# SCALABLE AND ENERGY EFFICIENT TASK OFFLOADING SCHEMES FOR VEHICULAR CLOUD COMPUTING

Mohammad Pasha[1] and Khaleel Ur Rahman Khan[2]

[1]Department of Information Technology, MJCET, Hyderabad, India
[2]Department of Computer Science Engineering, ACE, Hyderabad, India

***ABSTRACT***

*Smart vehicles of today on road are equipped with advanced computational units, multiple communication technologies, intelligent sensing platforms, and human-computer interaction devices which utilize Vehicular Edge Networks to support services offered by the remote cloud. This being named as Opportunistic Vehicular Edge Computing recently, has the possibility to supplement the services provided by the Edge gadgets. Many Vehicular Edge Computing architectures have been proposed as of late which support task offloading. One among the premier difficulties in these networks is efficiently utilizing the resources available at the vehicular nodes. The present work uses APEATOVC, a conveyed and versatile protocol for economical, efficient and effective task offloading in these networks which address the adaptability of vehicular clouds. The results obtained by extensive simulations are presented to assess and contrast its performance with existing protocols.*

***KEYWORDS***

*Vehicular Cloud Computing, Mobile Edge Computing, Vehicular Ad-Hoc Networks, Computation Offloading*

## 1. INTRODUCTION

Mobile Computing platform used in smart vehicles known as Vehicular Cloud is visualized lately due to the rapid growth in processing, storage and communication capabilities of Vehicular Nodes. It is termed as Vehicular Edge Computing [1]. To make strides in the interest the on-road safety and other services various infrastructural services are required. The under-used assets in vehicles can be used to a full extent to satisfy this need. The cutting-edge challenges in these dynamic situations are productive and effective resource utilization and task offloading.

The idea behind executing tasks remotely is overwhelmingly adjusted from mobile cloud computing (MCC) because of the inherent restrictions of mobile devices in processing heavy tasks and to an extent maintain a strategic distance from battery deplete [2]. But vehicular networks do not confront this challenge. These rather can offer assistance to overcome the need of depending on the remote servers for processing and storage by using services provided by potential nodes and road-side hardware equipment. Further, these vehicles can assist in distributed and facilitated execution of tasks that are related to viably overseeing and managing various transport activities on roads. Planned evacuation during road traffic congestion and accidents can be addressed using vehicular networks [3].

DOI: 10.5121/ijcnc.2018.10603 

Numerous domains depend on distributed processing and access of data [2]. Hence, considering the monetary investment in data access, increased latency and bandwidth requirement, and also the high burden of energy with using 4G/LTE connectivity, the local resource-rich cloudlet would be a good alternative to the conventional remote servers if wireless access is properly managed.

Vehicles can moreover provide context-based data by partaking in performing distributed computations by using sensory data. [5]. Wireless sensor networks are largely deployed in locales to perform data aggregation and handling processing tasks. Future applications of vehicular networks incorporate autonomous transport systems, platooning, planned evacuations,

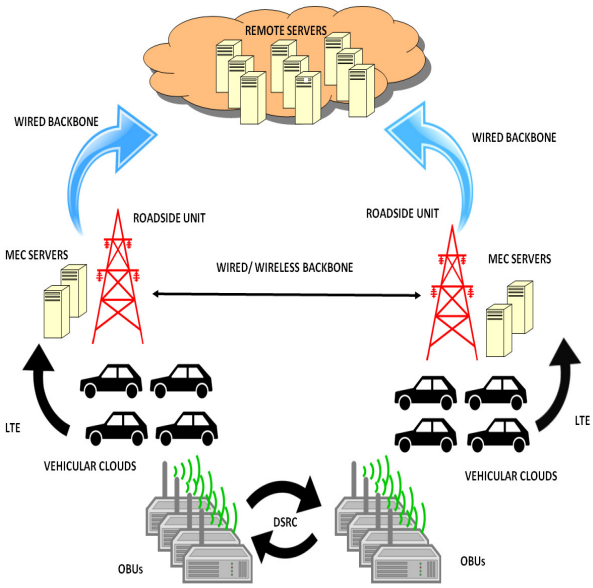


Figure. 1. Task Offloading in Vehicular Clouds using On Board Unit (OBU)

During accidents and virtual traffic light implementation at intersections [6]. Due to the various capabilities and features of these resource-rich mobile nodes, they are termed as Vehicular Cloud similar to a Cloud of resource-rich servers. One of the crucial challenges in vehicular cloud computing is the choice of surrogate vehicles that can satisfy client demands for data storage, computation offloading or providing processed sensory data.

Various recently proposed works address the concept of task offloading in vehicular networks [7], [16], Successful task completion [8], time constrained execution [9], [24], distributed task scheduling [10], [19], [27-30], fault tolerant task scheduling [11], [12], surrogate selection algorithms [13].

However, recently proposed schemes do not consider the joint effect of the dynamic wireless channel and dynamic resource availability in opportunistic vehicular ad-hoc networks. Moreover, the exact information about the wireless channel capacity and resource availability is difficult to find in VANETs. This is owing to the fact that the vehicular network is highly dynamic in nature with intermittent connectivity. To address the above-said issues in Energy- Aware Environments Adaptive Probabilistic Energy-Aware Task Offloading in Vehicular Clouds (APEATOVC) protocol is proposed. The novelty of this protocol is its scalability in case of dense vehicular traffic situations which are prominent in city scenarios along with energy awareness. Its practicability prevails in disaster management scenarios like planned evacuation which is part of the future Intelligent Transport Systems. The rest of the paper is organized as follows. The next

section reviews the related works. Section 3 presents the Problem formulation. The task offloading procedure is explained in section 4. In Section 5, we compare the proposed algorithm performance results with other approaches. Finally, the conclusions are briefed in Section 6.

## 2. RELATED WORK

The taking after segment provides an overview of the diverse angles of vehicular cloud computing that have been managed since its initiation. Broadly classifying vehicular cloud computing provides services in the form of stationary or dynamic resource centres [14].

In extension, the roadside hardware units can provide edge computing facilities to these resource centres. From the recent studies, we recognize the potential course of action these vehicular clouds can offer for the on-road client. We moreover discuss the challenges highlighted and their solutions proposed by different works. Authors in [15] to begin with presented opportunities and challenges in exploiting computing resources in vehicular ad-hoc networks through a hierarchical architecture for cloud-based vehicular networks that provides sharing of computational resources, storage resources among vehicles.

Authors in [17], examined the implementation of computation offloading in vehicular environments, proposing a framework to support it. The offloading algorithms were modelled utilizing MVC model but neither the vehicular mobility network nor the communication network environment was considered. Authors in [8], proposed a reliable task-scheduling model in Vehicular Cloud Computing environment to minimize execution time and satisfy job deadlines by defining a MILP optimization issue. But the proposed model is based on map-reduce which is implied for data-intensive applications. Deep reinforcement learning centered offloading procedure for the consumer to obtain the ideal offloading strategy in an ad-hoc itinerant cloud is proposed in [18]. The depositing problem is voiced as an MDP with the core objective of creating an optimal divesting action choice at each structure state as such that the convenience obtained by task completing is maximized by minimizing the energy usage, computing delay [18].

This is an inefficient and ineffective model for the case of vehicular networks since a large number of data transmission sessions are not achievable in practical scenarios due to irregular access mechanism which is used by Wireless LAN standards like 802.11p. Instead, the task offloading benefits only computation-intensive applications. Consequently, a fitting model needs to be based on virtual machine based architecture. In this case, only a brief set of variables are essential to invoke operations on the surrogate nodes.

The major commitments of present work are as follows. Firstly, we present task offloading from multiple client vehicles to surrogate nodes in vehicular edge networks. Furthermore, the proposed task offloading scheme offers a viable solution for offloading tasks of real-time applications by utilizing a viable vehicular resource discovery strategy to find out the up-to-date neighboring computational resources. Thirdly, it gives scalability by restricting the number of messages stormed amid request-response exchange between clients and surrogates in an Energy- Efficient way.

# 3. PROBLEM DEFINITION and FORMULATION

Table 1. Notations and their meanings used in problem formulation and related derivations

| Notation or Symbol | Meaning |
|---|---|
| *Task related parameters* | |
| $T_{id}$ | Task Identifier |
| $T_d$ | Time constraint of task or deadline |
| $C$ | Amount of computations/ instructions to be processed |
| *Surrogate related parameter* | |
| $\mu_s$ | Task processing capability of a node in million instructions per second (MIPS) |
| *T_Defferal* | Intentional Delay in Response time calculated by surrogate nodes |
| *Task completion time related parameters* | |
| $TCT_i$ | Task completion time for a given task i |
| $T_{tx}$ | Task transmission time |
| $T_{rx}$ | Task receiving time |
| $T_{exec}$ | Task execution time |
| $T_{queue}$ | Task queuing time |
| $D_i$ | Total size of the task parameters sent in bytes |
| $B_{c,s}$ | Shared Bandwidth between client and surrogate vehicle in Mbits/sec. |
| *Link lifetime related parameters* | |
| $T_{link}$ | Link lifetime between vehicles |
| $v_a$, $v_b$ | Instantaneous velocity of vehicle a, Instantaneous velocity of vehicle b |
| $d_{(a,b)}$ | Communication distance between vehicle a and vehicle b |
| $Mf$ | Mobility factor |
| $Mf_{pre}$ | Previous value of Mobility factor |
| $N$ | Node density |
| $w$ | Parameter used to regulate mobility factor value |
| $r$ | Relative velocity |
| *Parameters for estimating energy for task communication and computation* | |
| *TaskEnergy* | Total energy consumed by a task for execution at a surrogate |
| $E_{comp}$ | Total energy consumed for task execution |
| $E_{comm}$ | Total energy consumed for task workload transmission and reception |
| $E_{tx}$ | Total energy consumed for task workload transmission |
| $E_{rx}$ | Total energy consumed for task workload reception |
| *slackTime* | How early a task completes execution before deadline |
| $P_{i,s}$ | Probability of selecting a surrogate 's' with least energy requirement for executing task 'i' |

## 3.1 Problem Definition

### 3.1.1 Network Model

An opportunistic vehicular ad-hoc edge network consists of a number of vehicular nodes, modelled by an undirected communication graph ***G (V, E).*** To establish a direct communication between any two nodes, the distance between them has to be within their radio transmission range. The proposed system consists of client and surrogate nodes. Client nodes, which generate task offloading requests and Surrogate nodes, provide the computation service.

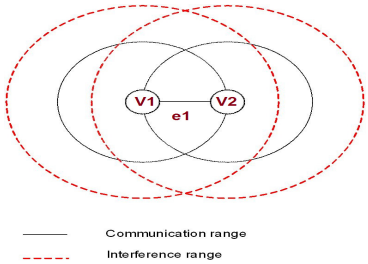


Figure 2. Communication graph for vehicular nodes v1 and v2 with edge e1

### 3.1.2 Task Model

A job is composed of tasks. A client is responsible for maintaining a queue for the generated tasks. The tasks are inserted as they are submitted in FCFS order. Each task in the task queue at the client node is denoted by triplet $\boldsymbol{<T_{id}, T_d, C>}$

where

- $T_{id}$, task Identifier;
- $T_d$ is time constraint of task or deadline;
- $C$, is amount of computations to be processed;

At each surrogate node, the queuing delay refers to the waiting time when the task is placed at the end of the queue until the moment that the task is processed. The task processing capability of a surrogate node, $\boldsymbol{\mu_s}$ is in units of MIPS (million instructions per second).

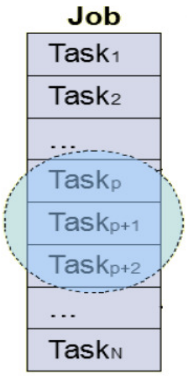


Figure 3. Computation Job split into multiple tasks.

### 3.1.3 Resource Modeling

The surrogates provide task computations for client vehicles. Moreover, the requested resource is made available to the client on request based on a two-phase reservation process. When the surrogate receives a request it holds the resource for a small period of time (50 ms). If the client does not send the task within the stipulated period, the resource is marked idle again.

## 3.2 Problem Formulation

Using the network and task models we present the calculations for the task completion time. The total task completion time for executing a task is composed of communication time, the computation time and the queuing delay of the task in the surrogate queue.

$$TCT_i = T_{tx} + T_{rx} + T_{exec} + T_{queue} \quad (1)$$

Where

- $T_{tx} = T_{rx} = D_i / B_{c,s}$ Where $D_i$ is the total size of task parameters and $B_{c,s}$ is the available shared bandwidth between the client and the surrogate;
- $T_{exec} = C_i / \mu_s$ Where $C_i$ is the computations requested for a task and $\mu_s$ is the computation capacity of the surrogate;
- $T_{queue}$ is the queuing delay of the task;

In vehicular networks the maximum time that can be allocated for completion of task is dependent on the link duration time between the client and surrogate. The link duration is derived from the relative velocity of the vehicles.

Thus we have

$$T_{link(a,b)} = (v_a - v_b) / d_{(a,b)} \quad (2)$$

Where

- $T_{link(a,b)}$ is the link duration time between vehicle *a* and vehicle *b*;
- $v_a$ and $v_b$ are instantaneous velocities of vehicle *a* and vehicle *b*;
- $d_{(a,b)}$ is the instantaneous communication distance between vehicle *a* and vehicle *b*;

Thus to successfully offload tasks to surrogates the following inequality needs to be realized.

$$2*(D_i / B_{c,s}) + (C_i / \mu_s) + T_{queue} \leq (v_a - v_b) / d_{(a,b)} \quad (3)$$

With the following constraints

- $TCT_i <= T_d$ (4)
- $\mu_s >> C_i$ per unit of time (5)
- $D_i <$ Wave Short Message Packet payload, i.e.,4096 bytes (6)

Constraint (4) is necessary for a task to be successfully offloaded to meet its deadline. Otherwise, it is considered failed. Constraint (5) ensures that a task is always executed at a surrogate with high computation capacity. Constraint (6) states that the task-related data being sent is within the bounds of the Wave Short Message Packet payload [4]. Constraint (7) and Constraint (8) ensures that the selected surrogates offer enhanced link lifetime and reliable task transmission. The equation has three components. (a) a task communication time component, (b) a task computation/ execution time component and (c) a link lifetime component

We infer the following from above component

- Though the task communication time is a smaller portion of the task completion time it influences the reliability of the scheduled task. Hence, to select highly reliable surrogates the nodes with the longest link lifetime should be selected. In the case of vehicular ad hoc network, the nodes with the least relative velocity and least distant to the client node can be the most reliable surrogates. This is owing to the fact that transmission errors are directly proportional to the distance to the surrogate nodes.

- The task completion time is dominated by the task execution time and the task queuing delay. It can be minimized by selecting surrogates with high computation capacity and with the least queuing delay.
- The link lifetime can be enhanced by selecting stable surrogates.

### 3.3 Evaluation of Link Stability

The selection of surrogate nodes with stable links to the client is important in highly dynamic networks like VANETs. We used the mobility factor parameter [20] obtained from the relative speed of the surrogate nodes. It provides enhanced link lifetime for task offloading. It is given by

$$Mf = \frac{(1+r)^{-w} + (Mf_{pre} * (N-1))}{N} \tag{7}$$

Where Mf is current mobility factor for a surrogate, Mfpre is previous mobility factor, r is the relative velocity, N is current node density, and w is a weight factor for emphasizing the small difference in speeds, initialized to 0.3. The value of Mf ranges from 0 to 1. With its maximum value, it specifies a stable link lifetime between the client and surrogate. Every client records and updates Mf for all its surrogate vehicles. Thus a client can select stable by evaluating its mobility factor.

### 3.4 Calculation of Mean Communication Time

To accurately measure the task transmission time between a client and a surrogate pair we introduce a parameter called the one-hop latency for a node pair. It is measured by the Medium Access Control (MAC) latency of the received message. To find the MAC latency of the received message we log the time at which the application layer inserts a message for transmission in the MAC queue. When this message is received by the receiver we calculate the one-hop latency as

$$\text{One-hop latency} = \text{Time-stamp application (sender)} - \text{current-time (receiver)} \tag{8}$$

Where Time-stamp application (sender) is the time at which the packet was inserted by the application layer into the MAC queue.

### 3.5 Estimation of communication and computation energy for surrogates

The total energy for task execution is composed of computation energy and the communication energy.

$$TaskEnergy_i = E_{comp} + E_{comm} \tag{9}$$

Where $E_{comp} = \mu^2 * C_i$ and $E_{comm} = (E_{tx} + E_{rx}) * D_i$

- $E_{comp}$ is task computation energy
- $E_{comm}$ is task communication energy
- $E_{tx}$ is WAVE packet Transmission energy
- $E_{rx}$ is WAVE packet Reception energy
- $\mu$ is processing capacity of the node
- $B_{c,s}$ is available shared bandwidth between the client and the surrogate
- $D_i$ is WSMP payload size.

Hence the total energy for offloading n tasks is given by

$$TaskEnergy = \sum_{i}^{n} TaskEnergy_i \tag{10}$$

Once a task is submitted, the client estimates the completion time on every potential surrogate node that satisfies

$$\text{TaskCompletionTime}_{i,s} < (\text{TaskDeadline}_{i,s} - \text{slackTime}) \tag{11}$$

The set of surrogates that satisfy the above equation are termed as potential surrogates and for these the energy requirements are considered. The probability that a potential surrogate with least task energy requirement is chosen is deduced as follows

$$P_{i,s} = (1/TaskEnergy_{i,s}) / \sum_{j \in S} (TaskEnergy_{i,j}) \tag{12}$$

In case scheduled tasks failed to complete within the deadline, select surrogate with high computation power or more distant to compromise energy savings for successful task completion. This is achieved by manipulating slack Time.

Using the expression to estimate the task completion time a novel task offloading protocol named APEATOVC (Adaptive Probabilistic Energy-Aware Task Offloading in Vehicular Clouds) is proposed as described in detail in the next section. Because the solution to MILP takes non-polynomial time to execute we propose a heuristic for solving it.

## 4. SCALABLE AND ENERGY EFFICIENT TASK OFFLOADING SCHEMES FOR VEHICULAR CLOUDS

The proposed algorithm is composed of three phases namely the resource discovery phase, fitness evaluation phase, and surrogate selection phase.

### 4.1 Vehicular Resource Discovery Protocol

The proposed resource discovery protocol is broadcast protocol to exchange both positional as well as task-related information from the local neighborhood of a node. All nodes participating in task offloading process compute a minimum threshold fitness score which is a function of its computation capacity (measured in million instructions per second i.e., represented by MIPS) and task queue length. The client nodes include it in the sent task offloading request and broadcast it.

### 4.2 Fitness Score Evaluation Process

Each target surrogate which receives a task offloading request compares the client fitness score to its own to determine its suitability for accepting the task for remote processing. Nodes that already have higher utilization of resources (as measured by the task queue length) avoid this calculation and simply drop the task offloading requests. This directly influences the overall power consumption of the network as a whole as it reduces the processor cycles at each node. A node which provides faster processing units and with a least bounded task queue computes a delay period for responding to client requests, called ***T_Defferal***. It is inversely proportional to its fitness score. This favors the less utilized surrogates over highly utilized ones. Hence the fittest surrogate gets a chance to reply to the client node first. This move reduces the response storm created by blind sending of replies by all surrogates. Moreover, taking advantage of the broadcast nature of the wireless medium the other surrogates when overhearing a response from a surrogate

compares the offered fitness score to its own fitness score. All surrogate nodes wait for a period equal to ***T_Defferal*** calculated individually before replying now overhearing surrogates may decide to cancel its delay timer of reporting the fitness score and avoid the reply to the client depending on the number of nodes in its neighbourhood. Because retransmissions are used to enhance the reliability of the transmitted messages it is important to adjust the retransmissions according to neighbourhood density.

Table 2. Task offloading using APEATOVC, Client part

| **Notations and their meanings used in APEATOVC :** |
|---|
| J: task queue length; CFS: client fitness score; SFS: surrogate fitness score; T_Defferal: delay period for sending reply to client; S: list of all surrogates; $S_l$: list of surrogates satisfying link lifetime criteria |
| **Input:** Task queue J at client node u |
| **Output:** best surrogate(s) selection |
| **Function:** Task Offloading in Vehicular Clouds |
| **Procedure body: For Client Nodes**<br>*If (node u is client) {*<br>*compute fitness score CFS*<br>*if (J = NULL) {*<br>*Broadcast task_offload_request;*<br>*If (received task_offload_response = true) {*<br>*for each surrogate s from S {*<br>*Calculate the estimated task completion time.*<br>*Calculate the Task Energy using computation energy and communication energy;*<br>*if (TaskCompletionTme i, s < TaskDeadline i, s - slackTime) then*<br>*Add the potential surrogate to selected surrogates list*<br>*for each surrogate from the potential surrogates list {*<br>*Pv= (1/TaskEnergy i, s)/ Total Energy);}*<br>*select surrogate node with highest probability;*<br>*Send task ti to s; and update the queue time for selected surrogate in the nodes knowledge-base;*<br>*if (received task ti's result) then {*<br>*if (TaskCompletionTimei < TaskDeadline,) {*<br>*initialize failed tasks to 0;*<br>*Initialize slackTime to INITIAL_SlackTime;}*<br>*else {*<br>*Increment the num. of failed tasks;*<br>*If (failed tasks >2)*<br>*decrement slackTime and recalculate Probability Pv;*<br>*}* |

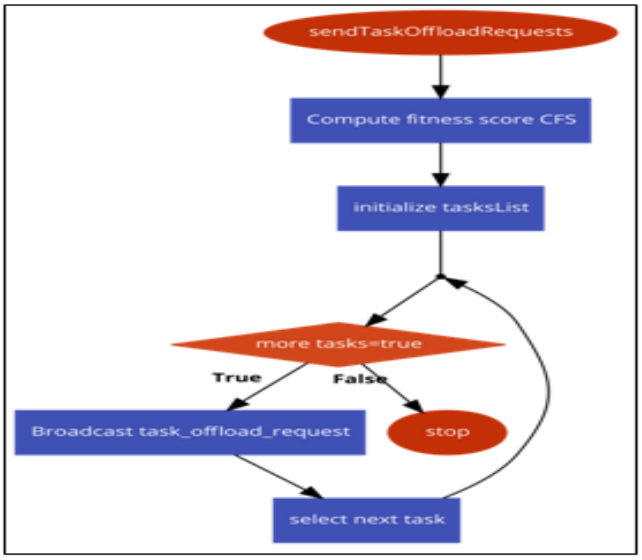


Figure 4. Procedure: send Task Offload Requests

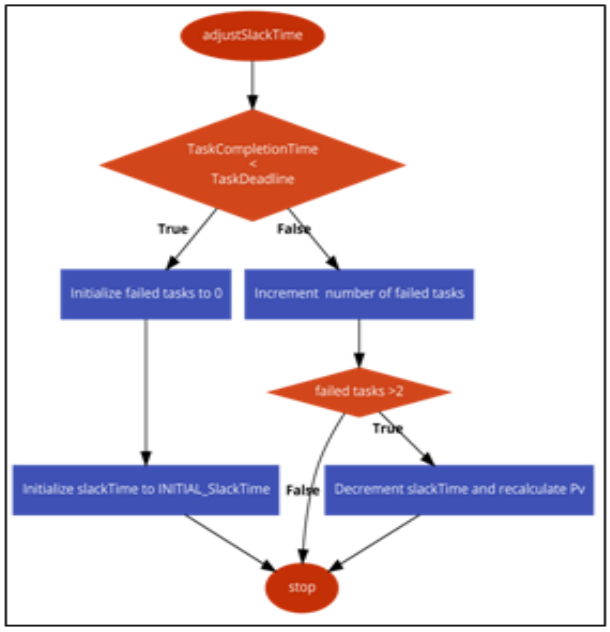


Figure 5. Procedure: adjust Slack Time

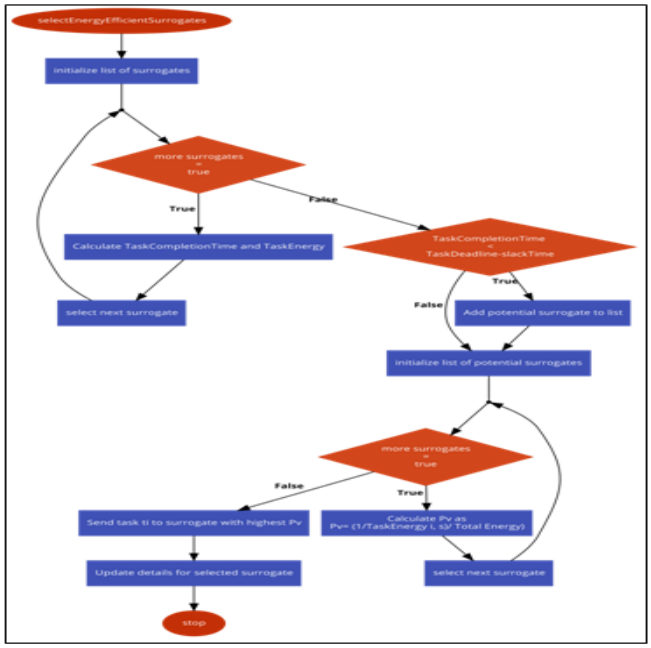


Figure 6. Procedure: select Energy Efficient Surrogates

Figure 4 and 5 elaborate on procedures adopted by client nodes sending task offload requests and adapting the slack Time to adjust for a given vehicular density. Figure 6 specifies how energy efficient surrogates are selected by evaluating the probability of consuming least energy of the total energy required among the selected surrogates.

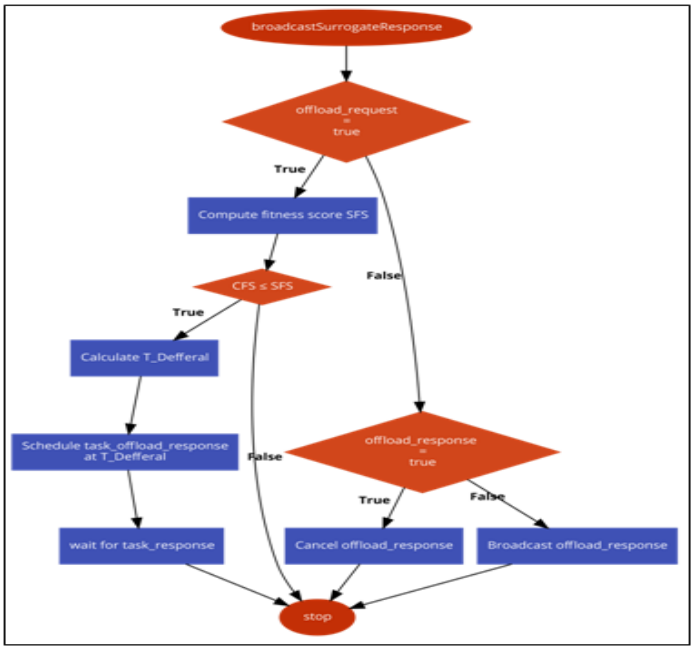


Figure 7. Procedure: broadcastSurrogateResponse

Figure 7 explains the steps taken for sending the task offload response by potential surrogates and how the network scalability is increased by reducing the broadcast storm through mitigating the unnecessary flooding of task response messages by surrogates.

Table 3. Task offloading using APEATOVC, Surrogate part

| **Notations and their meanings used in APEATOVC :** |
|---|
| J: task queue length; CFS: client fitness score; SFS: surrogate fitness score; T_Defferal: delay period for sending reply to client; S: list of all surrogates; $S_l$: list of surrogates satisfying link lifetime criteria |
| **Input:** Task queue J at client node u |
| **Output:** best surrogate(s) selection |
| **Function:** Task Offloading in Vehicular Clouds |
| **Procedure body: For Surrogate Nodes**<br>*If (node u is surrogate) {*<br>*If (received task_offload_request = true) {*<br>*compute fitness score SFS*<br>*//compare CFS and SFS*<br>*if (CFS > SFS)*<br>*drop request*<br>*else if (CFS ≤ SFS)*<br>*calculate T_Defferal*<br>*schedule task_offload_response at t = T_Defferal*<br>*}*<br>*If (received task_offload_response = true)*<br>*Cancel task_offload_response*<br>*Else Broadcast task_offload_response*<br>*}* |

### 4.3 Surrogate Selection Process

The client node waits for a predefined period of time to accept responses from surrogates. When it receives the fitness score from potential surrogates it calculates the total task completion time and slack-time for scheduling tasks on probable surrogates. It also evaluates the task energy consumption for scheduling on probable surrogates considering the communication and computation requirements. It selects surrogates with lease energy consumption for task scheduling. If the client is unable to receive response in stipulated time, it decreases the slackTime and recalculates the list of probable surrogates. Table 2. highlights the important phases of the protocol.

## 5. PERFORMANCE EVALUATION

The performance of APEATOVC algorithm compared with that of Greedy Surrogate Selection and PEATOVC schemes. The details of these are given below.

### 5.1 Energy Aware Task Offloading in Vehicular Clouds

Energy Aware Task Offloading in Vehicular Clouds is implemented in three variations

- **Greedy**: This considers the selection of surrogates based on successful task completion rate only. It tries to select surrogates with high processing power and least task queue length. This makes it inefficient in terms of energy savings.

- Probabilistic Energy-Aware Task Offloading in Vehicular Clouds (**PEATOVC**): This considers the task energy consumption in scheduling tasks to sacrifice successful task completion rate. This means less number of tasks are successfully scheduled to save energy.

- Adaptive Probabilistic Energy-Aware Task Offloading in Vehicular Clouds (**APEATOVC**): This algorithm tries to bring the best of both greedy and probabilistic surrogate selection schemes. It selects surrogates that consume the least energy to

schedule tasks to maintain a minimum successful task completion rate. It adjusts the energy savings to achieve better successful task completion rates by manipulating the slack Time, a parameter that determines how early a task can execute before its deadline.

The algorithms are implemented using vehicles in the network simulation framework [21] which couples OMNET++ [22] and SUMo [23] for realistic mobility modelling. To evaluate the performance, we consider that the client nodes generate 2 to 8 jobs per minute. These jobs are partitioned into 10 to 20 tasks each [8]. Thus we vary the number of tasks from 20 to 160 tasks for different experiments. Each task has a computation requirement of 1000 MIPS to 2000MIPS.

Table 4. Level of Service on Highways

| Level of Service | Quality | Speed (kmph) | Road Volume to Capacity ratio | Level of Comfort |
|---|---|---|---|---|
| A | Free-flow | 80 | 0.6 | High |
| B | Reasonable | 70 | 0.7 | Reasonable |
| C | Near | 60 | 0.8 | Low |
| D | Medium | 50 | 0.9 | Absent |

## 5.2 Simulation Setup

Table 5. Simulation Parameters

| Parameter | Value |
|---|---|
| **Vehicular Network Parameters** | |
| Warm-up distance | 1000meters |
| mac1609_4.txPower | 10mW |
| Transmission Range | 100 meters approx. |
| mac1609_4.bitrate | 6Mbps |
| Vehicular Density | 20, 25, 30, 35, 40, 45, 50 veh/km |
| Car_Following_Model | Krauss |
| Vehicle_InterArrival | Gaussian_Exponential_Mixture |
| Offloading schemes | 1. Greedy,<br>2. Probabilistic Energy Aware Task Offloading in Vehicular Clouds and<br>3. Adaptive Probabilistic Energy Aware Task Offloading in Vehicular Clouds |
| **Task Parameters** | |
| Jobs/ min | 2 to 8 |
| No. of tasks per job | 10 to 20 |
| Task Computation amount | Uniform (1000, 2000) |

We consider one-dimensional vehicular network formed on a 2-Lane highway which follows the Level of Service concept of transport management [25] as shown in Table 3 is considered. To evaluate the effect of the vehicular movement on the offloading decision we make use of the Gaussian exponential mixture of mobility model proposed in [26] which is necessary to realistic simulations.

We evaluate the proposed scheme to access the effect of (a) varying the network size and vehicular mobility, (b) varying the number of jobs offloaded. We evaluate the Average Job Execution Time, Successful Job Completion Rate for offloaded tasks in terms of control messages sent.

- **Successful Job Completion Rate:** It is defined as the ratio of successfully executed jobs to the total number of jobs offloaded. It is evaluated for varying Job arrival rates.
- **Average Energy Per Successful Job Execution:** It is defined as average energy consumed by Job composed of tasks. It is evaluated for varying vehicular densities and Job arrival rates.
- **Average Job Execution Time:** It is defined as the time taken by an Offloaded Job to complete when individual tasks are scheduled at surrogates. It is evaluated for varying Job arrival rates.

Initially, we compare the performance of APEATOVC with Greedy and PEATOVC for Average Task Completion Time-varying vehicular density and job rate. Simulation parameters for which are given below. The value for vehicular density is fixed at 40 vehicles per km so as to learn the optimal behavior of the proposed protocol. If we decrease the vehicular density below 40 vehicles per km we observe a lesser surrogate number than required for the application under study. On the other end if we increase the vehicular density above 40 vehicles per km we observe a significant increase in the broadcast storm phenomenon.

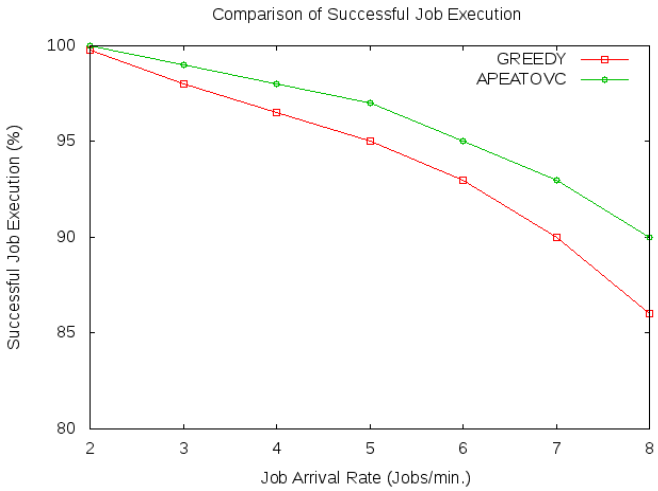


Figure 8. Comparison of percentage of Successful Job Execution for varying Job Arrival Rate

Figure 8 shows the performance of the proposed algorithms for varying job arrival rate. The vehicular density is fixed at 40 veh/ km. We can observe that when the number of tasks offloaded is small the percentage of Successful Job Execution achieved is full. For vehicles number increases, the Job Completion rate decreases. This is due to the fact that more network traffic is generated in case of higher job arrival rates.

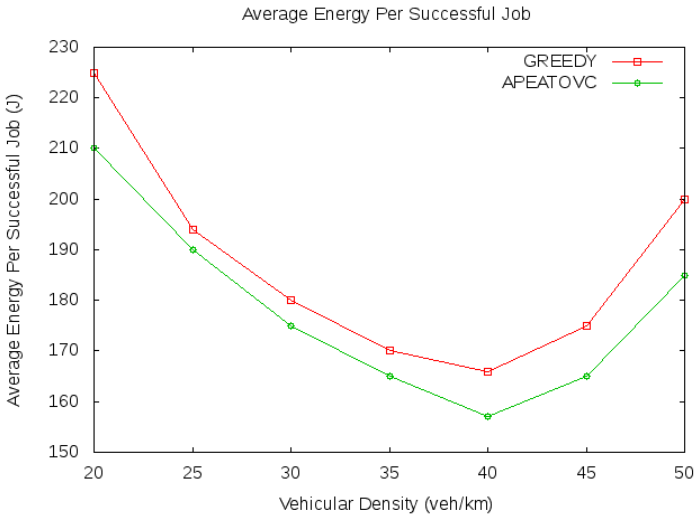


Figure 9. Comparison of Average Energy per Successful Job for varying vehicular densities

Figure 9 shows the performance of the proposed algorithms in terms of Average Energy per Successful Job for varying job arrival rate. The Job arrival rate is fixed at 6 jobs per min. We can observe that as the vehicular density increases the Average Energy per Successful Job decreases to a minimum at 40 vehicles per km. It then starts increasing from there-on. It is due to the increase in the number of retransmissions for task offloading required to find useful surrogates at sparse vehicular traffic. The retransmissions decrease as the number of useful surrogate nodes increase. But too many surrogates competing for channel again increases task retransmissions. Hence Average Energy per successful task increases.

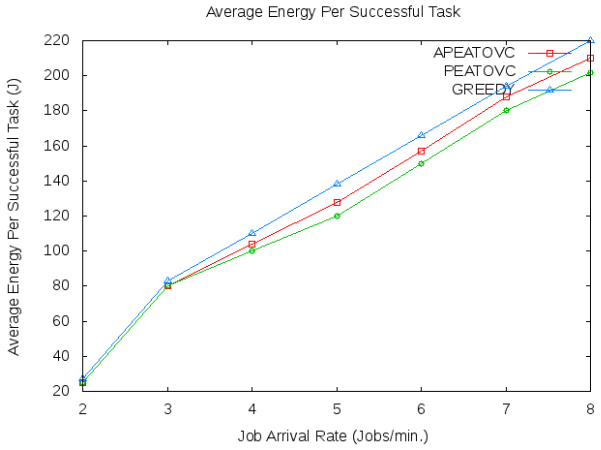


Figure 10. Comparison of Average Energy per Successful Job for varying Job Arrival Rate

Figure 10 shows the performance of the proposed algorithms for varying job arrival rate. The vehicular density is fixed at 40 veh/ km. As the number of tasks offloaded increases the Average Job Execution Energy also increases linearly. The Greedy surrogate selection scheme consumes more energy whereas Probabilistic scheme consumes the least energy. The adaptive probabilistic scheme plays in the middle by optimizing energy consumption.

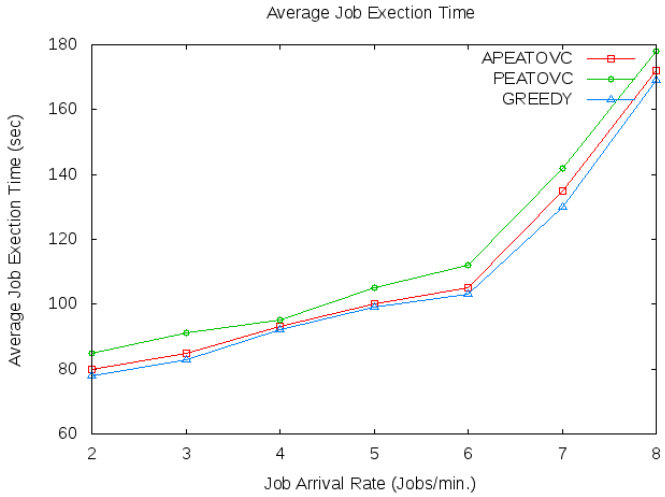


Figure.11. Comparison of Average Job Execution Time for varying Job Arrival Rate

Figure 11 shows the performance of the proposed algorithms for varying job arrival rate. The vehicular density is fixed at 40 veh/ km. As the number of tasks offloaded increases the Average Job Execution Time also increases linearly. The Greedy surrogate selection scheme provides faster Job execution whereas Probabilistic scheme provides slower job executions. The adaptive probabilistic scheme plays in the middle by optimizing successful job execution rates.

## 6. CONCLUSION

An efficient and effective task offloading conspire was presented for vehicular networks to back up the Edge Computing. The proposed scheme considers task offloading from different client nodes to adjacent surrogate nodes, fulfilling the different tasks limitations while considering the wireless channel dynamics. One noteworthy point is to note that it is just not enough that a particular surrogate is chosen purely based on fulfilling the tasks due to the limitation. The interface lifetime between the client and the surrogate ought to be of enough length for the client to send the task workload to the surrogate. The impact of the delay that happens due to contention at the MAC layer ought to be taken into consideration when offloading a task. The proposed algorithm outperforms existing protocols in terms of Scalability, Average Job Execution time and percentage of Successfully Completed jobs. The practical applicability of the proposed scheme is hindered by the ways in which the power measurements are made for a set of applications that run inside an intelligent vehicle. The model used in the proposed work only estimates the power consumption at the radio layer but not the application layer. The results may significantly vary when we consider power usage by background applications.

## REFERENCES

[1] Hu, Yun Chao, et al. "Mobile edge computing—A key technology towards 5G." ETSI white paper 11.11 (2015): 1-16.

[2] Fernando, Niroshinie, Seng W. Loke, and Wenny Rahayu. "Mobile cloud computing: A survey." Future generation computer systems 29.1 (2013): 84-106.

[3] Sardellitti, Stefania, Gesualdo Scutari, and Sergio Barbarossa. "Joint optimization of radio and computational resources for multicell mobile-edge computing." IEEE Transactions on Signal and Information Processing over Networks 1.2 (2015): 89-103.

[4] Uzcátegui, Roberto A. "Universidad Nacional Experimental Politécnica “Antonio José de Sucre” Guillermo Acosta-Marum." Georgia Institute of Technology “WAVE: A Tutorial” TOPICSIN Automotive Networking (2009).

[5] Abdelhamid, Sherin, Hossam S. Hassanein, and Glen Takahara. "Vehicle as a mobile sensor." Procedia Computer Science 34 (2014): 286-295.

[6] Chen, Lei, and Cristofer Englund. "Cooperative intersection management: a survey." IEEE Transactions on Intelligent Transportation Systems 17.2 (2016): 570-586.

[7] Shojafar, Mohammad, Nicola Cordeschi, and Enzo Baccarelli. "Energy-efficient adaptive resource management for real-time vehicular cloud services." IEEE Transactions on Cloud computing (2016).

[8] Adhikary, Tamal, et al. "Quality of service aware reliable task scheduling in vehicular cloud computing." Mobile Networks and Applications 21.3 (2016): 482-493.

[9] Turcanu, Ion, et al. "DISCOVER: a unified protocol for data dissemination and collection in VANETs." Proceedings of the 12th ACM Symposium on Performance Evaluation of Wireless Ad Hoc, Sensor, & Ubiquitous Networks. ACM, 2015.

[10] Liu, Kai, et al. "Cooperative data scheduling in hybrid vehicular ad hoc networks: VANET as a software defined network." IEEE/ACM Transactions on Networking (TON) 24.3 (2016): 1759-1773.

[11] Ye, Tianpeng, et al. "A Safety Resource Allocation Mechanism against Connection Fault for Vehicular Cloud Computing." Mobile Information Systems 2016 (2016).

[12] Ghazizadeh, Puya, et al. "Towards fault-tolerant job assignment in vehicular cloud." Services Computing (SCC), 2015 IEEE International Conference on. IEEE, 2015.

[13] Orsini, Gabriel, Dirk Bade, and Winfried Lamersdorf. "Computing at the mobile edge: Designing elastic android applications for computation offloading." IFIP Wireless and Mobile Networking Conference (WMNC), 2015 8th. IEEE, 2015.

[14] Whaiduzzaman, Md, et al. "A survey on vehicular cloud computing." Journal of Network and Computer Applications 40 (2014): 325-344.

[15] Zhao, Dong, et al. "Opportunistic coverage for urban vehicular sensing." Computer Communications 60 (2015): 71-85.

[16] Chaâri, Rihab, et al. "Cyber-physical systems clouds: A survey." Computer Networks 108 (2016): 260-278.

[17] Li, Bo, et al. "Computation offloading management for vehicular ad hoc cloud." International Conference on Algorithms and Architectures for Parallel Processing. Springer, Cham, 2014.

[18] Van Le, Duc, and Chen-Khong Tham. "A deep reinforcement learning based offloading scheme in ad-hoc mobile clouds." IEEE INFOCOM 2018-IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS). IEEE, 2018.

[19] Alfaqawi, Mohammed IM, et al. "Adaptive load balancing algorithm for wireless distributed computing networks." Intelligent Systems Engineering (ICISE), 2016 International Conference on. IEEE, 2016.

[20] Lee, Won–Il, et al. "Relative velocity based vehicle–to–vehicle routing protocol over ad–hoc networks." International Journal of Ad Hoc and Ubiquitous Computing 12.1 (2013): 14-22.

[21] Sommer, Christoph, Reinhard German, and Falko Dressler. "Bidirectionally coupled network and road traffic simulation for improved IVC analysis." IEEE Transactions on Mobile Computing 10.1 (2011): 3-15.

[22] Varga, András, and Rudolf Hornig. "An overview of the OMNeT++ simulation environment." Proceedings of the 1st international conference on Simulation tools and techniques for communications, networks and systems & workshops. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2008.

[23] Krajzewicz, Daniel, et al. "SUMO (Simulation of Urban MObility)-an open-source traffic simulation." Proceedings of the 4th middle East Symposium on Simulation and Modelling (MESM20002). 2002.

[24] Klingler, Florian, Falko Dressler, and Christoph Sommer. "IEEE 802.11 p unicast considered harmful." Vehicular Networking Conference (VNC), 2015 IEEE. IEEE, 2015.

[25] Barceló, Jaume. "Models, traffic models, simulation, and traffic simulation." Fundamentals of traffic simulation. Springer, New York, NY, 2010. 1-62.

[26] Gramaglia, Marco, et al. "New insights from the analysis of free flow vehicular traffic in highways." World of Wireless, Mobile and Multimedia Networks (WoWMoM), 2011 IEEE International Symposium on a. IEEE, 2011.

[27] Moon, YoungJu, et al. "A slave ants based ant colony optimization algorithm for task scheduling in cloud computing environments." Human-centric Computing and Information Sciences 7.1 (2017): 28.

[28] Zhou, Bowen, et al. "An Online Algorithm for Task Offloading in Heterogeneous Mobile Clouds." ACM Transactions on Internet Technology (TOIT) 18.2 (2018): 23.

[29] Volkov, Bc Nikolay. "Resource allocation for vehicular cloud computing." management 27.5 (2018): 48-55

[30] Jiang, Zhiyuan, et al. "Task Replication for Deadline-Constrained Vehicular Cloud Computing: Optimal Policy, Performance Analysis, and Implications on Road Traffic." IEEE Internet of Things Journal 5.1 (2018): 93-107.