Genetic Algorithm for Optimal Response Time Scheduling of Electric Vehicle Model

Zouhaira Abdellaoui[1][a] and Houda Meddeb[2]
[1]*University of Tunis El-Manar1, National Engineering School of Tunis – ENIT, Communication Systems Research Laboratory SYSCOM - LR-99-ES21, Tunis-Belvédère, BP 1002, Tunisia*
[2]*University of Lorraine, es and Romain, France*
zouhaira.abdellaoui@enit.rnu.tn, meddebhouda@gmail.com

Keywords: DDS, Electric Vehicles, FlexRay, Genetic Algorithm (GA), Optimization, QOS, Suspension Model, SAE Benchmark.

Abstract: Genetic Algorithms (GAs) are widely recognized for their ability to solve complex optimization problems. Gas are an effective computational tool designed to identify optimal solutions for optimization issues in electrical vehicle. In this context, we have developed GA for optimizing the response time based on static scheduling suspension model of SAE Benchmark electric vehicles. The implemented architecture consists of multiple nodes connected via the Real- Time middleware Data Distribution Service (DDS) and the protocol FlexRay in order to benefit from their high speed and QoS.

1 INTRODUCTION

Traditional vehicles have been gradually supplanted by electric vehicles (EVs) due to a major revolution in the automotive sector (Putrus et al.,2009- Clement-Nyns and all, 2010). This progress has led to the development of sophisticated internal networks within vehicles, enhancing their overall functionality and capabilities and highlighting importance of real time electronic systems in modern vehicle design and operation.

In recent years, Genetic Algorithms (GAs) have emerged as a valuable tool in the design of electric vehicles. These algorithms are being applied to optimize several crucial parameters. GAs have proven to be powerful optimization tools, able to identify solutions for scheduling problem that enhance multiple aspects of electric vehicle performance (Chandra et al., 2022).

In this study, we adopted GAs to optimize the parameters that minimize the suspension model response time in electrical vehicle. Our architecture is based on a modern vehicle of SAE Benchmark. This approach includes multiple nodes connected through the the real-time communication protocol FlexRay and the middleware Data Distribution Service (DDS); which is elaborated in (Abdellaoui and Hasnaoui, 2019).

Data Distribution Service (DDS) is a standard middleware, developed by the Object Management Group (OMG); it is used for real-time, scalable, and high-performance data exchange between distributed systems. DDS is frequently employed in large-scale systems with many nodes that need to communicate with each other, where reliability, low latency, and efficient communication are critical (Abdellaoui and Hasnaoui, 2019). However, FlexRay is a high-speed communication real time protocol used in automotive networks. It was developed as a robust, flexibility, deterministic and fault-tolerance communication system especially for modern vehicles. It is very important in safety-critical systems.

In this work, we developed a GA to estimate optimal periods of tasks that minimize the Worst Case Response Time (WCRT) in order optimize the response time of suspension model of a modern vehicle of Society of Automotive Engineers (SAE) Benchmark; this algorithm is applied on the static scheduling method of electric vehicles. In fact, optimizing task response times in automotive electrical systems can guarantee enjoyable driving experience, performance improvement, extended

[a] https://orcid.org/0000-0003-2638-3911

Abdellaoui, Z. and Meddeb, H.
Genetic Algorithm for Optimal Response Time Scheduling of Electric Vehicle Model.
DOI: 10.5220/0013276200003928
Paper published under CC license (CC BY-NC-ND 4.0)
In *Proceedings of the 20th International Conference on Evaluation of Novel Approaches to Software Engineering (ENASE 2025)*, pages 476-483
ISBN: 978-989-758-742-9; ISSN: 2184-4895
Proceedings Copyright © 2025 by SCITEPRESS – Science and Technology Publications, Lda.

battery life (Elsayed et al., 2014). The developed architecture of GA with DDS on top of the FlexRay bus helped us to provide a data-centric infrastructure used in fault-tolerant system and automotive domain whilst taking into account QoS of DDS.
This paper is structured as follows: Section 2 discusses related works. Section 3 presents the structure design of DDS middleware QoS on FlexRay protocol network. Section 4 is dedicated for Genetic Algorithm method. However, the last section discusses simulation results of suspension model to evaluate GA performances.

2 LITERATURE REVIEW

2.1 Genetic Algorithm Approach

It is Genetic algorithms were developed by John Holland in 1975 (Holland, 1975).

In genetic algorithm (GAs) terminology, a chromosome represents an individual solution. A collection of these individuals creates a population. Each chromosome within this population is a potential solution for the problem to be solved (Chuan-Kang, 2005).The process begins by generating an initial set of potential solutions, referred to population. The algorithm then simulates natural evolution by applying selection, crossover, and mutation operators to create a new generation for offspring solutions. The solution's effectiveness and performance is evaluated using a problem-specific objective function, which quantifies how well it addresses the given challenge.

The fitness value, or objective function, of an individual fixed its chances of survival into the next generation. Achieving an optimal balance between exploitation and exploration, by adjusting crossover and mutation probabilities, can ensure high-quality offspring and accelerates convergence in optimization algorithms. In contrast, poorly considered reproduction probabilities may result in undesirable convergence to a local optimum.

This evolutionary approach allows the algorithm to iteratively improve its solutions over multiple generations. Genetic algorithms are effective for many problems, including scheduling, optimization and control.

2.2 Genetic Algorithm Applications

Genetic algorithms are considered as global search heuristics. In fact, it is a search technique process employed in computing to find solutions for optimization and search problems.

GA is applied in Real Time Systems, to generate a result which satisfies timing constraints. In (Madureira and all, 2002), authors used GA for assigning task priorities and offsets in order to guarantee real time timing constraints, running on standard Real-Time Operating System (RTOS). GAs are also applied in planning of Robot Path based on sensor under real-time unstructured environment (Yasuda and Takai, 2001). Besides, job scheduling approves again the feasibility of genetic algorithm for the resolution of real scheduling problems, which is solved using a set of static scheduling by GA (Madureira et al., 2002). The authors in (Chandra and Lalwani, 2022) implemented Genetic Algorithm for control parameters setting optimization in hybrid and parallel EVs. GA algorithm was proposed to reduce FC (Engine Fuel Consumption and emissions) using standard criteria.

3 FLEXRAY PROTOCOL AND MIDDLEWARE DDS IN AUTOMOTIVE NETWORKS

3.1 The Middleware DDS

We find different classes of middleware such as DCOM, RMI, CORBA and RPC. They provide a remote synchronous invocation method. They have typically built on top of TCP and QoS. Also, they are familiar with the OO programming model and they are considered as the most-suited to closely-coupled and smaller systems. Data Distribution Service (DDS) is) a real time middleware and an open standard managed by the Object Management Group (OMG), used as an API above operating system (OS) and peripheral drivers that resume common interaction patterns. DDS is the first general-purpose standard middleware that addresses hard real-time requirements in data-centric applications and has a large number of configuration parameters QoS which help developers to complete maintainability of object state and its control in the system. That's why; it became actually the standard in embedded systems. It dissociates the low-level architecture and design of application (software components). It isolates the design and the validation of SW-components from hardware. It allows the description of hardware architecture independently of software application. Actually, electrical vehicles have higher build complexity and software system than a commercial aircraft. This is why middleware DDS is integrated, since it can manage this complexity by reuse of software components and exchangeability (Xiao et al, 2022). DDS's DCPS (Data-Centric Publish-subscribe layer) consists of Publsiher, Subscriber, Domain-Participant, Topic, DataWriter and DataReader entities. The idea is to broadcast data directly from a publisher to all its subscribers without intermediate servers. DDS is the best middleware that can be incorporated in an electrical vehicle to interface the infrastructure low- level. In fact DDS have (Putrus et al.,2009):

- *Self-healing communication: In case the network* is repaired, the network will quickly find new nodes and will work again thanks to the built-in discovering entities
- *Support for custom fault-tolerance:* Implementations can add freely further fault tolerance such as FFT (Xiao et al, 2024)
- The support of many network interfaces like in FlexRay channels (channel-A and Channel-B) as well as redundant Data Readers and Data Writers on each node results in networks completely separate. Even in the event of a complete network failure, the system will continue to operate.
- *No single point of failure:* DDS needs «special" nodes, there for it can be implemented without a single point of failure thanks to publisher and subscriber redundancy.

3.2 Flexray Protocol

We have chosen to work within a platform of a vehicular network based on the extended SAE BENCHMARK. In this system; a set of network processors subsystems produces routing data. This data must be distributed along the vehicular network.

In fact, we have applied the studied approaches on a new vehicle benchmark developed in (Abdellaoui and Hasnaoui, 2019) and based on the SAE Benchmark. However, this Benchmark was designed to the best fit the CAN network and with major modifications and adjustments it be adapted to the FlexRay protocol.

The resulting architecture is composed of 15 nodes connected by the FlexRay bus. The main objective of this architecture that we mentioned previously is to guarantee better performance of the vehicular network and ensure safety and reliability.

In each node we find an embedded a Real-Time Operatng System and a publish/subscribe middleware; in our case we are adapting the DDS as middleware for developed and modern vehicle. That will be the best manner to validate of our vehicle system design. FlexRay Networks is one of the newest X-by Wire communication systems that offer high speed which reaches 20 bits/s thanks to its two channels.

4 SUSPENSION MODEL AS DDS ENTITIES AND STATIC SCHEDULING MODEL

4.1 DDS Entities: Suspension Model

In an electric vehicle (EV), the suspension system plays a crucial role in ensuring ride comfort, handling stability, and maximizing the efficiency of the vehicle's electric power train. Given the unique characteristics of EVs, their suspension systems often incorporate specialized designs to address challenges like battery weight, energy efficiency, and precise handling. To evaluate this methodology, we integrated DDS middleware into the vehicle Blockset. This integration enables applications to communicate by publishing data they produce and subscribing to data types they consume. The DDS middleware allows components in the vehicle system to exchange information in a publish-subscribe model, facilitating efficient data distribution between different nodes of suspension unit.

4.2 Static Scheduling Model

The Real-time networks protocol FlexRay use either dynamic or static scheduling approach for management task. In static scheduling,TDMA Time Division Multiple Access (TDMA) technique is employed (*ISO, 199).* and tasks are scheduled based on a time-triggered approach. Each message is allocated a fixed interval time, "slot," within a recurring time frame. While this approach ensures deterministic message transfer, it does not fully exploit the available network bandwidth. In the FlexRay protocol, all frames are designed with a consistent and unchanging length. This standardization ensures predictability and efficiency in data transmission where each node employs a sophisticated tracking system and maintains two separate slot counters, known as vSlotCounters for

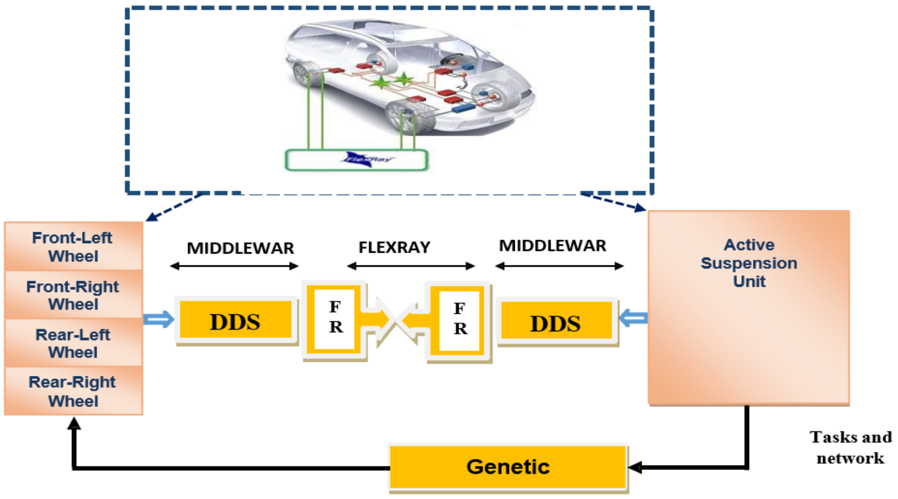

Figure 1: Application scenario: Suspension Model.

channel A and B, the two slot counters initialize at 1 and increment after each slot.Tasks in the static segment are periodic and maintain the equal priority across each communication cycle. The Worst Case Response Time (WRST) calculated based on the static scheduling model is shown below:

$$R_i = CS^1 + C_i + B_i + \left\lceil \frac{R_i}{T_{clk}} \right\rceil CT_c + \sum_{g \epsilon \Gamma_p} \left\lceil \frac{R_i}{T_g} \right\rceil CT_s \quad (1)$$

Table 1: Suspension Parameters.

CS^1	Costs associated to task switching
C_i	Worst-Case computation time for tasks i,
B_i	Worst case of blocking time for the task
T_{clk}	Clock period for a certain frequency
CT_c	Clock interrupt cost
Γ_p	Periodic tasks set
T_j	Minimum time a m o n g jobs, task releases or task periods
CT_s	Moving cost for single task from delay queue to the run queue,

5 GA FOR SUSPENSION MODEL OPTIMAL RESPONSE TIME IN EVS

In this work, genetic algorithms are employed to suspension model of electric vehicles in order to optimize the response time related to each task.

5.1 Genetic Algorithm's Architecture

The main purpose of this research is to find tasks period of Suspension model to minimize their response time. Let **T** be a decision variable represented as a column vector with size **n**, which specifies the periods for **n** tasks, and let **R** denote the response time as defined in equation **(1).** There for Suspension model can be outlined as shown in the given equations:

$$\min\ f(T) = \sum_{i=1}^{n} R_i(\mathrm{T}) \quad (2)$$

Under constraints of

$$5 \leq Ti \leq 320 \quad (3a)$$

$$Ri \geq Min\_sep \quad (3b)$$

$$Ri \leq Di \leq Ti \quad (3c)$$

In Equation (2), (T) represents the fitness of each chromosome T in the population, calculated as the total response time across all tasks. The objective is to identify a vector T* which minimizes this objective function(T*). T defined as solution space is further constrained by a set of conditions outlined in Equation (**3**). Four operators were taken into account for the development of the proposed genetic algorithm as shown in figure 2. The descriptions of these functions are:

Selection: It's a mechanism designed to enhance the probability that superior solutions will reproduce and transmit their beneficial traits to the next generation. This process enables the identification of the fittest individuals within the current population to contribute to the formation of the subsequent generation (Kinnear, 1994).

Crossover: Crossover in genetic algorithms (GAs) is a key genetic operator that combines the genetic information of two parent solutions to produce one or more offspring solutions. This mimics the biological reproduction process and helps in exploring the solution space more effectively.
Mutation: It is applied at the gene level to the chromosomes produced by the crossover operation. It involves selecting a gene as a mutation point and changing its value to a random integer in the range of [5, 320]. Here, the lower bound indicates the minimum value for the gene, while the upper one signifies its maximum value.

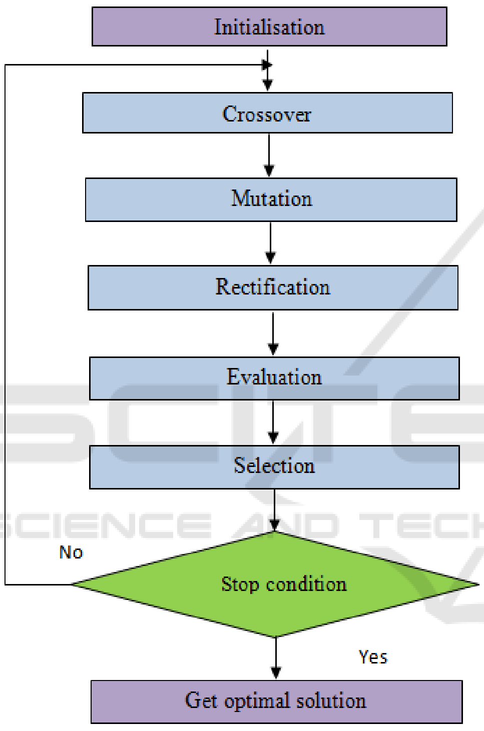

Figure 2: Genetic Algorithm Flowchart.

Rectification: An additional adjustment mechanism was required when the genetic values of an offspring do not meet last constraints outlined in equations (3). This led to the creation of a rectification function that produces a modified population, denoted as $\boldsymbol{Q}\tilde{}$. of size M which represents the set of n tasks periods. $\tilde{T}_k \in \tilde{Q}$ is the rectified individual in these conditions:

$$\text{If } T_{ki} < 5 \text{ then } \tilde{T}_{ki} = 5, \tag{4}$$

$$\text{If } T_{ki} > 320 \text{ then } \tilde{T}_{ki} = 320, \tag{5}$$

$$\text{If } T_{ki} < D_i \text{ then } \tilde{T}_{ki} = D_i, \tag{6}$$

$$\tilde{T}_{ki} = T_{ki} - mod(T_{ki}, 5) \tag{7}$$

With , $i = 1, .., n;$ $k = 1, ..., M$ and the mod is the operator which returns the remainder after division of T_{ki} by 5.

5.2 Genetic Algorithm Application Scenario

The scenario we have described, outlines a distributed system for a vehicle's active suspension control including Rear_Right_wheel, Front_Left_wheel, Rear_Left_wheel and Front_Right_wheel,using real time Data Distribution Service (DDS) middleware and the protocol FlexRay networks.
The inputs for genetic algorithm employed for our application are:

- Task parameters that involve ID, data field length, delay, deadline, clock period, interrupt handling cost and clock interrupt.
- Network parameters defined by the speed, which vaires between 5 Mbit/s, 10 Mbit/s, or 20 Mbit/s.

Its goal is to determine optimal task periods that minimize response times, there by facilitating low-latency signal transmission.

Table 2: Results of Suspension model with bus speed of 5 Mbit/s.

Tasks	Message ID	Size (byte)	T (ms)	D (ms)	Min_Sep (ms)	R (ms)
Front-leftwheel module	9	1	20	5	0.0206	0.0206
	23	2	10	10	0.0226	0.0227
Front-right wheel module	10	1	20	5	0.0206	0.0206
	24	2	10	10	0.0226	0.0227
Rear-leftwheel module	11	1	75	5	0.0206	0.0206
	25	2	10	10	0.0226	0.0227
Rear-right wheel module	12	1	15	5	0.0206	0.0206
	26	2	10	10	0.0226	0.0227
Active suspension unit	27	2	10	10	0.0226	0.0227

Table 3: Results of Suspension model with bus speed of 10 Mbit/s.

Tasks	Message ID	Size(byte)	T(ms)	Delay(ms)	Min separation (ms)	R(ms)
Front-leftwheel module	9	1	140	5	0.0103	0.0105
	23	2	125	10	0.0113	0.0115
Front-right wheel module	10	1	20	5	0.0103	0.0105
	24	2	20	10	0.0113	0.0115
Rear-leftwheel module	11	1	140	5	0.0103	0.0105
	25	2	130	10	0.0113	0.0115
Rear-right wheel module	12	1	20	5	0.0103	0.0105
	26	2	20	10	0.0113	0.0115
Active suspension unit	27	2	10	10	0.0113	0.0115

Table 4: Results of Suspension model with bus speed of 20 Mbit/s.

Tasks	Message ID	Size(byte)	T(ms)	D(ms)	Min_Sep (ms)	R(ms)
Front-leftwheel module	9	1	75	5	0.0052	0.0053
	23	2	45	10	0.0057	0.0058
Front-right wheel module	10	1	160	5	0.0052	0.0053
	24	2	270	10	0.0057	0.0058
Rear-leftwheel module	11	1	5	5	0.0052	0.0053
	25	2	45	10	0.0057	0.0058
Rear-right wheel module	12	1	235	5	0.0052	0.0053
	26	2	105	10	0.0057	0.0058
Active suspension unit	27	2	10	10	0.0057	0.0058

6 EVALUATION OF SUSPENSION MODEL SIMULATION RESULTS

For this evaluation, our application is presented by the standard SAE Benchmark model using middleware DDS, FlexRay real-time protocol as communication support and GA to estimate the worst-case response time (WCRT) for communication tasks across suspension's nodes.

6.1 Suspension Simulation Results

The genetic algorithm was performed with a population size of 30 and 300 iterations. The results are displayed in the following tables.

The results confirm that DDS Quality of Service (QoS) requirements are effectively met, especially those concerning deadline and minimum separation, while accounting for latencies introduced by the FlexRay. The objective function evolution for the suspension node is illustrated in figure 3.

6.2 GA Performance Evaluation with Suspension Model

To evaluate the GA's performance, tests were conducted with different population sizes and bus speeds. The table shows the results for the suspension-model.

Table 5 reveals that convergence time tends to increases as the population size grows. Moreover, the proposed GA occasionally converges to a local optimum. However, a population size of 30 yielded the global optimum. Cost or objective function evolution for suspension model is illustrated in figure Figure 4.

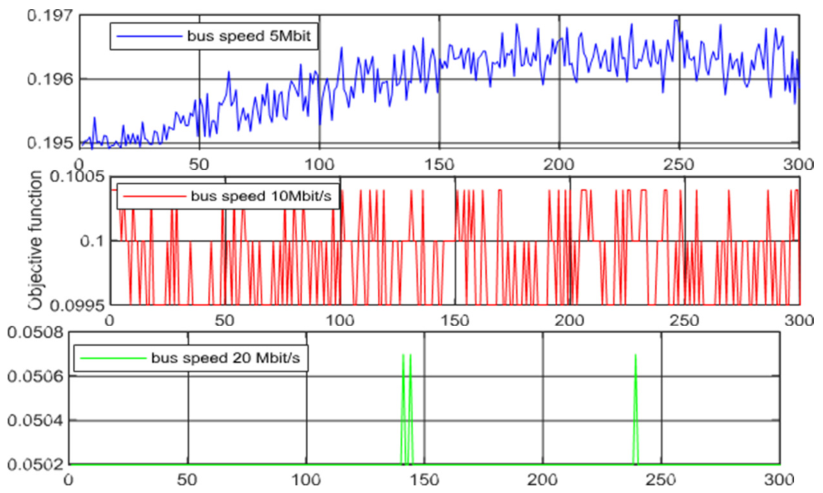

Figure 3: Objective function Evolution for Suspension model.

Table 5: Algorithm Performance Results in function of bus speed and population size.

Bus speed 5Mb/s et iteration 300						
Population	10	30	50	70	100	120
Convergence time	10.31	59.99	80.93	168.15	312.43	451.31
Objective function	0.1962	0.1959	0.1966	0.1963	0.1961	0.1961
Bus speed 10Mb/s						
Population	10	30	50	70	100	120
Convergence time	8.71	52.59	123.81	220.65	325.74	470.29
Objective function	0.0995	0.0995	0.1004	0.1004	0.1004	0.1000
Bus speed 20Mb/s						
Population	10	30	50	70	100	120
Convergence time	10.62	37.06	67.50	136.54	237.65	375.69
Objective function	0.0511	0.0502	0.0502	0.0502	0.0511	0.0511

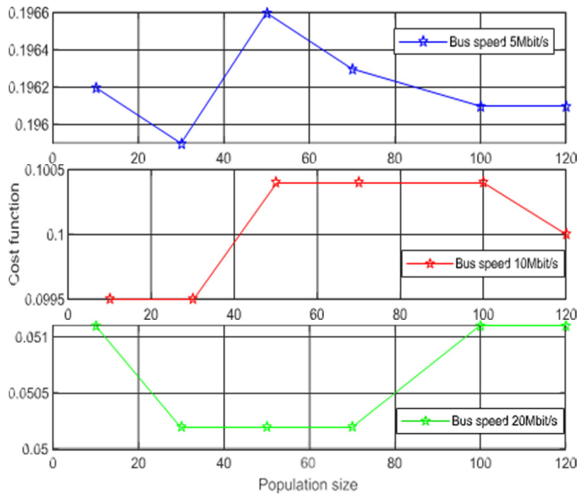

Figure 4: Objective function evolution for Suspension model.

In a second scenario, we defined the stopping criterion as the verification of constraint (3a). The algorithm halts once it finds a solution that satisfies this constraint. The adopted algorithm was implemented for suspension nodes, and the results for a bus speed of 5 Mb/s are presented in table 6:

Table 6: Performance of GA in function of population size.

Bus speed 5Mb/s					
Population	10	30	50	70	100
Convergence Time	3.43	10.12	24.07	67.41	99.10
Objective function	0.1963	0.1963	0.1963	0.1963	0.1963
Iteration number	123	90	100	127	114

7 CONCLUSION

This work introduces a novel approach of genetic algorithm based on the static scheduling model of SAE Benchmark electric vehicle EVs. The design integrates the Real-Time protocol FlexRay with Data Distribution Service middleware.

GA approach is applied on the Suspension model to identify the optimal task periods which led to minimize the required response time.

A performance evaluation is conducted to validate the efficiency of the proposed approach in the design of EVs.

ACKNOWLEDGMENTS

We wish to express our gratitude to the SYSCOM ENIT members for their help and assistance.

REFERENCES

Putrus G. A., Suwanapingkarl P., Johnston D., Bentley E. C., and Narayana M. (*2009*). “Impact of Electric Vehicles on Power Distribution Networks”, *in Proc. IEEE Vehicle Power and Propulsion Conf., pp. 827-831.*

Clement-Nyns K., Haesen E., and Driesen J. (2010). “The Impact of Charging Plug-In Hybrid Electric Vehicles on a Residential Distribution Grid”, *IEEE Trans. Power Systems, vol. 25, pp. 371-380.*

Chandra A., Lalwani J. and Jajodia B. (2022). “Towards an Optimal Hybrid Algorithm for EV Charging Stations Placement using Quantum Annealing and Genetic Algorithms,”*International Conference on Trends in Quantum Computing and Emerging Business Technologies (TQCEBT).*

Abdellaoui Z. and Hasnaoui S. (2019). “DDS Middleware On top of FlexRay Networks: Simulink Blockset Implementation of Electrical Vehicle and its Adaptation to DDS concept using the FlexRay Protocol’’, *International Journal of Soft Computing, Soft Computing Volume 23, pages11539–11556.*

Elsayed S. M., Sarker R. A., and Essam D. L. (2014), “A new genetic algorithm for solving optimization problems,” *Engineering Applications of Artificial Intelligence, vol. 27, pp. 57–69.*

Holland J. H. (1975). Adaptation in natural and artificial systems. *Ann Arbor: University of Michigan Press.*

Ting, Chuan-Kang (2005). “On the Mean Convergence Time of Multi-parent Genetic Algorithms without Selection”. *Advances in Artificial Life: 403–412. ISBN 978-3-540-28848*-0.

Madureira A., Ramos C., do Carmo Silva S. (*2002*). “A Coordination Mechanism for Real World Scheduling Problems using Genetic algorithms”, *Evolutionary Computation, 2002. CEC ’02. Proceedings of the 2002, Congress on, 1, pp 175 –180.*

Yasuda, G. and Takai, H. (2001). “Sensor-based Path Planning and Intelligent Steering Control of Nonholonomic Mobile Robots”, *Industrial Electronics Society, 2001. IECON ’01. The 27th Annual Conference of the IEEE, 1,Page(s): 317 -322,1.*

Chandra A., Lalwani J. and Jajodia B. (2022), “Towards an Optimal Hybrid Algorithm for EV Charging Stations Placement using Quantum Annealing and Genetic Algorithms,”*International Conference on Trends in Quantum Computing and Emerging Business Technologies (TQCEBT)*

Xiao H., et al. (2022)."Edge Intelligence: A Computational Task Offloading Scheme for Dependent IoT Application," *in IEEE Transactions on Wireless Communications, vol. 21, no. 9, pp. 7222-7237*, doi: 10.1109/TWC.2022.3156905.

Xiao H., et al. (2024). “VAAC-IM: Viewing Area Adaptive Control in Immersive Media Transmission”. *In Proceedings of the 34th edition of the Workshop on Network and Operating System Support for Digital Audio and Video (NOSSDAV '24). Association for Computing Machinery, New York, NY, USA, 8–14. https://doi.org/10.1145/3651863.3651877.*

International Standard Organization (1994), ISO 11519-2, Road Vehicles - *Low Speed serial data communication- Part 2: Low Speed Controller Area Network, ISO.*

Kinnear K. E. (1994).’’A Perspective on the Work’’in *this Book. In K. E. Kinnear (Ed.), Advances in Genetic Programming (pp. 3-17). Cambridge: MIT Press.*