CoTEV: Trustworthy and Cooperative Task Execution in Internet of Vehicles

Shashwat Pratap, Prajnamaya Dass, *Graduate Student Member, IEEE* and Sudip Misra, *Fellow, IEEE*

Abstract—Due to the increasing number of service requests from the vehicles, the load at the road side units (RSUs) increases, which affects the delay-sensitive vehicle services. In Internet of Vehicles (IoV), the vehicles can communicate directly with other vehicles and take help from the vehicles to cooperatively accomplish a task. However, it is very challenging to cooperatively execute a task in an IoV environment with high traffic and dynamic vehicle movements. Furthermore, it is difficult for a task vehicle to choose trustworthy and cooperative vehicles. In this paper, we propose algorithms for cooperative task execution by taking the help of trusted vehicles, when it is not possible to complete a deadline-specified task through the RSUs. We propose a hedonic coalition formation game-based approach to form distributed coalitions of cooperative vehicles. We consider the trust score of the vehicles along with their computational capabilities and journey routes. After each task execution, the service feedback is reflected in the trust score of each cooperative vehicle in the coalition. Our proposed algorithms allow the cooperative vehicles to autonomously choose the coalitions and select a vehicle task to maximize their payoffs. To satisfy the task deadlines in multiple coalitions, we design the merging of vehicle coalitions. We consider the simulation of urban mobility (SUMO) tool to generate the mobility traces of the vehicles in a real road network of Berlin city, which considers the traffic junctions and vehicle density on the roads. Through extensive simulations, we show that the proposed algorithms significantly increase the service rate of delay-sensitive task requests by at least 30.5% and the trust score by at least 20.61%, compared to the benchmark schemes.

Index Terms—Coalitional game, IoV, Cooperative game, Trust, Task offloading, Task execution

1 Introduction

The popularity and adoption of the Internet of Things (IoT) is growing exponentially due to rapid urbanization in various sectors around the globe [1], [2]. Due to different smart city projects, digitization platforms, and automation sectors, it is estimated that more than 75 billion IoT devices will be connected by 2025 [3]. The introduction of IoT in public transport has also included vehicles in the IoT system. The on-road moving vehicles get real-time services from the road side units (RSUs). Further, to execute the computation-intensive tasks, the vehicles offload their tasks to the RSUs [4], [5]. The RSUs schedule the incoming tasks and execute them based on priorities and deadlines. However, the number of vehicles is increasing at an exponential rate [6], and due to high traffic situations, it is not always possible for the RSUs to handle the time-sensitive task requests before deadlines [1], [7]. To support the system, Internet of Vehicles (IoV) has been coined from the IoT in the domain of intelligent transportation, in which the vehicles communicate among themselves and can cooperate to execute a task or accomplish a service request [8], [9].

1.1 Motivation

Due to the high task load at the RSUs, it is very difficult to meet the deadlines of the offloaded tasks. To overcome this issue, many works in the literature have considered task offloading to nearby vehicles, which are computationally capable of executing the tasks. The existing works on vehicle-to-vehicle (V2V) offloading focus only on the objectives of task offloading vehicles, which are to minimize the delay and energy consumption associated with the offloaded task [7], [10], [11]. However, in an IoV environment, along with the objectives of task offloading vehicles, the preferences of the cooperative vehicles also play an important role. For example, when multiple task offloading vehicles are available, a cooperative vehicle always prefers to share its resources with a vehicle giving the maximum incentive.

Furthermore, the existing works assume that the cooperative vehicles should move very close or at constant speed till the task completion to execute and deliver the task output [7], [10], [12], [13]. In a vehicular environment with heterogeneous vehicles and high traffic conditions, it is challenging to restrict the speed and movement of the vehicles, as required. In addition, due to multiple communications, the latency increases when a centralized entity (RSU or task offloading vehicle) takes all task offloading decisions for mobile vehicles.

The trustworthiness of the vehicles considered for cooperative task execution is crucial to deliver reliable task output [9], [1], [14], [15]. The existing trust models use the trust scores of the vehicles only during the selection of cooperative vehicles. However, in a distributed approach, a cooperative vehicle should also consider the trust scores of other vehicles that share resources for the same task. Therefore, the trust scores should be used by both the task offloading vehicle and cooperative vehicles.

S. Pratap is with the Electronics and Communications Engineering Department, Indian Institute of Information Technology Ranchi, 834010, India.
P.Dass and S. Misra are with the Computer Science and Engineering Department, Indian Institute of Technology, Kharagpur, 721302, India.
E-mail: shaswat.btech.ec19@iiitranchi.ac.in, prajnamaya@iitkgp.ac.in, smisra@cse.iitkgp.ac.in

1.2 Contributions

Considering the motivation points, there is a need for algorithms that will allow the task vehicles and cooperative vehicles to maximize their utilities, while satisfying the task deadlines and trustworthiness in the mobile IoV environment. The major contributions are as follows:

- We propose a hedonic coalition game-based solution approach for the formation of distributed coalitions of cooperative vehicles. Unlike the existing approaches, where only the objective of the task vehicle is considered, the proposed approach considers the utilities of the cooperative vehicles and the coalition utility (utility of the task offloading vehicle).
- For trustworthy task execution, we consider the trust score of the cooperative vehicles into account, along with their computational capabilities and mobility patterns. The trust scores of the cooperative vehicles help the task vehicle to maximize the coalition utility and the cooperative vehicles to maximize their utilities by selecting the best coalition according to their current trust scores.
- The proposed algorithms allow the vehicles moving in same route of the task offloading vehicle to autonomously make their decisions without any centralized entity. We also find out the maximum size of the independent disjoint coalitions and show the polynomial time stability of the coalitions after switch operations.
- We use the SUMO tool to generate the mobility traces for heterogeneous vehicles in a real road network of Berlin city. Finally, we show the effectiveness of the proposed algorithms over the benchmark schemes. The proposed algorithms significantly increases the service rate of delay-sensitive task requests by at least 30.5% and the trust score of the tasks by at least 20.61%, compared to the benchmark schemes.

2 Literature Survey

Due to the emergence of IoT-enabled intelligent transport systems over the last few years, researchers have extensively worked on different vehicle service provisioning approaches. In this Section, we briefly discuss the existing works that consider vehicle task execution in IoV. We divide the relevant works into centralized, distributed, and hybrid approaches based on the underlying principle of selecting the task-executing nodes. At the end of this Section, we synthesize the existing works and specify the motive for the proposed work.

2.1 Centralized Approaches

In a centralized approach, a single node decides the other nodes, who collaboratively complete a vehicle's task. Zhang *et al.* [14] investigated the multi-part collaborative task offloading and resource allocation problem in mobile edge computing applications. In the proposed approach, the base stations uses a genetic and deep deterministic policy gradient-based strategies to take decisions for multi-part computation offloading. Liu *et al.*[17] considered the task offloading problem in vehicular edge computing (VEC). In the proposed approach, the base station uses matching theory to select the RSUs that minimize the total offloading delay for a vehicle task. Cui *et al.* [10] proposed a collaborative approach to execute the computation-intensive task from the vehicles. The authors first select the computationally-capable vehicles and then use a double deep Q-network-based approach to minimize the total task offloading delay in a multi-vehicle collaboration system. Li *et al.* [7] formulated the task offloading problem in VEC network as an integer programming problem and solved the problem using an interior point method to minimize the offloading delay and cost. Jiang *et al.* [11] investigated the resource allocation and task offloading problems in IoV in which a Q-learning-based method was proposed to solve the trade-off issue between task delay and energy consumption. Chen *et al.* [18] proposed an asynchronous advantage actor-critic-based computation offloading algorithm to make task offloading decisions in a dynamic IoV environment. In satellite assisted V2V network, Cui et *al.* [26] proposed a solution to jointly optimize the offloading decision, computing, and communication resources, while adopting a Lagrange multiplier method for optimal resource allocation. In [24], a data offloading approach has been proposed to offload the data from the vehicles to the out of range RSUs, where a vehicle decides the other vehicle that helps in forwarding the data to the destination RSU. In another V2V offloading approach [25], the RSU or base station handles the resource allocation and helps in partial computation offloading through V2V communication.

2.2 Distributed Approaches

In distributed approaches, no single entity can select the nodes to execute an offloaded task. The cooperative vehicles have the freedom to share their resources with any vehicle. Wang *et al.* [1] proposed a multiuser non-cooperative game for efficient computational offloading in multi-access edge computing (MEC) servers through the RSUs. In the proposed distributive algorithm, the vehicles take the best response offloading strategies. Xu *et al.* [19] considered the problem of data offloading from the vehicles to the RSUs, where the vehicles associated with different access points share their Wifi resources to offload peer vehicle's data traffic. In [2], a distributive coalition formation among the nodes approach is proposed to create a cooperative environment in ad-hoc networks. Each node in the network independently decides the coalition to maximize its incentive. Halabi and Zulkernine [9] studied the security perspective on the internet of vehicles and proposed cooperative game-based trustworthy vehicle collaborations. The proposed trust-based vehicular coalition formation uses a hedonic game model. Saad *et al.*[22] proposed a task allocation model for the mobile and self-organizing agents in the wireless network. The authors considered the hedonic coalition formation game for the collaboration between the agents. In another work, Saad *et al.* [5] proposed a solution approach for distributed cooperation among the RSUs for the classification and organization of the data collected from the vehicles. In [20], a hedonic coalitional game-based approach has been proposed to solve the task allocation problem for a swarm of multiple agent system. The proposed game-theoretic approach allows the self-interested

TABLE 1: Summary of Existing Works

Parameters → / Works ↓	Delay	Energy	Trust	Communication		Cooperative V-V	Centralized (C) /Distributed (D) / Hybrid(H)
				V2R	V2V		
Wang *et al.* [2]	✗	✗	✗	✗	✓	✓	D
Wu *et al.* [4]	✓	✗	✗	✓	✗	✗	H
Hao *et al.* [16]	✓	✓	✗	✗	✓	✓	D
Zhang *et al.* [14]	✗	✓	✓	✓	✗	✗	C
Saad *et al.*[5]	✗	✗	✗	✓	✗	✗	D
Wang *et al.*[1]	✗	✓	✓	✓	✗	✗	D
Liu *et al.* [17]	✓	✗	✗	✓	✗	✗	C
Li *et al.*[7], Chen *et al.*[18]	✓	✗	✗	✓	✓	✗	C
Jiang *et al.*[11], Cui *et al.*[10]	✓	✓	✗	✓	✓	✗	C
Xu *et al.* [19]	✓	✗	✗	✓	✓	✗	D
Jang *et al.* [20],Dutta *et al.*[21] , Saad *et al.* [22]	✗	✗	✗	✗	✓	✓	D
Halabi and Zulkernine [9]	✗	✗	✓	✗	✓	✓	D
Ko *et al.*[15]	✗	✗	✓	✗	✓	✓	H
Guo *et al.*[23]	✗	✗	✗	✓	✓	✗	H
Saleem *et al.* [24], Shi *et al.* [25]	✓	✗	✗	✓	✓	✗	C
Proposed (CoTEV)	✓	✓	✓	✓	✓	✓	D

agents to participate in the game. In another task allocation among the robots, Dutta *et al.* [21] proposed a coalitional game approach, where multiple tasks are assigned to the coalitions of robots.

2.3 Hybrid Approaches

Some approaches consider both centralized and distributed approach to make offloading decisions for the vehicle tasks. The synergy between centralized and distributed data scheduling in V2V and V2I is investigated in [15] for offloading and balancing the workloads in RSUs. Furthermore, an adaptive scheduling algorithm with the cooperation from the RSUs is proposed. In [23], an SDN-enabled vehicular cooperation network (VCN) is formed and then a V2V offloading method is proposed for the VCN. The proposed offloading algorithm consists of two sub-algorithms – distributed V2V and centralized V2V offloading. Wu *et al.* [4] studied the problem of content caching at the edge nodes, while considering both static and mobile vehicles. In the proposed approach,the content infrastructure providers use a distribute approach to for coalitions and a central controller finds the optimal policy to make caching decisions.

Synthesis: The existing works related to the considered problem in this paper are summarized in Table 1. From the literature survey, we infer that the existing approaches on V2V task offloading consider only the objective of the task offloading vehicle, which is to minimize the delay and energy consumption associated with each task. However, in an IoV environment, along with the objective of task offloading vehicles, the preferences of the cooperative vehicles also play an important role when multiple options are available to share the resources. Furthermore, during collaborations, the trustworthiness of the task executing vehicles should be considered. In this paper, we propose a cooperative task execution approach through which the vehicles autonomously select the tasks and cooperate in task execution. We increase the successful task completion rate of delay-sensitive tasks through the proposed trustworthy and cooperative task execution environment created among the vehicles.

3 System Model

Consider a network where $\mathcal{V} = \{1, 2, ..., n\}$ represents the set of vehicles and $\mathcal{R} = \{1, 2, ..., m\}$ represents the set of RSUs. At any time instant t, an on-road vehicle is associated with the nearest RSU. A vehicle route consists of all the RSUs the vehicle is associated during its journey from a source RSU to a destination RSU. We denote the set of routes by $r = \{r_1, r_2.....r_k\}$. In the system, we consider three types of vehicles – task vehicles, cooperative vehicles, and non-cooperative vehicles. A vehicle that has some task to execute is considered a task vehicle, and we denote such vehicle as v^T. The vehicles that help the task vehicles execute their tasks are considered cooperative vehicles and denoted as v^c. The last type of vehicle, non-cooperative vehicles, do not belong to either of the first two types. The role of the vehicles in the network can change with time, depending on whether they have some tasks to be executed or resources to share. Therefore, at any time instant, each vehicle $i \in \mathcal{V}$ belongs to one of the three categories. We consider both vehicle-to-vehicle (V2V) and vehicle-to-RSU (V2R) communications in the system.

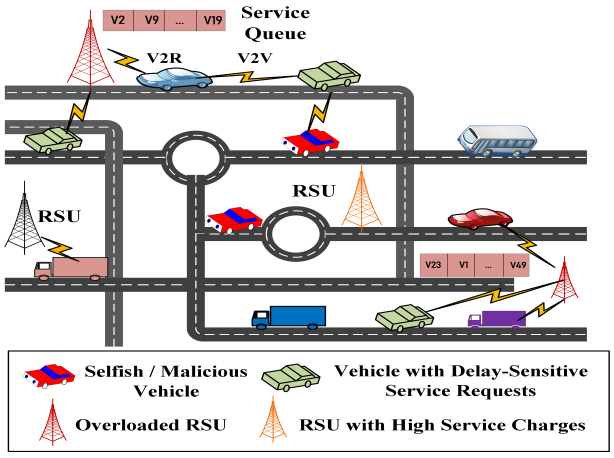


Fig. 1: System Architecture

In the network, we consider that a task vehicle $j \in v^T$ is not computationally capable to perform its task inde-

pendently. In a non-cooperative vehicular environment, the vehicle can only offload its task to the RSU. When an RSU is overloaded, it offloads the task to other RSUs, increasing the delay and service cost. In a network with high traffic, the task execution gets delayed due to the large waiting queues at the RSUs, which affects real-time vehicle services. Vehicles can cooperate in task execution to satisfy task deadlines and minimize costs. Here, we consider the cooperating vehicles that at least follow the route of the task vehicle during that time, starting from the task origin to the vehicle destination (or the location specified for task delivery).

The task vehicle $j \in v^T$ knows the number of CPU cycles C_j needed to complete the task in the maximum allowable delay of Δ_m. Based on this information, the task vehicle constantly approaches the other cooperative vehicles $v^c \in \mathcal{V}$ until a sufficient number of cooperative vehicles agree for cooperation. Each vehicle can perform only one task at a time. If there are some tasks in the queue, then it has to wait till the current task completes.

TABLE 2: Summary of Key Notations

Notation	Description [Units]
$\mathcal{V}$	Set of all vehicles in the network
v^c	Set of cooperating vehicle
v^T	Set of task vehicles
$\mathcal{R}$	Set of all RSUs
r	Set of all routes
τ_k	Trust score of vehicle k
f_k	CPU frequency of vehicle k [cycles/sec]
μ	Data transmission rate [bits/sec]
BW	Channel Bandwidth [Hz]
Δ_t, Δ_e	Transmission and execution delay [sec]
ε	Energy consumption [joule]
ρ, N_0	Transmit power, noise power [watt]
$d_{j,k}$	Distance between two nodes j and k [meter]
γ	Path loss exponent
S	Set of all coalitions
$\|s_j\|$	Size of coalition s_j
M_j	Maximum size of coalition s_j
$\top_j$	Trust score of a coalition s_j
$u_k(s_j)$	Payoff of vehicle k in coalition s_j
$U(s_j)$	Utility of coalition s_j
$R_k(s_j)$	Revenue of vehicle k in coalition s_j
$C_k(s_j)$	Cost of vehicle k in coalition s_j
C^{cost}	Coalition formation cost

3.1 Delay and Energy Consumption

Each task undergoes two phases before its completion: task offloading (direct V2V or via V2R) and task execution. We consider the IEEE 802.11p standard, which supports V2V and V2R communications in intelligent transport systems applications [27]. According to the standard, we have used carrier sense multiple access with collision avoidance (CSMA/CA) technique.

Transmission delay: A task vehicle interacts with the cooperative vehicles through either V2V communication or V2R communication. When the vehicles are in the communication range, they communicate directly (V2V). However, when the task vehicle is far from the other cooperative vehicles, it sends the information via RSU (V2R). The transmission delay between any two nodes j and k can be described as the total data size of the task divided by the data rate and computed as [27]:

$$\Delta_t = \frac{D}{\mu_{jk}} \tag{1}$$

where D is the data size and μ is the data transmission rate of the task vehicle. The maximum data transmission rate can be computed with the logarithmic expression, involving the signal-to-noise ratio (SNR) and the bandwidth involved in the communication, which is expressed as [27]:

$$\mu_{jk} = BW * log_2(1 + SNR) \tag{2}$$

Here, BW represents the bandwidth, the SNR is the ratio of the received signal and noise and computed as:

$$SNR = \frac{\rho_j}{N_0^2} |h_{j,k}|^2 \tag{3}$$

In the above Equation, ρ_j is the power transmitted by the vehicle j and N_0 represents the noise in the channel. The average channel gain of the link between nodes j and k is computed as, $|h_{j,k}|^2 = \frac{1}{d_{j,k}^{\gamma}}$, where $d_{j,k}$ denotes the distance between the nodes and γ is the pathloss exponent. In V2V and V2R communications, transmission delay varies due to the difference in the distance and SNR, between the vehicles and between the vehicle and RSU.

Execution delay: The execution delay for a task with CPU cycle requirement (c_x) can be computed as [28]:

$$\Delta_e = \frac{c_x}{\sum_k f_k} \tag{4}$$

where f_k represents the total combined CPU frequency of the cooperative vehicles that execute the task. The total delay for task offloading and execution is computed as $\Delta_{total} = [\Delta_t + \Delta_e]$.

Energy consumption: We consider the energy consumption for a task as the sum of transmitting energy and execution energy.

$$\varepsilon = [\rho \Delta_t + \sum_k \varrho_k f_k^2 c_k] \tag{5}$$

where $\rho\Delta_t$ is the transmitting energy for offloading the task to the vehicle directly or through RSU. The second term in Equation (5) represents the total energy consumption of all the vehicles for executing their respective part of the tasks. Here, ϱ_k is the energy coefficient of vehicle k, which depends on the chip architecture, f_k is the CPU frequency of vehicle k, and c_k is the number of CPU cycles executed to complete the task allotted to the vehicle.

3.2 Trust Model

The trust score of a vehicle, τ_k, reflects its service reliability and fairness in task completion. After successfully completing the allotted task, the trust score of each vehicle is updated. We assume that a trust controller maintains the trust scores of the cooperative vehicles and updates each time the task vehicle sends the feedback. The updated trust score of cooperative vehicle k after executing the task of j can be computed as [9]:

$$\tau_k' = F[\tau_k; x_k, y_k] \tag{6}$$

In Equation (6), τ_k' is the updated trust score of vehicle k. The incomplete beta function $F[.]$ helps show the trust score update in the range [0-1] while considering the feedback parameters. Here, we consider the values of m and n as the feedback, where n represents the feedback after the successful completion of the task and m represents the feedback of failure in the task completion. The value of m represents the score that increases with the increase in the unfairness of work by the vehicle, while n signifies how well the vehicle performed its tasks (including reliability and efficiency). An increased value of m decreases the trust score, whereas an increase in n increases the trust score. This is the main reason for the use of the Incomplete Beta function to show the trust score numerically, while considering the positive and negative behaviors of the vehicle.

After each task completion, the values of m and n are used to update the values of x_k and y_k.

$$x_k = x_k(1 + m_k),\ y_k = y_k(1 + n_k) \tag{7}$$

Now, the incomplete Beta function with the values x_k and y_k is:

$$F[\tau_k; x, y] = \frac{{\tau_k}^{x_k - 1}(1 - \tau_k)^{y_k - 1}}{\beta(x_k, y_k)} \tag{8}$$

where $\beta(x_k, y_k)$ is the Beta function of x_k and y_k, given as [29]:

$$\beta(x_k, y_k) = \int_0^1 (w^{x_k - 1})(1 - w)^{y_k - 1} \tag{9}$$

The Beta function, being a continuous positive function, effectively reflects the change, which varies between 0-1.

The trust score of an executed task is the average of the trust scores of all the cooperative vehicles that participated in completing the task. As the cooperative vehicles form the coalition, the task vehicle always tries to increase the coalition's trust.

Considering the system model parameters, we try to execute the task of a vehicle in a cooperative IoV environment while satisfying the task deadline. Furthermore, we try to maximize the trust score of each task and the utility of the cooperative vehicles. In the next section, we show the use of the coalitional formation game to achieve the above objectives in a cooperative IoV environment.

4 Coalitional Game Formation

In this section, we build an efficient, trustworthy, and cooperative IoV environment by considering the Hedonic coalition formation game. The coalitional game allows to formulate coalitions of vehicles, while maximizing the utility of both the coalition utility and the individual utility of cooperative vehicles. The switch operations in the coalitional game allow the cooperative vehicles to decide by executing the task of which vehicle, their utility is maximized. Furthermore, the objective of the task vehicle, which is to maximize the trust score of a task and satisfy the task deadline, is also met effectively through the coalition formation game. Finally, the stability of the coalition game ensures that no vehicle leaves the coalition in the middle of task execution [22]. Due to the above-stated reasons, we choose a coalitional formulation game to build a trustworthy cooperative task execution environment in IoV.

4.1 Game Formulation

By inspecting the problem of task execution by a group of cooperative vehicles, we consider the formation of coalitions for each task. Prior to the vehicle coalition formation, we introduce some of the definitions considered from [5], [30] with respect to the cooperative vehicular environment, which provides the analytical tool for studying the formation of collaborative groups among the vehicles.

Definition 1. ***(Vehicle Coalitional Game)****: A vehicle coalitional game is defined as a pair $(\mathcal{V}, U)$, where $\mathcal{V}$ is the set of vehicles and U is the real-valued utility function, denoted as $U(s)$ to represent the utility of coalition s.*

Definition 2. ***(Vehicle Coalition Structure)****: A vehicle coalition structure is defined as $S = \{s_1, s_2, ..., s_i\}$, which partitions the vehicles $\mathcal{V}$ into disjoint coalitions such that $s_i \bigcap s_j = \phi, i \neq j$ and $\bigcup_{i=1}^{s} s_i \subseteq \mathcal{V}$.*

The task vehicle $j \in v^T$ initiates the process of coalition formation, when the associated RSU is overloaded and not able to complete the task before the deadline. The task vehicle sends the request for task execution to the nearby vehicles. The RSU also helps in finding out the cooperative vehicles that move in the same route r_j. The interested cooperative vehicles $k \in v^c$ acknowledges their interest to the task vehicle j for joining the coalition s_j. Each coalition s_j has its own threshold criteria for the vehicles, specifying minimum trust $\top_j$ and computational requirement c_k for vehicle k. The task vehicle j maintains all the information required for coalition formation.

4.2 Utility Functions

A cooperative vehicle helps the task vehicle inside the coalition in task execution and tries to maximize its own utility. The utility of a cooperative vehicle k inside coalition s_j can be computed as:

$$u_k(s_j) = R_k(s_j) - C_k(s_j) \tag{10}$$

where $R_k(s_j)$ is the generated revenue of vehicle k from coalition s_j and $C_k(s_j)$ is the corresponding cost of vehicle k.

The revenue generated by vehicle k by contributing in the task execution of coalition s_j is computed as:

$$R_k(s_j) = (\top_j - \tau_k)w_1\phi_\tau + (f_k\Delta_e)w_2\phi_e \tag{11}$$

where f_k is the CPU clock cycle of vehicle k, Δ_e is the execution delay, w_1, w_2 are the weighted factors, and ϕ_τ and ϕ_e are the incentives.

The cost of vehicle k in coalition s_j is computed as:

$$C_k(s_j) = [\rho_k\Delta_t + \varrho_k f_k^2\Delta_e]\phi_\varepsilon \tag{12}$$

The first term in Equation (12) represents the transmitting energy cost and the second term represents the energy

cost for task execution. Here, ρ_k is the transmitting power, ϱ_k is the energy coefficient of vehicle k, and ϕ_ε is the cost per unit energy consumption.

The utility of a coalition s_j is the total utility of all the cooperative vehicles in s_j subtracted by the coalition formation cost C_j^{form}.

$$U(s_j) = \sum_{k \in s} \Big[R_k(s_j) - C_k(s_j) \Big] - C_j^{form} \tag{13}$$

It is necessary to synchronize all the vehicles in the coalition in order for smooth data flow inside the coalition. The expression to determine the coalition formation cost can be given as [5]:

$$C_j^{form} = \begin{cases} \alpha . \, |s_j| \; ; & if, s_j > 1 \\ 0 \; ; & otherwise \end{cases} \tag{14}$$

where α is the pricing factor and $|s_j|$ is the size of the coalition s_j. The coalition formation controls the utility of the coalition. As the number of vehicles increases in a coalition, the coalition formation cost linearly increases to coordinate properly with all the vehicles. Therefore, a tradeoff between coalition size and the coalition utility is enforced. Hence, a maximum optimal size of the coalition must exist, which can be seen in the coalition formation algorithm.

The cooperative vehicles always select a coalition, which maximizes their payoffs. A good payoff also increases the vehicle's trust score, which helps find the cooperative vehicles later when the vehicle has some task to offload.

The binary decision variable q_{kj} represents the vehicle's status, whether the vehicle k has joined the coalition s_j or not, represented as:

$$q_{kj} = \begin{cases} 1; & if \quad k \in s_j \\ 0; & otherwise \end{cases} \tag{15}$$

The optimization problem for each vehicle k which wants to cooperate a task vehicle by joining a coalition is defined in Equations (16) - (20).

$$\max_{q_{kj}} \left[u_k(s_j) \, q_{kj} \right] \tag{16}$$

$$\text{s.t.} \quad u_k(s_j) \succeq u_k(s_m) \; ; \; \forall m \in S / \{s_j\} \tag{17}$$

$$\sum_{s_j} r_{kj} \geq 1 \; ; \; \forall s_j \in S, \; \forall k \in s_j \tag{18}$$

$$\sum_{s_j} q_{kj} = 1 \; ; \; \forall s_j \in S, \; \forall k \in v^c \tag{19}$$

$$|s_j| \leq |s_m| \; \forall \, m \in S / \{s_j\} \tag{20}$$

The objective function is defined in Equation (16), where we want to maximize the payoff of a vehicle. Each cooperative vehicle prefers the coalition with maximum utility, which is specified in Equation (17). The cooperative vehicles can choose a coalition if and only if the task vehicle in that coalition follows the same route. However, there can be many coalitions $s_j \in S$ in which the task vehicles have the same route as the cooperative vehicle k, represented as binary variable $r_{kj} = 1$, enforced in Equation (18). The payoff of a vehicle k depends on its choice to be a coalition member S_j, represented through variable q_{kj}. At any time instant, a vehicle can be a part of only one coalition, which we enforce as a constraint in Equation (19). A vehicle always prefers a coalition with fewer members over a coalition with more vehicles, as specified in Equation (20).

4.3 Hedonic Coalition Formation Game for Cooperative Task Execution

The proposed solution approach, cooperative task execution in IoV, considers the formation of disjoint and distributive hedonic coalitions. Two main characteristics classify the considered coalitional game as hedonic and defined in the following way [22].

Definition 3. ***(Vehicle Hedonic Coalition)*** *: A vehicle coalition formation game is classified as hedonic if (i) the payoff of any vehicle in a coalition only depends on the other cooperative vehicles of the same coalition, and (ii) the formation of coalitions is the outcome of the preferences of the cooperative vehicles over the coalition set S.*

The conditions specified in the above definition fully apply to our coalition formation approach, where the utility of vehicle $u_k(s_j)$ in coalition s_j depends only on other cooperative vehicles $v^c \in s_j$. The vehicles autonomously select the coalition based on their preferences, defined as [5]:

Definition 4. ***(Preference Relation)****: For any cooperative vehicle $k \in v^c$, a preference relation or order $\succeq_k$ is defined as a complete, reflexive, and transitive binary relation over the task vehicle coalitions $s_j \in S$ that vehicle k can belong.*

The proposed cooperative task execution approach, CoTEV, involves the formation of disjoint coalitions. In any coalition, each vehicle has its own well-defined preference and liberty to decide whether to join or leave any coalition. Hence, for any given vehicle $k \in v^c$, the preference relation $s_1 \succeq_k s_2$ implies that vehicle k prefers to join the coalition s_1 over coalition s_2 or at least prefers both s_1 and s_2 equally, where $s_1, s_2 \in S$. In equally preferences, if vehicle k is in coalition s_2 then it will still not prefer to join the other coalition s_1. For migration from one coalition to another, there should be strict preference that s_1 is anyhow better than s_2, i.e., $s_1 \succ s_2$ [30].

Theorem 1. *In the proposed hedonic coalition formation game, a resulted coalition s_j is limited by the maximum number of cooperative vehicles $M_j = \left\lceil \frac{R(s)}{C'(s)} \right\rceil$*

Proof. For coalition formation, improvement of both vehicle utility $u_k(s_j)$ and coalition utility $U(s_j)$ is required. However, the coalition formation cost limits the maximum utility of a coalition. When the number of vehicles in a coalition increases, the coalition formation cost increases linearly, decreasing the coalition utility. A minimum cost function is modeled by the barrier function, given as [31]:

$$C_0(s_j) = \begin{cases} C_k(s_j) + k(s_j); if \, |s_i| \leq M_j \\ \infty \quad ; \qquad if \, |s_j| > M_j \end{cases} \tag{21}$$

The barrier cost function $C_0(s_j)$ states that when the size of coalition s_j is beyond M_j, the cost goes to infinity, and

the coalition does not form. Otherwise, the cost function depends on the penalty function $k(s_j)$, computed as:

$$k(s_j) = \begin{cases} 0; & if\ |s_i| \leq M \\ \infty; & if\ |s_i| > M \end{cases} \tag{22}$$

To find the maximum size of the coalition s_j, we consider the revenue of the coalition and the cost of vehicle k being in coalition s_j. An ideal scenario exists, where each vehicle equally contributes to the coalition and possesses the same trust score and high computational ability. Another condition for the ideal scenario is both the task and cooperative vehicles are in the V2V range. Therefore, considering the ideal conditions, the revenue $R_0(s_j)$ and cost equations $C_0(s_j)$ are given as:

$$R_0(s_j) = \left[\sum_{k \in S} (\tau_j) w_1 + (f_k \Delta_e) w_2 \right] \phi \tag{23}$$

$$C_0(s_j) = C_k(s) + C_k^{form} \tag{24}$$

Hence, the maximum size of a particular coalition is computed as:

$$|s_j| \leq \left\lceil \frac{R_0(s_j)}{C_0(s_j)} \right\rceil = M_j \tag{25}$$

where M_j is the Maximum size of coalition s_j. □

Considering the maximum size of a coalition and properties of hedonic coalition, the optimization problem of each vehicle coalition is described in Equations (26)-(30).

$$\max_{q_{kj}} U(s_j), \quad \forall k \in v^c \tag{26}$$

$$\text{s.t.} \quad \sum_k q_{kj} = 1 \ \ \forall s_j \in s, \ \forall k \in v^c \tag{27}$$

$$|s_j| \leq \ M_j \ ; \ \forall \ m \in S/\{s_j\} \tag{28}$$

$$\tau_k \geq \top_j \ ; \ \forall k \in v^c, \ \forall s_j \in S \tag{29}$$

$$\sum_{k \in s_j} f_k \geq c_j \tag{30}$$

The objective function is defined in Equation (26), where we want to maximize the coalition utility. The payoff of a coalition s_j depends on all the cooperative vehicles k in the coalition s_j. A vehicle k can join only one coalition at any instant, which is enforced in Equation (27). The number of cooperative vehicles in a coalition can not exceed the maximum coalition size M_j, specified in Equation (28). In Equation (29), we specify that the trust score of a vehicle must not be less than the coalition trust score. Finally, the combined computation capabilities of all the cooperative vehicles must be greater than the CPU requirement of the task vehicle, as mentioned in Equation (30).

Algorithm 1 shows the steps for the hedonic coalition formation between the task vehicles and cooperative vehicles. Each task vehicle in the network starts the coalition formation with itself as the initial member. Each cooperative vehicle k, which will at least follow the same route as the task vehicle, can join the coalition. Furthermore, vehicle k should possess the trust and computational requirements as specified in Step 5 in the algorithm. A cooperative vehicle always prefers a coalition with a smaller size. Moreover, a vehicle always joins a coalition, giving the maximum vehicle utility. In Step 6, vehicle k checks for all possible coalitions s_m and selects the best coalition. In this way, a cooperative vehicle enters into a coalition. However, after entering into a coalition s_j, vehicle k can prefer another coalition s_i, where the utility is maximized. A coalition is stable only after all switch operations are completed. The coalition stability is discussed in Algorithm 2.

Algorithm 1 Hedonic Coalition Formation of Vehicles

Inputs:
Vehicle and task information: v^T, v^c, C_x, f_k, r_k, τ_k
Output:
The coalition structure S

1: Coalition formation initiated by task vehicles v^T
2: Consider single member coalition s_j with v^T
3: **for** each $k \in v^c$ **do**
4: **if** $(r_{kj} == 1)$ && $(|s_j| \leq |s_m|), \ \forall \ m \in S/\{s_j\}$ **then**
5: **if** $(\tau_k \geq \top_j)$ && $(C_{x_k} \leq f_k)$ **then**
6: Vehicle k joins s_j over s_m when $u_k(s_j) \geq u_k(s_m)$
7: **end if**
8: **end if**
9: **end for**
10: **return** S

4.4 Convergence Analysis and Coalition Stability

The proposed algorithm 1 results in a coalition structure with disjoint coalitions of vehicles. The task execution only starts after a coalition is stable, which is achieved through the *switch* operations performed by the vehicles [5].

4.4.1 Convergence Analysis

Any cooperative vehicle $k \in v^c$, can easily perform a switch operation to leave its current coalition and join the new coalition if the condition specified in (17) is satisfied. On the other hand, a task vehicle $j \in v^T$ can also split from a cooperative vehicle if another better merge option is available. However, the task execution starts when the the game converges, which is guaranteed in the proposed approach. The steps for coalition convergence and stability are specified in Algorithm 2.

Theorem 2. *The proposed coalitional formation game always converges from an initial phase of a coalition $\Pi_{initial}$ to a final stable individual coalition Π_{final}.*

Proof. Every switch operation performed by a vehicle brings the coalition into an intermediate stage Π_q, where q denotes q^{th} switch operation associated with the coalition. Therefore, we can write $\Pi_{initial} \rightarrow \Pi_1 \rightarrow \Pi_2 \rightarrow ... \rightarrow \Pi_{final}$, where the operator $\rightarrow$ indicates a switch operation. Considering the preference relation of a cooperative vehicle in 17, it can be seen that every switch operation results in a unique intermediate stage of the coalition, that has not yet occurred. Between any two transformations Π_x and Π_y, such that $x \neq y$, the number of switch operations is $y - x$. As the number of cooperative vehicles is *finite*, the number of transformations is also *finite*, which in turn, shows that the coalition will always converge to a final partition Π_{final}.

Further, considering the convergence of all coalitions, the total network also converges to a final partition. □

4.4.2 Coalition Stability

A coalition $s_j \in s$ is stable if: i) no vehicle k in coalition s_j can improve its own utility by switching from the coalition or working independently and ii) no other vehicle coalition $s_i \in s$ can increase its utility by merging with another coalition s_j, $i \neq j$, $\forall s_i, s_j \in s$.

Algorithm 2 Stability of Vehicle Coalitions

Inputs:
Coalition structure S
Output:
Stable coalition structure S
1: **if** $|s_j|$ is insufficient **then**
2: *Merge* with coalition s_i with insufficient members
3: **end if**
4: **for** each coalition $s_j \in S$ **do**
5: **while** $Converge[s_j] = 0$ **do**
6: **for** Each vehicle $k \in s_j$ **do**
7: **if** $u_k(s_j) < u_k(s_i), \forall s_i \in S$ **then**
8: vehicle k *switches* from s_j to s_i
9: **end if**
10: **if** $U(s_j - k) > U(s_j)$ **then**
11: s_j *splits* from vehicle k
12: **end if**
13: **end for**
14: **end while**
15: **while** No $switch$ possible **do**
16: Set $Converge[s_j] = 1$
17: Start cooperative task execution
18: **end while**
19: **end for**
20: Return stable coalition structure S

Through the steps in Algorithm 2, the stability of vehicle coalitions is achieved. First, the merge operations take place between the coalitions with insufficient numbers. This step helps in satisfying the requirements of some tasks. Then, in the coalitions where stability is not achieved, each vehicle carries out the switch operations independently. If a vehicle k in s_i gets more payoff from coalition s_i then it switches from s_j to s_i. This operation continues till all the options are verified for vehicle k. It may happen that by joining another coalition, vehicle k maximizes the payoff; however, the coalition utility decreases. In this case, the whole coalition members may split from vehicle k, as specified in Steps 10–12. When no switch operations are possible inside a coalition, stability is achieved, and the task execution starts.

Time complexity: The complexity of the algorithms depends on the number of coalitions in the same routes and the number of cooperating vehicles participating in task execution. In Algorithm 1, each vehicle checks for all available coalitions that satisfy the required criteria. Therefore, in the worst case, a vehicle checks $O(|v^T_{r_k}|)$ number of coalitions with the same route of vehicle k, where $v^T_{r_k}$ represents the set of task vehicles forming the coaliition in route r_k. However, the coalition's stability depends on all the switch operations in Algorithm 2. Steps 1-3 take $O(|v^T|)$ time to check the coalitions with insufficient vehicles. Then, for each coalition s_j, the maximum switch operations is $O(|v^c| \times |v^T| - 1)$ in worst case. In other words, at any time instant, the stability over all the coalitions with same route r_k can be achieved in $O(|v^c_{r_k}| \times |v^T_{r_k}|)$.

5 Performance Evaluation

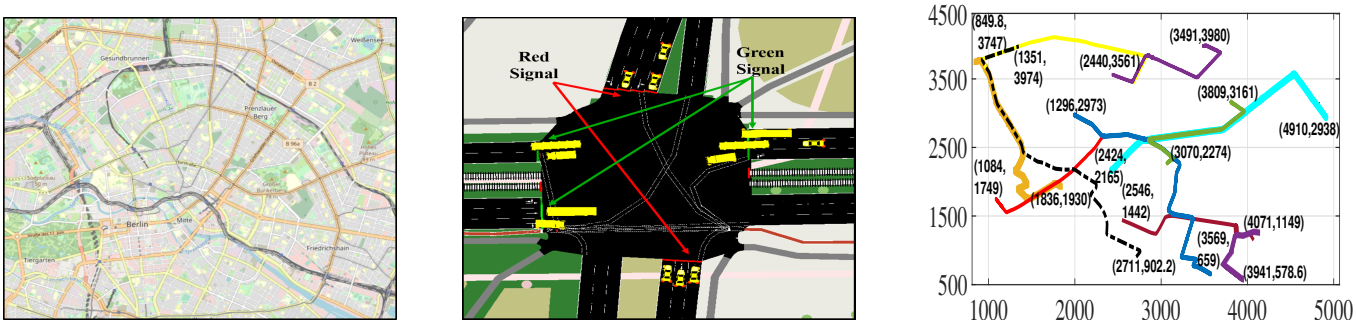
In this section, we study the effectiveness of the proposed method for cooperative and trustworthy task execution in IoV (CoTEV). We use the simulation of urban mobility (SUMO) tool [32] to generate the vehicle mobility traces for a selected area of Berlin city. The underlying technology of SUMO considers a road network as graph, where the road junctions are represented as nodes, and the edges represents the roads. SUMO uses the car-following model to control the speed of the vehicles. Figure 2(a) shows the selected area of Berlin city considered as the road network in SUMO. Figure 2(b) represents a road junction, where SUMO considers the traffic signals while generating the mobility traces. From the mobility traces, considering the starting and ending point of the vehicles, we find out the routes. Figure 2(c) shows different routes in the network. We use the IEEE 802.11p communication standard with channel bandwidth of 10 MHz, which is widely used for V2X communication. We use the mobility traces of 3100 vehicles (including trucks, buses, and cars) in MATLAB to form coalitions. During coalition formation, we consider 38-620 vehicles in each route, where route length lies in the range of 0.7 - 3.4 KM. For performance evaluation, we consider 100 iterations for each metric and show the results with 95% confidence interval. The parameters considered for simulation are listed in Table 3.

TABLE 3: Simulation Parameters

Parameters	Values
Number of iterations	100
Simulation Area	5 KM × 4.5 KM
Number of RSUs	56
Range of RSU	500 m
Number of vehicles	3100
Route length	0.7 – 3.4 KM
$\tau, \top$	[0 – 1] [9]
D	[2 – 5] MB [33]
ρ	10 mW [34]
F_v	[200 – 300] MHz [17]
C_x	[1100–1600] $\times 10^6$ CPU cycles [18]
N_o	0.1 mW [34]
γ	2 [16]
Bandwidth	10 MHz [27]
a	10 [5]
w_1, w_2, ϕ	0.33, 0.66, 15

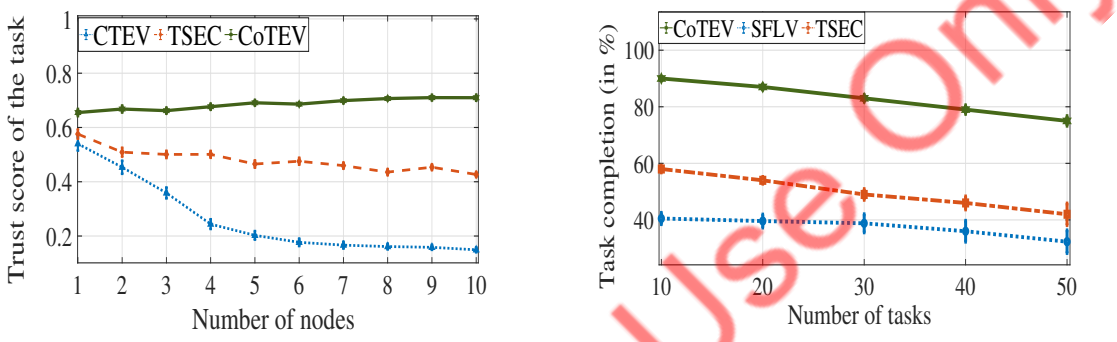
5.1 Benchmark Schemes and Performance Metrics

To evaluate the performance of the proposed approach, we consider some recently proposed methods on cooperative and trustworthy vehicle task execution, as the benchmark schemes. First, a game theoretical approach of task execution in mobile edge computing (GTOM) [28] is considered, where the offloaded task is being scheduled and executed among the MEC controllers. As a second benchmark scheme, we consider the adaptive task offloading approach



(a) (b) (c)

Fig. 2: (a) Considered area of Berlin city in SUMO, (b) A road junction with traffic signals in SUMO, and (c) Different routes in the selected area of Berlin



(a) Trust score vs. Number of cooperative nodes

(b) Task completion rate vs. Number of tasks

Fig. 3: Trust score and successful task completion rate

(ATOA) [12] in a heterogeneous IoV environment for time-critical task processing. A sustainable fog-federated load sharing in IoV (SFLV) [33] is considered to show the performance of the proposed scheme for delay-sensitive tasks in a cooperative vehicular environment. Finally, to show the trustworthiness in task execution, we consider a trust-based cooperative approach for secure collaboration in IoV (TSEC) [9]. Henceforth, we use the abbreviations of the proposed scheme and benchmark schemes to describe the results.

In this paper, we consider the following metrics to evaluate the performance of the proposed algorithms.

i) Average trust score of executed task: It represents the average trust score of the cooperative nodes that participate in executing a vehicle's task.

ii) Task completion rate: This metric represents the percentage of deadline-based tasks completed before the maximum allowable delay.

iii) Average delay per task: It represents the average delay in completing a task. We evaluate the task delay with three varying parameters – task size, CPU cycle requirement of task, and the number of task vehicles in the network.

iv) Average energy consumption per task: This metric specifies the average energy consumed to complete a task with the parameters considered for delay assessment.

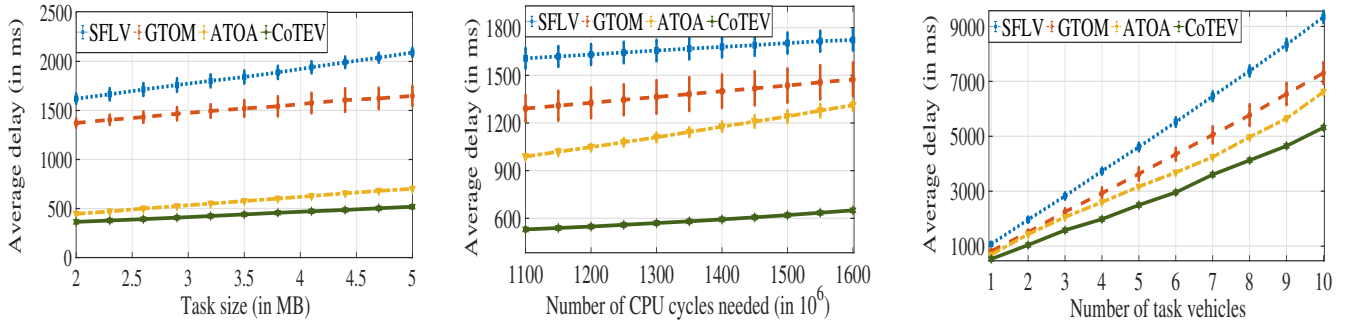
v) Average cost per task: To access vehicle utility, we consider the average cost of completing the task of a vehicle.

vi) Average profit per task: It represents the average profit a coalition earns after successfully executing a task.

5.2 Results

i) Average trust score of executed task: Figure 3(a) shows the trust score of an executed task with the varying number of cooperative nodes (either RSU or vehicle). In the figure, except for the benchmark scheme TSEC, we obtain the same result for all other benchmark schemes - GTOM, SFLV, and ATOA. Therefore, we represent them with the common name CTEV which means cooperative task execution in IoV. We observe that the trust score of both the benchmark schemes – CTEV and TSEC decreases with increased cooperative nodes. In CTEV, the trustworthiness of the nodes is not considered. Therefore, with an increase in the demand for task-executing nodes, the chance of selecting the nodes with low trust scores also increases, leading to a low average trust score for the task. On the other hand, the TSEC approach considers the trust scores of the vehicles that cooperate in task execution and performs better than CTEV. The trust score of a vehicle also depends on the computational ability of the vehicle, which is not considered in the TSEC approach. Due to this reason, TSEC has a lower trust score compared to CoTEV.

ii) Task completion rate: Due to the heavy task load in the RSUs, the number of successful task requests decreases. If we consider task execution through cooperative vehicles, the success rate depends on the vehicle density and cooperativeness of the vehicles in the routes of the task vehicles. In Figure 3(b), the rate of successful task completion of different approaches with the different number of task vehicles is shown. From the figure, we observe that the

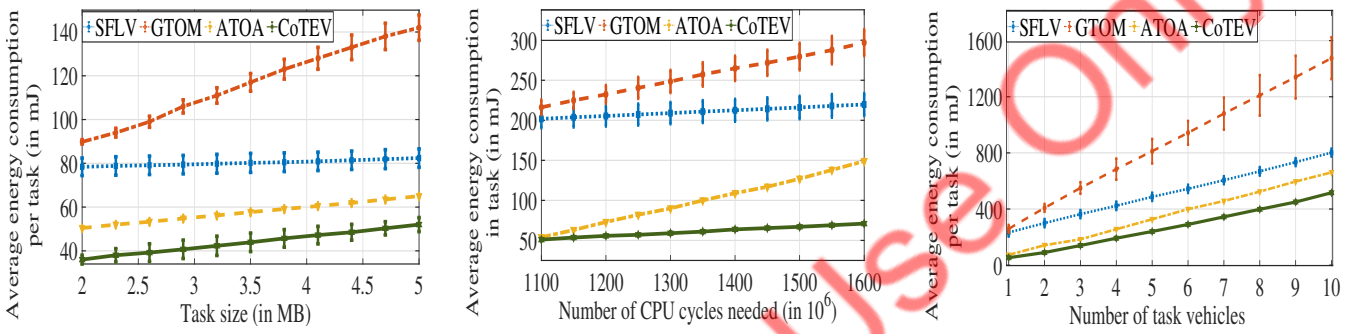


(a) Total delay in a task vs. Task size

(b) Total delay in a task vs. Number of CPU cycles needed

(c) Total delay in a task vs. Number of task vehicles

Fig. 4: Delay vs. different parameters



(a) Average energy consumption vs. Task size

(b) Average energy consumption in a task vs. Number of CPU cycles needed

(c) Average energy consumption in a task vs. Number of task vehicles

Fig. 5: Average energy consumption vs. different parameters

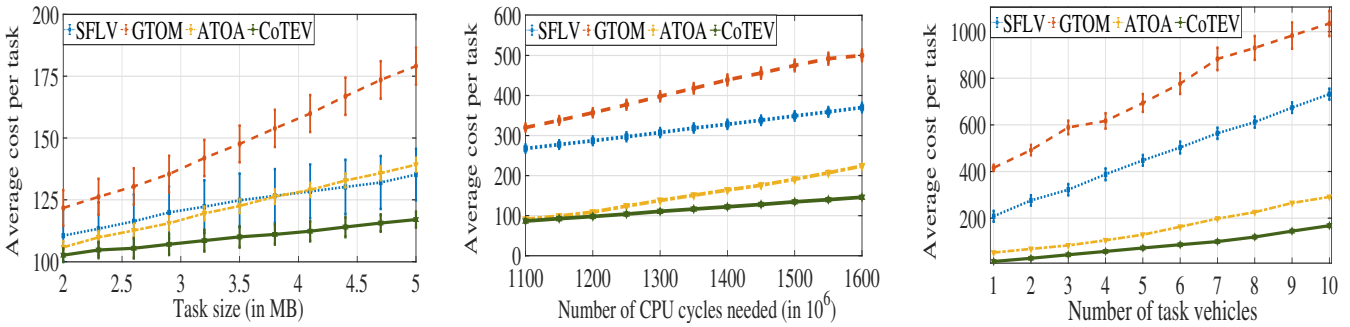
SFLV approach has a low task completion rate compared to the other two approaches due to the task execution by the overloaded RSUs. In TSEC, the task completion rate is higher than SFLV because it considers the task offloading to nearby vehicles, which have a higher trust score, increasing the possibility of successful task execution. However, there are other factors such as vehicle mobility, charged price, and computational requirements that decide the completion of the delay-sensitive tasks, which are not considered in TSEC. In CoTEV, we consider all the parameters required for task execution during the formation of coalitions of vehicles. We consider coalition formation only when the RSU is overloaded, or the cost is high. Through the cooperation of all the trustworthy nodes in the network, we achieve the highest task completion rate compared to SFLV and TSEC.

iii) Average delay per task: Figure 4(a) shows the average delay of a task with varying task sizes. From the figure, we observe that the overall delay of all the schemes increases with an increase in the task size. SFLV considers only RSUs for task execution due to the high waiting delay for increased task queue at the RSUs. In GTOM, the task delay is associated with many factors such as transmission, waiting, and execution. However, due to the use of a task scheduling algorithm, GTOM performs better than SFLV. In ATOA, multiple task execution layers are considered for task execution, and therefore, it performs better compared to the other benchmark schemes. In the proposed approach, CoTEV, the merging of coalitions increases the number of cooperating vehicles to execute the tasks with lower delay.

Figure 4(b) shows the average delay of a task with different requirements of CPU cycles for the task. We vary the number of required CPU cycles in the range of [1100 - 1600] $\times$ 10^6. In CoTEV, we consider the cooperative vehicles that are capable of executing a task based on its CPU cycle requirement. Due to this reason, a task with high CPU requirements is sent to a capable vehicle. However, in the benchmark schemes, there is no such provision. Furthermore, a high CPU requirement task increases the waiting delay for other tasks in the queue.

Figure 4(c) shows the trend of task delay due to the varying number of task vehicles. With the increase in the task vehicles, the demand for cooperative vehicles also increases. Therefore, the waiting delay increases to find out the cooperative vehicles for a task. Due to the merging of coalitions based on the task deadlines, in CoTEV, the waiting delay is less compared to the other schemes. ATOA tries to execute the tasks at the cloud if the load is high at the lower level nodes, with a penalty of transmission delay and considerably reduces the overall delay.

iv) Average energy consumption per task: Energy consumption for a task depends mainly on execution energy and transmitting energy. Figure 5(a) shows the average energy consumption for a task with a different task size. When the size of a task increases, the energy required to transmit the task also increases. In CoTEV, we try to offload the task to the nearby vehicles and therefore, possess less transmission energy compared to other methods. However, in GTOM, an increase in the task size increases both transmitting and



(a) Average cost per task vs. Task size

(b) Average cost spent on a task vs. Number of CPU cycles needed

(c) Average cost spent on a task vs. Number of task vehicles

Fig. 6: Average cost spent on a task vs. different parameters

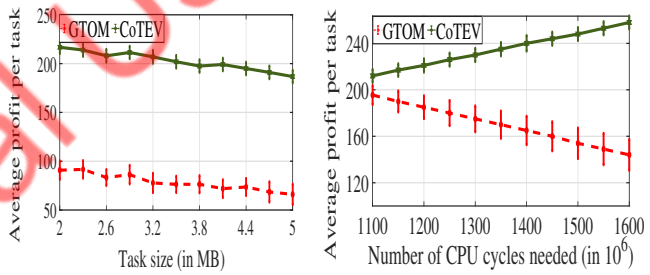
execution energy; therefore, it greatly affects total energy. However, a small task size may require high CPU cycles and a large task may require very few CPU cycles.

For the high CPU requirement task, CoTEV tries to find out more cooperative vehicles and parallelly execute the task in the coalition. Moreover, the energy consumption of the vehicles is very less compared to the RSUs and cloud. This is the increase due to which the execution energy rate is low in CoTEV, compared to GOTM and ATOA. In GTOM, the CPU cycles and task size are directly proportional to the CPU cycle demand of a task, which increases the transmitting energy and execution energy. In ATOA, due to the execution of some tasks in high servers in the cloud, the execution energy consumption is high. SFLV has a similar trend to CoTEV due to the task execution only at the RSUs.

Figure 5(c) shows the average energy consumption in completing a task with varying the number of task vehicles. An increase in task vehicles also increases the execution and transmitting energy to complete more tasks. Due to the execution of tasks through cooperating vehicles, the rate of energy consumption is less in CoTEV, compared to other benchmarks where the execution of some tasks is carried out at the cloud.

v) Average cost per task: The cost for executing a task depends on the energy consumption and the price charged by the executing entity for completing the task. Figure 6(a) shows the average cost spent per task with varying task sizes. Due to the execution of many tasks through nearby cooperative vehicles, the cost of task execution is less in CoTEV. Due to high transmission and execution energy for cloud-based task execution, the rate of increase in cost is high in ATOA. In the GTOM approach, the energy consumption is directly associated with the task size as another transmitting cost, which increases the cost with an increase in task size. In SFLV, the cost is high due to the monopoly of RSUs in task execution.

For high-intensive tasks, the task executing nodes usually charge a high price to the vehicles. However, the vehicles always try to retain good trust scores. For a vehicle with a good trust score, it is easier to find cooperative vehicles for their task execution than those with low trust scores. Figure 6(b) shows the average cost per task with varying CPU cycle requirements. In CoTEV, we try to execute more tasks through the vehicles, which reduces the average cost per task. Other benchmark schemes possess high costs due to the high price charged by the RSUs or cloud, when the CPU requirement for a task is high. Similarly, when the number of task vehicles increases, more tasks arrive at the RSUs, which in turn, increases the average cost of task execution. Due to the cooperation of the vehicles, in CoTEV, we observe a low increasing trend in cost compared to other schemes, as shown in Figure 6(c).



(a) Average profit of task execution vs. Task size

(b) Average profit of task execution vs. Number of CPU cycles

Fig. 7: Average profit vs. different parameters

vi) Average profit per task: Out of all the benchmark schemes, only the GTOM scheme considers a profit model. we compare the profit of CoTEV with the profit of the GTOM approach. Figure 7(a) shows the average profit per task when the task size varies. With an increase in the task size, the transmitting energy increases, reducing the profit. In GTOM, due to a limited budget per task, when the cost increases, the profit gets reduced. In CoTEV, the profit reduces due to less revenue generation.

In GTOM, the number of CPU cycles required for a task is directly proportional to the task size. Therefore, with a high CPU requirement for the task, the profit reduces. In CoTEV, with an increase in CPU requirement, more cooperative vehicles participate in task execution. This reason eventually increases the profit of the vehicles, as shown in Figure 7(b).

6 Conclusion

In this paper, we proposed a trustworthy and cooperative task execution environment among the task vehicles and the cooperative vehicles. When the RSUs are overloaded,

the task vehicle takes the help of the cooperative vehicles based on their trust scores, computational capabilities, and journey routes. We proposed a coalition formation game-based approach, which allows the task vehicle to find a suitable group of cooperative vehicles for task execution. Through switch operations, the vehicles maximize their own utilities and finally form disjoint coalitions in a distributed manner. After each task completion, the trust scores of the cooperative vehicles are updated. We also showed the coalition stability and maximum coalition size for a task. We considered the mobility traces of a real road map through SUMO for the performance assessment of the proposed algorithms. From the results, we showed that the proposed approach significantly increases the task completion rate of delay-sensitive tasks with high utility for the cooperative vehicles. In future, we plan to extend the work to a vehicle scenario, where the vehicle task has dependency among the task fragments. We plan to consider the cooperative vehicle scenario and execute the delay-sensitive tasks with dependency through vehicle collaborations.

REFERENCES

[1] Y. Wang, P. Lang, D. Tian, J. Zhou, X. Duan, Y. Cao, and D. Zhao, "A game-based computation offloading method in vehicular multiaccess edge computing networks," *IEEE Internet of Things Journal*, vol. 7, no. 6, pp. 4987–4996, 2020.

[2] B. Wang, L. Wang, G. Fu, W. Liu, and J. Cui, "A new distributed coalition formation algorithm for cooperation in ad hoc networks," in *2018 IEEE 4th International Conference on Computer and Communications (ICCC)*. IEEE, 2018, pp. 446–450.

[3] L. S. Vailshery, "Internet of things (IoT) and non-IoT active device connections worldwide from 2010 to 2025," *Online, https://bit.ly/2SwKVuB*, 2021.

[4] R. Wu, G. Tang, T. Chen, D. Guo, L. Luo, and W. Kang, "A profit-aware coalition game for cooperative content caching at the network edge," *IEEE Internet of Things Journal*, vol. 9, no. 2, pp. 1361–1373, 2021.

[5] W. Saad, Z. Han, A. Hjorungnes, D. Niyato, and E. Hossain, "Coalition formation games for distributed cooperation among roadside units in vehicular networks," *IEEE Journal on Selected Areas in Communications*, vol. 29, no. 1, pp. 48–60, 2010.

[6] S. El Madani, S. Motahhir, and A. El Ghzizal, "Internet of vehicles: concept, process, security aspects and solutions," *Multimedia Tools and Applications*, pp. 1–25, 2022.

[7] H. Li, X. Li, M. Zhang, and B. Ulziinyam, "Multicast-oriented task offloading for vehicle edge computing," *IEEE Access*, vol. 8, pp. 187 373–187 383, 2020.

[8] Z. Zhou, C. Gao, C. Xu, Y. Zhang, S. Mumtaz, and J. Rodriguez, "Social big-data-based content dissemination in internet of vehicles," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 2, pp. 768–777, 2017.

[9] T. Halabi and M. Zulkernine, "Trust-based cooperative game model for secure collaboration in the internet of vehicles," in *ICC 2019-2019 IEEE International Conference on Communications (ICC)*. IEEE, 2019, pp. 1–6.

[10] Y. Cui, L. Du, P. He, D. Wu, and R. Wang, "Multi-vehicle intelligent collaborative computing strategy for internet of vehicles," in *2022 IEEE Wireless Communications and Networking Conference (WCNC)*. IEEE, 2022, pp. 1647–1652.

[11] F. Jiang, W. Liu, J. Wang, and X. Liu, "Q-learning based task offloading and resource allocation scheme for internet of vehicles," in *2020 IEEE/CIC International Conference on Communications in China (ICCC)*. IEEE, 2020, pp. 460–465.

[12] B. K. Ray, A. Saha, S. Khatua, and S. Roy, "Quality and profit assured trusted cloud federation formation: Game theory based approach," *IEEE Transactions on Services Computing*, vol. 14, no. 3, pp. 805–819, 2018.

[13] H. E. Manoochehri and R. Z. Wenkstern, "Dynamic coalition structure generation for autonomous connected vehicles," in *2017 IEEE International Conference on Agents (ICA)*. IEEE, 2017, pp. 21–26.

[14] H. Zhang, Y. Yongjin, B. Shang, and P. Zhang, "Joint resource allocation and multi-part collaborative task offloading in mec systems," *IEEE Transactions on Vehicular Technology*, 2022.

[15] B. Ko, K. Liu, S. H. Son, and K.-J. Park, "Rsu-assisted adaptive scheduling for vehicle-to-vehicle data sharing in bidirectional road scenarios," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 2, pp. 977–989, 2020.

[16] X. Hao, M. H. Cheung, V. W. Wong, and V. C. Leung, "Hedonic coalition formation game for cooperative spectrum sensing and channel access in cognitive radio networks," *IEEE Transactions on Wireless Communications*, vol. 11, no. 11, pp. 3968–3979, 2012.

[17] P. Liu, J. Li, and Z. Sun, "Matching-based task offloading for vehicular edge computing," *IEEE Access*, vol. 7, pp. 27 628–27 640, 2019.

[18] C. Chen, H. Li, H. Li, R. Fu, Y. Liu, and S. Wan, "Efficiency and fairness oriented dynamic task offloading in internet of vehicles," *IEEE Transactions on Green Communications and Networking*, 2022.

[19] W. Xu, H. Wu, J. Chen, W. Shi, H. Zhou, N. Cheng, and X. S. Shen, "Vifi: Vehicle-to-vehicle assisted traffic offloading via roadside wifi networks," in *2018 IEEE Global Communications Conference (GLOBECOM)*. IEEE, 2018, pp. 1–6.

[20] I. Jang, H.-S. Shin, and A. Tsourdos, "Anonymous hedonic game for task allocation in a large-scale multiple agent system," *IEEE Transactions on Robotics*, vol. 34, no. 6, pp. 1534–1548, 2018.

[21] A. Dutta, V. Ufimtsev, T. Said, I. Jang, and R. Eggen, "Distributed hedonic coalition formation for multi-robot task allocation," in *2021 IEEE 17th International Conference on Automation Science and Engineering (CASE)*. IEEE, 2021, pp. 639–644.

[22] W. Saad, Z. Han, T. Basar, M. Debbah, and A. Hjorungnes, "Hedonic coalition formation for distributed task allocation among wireless agents," *IEEE Transactions on Mobile Computing*, vol. 10, no. 9, pp. 1327–1344, 2010.

[23] H. Guo, L.-l. Rui, and Z.-p. Gao, "V2v task offloading algorithm with lstm-based spatiotemporal trajectory prediction model in svcns," *IEEE Transactions on Vehicular Technology*, vol. 71, no. 10, pp. 11 017–11 032, 2022.

[24] Y. Saleem, N. Mitton, and V. Loscri, "A qos-aware hybrid v2i and v2v data offloading for vehicular networks," in *2021 IEEE 94th Vehicular Technology Conference (VTC2021-Fall)*. IEEE, 2021, pp. 1–5.

[25] J. Shi, J. Du, J. Wang, and J. Yuan, "Deep reinforcement learning-based v2v partial computation offloading in vehicular fog computing," in *2021 IEEE Wireless Communications and Networking Conference (WCNC)*. IEEE, 2021, pp. 1–6.

[26] G. Cui, Y. Long, L. Xu, and W. Wang, "Joint offloading and resource allocation for satellite assisted vehicle-to-vehicle communication," *IEEE Systems Journal*, vol. 15, no. 3, pp. 3958–3969, 2020.

[27] C. Chen, Y. Zeng, H. Li, Y. Liu, and S. Wan, "A multihop task offloading decision model in MEC-enabled internet of vehicles," *IEEE Internet of Things Journal*, vol. 10, no. 4, pp. 3215–3230, 2023.

[28] H. Teng, Z. Li, K. Cao, S. Long, S. Guo, and A. Liu, "Game theoretical task offloading for profit maximization in mobile edge computing," *IEEE Transactions on Mobile Computing*, 2022.

[29] B. Grewal and J. Grewal, "Higher engineering mathematics," *2002, Khanna Publishers, New Delhi*, 1996.

[30] W. Saad, Z. Han, M. Debbah, A. Hjorungnes, and T. Basar, "Coalitional game theory for communication networks," *IEEE signal processing magazine*, vol. 26, no. 5, pp. 77–97, 2009.

[31] J. Proakis and M. Salehi, "Digital communications 4th ed. mcgraw-hill," *New York*, 2001.

[32] P. A. Lopez, M. Behrisch, L. Bieker-Walz, J. Erdmann, Y.-P. Flötteröd, R. Hilbrich, L. Lücken, J. Rummel, P. Wagner, and E. Wießner, "Microscopic traffic simulation using sumo," in *The 21st IEEE International Conference on Intelligent Transportation Systems*. IEEE, 2018. [Online]. Available: https://elib.dlr.de/124092/

[33] Z. Sharmin, A. W. Malik, A. U. Rahman, and R. M. Noor, "Toward sustainable micro-level fog-federated load sharing in internet of vehicles," *IEEE Internet of Things Journal*, vol. 7, no. 4, pp. 3614–3622, 2020.

[34] W. Saad, Z. Han, M. Debbah, A. Hjorungnes, and T. Basar, "Coalitional games for distributed collaborative spectrum sensing in cognitive radio networks," in *IEEE INFOCOM 2009*. IEEE, 2009, pp. 2114–2122.