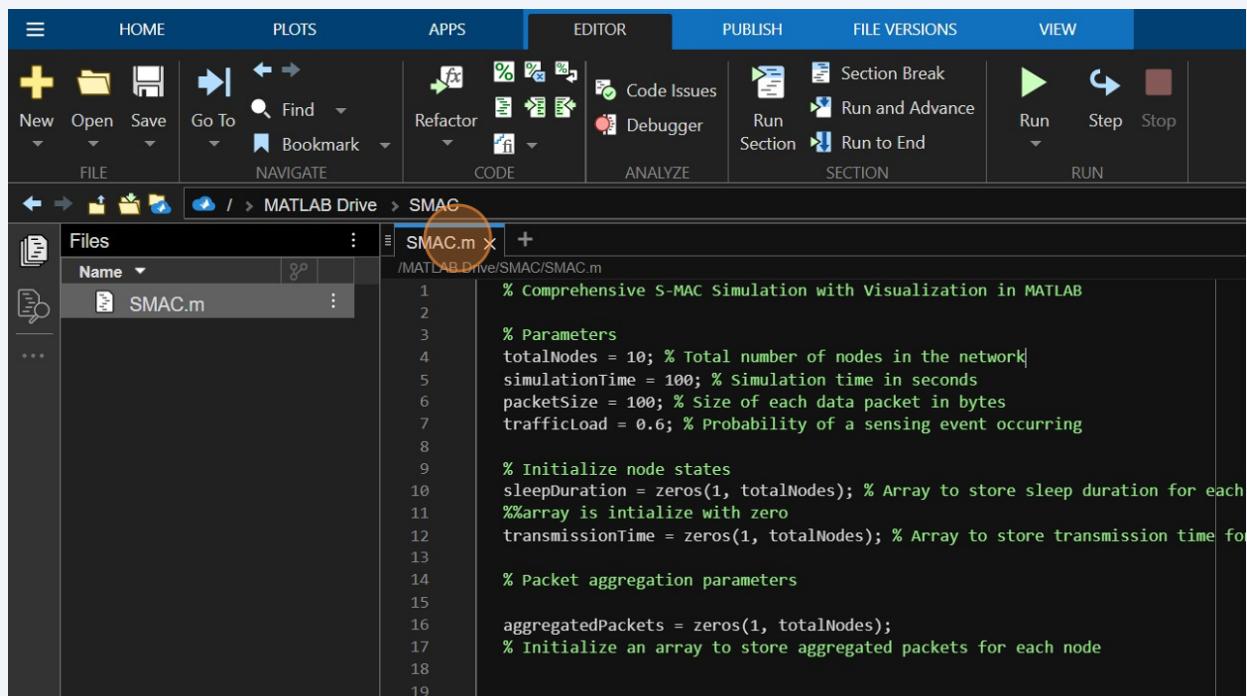


Step-by-step Implementation of SMAC for network simulation on MATLAB (with code)

Scribe

- 1 Navigate to <https://matlab.mathworks.com/>

- 2 We are going to Implement SMAC So First Lets know about SMAC:
The Sensor-MAC (S-MAC) protocol is a specialized medium access control (MAC) protocol tailored for the distinct requirements of wireless sensor networks (WSNs). Prioritizing energy optimization, S-MAC introduces a strategic **duty cycling mechanism**, enabling nodes to alternate between **active and sleep states**, effectively curbing unnecessary energy consumption during periods of inactivity and **alleviating the common issue of idle listening** found in traditional MAC protocols. In the simulated environment, each sensor node adheres to a predefined duty cycle, periodically awakening for synchronization with neighboring nodes. This synchronization involves the exchange of control packets—SYNC for periodic alignment, RTS to signal data transmission intent, and CTS for confirmation. Implemented with fixed sleep and awake periods, the protocol ensures a consistent sleep-wake cycle, enhancing energy management for prolonged network longevity. The communication during wake periods unfolds in three phases: SYNC for synchronization, RTS signaling data transmission, and CTS confirming readiness. This meticulous design minimizes energy consumption and addresses hidden terminal problems through the **RTS/CTS handshake**, ensuring efficient and collision-free communication in resource-constrained wireless sensor networks.



The screenshot shows the MATLAB Editor interface with the following details:

- Toolbar:** HOME, PLOTS, APPS, EDITOR (selected), PUBLISH, FILE VERSIONS, VIEW.
- File Explorer:** Shows 'MATLAB Drive' and 'SMAC' folder.
- Editor Area:** Displays the 'SMAC.m' script content.

```
% Comprehensive S-MAC Simulation with Visualization in MATLAB
%
% Parameters
totalNodes = 10; % Total number of nodes in the network
simulationTime = 100; % Simulation time in seconds
packetSize = 100; % Size of each data packet in bytes
trafficLoad = 0.6; % Probability of a sensing event occurring

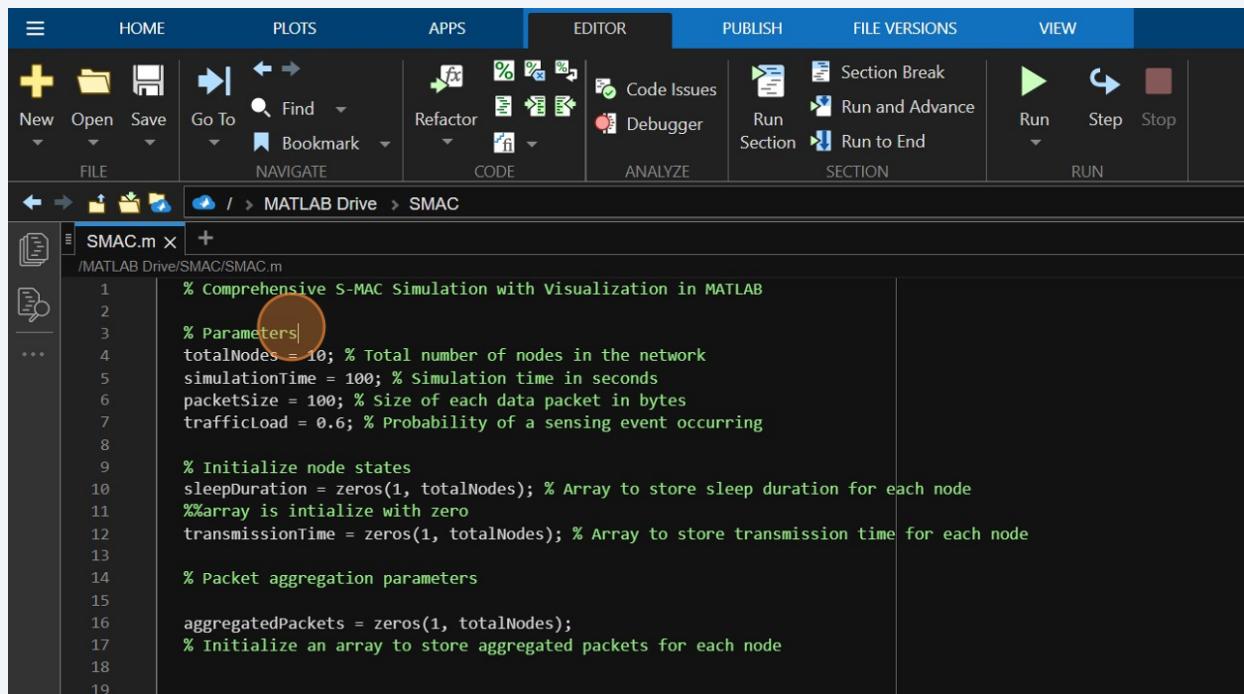
%
% Initialize node states
sleepDuration = zeros(1, totalNodes); % Array to store sleep duration for each
% array is initialize with zero
transmissionTime = zeros(1, totalNodes); % Array to store transmission time for
% each node

%
% Packet aggregation parameters

aggregatedPackets = zeros(1, totalNodes);
% Initialize an array to store aggregated packets for each node
```

3

Now lets see the implemented code first , Starting with the initialization of Parameters



The screenshot shows the MATLAB Editor window with the file `SMAC.m` open. The code is a MATLAB script for a S-MAC simulation. A red circle highlights the line `% Parameters`. The code defines several variables:

```
% Comprehensive S-MAC Simulation with Visualization in MATLAB
% Parameters
totalNodes = 10; % Total number of nodes in the network
simulationTime = 100; % Simulation time in seconds
packetSize = 100; % Size of each data packet in bytes
trafficLoad = 0.6; % Probability of a sensing event occurring

% Initialize node states
sleepDuration = zeros(1, totalNodes); % Array to store sleep duration for each node
%array is intialized with zero
transmissionTime = zeros(1, totalNodes); % Array to store transmission time for each node

% Packet aggregation parameters

aggregatedPackets = zeros(1, totalNodes);
% Initialize an array to store aggregated packets for each node
```

4

First parameter is the totalNodes that is selfexplanatory totalNodes:

- **Definition:** This parameter represents the total number of nodes within the wireless sensor network (WSN) being simulated.

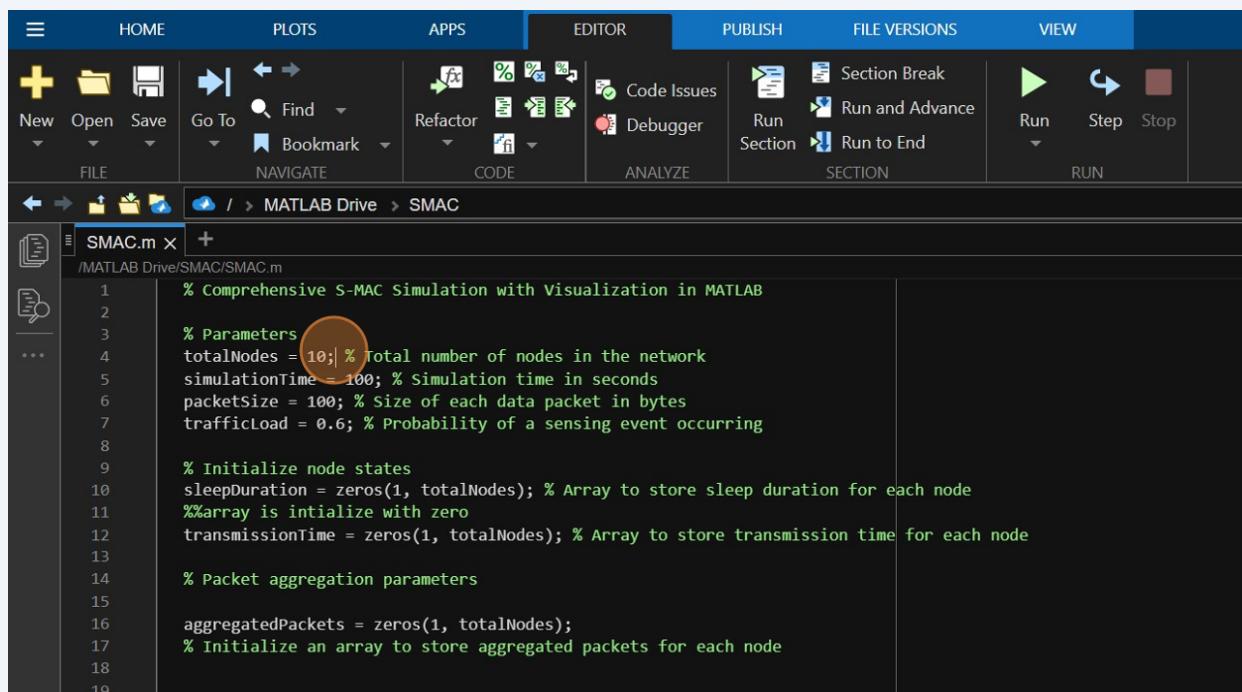
- **Usage:** It influences the size of arrays and loops in the simulation, as the script iterates over each node during the simulation time.

2nd is

simulationTime:

- **Definition:** This parameter specifies the total duration of the simulation in seconds.

- **Usage:** It determines the overall time span for which the simulation will run, influencing the number of iterations in the simulation loop.



The screenshot shows the MATLAB Editor interface with the following details:

- Toolbar:** HOME, PLOTS, APPS, EDITOR (selected), PUBLISH, FILE VERSIONS, VIEW.
- File Explorer:** Shows 'SMAC.m' in the current folder 'MATLAB Drive > SMAC'.
- Code Editor:** Displays the MATLAB script 'SMAC.m' with the following content:

```
% Comprehensive S-MAC Simulation with Visualization in MATLAB

% Parameters
totalNodes = 10; % Total number of nodes in the network
simulationTime = 100; % Simulation time in seconds
packetSize = 100; % Size of each data packet in bytes
trafficLoad = 0.6; % Probability of a sensing event occurring

% Initialize node states
sleepDuration = zeros(1, totalNodes); % Array to store sleep duration for each node
%array is intialized with zero
transmissionTime = zeros(1, totalNodes); % Array to store transmission time for each node

% Packet aggregation parameters

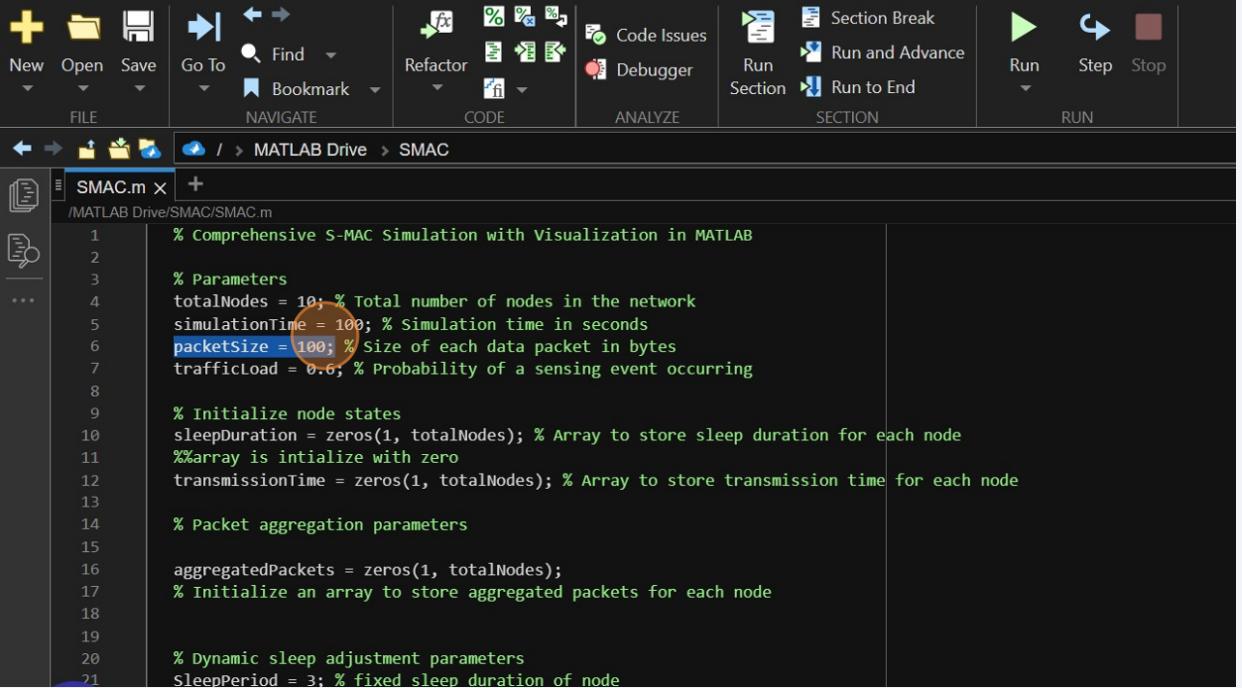
aggregatedPackets = zeros(1, totalNodes);
% Initialize an array to store aggregated packets for each node
```

5

Then comes the third one
packetSize:

- **Definition:** packetSize denotes the size of each data packet in bytes.

- **Usage:** It is crucial in modeling data transmission as it represents the amount of information exchanged between nodes during communication events.



```
% Comprehensive S-MAC Simulation with visualization in MATLAB
%
% Parameters
totalNodes = 10; % Total number of nodes in the network
simulationTime = 100; % Simulation time in seconds
packetSize = 100; % Size of each data packet in bytes
trafficLoad = 0.6, % Probability of a sensing event occurring

%
% Initialize node states
sleepDuration = zeros(1, totalNodes); % Array to store sleep duration for each node
%array is initialize with zero
transmissionTime = zeros(1, totalNodes); % Array to store transmission time for each node

%
% Packet aggregation parameters

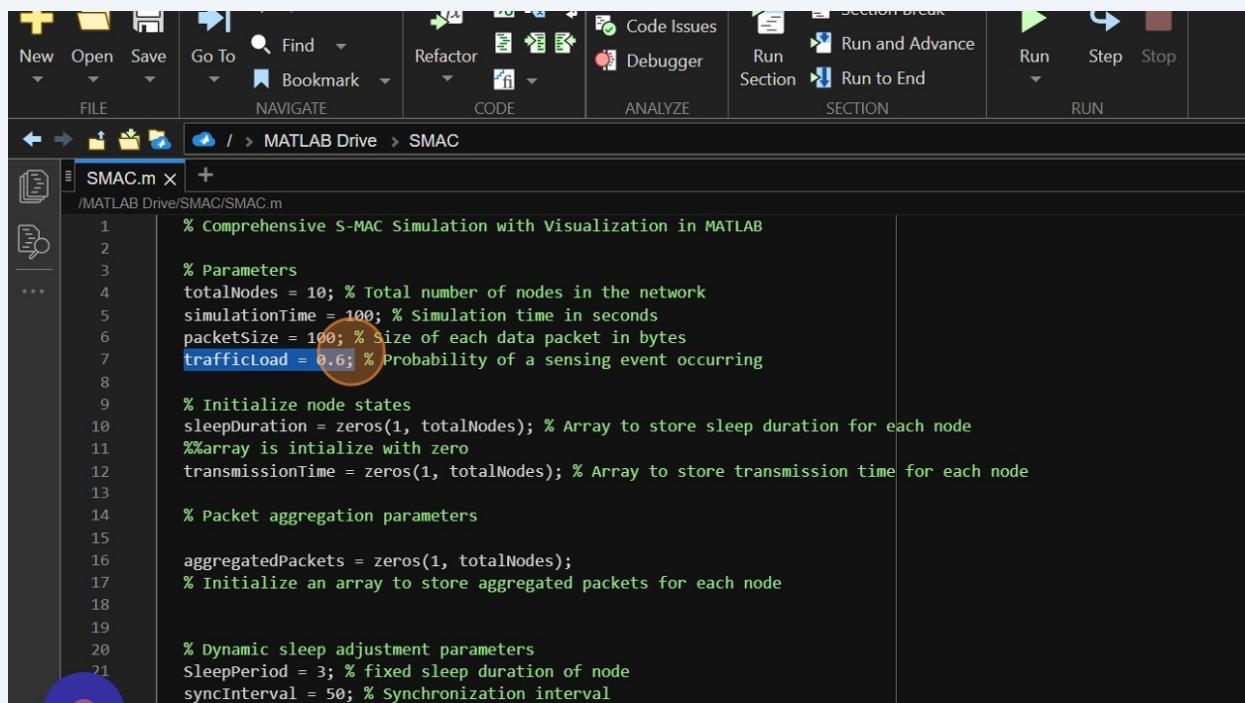
aggregatedPackets = zeros(1, totalNodes);
% Initialize an array to store aggregated packets for each node

%
% Dynamic sleep adjustment parameters
SleepPeriod = 3; % fixed sleep duration of node
```

6 trafficLoad:

• **Definition:** The trafficLoad parameter represents the probability of a sensing event occurring at a given node during each simulation time step.

• **Usage:** It is employed to simulate the occurrence of sensing events in the network. A higher trafficLoad implies a greater likelihood of nodes sensing and triggering communication events.



The screenshot shows the MATLAB IDE interface with the following details:

- Toolbar:** Includes New, Open, Save, Go To, Find, Refactor, Code Issues, Debugger, Run, Run and Advance, Run to End, Run, Step, and Stop buttons.
- Navigation Bar:** Shows the current path: MATLAB Drive > SMAC.
- Editor Area:** Displays the MATLAB script `SMAC.m`. The code defines parameters for a S-MAC simulation, including the total number of nodes (10), simulation time (100 seconds), packet size (100 bytes), and traffic load (0.6). The `trafficLoad` line is highlighted with a blue oval.

```
% Comprehensive S-MAC Simulation with Visualization in MATLAB
%
% Parameters
totalNodes = 10; % Total number of nodes in the network
simulationTime = 100; % Simulation time in seconds
packetSize = 100; % Size of each data packet in bytes
trafficLoad = 0.6; % Probability of a sensing event occurring

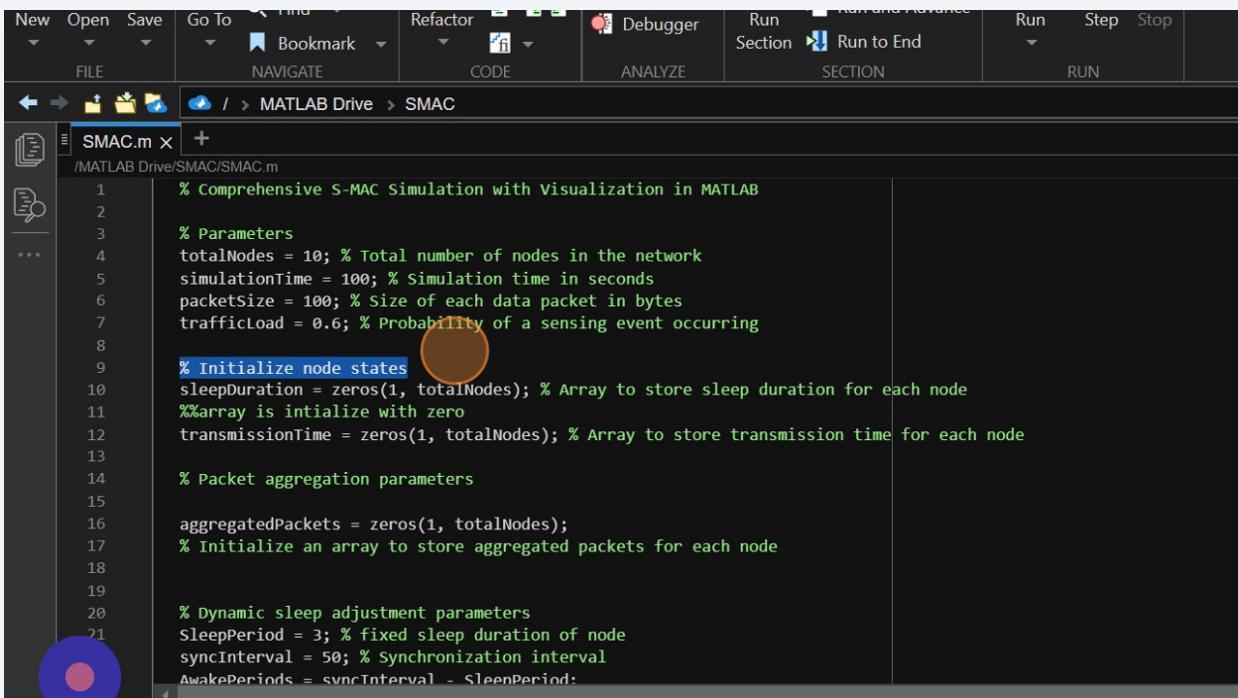
%
% Initialize node states
sleepDuration = zeros(1, totalNodes); % Array to store sleep duration for each node
%array is initialize with zero
transmissionTime = zeros(1, totalNodes); % Array to store transmission time for each node

%
% Packet aggregation parameters

aggregatedPackets = zeros(1, totalNodes);
% Initialize an array to store aggregated packets for each node

%
% Dynamic sleep adjustment parameters
SleepPeriod = 3; % fixed sleep duration of node
syncInterval = 50; % Synchronization interval
```

7 Now lets have some arrays to store the states of nodes



```
% Comprehensive S-MAC Simulation with Visualization in MATLAB

% Parameters
totalNodes = 10; % Total number of nodes in the network
simulationTime = 100; % Simulation time in seconds
packetSize = 100; % Size of each data packet in bytes
trafficLoad = 0.6; % Probability of a sensing event occurring

% Initialize node states
sleepDuration = zeros(1, totalNodes); % Array to store sleep duration for each node
% array is initialize with zero
transmissionTime = zeros(1, totalNodes); % Array to store transmission time for each node

% Packet aggregation parameters

aggregatedPackets = zeros(1, totalNodes);
% Initialize an array to store aggregated packets for each node

% Dynamic sleep adjustment parameters
SleepPeriod = 3; % fixed sleep duration of node
syncInterval = 50; % Synchronization interval
AwakePeriods = syncInterval - SleepPeriod;
```

8

The purpose is to initialize the variable sleepDuration. Let me break it down:

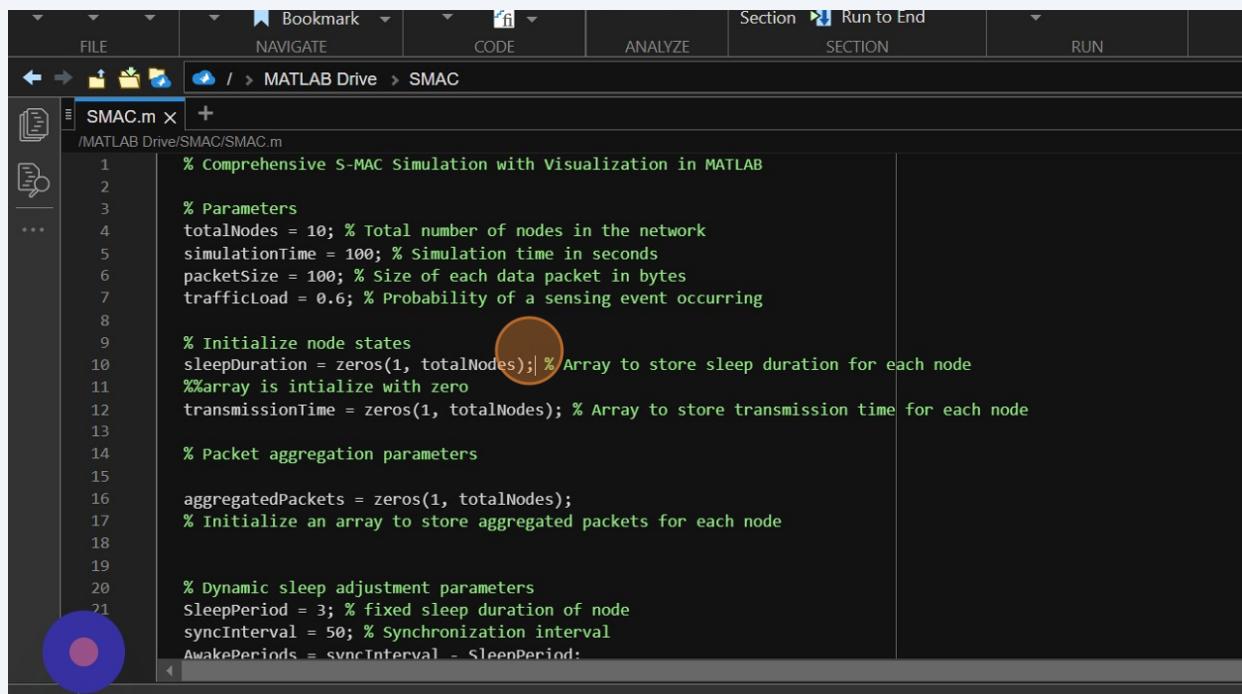
- `zeros(1, totalNodes)`: This part creates a row vector of size `totalNodes` filled with zeros. The `zeros` function in MATLAB generates an array of zeros, and in this case, it creates a row vector with `totalNodes` elements.

- `sleepDuration = ...`: This part assigns the newly created row vector to the variable `sleepDuration`. So, `sleepDuration` is now a row vector containing `totalNodes` elements, all initialized to zero.

`zeros(1, totalNodes)`: This part creates a row vector of size `totalNodes` filled with zeros. The `zeros` function in MATLAB generates an array of zeros, and in this case, it creates a row vector with `totalNodes` elements.

- `transmissionTime = ...`: This part assigns the newly created row vector to the variable `transmissionTime`. So, `transmissionTime` is now a row vector containing `totalNodes` elements, all initialized to zero.

This line of code sets up an array to keep track of the transmission time for each node in the network. Each element of the array corresponds to a specific node, and initially, all transmission times are set to zero. As the simulation progresses, this array will be updated to reflect the transmission time for each node based on the logic and conditions within the simulation loop.



```
% Comprehensive S-MAC Simulation with Visualization in MATLAB
%
% Parameters
totalNodes = 10; % Total number of nodes in the network
simulationTime = 100; % Simulation time in seconds
packetSize = 100; % Size of each data packet in bytes
trafficLoad = 0.6; % Probability of a sensing event occurring

% Initialize node states
sleepDuration = zeros(1, totalNodes); % Array to store sleep duration for each node
%array is intialized with zero
transmissionTime = zeros(1, totalNodes); % Array to store transmission time for each node

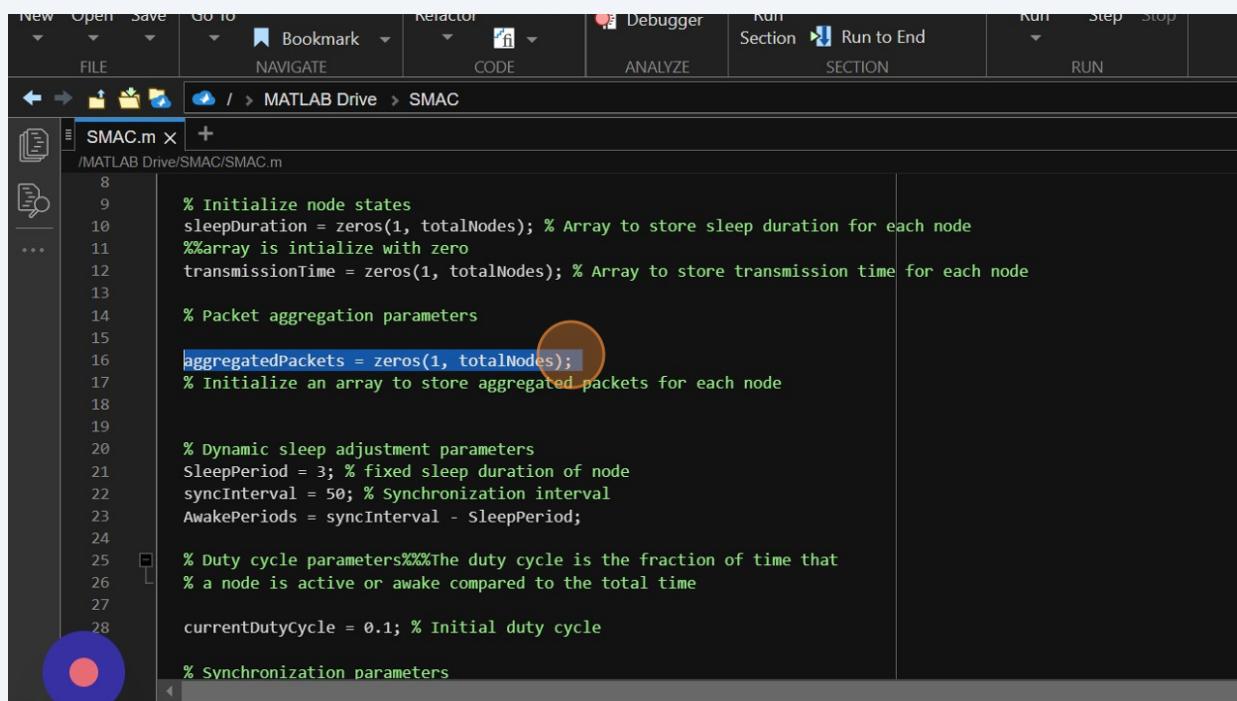
% Packet aggregation parameters

aggregatedPackets = zeros(1, totalNodes);
% Initialize an array to store aggregated packets for each node

% Dynamic sleep adjustment parameters
SleepPeriod = 3; % fixed sleep duration of node
syncInterval = 50; % Synchronization interval
AwakePeriods = syncInterval - SleepPeriod;
```

9

The line `aggregatedPackets = zeros(1, totalNodes);` initializes an array named `aggregatedPackets` with a length of `totalNodes`, where each element is initially set to zero. This array is intended to store the count of aggregated packets for each node in the network during the simulation.



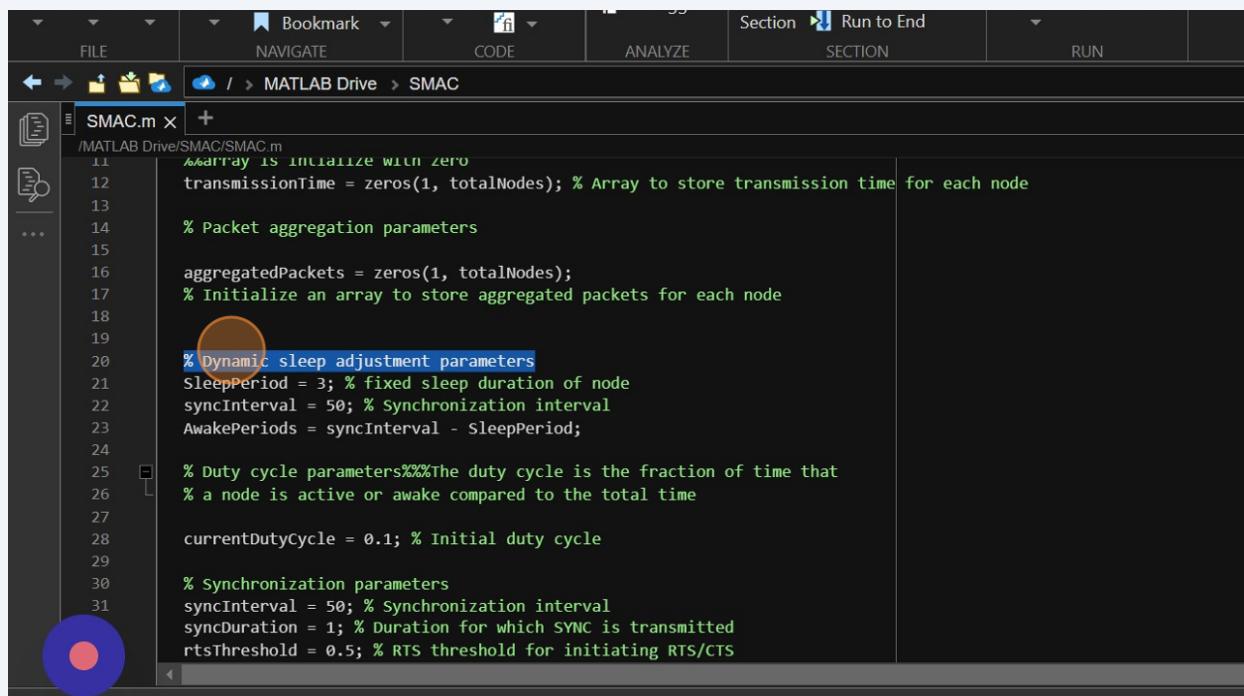
SMAC.m

```
8 % Initialize node states
9 sleepDuration = zeros(1, totalNodes); % Array to store sleep duration for each node
10 %%array is intialized with zero
11 transmissionTime = zeros(1, totalNodes); % Array to store transmission time for each node
12
13 % Packet aggregation parameters
14
15 aggregatedPackets = zeros(1, totalNodes); % Initialize an array to store aggregated packets for each node
16
17 % Dynamic sleep adjustment parameters
18 SleepPeriod = 3; % fixed sleep duration of node
19 syncInterval = 50; % Synchronization interval
20 AwakePeriods = syncInterval - SleepPeriod;
21
22 % Duty cycle parameters%%%The duty cycle is the fraction of time that
23 % a node is active or awake compared to the total time
24
25 currentDutyCycle = 0.1; % Initial duty cycle
26
27 % Synchronization parameters
```

10

These lines define dynamic sleep adjustment parameters for the S-MAC simulation. Here's a breakdown:

- `SleepPeriod = 3;:` This parameter represents the fixed sleep duration of a node. Each node will go to sleep for a duration of 3 simulation time units during its sleep period.
- `syncInterval = 50;:` This parameter sets the synchronization interval, indicating the total duration of the cycle (including both sleep and awake periods) for each node. In this case, it's set to 50 simulation time units.
- `AwakePeriods = syncInterval - SleepPeriod;:` This line calculates the awake periods within the synchronization interval. It subtracts the sleep duration (`SleepPeriod`) from the total synchronization interval (`syncInterval`), resulting in the duration for which the node remains awake. The variable `AwakePeriods` will be used in the simulation loop to adjust the duty cycle.

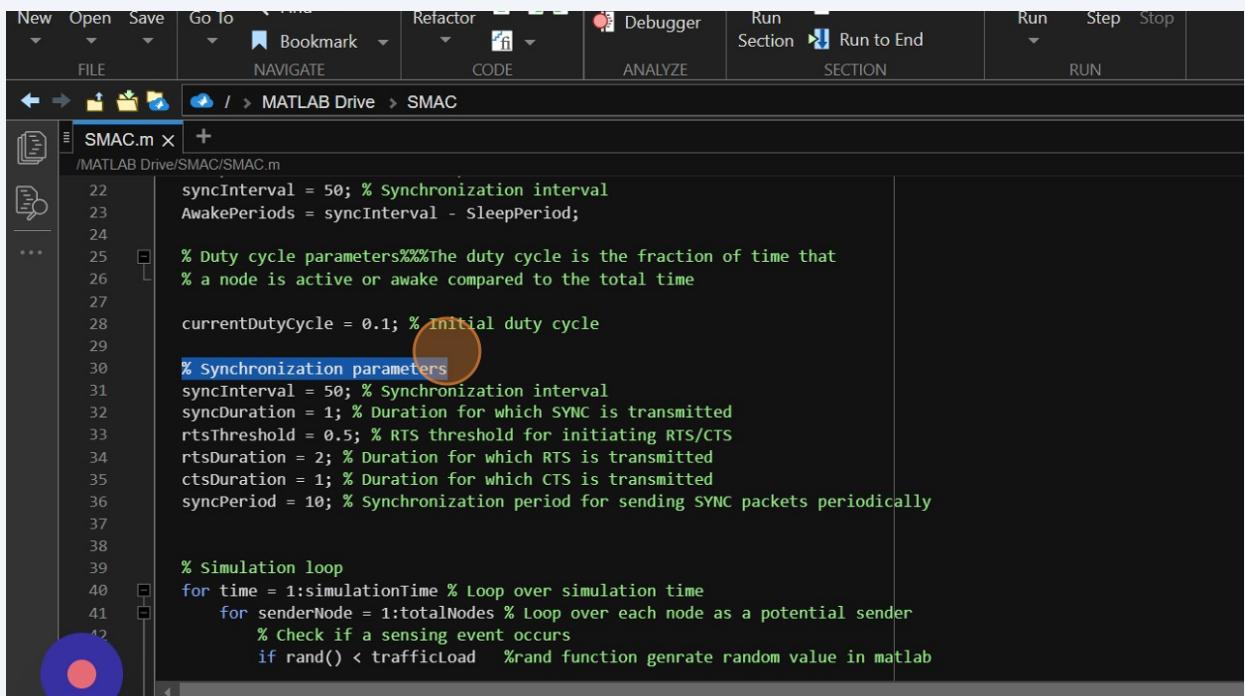


```
FILE NAVIGATE CODE ANALYZE Section Run to End SECTION RUN
MATLAB Drive / > MATLAB Drive > SMAC
SMAC.m x +
/MATLAB Drive/SMAC/SMAC.m
11 %&array is initialize with zero
12 transmissionTime = zeros(1, totalNodes); % Array to store transmission time for each node
13
14 % Packet aggregation parameters
15
16 aggregatedPackets = zeros(1, totalNodes);
17 % Initialize an array to store aggregated packets for each node
18
19
20 % Dynamic sleep adjustment parameters
21 SleepPeriod = 3; % fixed sleep duration of node
22 syncInterval = 50; % Synchronization interval
23 AwakePeriods = syncInterval - SleepPeriod;
24
25 % Duty cycle parameters%%%The duty cycle is the fraction of time that
26 % a node is active or awake compared to the total time
27
28 currentDutyCycle = 0.1; % Initial duty cycle
29
30 % Synchronization parameters
31 syncInterval = 50; % Synchronization interval
syncDuration = 1; % Duration for which SYNC is transmitted
rtsThreshold = 0.5; % RTS threshold for initiating RTS/CTS
```

11

And then we have set current Duty cycle to 0.1 from where the whole simulation starts and going to set some synchronization parameters related to the synchronization process in the S-MAC simulation:

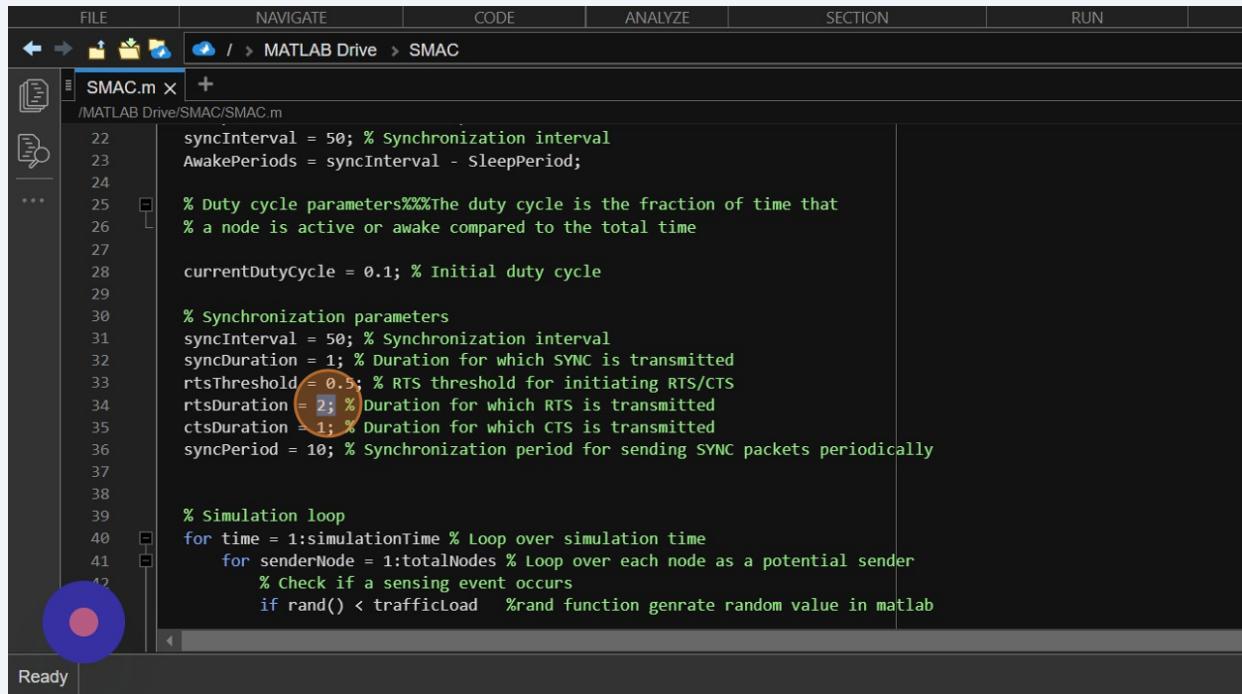
- syncInterval = 50;: This parameter sets the synchronization interval, representing the total duration of the cycle during which nodes synchronize and exchange control packets.
- syncDuration = 1;: It specifies the duration for which SYNC packets are transmitted during the synchronization phase.
- rtsThreshold = 0.5;: This parameter sets the RTS (Request to Send) threshold, representing the probability threshold for a neighboring node to initiate the RTS/CTS process. In the simulation, if a random number is less than this threshold, a neighbor initiates the process.



```
New Open Save Go To Refactor Debugger Run Step Stop
FILE NAVIGATE CODE ANALYZE SECTION RUN
< > MATLAB Drive > SMAC
SMAC.m x +
/MATLAB Drive/SMAC/SMAC.m
22 syncInterval = 50; % Synchronization interval
23 AwakePeriods = syncInterval - SleepPeriod;
24
25 % Duty cycle parameters%%%The duty cycle is the fraction of time that
26 % a node is active or awake compared to the total time
27
28 currentDutyCycle = 0.1; % Initial duty cycle
29
30 % Synchronization parameters
31 syncInterval = 50; % Synchronization interval
32 syncDuration = 1; % Duration for which SYNC is transmitted
33 rtsThreshold = 0.5; % RTS threshold for initiating RTS/CTS
34 rtsDuration = 2; % Duration for which RTS is transmitted
35 ctsDuration = 1; % Duration for which CTS is transmitted
36 syncPeriod = 10; % Synchronization period for sending SYNC packets periodically
37
38
39 % Simulation loop
40 for time = 1:simulationTime % Loop over simulation time
41     for senderNode = 1:totalNodes % Loop over each node as a potential sender
42         % Check if a sensing event occurs
        if rand() < trafficLoad %rand function generate random value in matlab
```

12

- rtsDuration = 2;: It specifies the duration for which the RTS packet is transmitted during the RTS phase.



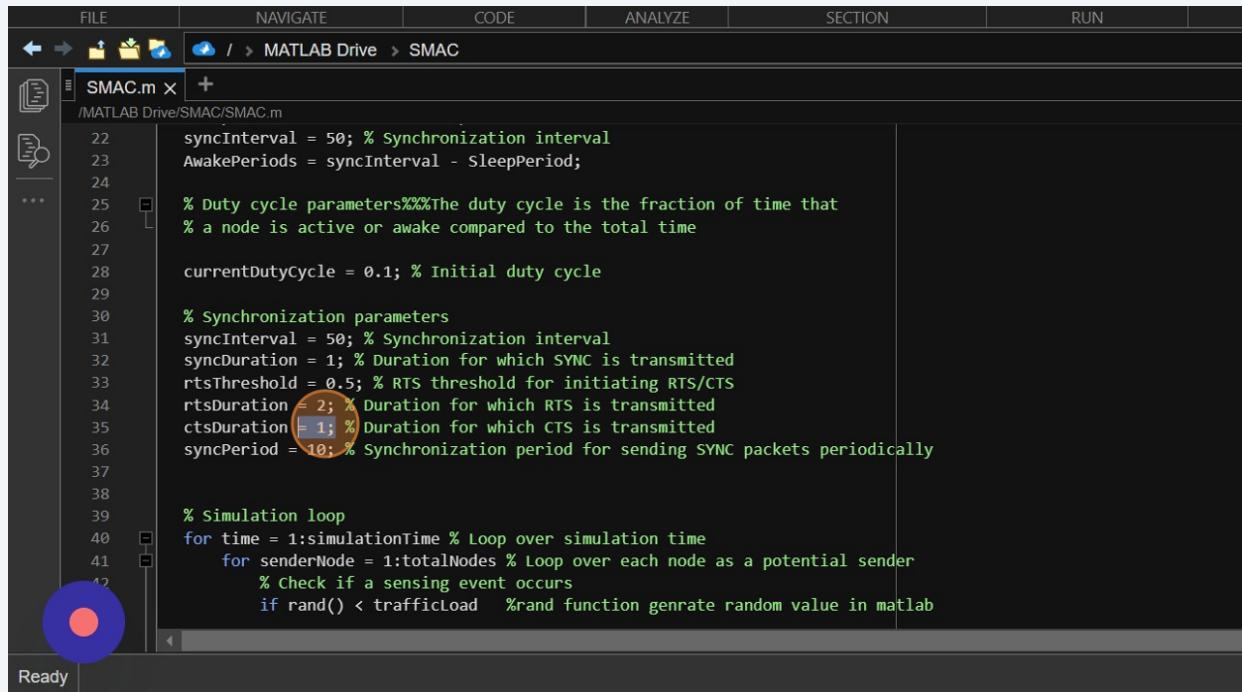
```
FILE NAVIGATE CODE ANALYZE SECTION RUN
MATLAB Drive > SMAC
SMAC.m x +
/MATLAB Drive/SMAC/SMAC.m
22 syncInterval = 50; % Synchronization interval
23 AwakePeriods = syncInterval - SleepPeriod;
24
25 % Duty cycle parameters%%%The duty cycle is the fraction of time that
26 % a node is active or awake compared to the total time
27
28 currentDutyCycle = 0.1; % Initial duty cycle
29
30 % Synchronization parameters
31 syncInterval = 50; % Synchronization interval
32 syncDuration = 1; % Duration for which SYNC is transmitted
33 rtsThreshold = 0.5; % RTS threshold for initiating RTS/CTS
34 rtsDuration = 2; % Duration for which RTS is transmitted
35 ctsDuration = 1; % Duration for which CTS is transmitted
36 syncPeriod = 10; % Synchronization period for sending SYNC packets periodically
37
38
39 % Simulation loop
40 for time = 1:simulationTime % Loop over simulation time
41     for senderNode = 1:totalNodes % Loop over each node as a potential sender
42         % Check if a sensing event occurs
         if rand() < trafficLoad    %rand function generate random value in matlab
```

13

- `ctsDuration = 1;`: It specifies the duration for which the CTS (Clear to Send) packet is transmitted in response to an RTS packet.

- `syncPeriod = 10;`: This parameter sets the synchronization period, indicating how often SYNC packets are transmitted periodically during the simulation.

These parameters play a crucial role in governing the timing and duration of various synchronization-related activities, contributing to the overall coordination and communication efficiency in the S-MAC protocol simulation



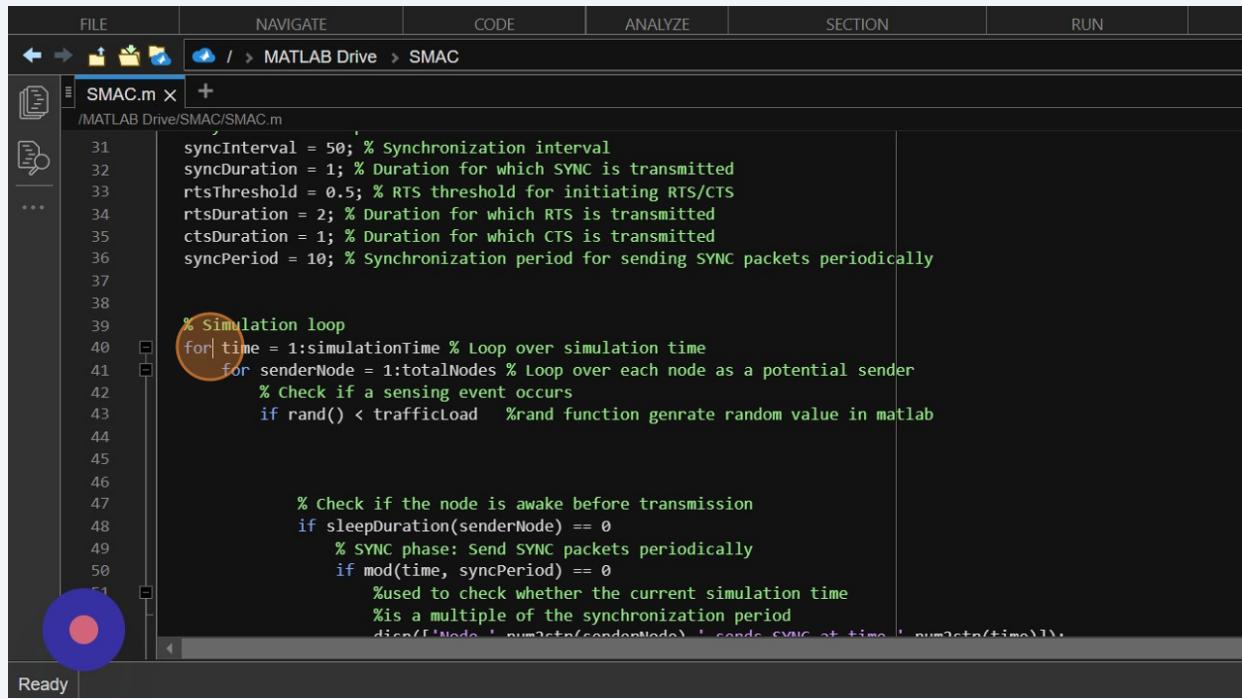
```
FILE NAVIGATE CODE ANALYZE SECTION RUN
MATLAB Drive / > MATLAB Drive > SMAC
SMAC.m X +
/MATLAB Drive/SMAC/SMAC.m
22 syncInterval = 50; % Synchronization interval
23 AwakePeriods = syncInterval - SleepPeriod;
24
25 % Duty cycle parameters%%%The duty cycle is the fraction of time that
26 % a node is active or awake compared to the total time
27
28 currentDutyCycle = 0.1; % Initial duty cycle
29
30 % Synchronization parameters
31 syncInterval = 50; % Synchronization interval
32 syncDuration = 1; % Duration for which SYNC is transmitted
33 rtsThreshold = 0.5; % RTS threshold for initiating RTS/CTS
34 rtsDuration = 2; % Duration for which RTS is transmitted
35 ctsDuration = 1; % Duration for which CTS is transmitted
36 syncPeriod = 10; % Synchronization period for sending SYNC packets periodically
37
38
39 % Simulation loop
40 for time = 1:simulationTime % Loop over simulation time
41     for senderNode = 1:totalNodes % Loop over each node as a potential sender
42         % Check if a sensing event occurs
        if rand() < trafficLoad %rand function generate random value in matlab
```

14

Now here comes the simulation Loop
Here's an overview of the key components within the loop:

- **Outer Loop** (for time = 1:simulationTime):

- The outer loop iterates over the simulation time, progressing the simulation forward in discrete time steps.

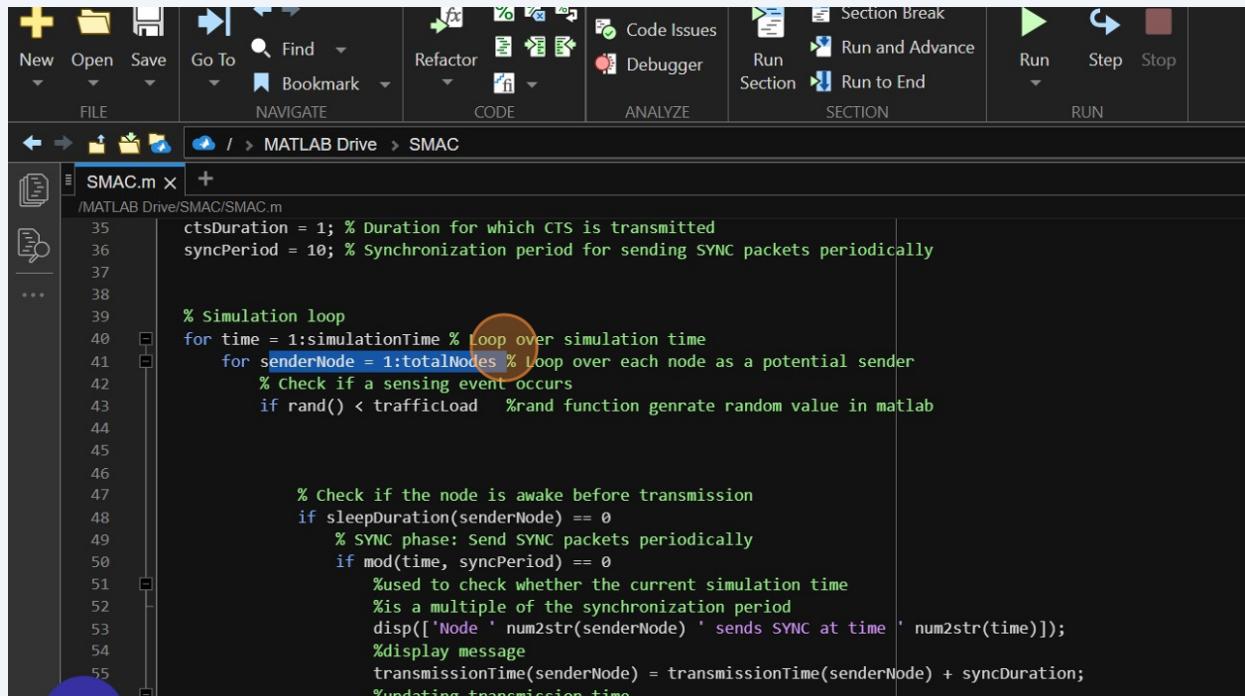


```
FILE NAVIGATE CODE ANALYZE SECTION RUN
SMAC.m x + / > MATLAB Drive > SMAC
/IMATLAB Drive/SMAC/SMAC.m
31 syncInterval = 50; % Synchronization interval
32 syncDuration = 1; % Duration for which SYNC is transmitted
33 rtsThreshold = 0.5; % RTS threshold for initiating RTS/CTS
34 rtsDuration = 2; % Duration for which RTS is transmitted
35 ctsDuration = 1; % Duration for which CTS is transmitted
36 syncPeriod = 10; % Synchronization period for sending SYNC packets periodically
37
38
39 % Simulation loop
40 for time = 1:simulationTime % Loop over simulation time
41     for senderNode = 1:totalNodes % Loop over each node as a potential sender
42         % Check if a sensing event occurs
43         if rand() < trafficLoad %rand function generate random value in matlab
44
45             % Check if the node is awake before transmission
46             if sleepDuration(senderNode) == 0
47                 % SYNC phase: Send SYNC packets periodically
48                 if mod(time, syncPeriod) == 0
49                     %used to check whether the current simulation time
50                     %is a multiple of the synchronization period
51                     disp(['Node ', num2str(senderNode), ' sends SYNC at time ', num2str(time)]);
```

15 Inner Loop (for senderNode = 1:totalNodes):

- The inner loop iterates over each node in the network, considering each node as a potential sender. **Sensing Event Check** (if rand() < trafficLoad):

- A random number is generated, and if it is less than the specified traffic load probability (trafficLoad), it is considered a sensing event. This models the probability of a node sensing an event.



```
ctsDuration = 1; % Duration for which CTS is transmitted
syncPeriod = 10; % Synchronization period for sending SYNC packets periodically

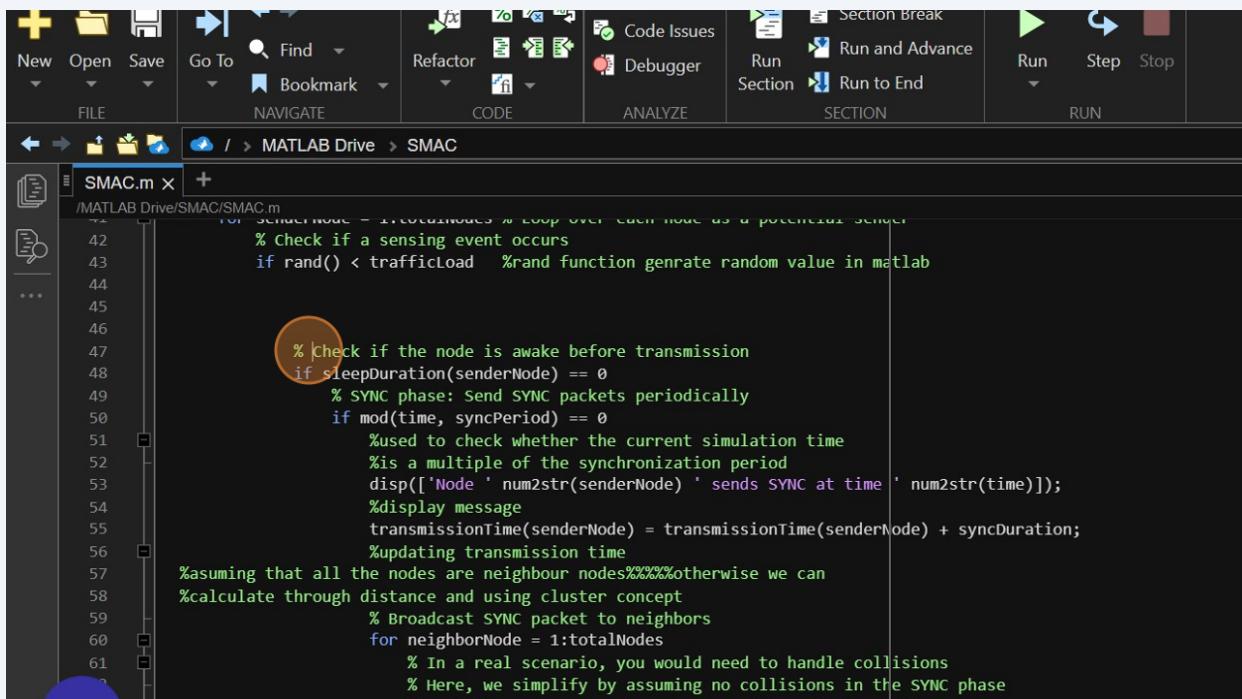
% Simulation loop
for time = 1:simulationTime % Loop over simulation time
    for senderNode = 1:totalNodes % Loop over each node as a potential sender
        % Check if a sensing event occurs
        if rand() < trafficLoad %rand function generate random value in matlab

            % Check if the node is awake before transmission
            if sleepDuration(senderNode) == 0
                % SYNC phase: Send SYNC packets periodically
                if mod(time, syncPeriod) == 0
                    %used to check whether the current simulation time
                    %is a multiple of the synchronization period
                    disp(['Node ' num2str(senderNode) ' sends SYNC at time ' num2str(time)]);
                    %display message
                    transmissionTime(senderNode) = transmissionTime(senderNode) + syncDuration;
                    %updating transmission time
                end
            end
        end
    end
end
```

16

- **Node Wakefulness Check (if sleepDuration(senderNode) == 0):**

- Checks if the node is awake (sleep duration is zero) before allowing any transmission.



The screenshot shows the MATLAB IDE interface with the SMAC.m script open. The code implements a MAC protocol, specifically SMAC, for a network of nodes. It includes a loop over each node as a potential sender, a sensing event check, and a synchronization phase where it sends SYNC packets periodically. A specific line of code, which checks if the node is awake before transmission, is highlighted with a red circle:

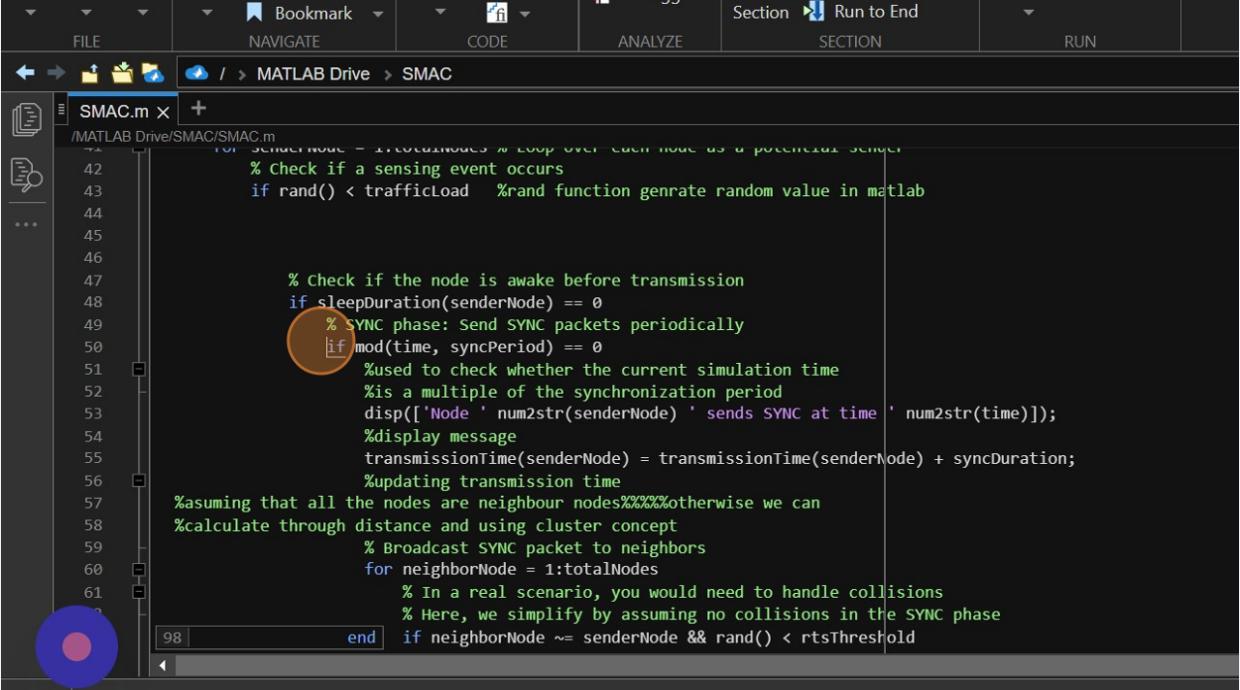
```
% Check if the node is awake before transmission
if sleepDuration(senderNode) == 0
```

17

- **SYNC Phase** (if $\text{mod}(\text{time}, \text{syncPeriod}) == 0$):

- During the SYNC phase, nodes periodically send SYNC packets. The condition checks if the current simulation time is a multiple of the synchronization period (syncPeriod).

- If true, the node broadcasts SYNC packets to neighbors, updating transmission times, and simulating synchronization.



```
FILE NAVIGATE CODE ANALYZE Section Run to End SECTION RUN
MATLAB Drive > SMAC
SMAC.m x +
/MATLAB Drive/SMAC/SMAC.m
42 % Check if a sensing event occurs
43 if rand() < trafficLoad
44
45
46
47 % Check if the node is awake before transmission
48 if sleepDuration(senderNode) == 0
49     % SYNC phase: Send SYNC packets periodically
50     if mod(time, syncPeriod) == 0
51         %used to check whether the current simulation time
52         %is a multiple of the synchronization period
53         disp(['Node ' num2str(senderNode) ' sends SYNC at time ' num2str(time)]);
54         %display message
55         transmissionTime(senderNode) = transmissionTime(senderNode) + syncDuration;
56         %updating transmission time
57 %assuming that all the nodes are neighbour nodes%%%%otherwise we can
58 %calculate through distance and using cluster concept
59         % Broadcast SYNC packet to neighbors
60         for neighborNode = 1:totalNodes
61             % In a real scenario, you would need to handle collisions
62             % Here, we simplify by assuming no collisions in the SYNC phase
63             if neighborNode ~= senderNode && rand() < rtsThreshold
64                 % Broadcast SYNC packet to neighbors
65             end
66         end
67     end
68 end
```

18

• RTS/CTS Handshake:

- If a neighbor receives SYNC and satisfies a random threshold ($\text{rand}() < \text{rtsThreshold}$), it initiates the RTS/CTS handshake:

- The neighbor initiates RTS, updating its transmission time.
- The sender node responds with CTS, updating its transmission time.
- Following this, data transmission is simulated, updating transmission times and recording the number of aggregated packets.

The screenshot shows a MATLAB interface with tabs for CODE, ANALYZE, SECTION, and RUN. The CODE tab is active, displaying the following script:

```
%% instance and using cluster concept
% Broadcast SYNC packet to neighbors
for neighborNode = 1:totalNodes
    % In a real scenario, you would need to handle collisions
    % Here, we simplify by assuming no collisions in the SYNC phase
    if neighborNode ~= senderNode && rand() < rtsThreshold

        %%node receives SYNC
        disp(['Node ' num2str(neighborNode) ' receives SYNC from Node ' num2str(senderNode)]);

        % RTS phase: Neighbor initiates RTS
        disp(['Node ' num2str(neighborNode) ' initiates RTS at time ' num2str(time)]);
        % Transmit RTS signal
        transmissionTime(neighborNode) = transmissionTime(neighborNode) + rtsDuration;

        % Neighbor responds with CTS
        disp(['Node ' num2str(senderNode) ' sends CTS at time ' num2str(time)]);

        %%updating the transmission time
    end
end
```

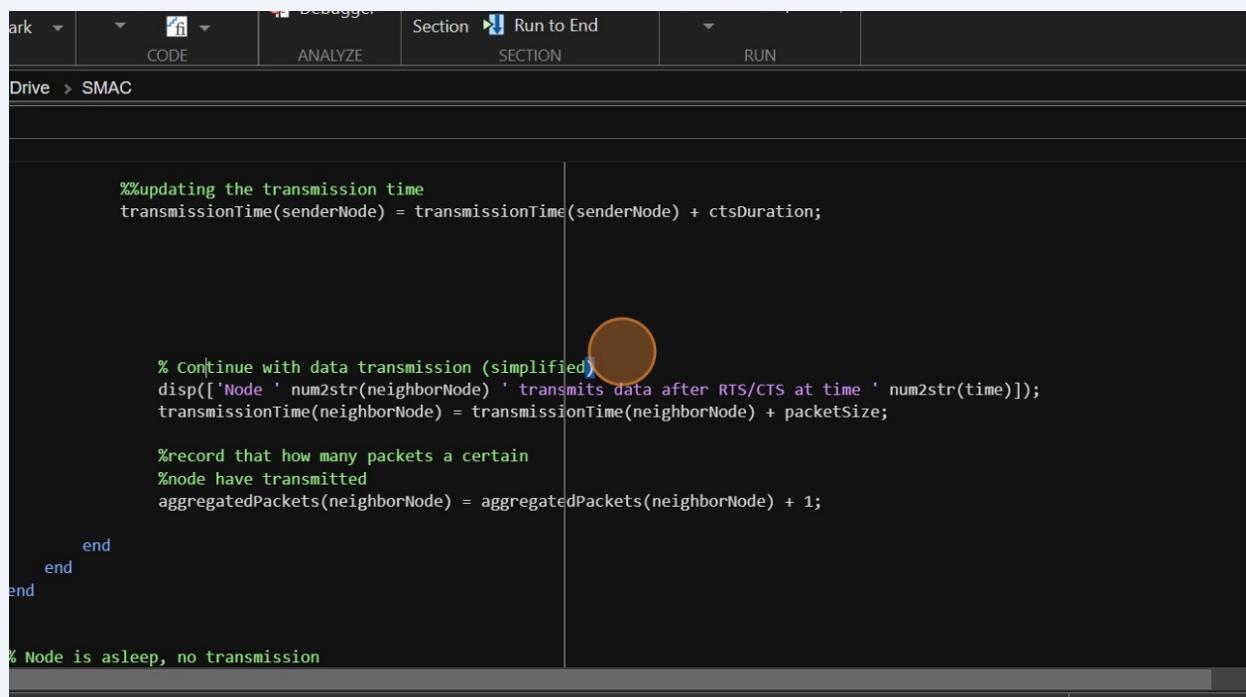
A yellow oval highlights the line `transmissionTime(neighborNode) = transmissionTime(neighborNode) + rtsDuration;`. The status bar at the bottom right shows "Zoom: 90% UTP".

19

• Sleep Duration Adjustment (else):

- If no sensing event occurs, the sleep duration of the node is increased, and the sleep duration is reset based on the duty cycle after reaching the sleep period.

This loop captures the dynamics of the S-MAC protocol, including synchronization, RTS/CTS handshakes, and sleep-wake cycles, providing insights into the behavior of nodes in a wireless sensor network over time.



Drive > SMAC

```
%%updating the transmission time
transmissionTime(senderNode) = transmissionTime(senderNode) + ctsDuration;

% Continue with data transmission (simplified)
% Node ' num2str(neighborNode) ' transmits data after RTS/CTS at time ' num2str(time) ];
transmissionTime(neighborNode) = transmissionTime(neighborNode) + packetSize;

%record that how many packets a certain
%node have transmitted
aggregatedPackets(neighborNode) = aggregatedPackets(neighborNode) + 1;

    end
end
end

% Node is asleep, no transmission
```

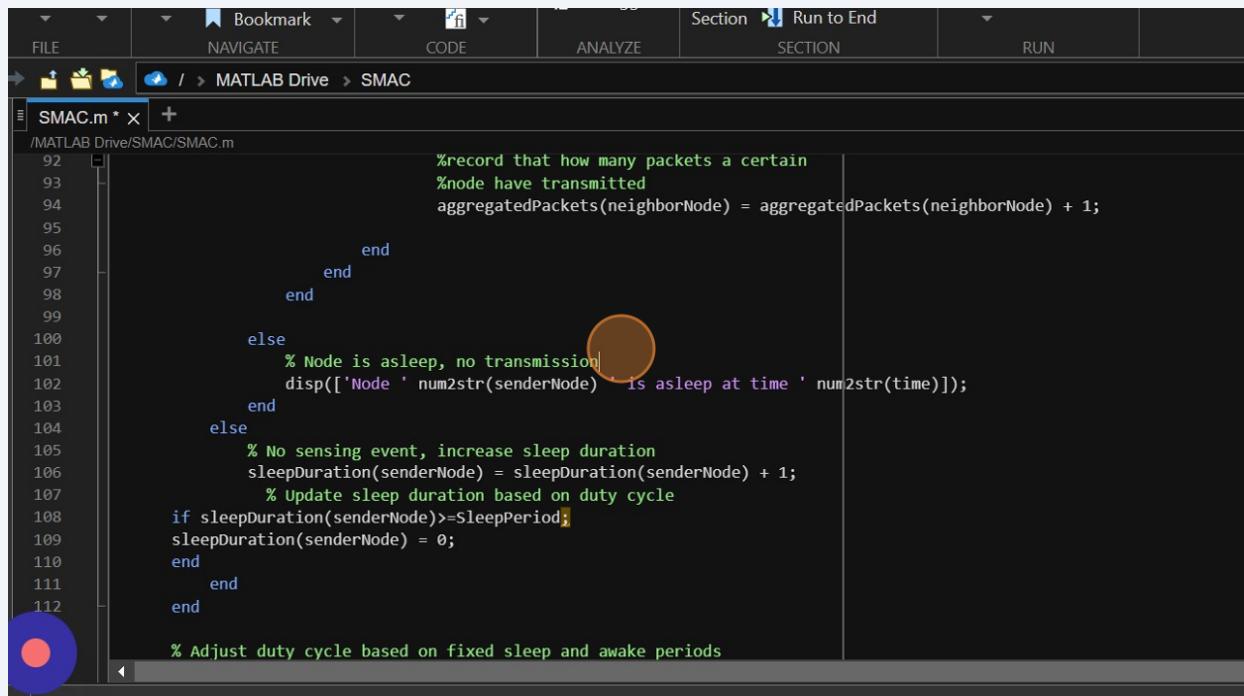
20

This part increments the count of aggregated packets for the neighbor node. It's a record-keeping mechanism to keep track of the number of packets transmitted by each node during the simulation.

```
LE NAVIGATE CODE ANALYZE SECTION RUN
MATLAB Drive / > MATLAB Drive > SMAC
SMAC.m * X +
MATLAB Drive/SMAC/SMAC.m
83
84
85
86
87
88 % Continue with data transmission (simplified)
89 disp(['Node ' num2str(neighborNode) ' transmits data after RTS/CTS at time ' num2str(time)
90 transmissionTime(neighborNode) = transmissionTime(neighborNode) + packetSize;
91
92 %record that how many packets a certain
93 %node have transmitted
94 aggregatedPackets(neighborNode) = aggregatedPackets(neighborNode) + 1;
95
96 end
97 end
98 end
99
100 else
101 % Node is asleep, no transmission
102 disp(['Node ' num2str(senderNode) ' is asleep at time ' num2str(time)]);
103 end
104 else
105 % No sensing event, increase sleep duration
```

21

If the node is determined to be asleep (sleep duration is not zero), the simulation prints a message indicating that the node is asleep at the current simulation time. In this case, no transmission occurs. If no sensing event occurs, the sleep duration of the node is increased by 1 unit. Additionally, there is a check to determine if the sleep duration has exceeded the fixed sleep period (SleepPeriod). If it has, the sleep duration is reset to 0,



The screenshot shows a MATLAB code editor window with the file 'SMAC.m' open. The code implements a sleep duration logic for a node. It includes a loop that checks for sensing events. If no event occurs, it increases the sleep duration by 1 unit. A condition checks if the sleep duration has reached or exceeded a 'SleepPeriod'. If so, it is reset to 0. The code also records transmitted packets and displays a message if a node is asleep.

```
%record that how many packets a certain
%node have transmitted
aggregatedPackets(neighborNode) = aggregatedPackets(neighborNode) + 1;

        end
    end
end

else
    % Node is asleep, no transmission
    disp(['Node ' num2str(senderNode) ' is asleep at time ' num2str(time)]);
end
else
    % No sensing event, increase sleep duration
    sleepDuration(senderNode) = sleepDuration(senderNode) + 1;
    % Update sleep duration based on duty cycle
if sleepDuration(senderNode)>=SleepPeriod;
    sleepDuration(senderNode) = 0;
end
end
end

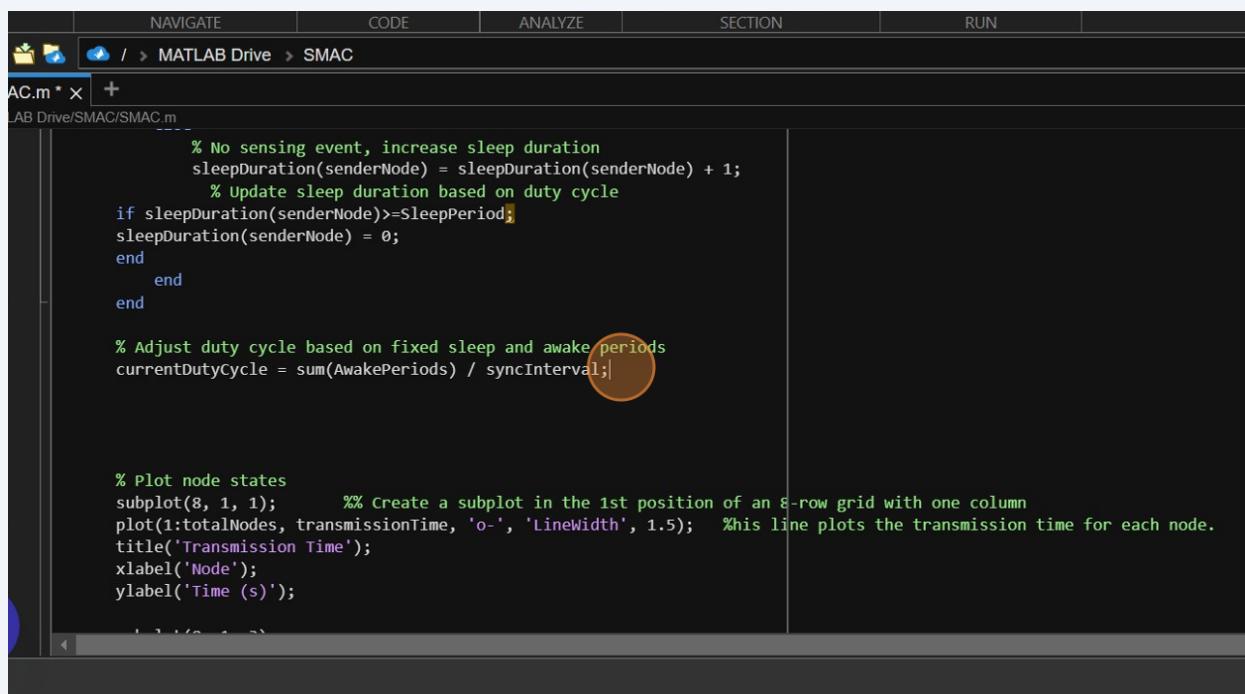
% Adjust duty cycle based on fixed sleep and awake periods
```

22

This section calculates the current duty cycle based on the fixed sleep and awake periods. Here's a breakdown:

- `sum(AwakePeriods)`: This part sums up the awake periods across all nodes. `AwakePeriods` is calculated as the difference between the synchronization interval (`syncInterval`) and the fixed sleep duration (`SleepPeriod`). It represents the total duration during which nodes are awake within a synchronization interval.
- `/ syncInterval`: The sum of awake periods is then divided by the synchronization interval to calculate the duty cycle. The duty cycle is defined as the fraction of time a node is active or awake compared to the total synchronization interval.
- `currentDutyCycle`: The result is assigned to the variable `currentDutyCycle`, representing the current duty cycle of the network.

This calculation provides an indication of how much time, on average, nodes are active compared to their total synchronization interval. It reflects the duty cycling behavior of the nodes in the network.



The screenshot shows the MATLAB IDE interface with the following details:

- Toolbar:** NAVIGATE, CODE, ANALYZE, SECTION, RUN.
- Path:** LAB Drive/SMAC/SMAC.m
- Code Editor:** The code is written in MATLAB. A specific line of code is highlighted with a red oval circle:

```
% Adjust duty cycle based on fixed sleep and awake periods
currentDutyCycle = sum(AwakePeriods) / syncInterval;
```
- Code Content:**

```
% No sensing event, increase sleep duration
sleepDuration(senderNode) = sleepDuration(senderNode) + 1;
% Update sleep duration based on duty cycle
if sleepDuration(senderNode)>=SleepPeriod;
sleepDuration(senderNode) = 0;
end
end

% Adjust duty cycle based on fixed sleep and awake periods
currentDutyCycle = sum(AwakePeriods) / syncInterval;

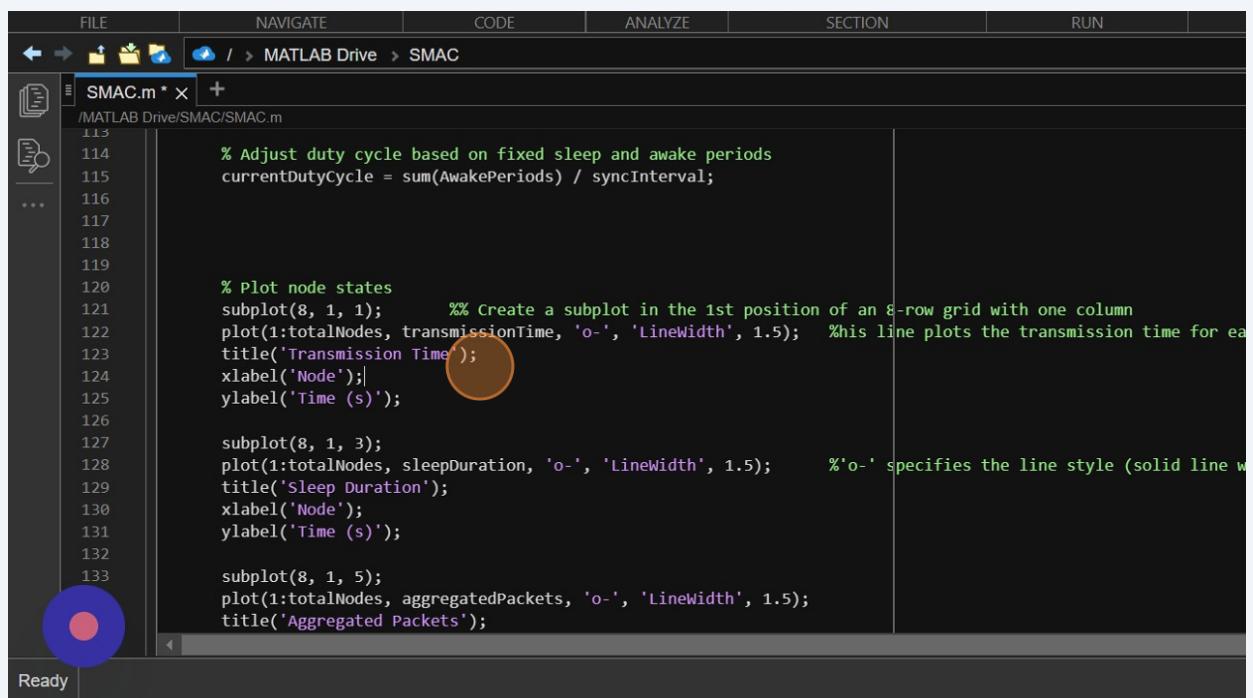
% Plot node states
subplot(8, 1, 1); % Create a subplot in the 1st position of an 8-row grid with one column
plot(1:totalNodes, transmissionTime, 'o-', 'LineWidth', 1.5); % this line plots the transmission time for each node.
title('Transmission Time');
xlabel('Node');
ylabel('Time (s)');
```

23

This code segment is responsible for creating a subplot to visualize the transmission time for each node. Here's a breakdown:

- `subplot(8, 1, 1);`: This line creates a subplot in an 8-row grid with one column and places the subsequent plot in the first position. This means the subsequent plot will be positioned at the top of the grid.
- `plot(1:totalNodes, transmissionTime, 'o-', 'LineWidth', 1.5);`: This line generates the actual plot. It plots the transmission time for each node, where the x-axis represents the node indices (1 to totalNodes), and the y-axis represents the transmission time. The data points are connected with lines (`'o-'` specifies the line style - solid line with circles at data points), and the line width is set to 1.5.
- `title('Transmission Time');`: Sets the title of the subplot to 'Transmission Time'.
- `xlabel('Node');`: Adds a label to the x-axis, indicating that it represents node indices.
- `ylabel('Time (s)');`: Adds a label to the y-axis, indicating that it represents time in seconds.

This subplot provides a visual representation of how the transmission time varies across different nodes in the network over the course of the simulation. The plot allows for an analysis of the temporal behavior of node transmissions.



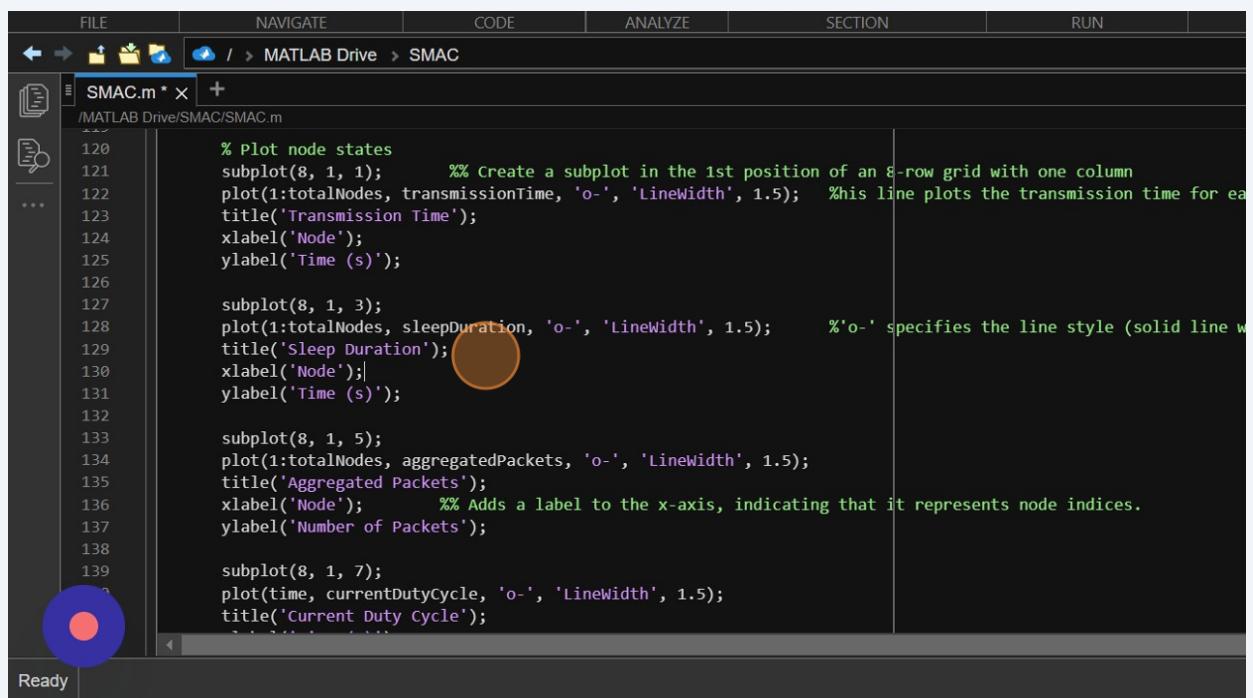
```
FILE NAVIGATE CODE ANALYZE SECTION RUN
< > MATLAB Drive > SMAC
SMAC.m * x +
MATLAB Drive/SMAC/SMAC.m
113
114 % Adjust duty cycle based on fixed sleep and awake periods
115 currentDutyCycle = sum(AwakePeriods) / syncInterval;
116
117
118
119
120 % Plot node states
121 subplot(8, 1, 1); % Create a subplot in the 1st position of an 8-row grid with one column
122 plot(1:totalNodes, transmissionTime, 'o-', 'LineWidth', 1.5); % This line plots the transmission time for each node
123 title('Transmission Time');
124 xlabel('Node');
125 ylabel('Time (s)');
126
127
128 subplot(8, 1, 3);
129 plot(1:totalNodes, sleepDuration, 'o-', 'LineWidth', 1.5); % 'o-' specifies the line style (solid line with circles)
130 title('Sleep Duration');
131 xlabel('Node');
132 ylabel('Time (s)');
133
134 subplot(8, 1, 5);
135 plot(1:totalNodes, aggregatedPackets, 'o-', 'LineWidth', 1.5);
136 title('Aggregated Packets');
```

24

This code segment creates a subplot to visualize the sleep duration for each node. Here's an explanation:

- `subplot(8, 1, 3);`: This line creates a subplot in an 8-row grid with one column and places the subsequent plot in the third position. This means the subsequent plot will be positioned in the third row of the grid.
- `plot(1:totalNodes, sleepDuration, 'o-', 'LineWidth', 1.5);`: This line generates the actual plot. It plots the sleep duration for each node over time, where the x-axis represents the node indices (1 to totalNodes), and the y-axis represents the sleep duration. The data points are connected with lines (`'o-'` specifies the line style - solid line with circles at data points), and the line width is set to 1.5.
- `title('Sleep Duration');`: Sets the title of the subplot to 'Sleep Duration'.
- `xlabel('Node');`: Adds a label to the x-axis, indicating that it represents node indices.
- `ylabel('Time (s)');`: Adds a label to the y-axis, indicating that it represents time in seconds.

This subplot provides a visual representation of how the sleep duration varies across different nodes in the network. It allows for the analysis of the temporal behavior of sleep patterns for each node during the simulation.



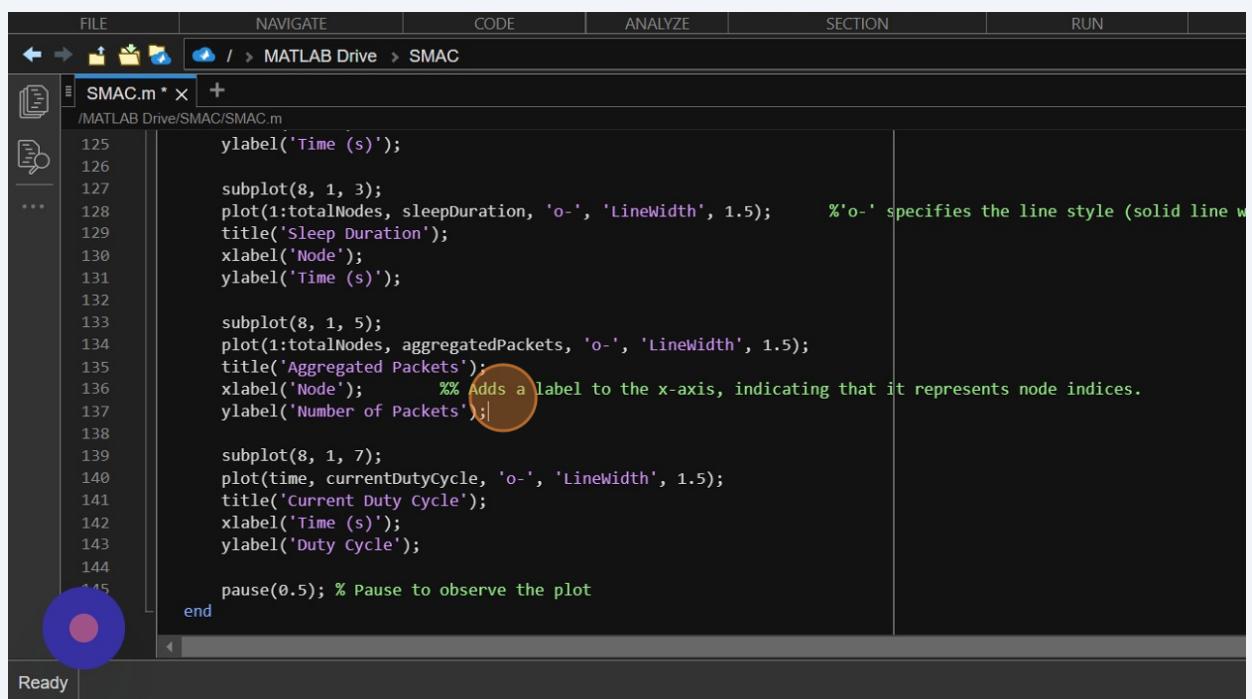
```
FILE NAVIGATE CODE ANALYZE SECTION RUN
SMAC.m * x / > MATLAB Drive > SMAC
/IMATLAB Drive/SMAC/SMAC.m
120 % Plot node states
121 subplot(8, 1, 1); % Create a subplot in the 1st position of an 8-row grid with one column
122 plot(1:totalNodes, transmissionTime, 'o-', 'LineWidth', 1.5); % This line plots the transmission time for each node
123 title('Transmission Time');
124 xlabel('Node');
125 ylabel('Time (s)');
126
127 subplot(8, 1, 3);
128 plot(1:totalNodes, sleepDuration, 'o-', 'LineWidth', 1.5); % 'o-' specifies the line style (solid line with circles)
129 title('Sleep Duration');
130 xlabel('Node');
131 ylabel('Time (s)');
132
133 subplot(8, 1, 5);
134 plot(1:totalNodes, aggregatedPackets, 'o-', 'LineWidth', 1.5);
135 title('Aggregated Packets');
136 xlabel('Node'); %% Adds a label to the x-axis, indicating that it represents node indices.
137 ylabel('Number of Packets');
138
139 subplot(8, 1, 7);
140 plot(time, currentDutyCycle, 'o-', 'LineWidth', 1.5);
141 title('Current Duty Cycle');
```

25

This code segment creates a subplot to visualize the aggregated number of packets for each node. Here's an explanation:

- `subplot(8, 1, 5);`: This line creates a subplot in an 8-row grid with one column and places the subsequent plot in the fifth position. This means the subsequent plot will be positioned in the fifth row of the grid.
- `plot(1:totalNodes, aggregatedPackets, 'o-', 'LineWidth', 1.5);`: This line generates the actual plot. It plots the aggregated number of packets for each node, where the x-axis represents the node indices (1 to totalNodes), and the y-axis represents the aggregated number of packets. The data points are connected with lines ('o-' specifies the line style - solid line with circles at data points), and the line width is set to 1.5.
- `title('Aggregated Packets');`: Sets the title of the subplot to 'Aggregated Packets'.
- `xlabel('Node');`: Adds a label to the x-axis, indicating that it represents node indices.
- `ylabel('Number of Packets');`: Adds a label to the y-axis, indicating that it represents the number of aggregated packets.

This subplot provides a visual representation of how the number of aggregated packets varies across different nodes in the network over the course of the simulation. It allows for the analysis of the data aggregation behavior for each node.

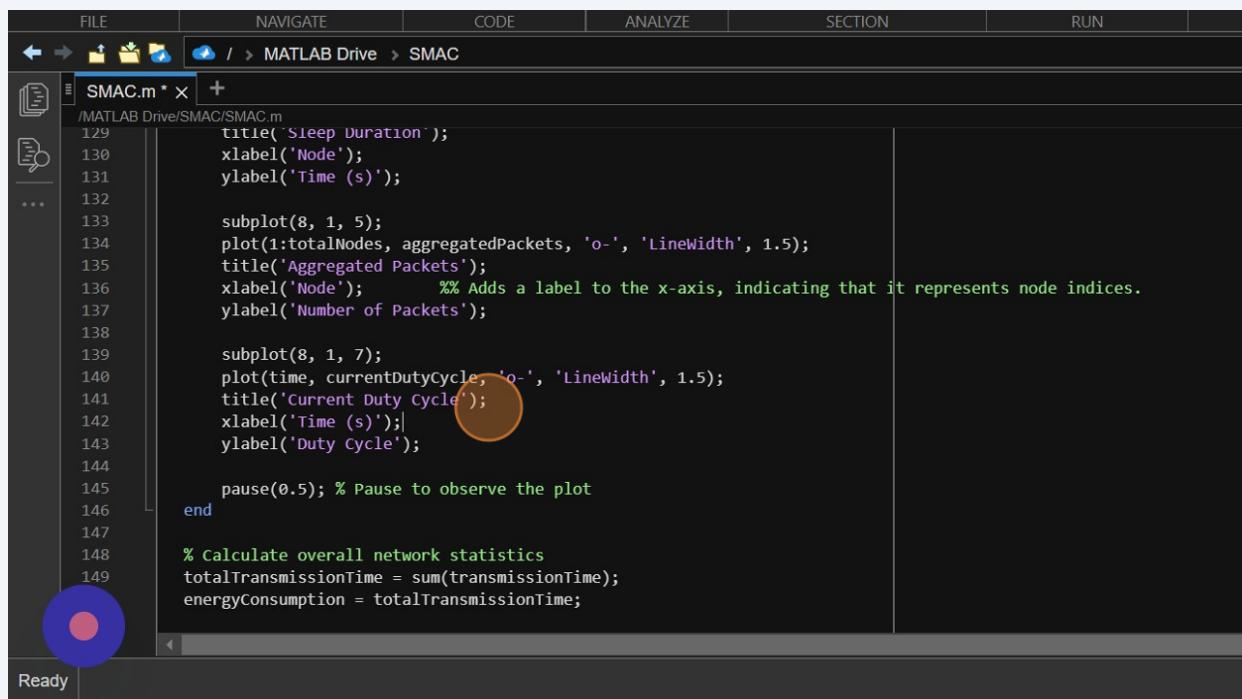


```
FILE NAVIGATE CODE ANALYZE SECTION RUN
MATLAB Drive / > MATLAB Drive > SMAC
SMAC.m * x + /MATLAB Drive/SMAC/SMAC.m
125     ylabel('Time (s)');
126
127
128 subplot(8, 1, 3);
129 plot(1:totalNodes, sleepDuration, 'o-', 'LineWidth', 1.5); %'o-' specifies the line style (solid line w
130 title('Sleep Duration');
131 xlabel('Node');
132 ylabel('Time (s)');
133
134 subplot(8, 1, 5);
135 plot(1:totalNodes, aggregatedPackets, 'o-', 'LineWidth', 1.5);
136 title('Aggregated Packets'); % Add a label to the x-axis, indicating that it represents node indices.
137 xlabel('Node');
138 ylabel('Number of Packets');
139
140 subplot(8, 1, 7);
141 plot(time, currentDutyCycle, 'o-', 'LineWidth', 1.5);
142 title('Current Duty Cycle');
143 xlabel('Time (s)');
144 ylabel('Duty Cycle');
145
146 pause(0.5); % Pause to observe the plot
end
```

26

This code segment creates a subplot to visualize the current duty cycle of the network over time. Here's an explanation:

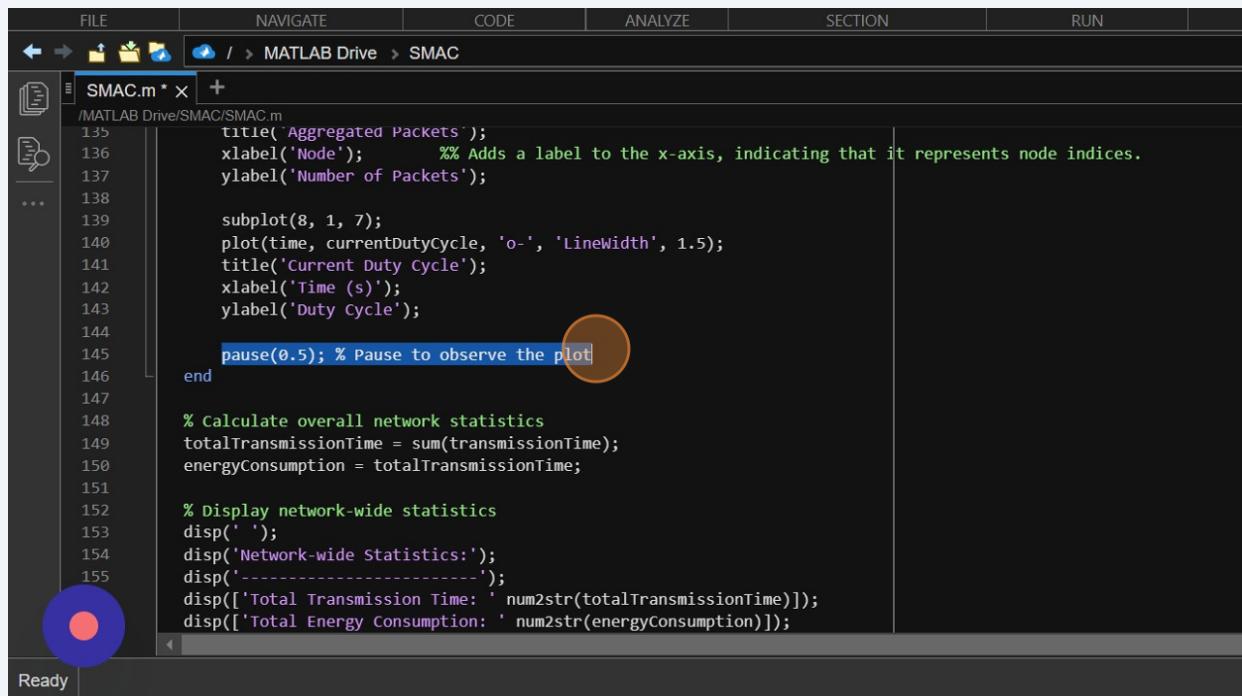
- `subplot(8, 1, 7);`: This line creates a subplot in an 8-row grid with one column and places the subsequent plot in the seventh position. This means the subsequent plot will be positioned in the seventh row of the grid.
- `plot(time, currentDutyCycle, 'o-', 'LineWidth', 1.5);`: This line generates the actual plot. It plots the duty cycle of the network over time, where the x-axis represents time (time), and the y-axis represents the duty cycle (currentDutyCycle). The data points are connected with lines ('o-' specifies the line style - solid line with circles at data points), and the line width is set to 1.5.
- `title('Current Duty Cycle');`: Sets the title of the subplot to 'Current Duty Cycle'.
- `xlabel('Time (s)');`: Adds a label to the x-axis, indicating that it represents time in seconds.
- `ylabel('Duty Cycle');`: Adds a label to the y-axis, indicating that it represents the duty cycle.



```
FILE NAVIGATE CODE ANALYZE SECTION RUN
MATLAB Drive / > MATLAB Drive > SMAC
SMAC.m * x +
/MATLAB Drive/SMAC/SMAC.m
129     title('Sleep Duration');
130     xlabel('Node');
131     ylabel('Time (s)');
132
133     subplot(8, 1, 5);
134     plot(1:totalNodes, aggregatedPackets, 'o-', 'LineWidth', 1.5);
135     title('Aggregated Packets');
136     xlabel('Node');      %% Adds a label to the x-axis, indicating that it represents node indices.
137     ylabel('Number of Packets');
138
139     subplot(8, 1, 7);
140     plot(time, currentDutyCycle, 'o-', 'LineWidth', 1.5);
141     title('Current Duty Cycle');
142     xlabel('Time (s)');
143     ylabel('Duty Cycle');
144
145     pause(0.5); % Pause to observe the plot
146
147
148
149 % calculate overall network statistics
totalTransmissionTime = sum(transmissionTime);
energyConsumption = totalTransmissionTime;
```

27

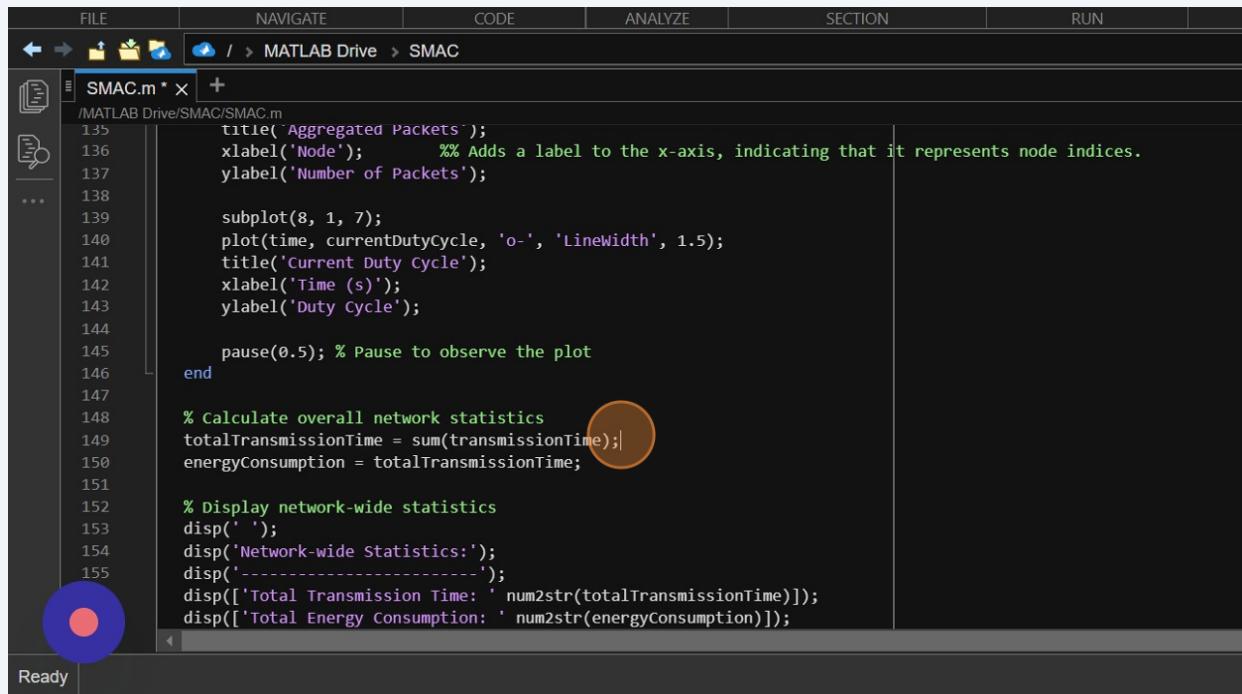
Pauses the execution for 0.5 seconds, allowing time to observe the plot.



```
FILE NAVIGATE CODE ANALYZE SECTION RUN
MATLAB Drive / > MATLAB Drive > SMAC
SMAC.m * x +
/MATLAB Drive/SMAC/SMAC.m
135 title('Aggregated Packets');
136 xlabel('Node'); % Adds a label to the x-axis, indicating that it represents node indices.
137 ylabel('Number of Packets');
...
138
139 subplot(8, 1, 7);
140 plot(time, currentDutyCycle, 'o-', 'LineWidth', 1.5);
141 title('Current Duty Cycle');
142 xlabel('Time (s)');
143 ylabel('Duty Cycle');
144
145 pause(0.5); % Pause to observe the plot
146 end
147
148 % Calculate overall network statistics
149 totalTransmissionTime = sum(transmissionTime);
150 energyConsumption = totalTransmissionTime;
151
152 % Display network-wide statistics
153 disp('');
154 disp('Network-wide Statistics:');
155 disp('-----');
156 disp(['Total Transmission Time: ' num2str(totalTransmissionTime)]);
157 disp(['Total Energy Consumption: ' num2str(energyConsumption)]);
```

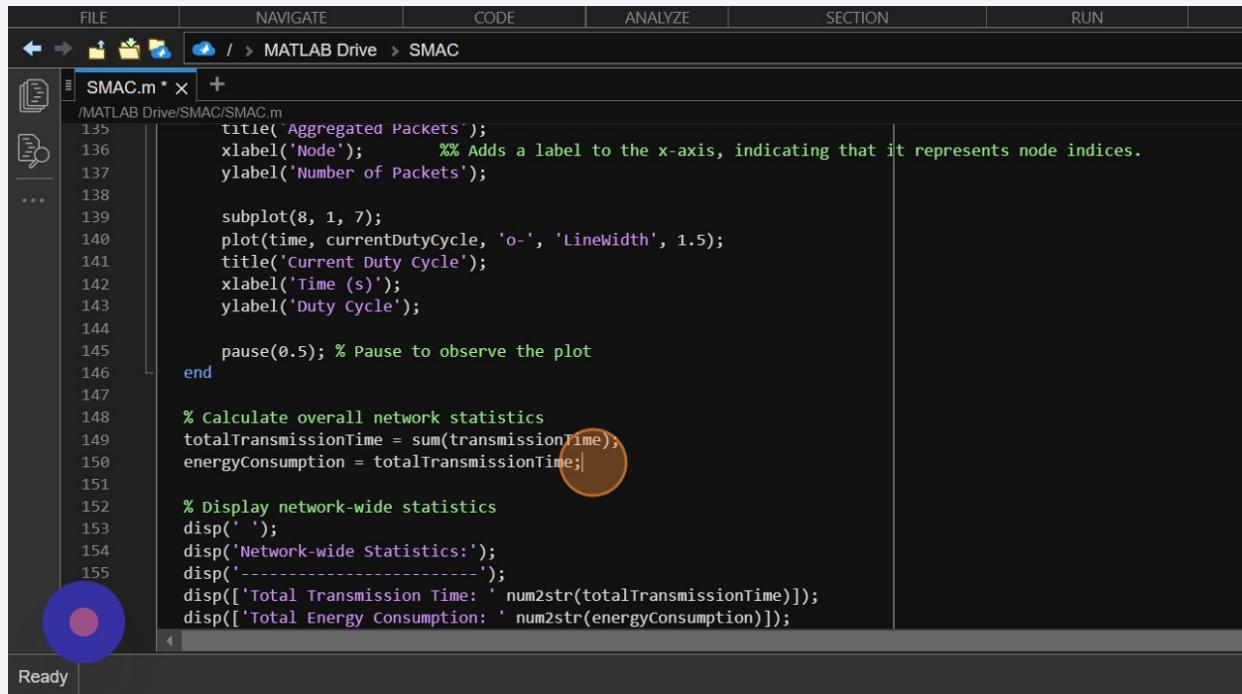
28

The total transmission time here is calculated by the sum of all the transmission Time involved in the simulation



```
FILE NAVIGATE CODE ANALYZE SECTION RUN
MATLAB Drive / > MATLAB Drive > SMAC
SMAC.m * x +
/MATLAB Drive/SMAC/SMAC.m
135 title('Aggregated Packets');
136 xlabel('Node'); % Adds a label to the x-axis, indicating that it represents node indices.
137 ylabel('Number of Packets');
...
138
139 subplot(8, 1, 7);
140 plot(time, currentDutyCycle, 'o-', 'LineWidth', 1.5);
141 title('Current Duty Cycle');
142 xlabel('Time (s)');
143 ylabel('Duty Cycle');
144
145 pause(0.5); % Pause to observe the plot
146 end
147
148 % Calculate overall network statistics
149 totalTransmissionTime = sum(transmissionTime); |-----|
150 energyConsumption = totalTransmissionTime;
151
152 % Display network-wide statistics
153 disp('');
154 disp('Network-wide Statistics:');
155 disp('-----');
156 disp(['Total Transmission Time: ' num2str(totalTransmissionTime)]);
157 disp(['Total Energy Consumption: ' num2str(energyConsumption)]);
```

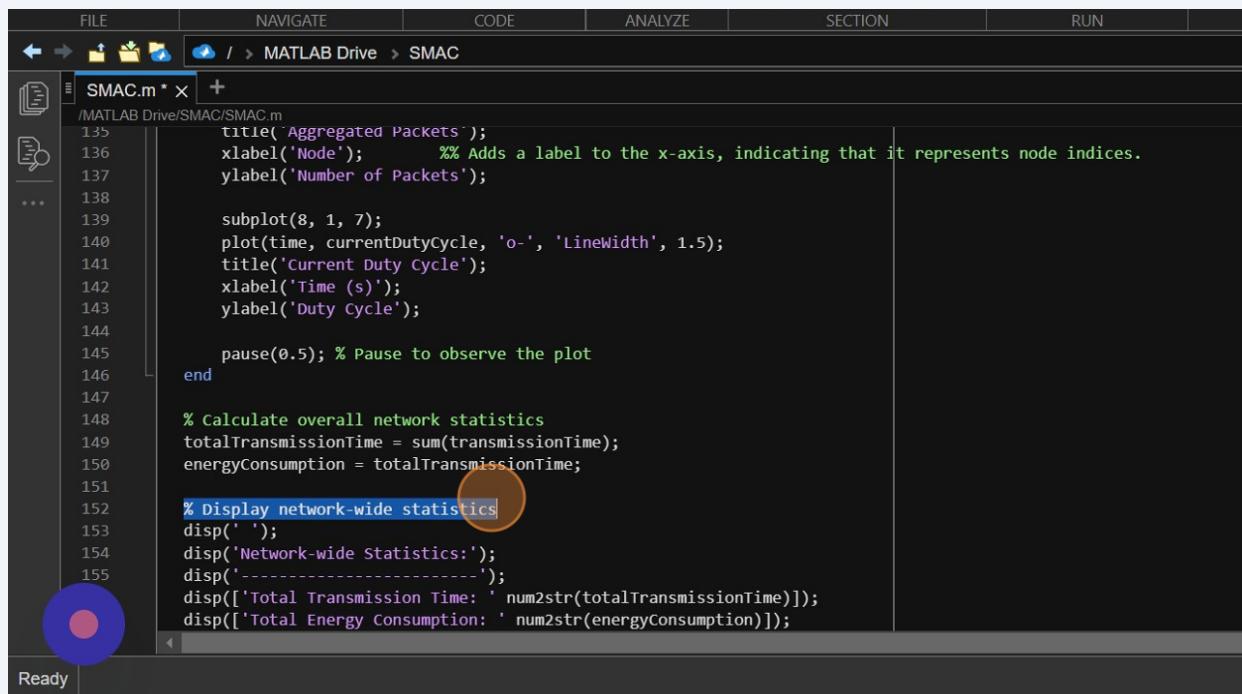
29 Then energy consumption will be obtained by the totalTransmissionTime



```
FILE NAVIGATE CODE ANALYZE SECTION RUN
MATLAB Drive / > MATLAB Drive > SMAC
SMAC.m * X +
/MATLAB Drive/SMAC/SMAC.m
135 title('Aggregated Packets');
136 xlabel('Node'); %% Adds a label to the x-axis, indicating that it represents node indices.
137 ylabel('Number of Packets');
138
139 subplot(8, 1, 7);
140 plot(time, currentDutyCycle, 'o-', 'LineWidth', 1.5);
141 title('Current Duty Cycle');
142 xlabel('Time (s)');
143 ylabel('Duty Cycle');
144
145 pause(0.5); % Pause to observe the plot
end
147
148 % Calculate overall network statistics
149 totalTransmissionTime = sum(transmissionTime);
150 energyConsumption = totalTransmissionTime; %
151
152 % Display network-wide statistics
153 disp(' ');
154 disp('Network-wide Statistics:');
155 disp('-----');
156 disp(['Total Transmission Time: ' num2str(totalTransmissionTime)]);
157 disp(['Total Energy Consumption: ' num2str(energyConsumption)]);

Ready
```

30 Now here is the code to display the overall statistics at the end of simulation

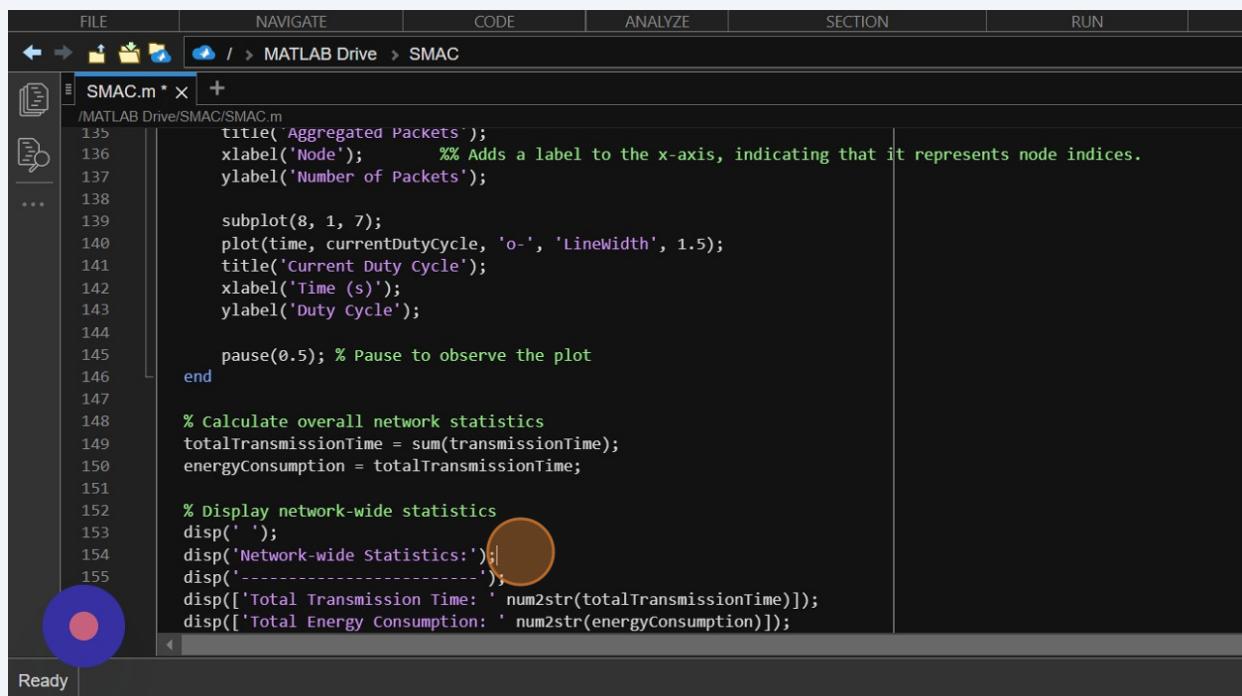


```
FILE NAVIGATE CODE ANALYZE SECTION RUN
MATLAB Drive / > MATLAB Drive > SMAC
SMAC.m * X +
/MATLAB Drive/SMAC/SMAC.m
135 title('Aggregated Packets');
136 xlabel('Node'); %% Adds a label to the x-axis, indicating that it represents node indices.
137 ylabel('Number of Packets');
138
139 subplot(8, 1, 7);
140 plot(time, currentDutyCycle, 'o-', 'LineWidth', 1.5);
141 title('Current Duty Cycle');
142 xlabel('Time (s)');
143 ylabel('Duty Cycle');
144
145 pause(0.5); % Pause to observe the plot
end
147
148 % Calculate overall network statistics
149 totalTransmissionTime = sum(transmissionTime);
150 energyConsumption = totalTransmissionTime;
151
152 % Display network-wide statistics %
153 disp(' ');
154 disp('Network-wide Statistics:');
155 disp('-----');
156 disp(['Total Transmission Time: ' num2str(totalTransmissionTime)]);
157 disp(['Total Energy Consumption: ' num2str(energyConsumption)]);

Ready
```

31

This is the header before printing the stats after that the two values previously calculated in 149 and 150 number line is displayed

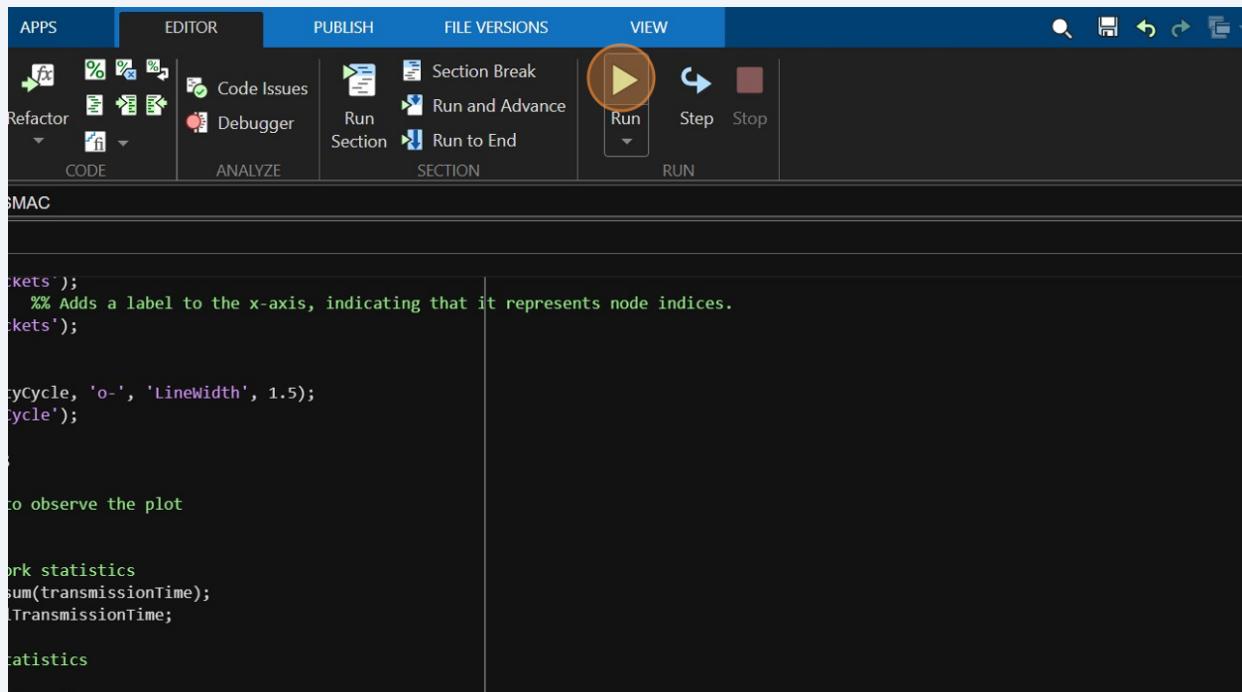


```
FILE NAVIGATE CODE ANALYZE SECTION RUN
< > / > MATLAB Drive > SMAC
SMAC.m * x +
/MATLAB Drive/SMAC/SMAC.m
135 title('Aggregated Packets');
136 xlabel('Node'); % Adds a label to the x-axis, indicating that it represents node indices.
137 ylabel('Number of Packets');
138
139 subplot(8, 1, 7);
140 plot(time, currentDutyCycle, 'o-', 'LineWidth', 1.5);
141 title('Current Duty Cycle');
142 xlabel('Time (s)');
143 ylabel('Duty Cycle');
144
145 pause(0.5); % Pause to observe the plot
end
148 % Calculate overall network statistics
149 totalTransmissionTime = sum(transmissionTime);
150 energyConsumption = totalTransmissionTime;
151
152 % Display network-wide statistics
153 disp(' ');
154 disp('Network-wide Statistics:');
155 disp('-----');
156 disp(['Total Transmission Time: ' num2str(totalTransmissionTime)]);
157 disp(['Total Energy Consumption: ' num2str(energyConsumption)]);

Ready
```

32

Now lets run it to see the simulation and output



APPS EDITOR PUBLISH FILE VERSIONS VIEW

Refactor Code Issues Section Break Run and Advance Run Step Stop

CODE Debugger Run Section Run to End SECTION

RUN

```
SMAC

    %% Adds a label to the x-axis, indicating that it represents node indices.

    %cycle, 'o-', 'LineWidth', 1.5);
    %cycle');

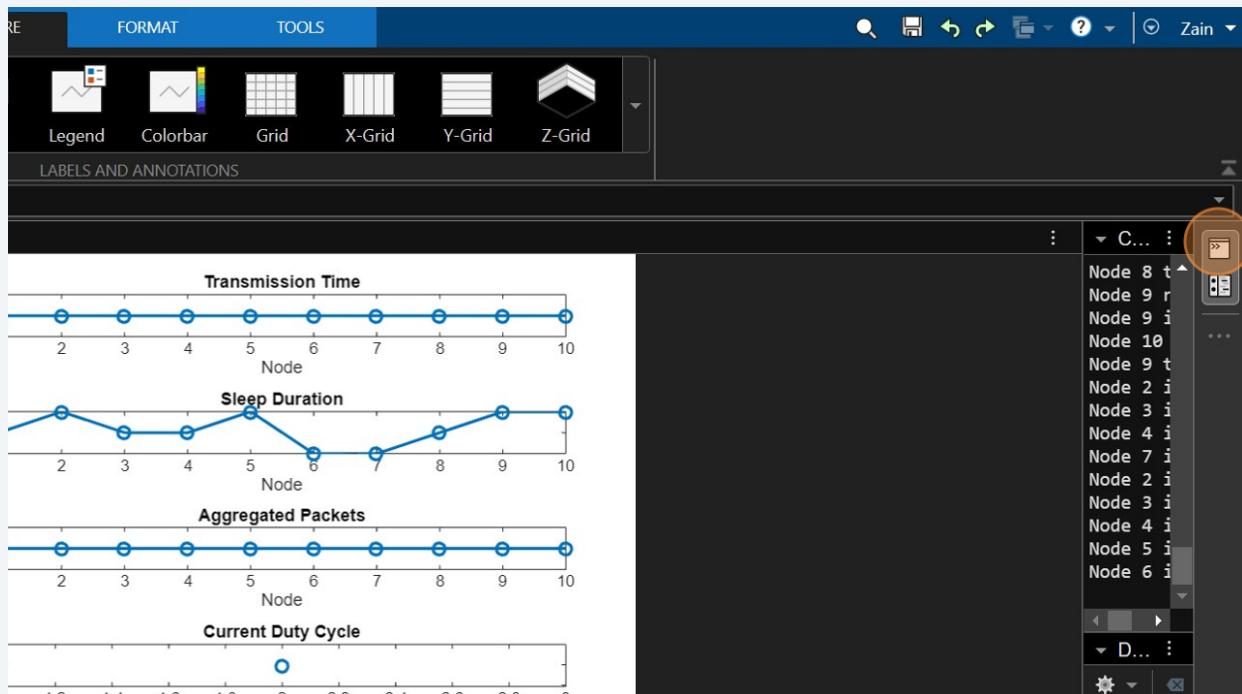
    % to observe the plot

    %ork statistics
    sum(transmissionTime);
    %TransmissionTime;

    %statistics
```

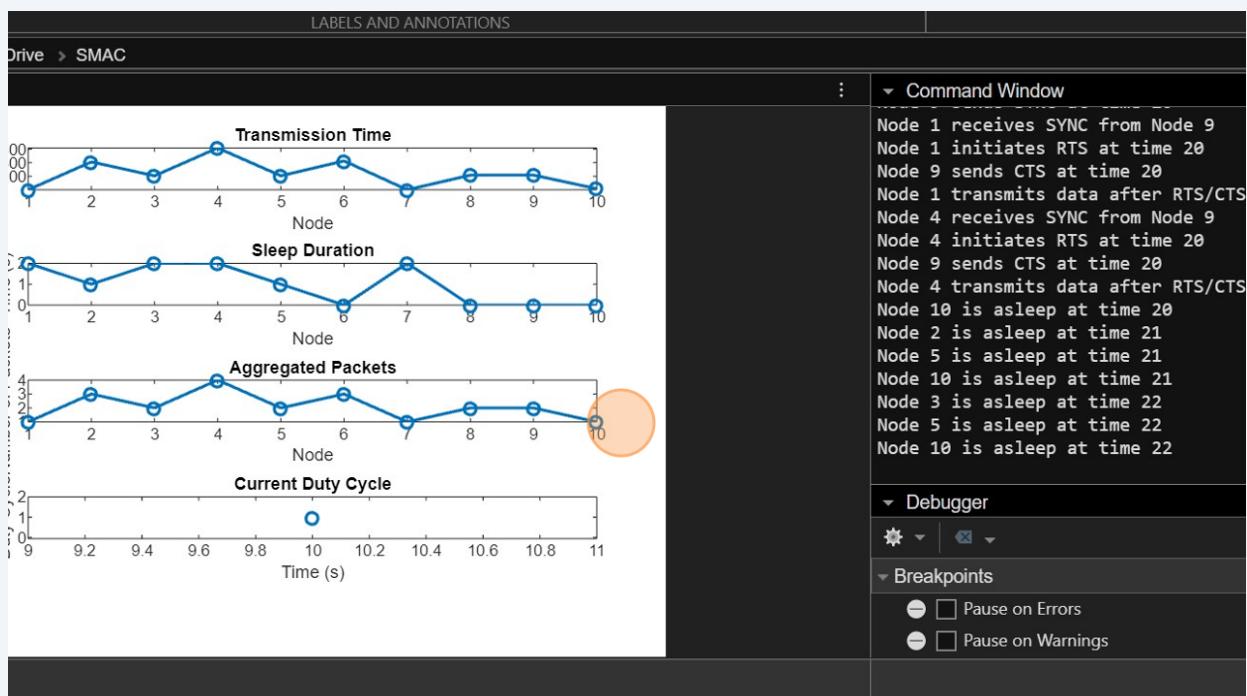
33

A figure will be shown after running the program
Let's see the commands displayed in the terminal that was previously discussed in the code by using disp commands



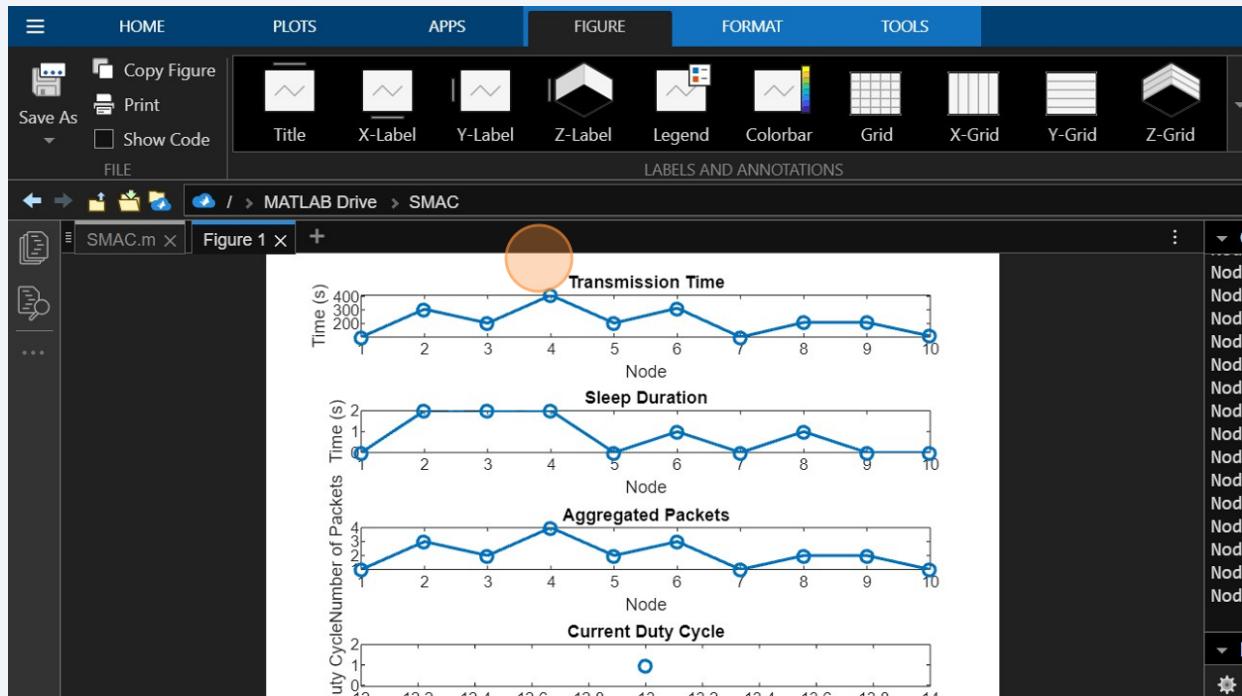
34

Here figure and command window both are shown



35

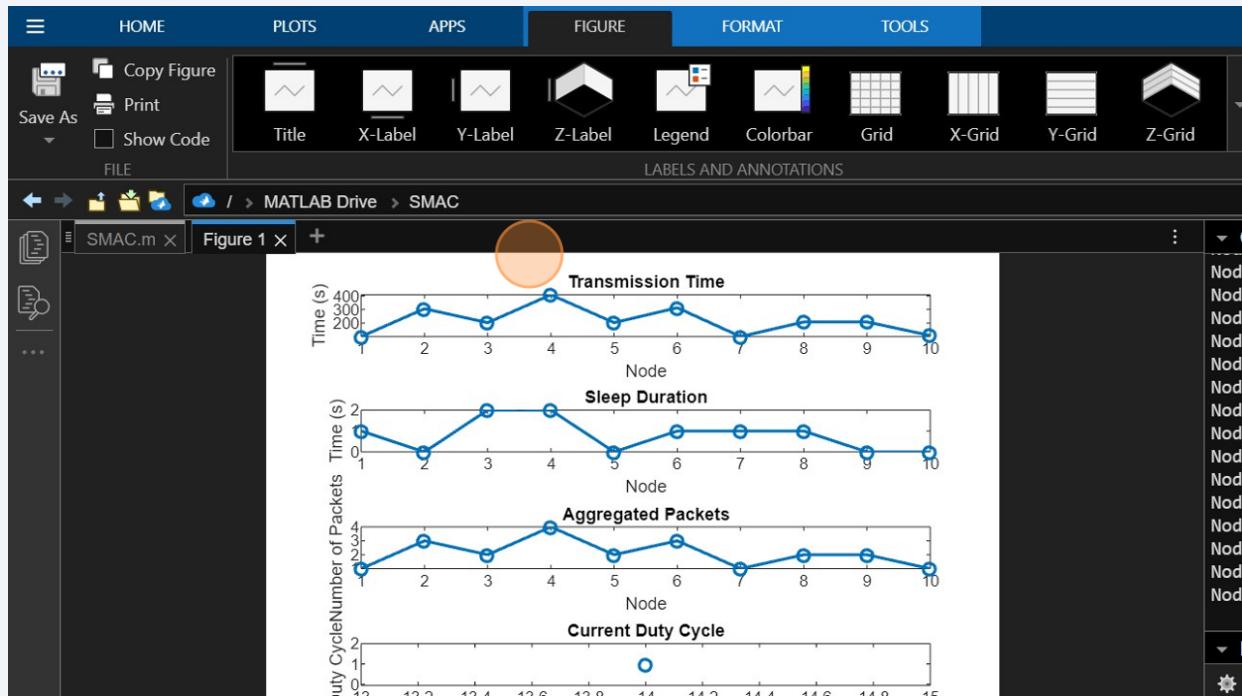
On the top of figure we plotted the first subplot as Transmission time and title was set to "Transmission Time"



36

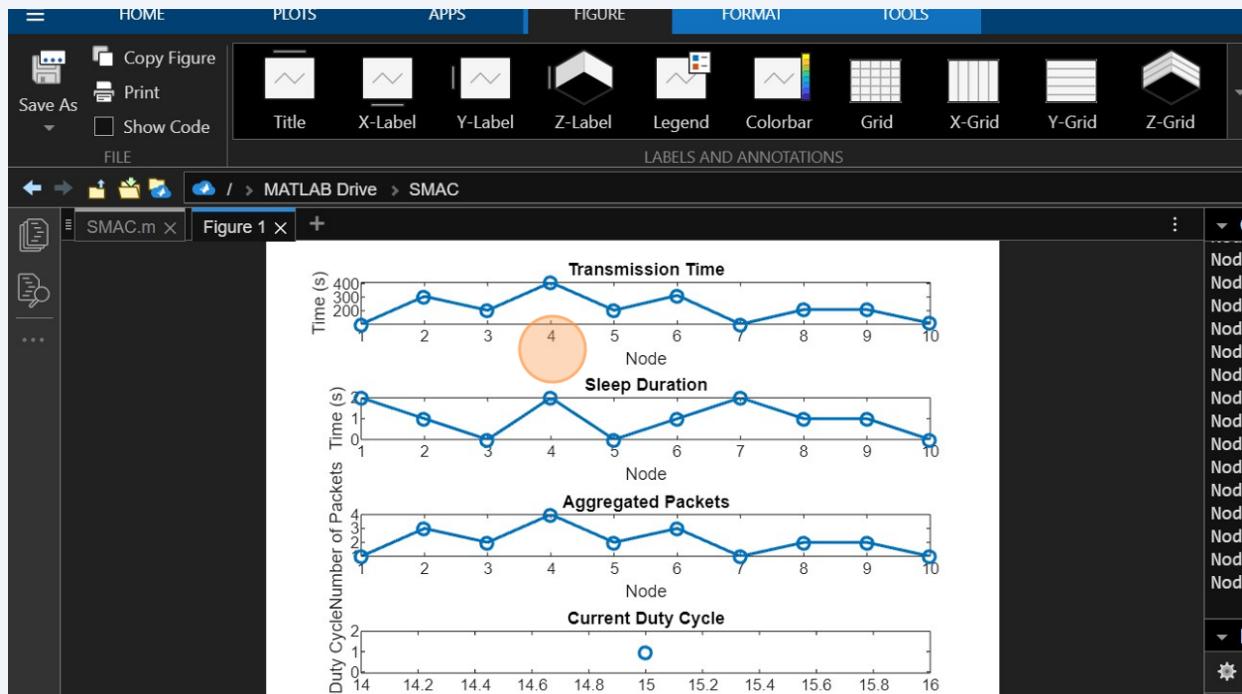
In the transmission time graph (subplot) on y axis there is time displayed in seconds and in this figure circles are showing the nodes In this simulation there are 10 nodes

We can take it on random positions as well



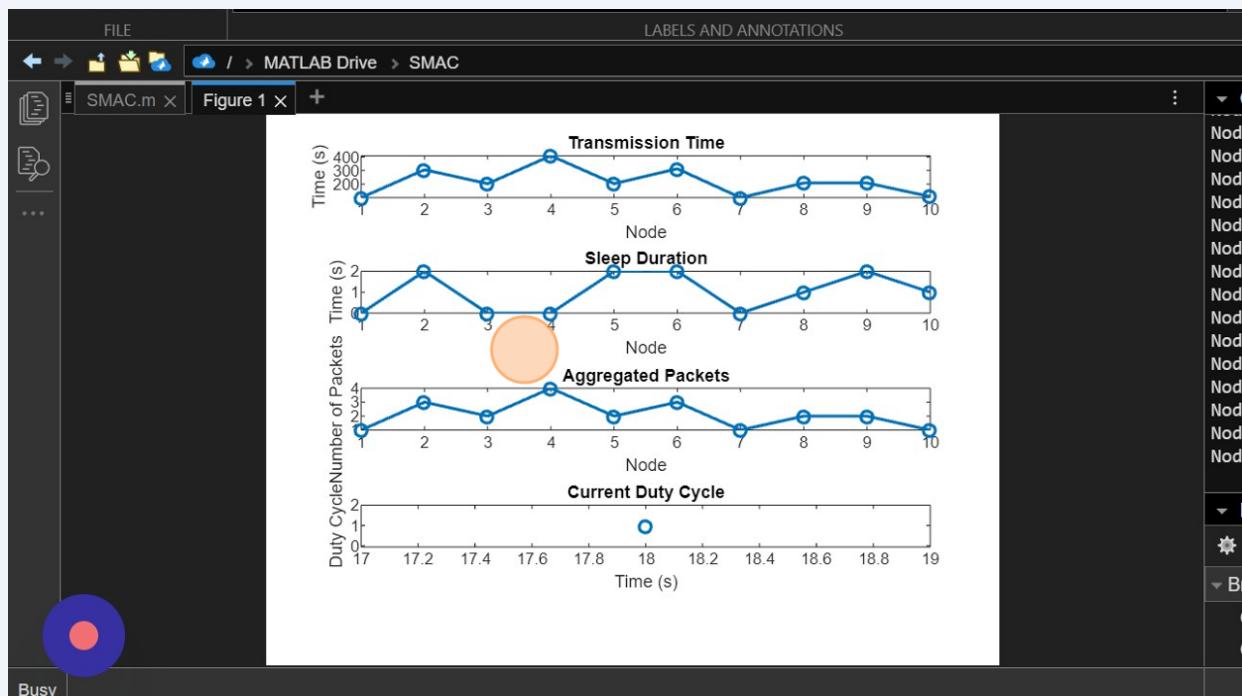
37

Next thing is sleep duration that is displayed here in the Subplot titled Sleep Duration and showing the sleep duration of each node in seconds that how much time the node is sleeping



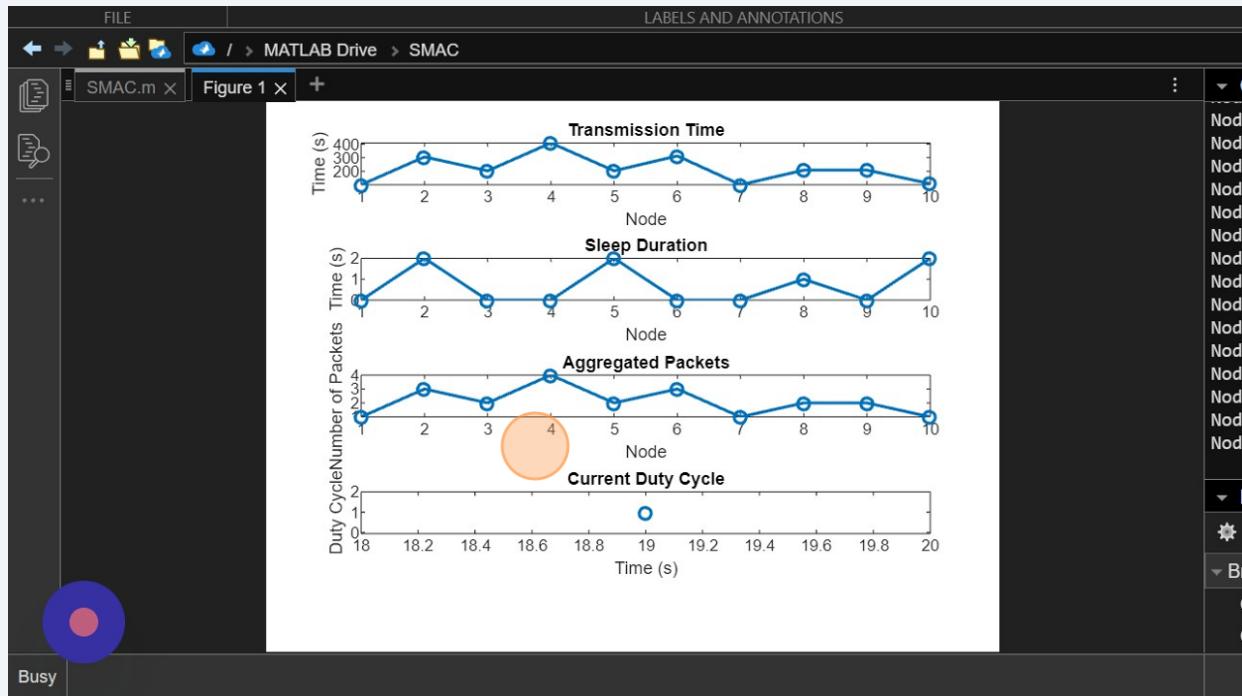
38

Then here comes the aggregated packets and in y axis here the number of packets are shown that each of 10 nodes is having how many aggregated packets



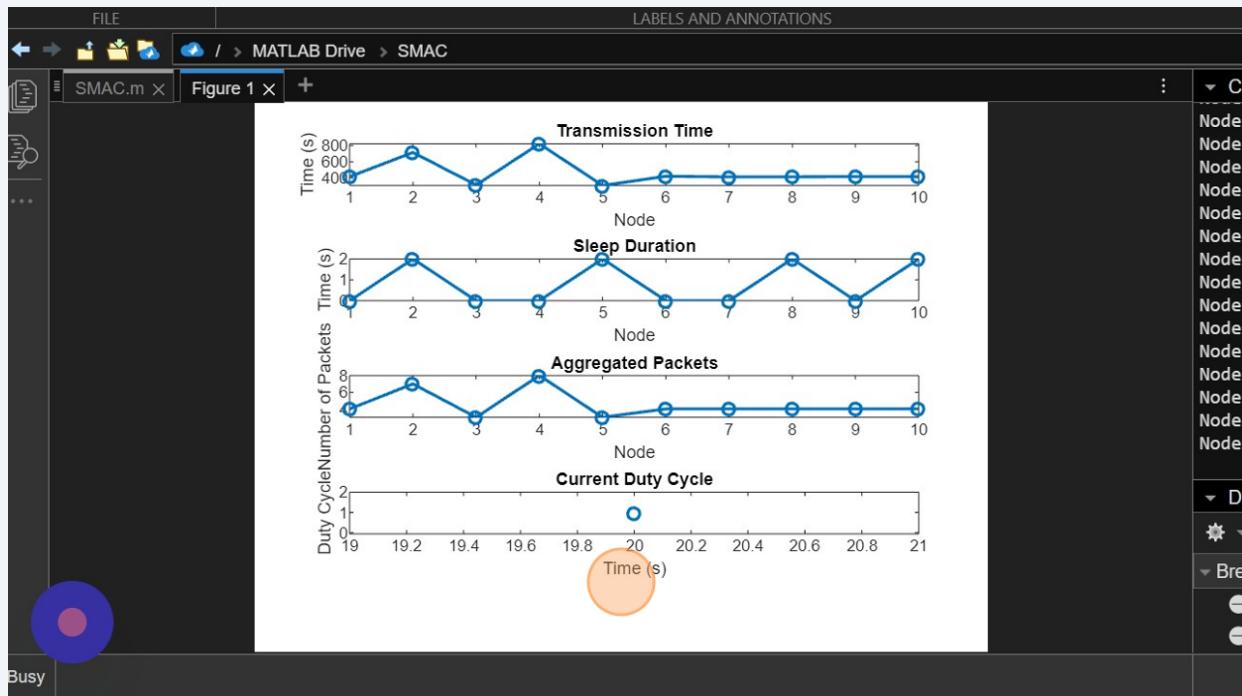
39

Then comes the Current Duty Cycle that is what we have set in the perimeters and initial was 0.5 and total dutycycles are 100

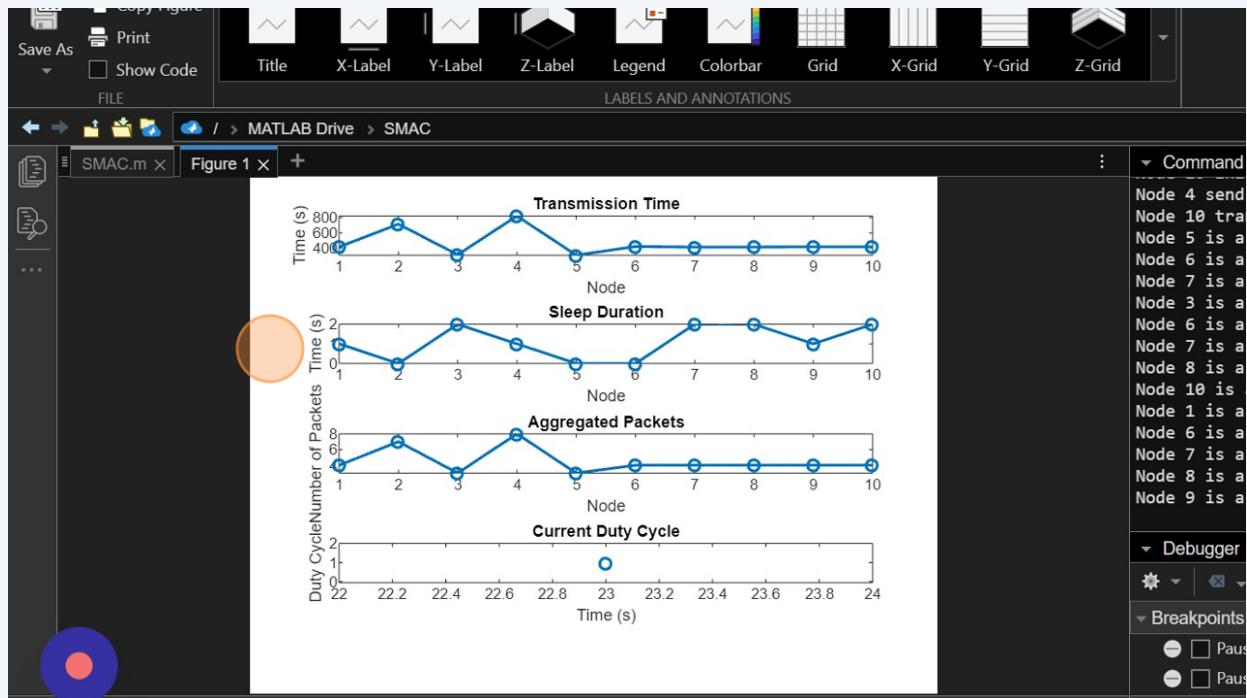


40

On the x axis of duty cycles time is shown bcz with time transmission occurs and duty cycles are changed by adding new to previous one



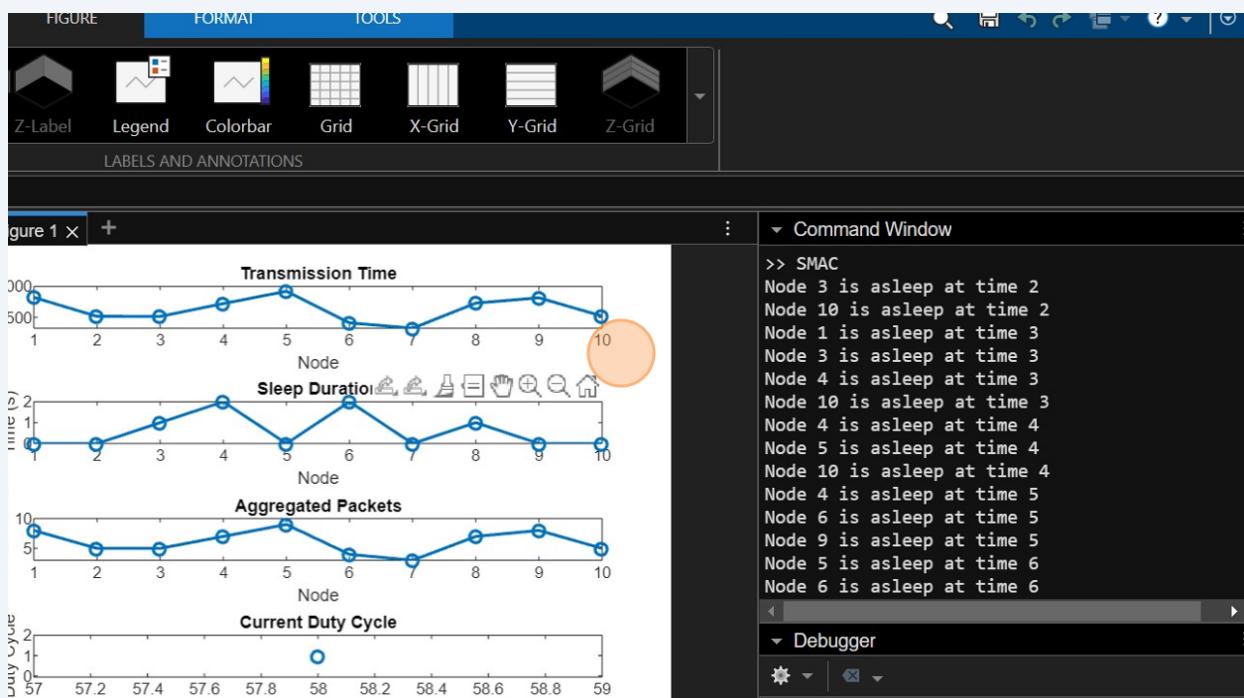
41 Sample figure at 23 Duty Cycles



Step-by-Step Guide to Running and Analyzing SMAC Implementation in MATLAB

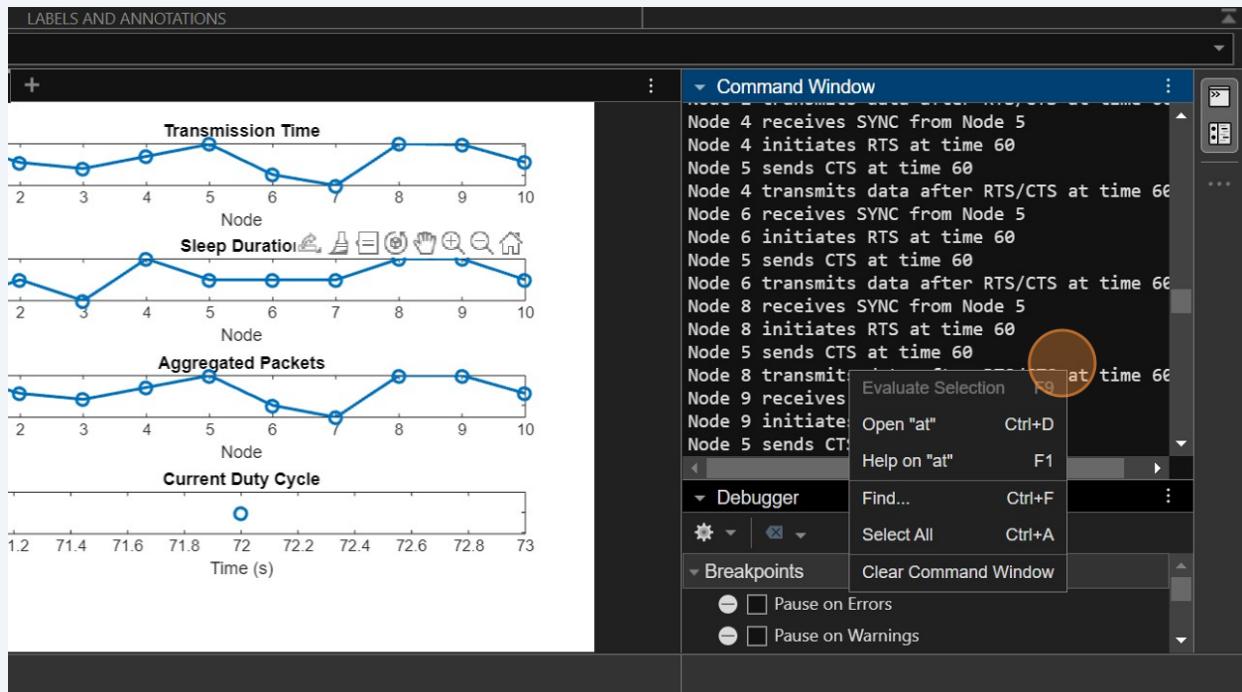
1

Again lets observe it at 58 duty cycles and in the command window from start we can see that the nodes at sleep mode are displayed and their time to sleep is also shown



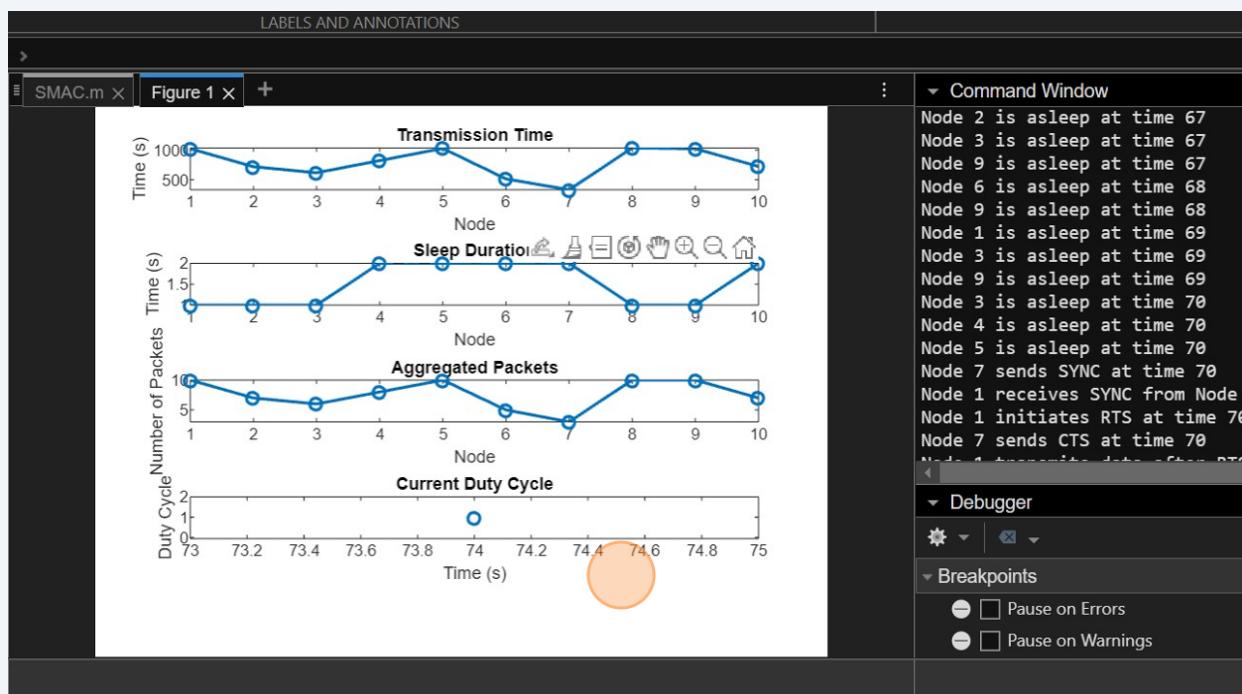
2

Now observing some other commands that were previously discussed in code and theory that node transmitting data is also displayed after the RTS CTS handshake and also the Sync packet received by nodes numbers



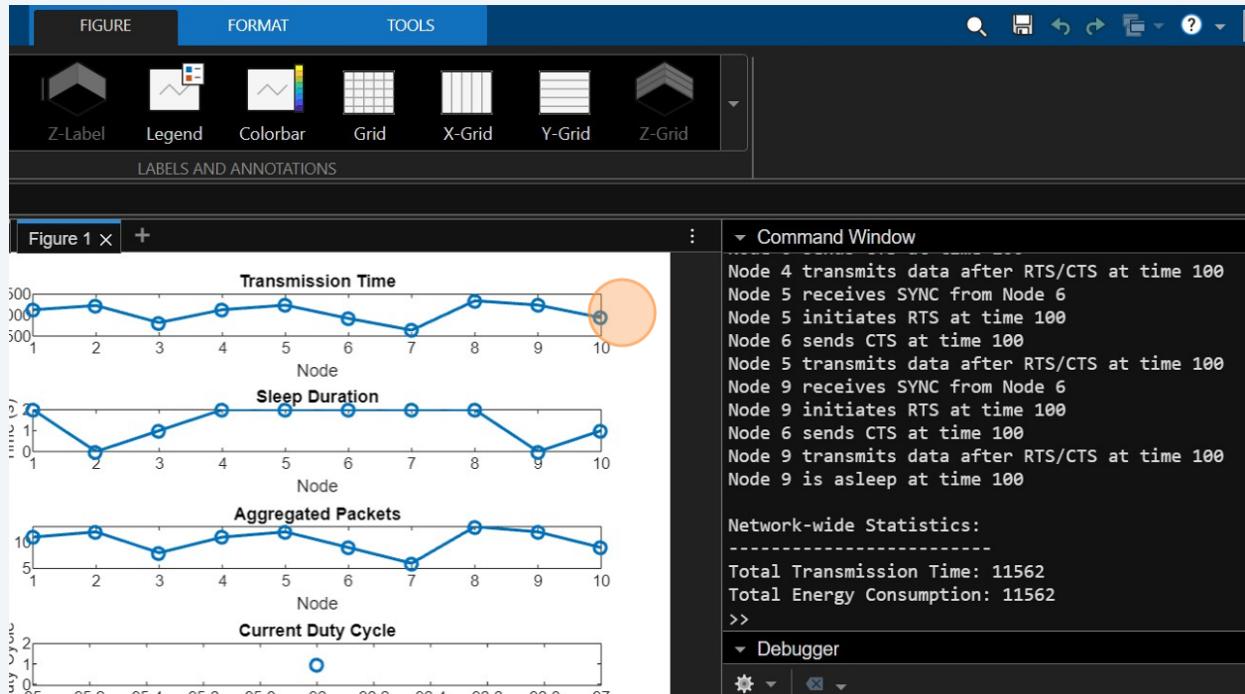
3

Similarly a node initiates RTS at some time and sends CTS at some time and display the msg



4

And after 100 duty cycles the total Network Wide Statistics are displayed as we have seen at the end of our code
Both Total Transmission time as well as total energy consumption is calculated and displayed



5 So this was all about simulation of SMAC and output

The code is Given Below:

```
% Comprehensive S-MAC Simulation with Visualization in MATLAB
```

```
% Parameters
totalNodes = 10; % Total number of nodes in the network
simulationTime = 100; % Simulation time in seconds
packetSize = 100; % Size of each data packet in bytes
trafficLoad = 0.6; % Probability of a sensing event occurring

% Initialize node states
sleepDuration = zeros(1, totalNodes); % Array to store sleep duration for each
node%%array is intialize with zero
transmissionTime = zeros(1, totalNodes); % Array to store transmission time for
each node

% Packet aggregation parameters
aggregatedPackets = zeros(1, totalNodes); % Initialize an array to store
aggregated packets for each node

% Dynamic sleep adjustment parameters
SleepPeriod = 3; % fixed sleep duration of node
syncInterval = 50; % Synchronization interval
AwakePeriods = syncInterval - SleepPeriod;

% Duty cycle parameters%%The duty cycle is the fraction of time that a node is
active or awake compared to the total time
currentDutyCycle = 0.1; % Initial duty cycle

% Synchronization parameters
syncInterval = 50; % Synchronization interval
syncDuration = 1; % Duration for which SYNC is transmitted
rtsThreshold = 0.5; % RTS threshold for initiating RTS/CTS
rtsDuration = 2; % Duration for which RTS is transmitted
ctsDuration = 1; % Duration for which CTS is transmitted
syncPeriod = 10; % Synchronization period for sending SYNC packets periodically

% Simulation loop
for time = 1:simulationTime % Loop over simulation time
    for senderNode = 1:totalNodes % Loop over each node as a potential sender
        % Check if a sensing event occurs
        if rand() < trafficLoad %rand function genrate random value in matlab

            % Check if the node is awake before transmission if sleepDuration(senderNode)
            == 0 % SYNC phase: Send SYNC packets periodically if mod(time, syncPeriod) == 0
            %used to check whether the current simulation time is a multiple of the
            synchronization period disp(['Node ', num2str(senderNode), ' sends SYNC at time ', num2str(time)]); %display message
            transmissionTime(senderNode) = transmissionTime(senderNode) + syncDuration;
            %updating transmission time
        end
    end
end
```

```

%assuming that all the nodes are neighbour nodes%%%%otherwise we can
%calculate through distance and using cluster concept
% Broadcast SYNC packet to neighbors
for neighborNode = 1:totalNodes
    % In a real scenario, you would need to handle collisions
    % Here, we simplify by assuming no collisions in the SYNC phase
    if neighborNode ~= senderNode && rand() < rtsThreshold

        ` %%node receives SYNC disp(['Node ' num2str(neighborNode) ' receives SYNC
        from Node ' num2str(senderNode)]); % RTS phase: Neighbor initiates RTS
        disp(['Node ' num2str(neighborNode) ' initiates RTS at time ' num2str(time)]); %
        Transmit RTS signal transmissionTime(neighborNode) =
        transmissionTime(neighborNode) + rtsDuration; % Neighbor responds with CTS
        disp(['Node ' num2str(senderNode) ' sends CTS at time ' num2str(time)]);
        %%updating the transmission time transmissionTime(senderNode) =
        transmissionTime(senderNode) + ctsDuration; % Continue with data transmission
        (simplified) disp(['Node ' num2str(neighborNode) ' transmits data after RTS/CTS at
        time ' num2str(time)]); transmissionTime(neighborNode) =
        transmissionTime(neighborNode) + packetSize; %record

```