RSU-based Dynamic Clustering in Vehicular Networks

Software: NetSim Standard v14.1, SUMO 1.19.0, Visual Studio 2022, MATLAB 2019 or higher

Project Download Link:

https://github.com/NetSim-TETCOS/RSU-based-Clustering-in-VANETs-in-v14.1/archive/refs/heads/main.zip

Follow the instructions specified in the following link to download and setup the Project in NetSim:

https://support.tetcos.com/en/support/solutions/articles/14000128666-downloading-and-setting-up-netsim-file-exchange-projects

VANETs:

Vehicular Ad Hoc Networks (VANETs) play a vital role in modern transportation systems, enabling vehicles to communicate and cooperate for enhanced safety and efficiency. In VANETs, dynamic vehicle clustering is a key technique used to organize vehicles into groups based on proximity and communication needs. This clustering, facilitated by Roadside Units (RSUs), optimizes communication, reduces overhead, and adapts to changing network conditions. In this document, we explore RSU-based dynamic vehicle clustering in VANETs, implementation, and simulation results to understand its potential for improving vehicular communication.

Figure 1: Vehicular Networks in Real world

Roadside Units (RSUs) are fundamental to the operation of Vehicular Ad Hoc Networks (VANETs). Positioned strategically along roads and intersections, RSUs serve as fixed points of communication, ensuring reliable connections unlike vehicles, which can move around. RSUs are equipped with strong communication capabilities and are linked to the internet and backend servers.

This facilitates the seamless operation of various applications and services within VANETs.

Clustering in VANETs

In vehicular networks, clustering denotes the systematic grouping of vehicles based on factors such as proximity, communication requirements, and mobility dynamics. This approach aims to enhance communication and coordination among vehicles by grouping them into clusters. Clusters facilitate localized communication, resource sharing, and dynamic adaptation to changing network conditions, thereby enhancing overall network efficiency, and promoting safer and more efficient transportation systems.

Clustering Algorithm

- Implements a basic clustering algorithm where vehicles are initially assigned to the closest RSU-based cluster.
- If a vehicle moves closer to another RSU, it will change its cluster accordingly.
- The RSU index serves as the cluster ID for each vehicle.
- The function focuses on assigning vehicles to clusters and visualizing the resulting clustering pattern.

Implementation and code modifications

The modifications involve calling MATLAB with the required inputs for the DSR project (vehicle and RSU coordinates, number of vehicles, and RSUs). MATLAB processes this data and returns the cluster assignments and the vehicles within each cluster.

Once the cluster is formed, the NetSim code is modified to route the data traffic from the source (vehicles) to the cluster heads (RSUs) and from the cluster heads to the destination.

A Clustering.c file is added to the DSR project which contains the following functions:

fn_NetSim_Dynamic_Clustering_Init() //This function initializes all parameter values.

fn_NetSim_dynamic_clustering_run() //This function makes a call to MATLAB interfacing function and passes the inputs from NetSim (i.e.) the num of vehicles and RSUs along with its coordinates.

fn_NetSim_dynamic_clustering_CheckDestination() //This function is used to determine whether the current device is the destination.

fn_NetSim_dynamic_clustering_GetNextHop() //This function statically defines the routes within the cluster and from the cluster to the sink node. It returns to the next hop based on the static routing that is defined.

NetSim_dynamic_clustering_IdentifyCluster() //This function returns the cluster id of the cluster to which a sensor belongs.

fn_netsim_dynamic_form_clusters() //This function assigns each vehicle to its respective clusters based on the cluster IDs obtained from MATLAB.

fn_netsim_assign_cluster_heads() //This function assigns the cluster heads for each cluster based on the cluster head IDs obtained from MATLAB.

NOTE: To run this code 64- bit version of MATLAB must be installed in your system.

Configuring Environment for MATLAB Execution:

1. In Control panel open>System>Advanced system settings>Edit the system environment variable>environment variables
2. Add the following MATLAB install directory path in the Environment PATH variable <MATLAB_INSTALL_DIRECTORY>\bin\win64
   For eg: C:\Program Files\MATLAB\R2023a\bin\win64

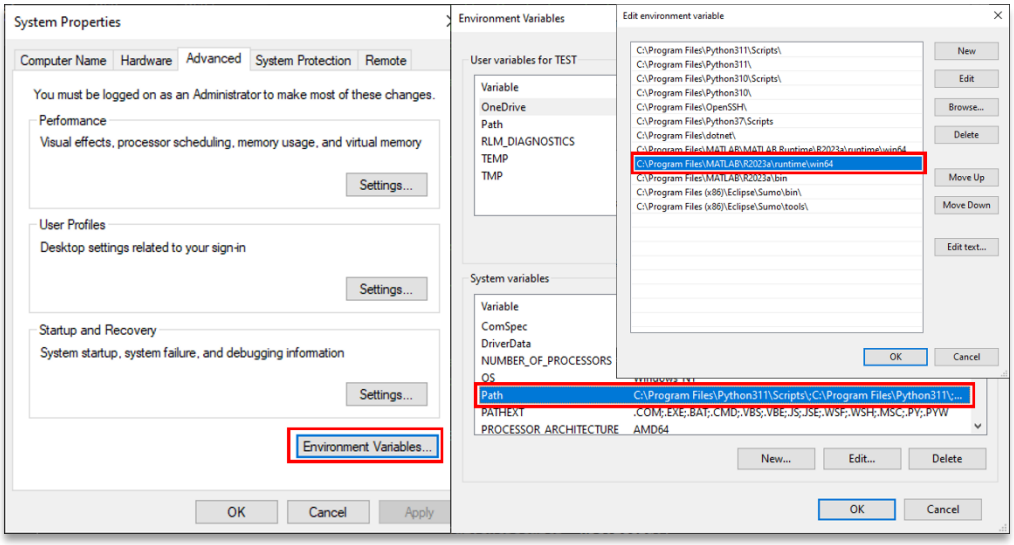

Figure 2: Environment variable PATH

Note: If the machine has more than one MATLAB installed, the directory for the target platform must be ahead of any other MATLAB directory (for instance, when compiling a 64-bit application, the directory in the MATLAB 64-bit installation must be the first one on the PATH).

Example

1. The **RSU_based_Clustering_in_VANETs_v14_1** comes with sample configuration, to open the sample, go to your work and click on **Clustering_Example_1** from the list of experiments.
2. The saved network scenario consists of 4 vehicles, 3 RSUs, a router and wired node (assuming as a SERVER)

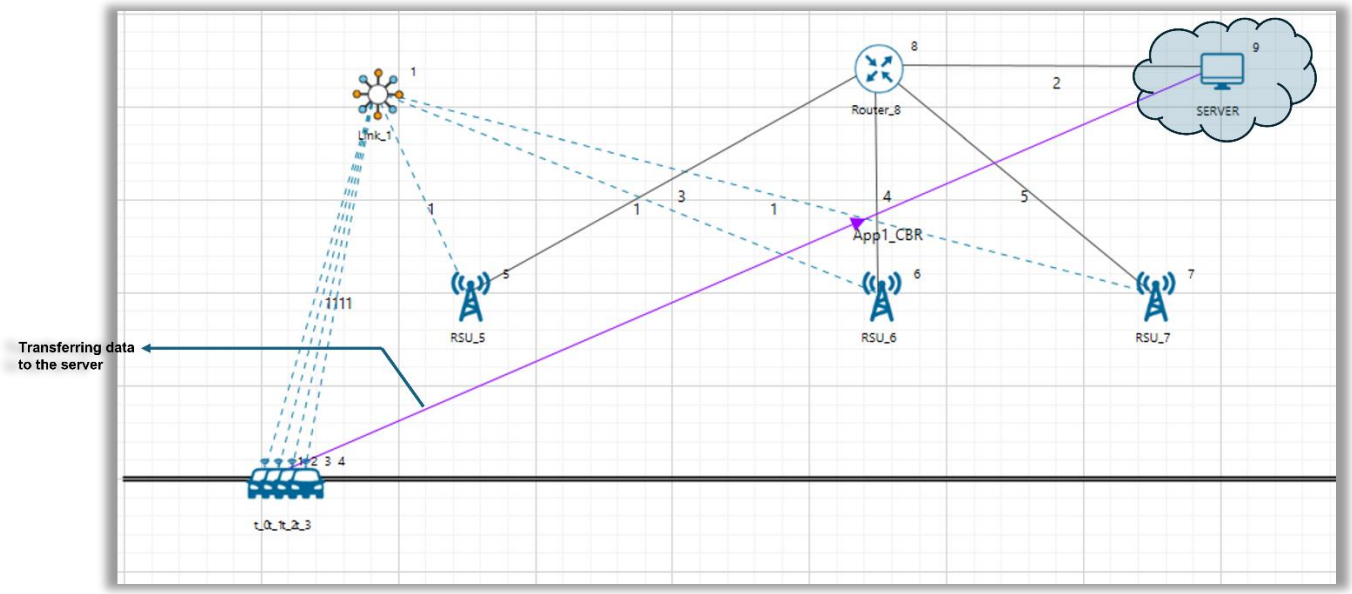

Figure 3: Network Scenario in NetSim

3. Run the simulation and press any key to continue. NetSim simulation console will show the following message in the console “Waiting for NetSim MATLAB Interface to connect...”. NetSim will automatically open the MatlabInterface.exe console window.

4. It will open the MatlabInterface.exe console window. You will observe that as the simulation starts in NetSim, MATLAB gets initialized and the graph associated with energy consumption in the sensor network is plotted during runtime.

Discussion

In this scenario, vehicles in a Vehicular Ad Hoc Network (VANET) dynamically join clusters led by nearby Roadside Units (RSUs). As vehicles move, they switch clusters depending on their proximity to different RSUs. This dynamic clustering ensures efficient communication and data transmission to the server.

By designating RSUs as cluster heads, vehicles can easily determine which cluster they belong to based on their location. This approach optimizes communication and reduces delays in transmitting data.

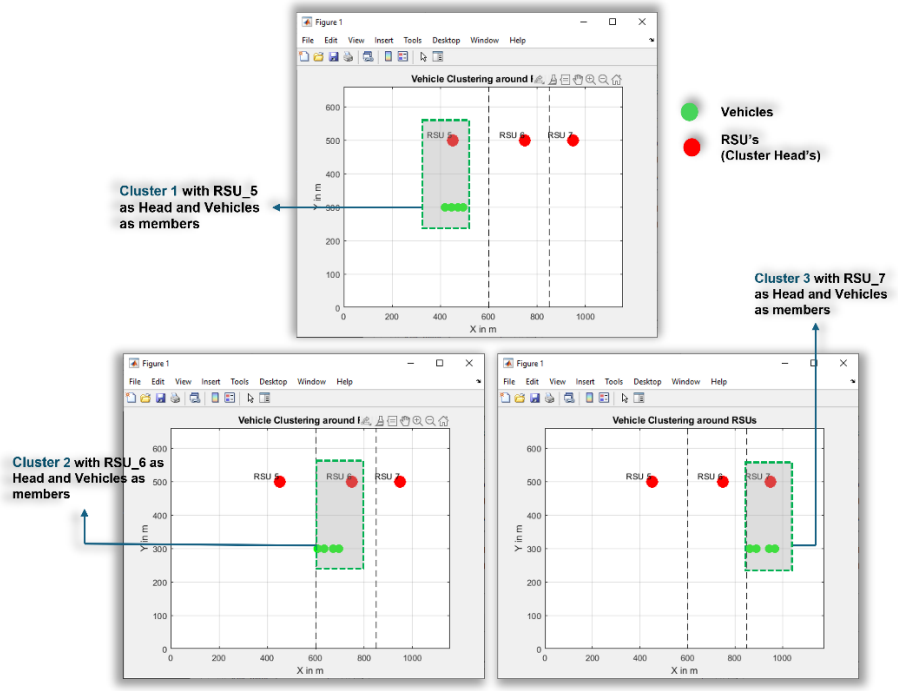

Figure 4: Dynamic clusters with vehicles in motion

Conclusion

The dynamic clustering approach used in this scenario proves effective in optimizing communication and data transmission in VANETs. By allowing vehicles to switch clusters as they move, the network maintains efficient connectivity and adapts to changing conditions.

Changes in clusters affect data transmission, which can be observed through packet traces.

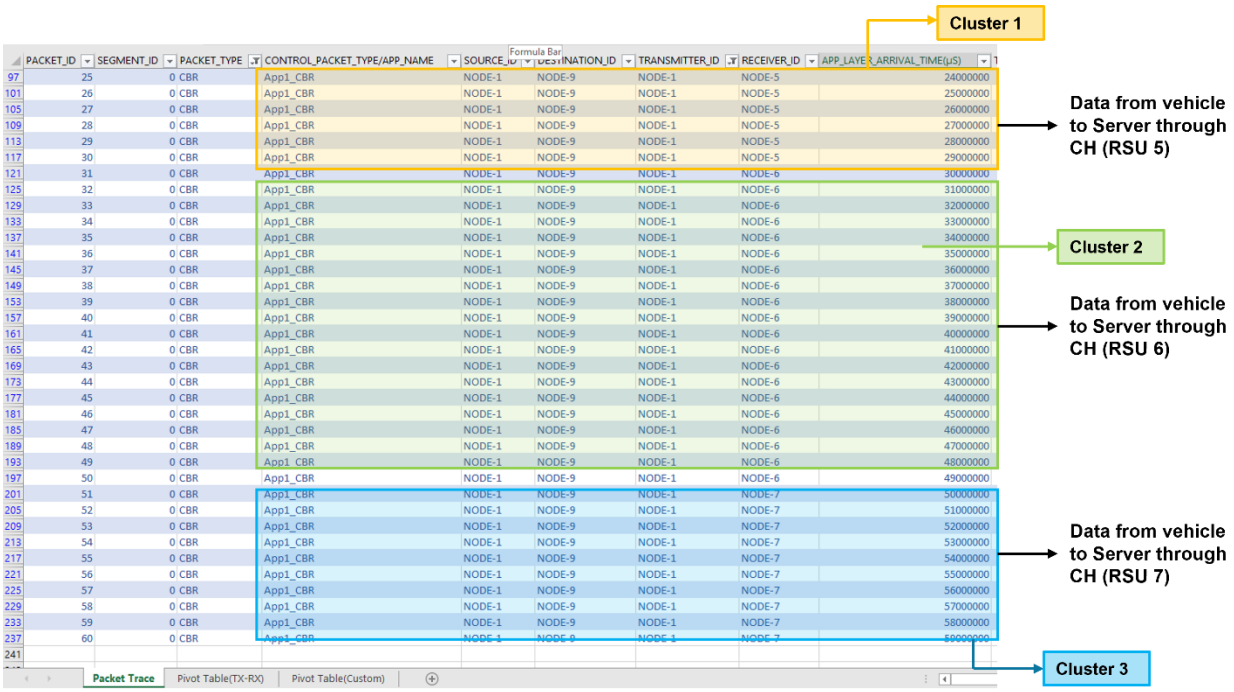

Figure 5: Packet Trace - Data Transmission as Vehicle Moves Across Clusters

Limitations

This MATLAB-based clustering approach is suitable for scenarios involving mobile vehicles. However, there are certain limitations to consider:

- Vehicles must be mobile.
- Roadside Units (RSUs) are necessary and act as cluster heads (CH) in the network.
- The number of clusters corresponds to the number of RSUs in the network.

Note:

When creating a VANET network, include RSUs for simulations using this modified workspace.

In the shared workspace, there are also two other experiments for reference: one with a real-world scenario and another with a large vehicular network.