Sudan
[b] Department of Information Technology, Cairo University, Egypt

ARTICLE INFO

Keywords:
Service Selection Algorithm (SSA)
Quality of Service (QoS)
Internet of Things (IoT)
Optimization Objective Algorithm]

ABSTRACT

The Internet of Things (IoT) has evolved over the last decade to connect a massive number of objects. These connected objects provide a vast amount of services to enhance the daily live of end users. The services provided comprise other services with similar functionalities but different quality of service (QoS) requirements. Thus, the problem of selecting and combining services that match the required QoS constraints is challenging. Therefore, in this review, we classify, and analyze state-of-the-art algorithms for service selection under QoS constraints in the IoT environment. We propose a classification system to review and analyze the various state-of-the-art algorithms described between 2012 and 2020. The aims of this review are to provide the research community with guidance and knowledge, and to identify the trends regarding the different algorithms used to solve the service selection problem. In addition, we discuss future research directions in terms of the design, implementation, and evaluation of new service selection algorithms.

1. Introduction

In the last decade, interest has increased in both academia and industry in the Internet of Things (IoT) concept, with high economic impacts [1]. It is predicted that devices and objects will be increasingly connected in the coming years [2], thereby leading to more services being exposed to end users.

Fundamental features of IoT systems include the ability to select and combine services that match the needs and preferences of end users. Service selection is regarded as the core of service composition (SC) [3] and it is defined as selecting the most appropriate service from candidate independent services that match the end user requirements [4]. These candidate services provide the same functionality but they may have different non-functional properties. The non-functional properties are quality of service (QoS) factors with different values. Thus, it is essential to consider QoS factors that satisfy the end user requirements.

The service selection process is even more challenging when combined with specific QoS requirements. In this case, the problem does not simply involve identifying, selecting, and combining a service with specific functionalities, but instead it is necessary to select the best service that matches both the functionality and the quality requirements of the end user. These conditions require using optimization algorithms to select the set of services that match specific criteria, such as the energy consumption [5–7], response time [8,9], reliability [5,8–10], availability [11–13], cost [14,15], throughput [16], and accuracy [10].

In recent years, researchers have focused on developing optimization algorithms for selecting services with a specific set of quality requirements in the IoT environment. Examples of these algorithms include cluster head selection [17], shortest path algorithms [16,18], mapping flow-based algorithms [19–21], co-locating services [14,15], physical services models [25,26], genetic algorithms (GAs) [6,11,22], and particle swarm optimization (PSO) algorithms [6,11,22].

To the best of our knowledge, few reviews have surveyed the SC problem in the IoT under QoS criteria, although one previous review [23] investigated the selection algorithms employed under QoS constraints in the IoT. Previous surveys related to this problem are summarized as follows (See Table 1).

Mqhele et al. [24] reviewed service selection methods for a range of dynamic environments, and provided a general overview of dynamic Web services, cloud, and the IoT. Their review considered different approaches, including the multi-agent approach, ontologies, QoS, functional requirements, and user-centered QoS. They focused on certain factors related to service selection methods, such as user preferences, domain specifications, storage, scalability, and evaluations. A case study

* Corresponding author.
E-mail address: aghape.nabeel@gmail.com (A.N. Abosaif).

https://doi.org/10.1016/j.array.2020.100041
Received 25 April 2020; Received in revised form 31 August 2020; Accepted 31 August 2020
Available online 3 September 2020

based on a smart campus with IoT services compared a content-based algorithm with a collaborative filtering algorithm. However, they did not discuss the methods or algorithms used to solve the service selection problem, such as optimization algorithms, and ensuring the QoS based on the IoT architecture was not addressed.

Bouzary et al. [25] presented a comprehensive survey of service selection and composition in cloud manufacturing. They classified six categories comprising objective functions, selection criteria, algorithm types, correlation awareness, mapping approaches, and dynamic composition. They did not consider the service selection problem in the IoT environment or ensuring QoS based on the IoT architecture. They only investigated some of the traditional QoS parameters comprising cost, response time, reliability, and availability. In addition, they did not discuss any of the evaluation metrics used to evaluate the performance of the proposed state-of-the-art approaches.

Yu et al. [26] reviewed the non-functional factor-based service selection problem in Web services. They proposed a model for non-functional properties, hierarchical properties, user preferences, evaluations of properties, dynamic aggregation, automation, scalability, and accuracy. Comparative classification was conducted based on three dimensions: policy vs. reputation, universal description, discovery, and integration (UDDI) extensions vs. semantic Web services, and graphic preference modeling vs. ontology-based preference modeling. They compared different types of service selection optimization algorithms and frequently used QoS factors, such as the response time, availability, and reliability, but not how to ensure the QoS in the IoT architecture. They reviewed various approaches based on the proposed requirements but details of the criteria employed were not provided, such as the model implementations, evaluation methods, or Results obtained.

Hamzei et al. [27] conducted a systematic survey of SC-aware methods based on QoS factors proposed in the IoT environment. They divided the methods into four main categories comprising framework-based, model-based, service-oriented architecture (SOA), representational state transfer (REST)ful-based, and heuristic-based approaches. They discussed the benefits and drawbacks of the state-of-the-art approaches and defined six QoS factors to examine for the SC approaches, i.e., scalability, execution time, cost, reliability, availability, and response time, but they did not consider the QoS factors based on the IoT architecture. In addition, they did not define the methods that can be used to design and evaluate service selection algorithms (SSAs).

Aoudia et al. [28] surveyed the techniques used for SC in the IoT. They compared the state-of-the-art approaches based on three essential requirements for the IoT system: management of heterogeneity, monitoring and fault tolerance, and reasoning for the environment and resources, as well as constraints comprising resource constraints, the response time, and real-time constraints. In particular, they compared traditional Web SC and IoT SC, but they only considered a few factors related to traditional QoS and did not investigate how to ensure the QoS based on the IoT architecture.

Asghari et al. [29] conducted a systematic review of SC approaches in the IoT based on seven main categories: SC approaches, platforms selected, tools employed, measurement environments used to evaluate SC, QoS factors, current algorithms supported by the SC, and future challenges of SC. In particular, they considered the functional behavior of SC and non-functional aspects of QoS in the IoT in SC approaches. They investigated most of the QoS factors in the IoT system but without referring to the IoT architecture. The benefits and drawbacks of the state-of-the-art approaches were discussed and some weaknesses were identified. A useful classification was provided for service selection in IoT but the methods that must be used to design SSAs were not discussed. In addition, they did not provide a complete review of the QoS factors, such as QoS in the network layer, and they only discussed one QoS factor in the sensor layer (energy). However, they discussed the languages and verification tools employed to measure the performance of the existing approaches. Detailed descriptions were not provided of the methods used to compare the performance of different methods, and the Results obtained from these comparisons were not presented.

Aoudia et al. [30] reviewed the existing SC approaches for IoT based on three groups of criteria. First, they considered the dynamic composition and adaptation of the proposed solutions, including the optimization method used, its description, the deployment areas, the techniques used, and whether they were applied online or offline. Second, they assessed the protection and security protocols, distribution and decentralization, automatic identification and resolution of failure and interaction problems, and independence and extensibility. Third, they considered the Results obtained performance of the service representations, standards and protocols used, models employed, and the optimization of the composition. They investigated the essential points of the selection problem but did not explain them in detail, such as declaring the dynamic composition without defining other composition types. Moreover, they presented the techniques employed but without discussing the differences between them. In addition, they did not discuss the different factors that affect traditional QoS or QoS in the IoT environment.

Dongre et al. [31] investigated the QoS parameters applied for service composition and selection by considering nine QoS parameters in the application layer: execution time, response time, availability, reliability, throughput, cost, price, reputation, and latency. However, they did not discuss technique or methods used to solve the selection problem.

Li et al. [23] conducted a systematic study of SSAs in the IoT environment, which they categorized as centralized, decentralized, and hybrid classes. They focused on the techniques used to solve the selection problem and the QoS parameters employed to assess the advantages and disadvantages of each algorithm. They identified the main issues and challenges that affect the service selection problem in the IoT. Moreover, they analyzed the distribution of state-of-the-art studies by year of publication and the percentage of studies by different publishers (IEEE, Springer, Hindawi, Elsevier, Sage, IJSRCSEIT, and other). The main methods used by service selection algorithms (SSAs) were discussed based on three categories but they did not classify the optimization methods employed or whether the behavior was based on the process time (static or dynamic). Moreover, they did not compare the types of software or data sets used to evaluate the performance of the state-of-the-art algorithms. Newly published studies were not discussed.

All of the previous reviews mentioned above are useful but our proposed classification differs because of the following three main points. We define the methods that can be used to design an SSA while considering the QoS based on the three main layers in the IoT environment (sensor, network, and application layers). Second, we provide detailed classifications of the optimization methods used in the state-of-the-art approaches for solving the service selection problem. Finally, we investigate the evaluation metrics and comparisons applied to measure the performance of the state-of-the-art approaches.

In this review, we surveyed, classified, and analyzed the state-of-the-art algorithms for service selection in the IoT under QoS constraints. We used a new classification method for reviewing and comparing various SSAs. Based on our review of state-of-the-art methods, two main problems must be addressed by the research community in order to propose a suitable solution for the services selection problem: identifying the methods used to design an appropriate IoT environment and determining the methodology used to implement the proposed solution. Moreover, a third important requirement is an evaluation step for assessing the efficiency and effectiveness of the proposed solution. In particular, we aimed to address the following questions.

Q1: What are the methods used to design SSAs in terms of the process time phase, behavioral workflow management, and optimization objectives?
Q2: What are the implementations proposed for SSAs in the QoS layers based on the IoT architecture and types of optimization algorithms?

Table 1
Comparison of state-of-the-art surveys introduced for service selection.

Authors	Year	Review type	Approach criteria
Mqhele et al. [24]	2017	Service selection in dynamic environments (dynamic Web services, cloud, and IoT)	Based on multi-agent approach, ontology, QoS, functional requirements
Bouzary et al. [25]	2018	Service selection and SC in cloud manufacturing	Based on the objective function, selection criteria, algorithms, correlation awareness, mapping approaches, and dynamic composition
Yu et al. [26]	2008	Service selection in Web services	Based on a model of the non-functional properties, hierarchical properties, user preferences, evaluation of properties, dynamic aggregation, automation, scalability, and accuracy
Hamzei et al. [27]	2018	SC in IoT	Using framework-based, model-based, SOA, RESTful-based, and heuristic-based approaches
Aoudia et al. [28]	2017	SC in IoT	Based on the management of heterogeneity, monitoring and fault tolerance, and reasoning for the environment and resources
Asghari et al. [29]	2018	SC in IoT	Based on SC approaches, platforms selected, tools used, measurement environments for evaluating SC, QoS factors, current algorithms supported by the SC, and future challenges of SC
Aoudia et al. [30]	2019	SC in IoT	Based on dynamic composition, adaptation used, automatic identification and resolution of failure, distributed and decentralized composition, protection and security protocol, optimization, performance, and Results obtained
Dongre et al. [31]	2020	QoS in service composition in IoT	Nine QoS parameters considered in the application layer: execution time, response time, availability, reliability, throughput, cost, price, reputation, and latency
Li et al. [23]	2020	Service selection in IoT	Centralized, decentralized, and hybrid classes

Q3: What are the techniques used to assess the performance of SSAs in terms of the specific evaluation approaches, the software employed, and data sets applied?

The answers to these questions are presented in the following, as well as the Results and suggestions for future research.

The main contributions of this review can be summarized as follows.

Classification of SSAs for use in the IoT system: we classified the proposed SSAs for the IoT system according to three main levels: algorithm design level, algorithm implementation level, and algorithm performance evaluation level. This classification covers most of the research requirements when proposing a new SSA.

Review and analysis of the current SSAs in IoT: we reviewed the state-of-the-art methods for SSAs for the IoT system. Furthermore, we analyzed the proposed solutions based on three main classification levels (algorithm design, algorithm implementation, and algorithm evaluation) to assess the details of the current selection solutions for IoT.

Results *and future research directions:* we identified future trends to help the research community identify areas that need further investigation in order to address the service selection problem, as well as providing knowledge and understanding to clarify the requirements for SSAs. These details should facilitate the construction of an efficient SSA that addresses the challenges of designing, implementing, and evaluating SSA in an appropriate manner for the IoT system.

The remainder of this paper is organized as follows. In Section 2, we discuss the background and present necessary definitions. The proposed state-of-the-art classification method is presented in Section 3. In Section 4, we discuss solutions for designing SSA and analyze the design solutions. In Section 5, we explain the solutions for implementing SSA and analyze the proposed solutions. In Section 6, we consider the performance evaluations proposed for SSA solutions. The Results and future research directions are presented in Section 7. We give our conclusions in Section 8. In Section 9, we outline research limitations and provide recommendations.

2. Background

In the following, we explain the context and concepts required to understand the proposed classification system. First, we explain the services, composition, and selection process concepts. We then describe the main components of our proposed classification method.

2.1. SC and selection processes

2.1.1. Web services

Web services are as an essential concept of the IoT environment because they allow sensors and devices to exchange data and information through the IoT network. The definitions of Web services vary from highly public to highly defined. Most consumers view a Web service as an application or end service that can communicate with other applications or end services through the Web. The World Wide Web Consortium (W3C) [32] introduced the concept of a web service as: “A software system designed to support interoperable machine-to-machine interaction over a network. It has an interface described specific format like Web Services Description Language (WSDL)." In general, Web services are based on three critical standard concepts comprising WSDL, the Simple Object Access Protocol (SOAP), and UDDI. We can view Web services as comprising connected components and integrated into a more complex distributed environment. The three main components of the Web services architecture are as follows.

- *Service Provider:* Different services are provided via interfaces to create, implement, and publish a Web service using the UDDI specification.
- *Service Consumer:* A service consumer is regarded as the end user of a Web service. A service consumer uses the service registry to obtain information about services and to access them.
- *Service Registry:* The service registry contains information about different services provided based on the UDDI specification, where services are listed and advertised to search.

Web services can divide into two main types, as follows.

Atomic Service: An atomic service is also known as an elementary service [33] and it is a sold service that cannot be divided into other Web services or that relies on other Web services to satisfy consumer requests. The interface of an atomic service is based on SOAP and WSDL.

Composite Service: [33] this services integrates or collects atomic services, where each has a functionality to implement a specific application or service. Atomic or composite Web services defined by an identifier comprise a set of service factors that deliver useful information for the services and a set of operations to identify a service’s functionality.

Thus, the SC process integrates and collects more than one service into a single service to perform more complex functions. In our reviewed,

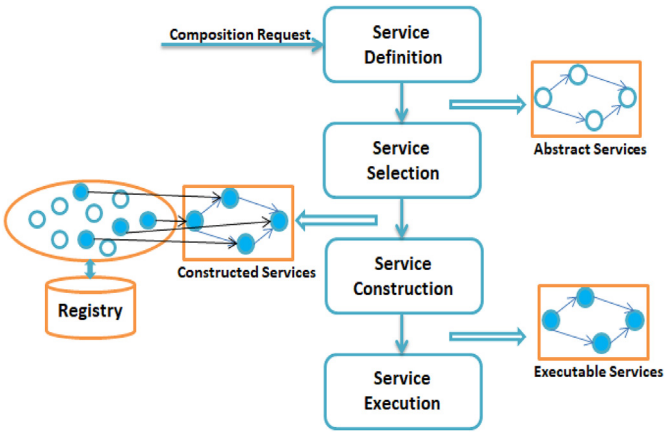

Fig. 1. Services composition lifecycle [34].

we focused on selecting the appropriate service for each composing service.

2.1.2. SC lifecycle

The Services Composition lifecycle [33–35] has four primary levels, as shown in Fig. 1

Service Definition: this level is also called the services description and it may be considered from the following two different perspectives:

Provider's perspectives: providers are responsible for describing the service's performance in terms of the functional and non-functional requirements using specific languages. Providing an excellent description of the performance of the service increases the likelihood of its selection and the validity of the resulting composition services.

End user's perspectives: the end user who requests a service defines their preferences and service performance requirements. These data are collected either semi-automatically or automatically in order to abstract a model to determine a set of activities that control the requirements and data flow among them, and to determine the QoS constraints and unusual behaviors.

Service Selection: Service selection is the core of the process SC [3], where appropriate services are selected from different services providers in the services registry that match the performance requirements specified by the end user. The selected services comprise more than one candidate service that satisfies the end user's preferences. Therefore, the optimal matched services must be selected and grouped into a composite service.

Service Construction: The constructed composite service is published to allow its utilization by end users. Thus, the executable composite service is produced at this level.

Service Execution: At this level, the composite service instance is created and executed by the execution engine using the orchestration or choreography engine in order to run the individual service components, and thus the end user service or application. In addition, the monitoring tasks should be performed [33] at this level, including logging, performance measurement, exception handling, and execution tracking. In some automated composition methods, the first two levels are merged and the constructed composite service is generated directly according to the composition requirements [33].

2.2. Analysis of SSA design

2.2.1. SSAs based on process time phase

In the following, we explain the deployment of SSAs based on time phases, which has two main stages.

Static Service Selection: The services are selected at the design time. The different service components that are essential for composition are selected, collected, and then deployed. Early service binding is important because if better alternative services are offered or one of the provided services becomes unavailable, the static composition process will not be able to offer a better choice in SC [33,36]. Static Web services composition is not flexible and adaptable when frequent runtime changes occur, but it is suitable when the service components do not change or they change rarely.

Dynamic Service Selection: The services are selected during the runtime. Thus, the service components can be determined and replaced during the runtime, thereby allowing the consideration of changes that can occur in the in-service components and better alternative services may be offered for the composite services [33,36]. Dynamic service selection is considered a more challenging task than static service selection because critical issues must be addressed, such as time limits, service correctness, and transactional support.

2.2.2. Behavioral workflow management

The behavior of services refers to how the services workflow can be organized and controlled. The services workflow can be managed in the following two ways.

Service Orchestration (Centralized): When several services interact, one centralized service is responsible for managing and controlling the communication and workflows among all of the other services. This central point is called the orchestrator and it organizes the connections between different services. The orchestrator has a global and complete view of the logics of the interactions, and it obtains the Results from various connected services, as shown in Fig. 2(A).

Services Choreography (Decentralized): The services choreography represents distributed services with no central control point. The services interact and connect with other services according to their logic [33], as shown in Fig. 2(B), thereby allowing sensors to interact directly with actuators.

2.2.3. Types of optimization objectives

Optimization algorithms are not novel solutions and they have been developed for decades in order to find the optimal maximum or minimum solution. Optimization algorithms are used widely to solve complex service selection problems, especially QoS selection problems, which have various names such as QoS-driven or QoS-aware selection problems.

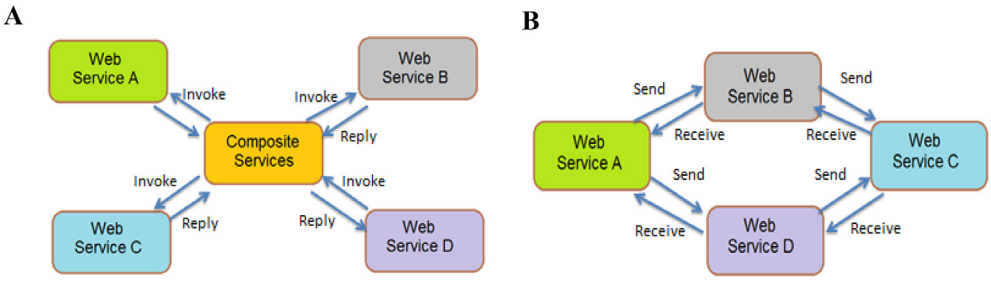

Fig. 2. A) Service orchestration.
B) Service choreography [33].

The three main types based on the number of objective functions solved by the optimization algorithms are as follows.

Single Optimization Problems: These are considered the standard optimization problems where one objective function is optimized at one time.

Multi-Objective Optimization Problems (MOPs): MOPs require that the optimal solution is found in the presence of trade-offs between more than one conflicting objective under a set of specific constraints. No single solution [37] can optimize all of the objectives at the same time. A set of infinite optimal solutions or set of points satisfy all of the predefined conditions for the optimum, which are called non-dominated or Pareto optimal solutions, as defined in 1906 [38], and they are the most widely used solutions in MOPs. The solution is regarded as a Pareto optimal solution if none of the objective functions optimize the fitness value without affecting one or more of the other objective values, or if they make none of them worse. If no further Pareto objective can be changed, the solution is called Pareto optimal.

The main challenge of MOPs is that no best unique solution exists [37]. Thus, only a set of non-dominated solutions can be found to obtain a good approximation of the real Pareto dominance.

Many-Objective Optimization Problems (MaOPs): These problems are extended versions of MOPs, where the MOPs often require the optimization of two or three objectives [37]. MaOPs require the optimization of more than three objectives as the non-dominated number of solutions and solution search space increase. These increases make the service selection problem more complex and it is difficult to achieve diversity and convergence for SSAs in the IoT [39].

2.3. Analysis of solution implementations

2.3.1. QoS based on IoT architecture

The classification proposed for the QoS architecture by Li et al. [40] meets the needs of the IoT environment and allows the QoS to be optimized in different IoT layers. The architecture considers three IoT layers comprising the sensor layer, network layer, and application layer. The traditional QoS attributes are integrated with other essential characteristics of the IoT architecture (e.g., cost of network deployment, information accuracy, energy consumption, and coverage). The traditional QoS classification is not applicable to the heterogeneity and complexity of the IoT architecture.

The sensor layer represents the physical IoT infrastructure including different edge–node links, such as data centers, RFID tags, sensor networks, mobile devices, and other heterogeneous devices. This layer is based on the concept of sensing as an independent service and it allows the IoT environment to provide sensing and actuating capabilities that can be modeled as services using edge–node devices through the cloud computing system.

The QoS in this layer involves selecting the sensor's necessary infrastructure based on the user/application requirements and sensing capabilities. Thus, this layer aims to deal with scheduling the acquisition of information and resource allocation. For example, the QoS in the sensor layer involves the energy consumption, system lifetime, and resource optimization [41].

The optimization of SSA is required to optimize the QoS for sensor services. An optimal SSA is vital for the sensor layer because multiple devices are available with differences in quality that can meet the user/application requirements. The availability of services in this layer determines the success or failure of a service request.

The *network layer* represents the IoT network infrastructure responsible for transferring data and information between sensor nodes. This layer includes different protocols, technologies, and edge networks, such as RFID tags, wireless sensor network (WSN), wireless local area network, or cellular network. The importance of the network layer is related to the network services in the IoT, which require the presence of various network technologies. Similar to WSN, it is necessary to provide universal coverage points and access to network nodes. In the network layer, the heterogeneous network environment is scheduled according to the traditional QoS in order to allocate various network resources. The different protocols used in heterogeneous networks depend greatly on the QoS requirements, e.g., the bandwidth, capacity, and throughput QoS in the network layer [40,41]. An optimal SSA is essential in the network layer because it allows service providers to provide optimal network technologies and protocols to meet the user/application requirements. Efficient QoS support for different network infrastructures in this layer is challenging when developing applications for use in the IoT.

The *application layer* represents the highest layer in the IoT architecture and it comprises many distributed services, which are combined to provide one service to the end user or an application. Many applications are deployed in different domains in the IoT environment, such as smart home, industrial automation, health care, and traffic management applications. The *application layer* aims to schedule services according to the QoS constraints. Under the QoS constraints, some network resources must be allocated to services selected in the application layer, thereby affecting the end services and user requirements, e.g., the accuracy, reliability, availability, and execution time QoS in the application layer [41]. Thus, it is crucial to select an optimal SSA and resource allocation scheme based on the information for each component service.

2.3.2. Search optimization algorithms

Search optimization algorithms are used to select the optimal solution in a specific search space and they may be classified into four main types comprising heuristics, meta-heuristics, hyper-heuristic algorithms, and non-heuristic algorithms.

The term *heuristic* comes from the Greek εὑρίσκω, which means "I find, discover." A heuristic is a type of artificial intelligence and a dependent optimization algorithm designed to find an approximate solution for complex problems when traditional algorithms cannot obtain an exact solution [42]. The main objective of heuristics is to obtain a satisfactory and valuable solution in an acceptable time, but not necessarily the exact solution. The main idea of heuristic algorithms involves ordering an alternative solution for each step based on the available information for the problem in order to decide the next step, which involves iterating the algorithm's rules and utilizing the output from the previous step as an input for the next step until the optimal solution is reached.

The term *meta-heuristic* was introduced by Glover in 1986 as a combination of the Greek prefix meta- (metá), which means high level, and heuristic (from the Greek heuristic or euriskein, which means to search or find) [43]. A *meta-heuristic* is a high-level optimization algorithm that identifies a set of concepts grouped in an algorithmic framework in order to provide a set of strategies or rules for improving heuristic optimization algorithms to find a suitable solution to a specific problem [43]. Meta-heuristics can be defined as general-purpose heuristic methods that can obtain high-quality solutions in the significant search space.

A *Hyper-heuristic* is a heuristic search method introduced in 1997, which involves the combination of more than one artificial intelligence algorithm in the context of automated theorem proving [44]. This term was subsequently applied in a combinatorial optimization context to denote *heuristics for choosing heuristics*. The main idea of hyper-heuristic algorithms is automating the use of high-level heuristic methodologies to select, combine, generate, and apply algorithms to a specific problem, which involves combining more than one appropriate heuristic method at each decision point to efficiently solve computational search problems [45].

Non-heuristic Algorithms are optimization algorithms for optimizing the search of a space and obtaining an optimal solution. In this class, the solution is found without employing evolutionary algorithms and an iteration process is not followed to find an optimal solution.

2.4. Performance evaluations

Researchers must evaluate the behavior or performance of SSAs after they have been developed. Our survey showed that two main methods are used to measure the performance of the proposed solution. Computer

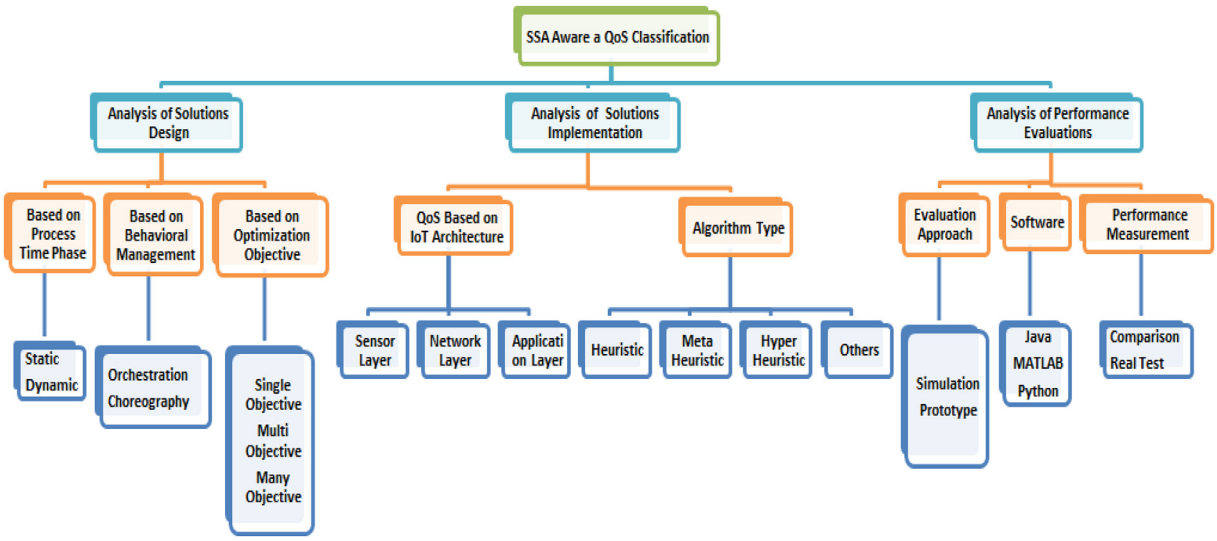

Fig. 3. Classification of QoS-aware SSAs in the IoT environment.

simulation is the most widely used method. Computer simulation involves mathematical modeling on a computer using simulation software in order to predict the behavior of a physical system or the real world. Computer simulation is a useful tool for mathematically modeling many natural systems, e.g., in computational physics, chemistry, manufacturing, biology, and human systems such as psychology, economics, health care, and engineering.

The other method employed involves using a prototype. A prototype is an early stage model or a sample of a specific product built to test processes and concepts. In general, a prototype is designed to evaluate a new design of a product and to enhance precision for system users. A prototype is used in various contexts, including semantics, electronics, design, and software programming [46]. The difference between a simulation and prototype [47] is that a simulation is an analytical process representing a real working system, whereas a prototype is a real physical object that provides specifications for the actual system.

3. Proposed classification approach for QoS-AWARE SSAS

In the following, we explain the proposed classification approach for QoS-aware SSAs used in the IoT environment. The state-of-the-art research approaches can be classified into three main types, as shown inFig. 3. Our classification criteria were obtained by analyzing the state-of-the-art methods. In order to develop a new method to solve the selection problem in the IoT environment, three main points should be considered. First, it is necessary to design an appropriate environment that allows the implementation of the proposed solution. Second, the proposed solution type is applied for selection in the QoS layer. Third, it is necessary to determine the methods used to evaluate the proposed solutions.

Thus, based on these three requirements for developing a new SSA, our proposed classification approach is described as follows. First, the design of SSAs for IoT must adhere to three basic concepts. SSAs are based on the process time phase where the time when the SSA is performed is specified. The workflow management behavior represents how the services are controlled and connected. The objective of the optimization algorithm is to establish the number of goals to satisfy when the algorithm is implemented.

Second, we consider two basic concepts related to the implementation of SSAs. Based on the QoS layer, it is necessary to define the layer where the SSA will be applied in the QoS architecture [40]. Moreover, the algorithm type can be divided into heuristics, meta-heuristics, hyper-heuristics, and other algorithms (non-heuristic algorithms).

Third, the evaluation approaches, software, and performance management can be used to evaluate the performance of proposed algorithms. Each of these sub-classifications is defined in Section 2.

Based on the definitions and concepts described in Section 2, we classified the state-of-the-art methods as shown in Fig. 3.

4. Analysis of solution DESIGNS

To design a new solution for the selection problem, researchers must first define an appropriate environment. Thus, we reviewed the state-of-the-art methods and identified three common categories comprising SSAs based on the process time phase, behavioral workflow management, and optimization objective type. Detailed definitions of these features are provided in Section 2. The methods analyzed according to these three categorizes are shown in Table 2.

5. Analysis of solution implementations

Many methods have been proposed to solve the QoS-aware selection problem in the IoT environment. We identified these solutions based on the three QoS layers introduced for the IoT architecture by Li. et al. [40], i.e., the sensor layer, network layer, and application layer. For each layer, we classified the implementations of the methods according to four optimization algorithms comprising heuristics, meta-heuristics, hyper--heuristics, and non-heuristic algorithms. Moreover, we considered the traditional QoS factors optimized in each proposed solution.

5.1. Sensor layer

A. Heuristic algorithms in the sensor layer

Yin et al. [49] addressed the single-source many-target k shortest paths problem in Map-Reduce, where they used a graph of Web services to find k shortest paths (selected services) from a candidate set of services for one source node while providing acceptable QoS. They focused on reducing the execution time and consuming less power. They proposed an efficient pruning algorithm for breadth-first search (BFSKNN) called PruningBFSKNN algorithm. The proposed algorithm built based on find the shortest paths between nodes in a graph using DijkstraKNN and BFSKNN for the single-source many-target shortest path problem.

Shukla et al. [55] introduced a collocation-based strategy for hosting IoT services and applications in devices where they considered a smart home scenario. They aimed to find the correct sensor for the required services from a set of sensors while satisfying the QoS objectives by

Table 2
Categorization of state-of-the-art methods according to the solution design.

Reference	Process Time Phase		Behavioral Workflow Management		Optimization Objective Type	
	Dynamic	Static	Choreography	Orchestration	MOPs	MaOPs
Huang et al., 2014 [5]	✓		✓			✓
Liu et al., 2013 [11]	✓		✓			✓
Na et al., 2015 [48]	✓		✓		✓	
Zhou et al., 2016 [12]	✓		✓			✓
Sun et al., 2017 [6]	✓		✓		✓	
Reddy et al., 2017 [17]	✓			✓		✓
Mejri et al., 2017 [8]	✓		✓		✓	
Yin et al., 2014 [49]	✓		✓		✓	
Huang et al., 2015 [14]	✓			✓	✓	
Dhondge et al., 2016 [7]	✓			✓	✓	
Anas et al., 2016 [50]	✓		✓		✓	
Gao et al., 2014 [51]	✓		✓			✓
Abinaya et al., 2017 [16]	✓		✓			✓
Nwe et al., 2014 [52]	✓			✓	✓	
Jin et al., 2014,2016 [53,54]	✓			✓		✓
Perera et al., 2014 [10]	✓		✓			✓
Yu et al., 2014 [19]	✓		✓			✓
Huang et al., 2014 [20,21]	✓		✓		✓	
Shukla et al., 2018 [55]	✓		✓			✓
Elhoseny et al., 2018 [22]	✓		✓			✓
Alsaryrah et al., 2018 [18]	✓			✓	✓	
Huang et al., 2014 [15]	✓		✓		✓	
Lin et al., 2017 [39]	✓		✓			✓
Khanouche et al., 2016 [13]	✓	✓		✓	✓	
Yuan et al., 2019 [56]	✓		✓			✓
Hosseinzadeh et al., 2020 [57]	✓		✓	✓	✓	
Gao et al., 2020 [58]	✓	✓	-	-	✓	
Quan et al., 2019 [59]	✓		✓			✓
Jatoth et al., 2019 [60]	✓		✓		-	-
Khan et al., 2019 [61]	✓			✓		✓
Abu-safe et al., 2019 [62]	✓		✓			✓
Singh et al., 2020 [63]	✓		-	-		✓

mapping flow-based process components on the system sensors. They identified three key factors comprising minimizing the latency time when collecting and transferring data from IoT devices and communicating it to the gateways, reducing the energy used by the system, and balancing the system energy requirements to increase the system lifetime. They proposed a collocation-based sensor-service mapping strategy (CBSSMS) to link the IoT services to appropriate sensors. The approach used two algorithms where one found the path that reduced the latency time and a collection algorithm assembled the components that formed a link on the same sensor.

Alsaryrah et al. [18] aimed to select an appropriate set of smart objects by considering the traditional QoS and the energy consumed to form a service. They divided their objectives into minimizing the traditional QoS (execution time, network latency, and cost) and reducing the energy consumption by the combined service, i.e., by maximizing the battery lifetime for the sensors. They proposed a bi-objective shortest path optimization (Bi-SPO) algorithm with four pruning techniques comprising pruning by the cycle, nadir point, efficient set, and label.

Perera et al. [10] investigated the properties of sensors and the information associated with data streams to search, select, and rank large-scale sensors with the same functionalities and capabilities in order to satisfy the user's requirements. They considered the user preferences and sensor characteristics, such as the accuracy, reliability, battery life, location, and other features, to identify appropriate sensors for data collection approaches. They designed and implemented an ontology-based context-aware sensor search, selection, and ranking model (CASSARAM). To improve the performance, efficiency, and capability of CASSARAM, they proposed a comparative-priority based weighted index to remove sensors with a lower weighted context property based on the user preferences technique called Top-K selection, comparative priority-based heuristic filtering to reduce the number of sensors ranked by removing sensors placed away from the user, relational expression-based filtering to speed up the search process and sensor selection by specifying an acceptable range of context property values using relational operators in semantics, before ranking the sensors by considering the account user's priorities, and distributed sensor searching to identify and select appropriate sensors that satisfy the user's requirements on multiple IoT distributed servers. Three different methods were identified to search distributed sensors based on query/data transformation over the network, i.e., chain processing, parallel processing, and hybrid processing.

B. Meta-heuristic algorithms in the sensor layer

Lin et al. [39] introduced a sensor selection algorithm for specifying multiple sensor devices in a large-scale environment. They defined their optimization parameters for energy and distance minimizing the energy consumed by communication between two sensor devices, balancing the energy among different sensors to reduce overloading on some sensor devices, maximizing the total energy harvested by supplementing the sensor's battery energy with natural energy (e.g., wind and solar), and green index optimization to reduce the total pollution level. Satisfying the QoS involved optimizing the cost, reliability, and availability of IoT services. Their proposed algorithm based on many-objective evolutionary algorithm decomposition (MOEA/D) solved larger-scale problems by decomposing them into multiple sub-problems and then finding the optimized solution for each sub-problem.

Na et al. [48] conducted service selection for IoT based on physical resources by using platform-independent middleware. They focused on increasing the IoT system lifetime by using the power on all devices equally to reduce the energy consumption and costs. To balance the energy consumption, they developed an evolutionary game approach by defining a fixed point of the replicator dynamics where the payoffs are equal for all players in the same group. They also presented some options for improving the service selection behavior. The initialization step was improved by estimating the remaining lifetime for all devices at the beginning instead of selecting them randomly. This approach may

require a long time to find the optimal initialization solution, but it will save time in the following steps by maintaining communication with the other selection process. The decision-making step was improved by allowing the algorithm to select a longer estimated remaining lifetime, which could accelerate the algorithm and reject the optimal solution when selecting a service with a shorter lifetime.

C. Non-heuristic algorithms in the sensor layer

Khanouche et al. [13] aimed to solve MOPs during service selection by managing the energy consumption and maintaining the availability of services, while slightly reducing the QoS level but without affecting the user satisfaction. They designed a QoS model by describing the QoS of an atomic service divided into quantitative attributes comprising the traditional QoS, such as the cost, response time, reputation, reliability, and availability, and qualitative attributes, such as security, privacy, and comfort. The QoS of a composite service is dependent on the structure of its atomic services connected through a sequential structure. The relative dominance of services conforms to the Pareto optimality set, which comprises the collection of possible solutions where at least one objective is optimized without affecting other goals. They proposed an energy-centered and QoS-aware service selection algorithm (EQSA) by effectively selecting a user-centered service from the most appropriate services that match the user's preferences while satisfying the specified QoS level. The proposed solution is executed in two main phases comprising pre-selection of services that provide the required QoS level for the user's and static selection before the runtime. The most appropriate services for SC are selected according to the relative dominance of the services.

5.2. Network layer

A. Heuristic algorithms in the network layer

Huang et al. [15,19–21], and [14] presented a service merging approach that maps and co-locates neighboring virtual service on the same physical devices to reduce the communication energy costs and to balance the energy consumption by sensors to prolong the system lifetime. They applied WuKong middleware to automatically discover and manage smart sensors and actuator devices, which could support flexible and interoperable IoT systems by selecting from predefined flow-based programs (FBP) to find the appropriate mapping to the abstraction of an application onto physical smart devices and actuators according to the QoS requirements. They proposed an energy sentient algorithm called the maximum weighted link (MWL) algorithm that treats selection as a two-co-locating problem and ignored the distance between devices [21]. They also updated their model [20] to consider the distance between devices because two remote devices require more energy for communication than closer devices. They treated the problem as a quadratic programming problem and proposed a reduction method to transform the problem into an integer linear programming (ILP) problem.

Huang et al. [19] presented a mapping model that considers the distance and runtime QoS requirements, such as the accuracy response time. Moreover, they attempted to reduce the total communication energy in IoT systems during each new update. They modeled the service matchmaking problem as a maximum weighted bipartite problem and solved it using the ILP model. Huang et al. [14,15] also used strategies for solving the maximum weighted independent set (MWIS) in their selection framework, which considered all possible co-location combinations for services. They implemented this method only in single-hop networks and treated the problem as a data clustering problem [15]. Heuristic algorithms were employed to find the maximum weight for the independent set, which comprised the basic decisions regarding service co-location. However, the method was subsequently implemented in a single-hop network and multi-hop network by modeling the problem as a quadratic programming problem and solving it with the ILP model [14].

Dhondge [7] presented a study of industrial IoT (IIoT) systems where they focused on collecting and controlling communication data and parameters obtained from sensors on factory floors. They aimed to reduce and balance the energy consumption in the IoT by proposing a heuristic and opportunistic link selection algorithm (HOLA) to maintain the energy efficiency in the IoT sensors by opportunistically transporting the IoT sensor data to smart devices. These intelligent devices had multiple radio links (3G/4G LTE, Wi-Fi, and Bluetooth) to transmit the received data to the cloud by using HOLA to select the best radio link based on the quality preserved by the Services Level Agreements and the energy cost of the relationship.

B. Meta-heuristic algorithms in the network layer

Sun et al. [6] proposed a solution for integrating and co-operating with smart IoT functionalities to satisfy the user requirements, which could be applied to more than one smart thing. They considered the conflicts between the device's features and balancing its energy consumption. Spatial constraints were defined by the physical location and communication radius of a smart thing on the IoT. The temporal constraint was defined as the specific time duration required to meet a user's requirement because smart things should only be available for predefined time duration. The energy efficiency was considered by balancing the energy load of intelligent devices or things and avoiding excessive consumption. Configurability of the IoT services occurred when two IoT services instantiated on the same smart device and their functionalities conflicted. Thus, the composition and selection of the services demanded consideration of the delay between two sequential services by building a set of aggregated alternative smart things. The two main QoS factors involved reducing and balancing the energy consumption, and prolonging the network lifetime. A two-tier framework was proposed and three different meta-heuristic algorithms (ant colony optimization (ACO), GA, and PSO) were implemented to search for the optimal IoT SCs. The two-tier framework was configured as shown in Fig. 4. The IoT smart thing tier encapsulated the functionalities of devices in IoT services. The services class tier categorized IoT services into service class chains using traditional Web service composition techniques. The service network was created between the two tiers by considering the possibility of calling between service classes.

Reddy et al. [17] proposed a clustering method for dividing a WSN-based IoT network into a small network, where each was called a cluster head. By using a meta-heuristic algorithm to optimize the network communication, the clustering method could divide the WSN into a small, reliable, and manageable network with efficient data transmission. The proposed algorithm used five parameters comprising the distance, energy, delay, network load, and temperature of the IoT devices. To efficiently select the cluster heads, a novel method was proposed by combining the gravitational search algorithm (GSA) with the artificial bee colony (ABC) algorithm.

Abinaya et al. [16] also aimed to minimize the energy, time, and loss of packets during the transfer of data among nodes when selecting and combining services. They aimed to increase the energy efficiency, time utilization, and throughput without any loss of data or reduction in the packet delivery ratio. They proposed a meta-heuristic algorithm (ACO) to find the shortest path between the nodes using the network routing protocol approach. The algorithm clustered the data nodes before then transferring data between the nodes with low power consumption.

C. Non-heuristic algorithms in the network layer

Khan et al. [61] proposed a QoS-aware secured communication approach for IoT-based networks called QoS-IoT. The Sybil attack detection mechanism was used for identifying Sybil nodes during multi-hop communication. To ensure the fair and efficient utilization of the available bandwidth, an optimal contention window (CW) was selected for QoS provisioning. The optimal CW size was selected by using the binary exponential back-off mechanism. The performance of QoS-IoT was evaluated based on measurements of Sybil attack detection, fairness, throughput, and buffer utilization.

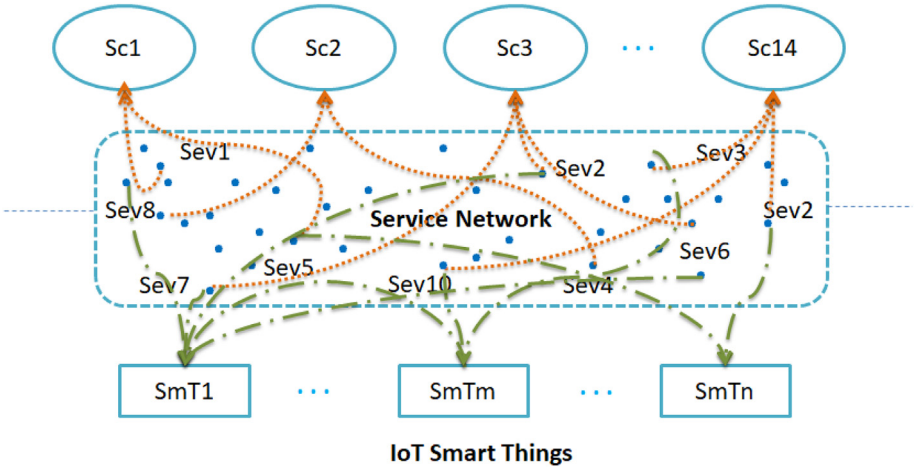

Fig. 4. Two-tier framework [6].

5.3. Application layer

A. Heuristic algorithms in the application layer

Nwe et al. [52] introduced a matching, ranking, and selection model to satisfy the distributed needs of dynamic networks in IoT environments. They selected services based on two factors to optimize the QoS, i.e., objective information supplied by the service providers and subjective information provided by the service consumers. To select a service, they proposed a flexible QoS-based service selection algorithm (FQSA). They calculated the subjective factors for user with a similarity aggregation method (SAM)) to evaluate the creditability of different users. Moreover, the user's input was extracted using the QoS ontology, WordNet, and ontological reasoning. To help understand the QoS characteristics, they analyzed a separate language glossary and evaluated the consistency among the QoS criteria chosen by the end users. The FQSA algorithm employed an artificial neural network back-propagation algorithm (ANN-BP) to find the objective factors and improve the selection performance rate for acceptable real-time service selection. They also provided a flexible, user-friendly assessment form to allow users to request any number of QoS criteria.

Mejri et al. [8] investigated the scalability of service selection in the IoT by using a self-adaptive approach based on a combination of a QoS prediction model, which considered the user context, service context, and network context by using the ANN, and the technique for order of preference by similarity to ideal solution (TOPSIS) model to introduce the best service to the consumer of the service. They optimized two QoS parameters comprising the response time and reliability.

Quan et al. [59] introduced a reinforcement learning approach called the linear reward inaction (LRI) algorithm in real time. They considered the dynamic IoT environment by calculating the user's mobility, which could affect the accessibility and connection location of services, thereby reducing the search space for service discovery. The latest subjective assessment obtained from user feedback concerning the user context similarity was used to estimate the QoS in a similar environment. A subjective evaluation was conducted by calculating four factors: privacy, reliability, availability, and response time. Three objective attributes comprising the availability, response time, and calculation speed were determined by obtaining a score for each service. The service with the highest score was selected.

B. Meta-heuristic algorithms in the application layer

Liu et al. [11] designed a cooperative evolution algorithm (CEA) for service composition and selection to solve MOPs when selecting an optimal service from a group of services with similar functionalities and diverse QoS requirements. They aimed to develop an efficient and robust approach by considering non-functional attributes comprising the cost, time, availability, and reliability. A heuristic optimization approach was developed by integrating GA and PSO in CEA. Their approach was characterized by improving the best local first strategy to select a service candidate, enhancing the global best policy, and fitting the self-adaptive mechanism for the learning rate.

Gao et al. [51] conducted global optimization for event SCs by using a meta-heuristic method based on GA but without the need to consider all possible combinations. The non-functional attributes were represented by QoS properties for the latency, price, energy consumption, bandwidth consumption, availability, completeness, accuracy, and security.

In particular, they provided a QoS aggregation schema for complex event service (QoS-AS for CES) composition in CES networks by treating complex event processing as reusable services where reusability was determined by examining intricate event patterns and primitive event types. The abstract architecture of these complex networks is shown in Fig. 5.

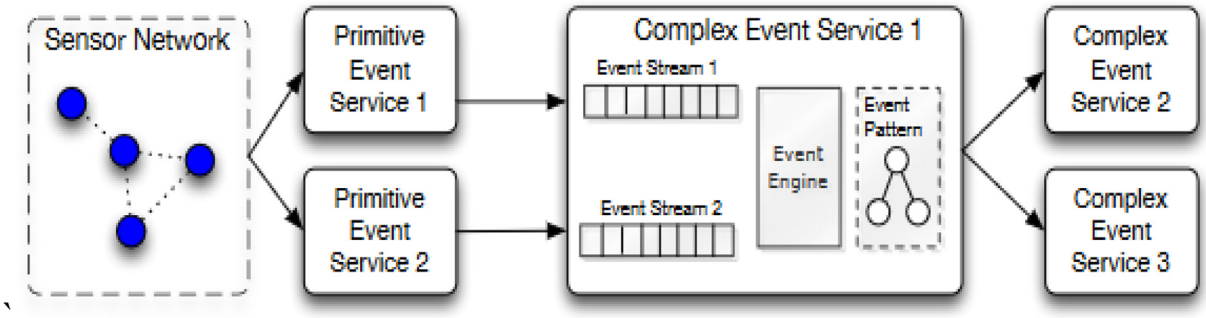

Fig. 5. Architecture of CES networks [51].

A GA was also developed to efficiently create optimal event SCs with the same standard GA steps (select, crossover, and mutate until the termination conditions are satisfied) but some differences from the standard implementation. A tree encoding schema was maintained by an event reusability forest.

Anas et al. [50] aimed to simulate human thinking when making a decision about multiple choices and using data collected from IoT sensors. They considered an example of a human making a decision while driving when faced with two paths that lead to the same place but with different trade-offs in terms of the time, distance, or cost. Their framework collected data to help systems to select their future path. The main problem was how to capture and use human heuristic information. The final solution reduced the total time and obtained more accurate Results. They used the heuristic-IoT framework for enhancing heuristic search algorithms and collecting data from IoT sensors. They implemented their framework with a GA using data regarding the habits and behavior of drivers collected from sensors deployed in taxis to solve the travelling salesman problem (TSP) with hidden edge costs. The proposed framework used heuristic information to generate smarter initial solutions for the GA to solve the TSP instead of generating it randomly.

On the cloud manufacturing side, Huang et al. [5] focused on solving the MOPs for cloud SC optimal selection (CSCOS) while considering non-functional QoS factors. They determined four parameters comprising the cost, execution time, energy consumption, and reliability. Non-functional attributes were considered for three types of cloud services: manufacturing software, hardware, and human resource services. They introduced a new chaos control optimal algorithm (CCOA) to solve the CSCOS problem in large-scale solution spaces.

Li et al. [9] focused on a cloud logistics platform based on IoT and cloud computing environments in order to study a logistics center by considering service encapsulation and resource virtualization. They defined the primary requirement of the logistics center in terms of service selection as how to find the best actual Web services rather than the best combination of abstract Web services. Non-functional constraints were used to compute the QoS for composite services by applying rules of Canfora [9] for an aggregation function and four QoS parameters comprising the time, cost, availability, and reliability. In addition, a dynamic service selection model was proposed based on PSO.

Abu-Safe et al. [62] proposed a service selection model that ranked services based on end user feedback and the reputation value. The Likert scale was employed as a user-friendly method for acquiring feedback from end users. An improved-PSO was used to select the optimal service from ranked services. Two quality groups used to calculate the QoS factors comprised the business quality group (BQG), i.e., reputation and execution price, and the system quality group (SQG), i.e., reliability, availability, and response time.

Jatoth et al. [60] introduced a meta-heuristic model using an adaptive genotype evolution-based GA (AGEGA). They balanced the QoS parameters and connectivity constraints to perform SC in a cloud environment. The discrete uniform rank distribution (DURD) and discrete uniform service rank distribution (DUSRD) were proposed to determine the service fitness and SC fitness, respectively, thereby allowing services to be pruned from the non-optimal solutions and reduce the search space. The specific QoS parameters employed were not defined. However, they used a synthetic data set with QoS parameters such as accessibility, cost, availability, throughput, response time, security, integrity, and reliability.

C. Hyper-heuristic algorithms in the application layer

Elhosenya et al. [22] considered health services applications where they proposed a new cloud-IoT based model for efficiently managing large amounts of data in an integrated industry (4.0) environment. They aimed to satisfy five factors, reducing the medical requests time (execution time, waiting time, and turnaround time), optimizing the storage space for patient data, improving task scheduling, providing a real-time data retrieval mechanism for health care applications, and maximizing the utilization of resources. They proposed a new model for optimizing virtual machine selection by using three optimization algorithms (GA, PSO, and Parallel PSO (PPSO)) to build the proposed model.

Zhou and Yao [12] focused on cloud manufacturing by introducing a solution for composited cloud manufacturing service optimal selection (CCSOS) under multi-objectives using four QoS parameters comprising time, cost, availability, and reliability. They introduced the hybrid ABC (HABC) algorithm for CCSOS problems with three main steps. First, the HABC was initialized before outputting feasible solutions at each iteration and these solutions were ordered from large to small settlements. The onlooker strategy was improved by updating all the solutions with the chaos algorithm, which had irregular properties in all states and it could help the worst bees. The bee colony's search space was searched more efficiently based on knowledge of the problem structure and social colony information by updating small solutions with Archimedean copula estimation of distribution.

Yuan et al. [56] proposed a dynamic approach by used a fuzzy logic technique and cultural GA to adapt the global QoS constraints. The global QoS constraints were decomposed into near-optimal local QoS constraints, before independently selecting a service component for each abstract service. They aimed to satisfy five QoS factors comprising the price, response time, availability, throughput, and successful execution rate.

Hosseinzadeh et al. [57] combined a machine learning method with a meta-heuristic algorithm in a hybrid ANN-PSO algorithm. They aimed to improve the execution time and reachability rate for a service selection model in cloud-edge computing. A labeled transition system was proposed based on a verification approach to check the correctness of the proposed model. Three QoS factors were considered comprising the response time, availability, and prices.

5.4. Aggregate layers

In some methods, the models were aggregated together in more than one layer, such as in the sensor layer and application layer [53,54].

Jin et al. [53] considered the services provided by IoT devices and designed a physical service model (PSM) to describe various physical IoT services as well as a method for selecting a candidate physical service that satisfies a user's requirements. The PSM model included three main components (devices, resources, and services) and the relationships between them were defined. The following four QoS properties were determined based on the features of physical services: the available time when environmental services may be accessible, the service area comprising descriptions of physical services for on-device resources that contain information about an entity, the processing time representing the computational time capacity of IoT devices, and the reputation calculated for a service to help users decide whether to use a service based depending on the service ratings given by different users (equipment or service) after requesting services. A physical service selection (PSS) algorithm was proposed in terms of spatio-temporal features to rate candidate physical services according to user preferences based on individual QoS rating functions. The PSS algorithm comprised pre-sorting, filtering, and final sorting phases.

In 2016, Jin et al. [54] improved their PSM to dynamically rate QoS values and select a physical service based on the user's preference. They added the following three types of QoS attributes to reflect the features of physical services: spatial-temporal attributes related to the problems that affect the mobility and availability of physical services due to network, energy-saving, or privacy issues, i.e., available time and service area; positive features that preferably have higher values, i.e., reputation and reliability; and negative features that preferably have lower values, i.e., processing time and execution cost.

Gao et al. [58] aggregated the application layer and network layer to provide recommendations for services and allow the selection of services, where they proposed a holistic framework for predicting the QoS values in the IoT. The proposed framework employs fuzzy C-means to cluster contextual information related to users and services, such as the network

location and geographic location, and neural collaborative filtering (NCF) is applied as a neural network model to learn the in-depth latent features. NCF utilizes local features such as similar users or similar services with historical QoS values and global elements comprising user latent vectors and latent service vectors. NCF then combines the contextual information with the historical QoS values to perform both the prediction and selection processes.

Jatoth et al. [60] also aggregated the application layer and network layer with a meta-heuristic model using AGEGA, where they balanced the QoS parameters and connectivity constraints to perform SC in a cloud environment. DURD and DUSRD were applied to determine the service fitness and the SC fitness, respectively, thereby allowing services to be pruned from the non-optimal solutions to reduce the search space. However, they did not define the specific QoS parameters used in their study, although they described the use of a synthetic data set containing QoS parameters, such as accessibility, cost, availability, throughput, response time, security, integrity, and reliability.

Singh et al. [63] introduced a framework based on multi-criteria decision making for directing the selection process. The framework aggregated the sensor, network, and application layers, and they combined two multi-criteria decision making methods comprising the analytic hierarchy process (AHP) and TOPSIS. AHP was used to calculate the weights for the QoS criteria and TOPSIS ranked the service providers. They describe a QoS parameter based on three IoT components, i.e., things, communication entity, and computing entity. Nine QoS parameters were considered: operating temperature range, resolution, accuracy, delay, jitter, pricing, availability, throughput, and response time.

5.5. Analysis of solutions for implementing SSAs

The analyses of selected state-of-the-art solutions implementations are categorized, as shown in Table 3. It is categorized based on the proposed solution, QoS layer, algorithm type, and QoS parameters.

6. Analysis of performance evaluations

Most previous studies used simulations to evaluate the performance of their proposed algorithms and they conducted comparisons with other SSAs in the same environment. In addition, prototypes were constructed in other studies to evaluate the performance of their proposed algorithms. In some cases, simulations and prototype were combined in the evaluations. In the following, we discuss the three methods used to evaluate the state-of-the-art algorithms.

6.1. Using simulation software to evaluate SSAs

Huang et al. [5] demonstrated the high performance of their proposed CCOA algorithm in CSCOS based on simulations. They found that their algorithm was better at searching large-scale solution spaces than GA and typical chaotic GA, where it reduced the time required and energy consumption. They recommended improving the effectiveness of CCOA to solve other combinatorial optimization problems by balancing the search capacity and time consumption, and tested the effects of other QoS factors.

Liu et al. [11] developed CEA by integrating GA and PSO. They conducted simulations and generated a data set at different scales based on real scenarios. They showed that CEA was a highly efficient search approach with greater stability and more rapid convergence compared with canonical PSO (CPSO) and the improved discrete immune algorithm based on CPSO (IDIPSO). They recommended extending the experiments to greater scales.

Na et al. [48] showed that their method decreased the time required for implementation and increased the rate of service utilization. When the service utilization reached 100%, the algorithm could not make any changes after initialization. They recommended focusing on group-based service selection to reduce the communication energy requirements.

Zhou et al. [12] evaluated the performance of HABC based on comparisons with GA, PSO, and the basic ABC algorithm using 15 different user QoS preferences. They randomly generated the data set and the Results showed that HABC exhibited a high search capacity and stability with acceptable time complexity. They recommended studying the performance of HABC in detail according to the characteristics of the cloud manufacturing environment, and integrating HABC with other heuristic algorithms.

Reddy et al. [17] assessed the performance of the GSA and ABC algorithm based on the trends in the network sustainability of live IOT nodes in the network and by evaluating its convergence compared with PSO, GA, ABC, and GSO. The Results demonstrated that their approach performed better than the other methods at cluster head selection for IoT devices.

Yin et al. [49] showed that the pruning algorithm was more efficient than the breadth-first search shortest path algorithm. Furthermore, the DijkstraKNN algorithm was suitable for small shortest paths, but the PruningBFS algorithm was better when the candidate set was small and the shortest paths were significant. In addition, the execution times were more stable for the BFSKNN and PruningBFSKNN algorithms according to tests using two real-world data sets comprising the Epinions Social network, and LiveJournal social network. They recommended identifying approximation algorithms that can handle more significant graphs where time is required to compute several nearest neighbors.

Anas et al. [50] used the T-drive data set based on 10,357 taxi drivers and compared the proposed Heuristic-IoT framework with GA. They found the quality of the TSP Results improved by up to 49% compared with the traditional TSP.

Abinaya et al. [16] compared the ACO algorithm with load balancing clustering for data clustering. They also compared the number of nodes versus the data throughput for time-efficiency prediction, where ACO reduced the time and energy required. ACO could find an almost optimal solution and it could be scaled to large-scale IoT environments.

Nwe et al. compared FQSA and other SSAs including (genetic algorithm and fuzzy logic based service selection algorithm (GAFLSS), agent proxy on user preference approach (APUP), and fuzzy linear programming approach (FLP)) [52], from the reliability of the trust mechanism-based service selection algorithm. They calculated the symmetric mean of the recall and precision for each system using the frequency for various evaluation metrics (QoS aggregation, QoS reasoning, QoS scalability, personalized confidentiality, and user friendliness). The Results showed that FQSA improved the service selection performance, user satisfaction level, and user friendliness rates. However, many computations were required to select the services, which was time consuming.

Jin et al. [53,54] evaluated the PSS method against the skyline-based algorithm called the one-pass algorithm (OPA) in terms of the execution time and user preferences for physical services. They used a random data set based on the Climatography of the United States Number 81 series (CLIM8144) data set. The Results showed that PSS was efficient with a large number of candidate physical services [53] and it performed better than OPA in the filtering step by reducing the number of CPUs to improve the selection performance [54]. For future research, they suggested creating and implementing an IoT service platform that allows users to register their devices, and to discover and select required physical services, as well as addressing privacy and security issues, and reducing the search space. They suggested the use of pruning methods and heuristic techniques.

Huang et al. [14,15,19–21] conducted simulations to assess the performance of their algorithm. Their algorithm reduced the total communication energy by about 20% in IoT systems [20,21]. They compared the performance of greedy matching and the ILP solution at service matching and found that the ILP solution was optimal but it required more time, and it might not be scalable to large-scale IoT systems [19]. In addition, they compared ILP [14,15] and the MWIS framework with MWL [19–21], as well as with other selection strategies (GWMAX, GWMIN,

Table 3
Categorization of selected state-of-the-art methods based on the proposed solution, QoS layer, algorithm type, and QoS parameters.

Proposed Solution	QoS Layer			Algorithm Type				QoS Parameters
	App	Sen	N·W	Heu	Meta-Heu	Hyper-Heu	Non-Heu	
New CCOA to solve CSCOS [5]	✓				✓			• Cost • Execution time • Energy consumption • Reliability
Integrating GA and PSO algorithms [11]	✓				✓			• Cost • Time • Availability • Reliability
Evolutionary game approach with platform-independent middleware [48]		✓			✓			• Save energy • Cost
HABC algorithm for CCSOS [12]	✓					✓		• Time • Cost • Availability • Reliability
Two-tier framework based on ACO, GA, and PSO [6]			✓		✓			• Reduce energy consumption • Prolong the network lifetime
Novel method that combines GSA and ABC [17]			✓		✓			• Reliable and manageable network • Efficient data transmission
Self-adaptive approach including ANN and TOPSIS models [8]	✓			✓				• Response Time • Reliability
DijkstraKNN, BFSKNN, and PruningBFSKNN Algorithms [49]		✓		✓				• Reduce execution time • Consume less power
HOLA in IIoT systems [7]			✓	✓				• Reduce energy consumption • Balance energy consumption across the IoT network
Heuristic IoT framework based on GA [50]	✓				✓			• Reduce the total time • Obtain more accurate Results
ACO [16]			✓		✓			Increase: • Energy efficiency • Time utilization • Throughput • Packet delivery ratio
FQSA comprising ANN-BP and SAM [52]	✓			✓				Specific QoS not defined
PSM model and PSS algorithm [53,54]	✓	✓					✓	In [53]: • Available time • Service area • Processing time • Reputation In [54]: • Spatial-temporal attributes • Positive attributes • Negative attributes
MWL [19–21]			✓	✓				• Reduce communication costs • Balance energy consumption
MWIS [14,15]			✓	✓				• Reduce communication costs • Balance energy consumption
CBSSMS [55]		✓		✓				• Minimize latency time • Reduce energy costs • Balance the system energy to increase the system's lifetime
New model based on GA, PSO, and PPSO [22]	✓					✓		• Reduce time • Optimize the required storage • Improve scheduling tasks • Provide real-time data retrieval mechanism • Maximize resource utilization
Bi-SPO [18]		✓		✓				• Minimize traditional QoS • Reduce energy consumption
MOEA/D Approach [39]		✓			✓			• Energy consumption • Energy balancing • Energy harvesting • Optimize the green index • QoS objectives
EQSA [13]		✓					✓	• Reduce energy consumption • Maintain high service availability
CASSARAM [10]		✓		✓				• Accuracy • Reliability • Battery life • Location
Dynamic PSO model [9]	✓				✓			• Response time • Cost • Availability • Reliability

(*continued on next page*)

Table 3 (*continued*)

Proposed Solution	QoS Layer			Algorithm Type				QoS Parameters
	App	Sen	N·W	Heu	Meta-Heu	Hyper-Heu	Non-Heu	
Fuzzy logic and cultural GA [56]	✓					✓		• Price • Response time • Availability • Throughput • Execution rate
Hybrid ANN-PSO [57]	✓					✓		• Response time • Availability • Prices
Fuzzy C-means and NCF [58]	✓		✓				✓	• Response time • Throughput

LRI [59]	✓			✓				• Subjective:
								  • Privacy
								  • Reliability
								  • Availability
								  • Response Time
								• Objective:
								  • Availability
								  • Response time
								  • Speed

AGEGA [60]	✓		✓		✓			• Balancing
								  • QoS parameters
								  • Connectivity constraints

QoS-IoT [61]			✓				✓	• Sybil attack detection • Fairness • Throughput • Buffer utilization

Likert-Improved-PSO [62]	✓				✓			• BQG:
								  • Reputation
								  • Execution price
								• SQG
								  • Reliability
								  • Availability
								  • Response time

AHP- TOPSIS [63]	✓	✓	✓	✓				• Temperature range • Resolution • Accuracy • Delay • Jitter • Pricing • Availability • Throughput • Response time

Abbreviations.
APP: application layer; *Sen*: sensor layer; *N·W*: network layer; *Heu*: heuristic; *Meta-Heu*: meta-heuristic; *Hyper-Heu*: Hyper-Heuristic; *Non-Heu*: non-heuristic.

and GWMIN2) [15]. The Results were improved [15] and the total communication energy was reduced by 10% compared with other methods [21]. They also implemented MWIS in a multi-hop network [14] and the total communication energy was reduced by more than 10% [15]. They planned to develop heuristic algorithms and test them with Adriano-based devices, as well as studying more complex applications by checking the automatic configuration module to support more users interacting with IoT systems.

Shukla et al. [55] presented CBSSMS to link IoT services with appropriate sensors. They conducted comparisons with existing collocation distance algorithms based on the random mapping of services on any IoT device, where they tested linear, random, and star FBP networks. The Results showed that the CBSSMS algorithm reduced the latency and energy consumption between devices compared with the collocation distance algorithms [20].

Elhosenya et al. [22] conducted a comparative study based on the execution time, system efficiency, and data processing speed. They evaluated the effectiveness of their model against GA, PSO, and PPSO. The Results showed that the proposed model improved the total implementation time by 50%. Moreover, the efficiency of the system at real-time data recovery improved significantly by 5.2%.

Alsaryrah et al. [18], evaluated the Bi-SPO algorithm against QoSC, which only considers the QoS, and EPC, which only considers the energy profile. The Results showed that Bi-SPO achieved the ideal balance between the traditional QoS level and energy consumed, and it performed better than the other algorithms.

Lin et al. evaluated their MOEA/D approach based on a sensor selection problem [39]. They showed that increasing the problem size led to an increase in the energy consumption, energy balancing, energy harvesting services, and pollution level, but it did not affect the QoS. They generated their data set.

Khanouche et al. [13] assessed the performance of EQSA by simulating the A2NEts European project scenario, which involves monitoring and smart metering for buildings. They synthetically generated data sets based on QoS factors and realistic energy models to specify the energy profiles of IoT services. The simulation Results demonstrated the efficient performance of EQSA in terms of the energy efficiency, selection time, composition lifetime, and optimality of the solution.

Li et al. [9] simulated their PSO method and showed that it was more efficient than using GAs. PSO optimized different fitness parameters and maximized the availability or reliability while maintaining a low cost and response time. They considered a real-world scenario involving the transport of furniture among countries by combining five Web services related to shipping cargo services. They applied their method to a previously reported data set (Mao data set) of QoS values. The feasibility of applying PSO was confirmed by implementing the simulation program in

Java. They recommended further research into logistics and Web service selection, improving the efficiency of SSA based on PSO, comparing PSO with other algorithms, and developing methods to confirm the consistency of QoS between service consumers and service providers.

Yuan et al. [56] evaluated the performance of a fuzzy logic technique and cultural GA by comparing it with a QoS constraints decomposition (QCD)approach based on cultural GA and an integer programming (IP)-based approach. The experimental Results showed that using the fuzzy logic technique and cultural GA was appropriate in terms of the adaptability and scalability to the environment, as well as satisfying the user preferences and increasing the number of candidate services. They used the Quality Web Service (QWS) data set containing 2508 real Web services with 10 QoS attributes factors [64]. Also they randomly generated other data set according to QWS (RQWS) using the Eclipse programming tool. They recommended increasing the number of fuzzy sets and formulating more appropriate fuzzy rules, before applying their approach in a distributed environment where a group of distributed QoS registries maintain the QoS values.

The hybrid ANN-PSO algorithm obtained better fitness values compared with PSO, GA, and PSOGA [57]. They evaluated their method based on simulations using the C# language as an integrated development environment (IDE) and the PAT model checker was employed to prove the correctness of the proposed algorithm. They employed QWS data set containing 2500 Web services. They recommended using deep learning methods to avoid the space explosion problem in the SC model.

Satisfactory experiments were conducted based on real-world WS-Dream data sets by Gao et al. [58]. The prediction performance was evaluated using the root mean squared error and mean absolute error. The experimental Results verified the effectiveness of the proposed NCF and context-aware NCF (CNCF) frameworks compared with well-known QoS prediction methods comprising user-based PCC (UPCC), item-based PCC (IPCC), web service recommender (WSRec), and location-based factorization machine (LBFM). They recommended implementing work-based models in the QoS prediction task, such as a recurrent neural network and convolutional neural network, and studying the time factor during QoS prediction.

The dynamic LRI model was compared with another based on user feedback [59], and the Results showed that the LRI model improved the effectiveness in a real-time scenario because it considers the similarity between users, although the time consumption was higher. The data set and scenario were generated in the study. For future research, they recommended applying a user-centric service management system based on the user's preferences in the IoT environment.

Jatoth et al. [60] compared the performance of AGEGA with other methods based on GA, i.e., GA, orthogonal GA (OGA), adaptive genetic programming (AGP), and transactional GA (TGA). The experimental Results showed that AGEGA obtained better fitness values within a lower execution time. They used QWS as the data set of QoS parameters and randomly generated some of the QoS parameters and their corresponding values. They recommended considering multiple service connectivity constraints and multiple QoS parameters in future research, as well as developing an efficient approach for various parallel data processing platforms.

Khan et al. [61] simulated QoS-IoT and compared FIFO, round-robin (RR), and cross-layer scheduling based on the utilization of the CW via adaptation using the network simulator NS-2. They simulated IoT-based networks that covered a city of 100×100 km^2 using a system model produced with the proposed approach. The total area was divided into smaller IoT-based networks, where each network comprised Sybil, mobile, static, and high power nodes. The simulation Results showed that QoS-IoT was resilient against the Sybil attack, as well as improving network performance with a large amount of data. They recommended studying the effect of Sybil node detection-aware QoS on the Internet of Vehicles and flying ad hoc networks.

Abu-Safe et al. [62] simulated their Likert-Improved-PSO model and evaluated its performance based on comparison with the original PSO and Improved-PSO. The proposed model had a lower execution time and it obtained better fitness value. For future research, they recommended testing more QoS parameters and combining with more than one meta-heuristic algorithm with respect to the end user feedback.

Singh et al. [63] applied their AHP–TOPSIS framework and existing AHP–AHP framework to a health care case study and compared the Results based on the execution time for the selection value. AHP–TOPSIS required a lower execution time than AHP–AHP to obtain the same selection value. They assembled real data sets from three different providers but did not describe them. The robustness of the proposed framework was measured but the sensitivity toward changes in the user or decision maker was not analyzed. They recommended extending their method to deal with fuzziness in human decision making.

6.2. Using prototypes to evaluate SSAs

Sun et al. [6] evaluated three heuristic algorithms (ACO, GA, and PSO) by building prototypes to calculate the fitness, minimum, and difference in the residual energy for smart devices. The Results showed that PSO performed better than GA and ACO at the optimization problem.

Mejri et al. [8] developed a parallel implementation of the ANN model and TOPSIS model to evaluate the scalability of SSA in the IoT. They used the mean absolute error to measure the quality of the prediction model. As the number of services increased, the mean absolute error decreased and the accuracy increased, but there was no significant increase in the execution time. They recommended using an evolutionary technique and pruning methods. A limitation was that the proposed approach was applied to a training set built during search steps that did not meet all Internet requirements. In addition, they only considered the response time and reliability as QoS factors in their study.

Gao et al. [51] proposed QoS-AS for CES and GA, which they compared with a brute-force enumeration algorithm in terms of the execution time and optimization degree, where the proposed algorithm improved the optimized Results from 79% to 97%. The performance of CASSARAM was also evaluated based on the change in the storage

Table 4
Categorization of state-of-the-art algorithms according to evaluation methods and performance measurement based on prototypes.

Reference	Software used	Performance measurement	Data set	Results
[6]	Java program, with Intel i7-6700 CPU, 8-GB of memory, and 64-bit Windows 7	Compared with ACO, GA, and PSO	Generated their own data set	PSO performed better than GA and ACO
[8]	Java and mean absolute error as the assessment metric	Evaluated ANN model and TOPSIS model	Different data sets generated randomly [0,1]	Increased the accuracy, but with no significant increase in execution time
[10]	Java on a computer with an Intel(R) Core i5-2557 M, 1.70 GHz CPU, and 4 GB RAM	Comparison based on execution time, memory required, and accuracy	Linked Sensor Middleware project	Reduced processing time and minimized storage requirements
[51]	Java using MacBook Pro with 2.53 GHz duo core CPU and 4 GB 1067 MHz memory	Compared with a brute-force enumeration algorithm	Built their own data set	Improved execution time and degree of optimization, and optimized Results from 79% to 97%

Table 5
Categorization of state-of-the-art algorithms according to evaluation methods and performance measurement based on simulations.

Reference	Software used	Performance measurement	Data set	Results
[49]	Java on Hadoop platform using Intel Core 2 Duo CPU and 1 GB of RAM, running CentOS v6.0	Compared DijkstraKNN, BFSKNN, and PruningBFSKNN algorithms	• Epinions Social network • LiveJournal social network	BFSKNN and PruningBFSKNN algorithms obtained more stable execution times
[16]	Java using Intel Core, Window 32-bit system	Compared ACO with load balancing clustering	Not stated	Reducing the time and energy required, and more efficient for large scale IoT
[53,54]	Java on a desktop computer with Intel Core i5-3570 dual CPU, 3.40 GHz, running on Windows 8 (64-bit)	Compared PSS and PSM with skyline-based algorithm OPA	Generated random data based on Climatography of the United States Number 81 series (CLIM8144) data set	Efficient with a large number of candidate physical services and improved the selection performance; More scalable and reduced the execution time [54]
[18]	Java under 64-bit Windows 7 OS, on Intel Core i5- 8 GB RAM	Compared QoS-C and EPC	Synthetically generated data set based on QoS	Performed better than algorithms that only considered QoS-C or EPC
[13]	Used JVM, JRE 1.6, for Windows 64-bit running on Intel Core i7-4712HQ CPU random memory	_	Synthetically generated data set based on QoS and EP	Improved energy efficiency, selection time, composition lifetime, and optimality of the solution
[9]	Simulation program in Java	Compared with GA	Mao Data set	Increased the availability and reliability, and maintained low cost and response time
[11]	MATLAB 7.0 + Intel Core2 Duo 2.10 GHz CPU	Compared CEA with CPSO and IDIPSO	Data generated from real scenarios	High performance in terms of search convergence and stability
[12]	MATLAB R2013b for Windows 7 on 2.50-GHz PC with 4-GB RAM	Compared HABC with GA, PSO, and basic ABC	Randomly generated data set among [0.7, 0.95]	High performance in terms of search stability within acceptable time
[17]	MATLAB R2015a	Compared GSA and ABC with PSO, GA, ABC, and GSO	Real-time data acquisition read through Xively IoT API	Improved cluster head performance
[55]	MATLAB R2017b on Core(TM) i3-5005U CPU @ 2.00 GHz (4 CPUs) and 4 GB RAM	Compared CBSSMS with distance algorithms at randomly mapping services on devices	Generated their own data set	Reduced latency and energy balance between devices
[22]	MATLAB + CloudSim package	Compared hyper-model with GA, PSO, PPSO	Generated their own data set	Enhanced implementation time and system's efficiency in real time
[52]	Not stated	Compared FQSA with GAFLSS, APUP, and FLP	Random QoS data sets generated from [0,1]	Improved selection performance and user satisfaction level
[19–21]	Not stated	Compare the performance of three models	Generated their own data set by randomly generating services data set	Reduced communication energy consumption and increased system lifetime
[14,15]	Not stated	Compared MWIS with MWL, GWMAX, GWMIN, and GWMIN2	Generated their own data set	Reduced communication energy and increased system lifetime
[39]	C++ programming language run on a PC with Intel Core i7-6700 CPU and 8 GB memory	Compared with sensor selection algorithms	Generated their won data set	Ran more efficiently
[7]	Python with NetworkX, SciPy, and NumPy libraries	Compared HOLA system with Vanilla System	Not stated	Reduced energy consumption by IoT sensors by reducing the internal communication in IoT devices, as well as reducing time required
[56]	Microsoft Visual C. Net on PC with an Intel Core i5 (1.6 GHz) CPU and 4 GB RAM	Compared fuzzy logic technique and cultural GA with QCD and WS-IP	QWS and RQWS generated randomly using Eclipse programming tool	Reduced runtime and approximation ratio
[57]	C# language used as an integrated development environment and PAT model checker	Compared ANN-PSO with PSO, GA, and PSOGA algorithms	QWS	Improve execution time and reachability rate
[58]	-	Compared NCF and CNCF against QoS prediction methods (UPCC, IPCC, WSRec, and LBFM)	WS-Dream	Superior prediction performance demonstrated
[59]	Mac-OS 10.14.3 within an Intel i5-7500U 2.30 GHz CPU, 8 GB RAM	Compared LRI with another method that considered user feedback	Generated their own data set	Verified the similarity between users in real time
[60]	Java and R language on Intel (R) Core (TM) i5 2.60-GHz processor and 8 GB of memory, running Windows 8.1	Compared AGEGA with GA, OGA, AGP, and TGA	QWS plus some random data	Obtained better fitness values in lower execution time
[61]	Network simulator NS-2	Compared QoS-IoT against FIFO, RR, and cross-layer based utilization of CW for scheduling	Designed their scenario	Improved network performance with a large amount of data
[62]	MATLAB on Windows 10, 2.90 GHz processor, and 8 GB RAM	Compared Likert-Improved-PSO with original PSO and Improved-PSO	Generated a random data set	Better fitness values and lower execution time
[63]	-	Compared with AHP–AHP	Health care case study using real data collected from various sources available online	Lower execution time and robust to changes in user or decision maker

requirements according to the sensor data descriptions, the requirements for sensor selection and indexing, the memory required to select sensors, and the change in the accuracy rate. They evaluated the processing time and memory requirements based on sensor selection and relational expressions during the semantic querying phase. The data sets employed were from the Linked Sensor Middleware project. They showed that CASSARAM could reduce the processing time and minimize the storage requirements. In the future, they plan to merge their algorithm with leading IoT middleware solutions such as SenseMA and Open-IoT to improve the automated sensor selection functionality in the IoT environment. They also recommended enhancing the efficiency of CASSARAM to integrate automated machine learning techniques using cluster-based sensor search and heuristic algorithms.

6.3. Using simulations and prototypes to evaluate SSAs

Dhondge et al. [7] validated HOLA based on simulation studies and designed a HOLA IoT sensor prototype with Adriano. In practical experiments, they measured the energy consumption of the HOLA IoT sensors in different operational scenarios and communication settings. In particular, they compared the energy consumption of HOLA and the Vanilla System, and showed that HOLA could reduce the energy consumed in IoT sensors by reducing the internal communication in the IoT device. The time consumption with HOLA was better compared with the Vanilla System. They recommended detecting the maximal energy efficiency that satisfies the SLA agreement and evaluating HOLA using different smartphone densities in the future.

6.4. Analysis of evaluations and performance measurements for SSAs

The analyses of selected state-of-the-art solutions evaluations used prototypes, and Simulationd are categorized, as shown in Tables 4 and 5 respectively. They categorized based on the software used, performance measurement, applied data set, and obtained results.

7. Results and future research directions

SSAs are essential for the IoT environment in order to satisfy the preferences of end users by selecting the required services based on QoS factors. In the following, we discuss the Results, possible future research directions, and limitations of the state-of-the-art solutions, thereby highlighting the basic requirements for a robust SSA structure.

Based on our proposed classification, the structure of a SSA can be divided into the design process to determine the appropriate environment for building the SSA, the implementation stage involving the definition of the structure required to implement the SSA, and the evaluation step to measure the performance of the SSA.

In order to design an appropriate SSA structure for the IoT system, the following specific features should be considered. The time allocated to the selection process is called the process time phase. The design time is rarely static before a user requires a service [13]. Thus, most of the state-of-the-art methods are dynamic during the runtime [5,7,11]. In terms of workflow management, IoT is a large-scale environment that requires complex management or orchestration [7,14,17], where the IoT devices and network communication structure have specific properties. Thus, most studies preferred to select a choreography workflow [6,50,51]. Moreover, to the best of our knowledge, no algorithms involved single-objective optimization because the QoS factors are related to others, and thus MOPs [6,48] and MaOPs [5,11] required optimization.

In order to implement and build an appropriate SSA for the IoT system, most of the state-of-the-art methods focused on selecting the QoS factors that need to be optimized or satisfied. We classified the QoS factors based on the IoT architecture proposed by Li et al. [40] in the application layer, sensor layer, and network layer. In future research, it would be useful to focus more on search in the network layer because its properties affect the selection of the required services. Moreover, the traditional QoS factors considered in most studies comprised the optimization time, cost, availability, reliability, and energy consumption, as shown in Fig. 7.

In addition, most of the solutions proposed for SSA used search optimization algorithms, particularly meta-heuristic algorithms [16,39,50] based on evolutionary algorithms (e.g., GA, PSO, and ACO), as well as heuristic algorithms [7,19–21,49]. To the best of our knowledge, very few studies have implemented hyper-heuristic algorithms for making selections in the IoT system, as shown in Ref. [22]. The remaining state-of-the-art methods employed other types of algorithms such as Pareto optimality [13] and the PSS method [53,54].

Thus, researchers have tended to produced improved algorithm by combining more than one to obtain more efficient solutions. Thus, new solutions can be obtained for selection problems by considering other methods such as fuzzy logic.

A new trend is the use of prediction in the selection process to enhance the end user preferences [58]. We consider that using a recommendation system that predicts the behavior and preferences of end users could result in a more effective selection process.

In order to evaluate and measure the performance of SSAs, studies have generally compared the proposed algorithms and models with others, as shown in Tables 4 and 5. Most studies conducted simulations but some involved building prototypes. The most commonly used language is Java, followed by MATLAB and other programming languages, as shown in Fig. 8. In most studies, data sets were generated for the experimental evaluations [12,19–21,52,55], although some used existing data sets that were not constructed specifically for IoT environments [9,49,53,54]. Thus, there is need to provide an appropriate data set that satisfies the QoS requirements for services in IoT environments.

The Results shown in Figs. 6–8 were extracted by analyzing reports of state-of-the-art methods. Fig. 6 was derived by analyzing the implementations of solutions for SSAs, as shown in Table 3. Fig. 8 was extracted by analyzing the evaluations and performance measurements for SSAs, as shown in Tables 4 and 5. Figs. 6–8 should help researchers to extract useful information to guide their research into SSAs in IoT environments.

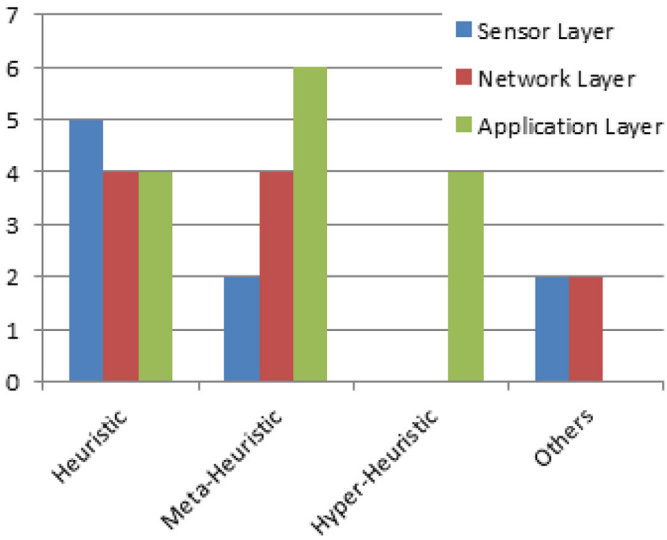

Fig. 6. Implementations algorithms in different IoT layers.

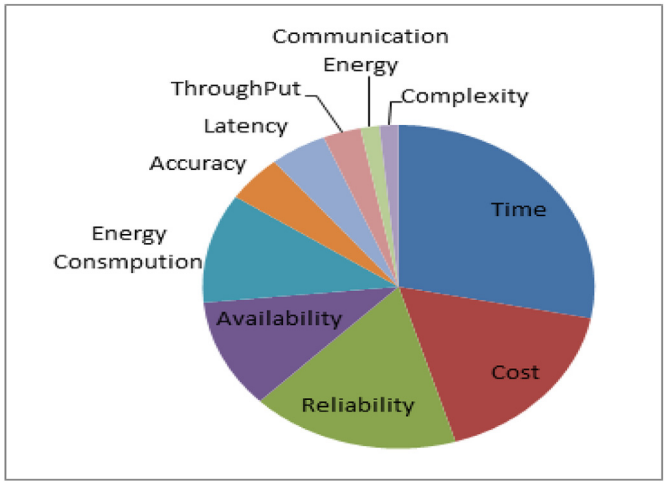

Fig. 7. Common QoS factors considered.

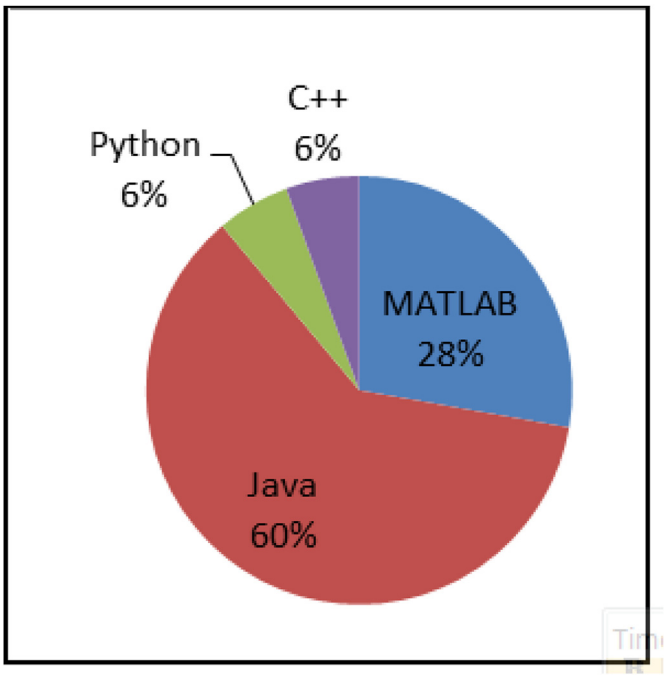

Fig. 8. Programming languages used for performance evaluations.

8. Conclusion

The development of the IoT environment has led to increases in the number of sensors that provide different services to end users. These services have similar functional properties but different non-functional properties (QoS factors). This problem requires the selection of algorithms that can identify the optimal service to meet the requirements of the end user. Many studies have investigated SSAs based on Web services but we focused on SSA solutions proposed for the IoT environment in this review. We presented the fundamental design structures, behaviors, and optimization objectives of SSA in IoT under QoS constraints. The state-of-the-art algorithms were analyzed based on the implementation and QoS layer in the IoT architecture (sensor layer, network layer, and application layer). Moreover, performance evaluations were analyzed to determine the most commonly used methods and data sets for assessing the performance of algorithms. Finally, we identified possible future research directions and deficiencies to help the research community develop appropriate SSAs for the IoT system and to optimize the critical QoS factors required by end users.

9. Research limitations and future recommendations

In the future, we recommend reviewing more SSA solutions by increasing the research scope. In this review, we focused mainly on heuristic, meta-heuristic, and hyper-heuristic algorithms, and thus we recommend studying other types of optimization algorithms. In addition, we recommend the construction of an appropriate QoS data set for the IoT environment.

Declaration of competing interests

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

Abbreviation, Definition

ABC	Artificial Bee Colony
ACO	Ant Colony Optimization
AGP	Adaptive Genetic Programming
AHP	Analytic Hierarchy Process
ANN	Artificial Neural Network
ANN-PSO	Artificial Neural Network-Based- Particle Swarm Optimization
Bi-SPO	Bi-Objective Shortest Path Optimization
CASSARAM	Context-Aware Sensor Search, Selection, And Ranking Model
CBSSMS	Collocation Based Sensor-Service Mapping Strategy
CCOA	Chaos Control Optimal Algorithm
CCSOS	Composited Cloud Manufacturing Service Optimal Selection
CEA	Cooperative Evolution Algorithm
CNCF	Context-Aware Neural Collaborative Filtering
DURD	Discrete Uniform Rank Distribution
DUSRD	Discrete Uniform Service Rank Distribution
EB	Energy Balancing
FBP	Flow-Based Program
FIFO	First In First Out
FQSA	Flexible QoS-Based Service Selection Algorithm
GA	Genetic Algorithm
GSA	Gravitational Search Algorithm
HABC	Hybrid Artificial Bee Colony
HOLA	Heuristic And Opportunistic Link Selection Algorithm
IIoT	Industrial IoT
ILP	Integer Linear Programming
IoT	Internet of Things
IPCC	Item-Based PCC
LBFM	Location-Based Factorization Machine
LRI	Linear Reward Inaction
MaOP	Many-Objective Optimization Problem
MOEA/D	Many-objective Evolutionary Algorithm Decomposition
MOPs	Multi-objective optimization problem
MWIS	Maximum Weighted Independent Set
MWL	Maximum Weighted Link
NCF	Neural Collaborative Filtering
OGA	Orthogonal Genetic Algorithm
OPA	One-Pass Algorithm
PSO	Particle Swarm Optimization
PSS	Physical Service Selection
QCD	QOS Constraints Decomposition
QoS	Quality of Service
QoS-AS for CES	QoS Aggregation Schema for Complex Event Service
RQWS	Randomly Generated QWS
RR	Round-Robin Scheduling
SC	Service Composition
SOA	Service-Oriented Architecture
SOAP	Simple Object Access Protocol
SSA	Service Selection Algorithm
TGA	Transactional Genetic Algorithm
TOPSIS	Technique for Order of Preference by Similarity to Ideal Solution
UDDI	Universal Description, Discovery, and Integration
UPCC	User-Based PCC
W3C	World Wide Web Consortium
WS	Web Services
WSDL	Web Services Description Language
WSREC	Web Service Recommender

References

[1] Nicolescu R, Huth M, Radanliev P, De Roure D. State of the art in IoT - beyond economic value. 2018.
[2] Evans D. The Internet of things - how the next evolution of the Internet is changing everything. 2011.
[3] Maaradji A. End-user service composition from a social networks analysis perspective. 2011. theses.fr.

[4] Xia Y, Chen P, Bao L, Wang M. 2011 IEEE international conference on web services. In: A QoS-aware web service selection algorithm based on clustering; 2011. p. 8.
[5] Huang B, Li C, Tao F. A chaos control optimal algorithm for QoS-based service composition selection in cloud manufacturing system. Enterprise Inf Syst 2014;8(4):445–63.
[6] Sun M, Shi Z, Chen S, Zhou Z, Duan Y. Energy-efficient composition of configurable Internet of things services. *IEEE Access*; 2017. p. 1–14.
[7] Dhondge K, Shorey R, Tew J. HOLA: heuristic and opportunistic link selection algorithm for energy efficiency in Industrial Internet of Things (IIoT) systems. In: 2016 8th international conference on communication systems and networks. COMSNETS 2016; 2016. p. 1–6.
[8] Mejri M, Ben Azzouna N. Scalable and self-adaptive service selection method for the internet of things. Int. J. ofComputer Appl. 2017;167(10):43–9.
[9] Li W, Zhong Y, Wang X, Cao Y. Resource virtualization and service selectionin cloud logistics. J Netw Comput Appl 2013;(1–9).
[10] Perera C, Zaslavsky A, Liu CH, Compton M, Christen P, Georgakopoulos D. Sensor search techniques for sensing as a service architecture for the internet of things. IEEE Sensor J 2014;14(2):406–20.
[11] Liu J, et al. A cooperative evolution for QoS-driven IoT service composition. Autom. – J. Control. Meas. Electron. Comput. Commun. 2013;54(4):438–47.
[12] Zhou J, Yao X. A hybrid artificial bee colony algorithm for optimal selection of QoS-based cloud manufacturing service composition. The International Journal of Advanced Manufacturing Technology 2017;88(9-12):3371–87.
[13] Khanouche ME, Amirat Y, Chibani A, Kerkar M, Yachir A. Energy-centered and QoS-aware services selection for internet of things. IEEE Transactions on Automation Science and Engineering 2016;13(3):1256–69.
[14] Huang Z, Lin K, Yu S, Hsu JY. Co-locating services in IoT systems to minimize the communication energy cost. Journal of Innovation in Digital Ecosystems 2014;1(1-2):47–57.
[15] Huang Z, Lin KJ, Yu SY, Hsu JYJ. Building energy efficient internet of things by Co-locating services to minimize communication. In: Medes 2014 - 6th international conference on management of emergent digital EcoSystems; 2014. p. 101–8. Proceedings.
[16] Abinaya S, Akshaya G, Dhivya J. Minimizing energy consumption using internet of things. *Int. J. Recent Innov. Trends Comput. Commun.*, no. March 2017:67–70.
[17] Reddy MPK, Babu MR. Energy efficient cluster head selection for internet of things energy efficient cluster head selection for internet of things. New Rev Inf Netw 2017;22(1):54–70.
[18] Alsaryrah O, Mashal I, Chung T. Bi-objective optimization for energy aware internet of things service composition. IEEE Access 2018;6:26809–19.
[19] Yu SY, Shih CS, Hsu JYJ, Huang Z, Lin KJ. QoS oriented sensor selection in IoT system. In: Proceedings - 2014 IEEE international Conference on Internet of things, iThings 2014, 2014 IEEE international Conference on green Computing and communications, GreenCom 2014 and 2014 IEEE international Conference on cyber-physical-social computing, CPS 20. iThings; 2014. p. 201–6.
[20] Huang Z, Lin KJ, Li C, Zhou S. Communication energy aware sensor selection in IoT systems. In: Proceedings - 2014 IEEE international Conference on Internet of things, iThings 2014, 2014 IEEE international Conference on green Computing and communications, GreenCom 2014 and 2014 IEEE international Conference on cyber-physical-social computing, CPS 20. iThings; 2014. p. 235–42.
[21] Huang Z, Lin KJ, Han L. An energy sentient methodology for sensor mapping and selection in IoT systems. IEEE International Symposium on Industrial Electronics 2014:1436–41.
[22] Elhoseny M, Abdelaziz A, Salama AS, Riad AM, Muhammad K, Sangaiah AK. "A hybrid model of internet of things and cloud computing to manage big data in health services applications. Future generation computer systems. 2018;86:1383–94.
[23] Li Y, Huang Y, Zhang M, Rajabion L. Service selection mechanisms in the Internet of Things (IoT): a systematic and comprehensive study. Cluster Comput 2020;23(2):1163–83.
[24] Manqele L, Dlodlo M, Coetzee L, Sibiya G. A survey for service selection approaches in dynamic environments. In: 2017 IEEE AFRICON: Science, Technology and Innovation for Africa. AFRICON 2017; 2017. p. 1049–54.
[25] Bouzary H, Frank Chen F. Service optimal selection and composition in cloud manufacturing: a comprehensive survey. Int J Adv Manuf Technol 2018;97(1–4):795–808.
[26] Yu HQ, Reiff-Marganiec S. Non-functional property based service selection: a survey and classification of approaches. CEUR Workshop Proc. January, 2008;411.
[27] Hamzei M, Navimipour NJ. Towards efficient service composition techniques in the internet of things. IEEE Internet Things J. Towar. 2018;4662. c.
[28] Aoudia I, Benharzallah S, Kahloul L, Kazar O. A comparative analysis of IoT service composition approaches. In: The international arab conference on information Technology yassmine hammamet; 2017. p. 1–7. Tunisia.
[29] Asghari P, Rahmani AM, Haj H, Javadi S. Service composition approaches in IoT: a systematic review. J. Netw. Comput. Appl. 2018;120:61–77.
[30] Kahloul L, Benharzallah S, Aoudia I. Service composition approaches for Internet of Things: a review. Int J Commun Network Distr Syst 2019;23(1):1.
[31] Dongre Y, Ingle R. An Investigation of QoS Criteria for Optimal Services Selection in Composition. In: 2nd Int. Conf. Innov. Mech. Ind. Appl. ICIMIA 2020 - Conf. Proc., no. Icimia; 2020. p. 705–10.
[32] Hugo Haas W, (until J M. Allen Brown, "web services glossary," W3C working group [Online]. Available: https://www.w3.org/TR/ws-gloss/; 2002. 20-Jan-2018.
[33] Sheng QZ, Qiao X, Vasilakos AV, Szabo C, Bourne S, Xu X. "Web services composition: a decade's overview. Inf Sci (Ny) 2014;280:218–38.
[34] Elfirdoussi S, Jarir Z. An integrated approach towards service composition life cycle: a transportation process case study. May 2018 J. Ind. Inf. Integr. 2019;15.138–146.
[35] Elqortobi M, Bentahar J, Dssouli R. "Framework for dynamic web services composition guided by live testing. In: Lecture notes of the institute for computer sciences, social-informatics and telecommunications engineering, vol. 206. LNICST; 2018. p. 129–39. 5.
[36] Lemos AL, Daniel F, Benatallah B. Web service composition: a survey of techniques and tools. ACM Comput Surv 2015;48(3):1–41.
[37] Caramia P, Massimiliano Dell'Olmo. Multi-objective optimization. In: Multi-objective management in freight logistics; 2008. p. 11–37.
[38] Marler RT, Arora JS. Survey of multi-objective optimization methods for engineering. Struct Multidiscip Optim 2004;26(6):369–95.
[39] Lin CC, Deng DJ, Lu ALY. Many-objective sensor selection in IoT systems. IEEE Wirel. Commun. 2017;24(3):40–7.
[40] Li L, Li S, Zhao S. QoS-Aware scheduling of services-oriented internet of things. IEEE Trans. Ind. Informatics 2014;10(2):1497–507.
[41] Marjanović M, Skorin-kapov L. Quality of service (QoS) for IoT services. 2014.
[42] Jiménez-Sáez NMH. Multi-criteria decision-making, evolution and characteristics. In: International series in operations research & management science. Cham: Springer; 2019. p. 3–13.
[43] Bianchi L, Dorigo M, Gambardella LM, Gutjahr WJ. A survey on metaheuristics for stochastic combinatorial optimization. Nat Comput 2009;8(2):239–87.
[44] R P, Edmund Burke SS, Hart Emma, Kendall Graham, Jim Newall. Hyper-heuristics: an emerging direction in modern search Technology. *ResearchGate*; 2003. no. January, pp. 0–23.
[45] Gendreau M, Potvin J-Y. A classification of hyper-heuristic approaches edmund, vol. 272. *Springer-Verlag*; 2010. July 2014.
[46] F. Wikipedia, "Basic prototype categories."
[47] 06-Jun-2020 Prototypes vs. simulations. *Applied Abstractions*; 2005 [Online]. Available: https://appliedabstractions.com/2005/04/12/prototypes-vs-simulations/.
[48] Na J, Lin K, Huang Z, Zhou S. An evolutionary game approach on IoT service selection for balancing device energy consumption. In: IEEE 12th international conference on e-business engineering; 2015. p. 331–8.
[49] Yin X, Yang J. Shortest paths based web service selection in internet of things. J. Sensors 2014;2014.
[50] Anas M, Khatab S, Badr A. Poster: HeuristicIoT: a framework for augmenting heuristic search algorithms by internet-of-things data. In: MobiSys 2016 companion - companion publication of the 14th annual international conference on mobile systems, applications, and services, vol. 12; 2016. p. 4. 1.
[51] Gao F, Curry E, Ali MI, Bhiri S, Mileo A. QoS-aware complex event service composition and optimization using genetic algorithms 2014;8831:386–93.
[52] Nwe NWE, Win H, Jian-min BAO, Gang CUI. Flexible user-centric service selection algorithm for Internet of Things services. J China Univ Posts Telecommun 2014;21:64–70. July.
[53] Jin X, Chun S, Jung J, Lee K. IoT service selection based on physical service model and absolute dominance relationship. In: IEEE international conference on service-oriented computing and application; 2014. p. 65–72.
[54] Jin X, Chun S, Jung J, Lee K. A fast and scalable approach for IoT service selection based on a physical service model. Information Systems Frontiers 2016;19(6):1357–72.
[55] Shukla J, Maiti P, Sahoo B. Low latency and energy efficient sensor selection for IoT services. In: International Conference on Technologies for Smart City Energy Security and Power: Smart Solutions for Smart Cities, ICSESP 2018 - Proceedings; 2018. p. 1–5. 2018-Janua.
[56] Yuan Y, Zhang W, Zhang X, Zhai H. Dynamic service selection based on adaptive global QoS constraints decomposition. Symmetry (Basel). 2019;11(3).
[57] Hosseinzadeh M, et al. A hybrid service selection and composition model for cloud-edge computing in the internet of things. IEEE Access 2020;8:85939–49.
[58] Gao H, Xu Y, Yin Y, Zhang W, Li R, Wang X. Context-aware QoS prediction with neural collaborative filtering for internet-of-things services. IEEE Internet Things J. 2020;7(5):4532–42.
[59] Quan H, Takahashi R, Yoshiaki F. Dynamic service selection based on user feedback in the IoT environment. In: CITS 2019 - proceeding of the 2019 international conference on computer. Information and Telecommunication Systems; 2019. p. 1–5.
[60] Jatoth C, Gangadharan GR, Buyya R. Optimal fitness aware cloud service composition using an adaptive genotypes evolution based genetic algorithm. Future Generat Comput Syst 2019;94:185–98.
[61] Khan F, et al. A quality of service-aware secured communication scheme for internet of things-based networks. Sensors 2019;19(19):1–18.
[62] Abu-safe AN, Elrofai SE. QOS – aware meta-heuristic services selection algorithm and Likert scale measurement for IOT environment. Int. J. Comput. Sci. Trends Technol. 2020;8(1):1–8.
[63] Singh M, Baranwal G, Tripathi AK. QoS-aware selection of IoT-based service. Arabian J Sci Eng 2020;20(2).
[64] Al-Masri's E. QWS-Datasets. *University of Guelph & University of Washington Tacoma*; 2007 [Online]. Available: https://qwsdata.github.io/.