sensors

MDPI

Article

Energy Efficiency Optimisation of Joint Computational Task Offloading and Resource Allocation Using Particle Swarm Optimisation Approach in Vehicular Edge Networks

Amjad Alam *, Purav Shah, Ramona Trestian, Kamran Ali and Glenford Mapp

Faculty of Science and Technology, Middlesex University London, The Burroughs, London NW4 4BT, UK; p.shah@mdx.ac.uk (P.S.); r.trestian@mdx.ac.uk (R.T.); k.ali@mdx.ac.uk (K.A.); g.mapp@mdx.ac.uk (G.M.)
* Correspondence: aa4423@live.mdx.ac.uk

Abstract: With the progression of smart vehicles, i.e., connected autonomous vehicles (CAVs), and wireless technologies, there has been an increased need for substantial computational operations for tasks such as path planning, scene recognition, and vision-based object detection. Managing these intensive computational applications is concerned with significant energy consumption. Hence, for this article, a low-cost and sustainable solution using computational offloading and efficient resource allocation at edge devices within the Internet of Vehicles (IoV) framework has been utilised. To address the quality of service (QoS) among vehicles, a trade-off between energy consumption and computational time has been taken into consideration while deciding on the offloading process and resource allocation. The offloading process has been assigned at a minimum wireless resource block level to adapt to the beyond 5G (B5G) network. The novel approach of joint optimisation of computational resources and task offloading decisions uses the meta-heuristic particle swarm optimisation (PSO) algorithm and decision analysis (DA) to find the near-optimal solution. Subsequently, a comparison is made with other proposed algorithms, namely CTORA, CODO, and Heuristics, in terms of computational efficiency and latency. The performance analysis reveals that the numerical results outperform existing algorithms, demonstrating an 8% and a 5% increase in energy efficiency.

Keywords: energy efficiency; meta-heuristic algorithm; vehicular edge computing; particle swarm optimisation; nature-inspired algorithm; task offloading; computation resource allocation; vehicular edge computing

Citation: Alam, A.; Shah, P.; Trestian, R.; Ali, K.; Mapp, G. Energy Efficiency Optimisation of Joint Computational Task Offloading and Resource Allocation Using Particle Swarm Optimisation Approach in Vehicular Edge Networks. *Sensors* **2024**, *24*, 3001. https://doi.org/10.3390/s24103001

Academic Editors: Omprakash Kaiwartya and Tomás Mateo Sanguino

Received: 28 March 2024
Revised: 3 May 2024
Accepted: 8 May 2024
Published: 9 May 2024

1. Introduction

With the rapid development of wireless technologies and the Internet of Things (IoT) in the field of vehicular networks, there is an increasing need for computational operations to process large amounts of data collected through sensors and other communication systems in connected and autonomous vehicles (CAVs) [1]. These computational operations are required by the vehicles' onboard systems to manage complex tasks such as route planning, immersive gaming, and vision-based object detection [2]. In addition to the computationally demanding requirements, these applications have significant energy consumption and are delay-sensitive [3,4]. This lowers the overall mileage endurance of the vehicle and affects the quality of service (QoS) as well. However, it is quite tough to fulfil the computational demand of CAVs due to their limited onboard computational power and energy presence.

Offloading the tasks to computational offloading technologies like vehicle cloud computing (VCC), vehicular fog computing (VFC), and vehicular edge computing (VEC) is now an alternative approach to solving the above issues [5]. Although cloud computing infrastructure has been around for a while, resource-intensive applications still have drawbacks, which include costly bandwidth problems, increased latency, and jitter. VFC and VEC computing are two examples that are employed to the edge idea to bring cloud-like resources closer to users. VFC uses adjacent cars as its computing node to determine

the best resources [6]. However, a few limitations with fog computing include resource constraints, problems with mobility, and changeable network circumstances [7]. VFC has challenges like coping with optimal performance across a diverse set of vehicles due to the heterogeneity of the vehicles, as it lacks standardisation and interoperability. VFC also has security and privacy concerns due to the exchange of sensitive data between vehicles and fog nodes [8].

Multi-access edge computing in a vehicular environment (i.e., vehicular edge computing) has now been used to achieve the demands of low-latency data transmission and computational activities. This is typically seen in base stations that are positioned close to edge devices [9]. VEC works in a wider range of traffic and mobility scenarios. Hence, the energy optimisation issue can be addressed by using computational offloading technology using VEC computing where tasks are being offloaded for computation toward the edge servers [4,10]. To regulate resource allocation and the trade-offs between energy efficiency and delay management, this article considers the VEC approach. VEC has been taken into consideration as part of a four-layer system framework to enable granular control over VEC functions, including perception, storing, processing, and communication. The minimum assignable resource block (RB) has been taken into consideration to increase the allocation efficiency and resource management performance, as wireless communication plays a major role in the system's quality of service. In addition, the mobility rate has been considered as a reference variable for this article to address the spectrum's drift taking place under different moving speeds.

1.1. Contributions

The key contributions of this article are listed as follows.

1. A novel four-layer VEC framework is proposed that facilitates more granular control over real-time computation, storage, compatibility, and interconnection of the HetVNets. The use of a four-layer VEC framework will support scalability and would provide better vehicle-related system resources.
2. This article aims to increase overall system energy efficiency by maximising resource allocation and task offloading considering energy constraints.
3. The PSO algorithm has been used for resource allocation and decision analysis (DA) has been used in making optimal decisions towards task offloading. The results has been compared against CTORA, CODA, and Huestristics.

1.2. Paper Organisation

The rest of the paper is organised as follows: Section 3 presents the four-layered VEC system framework and the vehicular computation offloading model in VEC (system model). The resource allocation problem formulation and methodology are presented in Section 4. Section 5 illustrates the proposed task offloading and resource allocation scheme. Section 6 presents a comprehensive set of simulation data and interpretation of results, with Section 7 concluding our findings and presenting future directions.

2. Related Works

There has been a significant amount of research on task offloading and VEC in recent years within the context of heterogeneous vehicle networks (HetVNets). The aim is to maximise resource utilisation, enhance performance, and achieve energy efficiency. In study [11], two heuristic-distributed and context-aware task offloading approaches (random and exhaustive) have been modelled in order to manage the delay. An online task scheduling method was employed for effective tasks in the edge cloud in [12]. This model was proposed to reduce the communication delay pool. In another study presented in [13], the allocation of computational resources was based on VEC and combined the efficient use of fifth-generation (5G) and short-range communication. In [14], joint optimisation of resource allocation and load balancing was considered in a multi-server multi-user vehicle network. In [15,16], multi-objective optimisation was used to reduce the energy

consumption of edge devices and the execution time of computational tasks while preserving privacy. In the work presented in [17], a reinforcement learning-based scheme was implemented on Edge Cloud to find optimal routes for task offloading. In a study from [18], weighted energy consumption was considered in optimising task loading for mobile users. It examined Orthogonal Frequency Division Multiple (OFDM) access and Time Division Multiple (TDM) access with resource allocation on Mobile Edge Computing (MEC). Ref. [19] focused on utilising the mobile edge server by proposing a contract-based computational resource allocation and task offloading. In another study [20], an optimal portion of the workload is considered for offloading, taking factors like workload execution, data transmission, and latency into account. Task offloading and resource allocation are subjects of consideration in many edge computing research studies that address energy efficiency. In [21], optimised resource allocation between the cloud and fog to minimise energy use under different latency constraints. The emphasis on profit optimisation for edge-cloud service providers was presented in [22], where the maximum response time limit and service-level agreements to estimate revenue and penalty costs for each activity were considered. While the above studies focus on low-latency networks and address high-reliability issues such as energy saving, limited onboard vehicle energy has not been thoroughly addressed. In [23], Chang et al. proposed a computational offloading decision optimisation (CODO) to determine the optimal portion of workload to be offloaded based on the dynamic states of energy consumption and latency in workload execution; however, the handover issue was not addressed.

Incorporating cellular and wireless technology, [24] proposed a hybrid VEC in a 5G network for real-time traffic management to maximise the total offloading rate. They addressed a joint power allocation problem, subchannel assignment, and joint task distribution. Ref. [25] presented a low-complexity online algorithm that concurrently determines CPU-cycle frequencies for mobile execution, transmission power for computation offloading, and offloading decisions. The primary objective was to minimise the long-term average execution cost in MEC. The study presented in [26] on downlink spectrum resource management for VEC considered transmission power management among WiFi access points, resource allocation among vehicles, and spectrum slicing in base stations. In some of the above-mentioned studies (e.g., [3,13,14]), the focus was on reducing vehicle energy consumption alongside computational offloading. In other studies (e.g., [12,16,18]), the emphasis was on latency management along with the task offloading issue, without concerns for vehicle energy efficiency. Many studies employed a centralised optimisation method as a prior solution, leading to an issue where the computational complexity dramatically increases with the number of vehicles. This complexity issue can be addressed by adopting a distributed approach to manage energy efficiency and latency. In [27], Fan et al. proposed a joint computational task offloading and resource allocation scheme (CTORA) to minimise the total task processing delay through task scheduling, channel allocation, and computing resource allocation for the vehicles and RSU; however, the work did not make use of 5G.

In [28], a distributed context-aware assignment of tasks is being considered on vehicular networks using a heuristic algorithm to minimise delay. The article [29] combined convolutional neural networks (CNN) with proximal policy optimisation to provide a workload offloading method. They considered tasks lacking strict latency requirements or execution priorities. Ref. [30] examined a task offloading problem involving parked cars acting as servers, using blockchain for decentralised offloading. They proposed and solved this problem using the game system to minimise users' overall payments. In [31], a Bayesian coalition game to improve energy efficiency and computing resource utilisation in a vehicle cloud was presented. Ref. [32] took into consideration the task offloading issue to reduce the edge server's communication load. They applied game theory to choose appropriate channels and select the best offloading strategies.

However, some earlier research examined approaches to optimise work offloading or computing resource allocation without optimising both at the same time. For example, the studies reported in [28,29] only looked at task offloading; they neglected to consider com-

puting resource allocation, even though each vehicle was frequently given a variety of computationally demanding real-time tasks. Moreover, task offloading optimisation—which entails unloading the entire work to the MEC server—was ignored in the research in [30,31]. Each vehicle that engages in task offloading chooses whether to unload and where its work will be processed. As a result, vehicles share constrained computing and communication resources. To improve system performance, the job offloading and resource allocation strategies must be optimised. Furthermore, most studies on vehicle task offloading ignore an important component of task offloading and resource allocation with stringent latency limits and energy requirements. We developed a multi-vehicle task offloading game that takes vehicle movement into account in addition to task deadlines and energy consumption constraints, which sets it apart from earlier task offloading techniques for VEC.

Overall, the main research gaps identified are represented below in Table 1.

Table 1. Research gaps.

Works Cited	Contribution	Gaps
[12,16,18]	Emphasis was on latency management along with the task offloading issue	No concern for vehicle energy efficiency
[24]	Proposed a hybrid VEC in a 5G network to maximise the total offloading rate	No concern for joint optimisation of resource allocation
[25,26]	Efficient task offloading employed and resource allocation	No joint optimisation on task offloading and resource allocation
[27,28]	Optimisation of task offloading and resource allocation	Optimisation was focused only on MEC server

In this article, we propose using a meta-heuristic Particle Bee Colony Swarm Optimisation (PSO) and decision analysis (DA) algorithm to minimise the overall energy consumption by jointly optimising the computational task offloading and resource allocation algorithm.

We have analysed the computation efficiency problem for CAVs by making an optimised decision on allocating resources and deciding where to upload the tasks. However, designing an efficient offloading approach is tough due to the highly dynamic scenario. Along with the mobility factor, other factors like the required CPU cycle, task data size, and energy consumption will impact the transmission rate and computation efficiency drastically. In this article, computation efficiency has been used as a performance metric which is the ratio of computed bits to the total energy consumed.

We take into considerations the previously mentioned CTORA and CODO for comparison.

3. VEC System Framework and System Model

Figure 1 below illustrates the 4 layers of the VEC system architecture, i.e., the perception layer, processing layer, transport layer, and application layer.

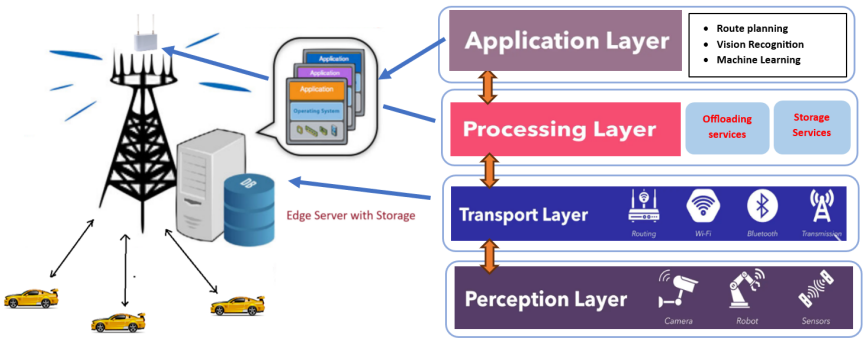

Figure 1. VEC system framework.

In Figure 1, the perception layer has been developed to include two types of sensors: an external and an internal system. The internal sensors of CAVs include cameras, millimetre wave radar, Lidar, and other devices, which are the primary focus of current

technology. The external sensors, on the other hand, provide extended sensor information from neighbouring vehicles, infrastructure sensors, and the Internet data. The situation awareness capabilities of this layer will aid CAV in planning and decision-making [33]. The communication with all types of vehicles on the road and other RSU units is supported by the transport layer, which is the second layer of our 4-layer approach. The processing layer consists of a storage system, computation offloading service strategy, and decision system. It is primarily responsible for the gathering, processing, and offloading of computations. Intense applications such as intelligent traffic signal management, route planning, and other real-time vehicular onboard Virtual Reality/Augmented Reality (VR/AR), as well as driver behaviour recognition, which can offer immersive services for human–vehicle interactions, are managed by the top application layer.

For vehicle-to-infrastructure (V2I) communication, we consider that each RSU and vehicle can support both cellular (5G-NR-V2X) and millimetre wave-based communication systems. Each RSU and vehicle is equipped with multiple antennas to enable communication over 5G links and mmWave as shown in the system model represented in Figure 2. The communication speed depends on the distance between the RSU and the vehicle. Using mmWave-based V2X within the range of 300 m, a throughput of up to 10 Gb/s can be achieved. Therefore, we have considered a cellular link range to 200 m and mmWave communication range to 150 m [34].

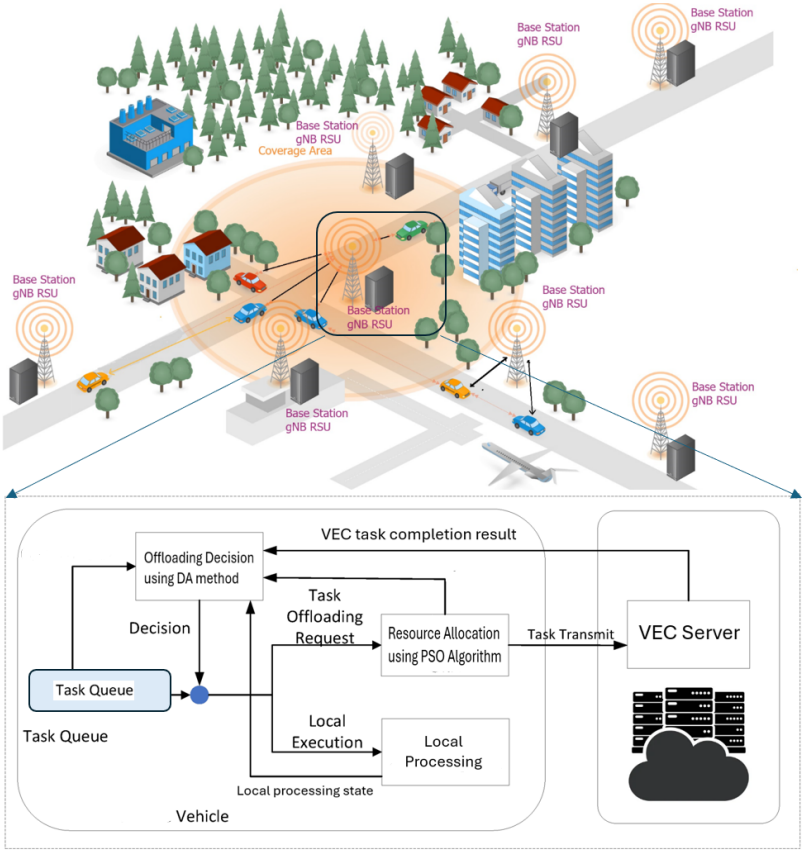

Figure 2. System model.

3.1. System Topology

In the proposed network we have taken n number of vehicles and m number of roadside units (RSU's) in a unidirectional road. The coverage or communication range for all the RSUs has been considered as r. Each RSU is integrated with an edge server consisting of R_{edge} computing resources. The vertical distance between the RSU and the road has been considered as v. The sets of vehicles have been represented as $X = 1, 2, 3, \ldots, n$ and the set of RSU's is represented as $Y = 1, 2, 3, \ldots, m$. Here, we need to consider that

each vehicle has some computation tasks that need to be either offloaded to the VEC server or should be computed locally. The offloading decision set is represented here as $A = loc, vec$. Any vehicle $n \in X$ will connect to RSU $m \in Y$, provided that the vehicle is in the coverage region of the RSU. In this article, the vehicular offloading strategy is defined as S = $s_i \mid s_i \in [\, s_i^{loc}, s_i^{vec}\,]$, $s_i^k \in 0, 1, : i \in X, j \in$ [loc, vec].

3.2. Mobility Model

Since the vehicle speed will change dynamically with time along the road, every vehicle has been assigned a random speed vs. chosen from the Gaussian probability density function. Due to the practical nature of traffic, the velocities are bounded away from zero and cannot be negative such as in a congested traffic where vehicles can stop due to traffic signals. Free-flow traffic is considered. As such, a truncated Gaussian probability density function (PDF) applied using the below formula (Equation (1)) with $V \epsilon (V_{min}, V_{max})$ where $V_{min} = \mu - 3\sigma$ and $V_{max} = \mu + 3\sigma$;

$$\hat{f}_V(v) = \frac{2 f_V(v)}{Err(\frac{V_{max}-\mu}{\sigma\sqrt{2}}) - Err(\frac{V_{min}-\mu}{\sigma\sqrt{2}})} \tag{1}$$

where $f_V(v) = [1/(\sigma\sqrt{2\pi})] exp(-(v-\mu)^2/(2\sigma^2))$ which is the Gaussian PDF. $Err()$ is the error function, σ is defined as a standard deviation of vehicular speed, μ is the average speed, V_{max} is the maximum velocity, V_{min} is the minimum velocity, and v is the random chosen velocity of the vehicle [35].

From Equation (1), a corresponding speed (μ_v) has been derived, and it lies between (V_{min}, V_{max}), i.e., $V_{min} \leq \mu_v \leq V_{max}$ [35].

$$\mu_v = \frac{Err(\frac{V_{max}-\mu}{\sigma\sqrt{2}}) - Err(\frac{V_{min}-\mu}{\sigma\sqrt{2}})}{\frac{2}{\sigma\sqrt{2\pi}} \int_{V_{min}}^{V_{max}} \frac{exp(-\frac{(v-\mu)^2}{2\sigma^2})}{v} dv} \tag{2}$$

where $Err()$ is the error function, σ is defined as a standard deviation of vehicular speed, μ is the average speed, V_{max} is the maximum velocity, and V_{min} is the minimum velocity [35].

The task must be completed before the vehicle leaves the connected RSU and moves to the next RSU on the road. Hence, it is important to know the vehicle's duration of stay within the connected RSU coverage area. Therefore, the vehicle stay time can be expressed as

$$t_i^{hold} = \frac{2\sqrt{r^2 - p^2}}{v_i} \tag{3}$$

where p is the vertical distance between the road and RSU, the communication radius of its connected RSU is r, and v_i is the velocity of a vehicle.

3.3. Communication Model

mmWave Mode: Each RSU and vehicle is assumed to be installed with directional antenna to have antenna gain. Hence, the transmission rate of vehicle U_i' can be expressed as

$$U_i^{'} = B_{mm} log_2(1 + SNR_{i,j}) \tag{4}$$

where $SNR_{i,j}$ is the SNR between associated RSU $j \in Y$ in the mmWave mode and the vehicles $i \in X$ and is outlined below in Equation (5) [36].

$$SNR_{i,j} = (P_{i,j} - 10log_{10}(B_{mm}) - 10\lambda log_{10}([r\frac{L_i}{r} - L_i])) - SF_\alpha - 69.6 - a_{mm} + A_i^{max} A_j^{max} \tag{5}$$

where $P_{i,j}$ is vehicle i's transmission power over its corresponding RSU, B_{mm} is the mmWave channel, λ is the path loss exponent, $[r\frac{L_i}{r} - L_i]$ is the distance travelled by vehicle, L_i denotes

the current position of $vechicle_i$, r is the communication radius of its connected RSU, SF_α is the shadow fading which has been set to 4 dB in line of sight (LOS), A_i^{max} is the $vehicle_i$ antenna gain, A_j^{max} is the RSU_j antenna gain, and a_{mm} is the Gaussian noise [37,38].

Cellular Mode: In V2I communication, the cellular link is under NR-V2X. More demanding QoS requirements than those supplied by Cellular V2X can be met by sophisticated V2X applications that NR-V2X can support. Within 5G technology, NR-V2X guarantees enhanced performance in relation to throughput, latency, dependability, connection, and mobility [39].

The data transmission rate between vehicle $i \in X$ and the RSU $j \in X$ m is derived as

$$U_i^{'} = B_{cc} log_2(1 + \frac{P_{i,j} * d * |\Omega|^2}{g_{cc}^2}) \tag{6}$$

where d is

$$d = (r(\frac{L_i}{r} - L_i)^{-\theta}) \tag{7}$$

B_{cc} is the cellular channel bandwidth, $P_{i,j}$ is the transmission power of the vehicle $i \in X$, $j \in Y$, L_i represent the vehicle's current position, $g_{cc}^2$is the Gaussian noise, θ is the path loss, and $|\Omega|^2$ is the uplink channel fading coefficient [40].

3.4. Computational Model

We can assume that each vehicle will have a computational task along with a maximum acceptable delay, i.e., $T_i = C_i, d_i, t_i^{max}$, where $C_i = d_i * K_i$; d_i represents the size of the data block, K_i represents the service co-efficient of the vehicle, C_i is the computational resource required to complete the task T_i, and t_i^{max} represents the maximum acceptable delay for that task. The threshold acceptable delay should be less than the stay time of the vehicle within the coverage area of the connected RSU. Hence, the acceptable delay will be min $[t_i^{max}, t_i^{hold}]$.

1. Local computational time and energy consumption: If the vehicles need to compute the task locally, then the time taken, and the energy consumed to complete the task are presented below.

$$t_i^{local} = \frac{C_i}{\alpha_i^{local}} \tag{8}$$

and

$$e_i^{local} = C_i * \tau_i \tag{9}$$

where α_i^{local} is the vehicle own computing resource, and τ_i represents the energy consumed per task.

2. Time and energy consumption on the VEC: The vehicles need to offload the task to VEC, if the local computation is not feasible. Here, we need to add the uplink transmission time along with VEC execution time. Hence the time taken and the energy consumed to complete the task on VEC are presented below.

$$t_i^{VEC} = \frac{C_i}{\alpha_i^{VEC}} + \frac{d_i}{U_i} \tag{10}$$

where α_i^{VEC} is the computational resource assigned to any connected vehicle by VEC, and U_i is the vehicle upload data transmission rate.

The total energy consumed in the system (E_i), i.e., energy consumed by the vehicle i ∈ X, while offloading the task to VEC and energy consumed on VEC while executing the assigned task by the vehicle i.

$$E_i = e_i^{local} + p_i * \frac{d_i}{U_i} + \Delta E_{loss} \tag{11}$$

where P_i represents the average transmission power of $vehicle_i$, and ΔE_{loss} is the energy lost due to the multipath fading effect.

3. Energy time tradeoff (ETT): In optimisation or decision-making situations, ETT is defined as the weighted sum of energy consumption and task execution time to reflect the tradeoff between the utilisation of energy resources and the time to accomplish a specific objective, i.e., the vehicle's task computation requirement. Hence, the ETT for any vehicle i for local computation can be represented as

$$ETT_i^{local} = \beta * e_i^{local} + \gamma * t_i^{local}; \quad ETT\ for\ local\ computing \tag{12}$$

$$ETT_i^{VEC} = \beta * e_i^{VEC} + \gamma * t_i^{VEC}; \quad ETT\ for\ VEC\ computing \tag{13}$$

where β and γ indicate weights of task executing time and energy consumption for vehicle i such that $0 \leq \beta \leq 1$ and $0 \leq \gamma \leq 1$.

In the above ETT for local and VEC computing, the values of β and γ can be selected by the vehicle according to the priority needed on energy or time. For example, vehicles can select a higher β value, if they have a higher energy priority or can select a higher γ value if they have a higher time priority.

In addition to ETT for local and VEC computing, we also need to find max(ETT_i^{local}) which is the task's maximum energy consumption and tolerable delay, if executed locally. We also need to find max(ETT_i^{VEC}) which is the task's maximum energy consumption and tolerable delay on VEC, if executed on VEC.

$$max(ETT_i^{local}) = \beta * e_i^{max(local)} + \gamma * t_i^{max(local)}; \tag{14}$$

$$max(ETT_i^{VEC}) = \beta * e_i^{max(VEC)} + \gamma * t_i^{md}; \tag{15}$$

where t_i^{md} is the maximum manageable or acceptable delay

4. Offloading decision-making function: The vehicle needs to decide whether the tasks should use its local computational resources or to offload to VEC. For making this decision, the below functions are used to guarantee the execution within the maximum acceptable delay.

(1) Decision Function for local execution: The local computing decision function DF_i^{local} for a vehicle i can be represented as

$$D_i^{Local} = ln(1 + max(max(ETT_i^{local}) - ETT_i^{local}) - \psi b(max(ETT_i^{local}) \tag{16}$$

provided that $max(ETT_i^{local}) < ETT_i^{local}$, where ψ is used to keep the condition satisfied that ETF_i^{local} should be less than $s_i^{VEC} * ETT_i^{VEC}$. The function b(.) is used as a boolean function that will either return 0 if $max(ETF_i^{local}) < ETT_i^{local}$ is false, or it is equal to 1, if it is true.

(2) Decision Function for VEC execution: The VEC computing decision function for a vehicle i can be represented as DF_i^{VEC} for a vehicle i

$$D_i^{VEC} = \mu_i * ln(1 + max(max(ETT_i^{VEC}) - ETT_i^{VEC})) - (1 - \mu_i) * \zeta_{vec} * \alpha_i^{VEC} \tag{17}$$

where ζ_{vec} is the unit per computing cost on VEC, μ_i is the weight coefficient of the decision function, and α_i^{VEC} is the resource allocation on VEC for a vehicle [41].

4. Problem Formulation and Methodology

In this section, we need to frame the total computational efficiency which is defined as the ratio of the total computed bits to the vehicle's energy usage. The total computation efficiency (TE) of the whole system can be framed as

$$TE = \sum_{i=1}^{X} \sum_{j=1}^{Y} \frac{DF_i^j * S_i^j * T_i}{E_i} \tag{18}$$

We now need to address energy efficiency and computational enhancement of the whole system. The goal of optimisation is to minimise the total energy consumption of the system by maximising the total efficiency function (TE) in Equation (16). In order to achieve this, we create an optimisation problem that maximises the system's utility by optimising resource allocation and the task offloading technique which can be expressed as

$$\max(\text{TE}) \tag{19}$$

provided that the following requirements are met such that

$$s_i^{VEC}(max(ETT_i^{VEC})) + s_i^{local}(max(ETT_i^{local})) \geq s_i^j * ETT_i^j \tag{20}$$

R1: $\sum_{i=1}^{X} \alpha_i^{vec} \leq R_i^{VEC}, \quad \forall_i \in X$
R2: $s_i^{local} + s_i^{VEC} \leq 0$ where R = $\{\alpha_1^{vec}, \alpha_2^{vec}, \alpha_3^{vec}, \alpha_4^{vec} \ldots\ldots \alpha_X^{vec}\}$ is all the resource allocation on VEC, and S = $\{s_1, s_2, s_3 \ldots s_n\}$ is the execution vehicle offloading strategy.

As the above Functions (17) and (18) involves sum-of-ratio maximisation, the Particle Swarm Optimisation (PSO) algorithm [42] has been used for computation resource allocation. The decision analysis (DA) algorithm has been used for the vehicles' offloading decisions. Once the offloading decisions take place, then the optimisation of computation resource allocation on VEC takes place using the Particle Swarm Optimisation (PSO) algorithm.

After the resource allocation stage, the DA algorithm modifies the task offloading mechanisms until an optimal point is reached. To maximise the utility of each offloading vehicle, the computation resource allocation of VEC computing must be optimised once all vehicles have made their offloading decisions. The system reaches the nearly ideal solution through mutual iteration and reaches a steady state.

Particle Swarm Optimisation (PSO) Algorithm

Particle Swarm Optimisation (PSO) is a computational approach used in computer science to optimise a problem by repeatedly attempting to improve a candidate solution in relation to a specified quality metric. To obtain an improved solution, the particle swarm optimisation technique uses a cluster of particles. By using a population of potential solutions, referred to as particles in this instance, and manipulating their position and velocity inside the search space, it solves the problem. Using basic formulas, these particles are shifted around in the search space. Both the best-known position of the entire swarm and the particles themselves serve as guides for their travels in the search space. These will eventually start to direct the swarm's motions as better sites are found. A workable solution is eventually found if the procedure is repeated [42]. Additionally, in this method, N particles are initialised by the population, and each particle has a unique position a, velocity v_a, and personal best position P_{best_i}.

As stated above, every particle updates its position and velocity, particularly by learning from g_{best_i} and P_{best_i}, the global and personal best positions. The new positions can be represented by the following equation.

$$\begin{cases} v_{a^{new}} = \delta * (v_{a^{old}}, x_{a^{old}}, p_{best_i}, g_{best_i}) \\ x_{a^{new}} = x_{a^{old}} + v_{a^{new}} \end{cases} \tag{21}$$

From the above Equation, $x_{a^{new}}$ and v_a^{new} indicate the position and new velocity of the ath particle in the present iteration where δ represents the velocity updating approach in PSO. In the next equation, x_a^{old} and v_a^{old} are considered as the position and velocity of the old ath particle in the preceding iterations. It was noticed that if x_a^{new} was better than the previous

old position, then the new best position is replaced by x_a^{new}. Consequently, for particles that find a better position, their counts are reset, whereas for the particles that fail to update p_{best_i}, their count will be significantly increased.

In the next step for performing a better search, the particles with better fitness values are selected. Further, the fitness value for every particle is computed on the basis of the best position with the following equation.

$$fit(x_a) = \begin{cases} \frac{1}{1+f(p_{best_i})}, if\ f(p_{best_i}) \geq 0 \\ fit(x_a) = 1 + |f(p_{best_i})|, otherwise \end{cases} \tag{22}$$

The probability p_a for the selection of the ath particle is calculated as the following equation.

$$P_a = fit(x_a) / \sum_{a=1}^{n} fit(x_a) \tag{23}$$

The particles were selected on the basis of probability pa by utilising the roulette method. Further, the particles which have better p_{best_i} can possibly be selected. When assuming that the sth particle x_a was selected, Equation (26) will be utilised for generating the new position x_s^{new}. If this new position was better than p_best_s, then the p_best_s will be replaced by x_s^{new}. Subsequently, and for particles that fail to update their p_best_s, their counter will be significantly increased.

The particles that fail to update its p_{best_i} in some iterations are considered exhausted and are abandoned. Velocity and position as well as p_{best_i} are randomly initialised in the search space.

So the position and velocity updating equations in PSO algorithm can be represented with the following equation.

$$\begin{cases} v_a^{new} = w.v_i^{old} + c.rand.(pbest_{\tau a} - x_a^{old}) \\ x_a^{new} = x_a^{old} + v_a^{new} \end{cases} \tag{24}$$

In the above equation, rand indicates the random vector within [0, 1], w indicates the inertia weight, and c is said to be the learning factor. τa and $p_{best_{\tau a}}$ indicate the index vector for the ath particle.

5. Our Proposed Task Offloading and Resource Allocation Scheme

5.1. Task Offloading

The decision of the vehicle to offload a task depends on its offloading demands and offloading strategies of other vehicles. In this paper, the use of the decision analysis (DA) algorithm will help to make the decision feasible and ensure convergence, thereby making an optimal decision.

The DA method is used to formulate the multivehicle computing problem. In this problem, every vehicle is considered as a player and has needs for computational tasks. Players in this game will create a decision tree to maximise the payoff (i.e., resource utility). For vehicle i, the decision strategy function is given as $D(s_i, s_{-i})$, where $s_{-i} = (s_1, s_2 \ldots, s_{i-1}, s_{i+1}, \ldots, s_n)$ denotes the other vehicles' offloading strategies except vehicle i. The objective of each vehicle is to choose a valuable offloading strategy.

The task offloading strategy DA is described as

$$\max_{s_i} D(s_i, s_{-i}) = s_i^{vec} * D_i^{vec} + s_i^{loc} * D_i^{loc} \tag{25}$$

where vehicle set is n, and the set of offloading strategies of vehicle i is denoted by D_i.

Therefore, all vehicles calculate the expected payoff and look for the maximum payoff. Through mutual iteration, the system enters into a steady state and achieves the near-optimal solution.

5.2. Computation Resource Allocation

This section aims to maximise the vehicles' utility by offloading the tasks to the VEC server and the optimum resource allocation can be derived by the using the below formula.

$$\max_{R} \Sigma \mu_i * ln(1 + max(max(ETT_i^{VEC}) - ETT_i^{VEC})) - (1 - \mu_i) * \zeta_{vec} * \alpha_i^{VEC} \quad (26)$$

For solving the above resource allocation problem, we have used the particle swarm optimisation (PSO) algorithm which is a metaheuristic optimisation algorithm [42] in which a population of particles (i.e., vehicles), each of which represents a possible solution to an optimisation problem which direct the search process towards the best solution.

The process of the PSO optimisation has been presented in Algorithm 1 below.

Algorithm 1: Resource allocation and offloading strategy

Decision Strategy Calculation using DA method

1. To start with n number of vehicles, i.e., $X = \{1, 2, 3, \ldots, N\}$
2. for all vehicle $i \epsilon X$ repeat
    3. calculate the offloading strategy using Equation (15) (Solving using DA method)
    4. if $(D^{VEC} > D^{local})\ D^{VEC} = 1\ and\ D^{local} = 0$
    5. if $(D^{VEC} < D^{local})\ then D^{VEC} = 0\ and\ D^{local} = 1$
    6. calculate the total energy of the system, i.e., TE from Equation 16
7. end of for

Resource Allocation Calculation using PSO method

8. Starting with again n number of vehicles, i.e., $X = \{1, 2, 3, \ldots, N\}$ and
Problem Formulation: Considering other factors as stated in Section 3.4: $T_i = C_i, d_i, t_i^{max}$, where $C_i = d_i * K_i$; d_i represents the size of the data block, K_i represents the service co-efficient of the vehicle, C_i is the computational resource required to complete the task T_i, and t_i^{max} represents the maximum acceptable delay for that task.
9. Particle Representation: Randomly choose a swarm of particles, for each particle, initialise position and velocity (particle's position represent parameters such as vehicle speed).
10. Define the fitness function, often denoted as f(x_i), $i = 1, \ldots\ldots, N$; where x represents a particle's position in the search space (Here fitness function related to the objective of the optimisation problem).
11. Let $pbest_i$ be current best position, let also g_{best} be current best position among all particles.
12. while the convergence condition is not satisfied do
13. for each index $i = 1 \rightarrow N$ do
14. Determine the global best position $gbest_a$ among all particles
15. Update the particle velocity v_a and the position x_a using $v_i(t+1) = \omega^* v_i(t) + c_1 {}^*r_1{}^*(pbest_i - x_i(t)) + c_2 {}^*r_2{}^*(gbest - x_i(t))$
16. Update new position f(x_i^{t+1}) = $x_i^t + v_i(t+1)$
17. Evaluate the fitness of each particle's new position using the fitness function: If x_i^{new} is better than $pbest_i$ then step 10 otherwise step 11
18. $pbest_i = x_i^{t+1}$
19. Evaluate the fitness of each particle's new position using the fitness function: If x_i^{new} is better than $pbest_i$ then step 12 otherwise step 13
20. $gbest_i = x_i^{t+1}$
21. endIf
22. endfor
23. Return the best solution found, represented by the global best position $gbest_i$ for obtaining resource allocation as specified in Equation (18).

r_1 and r_2 are random numbers. c_1 and c_2 are cognitive and social parameters, respectively. ω is the inertia weight. $v_i(t)$ is the velocity of particle i at time t. g_{best} is the global

best particle among all particles, and $pbest_i$ is the personal best position of particle i. Finally, $x_i(t)$ is referred as the current position of particle I [42].

6. Results

In our system model, RSUs broadcast beacon messages to all the vehicles on computation resource information in their communication range. All the vehicles also periodically share their relevant information with the RSUs, and once the connection is established, communication goes into the unicast mode between the RSUs and vehicles. In this simulation, we have ignored communication overhead, as the size of the message is too small in comparison to the higher bandwidth used in 5G NR-V2X.

Performance Analysis

Using MATLAB software, 2022b the performance of the proposed algorithm has been evaluated in comparison with different algorithms, i.e., the computation task offloading and resource allocation (CTORA) algorithm, the computation offloading decision optimisation (CODO) algorithm, and the heuristic scheme algorithm.

Our proposed work is evaluated against the CTORA, CODO, and heuristic scene algorithm.

The heuristic scheme [43] allows work to be offloaded to the VEC server when a vehicle's time and energy restrictions cannot be met by doing computation locally. In this algorithm, other cars are not taken into consideration throughout this process. In the CODO scheme [44], the tasks are either performed locally or offloaded to the VEC server using the computation offloading decision optimisation scheme. The primary distinction between our proposed method and CODO is that our method took mmWave communication into account. Finally, the CTORA method [45] solely focuses on optimising the decisions related to offloading inside a specific computing resource.

In this section, we present the numerical findings of our proposed algorithm. In this scenario, vehicles are simulated to move in a single direction. We have considered six RSUs in a line, and each RSU has a VEC server along with it. In the simulation, we have used [15, 20] GHz as the computation resource for each vehicle.

The detailed settings of other simulation parameters are summarised in Table 2.

Table 2. Table below represents all the simulation parameters used in this article.

Parameter Used	Value
Cellular link Bandwidth	25 MHz
mmWave Bandwidth	250 MHz
Number of RSUs'	6
RSU coverage length	250 m
Communication range for mmWave and cellular links	150 and 250 m
Input data size	[30, 80] kB
Maximum latency constraint	[0.1, 1] s
Vehicles arrival rates	0.1 vehicles per second
Vehicle (Vi) transmission rate ((p_i))	1.5 W
Average velocity of Vehicles	45 km/h
Computational resource price/VEC	0.05 4/GHz
Vehicle Antenna gain (A_i)	20 db
RSU antenna gain (A_j)	20 db
Energy consumption/computing unit	2×10^{-10} w
Input data size	[30, 60] kB
Path loss exponent (λ)	3.1

Figure 3 shows the computational efficiency with respect to the number of vehicles and illustrates the impact of communication performance with the varying number of vehicles on computational efficiency. For these results, the data size of the task has been kept at 1 kilobit, and the vehicle speed has been maintained at 45 km per hour. The locations of the

vehicles around the RSU have been considered random. Upon analysis of the results, it can be observed that as the number of vehicles increases, computational efficiency decreases for all algorithms as well as the proposed algorithms. The decrease in computational efficiency is attributed to various factors such as the location of the vehicles, SNR, and task data size. Depending on the vehicle's location, the signal-to-noise ratio decreases with the increased vehicle distance from the RSU. Computational efficiency also depends on the communication technology, such as the use of mm waves and cellular waves. From the results, it can be seen that the CODO Scheme performs worse due to its lack of utilisation of mm wave communication. Additionally, it is observed that the heuristic scheme performs better than the CODO scheme and provides better performance until the vehicle count exceeds 95. Finally, it is also evident that our proposed scheme performs the best of all, even when the number of vehicles is up to 95, but there is slight fall in the performance after 95.

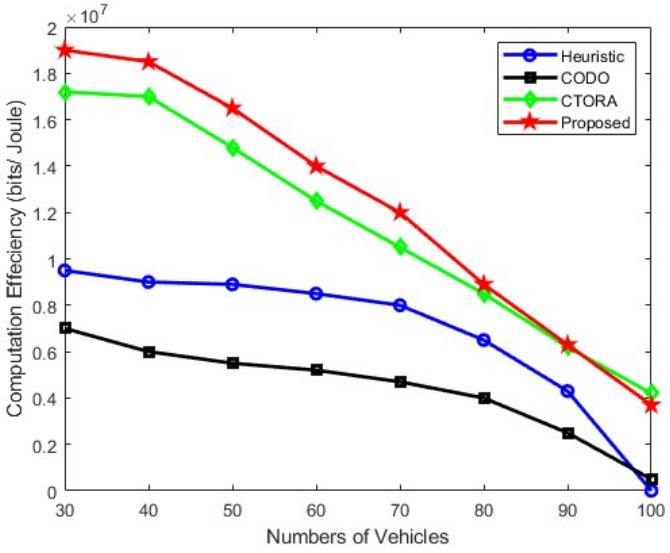

Figure 3. Computation efficiency vs. varying number of vehicles for communication performance.

In Figure 4, the computational efficiency is analyzed with respect to the required computational data size used during communication. It can be observed from the results that our proposed scheme performs better than the others, i.e., CODO and CTORA. Furthermore, it is evident that as the required computing data size increases, the computational efficiency decreases due to various reasons. Firstly, processing times and latency increase as a result of handling higher data quantities, demanding more processing power and memory. Secondly, higher data volumes may potentially cause network congestion, requiring more bandwidth to send and receive data packets, leading to packet loss and re-transmissions. Additionally, the need for more sophisticated data compression, routing, and management methods due to larger data sizes further increases computational cost. In Figure 4, it can be seen that the computational efficiency for our proposed scheme is closer to the others when the required computing data size is smaller. However, as the computation data size increases, our proposed scheme exhibits a decrease in computational efficiency; although, it remains superior to others. In the heuristic approach, offloading decisions are made by vehicles if energy constraints and local computation time fail to meet requirements, resulting in lower and relatively unchanged computational efficiency. The local computation is deemed the best option when the required computing data size is smaller, as offloading depends on available bandwidth and channel gain between the RSU and vehicles. Conversely, offloading to the VEC server becomes the preferred option when the required computing data size increases.

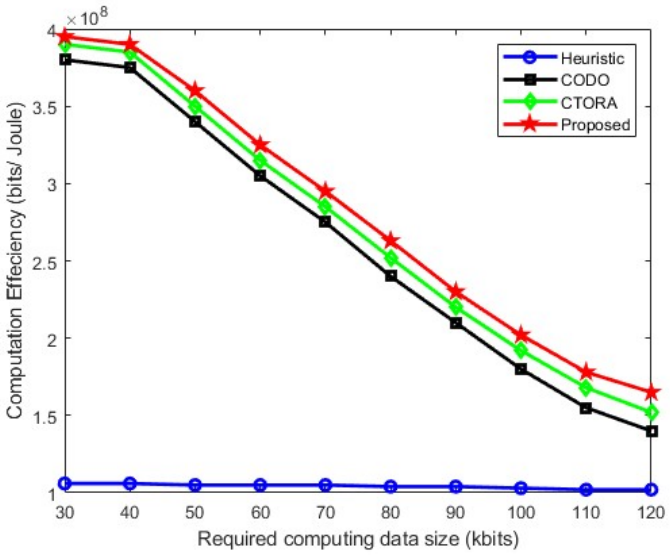

Figure 4. Computation efficiency vs. computing data size needed.

Figures 5–7 illustrate the computational efficiency when the number of vehicles ranges from approximately 2 to 20, maintaining an average speed of 25 km/h, 45 km/h, and 65 km/h, respectively. In this analysis, the required computational data size remains constant and uniform across all vehicles. Notably, the heuristic schemes operate at a diminished capacity, as vehicles tend to prioritise local computation to minimise practical tolerable delay in Figure 5. Conversely, other schemes, namely CODO and CTORA, perform commendably, alongside our proposed scheme. In Figures 6 and 7, the efficacy of our proposed scheme surpasses that of CODO and CTORA, attributed to the integration of mmWave communication and optimised resource allocation strategies, enhancing the overall performance and efficiency and shows a stable gain. A slight gain has been observed in Figures 6 and 7 for Heuristic method, but the gain is too low in comparison to other models. Similar trend is been observed in Figures 6 and 7, for CODO, CTORA, and our proposed model where an average speed of 45 km/h and 65 km/h has been taken into consideration. Notably, our proposed model does shows similar gain. Our suggested scheme also works well in conjunction with CODO and CTORA, the other two schemes. Hence, it shows that the use of the PSO algorithm does help in optimised resource allocation and thereby improving the overall performance and efficiency.

Figure 8 presents an analysis of computational efficiency with regard to the maximum tolerable delay. The findings indicate a notable trend: as the maximum tolerable delay increases, there is a corresponding decrease in the total energy consumption. In this context, both the CODO, CTORA, and our proposed algorithm exhibit a strong performance compared to heuristic approaches. This can be attributed to their shared strategy of prioritising the offloading of most tasks to the VEC server when the maximum tolerable delay is set at 2. Conversely, the heuristic algorithm tends to favour local computation once the maximum tolerable delay surpasses 2. A distinguishing feature of our proposed algorithm is its superior performance when compared to other algorithms (heuristic, CODO, and CTORA) as the maximum tolerable delay increases further. Despite this advantage, the performance gap of our proposed algorithm aligns closely with that of the CODO and CTORA algorithms; although, the disparities widen slightly by 8% and 5%, respectively. This suggests that our proposed algorithm maintains competitiveness and performed well, particularly in scenarios with higher maximum tolerable delays.

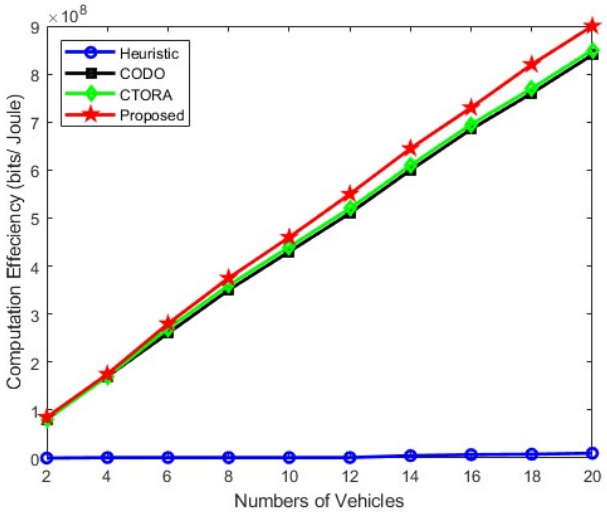

Figure 5. Computation efficiency vs. varying the number of vehicles under speed of 25 km/h.

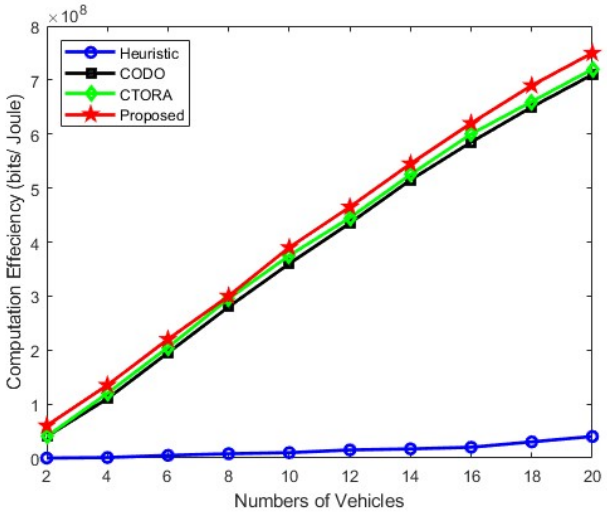

Figure 6. Computation efficiency vs. varying the number of vehicles under under speed of 45 km/h.

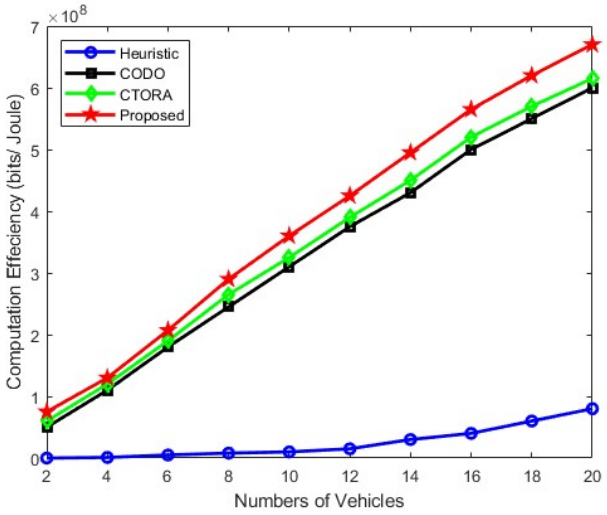

Figure 7. Computation efficiency vs. varying the number of vehicles under under speed of 65 km/h.

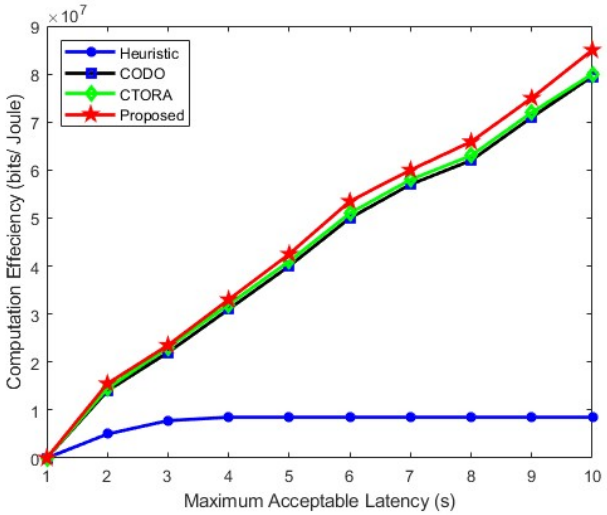

Figure 8. Computational efficiency vs. maximum acceptable latency.

7. Conclusions

Within the context of a tradeoff between computing time and energy consumption, we examined a vehicle's strategy to offload duties in order to maximise computation efficiency in this research. We combined job offloading and computation resource allocation to construct the computation efficiency problem. Our PSO approach combined with the DA method allowed us to accomplish an efficient resource allocation and task offloading. In addition, we made use of mmWave and cellular link in the 5G NR-V2X communication paradigm to enhance system communication latency and performance. According to the numerical results, the suggested technique shows an increase in computation efficiency adhering to energy and computation time limits. Hence, the comparison of different algorithms with respect to the proposed algorithm, i.e., the PSO optimisation algorithm, is shown in Figures 4–7. Among them, the average energy consumption of the PSO algorithm shows an enhancement in all. In comparison, we can see that the proposed algorithm (PSO algorithm) consumed less energy in comparison to CTORA, CODO, and heuristic; however, CTORA and CODO seems to be the same or very close at the start, but on the further end of each result, our proposed model showed an enhancement in comparison to other algorithms. In general, the suggested method, PSO optimisation with the DA method demonstrates superior performance compared to other algorithms, while keeping 1KB packet sizes. We would expand our research in the future to investigate and use other optimisation approaches such as Sequential Quadratic Programming (SQP) or the Genetic Algorithm approach to enhance the long-term delay performance and fortify the job offloading procedure along with the combined use of mmWave and 5G NR communications.

Author Contributions: The Conceptualization of the novel approach was done by A.A. and K.A. The Methodology section of the article was completed by A.A., P.S., R.T. and G.M. The software simulations were performed by A.A. under the guidance of K.A. and P.S. The system model validations were completed by A.A., P.S. and K.A., with formal analysis being presented by K.A. and G.M.; A.A. performed a thorough investigation with resources provided by P.S. The original draft was written by A.A. and K.A. with extensive review and editing support from P.S. and R.T. The supervisory team for A.A. included K.A., P.S., R.T. and G.M. All authors have read and agreed to the published version of the manuscript.

Funding: This research was conducted as part of the UKIERI-SPARC project titled "DigIT—Digital Twins for Integrated Transportation Platform". The grant number is UKIERI-SPARC/01/23.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data presented in this study are available on request from the corresponding author.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Jo, K.; Kim, J.; Kim, D.; Jang, C.; Sunwoo, M. Development of autonomous car—Part II: A case study on the implementation of an autonomous driving system based on distributed architecture. *IEEE Trans. Ind. Electron.* **2015**, *62*, 5119–5132. [CrossRef]
2. Lin, S.; Zhang, Y.; Hsu, C.; Skach, M.; Haque, M.; Tang, L.; Mars, J. The architectural implications of autonomous driving: Constraints and acceleration. In Proceedings of the Twenty-Third International Conference on Architectural Support for Programming Languages and Operating Systems, Williamsburg, VA, USA, 24–28 March 2018; pp. 751–766. [CrossRef]
3. Ashok, A.; Steenkiste, P.; Bai, F. Vehicular cloud computing through dynamic computation offloading. *Comput. Commun.* **2018**, *120*, 125–137. [CrossRef]
4. Feng, J.; Liu, Z.; Wu, C.; Ji, Y. Mobile edge computing for the internet of vehicles: Offloading framework and job scheduling. *IEEE Veh. Technol. Mag.* **2018**, *14*, 28–36. [CrossRef]
5. Mao, Y.; You, C.; Zhang, J.; Huang, K.; Letaief, K. A survey on mobile edge computing: The communication perspective. *IEEE Commun. Surv. Tutor.* **2017**, *19*, 2322–2358. [CrossRef]
6. Hu, Y.; Patel, M.; Sabella, D.; Sprecher, N.; Young, V. Mobile edge computing—A key technology towards 5G. *ETSI White Pap.* **2015**, *11*, 1–16.
7. Zhang, X.; Zhang, J.; Liu, Z.; Cui, Q.; Tao, X.; Wang, S. MDP-based task offloading for vehicular edge computing under certain and uncertain transition probabilities. *IEEE Trans. Veh. Technol.* **2020**, *69*, 3296–3309. [CrossRef]
8. Raza, S.; Wang, S.; Ahmed, M.; Anwar, M. Others A survey on vehicular edge computing: Architecture, applications, technical issues, and future directions. *Wirel. Commun. Mob. Comput.* **2019**, *2019*, 3159762.
9. Fog Computing. The Internet of Things: Extend the Cloud to Where the Things are. *Cisco White Pap.* **2015**, 3.
10. Li, X.; Dang, Y.; Chen, T. Vehicular edge cloud computing: Depressurize the intelligent vehicles onboard computational power. In Proceedings of the 2018 21st International Conference on Intelligent Transportation Systems (ITSC), Maui, HI, USA, 4–7 November 2018. [CrossRef]
11. Gu, B.; Zhou, Z. Task offloading in vehicular mobile edge computing: A matching-theoretic framework. *IEEE Veh. Technol. Mag.* **2019**, *14*, 100–106. [CrossRef]
12. Pu, L.; Chen, X.; Mao, G.; Xie, Q.; Xu, J. Chimera: An energy-efficient and deadline-aware hybrid edge computing framework for vehicular crowdsensing applications. *IEEE Internet Things J.* **2018**, *6*, 84–99. [CrossRef]
13. Zhao, J.; Li, Q.; Gong, Y.; Zhang, K. Computation offloading and resource allocation for cloud assisted mobile edge computing in vehicular networks. *IEEE Trans. Veh. Technol.* **2019**, *68*, 7944–7956. [CrossRef]
14. Dai, Y.; Xu, D.; Maharjan, S.; Zhang, Y. Joint load balancing and offloading in vehicular edge computing and networks. *IEEE Internet Things J.* **2018**, *6*, 4377–4387. [CrossRef]
15. Xu, X.; Xue, Y.; Qi, L.; Yuan, Y.; Zhang, X.; Umer, T.; Wan, S. An edge computing-enabled computation offloading method with privacy preservation for internet of connected vehicles. *Future Gener. Comput. Syst.* **2019**, *96*, 89–100. [CrossRef]
16. Saif, F.; Latip, R.; Hanapi, Z.; Alrshah, M.; Shafinah, K. Workload Allocation Towards Energy Consumption-delay Trade-off in Cloud-fog Computing using Multi-objective NPSO Algorithm. *IEEE Access* **2023**, *11*, 45393–45404. [CrossRef]
17. Wu, C.; Liu, Z.; Liu, F.; Yoshinaga, T.; Ji, Y.; Li, J. Collaborative learning of communication routes in edge-enabled multi-access vehicular environment. *IEEE Trans. Cogn. Commun. Netw.* **2020**, *6*, 1155–1165. [CrossRef]
18. You, C.; Huang, K.; Chae, H.; Kim, B. Energy-efficient resource allocation for mobile-edge computation offloading. *IEEE Trans. Wirel. Commun.* **2016**, *16*, 1397–1411. [CrossRef]
19. Zhang, K.; Mao, Y.; Leng, S.; Vinel, A.; Zhang, Y. Delay constrained offloading for mobile edge computing in cloud-enabled vehicular networks. In Proceedings of the 2016 8th International Workshop on Resilient Networks Design and Modeling (RNDM), Halmstad, Sweden, 13–15 September 2016; pp. 288–294. [CrossRef]
20. Bi, J.; Wang, Z.; Yuan, H.; Zhang, J.; Zhou, M. Cost-Minimized Computation Offloading and User Association in Hybrid Cloud and Edge Computing. *IEEE Internet Things J.* **2024**, *11*, 16672–16683. [CrossRef]
21. Jafari, V.; Rezvani, M. Joint optimization of energy consumption and time delay in IoT-fog-cloud computing environments using NSGA-II metaheuristic algorithm. *J. Ambient Intell. Humaniz. Comput.* **2023**, **14**, 1675–1698. [CrossRef]
22. Lahlou, L.; Tata, C.; Kara, N.; Leivadeas, A.; Gherbi, A. Edge-cloud online joint placement of Virtual Network Functions and allocation of compute and network resources using meta-heuristics. *J. Ambient Intell. Humaniz. Comput.* **2023**, *14*, 7531–7558. [CrossRef]
23. Zhou, Z.; Feng, J.; Chang, Z.; Shen, X. Energy-efficient edge computing service provisioning for vehicular networks: A consensus ADMM approach. *IEEE Trans. Veh. Technol.* **2019**, *68*, 5087–5099. [CrossRef]
24. Patsias, V.; Amanatidis, P.; Karampatzakis, D.; Lagkas, T.; Michalakopoulou, K.; Nikitas, A. Task allocation methods and optimization techniques in edge computing: A systematic review of the literature. *Future Internet* **2023**, *15*, 254. [CrossRef]
25. Pervez, F.; Sultana, A.; Yang, C.; Zhao, L. Energy and latency efficient joint communication and computation optimization in a multi-UAV assisted MEC network. *IEEE Trans. Wirel. Commun.* **2023**, *23*, 1728–1741. [CrossRef]

26. Fan, X.; Gu, W.; Long, C.; Gu, C.; He, S. Optimizing Task Offloading and Resource Allocation in Vehicular Edge Computing Based on Heterogeneous Cellular Networks. *IEEE Trans. Veh. Technol.* **2023**, 1–13. [CrossRef]
27. Fan, W.; Su, Y.; Liu, J.; Li, S.; Huang, W.; Wu, F.; Liu, Y. Joint task offloading and resource allocation for vehicular edge computing based on V2I and V2V modes. *IEEE Trans. Intell. Transp. Syst.* **2023**, *24*, 4277–4292. [CrossRef]
28. Hussain, M.; Azar, A.; Ahmed, R.; Umar Amin, S.; Qureshi, B.; Dinesh Reddy, V.; Alam, I.; Khan, Z. SONG: A multi-objective evolutionary algorithm for delay and energy aware facility location in vehicular fog networks. *Sensors* **2023**, *23*, 667. [CrossRef]
29. Zhang, C.; Wu, C.; Lin, M.; Lin, Y.; Liu, W. Proximal Policy Optimization for Efficient D2D-Assisted Computation Offloading and Resource Allocation in Multi-Access Edge Computing. *Future Internet* **2024**, *16*, 19. [CrossRef]
30. Li, C.; Chen, Q.; Chen, M.; Su, Z.; Ding, Y.; Lan, D.; Taherkordi, A. Blockchain enabled task offloading based on edge cooperation in the digital twin vehicular edge network. *J. Cloud Comput.* **2023**, *12*, 120. [CrossRef]
31. Hua, W.; Liu, P.; Huang, L. Energy-efficient resource allocation for heterogeneous edge-cloud computing. *IEEE Internet Things J.* **2023**, *11*, 2808–2818. [CrossRef]
32. Fan, W.; Hua, M.; Zhang, Y.; Su, Y.; Li, X.; Wu, F.; Liu, Y. Game-Based Task Offloading and Resource Allocation for Vehicular Edge Computing with Edge-Edge Cooperation. *IEEE Trans. Veh. Technol.* **2023**, *72*, 7857–7870. [CrossRef]
33. Huang, T.; Liu, J.; Zhou, X.; Nguyen, D.; Azghadi, M.; Xia, Y.; Han, Q.; Sun, S. V2X Cooperative Perception for Autonomous Driving: Recent Advances and Challenges. *arXiv* **2023**, arXiv:2310.03525.
34. Liu, X.; Yang, L.; Liao, T.; Luo, Z.; Yu, D.; Yue, G. Measurements and Analysis of Millimeter-Wave Propagation from In-station to Out-station in High-Speed Railway between Train and Trackside. *IEEE Trans. Wirel. Commun.* **2023**. [CrossRef]
35. Okello, K.; Mwangi, E.; Abd El-Malek, A. Connectivity probability analysis for VANETs with big vehicle shadowing. In Proceedings of the 2023 International Symposium on Networks, Computers Furthermore, Communications (ISNCC), Doha, Qatar, 23–26 October 2023; pp. 1–6. [CrossRef]
36. Ren, S.; Zhao, J.; Zhang, H.; Li, X. Connectivity Analysis with Co-Channel Interference for Urban Vehicular Ad Hoc Networks. *Electronics* **2023**, *12*, 2021.
37. Ziemer, R.E.; Tranter, W.H. *Principles-of-Communications*, 7th ed.; John Wiley & Sons: Hoboken, NJ, USA, 2015; Volume 12. Available online: https://physicaeducator.files.wordpress.com/2018/03/principles-of-communications-7th-edition-ziemer.pdf (accessed on 22 August 2023).
38. Giordani, M.; Zanella, A.; Higuchi, T.; Altintas, O.; Zorzi, M. Performance study of LTE and mmWave in vehicle-to-network communications. In Proceedings of the 2018 17th Annual Mediterranean Ad Hoc Networking Workshop (Med-Hoc-Net), Capri, Italy, 20–22 June 2018; pp. 1–7. [CrossRef]
39. Qin, P.; Wang, Y.; Cai, Z.; Liu, J.; Li, J.; Zhao, X. MADRL-Based URLLC-Aware Task Offloading for Air-Ground Vehicular Cooperative Computing Network. *IEEE Trans. Intell. Transp. Syst.* **2023**, 1–14. [CrossRef]
40. Wang, H.; Li, X.; Ji, H.; Zhang, H. Federated offloading scheme to minimize latency in MEC-enabled vehicular networks. In Proceedings of the 2018 IEEE Globecom Workshops (GC Wkshps), Abu Dhabi, United Arab Emirates, 9–13 December 2018; pp. 1–6. [CrossRef]
41. Zhang, J.; Xia, W.; Yan, F.; Shen, L. Joint computation offloading and resource allocation optimization in heterogeneous networks with mobile edge computing. *IEEE Access* **2018**, *6*, 19324–19337. [CrossRef]
42. Chen, X.; Tianfield, H.; Du, W. Bee-foraging learning particle swarm optimization. *Appl. Soft Comput.* **2021**, *102*, 107134. [CrossRef]
43. Kuo, T.; Li, D. Gbho: A gain-based heuristic offloading algorithm in vehicular edge computing. In Proceedings of the 2022 IEEE 95th Vehicular Technology Conference: (VTC2022-Spring), Helsinki, Finland, 19–22 June 2022; pp. 1–7.
44. Xu, X.; Xue, Y.; Li, X.; Qi, L.; Wan, S. A Computation Offloading Method for Edge Computing With Vehicle-to-Everything. *IEEE Access* **2019**, *7*, 131068–131077. [CrossRef]
45. Chen, X.; Jiao, L.; Li, W.; Fu, X. Efficient Multi-User Computation Offloading for Mobile-Edge Cloud Computing. *IEEE/ACM Trans. Netw.* **2016**, *24*, 2795–2808. [CrossRef]