Wireless Networks (2020) 26:2329–2338
https://doi.org/10.1007/s11276-019-02121-4

Genetic algorithm based adaptive offloading for improving IoT device communication efficiency

Azham Hussain[1] · S. V. Manikanthan[2] · T. Padmapriya[2] · Mahendran Nagalingam[3]

Published online: 23 August 2019
© Springer Science+Business Media, LLC, part of Springer Nature 2019

Abstract
Improving the communication of Internet of Things (IoT) network is a challenging task as it connects a wide-range of heterogeneous mobile devices. With an extended support from cloud network, the mobile IoT devices demand flexibility and scalability in communication. Increase in density of communicating devices and user request, traffic handling and delay-less service are unenviable. This manuscript introduces genetic algorithm based adaptive offloading (GA-OA) for effective traffic handling in IoT-infrastructure-cloud environment. The process of offloading is designed to mitigate unnecessary delays in request process and to improve the success rate of the IoT requests. The fitness process of GA is distributed among the gateways and infrastructure to handle requests satisfying different communication metrics. The process of GA balances between the optimal and sub-optimal solutions generated to improve the rate of request response. Experimental results prove the consistency of the proposed GA-OA by improving request success ratio, achieving lesser complexity, delay and processing time.

Keywords Fitness function · Genetic algorithm · IoT · Request processing · Traffic offloading

1 Introduction

Internet of things (IoT) is development in wireless technology that facilitates smart device communication between the real-world "things". The real-world entities are connected to the communicating equipment or device through internet. The real-world entities called "things" possess smart computing and communicating abilities aided by the built-in hardware and software. The hardware-software combo includes actuating and sensing units, human-interacting units, middleware, graphical user interface, and storage and power source. IoT devices support anywhere anytime access to distributed information, applications and services by establishing communication with a common network (like internet). Internet service providers (ISP) are the listening agents for serving IoT requests and application demands by adapting a wide-range of communication technologies such as wireless LANs, Wi-Fi and LTE. IoT technology is scalable, flexible and adaptable to support the dynamicity of the end-user devices. These features of IoT are reliable for both fixed and ad-hoc infrastructures [1, 2]. Information handling and forwarding are the vital tasks of in an IoT environment, demanding the co-operative working of the other devices. The rate of information handled by the IoT devices and the network relies on the density of users and requests instigated [3].

The existing infrastructure support of IoT devices lacks efficiency in handling co-operative and flexible communications. The density of the users and scalability of the network are the two influencing factors that requires

✉ Azham Hussain
azhamuum@gmail.com

S. V. Manikanthan
manikanthan.s.v@gmail.com

T. Padmapriya
padmapriyaa85@pec.edu

Mahendran Nagalingam
drnmpower@gmail.com

[1] Human-Centered Computing Research Lab, School of Computing, Universiti Utara Malaysia, 06010 Sintok, Malaysia

[2] Melange Academic Research Associates, Puducherry, India

[3] Department of EEE, SAINTGITS College of Engineering, Pathamuttom, Kottayam, Kerala, India

external network support for balancing request handling and resource allocation problems. Resource allocation is successful if the request is of the IoT user is responded with a proper in-time response. To achieve maximum profit in service efficiency, scalable architecture such as cloud orchestration serves as a support for IoT network. Cloud-integrated IoT architecture achieves improved resource allocation, access and sharing surviving user demands. The features and facilities of cloud are adopted by the IoT devices to manage their communication demands. Besides, this orchestration provides infrastructure related services in a heterogeneous manner with scalability [4–6].

Traffic handling and congestion are the prime factors that are to be optimized in this orchestration process. Unhandled traffic regulation results in service drop-outs, defacing the network performance. Communication defacing process instigates in infrastructure and gateway storage present in the IoT-Cloud architecture. Unhandled congestion results in response loss and increase in retrieval time. To smooth congestion, the gateways and other infrastructure require additional queuing and processing time that reflects in service delays of the IoT users. The existing challenges in IoT are addressed by process virtualization, distributed access and process synchronization. The process of request offloading is recommended in a congested scenario for preventing unnecessary request loss and communication failures. The features of IoT communications are synchronized with the cloud services to provide better service reliability. The contrary adjustments between IoT users and processing gateways/infrastructure need to be well addressed to minimize the impact of congestion over the communication [7, 8]. The contributions of the work are as follows:

(1) Designing an adaptive request offloading method for user concentric IoT environment to improve the rate of processing and service response.
(2) Applying genetic concepts for un-biasing sub-optimal and optimal solutions during offloading so as to improve the success rate of request handling.
(3) Presenting a comparative analysis for verifying the reliability of the proposed method through conventional metrics considered.

The organization of the paper is as follows: Sect. 2 provides the description of the works related to the conceptual analysis and followed the proposed genetic based adaptive offloading in Sect. 3. The performance analysis of the proposed method with a comparative study is presented in Sect. 4. Section 5 provides a concluding discussion for the proposed method.

2 Related work

The authors in [9] introduced a delay minimization framework for IoT-fog-cloud applications. This method incorporates collaboration and offloading rules for minimizing service response time. The rules are constructed with the consideration of request types and storage size for minimizing delay. This framework is adaptable to any network architecture despite of the density of users and processing servers.

A computation offloading based on game-theoretic approach [10] is introduced for minimizing the operation cost and to improve the user benefits. The offloading process improves the rate of fog and cloud resource allocation to the IoT users. The satisfaction of the users is improved through a self-achievable quality of experience maximization process. User benefits are improved in terms of energy and delay through optimized power exploitation.

Collaborative offloading scheme is presented in [11] for achieving energy conservation in wireless networks integrated IoT devices. In this scheme, energy efficient computation offloading process is distributed among mobile devices, cloud servers and edge computing devices. Using the conventional multiple access communication technology, this scheme achieves better energy efficiency in IoT devices.

An edge-computing offloading is exclusively proposed for dense IoT networks by Guo et al. [12]. This offloading process operates in a greedy manner in two-tiers to disseminate incoming tasks among local and mobile-edges. This offloading process is suitable for dense IoT network as it minimizes processing time, energy consumption and computation overhead.

Stabilized green crosshaul orchestration (SGCO) [13] is a joint offloading optimization introduced for service concentrated IoT network. The joint offloading process is targeted to improve energy efficiency and network stability along with latency minimization. This offloading process makes use of Lyapunoy-theory based drift and penalty policy to determine the rate of data processing to achieve energy efficiency.

A light weight request and admission framework [14] is designed to improve the scalability in integrating different network architectures. This framework best suits for cloud-edge computing and IoT integrated architecture. This framework facilitates selective offloading by operating at the IoT and cloud layers independently. The experimental analysis of the framework proves its consistency by improving energy efficiency and minimizing latency of IoT communication devices.

The authors in [15] analyzed the modeling and deployment of heterogeneous mobile cloud with offloading and non-offloading devices. The outage analysis of the cloud system with its support for remote systems and other network are analyzed through a stochastic offloading method. The analysis concludes that proper placement of cloudlets and distribution of offloading process minimizes the rate of outage in IoT communications.

The authors in [16] integrated k-means algorithm and decision offloading for improving the performance of IoT device communications. K-means algorithm segregates the network to position edge servers to balance computation offloading. The distributed edge servers are used to make decisions at the time of offloading. This method minimizes the latency and operation cost of the devices through decision based offloading.

Aura [17] is an IoT based cloud model that delivers ad-hoc and flexible computing for offloading the tasks of IoT users. With the help of localized IoT devices, aura provides migration, initialization and processing features of the tasks from distributed locations. This model improves the task handling capacity with lesser energy and cost factors.

The authors in [18] designed a proactive decision making process for improving the device-to-device communication reliability. The decision making process operates in a cooperative manner considering different optimization metrics to minimize complexity in processing. This cluster-based cooperative decision making process minimizes the failure cost of the devices with minimum cluster-head changes.

A clustering based IoT service classification is designed in [19] for improving the efficiency in platform integration. The classification is preceded by an expectation maximization algorithm for improving the integration process. The platform integration is facilitated by estimating the similarities between platform and services.

A proactive caching technique is introduced in [20] for task distribution in edge nodes. The task distribution process among edge nodes connected to fog network is facilitated using popularity of the process. The user nodes are calibrated using one-to-one game model for task distribution from the cloudlets. The game modeling follows queuing and offloading process simultaneously for minimizing latency and delay.

For improving the efficiency of IoT services forwarded by peer-to-peer (P2P) communication networks, an efficient data relaying method is proposed in [21]. This method pre-estimates the connectivity between P2P nodes and IoT service providers using Bayesian classifier. Simultaneously, the data relaying process is monitored by the classifier to ensure the successful data delivery at the sink node. This method handles higher traffic minimizing delay and improves the transmission count.

Loadbot [22] is an agent based load balancing scheme designed for IoT networks. This agent based scheme pre-estimates the network load and the available users to disseminate incoming network traffic. A deep learning method is used to disseminate load over the available users for improving the communication efficiency. This scheme minimizes delay despite the density of sensor and learning iterate.

The existing methods concentrate in smoothing traffic without considering the maximum service possibility for the increasing request demands. In [22], load is handled with the prior knowledge of the user irrespective of the request classification. Contrarily, the offloading proposed in [12] requires a two-level processing for offloading the traffic. This requires additional processing complexity wherein the success rate achieved is less. Different from the above methods and the particular issues highlighted, the proposed GA-OA method, considers the classification of message along with the communication attributes. With the knowledge of the request message and the external influencing factors, the genetic process results in optimal solution for improving the service efficiency of IoT users.

3 Genetic algorithm based adaptive offloading for dense IoT

The proposed adaptive offloading process is designed to minimize gateway overloading due to increasing IoT user demands. The process is induced by genetic approach where offloading and request servicing are performed simultaneously to achieve maximum success rate. The architecture of a typical IoT environment is illustrated in Fig. 1.

The architecture is divided into three parts: IoT devices, Infrastructure and cloud. This architecture is presented as a general process by which the same is adapted in the proposed method. The proposed GA-AO is performed in an environment as illustrated in Fig. 1.

3.1 IoT device

IoT devices are smart storage and computational enabled communicating components. They possess built—in radio units for transmitting and receiving information. The devices are mobile and fixed depending upon the application it is used. The IoT devices possess two types of communication: device-to- device and device-to-infrastructure. IoT access cloud using the infrastructure units deployed.

3.2 Infrastructure

The infrastructure part consists of base station, access – points and other forwarding wireless routers. The

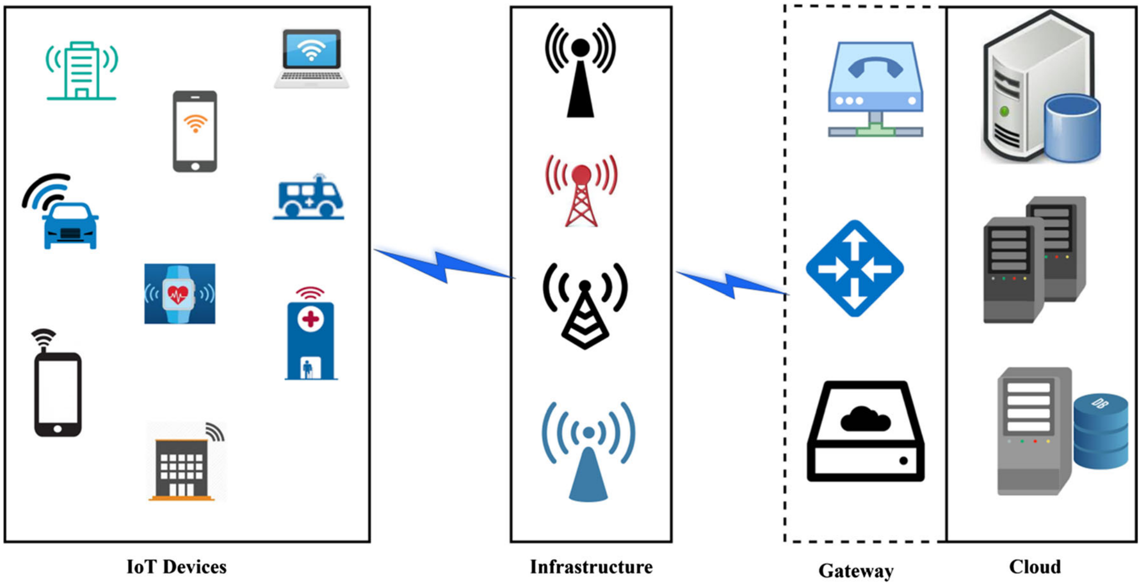

Fig. 1 IoT-infrastructure-cloud architecture

infrastructure facilitates inward and outward communication between IoT and cloud components. Infrastructures are both ad-hoc and fixed to supports IoT communication flexibility.

3.3 Cloud

The cloud part of the architecture is packed with dedicated servers that are distributed and centralized. Servers provide access to applications and services as required by the IoT devices. The cloud endorses both local and distributed storage information access and retrieval.

3.4 Gateway

Cloud gateways streamline the IoT traffic to the servers. Request forwarding and response reply are maintained by the gateways. Gateways are also called load-balancers that aid seamless communication and in-time response to both the connected end devices.

4 Methodology

The process of traffic handling and request processing is carried out in both infrastructure and gateway part of the architecture. The notations used in the methodology is given in Table 1.

The density of the requests observed at a time t_i is estimated using Eq. (1)

Table 1 Notation and description

Notation	Description
a_r	Arrival rate of the requests
s_r	Service rate of the requests
L_n	Request load from n IoT users
p_s	Processing rate of the server
t_p	Time for processing a request
t_q	Queuing time of a request
t_{res}	Response time
ρ_r	Density of the requests
t_i	Request initialization time
t_{max}	Maximum time of a request
c	Capacity of the wireless link
g_s	Storage capacity of the gateway infrastructure
t_o	Offloading time

$$\rho_r = \sum_{j=1}^{k} \frac{L_n(j)}{t_j} \tag{1}$$

where,

$$\sum L_n = \frac{a_r(j) * t_i}{p_s(j)} \tag{2}$$

Subs (2) in (1), we get

$$\rho_r = \sum_{j=1}^{k} \frac{a_r(j)}{p_s(j)} \tag{3}$$

Considering Eq. (3) as the density of requests that is to be serviced, the objective of the proposed offloading process is defined as follows:

$$\max \sum_{j=1}^{k} a_r(j) \forall \frac{s_r}{\rho_r} = 1 \tag{4}$$

Such that

$$\min \sum_{j=1}^{k} t_{res}(j) \tag{5}$$

In Eq. (4), the offloading process is defined such that the ratio between request service rate and density of the requests in the cloud layer must be 1. This means that all the received requests are processed without loss; this is an ideal condition for offloading. The service ratio relies on the response time of each request. If the response time is less, high is the number of requests serviced.

Where $t_{res} = t_i + 2(t_q + t_p)$.

The process of offloading occurs when $a_r > s_r$ or $(a_r * t_i) > (p_s * t_p)$. In this case, the time taken for processing a request increases or the request is lost.

These conditions degrade the performance of the network and henceforth Eq. (4) subject to Eq. (5) to be achieved.

There are two cases that are to be considered at the time of request processing: (1) Overloaded requests and (2) underflow storage. These two cases are balanced in a optimal manner by generating possible solution using genetic implication. The process of genetic optimization includes four stages: population initialization, estimated of fitness function, chromosome selection and genetic operator exploitation. Request load and density are different as the density represents the total requests experienced in the cloud for service. On the other hand, the load factor represents the number of requests originating from the IoT devices. In other words, the density represents the filtered requests from the available load of the IoT devices. Therefore, these two factors are estimated separately.

5 Population initialization

The initial population of the GA is assorted with n and it's corresponding t_{max}. Each of the n is mapped with its t_{max} and the sequence of processing. The sequence for processing is ordered in the first come first serve basis let G (L_n) be the genetic function that is represented as G $(L_n) = \{n, t_{max}, seq\}$. The three tuples (n, t_{max}, seq) are the population variables that are subjected to change with respect to the c of the wireless link. The first process of GA is illustrated as in Fig. 2.

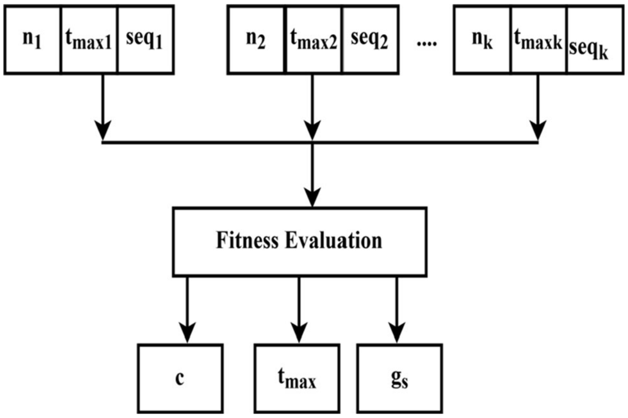

Fig. 2 Population initialization

5.1 Fitness evaluation

Let F(G) represent the fitness of an initial genetic population G. The metrics considered for evaluating the fitness are: c, t_{max} and g_s. The t_{max} metric is satisfied by serving request with $t_{res} < t_{max}$. The fitness of each metric is independently assessed; Let $f_1, f_2$and f_3 represented the local fitness of $c, g_s$and t_{max} respectively. The local fitness of f_3 is satisfied only if $f_1 \cup f_2 = 1$. This constraint for the considered metrics and their evalution is explained as follows:

The ideal number of n flowing at $t_i$and t_p requires c as using Eq. (6)

$$c = \begin{cases} a_r * t_i, \text{for request} \\ p_s * t_p, \text{for response} \end{cases} \tag{6}$$

If $(n^*a_r * t_i) > c$, then the wireless links is said to be congested. If this case is true, $f_1 \neq 0$ and $f_1 \neq 1$ (i.e.) $0 < f_1 < 1$. Similarly, the storage of the gateway/infrastructure is estimated for its acceptance rate. The storage occupancy of a gateway is estimated using Eq. (7)

$$g_s^* = g_s - \left[\left(\frac{a_r * t_i}{n}\right) * g_s\right] \tag{7}$$

There are two possible outcomes for g_s^* (i.e.) if $g_s^* > n$, then the storage is underflow and hence it can accept further requests. Therefore, from (6), even if $(n^*a_r * t_i) > c$, the gateway is capable of accepting the request and $f_1 = 1$. Contrarily, if $g_s^* < n$, the storage is overloaded and it cannot accepts further requests. In this case, if $t_q > t_{max}$, the request is left unserviced and obviously, $f_1 = 1$. The local fitness of g_s(i.e.)$f_2 = 1$, if $g_s^* > n$ provided $(n^*a_r * t_i) \leq g_s$. In other cases, $f_2 = 0$. The union of the local fitness is thus represented as

$$f_1 \cup f_2 = \begin{cases} 1, \text{if } g^* > n\&\&(n * c) \leq g_s \\ 0, \text{otherwise} \end{cases} \tag{8}$$

Considering the optimal case when $f_1 = f_2 = 1$, the t_{max} local fitness is estimated. The first two local fitness in contrary with t_{max} must be maximum (i.e.)$f_1 \cup f_2 \cap f_3 = 1$, is the ideal condition. For all $(f_1 \cup f_2) = 1, t_{max}$ may vary depending upon the availability of queuing space. Therefore, the two cases of overload and underflow is assessed for determining the third fitness. The two cases discuss the state of the requests with respect to time such that the response time is greater or less than the maximum time observed. In those cases, the process of offloading is elaborated in the following explanation.

Case 1: If $t_{res} > t_{max}$, then the request is dropped with a delay of $(t_i + t_q)$ as it is not processed yet. In this case,$f_3 = 0$ and therefore offloading requests is required and the $t_0 \in [t_i, t_{max}]$. If $t_0 = t_{max}$, then $f_3 = 0$ and this request need not to be offloading.

Case 2: If $t_{res} < t_{max}$, $f_3 = 1$ if $(t_i + 2t_q) < t_p$ and therefore, the maximum response time is $t_i + t_p + 2t_q$. Contrarily, if $(t_i + 2t_q) < t_p$, then $f_3 \neq 0$ and $f_3 \neq 1$ (i.e.) $0 < f_3 < 1$. In this case, the resulting fitness solution is sub optimal. The next generation of chromosomes selection is designed to meet the above condition to confine $t_p \not> t_{max}$. The process of optimal and sub- optimal fitness evaluation is illustrated in Fig. 3

5.2 Chromosome selection

For the available n messages, the gateways placed are selected counting the g_s. If $g_s > n$, then the next $(n - x)$ messages offloading to the free gateways. Contrarily, if no gateway is available, then sub- optimal solution results. In an sub-optimal solution, the chromosome (represented as the mapping is operated to find additional result. The set of genetic operator implication is instigated in this process. For an sub-optimal solution to be solved, the possible estimation is achieved by genetic cross over and mutation operators.

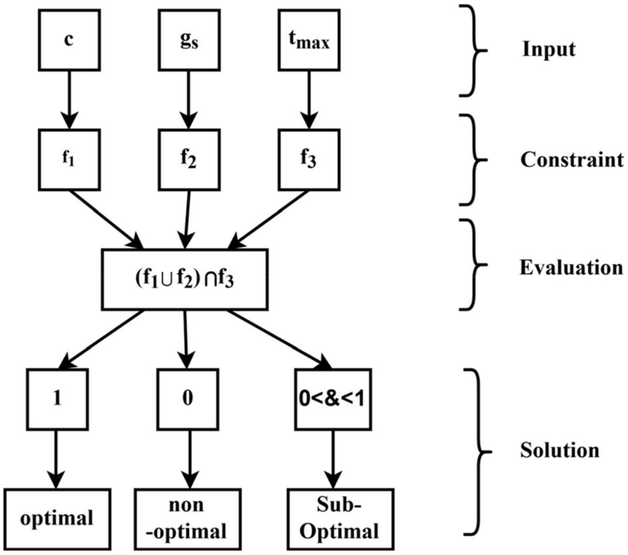

Fig. 3 Fitness evaluation

5.3 Genetic operator exploitation

In this phase, the possible outcomes for resolving sub-optimality achieved in fitness evolution. The possibilities for sub optimality are: (1) $g_s^* < n$ (ii)$t_p > t_{max}$. The sub-optimal solution is resolved to extract possible outcomes to derive solutions that are optimal for performing offloading process. The offloading process is time-dependent and hence the underflow and overflow of the neighbors are resolved using the evolution of genetic operators. These genetic operators suggest time-dependent solutions for offloading resulting in responsive requests. This helps to retain the profitability of the process and the solutions confined within t_{max} are the balanced solutions.

These two sub-optimal results are either partially addressed or open to failure. The above two cases are addressed as follows

(1) $g_s^* < n$: If the neighboring gateway is free then the excess requests are offloaded.

If the neighbor is overloaded, alternate is selected such that $t_0 \in [t_i, t_{max}]$ and $t_p > t_{max}$. In this case response delay is estimated as in Eq. (9)

$$t_{res} = t_i + 2t_q + t_p + t_0, t_p < t_{max} \tag{9}$$

The final solution generated as per Figs. 4 or 5 will satisfy Eq. (5) provided $\frac{s_r}{p_s} = 1$. $\frac{s_r}{p_s}$ Will be high as the n in c is not met its end (i.e.) the number of requests is always busy in the observed wireless link.

(2) $t_p > t_{max}$

If the above condition is true, the requests until $(t_q + t_p) < t_{max}$ can be processed without drop/unallocated service. This means, a maximum of the request is processed by classifying their processing time and hence success ratio is improved. The process with respect to time factor is illustrated in Fig. 6. The case 1 and case 2 of the process is explained in the GA process that explains the type of operator implication and generated chromosomes. The generated chromosomes are adaptable to the time

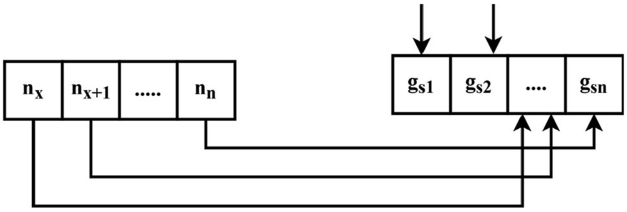

Fig. 4 Offloading process when neighbor is underflow

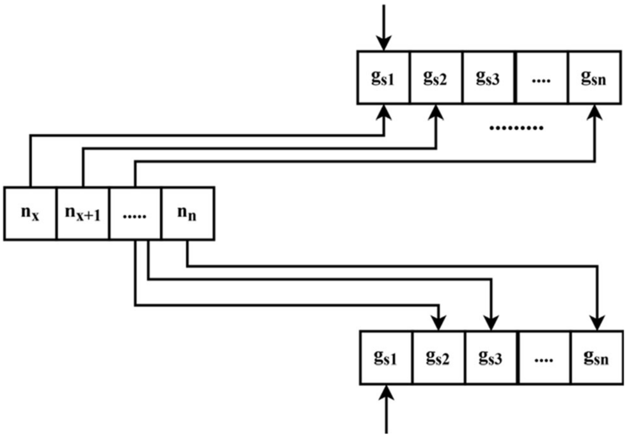

Fig. 5 Offloading process when neighbor is overloaded

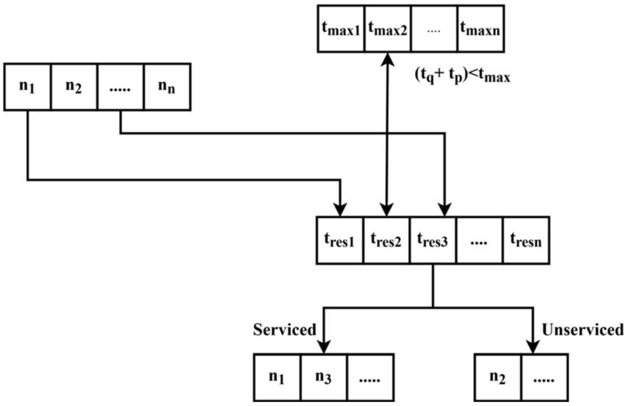

Fig. 6 Offloading process when $\left(t_q + t_p\right) < t_{max}$

factor of the devices to ensure the available requests are serviced before the maximum time.

As a result of genetic process, optimal solution results in mapping all ρ_s to the resource available and the appropriate response is given by the cloud source. In the non-optimal output of the genetic process, the requests $(n - x)$ are served with the constraint $\left(t_q + t_p\right) < t_{max}$ achieving higher success ratio.

6 Results and discussion

The proposed GA-AO for request handling in IoT is modeled using Opportunistic Network Environment (ONE) simulator. This simulation is supported by an open-cloud model that owns a local and distributed repository for processing messages. The simulation requirements and its values are presented in Table 2.

The proposed GA-AO is compared with the existing Loadbot [22] and game-theoretic greedy approximation offloading algorithm (GT-GAOA) [12] for the metrics: average delay, processing time, response complexity and success ratio.

Table 2 Simulation configuration and values

Parameter	Value
IoT devices	200
Gateways	16
Request size	64 Kb
Request/s	10
t_{max}	12 s
Bandwidth	1 Mbps
g_s	24
p_s	300 messages/s

6.1 Average delay analysis

Figure 7 illustrates the average delay observed in the existing methods and the proposed GA-AO. In the proposed GA-AO, the processing time of the incoming request traffic is analyzed forehand with the knowledge of t_{maz}. The requests, whose processing time is greater than the maximum time, are offloaded to the neighboring gateways. Therefore, the requests are prevented from being queued or dropped; the additional wait time or re-servicing time is denied for the ρ_r in c. Besides, the sub-optimal results are resolved through genetic operator exploitation with the help of local fitness evaluation. These two processes are advantageous in minimizing the delay in the proposed offloading process. Compared to the existing Loadbot and GT-GAOA, the proposed method minimizes delay by 9.4% and 8.25% respectively.

6.2 Processing time analysis

The processing time of the requests with increasing gateway density is illustrated in Fig. 8. In the proposed GA-AO, the requests are processed by considering two different cases: the overloaded and underflow gateway process.

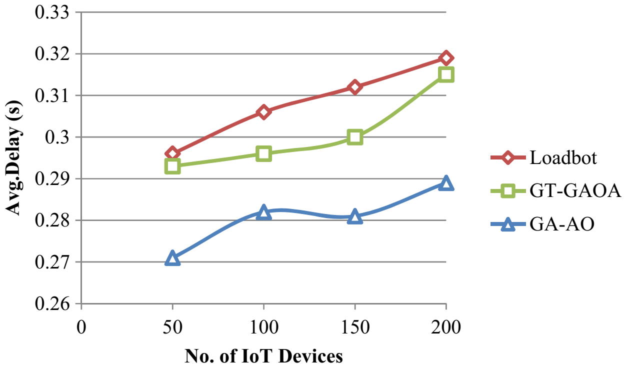

Fig. 7 Average delay comparisons

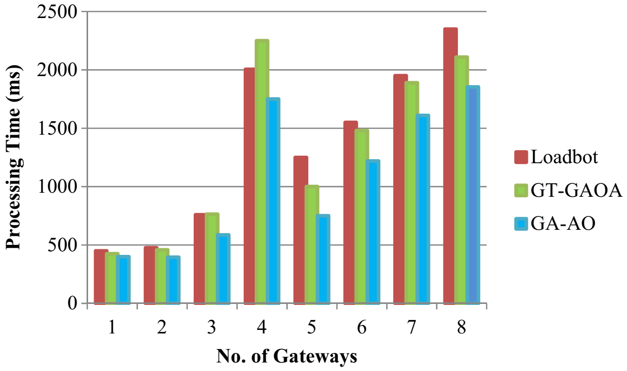

Fig. 8 Processing time comparisons

Even if $(n^*a_r * t_i) > c$, the fitness derivative of $g_s^* < n$ decides the request service. The request is processed for all $t_p < t_{max}$ and also the genetic operator exploitation process optimizes the number of sub-optimal results by offloading the requests among the available neighbors, with a flexibility of $(t_p + t_q) < t_{max}$. Therefore, the processing time of the requests are confined within t_{max}, where, Eq. (5) is satisfied. Henceforth, the processing time of the request irrespective of the delay is less in the proposed GA-OA. The proposed method minimizes processing time by 21.11% and 12.13% compared to Loadbot and GT-GAOA correspondingly.

6.3 Response complexity

The comparisons of response complexity of the existing Loadbot and GT-GAOA are compared with the proposed GA-OA in Fig. 9. The number of non-optimal solutions of request processing increases the complexity of request processing in IoT systems. The genetic operator exploitation phase of the proposed offloading method minimizes the number of non-optimal solutions by confining response to t_{max}. The number of un-attended requests from the IoT devices is less by assigning requests for processing based on response time, capacity and gateway storage. A set of local fitness estimates the favorable conditions of a message and detects the presence of non-optimal solutions at the time of fitness evaluation. The final requests processing solutions are segregated based on the fitness. The operator exploitation part ensures maximum requests are processed. Therefore, the number of requests that are to be re-serviced is less in the proposed GA-OA method. This minimizes the complexity in response by 38.1% and 38.18% compared to Loadbot and GT-GAOA respectively.

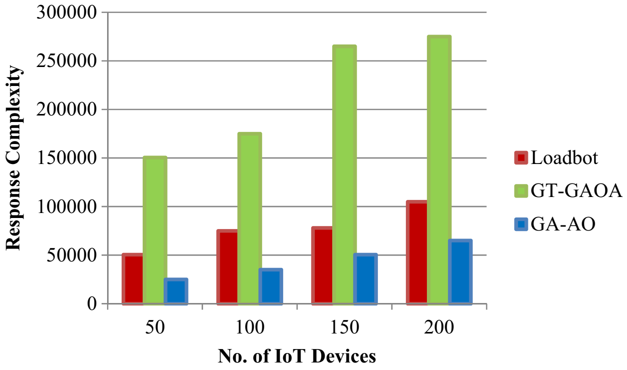

Fig. 9 Comparisons of response complexity

6.4 Success ratio

Figure 10 illustrates the comparisons of success ratio between the existing Loadbot and GT-GAOA and the proposed GA-OA. The chance for improving the request service rate in the proposed method is high. The genetic operator exploitation differentiates the non-optimal solutions that are independently concentrated to provide response for the delayed requests. The requests satisfying $(t_p + t_q) < t_{max}$ are classified to improve the success rate along with the conventional requests that satisfy $\min \sum_{j=1}^{k} t_{res}(j)$. Therefore, the rate of success is improved by 45% and 33% compared to the existing Loadbot and GT-GAOA respectively. Table 3 presents the experimental values of the proposed and existing method as in comparison.

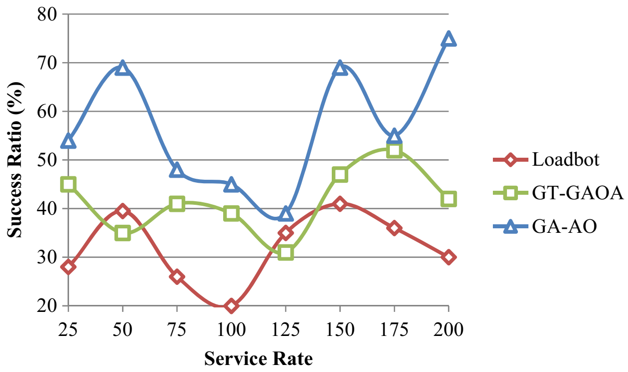

Fig. 10 Success ratio comparisons

Table 3 Experimental value comparisons

Metrics	Loadbot	GT-GAOA	GA-AO
Avg. delay (s)	0.319	0.315	0.289
Processing time (ms)	2350	2110	1854
Response complexity	105,000	275,000	65,000
Success ratio (%)	30	42	75

7 Conclusion

In this manuscript, a genetic algorithm based adaptive offloading method is proposed to improve the communication efficiency of IoT network devices. The genetic process is distributed among the infrastructure and gateways to effectively handle the requests. The process of genetic algorithm flow estimates the local fitness considering the capacity, gateway storage and maximum time of the request to achieve fair optimization. The fair optimization in request processing is achieved by minimizing processing time, delay and processing complexity. The consistency of the proposed method is proved by achieving higher success rate. The offloading process is adaptive to distribute traffic requests among the neighbors by classifying them based on response time. In the future, adaptive offloading method is planned to be enhanced with intelligent learning algorithms for extending the flexibility and scalability features in a heterogeneous network.

References

1. Al-Fuqaha, A., Guizani, M., Mohammadi, M., Aledhari, M., & Ayyash, M. (2015). Internet of things: A survey on enabling technologies, protocols, and applications. *IEEE Communications Surveys Tutorials, 17*(4), 2347–2376.
2. Deng, S., Huang, L., Wu, H., Tan, W., Taheri, J., Zomaya, A. Y., et al. (2016). Toward mobile service computing: opportunities and challenges. *IEEE Cloud Computing, 3*(4), 32–41.
3. Ning, H., & Hu, S. (2012). Technology classification, industry, and education for Future Internet of Things. *International Journal of Communication Systems, 25*(9), 1230–1241.
4. Haw, R., Alarm, M., & Hong, C. (2014). A context-aware content delivery framework for QoS in mobile cloud. In *Proceedings of IEEE NOMS* (pp. 1–6).
5. Munoz, R., Vilalta, R., Yoshikane, N., Casellas, R., Martinez, R., Tsuritani, T., et al. (2018). Integration of IoT, transport SDN, and edge/cloud computing for dynamic distribution of IoT analytics and efficient use of network resources. *Journal of Lightwave Technology, 36*(7), 1420–1428.
6. Lin, J.-W., Chen, C.-H., & Chang, J. (2013). Qos-aware data replication for data-intensive applications in cloud computing systems. *IEEE Transactions on Cloud Computing, 1*(1), 101–115.
7. Deng, Y., Chen, Z., Zhang, D., & Zhao, M. (2018). Workload scheduling toward worst-case delay and optimal utility for single-hop Fog-IoT architecture. *IET Communications, 12*(17), 2164–2173.
8. Mubeen, S., Nikolaidis, P., Didic, A., Pei-Breivold, H., Sandstrom, K., & Behnam, M. (2017). Delay mitigation in offloaded cloud controllers in industrial IoT. *IEEE Access, 5,* 4418–4430.
9. Yousefpour, A., Ishigaki, G., Gour, R., & Jue, J. P. (2018). On reducing IoT service delay via fog offloading. *IEEE Internet of Things Journal, 5*(2), 998–1010.
10. Shah-Mansouri, H., & Wong, V. W. S. (2018). Hierarchical fog-cloud computing for IoT systems: A computation offloading game. *IEEE Internet of Things Journal, 5*(4), 3246–3257.
11. Guo, H., Liu, J., & Qin, H. (2018). Collaborative mobile edge computation offloading for IoT over fiber-wireless networks. *IEEE Network, 32*(1), 66–71.
12. Guo, H., Liu, J., Zhang, J., Sun, W., & Kato, N. (2018). Mobile-edge computation offloading for ultradense IoT networks. *IEEE Internet of Things Journal, 5*(6), 4977–4988.
13. Dao, N.-N., Vu, D.-N., Na, W., Kim, J., & Cho, S. (2018). SGCO: Stabilized green crosshaul orchestration for dense IoT offloading services. *IEEE Journal on Selected Areas in Communications, 36*(11), 2538–2548.
14. Lyu, X., Tian, H., Jiang, L., Vinel, A., Maharjan, S., Gjessing, S., et al. (2018). Selective offloading in mobile edge computing for the green Internet of Things. *IEEE Network, 32*(1), 54–60.
15. Lee, H.-S., & Lee, J.-W. (2018). Task offloading in heterogeneous mobile cloud computing: Modeling, analysis, and cloudlet deployment. *IEEE Access, 6,* 14908–14925.
16. Zhang, C., Zhao, H., & Deng, S. (2018). A density-based offloading strategy for IoT devices in edge computing systems. *IEEE Access, 6,* 73520–73530.
17. Hasan, R., Hossain, M., & Khan, R. (2018). Aura: An incentive-driven ad-hoc IoT cloud framework for proximal mobile computation offloading. *Future Generation Computer Systems, 86,* 821–835.
18. Sharafeddine, S., & Farhat, O. (2018). A proactive scalable approach for reliable cluster formation in wireless networks with D2D offloading. *Ad Hoc Networks, 77,* 42–53.
19. Lee, D., & Lee, H. (2018). IoT service classification and clustering for integration of IoT service platforms. *The Journal of Supercomputing, 74*(12), 6859–6875.
20. Elbamby, M. S., Bennis, M., & Saad, W. (2017). Proactive edge computing in latency-constrained fog networks. In *2017 European conference on networks and communications (EuCNC)*.
21. Kim, S., & Kim, D.-Y. (2017). Efficient data-forwarding method in delay-tolerant P2P networking for IoT services. *Peer-to-Peer Networking and Applications, 11*(6), 1176–1185.
22. Kim, H.-Y. (2017). A load balancing scheme with Loadbot in IoT networks. *The Journal of Supercomputing, 74*(3), 1215–1226.

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Dr. Azham Hussain is the Associate Professor of Software Engineering at School of Computing, Universiti Utara Malaysia, Kedah, Malaysia. He is the founder of Human-Centered Computing Research Group, which is affiliated with the Software Technology Research Platform Center at School of Computing, Universiti Utara Malaysia. Azham Hussain is a member of the US-based Institute of Electrical and Electronic Engineers (IEEE), and actively involved in both IEEE Communications and IEEE Computer societies.

Dr. S. V. Manikanthan is a Director of Melange Academic Research Associates, Puducherry, India. His area of Research Interest is Wireless Sensor Networks. He has 21 years of experience in Anna University Affiliated Colleges and in Industrial Research Projects.

Dr. T. Padmapriya is a Managing Director of Melange Academic Research Associates, Puducherry, India. She obtained her Ph.D. in Pondicherry Engineering College, Puducherry, India. She has 11 years of experience in Academic and Research. Her area of Research Interest is LTE, Wireless Networks.

Dr. Mahendran Nagalingam, Ph.D received his B.E. in Electrical and Electronics Engineering from the Madras University in 2000, his M.Tech. in Control Systems and Instrumentation Engineering from the University of SASTRA, Thanjavur in 2004 and received Ph.D. in Electrical Engineering from the Anna University-Chennai in 2011. Currently, he is working as a Professor and PG Coordinator in the Department of Electrical and Electronics Engineering, Saintgits College of Engineering at Kottayam, Kerala. He is also a Fellow member in Institute of Engineers and the Institution of Electronics and Telecommunication Engineers. He is a recognized supervisor in Anna University Chennai and APJKTU, Kerala. He has presented and published several papers in conferences and journals. Most of the publications includes DC Chopper and Matrix Converter. His research interests include machine controls, electric drives, and artificial intelligence and control systems.