COMPUTER NETWORKS LAB

NAME:	ARITRA DATTA
ROLL NO:	002010501054
CLASS:	BCSE – III
GROUP:	A2
ASSIGNMENT:	3
DEADLINE:	22rd September, 2022

Problem Statement: In this assignment you have to implement CDMA for multiple access of a common channel by n stations. Each sender uses a unique code word, given by the Walsh set, to encode its data, send it across the channel, and then perfectly reconstruct the data at n stations.

Date of Submission: 26th September, 2022

DESIGN

CDMA is based on coding theory. Each station is assigned a code, which is a sequence of numbers called chips. They are called orthogonal sequences and have the following properties:

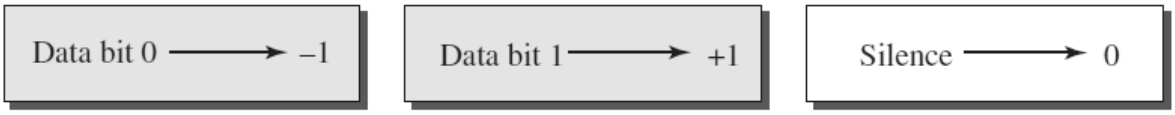
- Product of any two different chips is 0
- A chip multiplied with itself gives the result N (= total number of stations)

The chips are generated using Walsh Table.

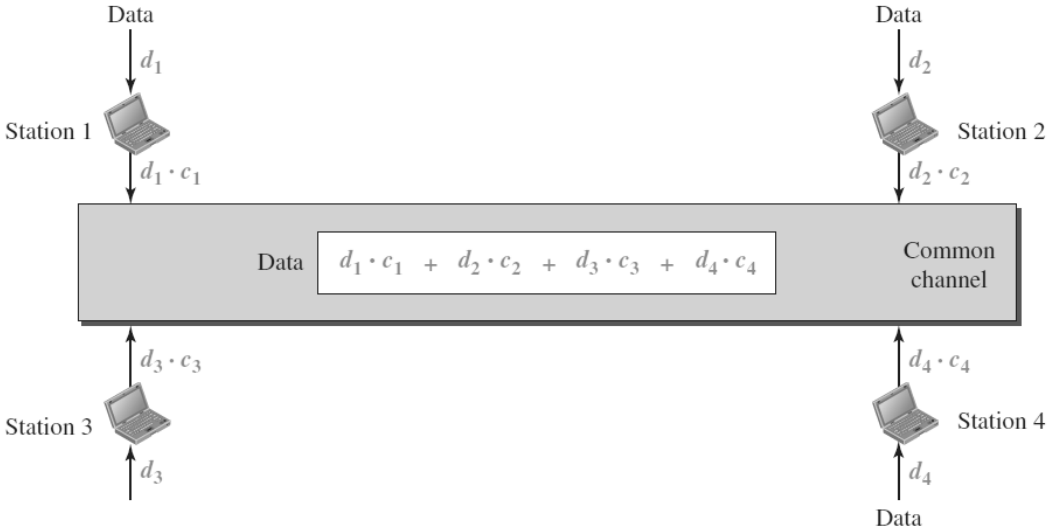
$$W_1 = [+1] \quad W_{2N} = \begin{bmatrix} W_N & W_N \\ W_N & \overline{W_N} \end{bmatrix}$$

The number of sequences in a Walsh Table needs to be a power of 2.

The encoding of bits are as follows:



The data on the channel is present in the form:



To listen to any particular station, the station multiplies the data with the chip sequence of that station.

IMPLEMENTATION

Generation of the Walsh Table

We first get the next greater power of 2, which will be equal to the dimension of the Walsh Table. Then, we build the Walsh Table recursively using the recursive formula given above.

```
def generateWalshTable(n):
    print('Generating Walsh table ...')
    p = 1
    prevtable = [ [1] ]
    while p < n:
        p *= 2
        curtable = []
        for i in range(p):
            tup = []
            for j in range(p):
                tup.append(0)
            curtable.append(tup)
        for i in range(0,p//2):
            for j in range(0,p//2):
                curtable[i][j] = prevtable[i][j]
                curtable[i+p//2][j] = prevtable[i][j]
                curtable[i][j+p//2] = prevtable[i][j]
                curtable[i+p//2][j+p//2] = -1*prevtable[i][j]
        prevtable = curtable

    print('Printing Walsh table :')
    for i in range(p):
        for j in range(p):
            if prevtable[i][j] == 1:
                print(end=' ')
            print(prevtable[i][j],end=' ')
        print()
    return prevtable,p
```

Encoding and Decoding Data

We get the data each station will be sending in the form of a binary string. Each bit of the binary string will be encoded as given :-

```
code = {'0' : -1 , '1' : 1}
dcode = {1 : 1, -1 : 0}
```

We Multiply each part of the encoded data with the chip sequence of the station we want to listen to

```
def updateWalshTable(walshtable, bit, data, p):
    n = len(data)
    table = copy.deepcopy(walshtable)
    for station in range(n):
        for j in range(p):
            if bit >= len(data[station]):
                table[station][j] = 0
            else:
                table[station][j] *= code[data[station][bit]]

    sum_t = []
    for j in range(p):
        tsum = 0
        for i in range(n):
            tsum += table[i][j]
        sum_t.append(tsum)
    return sum_t

def receive_bit(station, bit, data, wtable,sum_t,p):

    if bit >= len(data[station]):
        print("Received Nothing |",end=' ')
    else:
        recv = 0
        for i in range(p):
            recv += wtable[station][i]*sum_t[i]
        recv = recv//p
        print(f"Received bit for station {station} : {dcode[recv]} |", end = ' ')
```

I/O of the Data

```
if __name__ == '__main__':
    n = int(input('Enter number of stations: '))
    data = []
    table,p = generateWalshTable(n)

    for i in range(n):
        x = input(f"Enter Data for Station {i} : ")
        data.append(x)

    l = 1
    for d in data:
        l = max(l,len(d))
    print("Data Received by Channel\n")

    for j in range(0,l):
        sum_table = updateWalshTable(table,j,data,p)
        print("Channel Status : ")
        for i in range(n):
            receive_bit(i,j,data,table,sum_table,p)
        print()
        time.sleep(2)
```

OUTPUT

```
E:\NETWORK\Carrier Sense\CDMA>python channel.py
Enter number of stations: 3
Generating Walsh table ...
Printing Walsh table :
 1  1  1  1
 1 -1  1 -1
 1  1 -1 -1
 1 -1 -1  1
Enter Data for Station 0 : 0011101
Enter Data for Station 1 : 11011
Enter Data for Station 2 : 001011
Data Received by Channel

Channel Status :
Received bit for station 0 : 0 | Received bit for station 1 : 1 | Received bit for station 2 : 0 |
Channel Status :
Received bit for station 0 : 0 | Received bit for station 1 : 1 | Received bit for station 2 : 0 |
Channel Status :
Received bit for station 0 : 1 | Received bit for station 1 : 0 | Received bit for station 2 : 1 |
Channel Status :
Received bit for station 0 : 1 | Received bit for station 1 : 1 | Received bit for station 2 : 0 |
Channel Status :
Received bit for station 0 : 1 | Received bit for station 1 : 1 | Received bit for station 2 : 1 |
Channel Status :
Received bit for station 0 : 0 | Received Nothing | Received bit for station 2 : 1 |
Channel Status :
Received bit for station 0 : 1 | Received Nothing | Received Nothing |

E:\NETWORK\Carrier Sense\CDMA>
```