International Journal of Recent Technology and Engineering (IJRTE)
ISSN: 2277-3878 (Online), Volume-8 Issue-4, November 2019

Efficient Task and Data Scheduling Policy for Vehicular Fog Computing Based on Link Weight

Santosh Kumar Sahoo, Asif Uddin Khan, Ajit Kumar Nayak

Abstract*: Vehicular network has several applications in the smart city and IoT. Recently with the advancement in the computing technology such as fog computing and its application in the vehicular network and its services, a new paradigm known as vehicular fog computing has evolved as a hot topic of investigation in the research community because of the next generation computing and communication requirements. Vehicular fog computing can be used to solve the issues of next generation computing and communication scenario. There are several issues in vehicular fog computing. Efficient task computing and data dissemination is an important issue. Several approaches are proposed by different authors to solve the issues, but none of them has addressed the service completion and failure rate which is very important in the vehicular scenario as the vehicles move very fast and its contact time with the RSU controller is limited. The task has to be completed by the vehicular server within that time period, otherwise computation will fail. Once the computation and communication fails, the RSU controller will reinitiate to form the vehicular fog resulting high overhead. In this paper we address this issue and proposed an efficient scheduling algorithm based on multiple parameters namely queue length, response time and link weight. We simulated the algorithm using java and compared with the existing algorithm showing better performance.

Keywords: VANET, fog computing, queuing model, scheduling.

I. INTRODUCTION

Vehicular fog computing has several data and latency intensive applications, ranging from pattern recognition to augmented reality [1]. There are applications where the operations are very critical to latency, for example preventing collisions and accidents can't afford the delay of communication between the vehicular user and central cloud sensors. Further many applications and data have local and time constraint relevance in the vehicular network and it must be processed within a specified time period and specific locality [2]. To solve the above problems a new computing paradigm known as vehicular for computing can be used where the computing resources, applications and other cloud services can be provided by the edge devices such as vehicles. The fog computing paradigm has three layers namely central cloud layer (CCL), Road-side Cloud Layer (RCL) and Vehicular Cloud Layer (VCL). Here RSU controller in the RCL acts as the broker.

The time sensitive and real time application services are provided in the vehicular fog layer. If it is not available in this layer then request is forwarded to the RCL. If it is also not available in this layer then it is finally forwarded to the central cloud layer to be served as shown in the Fig. 1 [3].

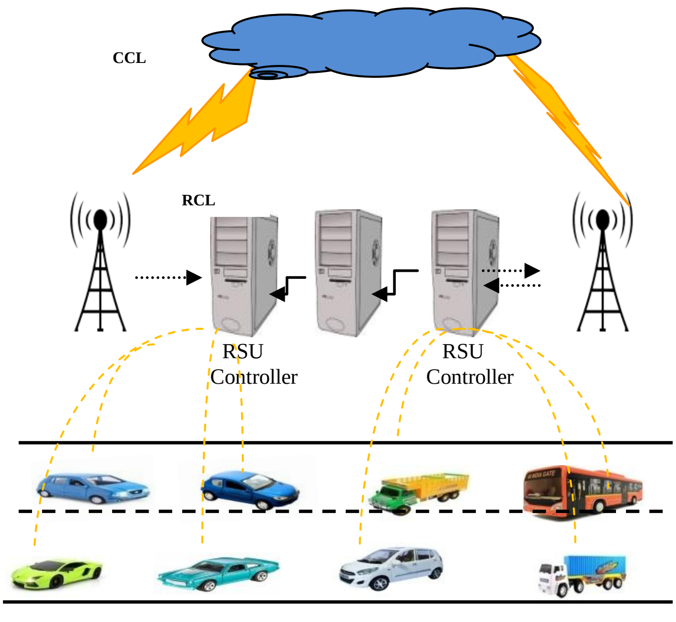

Fig. 1. Vehicular Fog Architecture

In this work we proposed an efficient task scheduling algorithm based on multiple parameters namely queue length, response time and link weight [4]. Link weight is the probability that a link between the cloud service provider and user or the broker will be continuously available for a time period Δt.

The rest of the paper is organized as follows:

Section-II describes the problem statement. In section-III, we discuss the related work. In section-IV, we present the solution approach. Section-V presents performance analysis and finally section-VI, we conclude the paper.

II. PROBLEM STATEMENT

Large number of data computation and data dissemination tasks are provided by the vehicular servers in vehicular fog layer under the supervision of controller in the RSU controller layer.

Manuscript published on November 30, 2019.
* Correspondence Author
Santosh Kumar Sahoo*, Department of Computer Science & Engineering, Trident Academy of Technology, Bhubaneswar, Odisha, India. Email: esantoshsahoo@gmail.com
Asif Uddin Khan, Department of Computer Science & Engineering, Silicon Institute of Technology, Bhubaneswar, Odisha, India. Email: asif.khan@silicon.ac.in
Ajit Kumar Nayak, Department of IT, Siksha O Anusandhan Deemed to be University, Odisha, India. Email: ajitnayak2000@gmail.com

The tasks are scheduled by the RSU controller and allocated to the vehicular servers for processing. Vehicular servers are the mobile vehicles, their connection time with the controller is limited.

Each task has a certain processing time requirement. If proper vehicular server with enough processing time is not selected to process the task then it can't be completed within the specified time period. If processing fails, the RSU controller will again reinitiate vehicular node selection and cloud/fog formation which is an overhead.

Several scheduling algorithms for vehicular fog computing have been proposed by different authors in the literature are based on queue length and response time, where the task is scheduled and allocated to the vehicular server with less queue length and less response time respectively. In this paper, we proposed an efficient scheduling algorithm based on multiple parameters namely queue length, response time and link weight to select the stable vehicular server for processing tasks.

Example – Scenario-1:

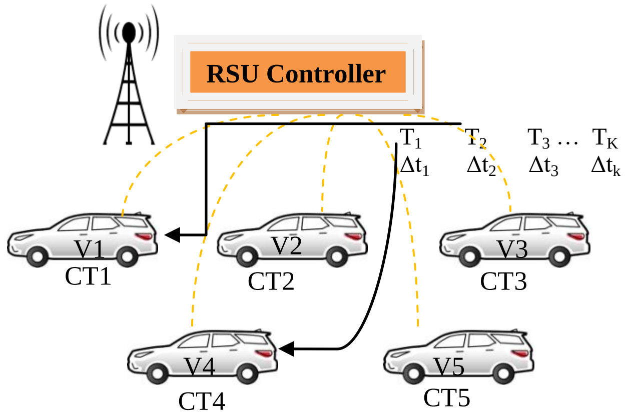

Fig. 2. Example Scenario

Δt_1 : required processing time of task-1(T_1)

Δt_2 : required processing time of task-2 (T_2)

Δt_3 : required processing time of task-3 (T_3)

Δt_4 : required processing time of task-4 (T_4)

.Δt_k: required processing time of task-k (T_k)

CT_i : connection time of vehicular server i with the RSU controller.

In Fig. 2, $\Delta t_1 > CT_2$, $\Delta t2 > CT_3$, $\Delta t1 <= CT4$, and $\Delta t_2 <= CT_1$ If T_1 and T_2 are allocated to V_2 and V_3 then it will not complete execution and the task completion failure rate will increase. Where as if the tasks T_1 and T_2 are allocated to the vehicular servers V_4 and V_1 where required service/processing time is less than the available time then the task can easily be completed resulting high task completion rate.

The vehicular fog scheduling problem can be formulated as shown in equation-1, equation-2, equation-3 and equation-4.

Problem Formulation

Let TC_i be the number of task completed at the server i successfully within the specified time period. T_i is the time required to complete the $task_i$ at the $server_i$. The problem is formulated as P1:

P1: Maximize $\sum_{i=1}^{n} TC_i \ / \ n$ (1)

Minimize $\sum_{i=1}^{n} T_i$ (2)

Subject to

$$\sum_{i=1}^{n} \frac{RT_i}{n} < rt \quad (3)$$

$$\frac{\sum_{i=1}^{n} WT_i}{n} < wt \quad (4)$$

In this problem our objective is to maximize rate of successfully task completion and minimize the total time required to complete the entire task by reducing the waiting time as shown in (1) and (2). The constraint of the problem is to keep average response time less than a threshold value *rt* and average waiting time less than the threshold value *wt* as shown in (3) and (4) respectively.

III. RELATED WORK

Day by day vehicles are becoming very smart using on board computing, storage and communication capabilities. A lot of research works are carried out on development of vehicles with ICT and networking connections to provide a cloud computing services on vehicles which is called Vehicular Cloud Computing [2]. The vehicular cloud computing similar to vehicular ad-hoc network (VANET) that starts with physical level communications to challenges in the network in vehicle-to-vehicle (V2V) and vehicle-to-infrastructure (V2I). Fog computing applications integrated on conventional vehicle ad-hoc network (VANET) obtain Vehicular Fog Computing (VFC) or Internet of Vehicles. Now-a-days vehicles are becoming mobile and intelligent added with multiple sensors having computational and communication capabilities can collect useful traffic information. At the edge of the vehicular network fog computing services are deployed for better performance in real data collection, storage and processing them. Road safety improvement and smart traffic control are some example of Vehicular Fog Computing [5]. Using a novel compositional technique Performance Evaluation Process Algebra (PEPA) models are formulated on stochastic process algebra [3]. PEPA is used to design complex models where the detail inputs from system components are known. Using PEPA dynamic scheduling for the vehicular cloud can be formed with queue length. Link weight is not used in PEPA in the earlier models.

IV. PROPOSED SOLUTION

Our solution is based on M/M/c [6] queuing model in which the arrival of jobs follows a poison distribution process and the service time is distributed exponentially having multiple servers as shown in Fig. 3.

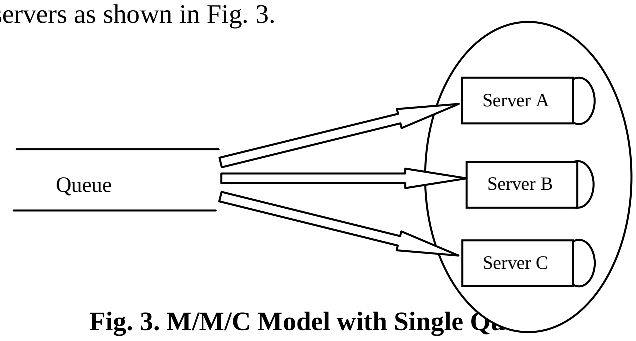

Fig. 3. M/M/C Model with Single Qu

The assumptions of the model are as below.

1. The number of jobs arriving follows a poison distribution with parameter ⋋.

2. The rate of service is exponentially distributed with parameter μ.
3. The number of server is three.
4. Both the population and the size of the queue although not explicitly specified can be infinity.

This M/M/c model can be described as a continuous time Markov chain with transition rate matrix as shown in Fig. 4 [7]. It is assumed that the jobs finding more than one vacant server choose their servers randomly.

$$Q = \begin{pmatrix} -\lambda & \lambda & & & & & & & \\ \mu & -(\mu+\lambda) & \lambda & & & & & & \\ & 2\mu & -(2\mu+\lambda) & \lambda & & & & & \\ & & 3\mu & -(3\mu+\lambda) & \lambda & & & & \\ & & & & \ddots & & & & \\ & & & & c\mu & -(c\mu+\lambda) & \lambda & & \\ & & & & & c\mu & -(c\mu+\lambda) & \lambda & \\ & & & & & & c\mu & -(c\mu+\lambda) & \lambda \\ & & & & & & & & \ddots \end{pmatrix}$$

Fig. 4. Continuous Time Markov Chain with Transition Rate Matrix

The state space diagram for the chain of M/M/c model is shown in Fig. 5 [6].

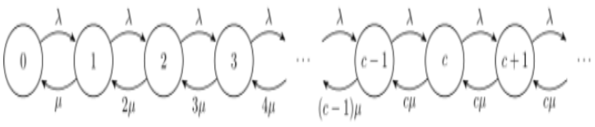

Fig. 5. The State Space Diagram

In the proposed solution the tasks are coming and stored in a queue of M/M/c queuing model with task arrival rate λ and service rate of the server μ. we have scheduled the incoming tasks based on the proposed algorithm-1: Dynamic Scheduling with Multiple Metrics and then the tasks are dispatched to the scheduled vehicular server. We have used multiple metrics namely queue length (Q_l), Response Time (R_t) and link weight (W_t).The response time and queue length probability is calculated using [3]. The link weight is calculated based on the speed, position and direction of vehicular server with respect to the local RSU controller where the scheduling is done, which we have already published in our previous paper [4]. The proposed algorithm is explained using example-1.

Example-1

WTS_1 : Link Weight of S_1
WTS_2 : Link Weight of S_2
WTS_3 : Link Weight of S_3
QLS_1 : Length of Queue at S_1
QLS_2 : Length of Queue at S_2
QLS_3 : Length of Queue at S_3
RTS_1 : Response Time at S_1
RTS_2 : Response Time at S_2
RTS_3 : Response Time at S_3
Example
3 servers S_1, S_2, and S_3
μ_1 : 4 tasks processed per second
μ_2 : 5 tasks processed per second
μ_3 : 6 tasks processed per second
λ = 4 task arrived per second
Simulate for 2 seconds
Total number of task = $T_1, T_2, \ldots T_8$

The tasks and servers used in this example are shown in Table-I and Table-II respectively.

Table-I : Task Table

Task Table	
TaskID	***Duration***
T_1	6
T_2	5
T_3	6
T_4	7
T_5	4
T_6	5
T_7	6
T_8	7

Table-II : Server Table

Server Table				
Server	P(QL)	P(RT)	P(CT)	***CT***
S_1	0.8	0.8	0.7	7
S_2	0.7	0.7	0.8	8
S_3	0.9	0.9	0.9	9

In Table-II, P(QL) is the probability of queue length, P(RT) is the probability of response time, P(CT) is the probability of contact time which is the link weight and CT is the contact time of the servers with the RSU controller.

$P(S_3)$ for T_1 = 0.729
$P(S_2)$ for T_1 = 0.392
$P(S_1)$ for T_1 = 0.448

Here, $P(S_3) > P(S_1) > P(S_2)$ and CT of S_3 satisfies the required time constraint of T_1.

So, T_1 is allocated to server S_3 and it will successfully be processed.

Algorithm-1: Dynamic Scheduling with Multiple Metrics

Input: J: set of tasks, S : Set of Vehicular Servers, K : No of Vehicles, λ : arrival rate and μ : service rate

Output : $Alloc_{j,s}$: Schedule a task j ∈ J to a server s ∈ S

1. Initialize : J, S, λ, μ
2. Set a M/M/c/K terminal model with K vehicles
3. For all server s ∈ S
4. Calculate $E_{r,t}$: mean response time at all servers using[3]
5. For all server s ∈ S
6. Calculate Queue length of the server using[3]
7. for all server s ∈ S
8. Calculate link weight WT of each server using[4]
9. For each task j ∈ J do
10. Calculate Transient waiting time using $Tw = \frac{Q_s}{\mu}$
11. Find transient service time $Ts = \frac{1}{\mu}$
12. Find transient response time Tr = Tw + Ts
13. For all j ∈ J
14. Find probability of response time $Pr,t = \frac{E_{r,s}(K)}{T_r}$
15. For all j ∈ J and all s ∈ S
16. Find probability of job allocation $P_{j,s}$ = P(QL). P(RT).(PCT)
17. Allocate job to server with high probability $P_{j,s}$

V. PERFORMANCE ANALYSIS

We have simulated the proposed algorithm using java and compared the success rate and failure rate of job completion by the vehicular servers with other approaches namely scheduling based on queue length and scheduling based on response time.

The simulation parameters are summarized in Table-III. We simulated the algorithms for 5 minutes taking 3 servers, one terminal, and task arrival rate from 1 to 5 and service rate from 1 to 5.

Table-III: Simulation parameter

Parameter	Value
S(Servers)	3
K	1
λ	1, 2, 3, 4, 5
μ	1, 2, 3, 4, 5
Simulation Time	5 minute

Success Rate: Task completion rate in percentage
Failure rate: Task Failure rate in percentage

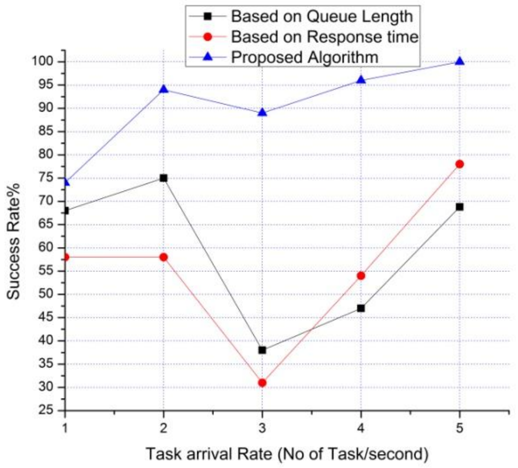

Fig. 6. Success rate vs Task Arrival Rate

In Fig. 6 we have plotted Success rate vs varying task arrival rate. From this figure it is observed that success rate of the proposed scheduling algorithm is more than the other approaches. Success rate of tasks using link weight scheduling starts from 74% as compared to 58% and 68% of queue length and response time scheduling respectively. During various tasks scheduled with different task execution time the success rate of link weight scheduling remains higher and better than that of other types of scheduling. Success rates at various interval of task arrival rate are 94, 89, 96 and 100 percentages.

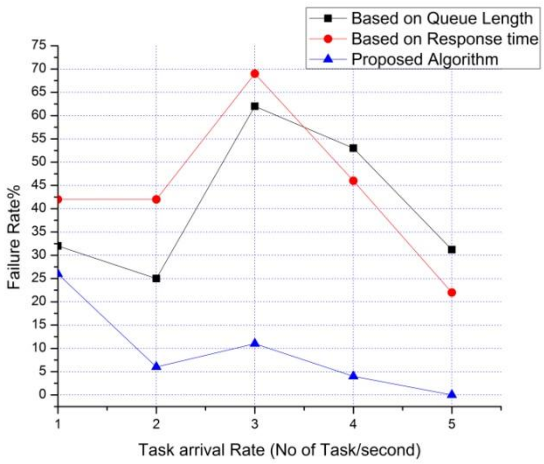

Fig. 7. Success rate vs Task Arrival Rate

In Fig. 7 we plotted failure rate vs varying task arrival rate. From the figure it is observed that failure rate of the proposed scheduling algorithm is less than the other approaches. Failure rate of tasks using link weight scheduling starts from 26% as compared to 32% and 42% of queue length and response time scheduling respectively. During various tasks got scheduled with different task execution time the failure rate of link weight scheduling continuously decreasing in a higher rate than that of other types of scheduling. Failure rates at various interval of task arrival rate are 6, 11, 4 and 0 percentages.

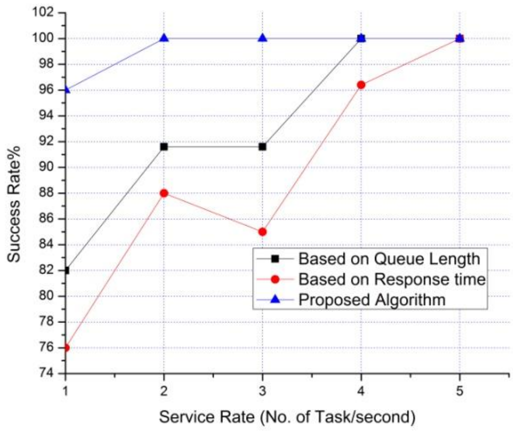

Fig. 8. Success rate vs Service Rate

In Fig. 8 we have plotted Success rate vs varying service rate. From this figure it is observed that success rate of the proposed scheduling algorithm is more than the other approaches. Success rate of tasks executed using link weight scheduling starts from 96% as compared to 82% and 76% of queue length and response time scheduling respectively. During various tasks scheduled with different task execution time the success rate of link weight scheduling remains higher and continuously increasing than that of other types of scheduling. Success rates at various intervals of task arrival rates are 100, 100, 100 and 100 percentages.

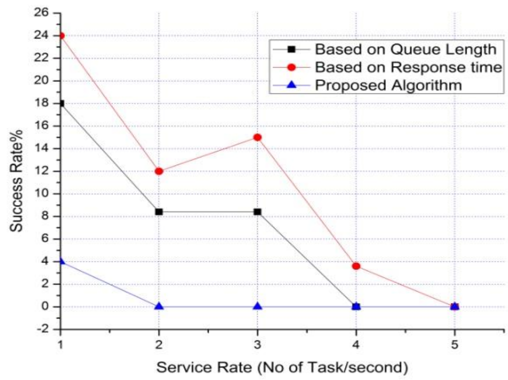

Fig. 9. Failure rate vs Service Rate

In Fig. 9 we plotted failure rate vs varying service rate. From the figure it is observed that failure rate of the proposed scheduling algorithm is less than the other approaches. Failure rate of tasks using link weight scheduling starts from 4% as compared to 18% and 24% of queue length and response time scheduling respectively.

During various tasks got scheduled with different task execution time the failure rate of link weight scheduling continuously decreasing in a higher rate than that of other types of scheduling. Failure rates at various interval of service rate are 0, 0, 0 and 0 percentages.

VI. CONCLUSION

In this paper we investigated the important issue of vehicular fog computing that is task and data scheduling. We discussed how the overhead is increased and performance is reduced during the failures in task completion because of the contact time issue. Then we proposed an efficient task scheduling algorithm based on multiple metrics and we compared and analyzed its performance with the existing policies showing better performance and solve the issue.

REFERENCES

1. Whaiduzzaman, Md, et al. "A survey on vehicular cloud computing." Journal of Network and Computer applications 40 (2014): 325-344.
2. Lee, Euisin, et al. "Vehicular cloud networking: architecture and design principles." IEEE Communications Magazine 52.2 (2014): 148-155.
3. Chen, Xiao, and Liangmin Wang. "Exploring fog computing-based adaptive vehicular data scheduling policies through a compositional formal method—PEPA." IEEE Communications Letters 21.4 (2017): 745-748.
4. Khan, Asif Uddin, and Bikram Kesari Ratha. "Time series prediction QoS routing in software defined vehicular ad-hoc network." 2015 International Conference on Man and Machine Interfacing (MAMI). IEEE, 2015.
5. Huang, Cheng, Rongxing Lu, and Kim-Kwang Raymond Choo. "Vehicular fog computing: architecture, use case, and security and forensic challenges." IEEE Communications Magazine 55.11 (2017): 105-111.
6. YANG, JIANJUN, and YUMING MAO. "Compound Model of Task Arrivals and Load-Aware Offloading for Vehicular Mobile Edge Computing Networks."
7. Ning, Zhaolong, Jun Huang, and Xiaojie Wang. "Vehicular fog computing: Enabling real-time traffic management for smart cities." IEEE Wireless Communications 26.1 (2019): 87-93.

AUTHORS FROFILE

Santosh Kumar Sahoo, is working as an Assistant Professor in the Department of Computer Science & Engineering and Information Technology, Trident Academy of Technology, Bhubaneswar, Odisha. He has done his M.Tech. in Computer Science and Engineering from KIIT Deemed to be University, Bhubaneswar, Odisha in the year 2008 and continuing his Ph.D. in the Department of Computer Engineering, SoA Deemed to be University, Bhubaneswar, Odisha. He has twelve years of academic experience and five years of experience in innovation, entrepreneurship and incubation development from MSME, DST, EDII and other Govt. Agencies. His research area of interests are Computer Network, Wireless Sensor Network, Ad-hoc & Vehicular Ad-hoc Network and Mobile Edge Computing. He has published six research papers in various journals and conferences. He is a life time member of Indian Society of Technical Education (ISTE) and Odisha Information Technology Society (OITS).

Asif Uddin Khan, is working as an Asst. Professor in the Dept of CSE, Silicon Institute of technology Bhubaneswar. He has done BE in Computer Science & Engg. from C.V Raman College of Engineering, Bhubaneswar, M.Tech in Computer Science & Engg from IIIT Bhubaneswar and PhD in Computer science from Utkal University, Bhubaneswar. Served in both industry and academics. His research interest includes Connected Vehicles, Cryptography and Network security, Fog Computing, Edge Computing, Cloud Computing and Software Defined Networks.

Ajit Kumar Nayak, is the professor and HoD of the Department of Computer Science and Information Technology, Siksha 'o' Anusandhan Deemed to be University, Bhubaneswar, Odisha. He has done his graduation in Electrical Engineering from the Institution of Engineers, India in the year 1994, M. Tech degree in computer science from Utkal University in 2001 and Ph. D Computer Science from Utkal University in 2010. His research interests include Computer Networking, Adhoc & Sensor Networks, Machine Learning, Natural Language Computing, Speech and Image Processing etc. He has published about 55 research papers in various journals and conferences. Also co-authored a book 'Computer Network Simulation using NS2'. He has also participated as organizing member of several conferences and workshops in international and national level.