Network 2 Final Course Project

Multiple Access Techniques
(TDMA - Dynamic TDMA - CDMA)

Head

Dr. Peyman Pahlevani

Teaching Assistant

Kiana Ghassabi

Written By

Solmaz Mohamadi

Sepide Bahrami

IASBS – Spring 2020

Institute for Advanced Studies
in Basic Sciences, IASBS

Part A
Implementation with 4 Nodes in Channel

In this project we have used Python programming language to implement the desired multiple access techniques. Looking generally at the structure of code, it is just the same as the description in the project document.
There is a class named **Node** that represents each present node in our channel. An **Array** is declared for Node so the data to be sent for each node will be saved in it. Also **Size**, **Bitrate**, and a **Pointer** variable, which were needed. In the main() part of the code you can easily choose whether you want to have 4 or 400 users in the network.
Our simulated implementations in detail are in the following of the documentation.

Time Division Multiple Access (TDMA)

It is a method of time-sharing between sender nodes. There is a virtual time frame, and every node gets a share of it, which is called **Time Slot**. In a fixed TDMA, slots are equal with no priorities, and they don't change in the process of sending. There is another type of TDMA, which is explained below.

Dynamic TDMA

A different form of TDMA in which the time slots constantly change based on an algorithm or a method. In this project, the dynamic algorithm sums up the bit data that each node has to send individually. It predicts the required slots for each node to send its data based on the node's bitrate, and expected value (which is 2 for all nodes in this case, because of sending probability of ½ for each node). After transforming this information into percentage format, each node is given a share, based on it requirements and number of nodes. The following figures show the transformations of time slots.

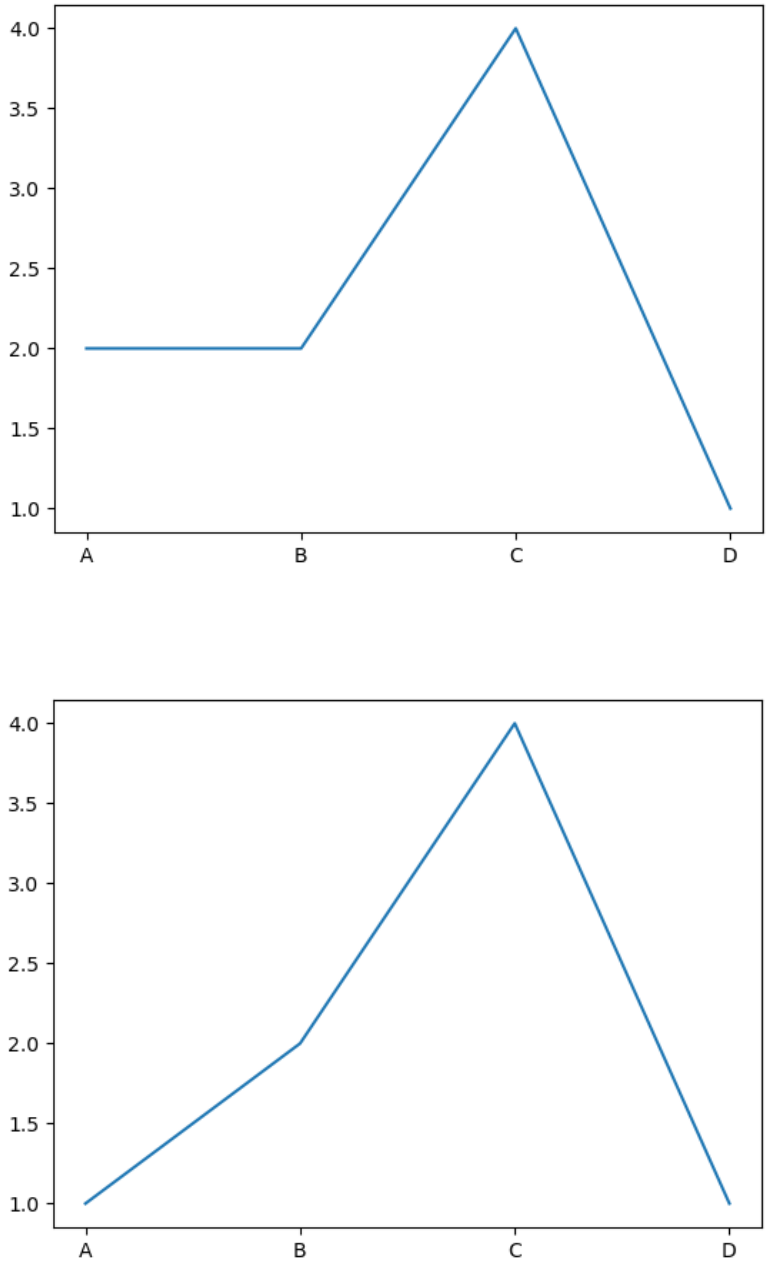
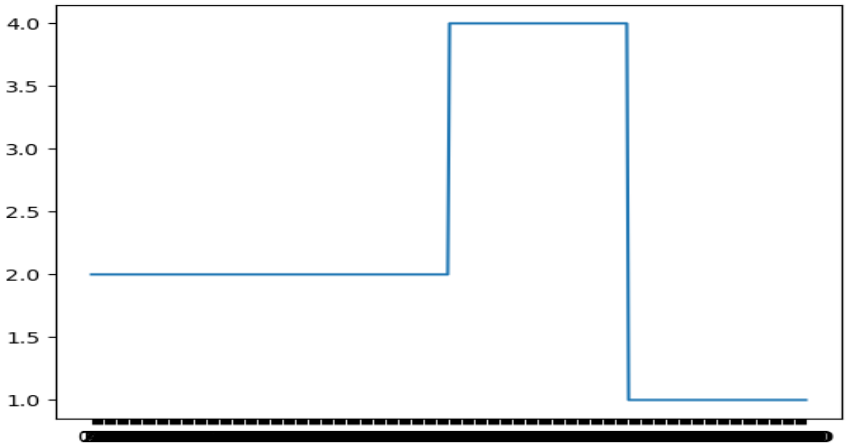
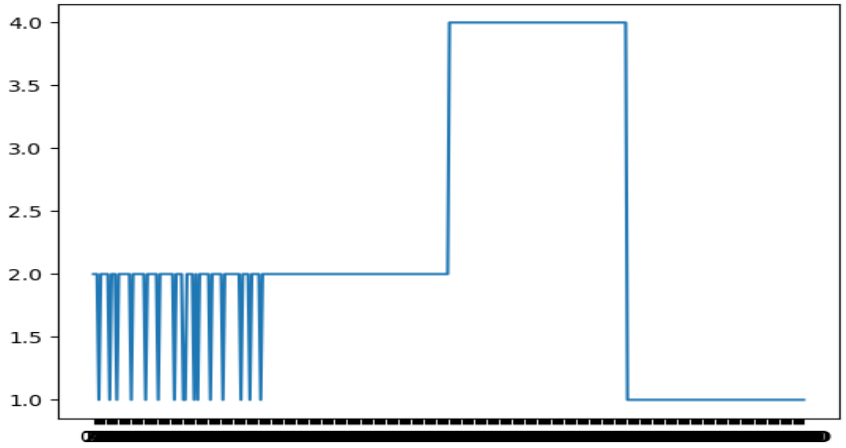


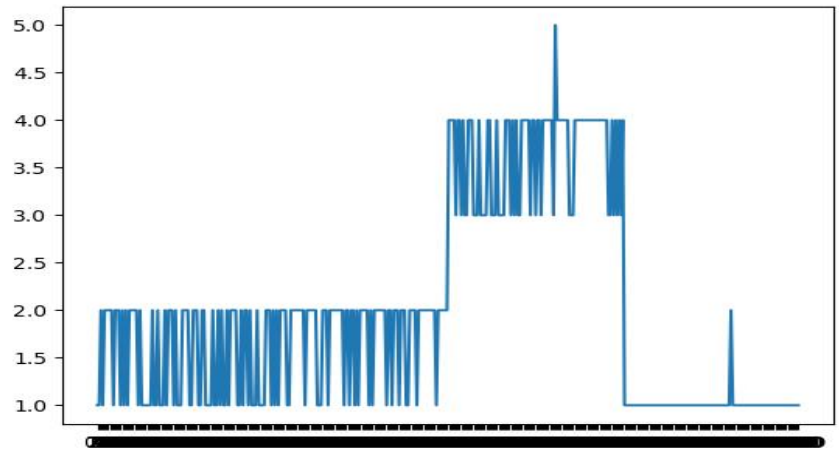
Figure A (2 Graphs)

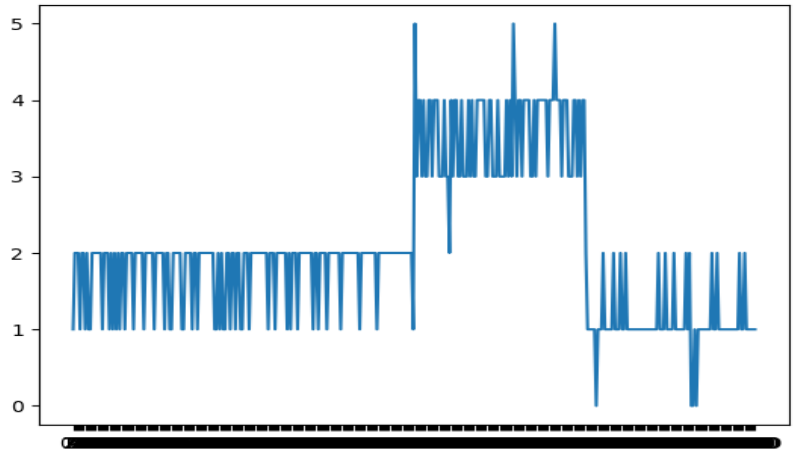
Dynamic TDMA - 4 Nodes Implementation

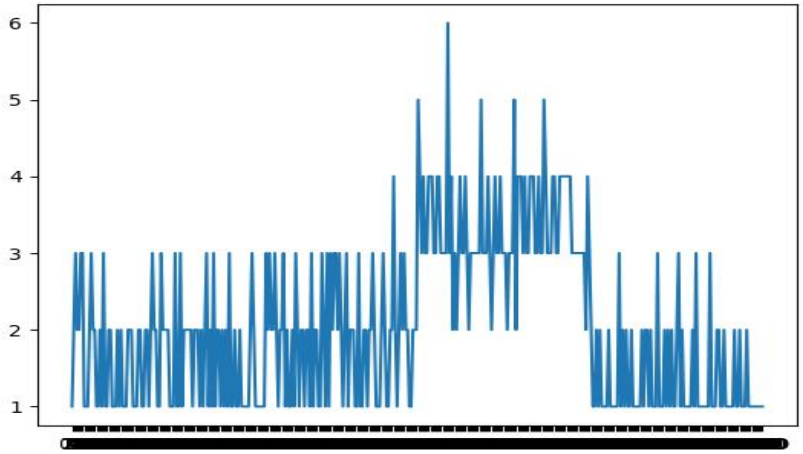
(**Row:** Nodes - **Column:** Time Slot size in scale of 0 to 10)

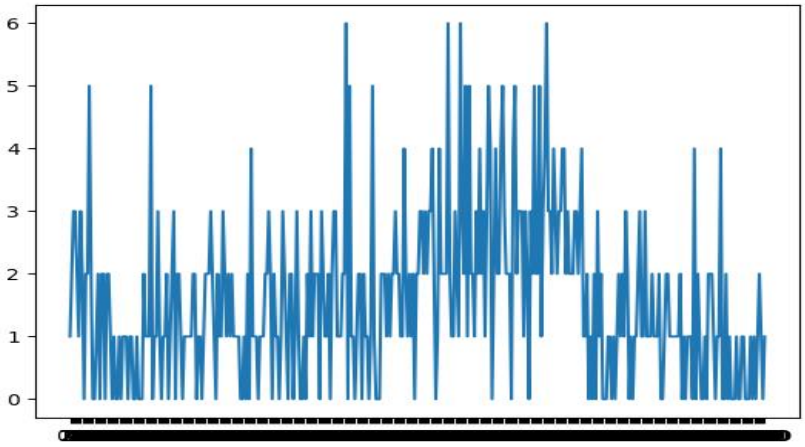

4.0
3.5
3.0
2.5
2.0
1.5
1.0


4.0
3.5
3.0
2.5
2.0
1.5
1.0


5.0
4.5
4.0
3.5
3.0
2.5
2.0
1.5
1.0


5
4
3
2
1
0


6
5
4
3
2
1


6
5
4
3
2
1
0

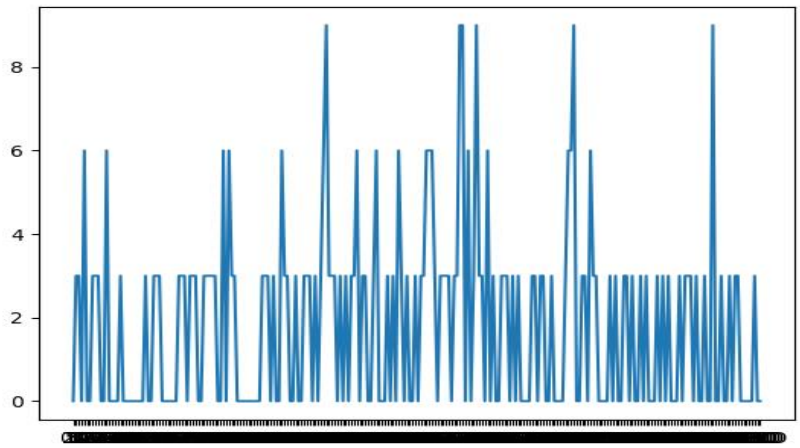


Figure B (7 Graphs)

Dynamic TDMA - 400 Nodes Implementation

(**Row:** Nodes - **Column**: Time Slot size in scale of 0 to 10)

Code Division Multiple Access (CDMA)

It differs from TDMA because all stations can send data simultaneously and there is no time-sharing. In this method, each node with a sending probability of ½ sends its bit through the channel. So how does a node send its bit?

Each node has a specific code based on the **Walsh-Table**. Therefore, the node multiplies either 1, -1 or 0 to the mentioned code. If the bit to be sent is 1 it will be mapped to 1, if the bit to be sent is 0 it will be mapped to -1 and if the node did not got the chance to send a bit or it has nothing to send at that moment, it will be mapped to zero. Then these 1, -1 or 0s are multiplied to the node's code (inner product), which generates a new vector. Therefore, for each node there would be a new generated vector. Finally, all these vectors are summed with each other generating the final summation of all nodes' signals. This process happens in each iteration of sending a bit from the whole data of the nodes in channel. In addition, our Walsh-Table generates valid orthogonal sequences since we tested on some different matrix sizes.

Part B (Modified)
Implementation of 400 Nodes in Channel

We implemented 400 nodes by given details with random binary data using **NumPy** library. We have also changed some initial settings in algorithms in order to work with bigger capacity.

Comparing

TDMA 4 vs. TDMA 400

There is not a worthy difference to mention here. It is just the number of nodes.

D-TDMA 4 vs. D-TDMA 400

In dynamic TDMA, sharing of a time frame has increased from 10 to 1000.

CDMA 4 vs. CDMA 400

In part A, since we have 4 nodes we can use all of the code rows generated by Walsh-Table because we can only build Walsh-Table with size $N = 2^m$. However, in Part B since we have 400 nodes present in the channel we have to build a 512*512 Table for codes. Therefore, 112 of the codes are useless and we cannot make it less because in that situation we can build a 256*256, which does not support many of our nodes.

Question Answers and Analysis

Question 1. A.

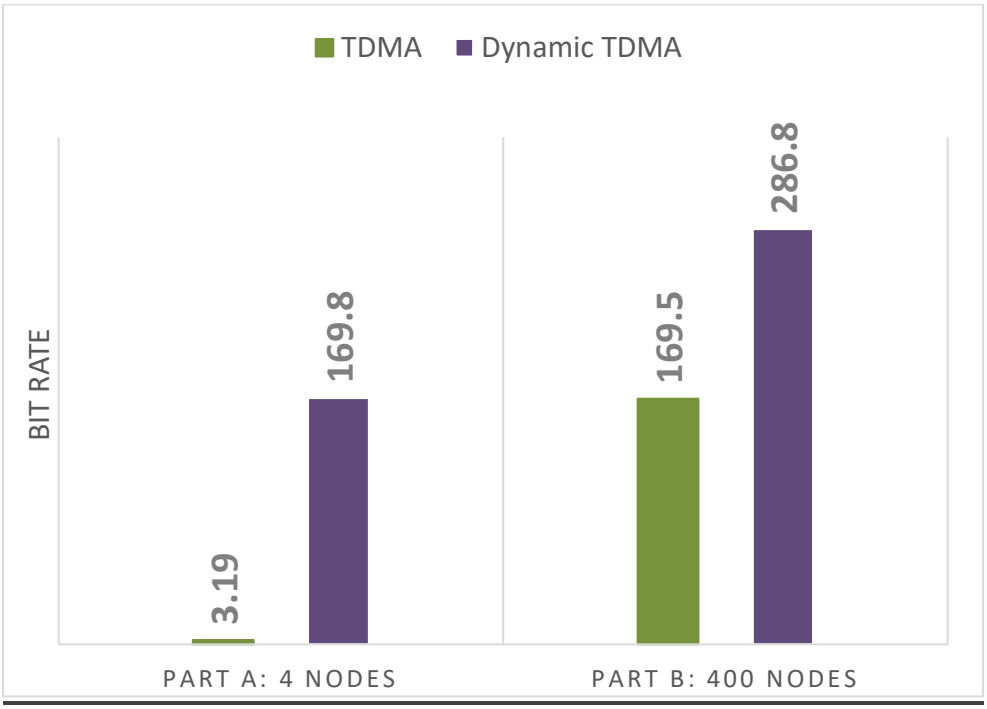
For the CDMA method, explain the result of increasing users in its efficiency.

- As we increase the number of users, we are also increasing the length of our Walsh-Code. So it means that instead of for example sending 4 bits at once (having 4 nodes) we are sending 512 bits (having 400 nodes). It means that we are wasting our channel. So the more increasing in the number of users, the less the efficiency of the method.

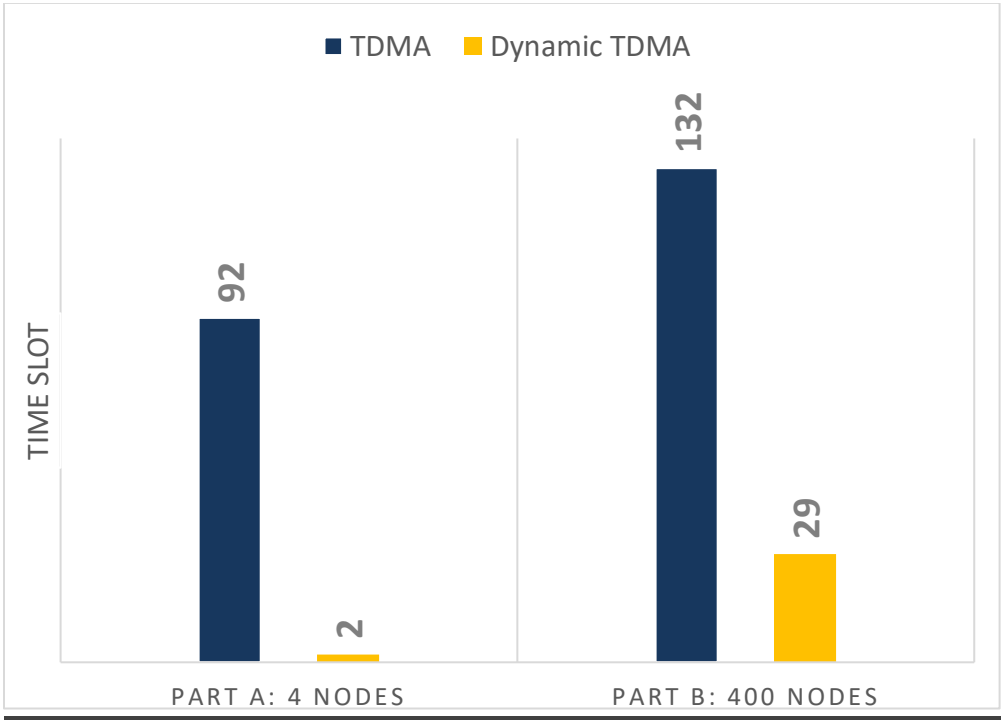
Question 1. B.

For the TDMA and dynamic TDMA method implementation, compute the rate.

TDMA vs. Dynamic TDMA Bit Rate Chart



TDMA vs. Dynamic TDMA Time Slots Chart



Part A with 4 nodes has a total size of 339 bits; Part B with 400 nodes has a total size of 33561 bits. As you can see, the bit rate has increased tremendously in dynamic TDMA, and therefore it uses less timeslots to send data, because it increases good usage of slots.

Question 2.

Which method performs better in different situations? Explain.

- Unlike TDMA, CDMA allows all stations to have access to the full bandwidth of the channel for the entire duration. However, the complexity of multiplications and also building the Walsh-Table itself is a lot. By increasing the number of users, it can be a waste of channel sometimes like the condition in Part B of the project. In fact, it seems to have higher bit rate but it is occupying the channel for a long time; so it is kind of like dividing the channel into multiple channels with less bit rate.
- TDMA works properly in an ideal situation when all nodes have data to send and channel is used efficiently. However, in reality, the ideal situation cannot exist. This is why dynamic TDMA works far better than fixed TDMA. It adapts itself with new situations that has risen and shares time slot in order to have the best possible functionality. Even better than CDMA, just because of adaptive nature of the method.

  In conclusion, we can say that: **TDMA < CDMA < Dynamic TDMA.**

Citation

We got the idea of recursive implementation of building Walsh-Table from this helpful source:

- https://www.geeksforgeeks.org/java-cdma-code-division-multiple-access/

In addition, other useful links, which helped us, are:

- https://ieeexplore.ieee.org/document/5723844
- https://scialert.net/fulltext/?doi=itj.2014.493.500