QOS Aware Service Composition in IoT Using Heuristic Structure and Genetic Algorithm

Pavan Kumar V
Research Scholar, CSE, PES Institute of Technology and Management, Shivamogga, Karnataka, 577204, India
sadgurupavan@gmail.com
SudheerShetty
Associate Professor, ISE, Alva's Institute of Engineering & Technology, Moodubidire Karnataka, 574225, India
sudheershetty06@gmail.com
Janardhana D R
Assistant Professor, CSE, NitteMeenakshi Institute of Technology, Bengaluru Karnataka, 560064, India
janardhana.dr@nmit.ac.in
Manu A P
Professor, CSE, PES Institute of Technology and Management, Shivamogga, Karnataka, 577204, India
apmanu@gmail.com

Article Info
Page Number: 750 – 766
Publication Issue:
Vol. 71 No. 3 (2022)

Article History
Article Received: *12 January 2022*
Revised: *25 February 2022*
Accepted: *20 April 2022*
Publication: *09 June 2022*

Abstract
In the current Internet of Things (IoT) environment, the intelligent objects develop as IoT services, some having the same functionalities but different QoS attributes. The complexity of the user's requirements to fulfill a new service (i.e., composition service) is acquired from the collection of atomic services. On the other hand, fulfilling QoS constraints in composite service becomes an NP-hard problem. The proposed approach uses a Decision Tree and Genetic Algorithm (GA) for QoS aware IoT optimal service composition. The process has used two levels of QoS attributes to achieve the optimal service composition. First-level QoS attributes are applied to the Heuristic Structure (i.e., Decision tree) to eliminate unsuitable concrete services from the abstract service that reduce search space and composition time. Finally, second-level QoS is applied to GA to find the best optimal service composition. Moreover, the scalability of the proposed system is stable and more accurate compared with the regular optimal service composition.

Keywords: Service Composition, QoS, Optimization, Decision Tree, Genetic Algorithm, Internet of Things (IoT), Service Directed Graph.

1 Introduction

In current years, the passages in the Internet of Things (IoT) technology has improved a lot [1]. This technology enters all sides of human life (agriculture, supply chain, smart home, health, etc.) [2]. It is the most reasonable way to attach real and virtual worlds [3]. It allows the binding between information and the physical world using the Internet. It offers the physical world functionality as a Service to the information world [4]. The IoT services are used to observe the real-time and control specific objects in every life [5]. On the other hand, the complexity of the user's requirements to fulfill a new service is acquired from the collection of atomic services. Such services are called Composite Services [6].

Many intelligent objects develop as IoT services with the same functionalities, but different QoS attributes [7,8]. It creates a complex problem in fulfilling the user requirements. On the other hand, fulfilling QoS constraints in composite service become a challenge [9].

The service composition is categorized into two approaches considering QoS [10]. (1) Based on Local attributes applied in individual services to optimize composite service. However, this approach may not guarantee to fulfill the user's requirements at the composite level. (2) Based on Global attributes to optimize composite service. It is a noticeable way to perform all possible service compositions at this level and find the best Composition that maximizes the satisfaction of the user's requirements. However, this directs to an exponential complexity problem (NP-hard)[6].

This paper proposed an approach based on a Decision Tree and Genetic Algorithm that works for QoS-aware IoT Service composition to handle the above challenges. Furthermore, it uses global QoS optimization for service composition in IoT. This approach also uses local attributes at an individual level to discover the Optimal Composition and reduce search space and composition time.

This paper, organized as Section 2 discussed related work of service composition in an IoT environment considering QoS. In Section 3 discussed related knowledge and rules of QoS evaluation for the proposed approach. In Section 4 the proposed approach and algorithms discussed. In Section 5 discussed performance analysis, evaluation methodology, and comparative study with normal Genetic Approach and Proposed Approach. Finally concluded with Conclusion in Section 6.

2 Related Work

The IoT environment is broadcaste oriented, highly dynamic, scalable, and adapt to new types of things. However, service Composition is one of the most prevalent problems in the IoT environment.

To handle the IoT service Composition problem, various solutions have been proposed. Few solutions: "Bio-Inspired Meta-Heuristic" Genetic Algorithms (GA) [11-13], GA and "Particle Swarm Optimization (PSO)" [14], and "Cukoo-Inspired" methods [15].

To handle the user's QoS requests, the author of [13], has designed the IoT service composition problem as a "Multi-Criteria Goal Programming (MCGP)" instance and implemented a "Multi-population Genetic Algorithm (MGA)."

In [14], author created a comparison of "Genetic Algorithm" and "Particle Swarm Optimization" in IoT service composition. As a result, the author proves that both algorithms

solve the service composition in IoT domain. Moreover, the observed effects on the real-time dataset indicate that the solutions obtained using GA have better performance than the PSO.

In author of [12], presents a "Multi-Objective Meta-Heuristic Search" algorithm to give a solution for IoT service composition using the QoS problem. Here Optimal Composition is obtained based on the "Non-dominated Sorting Genetic Algorithm (NSGA-II)."

In author of [15], created an algorithm based on Cuckoo to locate available service composition. As a result, it fulfills the user's requirements in a "Multi-Cloud Environment (MCE)," and execution show less resource and time.

Most research focused on Service composition as the IoT environment is more dynamic or the QoS of the user's requirement needs to match in service composition. Unfortunately, few research articles are available that satisfy the scalability of service composition (number of services per task) based on QoS.

3 Related Knowledge and Rules of Proposed Framework

3.1 Service Dependency Graph

A Service Dependency Graph (SDG) represents the service composition relation with QoS. Here SDG define as G = (V, E). Where V is a set of a node of the graph, each node represents an Abstract Service (AS), and E is a set of directed edges; each edge represents a set of linkages between nodes and defines with a set of a parameter such as global constraints. Abstract service represents a collection of homogeneous services (Atomic or Concrete Services (CS)) with different QoS constraints.

$$AS_i = \{CS_{i1}, CS_{i2}, \ldots, CS_{ik}\} \quad (1)$$

$$CS_{ij} = \{QoS(CS_{ij}), Action(CS_{ij})\} \quad (2)$$

Here i^{th} vertex represented with (AS_i) and CS_{ij} represents j^{th} service of of i^{th} vertex and CS_{ij} represents functionality as $Action(CS_{ij})$ and QoS levels as $QoS(CS_{ij})$.

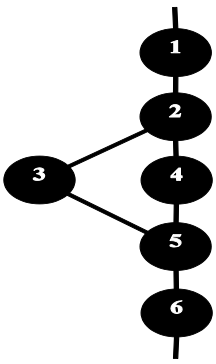

Fig 1.Sample Service Dependency Graph

An SDG is represented in figure 1, here a service composed with set of Abstract Services and composition possible in order $AS_1 \rightarrow AS_2 \rightarrow AS_3 \rightarrow AS_5 \rightarrow AS_6$or $AS_1 \rightarrow AS_2 \rightarrow AS_4 \rightarrow AS_5 \rightarrow AS_6$.

Abstract Service compositions are later converted into Concrete Service Compositions by allocating concrete service (CS_{ij}) to every node in the Composition. Considering the many services available in the service environment, and expected that many concrete services are found for every node. Therefore, it refers to these services are regarded as node's candidate

services. Considering user QoS requirements are formulated with several constraints (called global QoS) in Composition. These constraints match with evaluated QoS constraints carried by edges. Additional details related to QoS attributes are provided in Section 3.2.

3.2 QoS Composition Rules

The evaluation of QoS depends on possible service composition structure elements and how QoS is aggregated, and referred to as composition mode. It describes the evaluation formula related to QoS attributes based on composition mode.[16,17] Composition modes are classified into Five types, Sequence, AND, OR, Split, and Loop, as shown in figure 2.

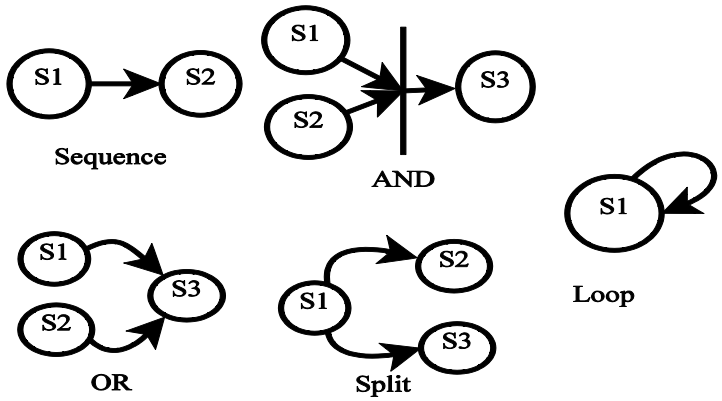

Fig 2. Types of Composition Modes

Node of the abstract service composition, depends on the QoS of the concrete service CS_ij. And denote with the help of vector as

$$QoS(CS_{ij}) = \{q_1^{ij}, q_2^{ij}, \dots q_n^{ij}\} \quad (3)$$

Here $'n'$ number of attributes of QoS and q_k^{ij} represents value of k^{th} QoS attribute $(1 \leq k \leq n)$.

The service composition QoS evaluation also depends on the QoS vector and takes account of composition mode. QoS is associated with AND and OR; it assumes the worst-case QoS values. For example, taking a response time has maximum value in the case of AND and OR. These values effects based on the type of QoS. QoS types usually are categorized into two types, Positive type and Negative type. In case of Positive QoSattribute,it has higher scale value and, defines the best QoS of services, such as reliability and reputation. The Negative type of QoS attribute, it has less scale value, defines the best QoS of service, such as Cost, Response Time. To determine the QoS evaluation formula concerning the type of QoS and Composition mode following formulas are used and is tabulated as shown in Table 1. In our case considered QoS attributes such as Cost (C), Reliability (R), Availability (A), Response time (Rt), Throughput (Th), and Reputation (Rp).

TABLE 1.QoS Composition Evaluation Rules

QoS Attribute	Composition Mode				
	Sequence	AND	OR	Split	Loop
Cost (C)	$\sum_{i=0}^{n} C_i$	$\sum_{i=0}^{n} C_i$	$max(C_i)$	$C + C_i$	$C * k$
Response Time (Rt)	$\sum_{i=0}^{n} Rt_i$	$\sum_{i=0}^{n} Rt_i$	$max(Rt_i)$	$Rt + Rt_i$	$Rt * k$
Availability (A)	$\prod_{i=0}^{n} A_i$	$\prod_{i=0}^{n} A_i$	$min(A_i)$	$A * A_i$	A^k
Reliability (R)	$\prod_{i=0}^{n} R_i$	$\prod_{i=0}^{n} R_i$	$min(R_i)$	$R * R_i$	R^k
Success rate (Sr)	$\prod_{i=0}^{n} Sr_i$	$\prod_{i=0}^{n} Sr_i$	$min(Sr_i)$	$Sr * Sr_i$	Sr^k
Throughput (Th)	$min(Th_i)$	$min(Th_i)$	$min(Th_i)$	$min(Th_i)$	Th
Reputation (Rp)	$avg(Rp_i)$	$avg(Rp_i)$	$avg(Rp_i)$	$avg(Rp_i)$	Rp

4 IoT Service Composition: Proposed Framework

The service composition framework works with a predefined service dependency graph and mapped abstract services with every node. The framework, as shown in figure 3, it presents three phases: Traversal Sequence Acquiring Phase, Hierarchy Service Filtering Phase, and Optimal Service Composition.

$$Req = \{LQoS, GQoS\} \quad (4)$$

In addition, user requests (Req) come in two parts; Local attributes and Global attributes. Local attributes are applied to filter the services in the second phase (Hierarchy Service Filter), and Global attributes are used to obtain optimal service composition in the final stage. User requests are represented as follows.

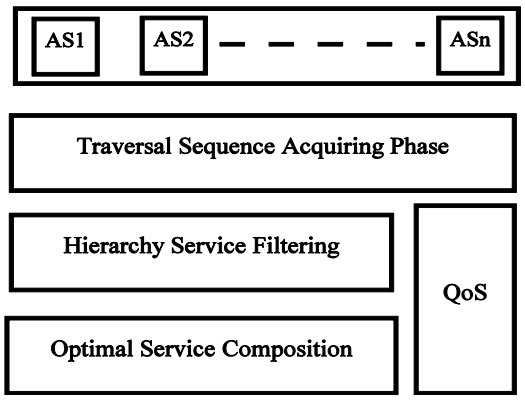

Fig 3. Service Composition Framework

4.1 Traversal Sequence Acquiring Phase

In the current phase, the framework try to find all possible compositions with the help of the Depth First Search (DFS)[18] algorithm, as shown in Algorithm 1.

4.2 Hierarchy Service Filtering Phase

In current phase, compares each node's service set with the user's requirement (called local QoS attributes) to filter out irrelevant services. Finally, it removes outside the service compositions related to the user's choice.

$$T_{sc} = \{AS_i, AS_j, \dots, AS_n\} \quad (5)$$

An abstract service composition (T_sc) has a sequence of abstract services (however, it refer as a task) represented in equation (5).

$$AS_i^{QoS} = \{aq_1^i, aq_2^i, \dots, aq_x^i\} \quad (6)$$

Where each abstract service has a collection of Concrete services as shown in equation (1) and evaluated with set of QoS attributes as shown in equation (6). Every Concrete service has a set of QoS attribute values to predict the performance of the service, represented in equation (7)

$$CS_{ij} = \{q_1^{ij}, q_2^{ij}, \dots, q_x^{ij}\} \quad (7)$$

Algorithm 1: Finding All Possible Path

---

1 F I N D-AllPath(Graph(G), Src(S), Dest(D))
Output: List all path
2if S == D **then**
3 Path Found
4 ADD Path to List all path (list of list)
5else
6 **for** every adcj node that is adjacent to 0S 0 do
7 **PUSH**adcj node to Path
8 F I N D-AllPath(G, adcj node, D)
9 **POP**adcj node from Path

User requests (Req) come in two parts, as shown in equation (4). In the current phase, it uses Local attributes $(LQoS)$. The Local QoS is defined as every individual abstract service in the service composition sequence, and is denoted as

$$LQoS_i = \{lqos_1^i, lqos_2^i, \dots, lqos_y^i\} \quad (8)$$

Where $LQoS_i$ represents to local QoS of abstract service (AS_i) in the composition sequence and $lqos_{ik}$ describes k^{th} user's local attribute requirement value of AS_i.

The current phase filters the services as per the user's requirements. It needs to match the user's requirements with every service in abstract services. As defined in the equation (8), every abstract service user's needs are different for each abstract service. So, it has to filter Concrete services from each abstract service. The author constructed a Decision tree (T_i) for every abstract service (AS_i) to achieve this. The proposed system used the "Iterative Dichotomiser 3" (ID3) algorithm to construct a decision tree.

One of the most popular "Machine Learning" algorithms is Decision trees, which constantly splits data into classes or groups. J. Quinlan introduced "Iterative DiChaudomiser 3" (ID3) [19] in 1986. It works based on the "Concept Learning System (CLS)" algorithm. ID3 comes under "Supervised Learning Algorithm" for regression, or classification of

continuous data [20], construct a decision tree from a set of examples, and classify future sample uses resulting tree. ID3 algorithm makes a decision tree based on IG ("Information Gain") acquired from the training instances and later used to classify the test data. The proposed Decision tree construction algorithm is as shown in Algorithm 2

Algorithm 2: Decision Tree Construction

1D-Tree(Services (C S) , attrs(A) , Target_attr(T))
2if CS is *Empty***then**
3 **return** Create Single node with Failure Value ; /* No Data Available */
4if$\forall_{i=1}^{k} CS_I$ values ∈ same targetattributes(T) **then**
5 **return** Create Single node with CS ; /* k number of service available */
6if A is *Empty***then**
7 **return** Create Single node with ∃CS ∈ most common values of T
8*bestAtr* ← the attribute with highInformationGain(IG)
9 node = Create a node with *bestAtr*
10for each possible value v of bestAtr**do**
11 *subset* = subset of ∃(CS) that having value v for *bestAtr*
12if*subset* is not *Empty***then**
13 node.addBranch(D-Tree (subset, A, T)
14return node.

The Algorithm has three input parameters; S_i 's Concrete services $\left(j = 0\middle|\ \middle|kCS_{ij}\right)$ list, QoS attributes that evaluate respective services (A_i), and the last target attributes (T_i). This Algorithm applies to every Abstract service in Abstract Service Composition. Passing attributes (i.e., $A_i \wedge T_i$) must match the user's constraints and need to follow rules. The rules are as follows.

$$A_i \subseteq AS_i^{QoS} \quad (9)$$

$$LQoS_i \subseteq A_i \quad (10)$$

$$T_i \in GQoS \quad (11)$$

4.2.1 Information Gain

The Information Gain is used to decide which attribute in a given training vector is beneficial for distinguishing between the classes.

Algorithm 3: Information Gain

---

1 I N F O R M A T I O N-Gain(S, a, T) /* Here S ⊆ CS, a ⊆A,and T is target attributes */
2ig = E N T R O P Y (S, T)
3for v in attributeValues(S,a) **do**
4 *ss* = subset(S,a,v)
5 *ig*− = (number of *ss*)/(total number of S) * E N T R O P Y (ss, T)
6 **return***ig*

7 E N T R O P Y(S, T)
8 r ← 0
9 group[ ] = number of values s ∈ S based on t ∈ T
10for x in range(group) **do**
11 *temp* = group[x]/totalnumberofS
12 *r*− = temp ∗ logtemp2
13 **return***r*

Local attributes $LQoS_i$ are applied to $AS_i's$ decision tree, and matched services will filter. These services are called candidate services for a specific task (AS_i).

4.2.2 Optimal Service Composition

The mix of services is referred to as Service Composition (SC) [21]. This problem can be solved by using or reducing available services. It enables professional application betterment, service use, and complete service with finite use. The primary objective of SC is to discover a collection of services (i.e., one candidate service (CS_{ij}) from each task (AS_i)), which add to SC and handle the user's constraints such as QoS attributes $(GQoS)$ and W weights of the attributes, as shown

$$GQoS = \{gQoS_1, gQoS_2, \dots, gQoS_z\} \quad (12)$$

$$W = \{w_1, w_2, \dots, w_z\} \wedge \sum_{k=1}^{z} w_k = 1 \quad (13)$$

The Genetic Algorithm (GA)[22] is one of the oldest and most famous optimization techniques based on nature. It searches for a solution space that mimics the natural process of the environment. GA has an individual (possible service composition) population, generally called a "Chromosome," representing possible solutions. The problem is defined by Objective Function (OF). The value of an individual on the Objective function decides the quality. The value of an individual is referred to as the "Fitness of the individual." The optimal value (maximum or minimum) of an individual could chance to select the next generation of the population. The algorithm 4 has three stages; Selection, Crossover, and Mutation.

Selection is a vital motivation stage of the Genetic Algorithm. Here, the best individual gets a higher probability of food and mating. In addition, it compels their genes to donate more to producing the next generation of identical species. Motivating from this straightforward idea, the GA uses a roulette wheel to allocate probabilities to individuals and select them for forming the next generation symmetrical to their fitness values.

After selecting individuals, it must be used to form a new generation called Crossover. In nature, combining genes in male and female chromosomes for a new chromosome. The same idea connects selected solutions (i.e., Parents) to produce two off-springs (i.e., Children). Several techniques of crossover operator in literature, such as Single-point and double-point[23] are shown in figure 4.

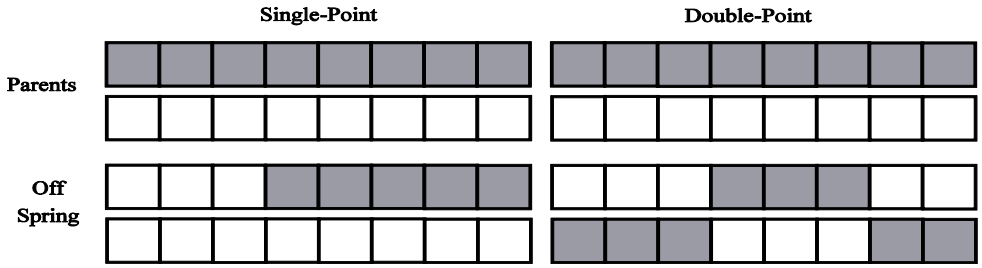

Fig 4: Crossover Techniques

Every chromosome's number of the crossover point depends on the number of Tasks (AS_i) in service composition. Due to crossover points, chromosomes divide into multiple blocks, each block represented by one Task. Consider $T_{sc} = \{AS_1, AS_2, AS_3\}$ using sequence composition mode and each abstract service having 3, 7, 6 number of candidate services (CS_i) respectively, so defined chromosome as shown in figure 5 and it represents one of the possible service composition with 2^{nd}, 5^{th} and 4^{th} candidate services from each abstract service.

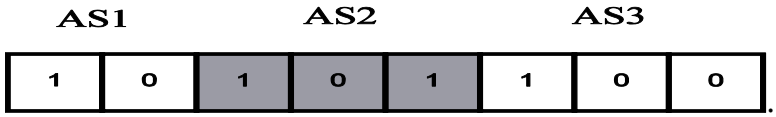

Fig 5: Sample Chromosome

In mutation, one or multiple genes are updated after off-springs are generated. Usually set low mutation rate in GA because high mutation rate converts "Genetic Algorithm to Primitive Random Search." Mutation presents another level of randomness to maintain the variety of the population. Sample example as shown in figure 6 and black cells are modified values in off springs

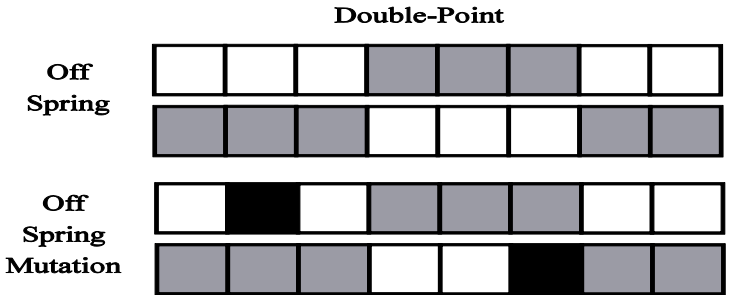

Fig 6: Sample Mutation

Objective Function (OF) applies to the individual, and the result value defines an individual's fitness. Based on these values (i.e., optimization (maximum or minimum)), an individual could have a chance to select the next generation of the population.

Service Composition (T_{sc}) has $'n'$ number of tasks, and possible Service Composition (SC_i) is represented as

$$SC_i = \{CS_{1a}, CS_{2b}, \ldots CS_{nc}\} \text{ and } \forall a, b, c \in No.of.service \qquad (14)$$

As per equation (12), user's having $'z'$ number of QoS requirements. These QoS attributes of SC_i evaluated with help of Table 1 as shown

$$SC_i^{QoS} = \{scqos_1, scqos_2, \dots, scqos_z\} \quad (15)$$

Fitness of SC_i is defined with the help of OF. Where OF depends on the type of QoS and the positive and negative attributes discussed in the section 3.2. The objective function define as

$$Fitness_i = W_{\sum - W_{\sum}} \quad (16)$$

$$W_{\sum} \quad \sum w_p * scqos_p \quad (17)$$

where $w_p \in W$ eq. (13) and$p \in$ all Positive attributes

$$W_{\sum} \quad \sum w_o * scqos_o \quad (18)$$

where $w_o \in W$ eq.(13) and $o \in$ all Positive attributes

Algorithm 4: Service Composition - Genetic Algo

---

1 GA-Service-Comp(*SPop,gqos,nt*) /* *gqos*∈*GQoS*and *nt* − *numberoftask**/
2 Create initial Population(*P op*) from *SP op*
3 while *termination not satisfied* **do**
4 *fitness* = OBJ-Function(*P op, gqos,nt*)
5 append *best fit* ← = (minimum.*fitness*) && (match ∃*gqos*)
6 *parents = selection* from *P op*
7 *offsprings= crossover* of *parents*
8 *offspr mutation = mutation* of *offsprings*
9 *P op= combine* (*parents, opffspr mutation*) /* new Service Population */
10 *best sc*= minimum.*best fit* /* minimum optimal value */
11 return *best sc*

12 OBJ-Function(*Pop, gqos, nt*)
13 for *p in range{|P op|}* **do**
14 **for** *g in range{|gqos|}* **do**
15 **for** *t in range{nt}* **do**
16 Evaluate *sqosg*using equations (5, 6, 7, 15) and Table I.
17 Evaluate *W Sum P attr*using equation (17).
18 Evaluate *W Sum N attr*using equation (18).
19 Evaluate *fitnessi*using equation (16).
20 append *fitness* ← *fitnessi*.
21 return *fitness*

5 Performance Study

This evaluation aims to verify our proposed service composition using the Decision Tree and Genetic Algorithm and achieves close and optimal results considering the QoS of the user's choice.

The scalability of Service Composition with QoS affects the performance of the service composition algorithm. It depends on three factors, the $'n'$ number of required Tasks (AS_i), the S_i number of candidates (Concrete) service (CS_{ij}), and the $'z'$ number of QoS constraints by the user. This problem was identified as NP-hard Problem [24].

5.1 Experimental Evaluation

It has conducted comprehensive simulations to estimate the performance of the proposed Service Composition based on the QoS approach as describe in this section.

5.1.1 Evaluation Methodology

It has made different test cases of QoS aware Service composition. Every test case considers the $'n'$ number of Abstract Services and $'s'$ number of Concrete Services for each Abstract Service and $'z'$ number of QoS constraints. By changing these values to create a group of test cases, each identical combination of these values represents one test case.

To start a test case process, it has to create random QoS datasets for each task. Each datasets contain measurements of seven QoS parameters of 1000 services. Table 2 shows each parameter description and range and sample dataset as shown in Table 3.

To process the performance of the proposed approach, here, consider sequence composition mode and a varied number of tasks (n) such as 2, 4, 8, and 10. It allocates an equal number of random services assigned to all tasks. Allot (s) 5 to 200 concrete services by incrementing 5 services in every iteration of a task and evaluating the proposed Service composition. The service composition depends on user QoS constraints (Req) are local and global as per equation (4).

In Hierarchy Service Filtering Phase, applying user request $(LQoS)$ to a set of services in a task and constructing a Decision tree as per algorithm 2. In the test cases, $LQoS$ are Availability (A), Throughput (Th),

TABLE 2: QoS Parameters of Datasets

QoS Attribute	Description	Range
Cost(C)	Amount of service use	80 to 300
Response Time(Rt)	Difference of time from service request to response	0.1 to 2.5Sec
Availability(A)	No. of Successful request per total request	80% ≈ 0.8 to 0.9999
Reliability(R)	Ratio of no. of error per total response	80% ≈ 0.8 to 0.9999
Success rate(Sr)	No. of Successful complete task per total response	65% ≈ 0.65 to 0.9999
Throughput(Th)	Average number of jobs completed in unit time	1 to 20
Reputation (Rp)	Service Credibility include capability of service and certain condition	1 to 5

and Response Time (Rt), and the Target attribute as per equation (11) is $Reliability\ (R) \geq 0.85$. A sample Decision Tree parameters (Information Gain and Entropy) of 15 concrete services of a task as shown in Table 4 and Decision Tree as shown in figure 7.

TABLE 3: Sample Datasets

Sr.No	C	Rt	A	R	Sr	Th	Rp
0	228	1.771	0.904	0.866	0.895	12	2
11	105	1.452	0.843	0.938	0.817	5	2
992	299	2.317	0.939	0.864	0.929	6	3

Applying $LQoS$ are $Availability\ (A) \geq 0.85$, $Throughput\ (Th) \geq 6$, and $ResponseTime\ (Rt) \leq 1.5$, and the Target attribute $Reliability\ (R) \geq 0.85$ and filter out candidates services for each task.

TABLE 4: Sample Decision Tree Parameters

Parameter	Value
All Services	
Entropy (R)	0.970951
Information Gain (A)	0.159964
Information Gain (Rt)	0.236314
Information Gain (Th)	0.278260
Throughput (≤ 10)	
Entropy (R)	0.863121
Information Gain (A)	0.291692
Information Gain (Rt)	0.469565
Throughput (> 10)	
Entropy (R)	0.543564
Information Gain (A)	0.199204
Information Gain (Rt)	0.199204

In Optimal Service Composition, applying the user's request $GQoS$ in Service Composition and evaluating optimal fitness value based on GA algorithm 4. To prepare the test case, it has considered two categories of $GQoS$, that is, applying two $GQoS$ attributes (i.e., Cost(C), Reliable(R)) and four $GQoS$ attributes (i.e., Cost(C), Reliable(R), Response Time(Rt), Success Rate(Sr)). Here Reliability(R) consider in GQoS because of equation (11). Both $GQoS$ categories have applied to the same data and evaluated the performance of the proposed approach. $GQoS$ attributes values vary based on how many tasks are considered for Service composition.

To process this, the author has used Python 3.3 programming on the open platform "https://colab.research.google.com." The results are analysed as follows.

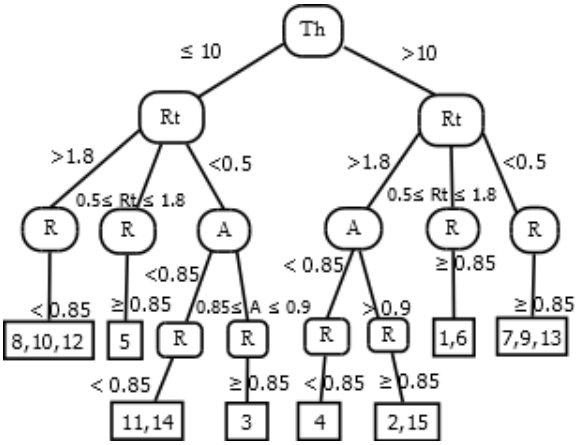

Fig. 7 Sample Decision Tree

5.1.2 Performance Results

The figure 8 shows considering 4 tasks and applying the exact data for two and four global attributes. Here, two global QoS details (i.e., Cost (C) and Reliable (R)) values are set at 330 and 0.98, respectively, and the weight of attributes is 0.3 and 0.7, respectively. In addition, four global QoS details (i.e., Response Time(Rt), Success Rate (Sr)) values are set at 0.8 and 0.8, respectively, and all four attributes weights are [0.2, 0.4, 0.3,0.1], initial two weights have changed because equation (13).

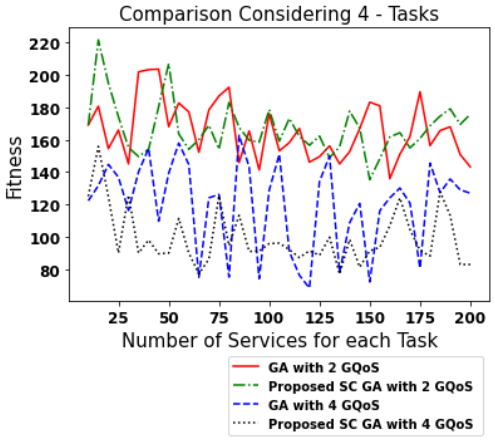

Fig 8. Comparison Considering 4 number Tasks

The figure 9 shows considering 6 tasks and similar constraints and weights as applied in Task 4 with different QoS values (i.e., C = 600, R = 0.8, Sr = 1 and Rt = 0.75).

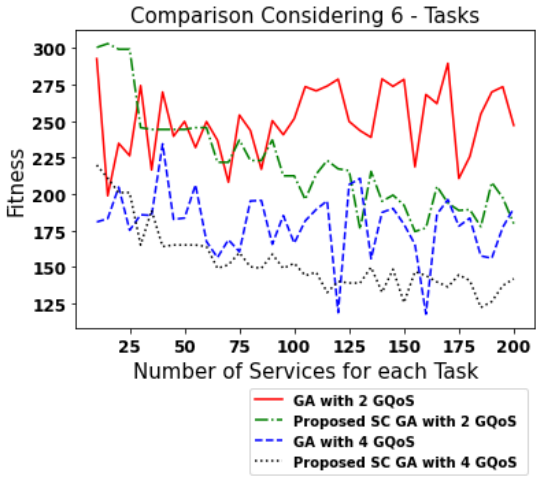

Fig 9. Comparison Considering 6 number Tasks

The figure 10 shows considering 8 tasks and similar constraints and weights as applied in Task 4 with different QoS values (i.e., C = 800, R = 0.8, Sr = 2 and Rt = 0.65).

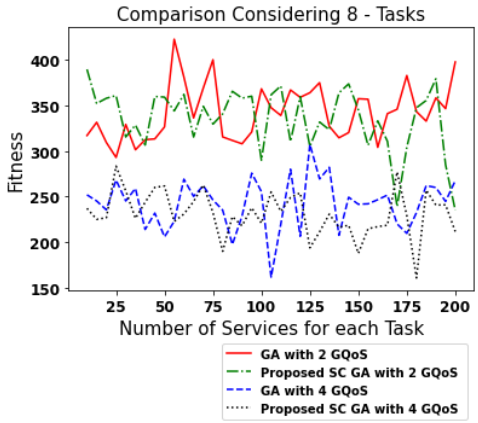

Fig 10. Comparison Considering 8 number Tasks

The figure 11 shows considering 10 tasks and similar constraints and weights as applied in Task 4 with different QoS values (i.e., C = 1000, R = 0.7, Sr = 5 and Rt = 0.65).

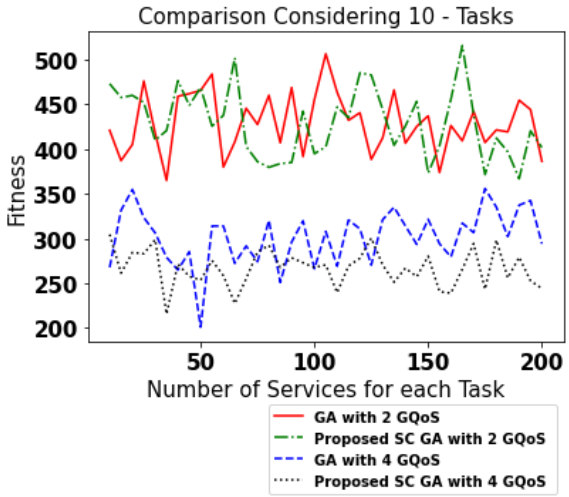

Fig 11. Comparison Considering 10 number Tasks

6 Conclusion

This paper presented a well-planned heuristic for the QoS aware service composition, also known as NP-hard. The proposed approach combined global optimization with the help of the local hierarchy filter method. This approach dramatically reduces execution costs. The evaluations represent a considerable improvement in mapping with the user's requests and gaining close to optimal results. This approach benefits applications with dynamic nature and real-time requirements and the most suitable for IoT environments. The proposed method has three stages, Traversal Sequence Acquiring Phase, Hierarchy Service Filtering Phase, and Optimal Service Composition. In addition, user requests (Req) come in two parts; Local attributes and Global attributes. Local attributes are applied to filter the services in the Hierarchy Service Filter, and Global attributes are used to obtain optimal service composition in the last stage. For each phase, algorithms are defined as results, the current approach has a more stable optimal value even number of services, tasks, and QoS increases linearly. Moreover, the scalability of the proposed system is stable and more accurate. The approach has given more accurate results compared with fewer number constraints.

References

[1] D. Gupta, S. Rani, S. H. Ahmed, S. Verma, M. F. Ijaz, and J. Shafi, "Edge Caching based on Collaborative Filtering for Heterogeneous ICN-IoT Applications." Sensors 21 (16): 5491.2021

[2] A. Bouguettaya, M. Singh, M. Huhns, Q. Z. Sheng, H. Dong, Q. Yu,A. G. Neiat, S. Mistry, B.Benatallah, B. Medjahed et al. "A Service Computing Manifesto: The Next 10 Years." Communications of the ACM 60 (4): 64–72.2017

[3] S. Rani, D. Koundal, M. F. Ijaz, M. Elhoseny, M. I. Alghamdi et al., "An Optimized Framework for WSN Routing in the Context of Industry 4.0." Sensors 21 (19): 6474.2021

[4] I. Aoudia, S. Benharzallah, L. Kahloul, and O. Kazar, "Service Composition Approaches for Internet of Things: a Review." International Journal of Communication Networks and Distributed Systems 23 (2): 194–230.2019

[5] M. A. Razzaque, M. Milojevic-Jevric, A. Palade, and S. Clarke, "Middleware for Internet of Things: a Survey." IEEE Internet of Things Journal 3 (1): 70–95.2015

[6] Manu A P. "Investigating Service Oriented Network Architecture for Internet Functionalities." PhD thesis, Thesis, Indian Institute of Information Technology, Allahabad, India, October.2012

[7] A. Botta, W. De Donato, V. Persico, and A. Pescape, "Integration of Cloud Computing and Internet of Things: a Survey." Future Generation Computer Systems 56: 684–7002016

[8] D. Rai and V. P. Kumar, "Instance based Multi Criteria Decision Model for Cloud Service Selection using TOPSIS and VIKOR." International Journal of Computer Engineering and Technology 7 (1): 78–87.2016

[9] C. Jatoth, G. Gangadharan, U. Fiore, and R. Buyya, "QoS-Aware Big Service Composition using MapReduce based Evolutionary Algorithm with Guided Mutation." Future Generation Computer Systems 86: 1008–18.2018

[10] M. Alrifai and T. Risse, "Combining Global Optimization with Local Selection for Efficient QoS-aware Service Composition." Proceedings of the 18th International Conference on World Wide Web, 881–90.2009

[11] F. Gao, E. Curry, M. I. Ali, S. Bhiri, and A. Mileo, "Qos-aware Complex Event Service Composition and Optimization using Genetic Algorithms." International Conference on Service-Oriented Computing, 386–93.2014
[12] N. Kashyap, A. C. Kumari, and R. Chhikara, "Multi-Objective Optimization using NSGA II for Service Composition in IoT." Procedia Computer Science 167: 1928–33. 2020
[13] Q. Li, R. Dou, F. Chen, and G. Nan, "A QoS-Oriented Web Service Composition Approach based on Multi-Population Genetic Algorithm for Internet of Things." International Journal of Computational Intelligence Systems 7 (sup2): 26–34.2014
[14] N. Kashyap, A. C. Kumari, and R. Chhikara, "Service Composition in IoT using Genetic Algorithm and Particle Swarm Optimization." Open Computer Science 10 (1): 56–64. 2020
[15] H. Kurdi, F. Ezzat, L. Altoaimy, S. H. Ahmed, and K. Youcef-Toumi. "Multicuckoo: Multi-Cloud Service Composition using a Cuckoo-Inspired Algorithm for the Internet of Things Applications." IEEE Access 6: 56737–49.2018
[16] T. Yu, Y. Zhang, and K.-J. Lin, "Efficient Algorithms for Web Services Selection with End-to-End QoS Constraints." ACM Transactions on the Web (TWEB) 1 (1): 6 2007
[17] L. Zeng, B. Benatallah, A. H. Ngu, M. Dumas, J. Kalagnanam, andH. Chang, "QoS-aware Middleware for Web Services Composition." IEEE Transactions on Software Engineering 30 (5): 311–27.2004
[18] R. Tarjan, "Depth-First Search and Linear Graph Algorithms." SIAM Journal on Computing 1 (2): 146–60.1972
[19] J. R. Quinlan, "Induction of Decision Trees." Machine Learning 1 (1): 81–106.1986
[20] S. Li, D. Xu, Y. Liu, R. Wang, and J. Zhang, "Identification Method of Influencing Factors of Hospital Catering Service Satisfaction Based on Decision Tree Algorithm." Applied Bionics and Biomechanics 2022.
[21] P. Asghari, A. M. Rahmani, and H. H. S. Javadi, "Service Composition Approaches in IoT: A Systematic Review." Journal of Network and Computer Applications 120: 61–77.2018
[22] S. Mirjalili, "Evolutionary Algorithms and Neural Networks: Theory and Applications." In, 43–55. Cham: Springer International Publishing.2019
[23] M. Srinivas and L. M. Patnaik, "Genetic algorithms: A Survey." Computer 27 (6): 17–26.1994
[24] M. Zhang, L. Liu, and S. Liu, "Genetic Algorithm based QoS-aware Service Composition in Multi-Cloud." IEEE Conference on Collaboration and Internet Computing (CIC), 113–18. 2015

Authors Profile

Pavan Kumar V, completed M. Tech., from Indian Institute of Information Technology, Allahabad, India, and pursing Ph.D. in Computer Science & Engineering from Visvesvaraya Technological University, Belagavi, India. He is a member of CSI. He has 11+ years of Industry & Teaching experience and 6+ years of Research experience. He has 14 publications in Conference and International Journals. His research interest includes Cloud Computing, and IoT..

SudheerShetty is working as Associate Professor & Head the Department of Information Science & Engineering at Alva's Institute of Engineering & Technology, Moodbidri, Mangaluru, Karnataka. He has 21+ years of experience in teaching and 5+ years of experience in research domains. In Cloud Computing area pursuing his Ph.D.

Janardhana D R pursued M.Tech., degree from the Visvesvaraya Technological University, Belagavi, Karnataka, India. He works for NMIT, Bengaluru as Assistant Professor in the department of Computer Science. His research interest includes IoT, Data Science, Network Security, and Machine Learning. Currently pursuing Ph.D. in the area of IoT and its security.

Manu A P, Professor in the Department of Computer Science and Engineering, having 20+ years of academic and industrial experience. Awarded Doctoral degree from IIIT, Allahabad. India. Has authored two books titled "A to Z Linux" and "ABC of Hadoop, Docker and Ansible". He is a Fellow of the Institution of Engineers (India), Members of CSI, and ISTE. His research area of interests are Cloud Computing, and Next Generation Networks. He is a reviewer of Elsevier, InderScience, Taylor and Francis journals.