Dynamic Clustering in WSN

Software Recommended: NetSim Standard v12.0 (32/64 bit), Visual Studio 2015/2017/2019, MATLAB (32/64 bit)

Follow the instructions specified in the following link to clone/download the project folder from GitHub using Visual Studio:

<u>https://tetcos.freshdesk.com/support/solutions/articles/14000099351-how-to-clone-netsim-file-exchange-project-repositories-from-github-</u>

Other tools such as GitHub Desktop, SVN Client, Sourcetree, Git from the command line, or any client you like to clone the Git repository.

Note: It is recommended not to download the project as an archive (compressed zip) to avoid incompatibility while importing workspaces into NetSim.

Secure URL for the GitHub repository:

https://github.com/NetSim-TETCOS/Dynamic_Clustering_Project_v12.0.git

Clustering in WSN:

Clustering is the process partitioning a group of sensors into small numbers of clusters. In environments where the sensors are mobile clusters cannot be static. Like cluster heads in each cluster are elected dynamically, the members in each cluster also need to be dynamically identified. Therefore, the size of each cluster is not fixed and can vary depending on the position of the sensors.

Dynamic Clustering helps in efficiently grouping sensors into clusters dynamically. There is no fixed cluster size and the sensors are divided into the required number of clusters with members of each cluster calculated dynamically.

Clustering using k-means algorithm:

kmeans(X,k) partitions the points in the n-by-p data matrix X into k clusters. This iterative partitioning minimizes the sum, over all clusters, of the within-cluster sums of point-to-cluster-centroid distances. Rows of X correspond to points, columns correspond to variables. kmeans returns an n-by-1 vector IDX containing the cluster indices of each point. By default, kmeans uses squared Euclidean distances. When X is a vector, kmeans treats it as an *n*-by-1 data matrix, regardless of its orientation.

The sensor positions and number of clusters,

X - a matrix containing the x, y coordinates of the sensors in the scenario

k- the number of clusters

are passed to k-means algorithm.

[IDX,C] = kmeans(X,k)

IDX – Contains the cluster id's of each sensor (i.e) the cluster to which the sensor belongs.

C - Centroids of each cluster

Clustering using Fuzzy C-Means Algorithm:

Fuzzy c-means (FCM) is a data clustering technique in which a dataset is grouped into n clusters with every data point in the dataset belonging to every cluster to a certain degree. For example, a certain

data point that lies close to the center of a cluster will have a high degree of belonging or membership to that cluster and another data point that lies far away from the center of a cluster will have a low degree of belonging or membership to that cluster.

Cluster head election based on distance from Centroid:

After grouping the sensors into different clusters, the cluster heads are determined based on the distance between the sensor and the centroid of the cluster to which it belongs.

The sensor which is closer to the centroid will be elected as the cluster head. Here the position values (i.e. value of x-coordinate and y-coordinate) of each sensor are passing from NetSim to MATLAB as a sole parameter.

Cluster head election based on distance and power:

After grouping the sensors into different clusters, the cluster heads are determined based on the distance between the sensor and the remaining power of each sensor. After that the sensors are assigned in respective cluster.

The sensor which is closer to the centroid and has the more power than other sensor will be elected as the cluster head. Here the position values (i.e. value of x-coordinate and y-coordinate) of each sensor and power are passing from NetSim to MATLAB as a sole parameter.

Dynamic Clustering in NetSim with MATLAB Interfacing:

Dynamic Clustering is implemented in NetSim by Interfacing with MATLAB for the purpose of mathematical calculation. The sensor coordinates are fed as input to MATLAB and k-means algorithm that is implemented in MATLAB is used to dynamically perform clustering of the sensors into n number of clusters.

In addition to clustering we also determine the cluster head of each cluster mathematically in MATLAB. The distance of each sensor from the centroid of the cluster to which it belongs is calculated. Then the sensor which has the least distance is elected as the cluster head.

From MATLAB we get the cluster id of each sensor, cluster heads of each cluster and the size of each cluster.

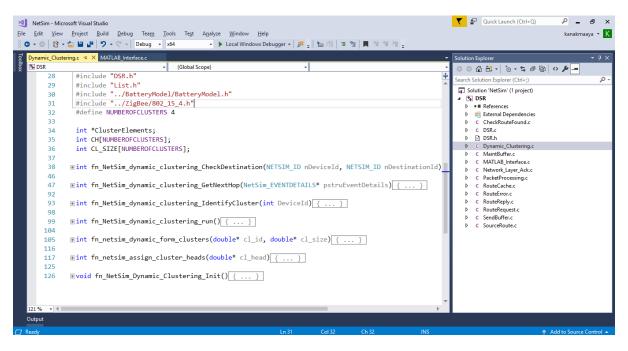
All the above steps are performed periodically which can be defined as per the implementation. Each time the cluster members and the cluster heads are determined based on the current position and they are not fixed.

The codes required for the mathematical calculations done in MATLAB are written to a **clustering.m** file as shown below:

Ile Edit Format ⊻lew Help	- 1	3
% % Copyright (C) 2016 % TETCOS, Bangalore. India % % Tetcos sums the intellectual property rights in the Product and its content. % The copying redistribution, reselling or publication of any or all of the % Product or its content without express prior written consent of Tetcos is % prohibited. Omership and / or any other right relating to the software and all % intellectual property rights therein shall remain at all times with Tetcos. % Author: Dhruwang % Author: Dhruwang	% %	x
% = 3 KMeans using distance and	Means using distance power Means using distance and power	
<pre>% save dynamic_clustering.mat %change here for different algorithm if(Clustering_Method == 1 Clustering_Method == 3) [IDX,C]= k_means(x,num_cls); else [IDX,C]= fuzzy(x,num_cls); end</pre>		

The clustering.m file can be run in four different modes cluster head election.

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



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 cluster to sinknode. It returns the next hop based on the static routing that is defined.

fn_NetSim_dynamic_clustering_IdentifyCluster()

This function returns the cluster id of the cluster to which a sensor belongs.

fn_NetSim_dynamic_clustering_run()

This function makes a call to MATLAB interfacing function and passes the inputs from NetSim (i.e) the sensor coordinates, number of clusters and the sensor count.

fn_netsim_dynamic_form_clusters()

This function assigns each sensor to its respective clusters based on the cluster id's obtained from MATLAB.

fn_netsim_assign_cluster_heads()

This function assigns the cluster heads for each cluster based on the cluster head id's obtained from MATLAB.

fn_NetSim_Dynamic_Clustering_Init()

This function initializes all parameter values.

Static Routing:

Static Routing is defined in such a way that the sensors in the cluster send the packets to the cluster head. The cluster head then directly sends the packets to the destination (sinknode).

If the current sensor is the source device and if it is not a cluster head then its next hop is its cluster head.

If the current sensor is the source device and if it is a cluster head then its next hop is the destination (i.e) the sinknode.

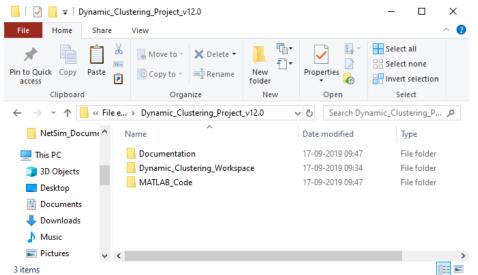
If the current sensor is not the source then the packet is sent to the destination (i.e) the sinknode.

NOTE:

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

Steps:

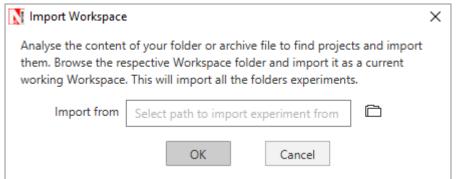
1. The downloaded project folder contains the folders Documentation, MATLAB_Code and Dynamic_Clustering_Workspace directory as shown below:



2. Import Dynamic_Clustering_Workspace by going to Open Simulation->Workspace Options->More Options in NetSim Home window. Then select Import as shown below:

NetSim Home					- 0	×
N Reserve						~
NetSim Standard Network Simulation/Emulation Platfor Version 12.0.18 (64 Bit)					www.tetc	NETSIM
	Current workspace: NetSim_1	2.0.18_64_std_default				
New Simulation Ctrl+N	Workspace name	Location	Description			
	NetSim_12.0.18_64_std_default	C:\Users\Namrata\Documents	-	Export	Ē	
Open Simulation Ctrl+O						
Examples						
License Settings	New	Set as Current			Back]
Support	Learn	ſ	Documentation	Contact us		
Answers/FAQ	Videos		lser Manual	Email - sales@t Phone - +91 76		
Contact Technical Support Email - support@tetcos.com	Experiments Manu		echnology Libraries ource Code Help	Phone - +91 /6	005 4321	

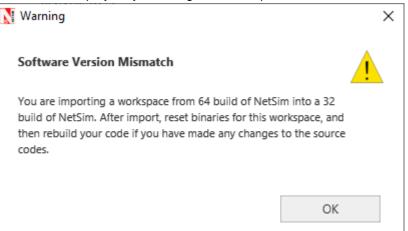
3. It displays a window where users need to give the path of the workspace folder and click on OK as shown below:



4. Browse to the Dynamic_Clustering_Workspace folder and click on select folder as shown below:

N Select Folder				×
🔶 🔶 👻 🕇 📙 « Dyna	mic_Clustering_Proj > Dynamic_Clustering	_Workspace v ඊ	Search Dynamic_0	Clustering 🔎
Organize 👻 New folder				∷ .
 OneDrive 	Name	Date modified	Туре	Size
> NetSim_Docume	📙 bin	16-09-2019 18:09	File folder	
🗸 💶 This PC	Dynamic_Clustering_Example	17-09-2019 09:20	File folder	
_	lcons	16-09-2019 18:09	File folder	
> 🧊 3D Objects	src src	16-09-2019 18:09	File folder	
🗧 📃 Desktop				
🗧 🗄 Documents				
> 🕹 Downloads				
> 🁌 Music				
> 💽 Pictures				
> 📑 Videos				
🔿 🏪 New Volume (C:				
💦 👝 New Volume (D: 🗸 <				:
Folder:	Dynamic_Clustering_Workspace			
		l l	Select Folder	Cancel

- 5. After this click on OK button in the Import Workspace window.
- **6.** While importing the workspace, if the following warning message indicating Software Version Mismatch is displayed, you can ignore it and proceed.



7. The Imported workspace will be set as the current workspace automatically. To see the imported workspace, click on Open Simulation->Workspace Options->More Options as shown below:

NetSim Home						- 🗆 X
NetSim Sta Network Simulation/En Version 12.0.18 (64 Bit)	nulation Platfo					www.tetcos.com
		Current workspace: Dynamic_	Clustering_Workspace			
New Simulation	Ctrl+N	Workspace name	Location	Description		
		Dynamic_Clustering_Workspace	D:\File exchange v12.0\v12.0\Dynam.		Export	Ē
Open Simulation	Ctrl+O	NetSim_12.0.18_64_std_default	C:\Users\Namrata\Documents	-	Export	Ē
Examples License Settings		New Im	port Set as Current			Back
Support		Learn	Do	cumentation	Contact us	
Answers/FAQ Contact Technical Supp Email - support@tetco:		Videos Experiments Manu	al Tech	r Manual nnology Libraries rce Code Help	Email - sales@tet Phone - +91 767	

8. Create a user variable with the name of MATLAB_PATH and provide the path of the installation directory of user's respective MATLAB version.

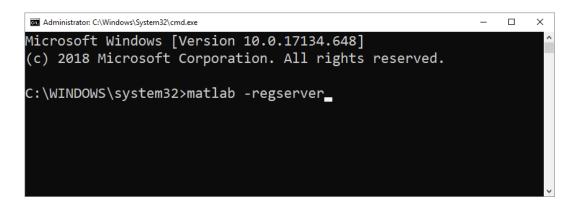
Edit User Variable	—
Variable name: Variable value:	MATLAB_PATH C:\Program Files\MATLAB\R2016a
	OK Cancel

9. Make sure that the following directory is in the PATH(Environment variable) <Path where MATLAB is installed>\bin\win64

Edit System Variable	—
Variable name: Variable value:	Path h Files \MATLAB \R 20 16a \bin \win64 C: \Progra OK

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).
10. Open Command prompt as admin and execute the command "matlab -regserver". This will register MATLAB as a COM automation server and is required for NetSim to start MATLAB

automation server during runtime.



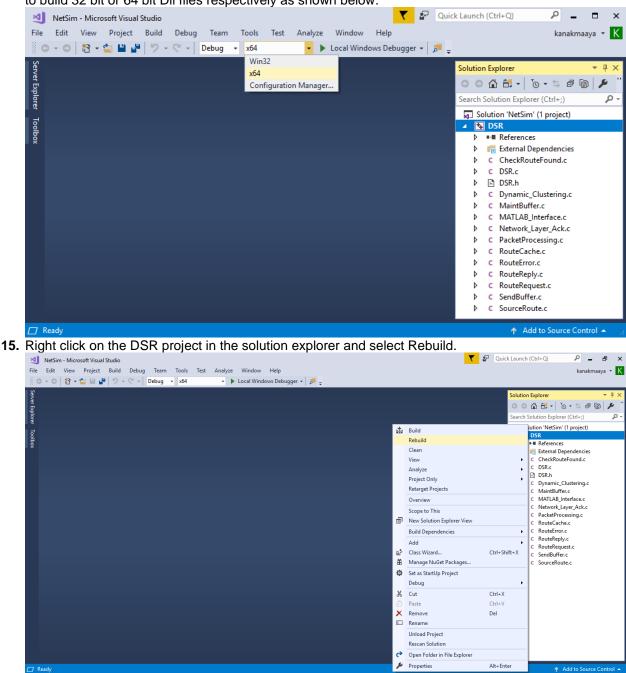
11. Place **clustering.m** present in the MATLAB_Code folder inside the root directory of MATLAB. For Example: **"C:\Program Files\MATLAB\R2016a"**.

📙 🛃 📕 🖛 M	/IATLAB_Code	2				_		×
File Home	Share	View						^ ?
Pin to Quick Copy access	Paste	Move to •	K Delete ▾ I Rename	New folder	Properties	Select a	none	
Clipboard	đ	Organiz	e	New	Open	Sele	ct	
← → • ↑	, « Dynami	c_Clustering_Pro	> MATLAB	Code 🗸	・ Ö Search MA	TLAB_Code		Q
MATLAB	^ N	lame	^		Date modified	Тур	be	
PerfLogs		clustering.m			16-09-2019 18:07	М	File	
Program F	iles							
📙 Program F	iles (
Python27								
Reprise								
sumo-0.32	2.0							
sumo-1.2.	0 🗸 <							>
1 item 1 item se	lected 5.33 K	В						:==

12. Open the Source codes in Visual Studio by going to Open Simulation-> Workspace Options and Clicking on Open code button as shown below:

NetSim Home					- 🗆 X
NetSim Standard Network Simulation/Emulation Platfo Version 12.0.18 (64 Bit)					www.tetcos.com
	Current workspace: Dynamic_Clustering	g_Workspace		C Experiment name	
New Simulation Ctrl+N	Experiment name	Date modified	Network type		
Open Simulation Ctrl+O	Dynamic_Clustering_Example	17-09-2019	Wsn	View Results	Export 🗂
Examples					
License Settings	Open code Reset Code	Reset Binarie	s More optic	ns	Back
Support	Learn		Documentation	Contact u	JS
Answers/FAQ Contact Technical Support Email - support@tetcos.com	Videos Experiments Manual	1	User Manual Technology Libraries Source Code Help		les@tetcos.com -91 767 605 4321

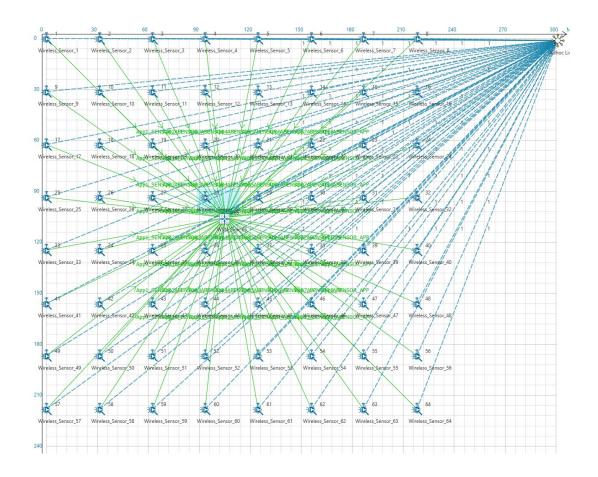
- **13.** Under the DSR project in the solution explorer you will be able to see that **MATLAB_Interface.c** and **Dynamic_Clustering.c** files which contain source codes related to interactions with MATLAB and handling clustering in NetSim respectively.
- **14.** Based on whether you are using NetSim 32 bit or 64 bit setup you can configure Visual studio to build 32 bit or 64 bit DII files respectively as shown below:



- **16.** Upon successful build modified libDSR.dll file gets automatically updated in the directory containing NetSim binaries.
- 17. Run NetSim as Administrative mode.
- **18.** Then Dynamic_Clustering_Workspace comes with a sample configuration that is already saved. To open this example, go to Open Simulation and click on the Dynamic_Clustering_Example that is present under the list of experiments as shown below:

NetSim Home						- 🗆 X
NetSim Star Network Simulation/Emula Version 12.0.18 (64 Bit)						www.tetcos.com
		Current workspace: Dynamic_Cluster	ring_Workspace		C Experiment name	
New Simulation C	trl+N	Experiment name	Date modified	Network type		
Open Simulation C	Ctrl+O	Dynamic_Clustering_Example	17-09-2019	Wsn	View Re	esults Export
Examples						
License Settings						
		Open code Reset Co	de Reset Bina	ries More opti	ions	Back
Support		Learn		Documentation	Co	ntact us
Answers/FAQ Contact Technical Support Email - support@tetcos.cc		Videos Experiments Manual		User Manual Technology Libraries Source Code Help		iail - sales@tetcos.com one - +91 767 605 4321

19. The saved network scenario consisting of 64 sensors uniformly distributed in the grid environment along with a sink node forming a Wireless Sensor Network. Traffic is configured from each sensor node to the Sink Node.



20. Run the Scenario. You will observe that as the simulation starts in NetSim, MATLAB gets initialized and graph associated with energy consumption in the sensor network is plotted during runtime.

Analysis:

A total of 64 sensors are placed evenly on the grid environment and each sensor is set to have equal initial energy.

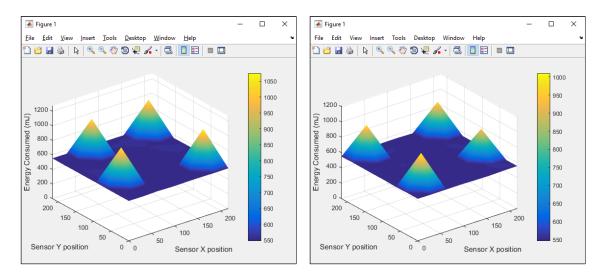
At the end of the simulation, NetSim provides Battery Model Metrics which provides detailed information related to energy consumption in each sensor node with respect to transmission, reception, idle mode, sleep mode etc as shown below:

Batter model	\checkmark	Detailed View				
Device Name	Initial energy(mJ)	Consumed energy(mJ)	Remaining Energy(mJ)	Transmitting energy(mJ)	Receiving energy(mJ)	Idle energy(mJ
WIRELESS_SENSOR_1	6480.000000	558.318835	5921.681165	20.038487	0.000000	538.280348
WIRELESS_SENSOR_2	6480.000000	556.923016	5923.076984	18.593404	0.000000	538.329613
WIRELESS_SENSOR_3	6480.000000	558.504945	5921.495055	20.231165	0.000000	538.273780
WIRELESS_SENSOR_4	6480.000000	557.202180	5922.797820	18.882420	0.000000	538.319760
WIRELESS_SENSOR_5	6480.000000	556.457743	5923.542257	18.111709	0.000000	538.346034
WIRELESS_SENSOR_6	6480.000000	557.853562	5922.146438	19.556793	0.000000	538.296770
WIRELESS_SENSOR_7	6480.000000	557.481344	5922.518656	19.171437	0.000000	538.309907
WIRELESS_SENSOR_8	6480.000000	555.806361	5924.193639	17.437337	0.000000	538.369024
WIRELESS_SENSOR_9	6480.000000	557.574399	5922.425601	19.267776	0.000000	538.306623
WIRELESS SENSOR 10	6480.000000	629.662276	5850.337724	35,163691	58,563994	535,934592

This information can also be obtained at different points of simulation time either to log or to send to other external tools. The battery information and the position coordinates are passed to MATLAB periodically for clustering (number of cluster is set to 4), cluster head election and to obtain energy consumption plots.

Cluster head election using distance alone as a parameter:

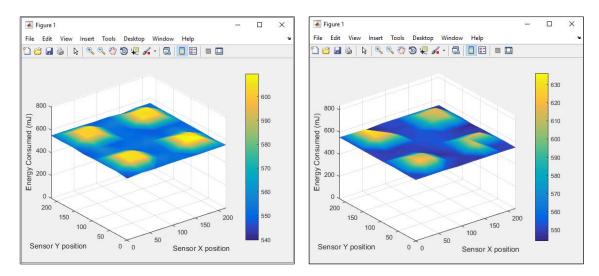
Running simulations with Clustering Method set to 1 and 2 in the **clustering.m** file will provide energy consumption plots for kmeans and fuzzy c-means algorithms respectively as shown below:



As it is seen from the plot, there are 4 peaks in the plot corresponding to higher energy consumption in the nodes in the center of the cluster, as they always become the cluster heads. This is because distance is used as a parameter for electing the cluster heads.

Cluster head election using distance and remaining energy as parameters:

Running simulations with Clustering Method set to 3 and 4 in the **clustering.m** file will provide energy consumption plots for kmeans and fuzzy c-means algorithms respectively as shown below:



In the initial phase the plot resembles the previous one. However as the time passes, it can be observed that the power is consumed by all the sensors at approximately the same rate.

There are no sharp peaks in this plot unlike the previous one because modified K-means takes into account the power level of each sensor and thus sensors other than those in the center of the cluster will also get a chance to be elected as the cluster head in its respective cluster.