Result Graphs

April 2, 2019

0.1 Comparing Flooding protocols in a WSN

The sink of a WSN have to periodically (every 60s) distribute some data to all the other nodes. Implement a flooding protocol to distribute such data. Implement two version of the flooding protocol. A trivial version in which each node simply reforward (in broadcast) the packet it receives, and a more efficient version (similar to the CCBR protocol we studied) in which each node waits for a random delay before reforwarding and only reforwards (in broadcast) if during that delay no other node reforwarded the same packet (opportunistic flooding).

Compare the two protocols in terms of:

1. The percentage of nodes that actually receive the flooded data
2. The number of packets used to flood the network

Perform the comparison under different conditions in terms of: total number of nodes, distance among nodes, size of the data to distributed (from 2 bytes up to 20 bytes).

0.2 Simulation Setup

We've divided the simulation into three categories by varying only one of the three conditions stated above at a time while fixing the other two. That way we have the following configurations: 1. **Variable Packet Sizes** (fixed number of nodes and distance among nodes); 2. **Variable Number of Nodes** (fixed packet sizes and distance among nodes); 3. **Variable Distance** (fixed number of nodes and packet sizes);

For each of these three configurations we compare the two protocols in terms of **the percentage of nodes that actually received the flooded data** and **the number of packet used to flood the network**.

```
In [2]: import matplotlib.pyplot as plt
        fig_width = 12
        fig_height = 10
        import seaborn as sns
        plt.style.use('seaborn-whitegrid')

        class bcolors:
            HEADER = '\033[95m'
            OKBLUE = '\033[94m'
            OKGREEN = '\033[92m'
            WARNING = '\033[93m'
```

```
    FAIL = '\033[91m'
    ENDC = '\033[0m'
    BOLD = '\033[1m'
    UNDERLINE = '\033[4m'
```

1 1. Variable Packet Sizes

- packet sizes = [2, . . . , 20]
- number of nodes = 20
- distance = 25
- TX ratio = 80%, RX ratio = 0%
- topology:

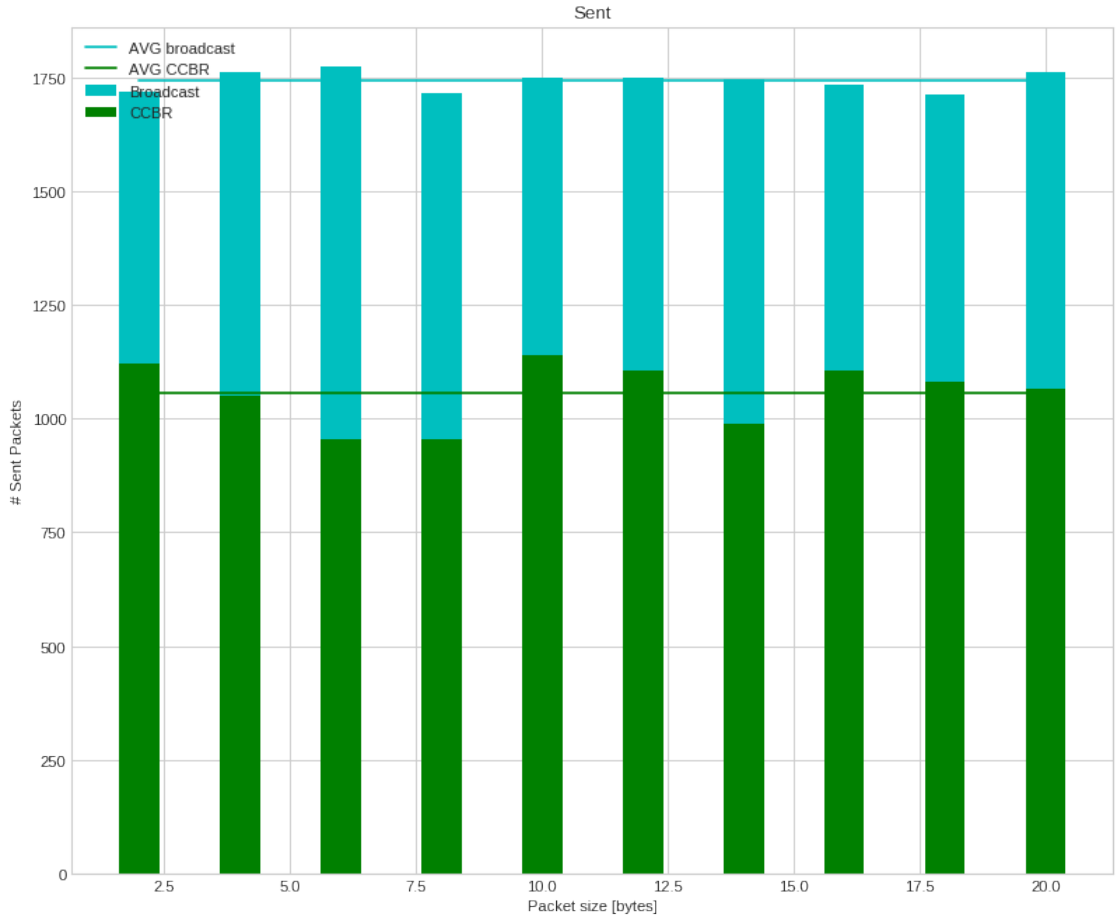
```
In [12]: algorithms = ["broadcast", "ccbr"]
         packet_sizes = [2,4,6,8,10,12,14,16,18,20]
         packets = 100
         nodes = 20
         distance = 25

         sent = {}
         received = {}

         for algorithm in algorithms:
             print("\n\t", bcolors.BOLD + algorithm + bcolors.ENDC)
             sent[algorithm] = list()
             received[algorithm] = list()

             for packet_size in packet_sizes:
                 filename = "results/packet_size/" + algorithm + "_" + str(packet_size) + "_" +
                 file = open(filename, "r")

                 received_by_nodes = list()
                 sent_by_nodes = list()
                 sent_by_sink = list()
                 received_by = []

                 for i in range(0, nodes+1):
                     received_by.append(list())
                 for line in file:
                     if ('SinkC: Broadcasting packet 101' in line):
                         break
                     if ('SinkC: Packet sent' in line):
                         sent_by_sink.append(line)
                     if ('NodeC: Packet sent' in line):
                         sent_by_nodes.append(line)
                     if ('NodeC: Received the new packet' in line):
                         received_by_nodes.append(line)
```

```
        print("Packet size: %d" % packet_size)
        print("\tPackets sent by sink: %d" % len(sent_by_sink))
        print("\tPackets sent:received by nodes: %d:%d" % (len(sent_by_nodes),len(recei
        print("\tReceived packets total: %d/%d, %.2f%%\n" % (len(received_by_nodes), (p

        sent[algorithm].append(len(sent_by_sink) + len(sent_by_nodes))
        received[algorithm].append(len(received_by_nodes)/(packets*nodes)*100)


print("------------ AVG sent Broadcast VS CCBR ------------")
print("Sent AVG broadcast: %.2f" % (sum(sent["broadcast"])/len(sent["broadcast"])))
print("Sent AVG ccbr: %.2f" % (sum(sent["ccbr"])/len(sent["ccbr"])))

print("\n------------  sent Broadcast VS CCBR ------------")
d_sent = []
for i in range(0, len(sent["broadcast"])):
    d_sent.append(sent["broadcast"][i] - sent["ccbr"][i])
    print(" sent %d: %d" % (packet_sizes[i], d_sent[i]))
print("\n sent AVG: %.2f" % (sum(d_sent)/len(d_sent)))

x1 = packet_sizes
y1 = sent["broadcast"]

x2 = packet_sizes
y2 = sent["ccbr"]

plt.figure(figsize=(fig_width, fig_height), dpi= 80)
plt.bar(x1, y1, color = 'c', align = 'center', label="Broadcast")
plt.bar(x2, y2, color = 'g', align = 'center', label="CCBR")
plt.plot(x1, [(sum(sent["broadcast"])/len(sent["broadcast"]))]*len(packet_sizes), color
plt.plot(x1, [(sum(sent["ccbr"])/len(sent["ccbr"]))]*len(packet_sizes), color = 'g', la
plt.title('Sent')
plt.ylabel('# Sent Packets')
plt.xlabel('Packet size [bytes]')
plt.legend(loc='upper left')
plt.show()


print("------------ AVG received Broadcast VS CCBR ------------")
print("Received AVG broadcast: %.2f%%" % (sum(received["broadcast"])/len(received["broa
print("Received AVG ccbr: %.2f%%" % (sum(received["ccbr"])/len(received["ccbr"])))

print("\n------------  received Broadcast VS CCBR ------------")
d_received = []
for i in range(0, len(received["broadcast"])):
    d_received.append(received["broadcast"][i] - received["ccbr"][i])
    print(" received %d: %.2f%%" % (packet_sizes[i], received["broadcast"][i] - receive
```

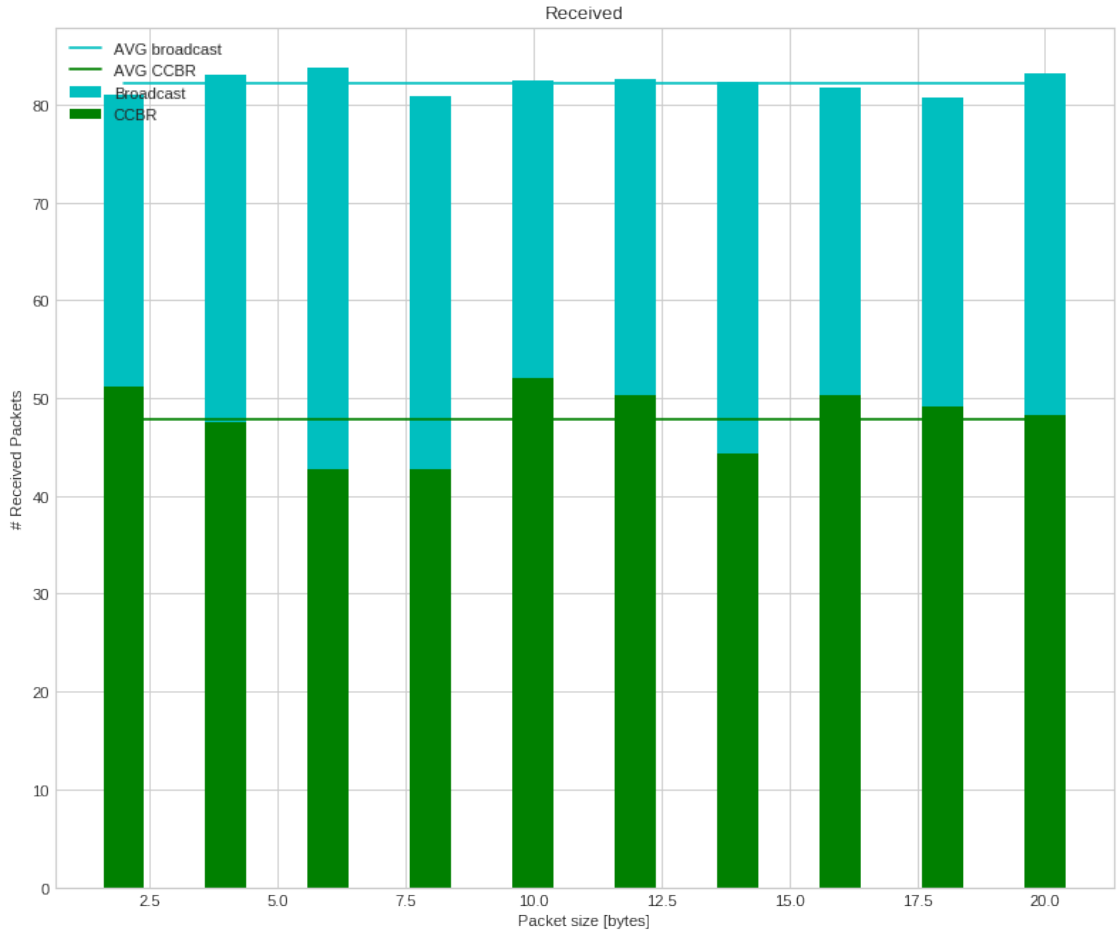
```
        print("\n received AVG: %.2f%%" % (sum(d_received)/len(d_received)))

        x1 = packet_sizes
        y1 = received["broadcast"]

        x2 = packet_sizes
        y2 = received["ccbr"]

        plt.figure(figsize=(fig_width, fig_height), dpi= 80)
        plt.bar(x1, y1, color = 'c', align = 'center', label="Broadcast")
        plt.bar(x2, y2, color = 'g', align = 'center', label="CCBR")
        plt.plot(x1, [(sum(received["broadcast"])/len(received["broadcast"]))]*len(packet_sizes
        plt.plot(x1, [(sum(received["ccbr"])/len(received["ccbr"]))]*len(packet_sizes), color =
        plt.title('Received')
        plt.ylabel('# Received Packets')
        plt.xlabel('Packet size [bytes]')
        plt.legend(loc='upper left')
        plt.show()


        broadcast
Packet size: 2
        Packets sent by sink: 100
        Packets sent:received by nodes: 1621:1621
        Received packets total: 1621/2000, 81.05%

Packet size: 4
        Packets sent by sink: 100
        Packets sent:received by nodes: 1662:1662
        Received packets total: 1662/2000, 83.10%

Packet size: 6
        Packets sent by sink: 100
        Packets sent:received by nodes: 1674:1674
        Received packets total: 1674/2000, 83.70%

Packet size: 8
        Packets sent by sink: 100
        Packets sent:received by nodes: 1616:1616
        Received packets total: 1616/2000, 80.80%

Packet size: 10
        Packets sent by sink: 100
        Packets sent:received by nodes: 1650:1650
        Received packets total: 1650/2000, 82.50%

Packet size: 12
        Packets sent by sink: 100
```

```
        Packets sent:received by nodes: 1652:1652
        Received packets total: 1652/2000, 82.60%

Packet size: 14
        Packets sent by sink: 100
        Packets sent:received by nodes: 1647:1647
        Received packets total: 1647/2000, 82.35%

Packet size: 16
        Packets sent by sink: 100
        Packets sent:received by nodes: 1635:1635
        Received packets total: 1635/2000, 81.75%

Packet size: 18
        Packets sent by sink: 100
        Packets sent:received by nodes: 1615:1615
        Received packets total: 1615/2000, 80.75%

Packet size: 20
        Packets sent by sink: 100
        Packets sent:received by nodes: 1663:1663
        Received packets total: 1663/2000, 83.15%


         ccbr
Packet size: 2
        Packets sent by sink: 100
        Packets sent:received by nodes: 1022:1022
        Received packets total: 1022/2000, 51.10%

Packet size: 4
        Packets sent by sink: 100
        Packets sent:received by nodes: 950:950
        Received packets total: 950/2000, 47.50%

Packet size: 6
        Packets sent by sink: 100
        Packets sent:received by nodes: 856:856
        Received packets total: 856/2000, 42.80%

Packet size: 8
        Packets sent by sink: 100
        Packets sent:received by nodes: 856:856
        Received packets total: 856/2000, 42.80%

Packet size: 10
        Packets sent by sink: 100
        Packets sent:received by nodes: 1040:1040
```

```
        Received packets total: 1040/2000, 52.00%

Packet size: 12
        Packets sent by sink: 100
        Packets sent:received by nodes: 1007:1007
        Received packets total: 1007/2000, 50.35%

Packet size: 14
        Packets sent by sink: 100
        Packets sent:received by nodes: 888:888
        Received packets total: 888/2000, 44.40%

Packet size: 16
        Packets sent by sink: 100
        Packets sent:received by nodes: 1005:1005
        Received packets total: 1005/2000, 50.25%

Packet size: 18
        Packets sent by sink: 100
        Packets sent:received by nodes: 982:982
        Received packets total: 982/2000, 49.10%

Packet size: 20
        Packets sent by sink: 100
        Packets sent:received by nodes: 966:966
        Received packets total: 966/2000, 48.30%

------------ AVG sent Broadcast VS CCBR ------------
Sent AVG broadcast: 1743.50
Sent AVG ccbr: 1057.20

------------  sent Broadcast VS CCBR ------------
 sent 2: 599
 sent 4: 712
 sent 6: 818
 sent 8: 760
 sent 10: 610
 sent 12: 645
 sent 14: 759
 sent 16: 630
 sent 18: 633
 sent 20: 697

 sent AVG: 686.30
```




```
------------ AVG received Broadcast VS CCBR ------------
Received AVG broadcast: 82.17%
Received AVG ccbr: 47.86%

------------  received Broadcast VS CCBR ------------
 received 2: 29.95%
 received 4: 35.60%
 received 6: 40.90%
 received 8: 38.00%
 received 10: 30.50%
 received 12: 32.25%
 received 14: 37.95%
 received 16: 31.50%
 received 18: 31.65%
 received 20: 34.85%

 received AVG: 34.31%
```



1.0.1 Resulting plots analysis

Given the plot above we can observe that **packet size does not influence the percentage of nodes that actually received the flooded data and/or the number of packet used to flood the network**, or at least this does not happen in the interval specified [2-20].

We can also notice that CCBR on average uses **39%** less packets to flood the network that the Broadcast version, while the percentage of nodes that actually receive the flooded data is **34%** less that the Broadcast version.

So while CCBR has a lower rate for received packets it uses a less packets to flood the network.

2 2. Variable Number of Nodes

- number of nodes = [8, . . . , 28]
- packet size = 2 bytes
- distance = 30
- TX ratio = 80%, RX ratio = 80%
- topologies:

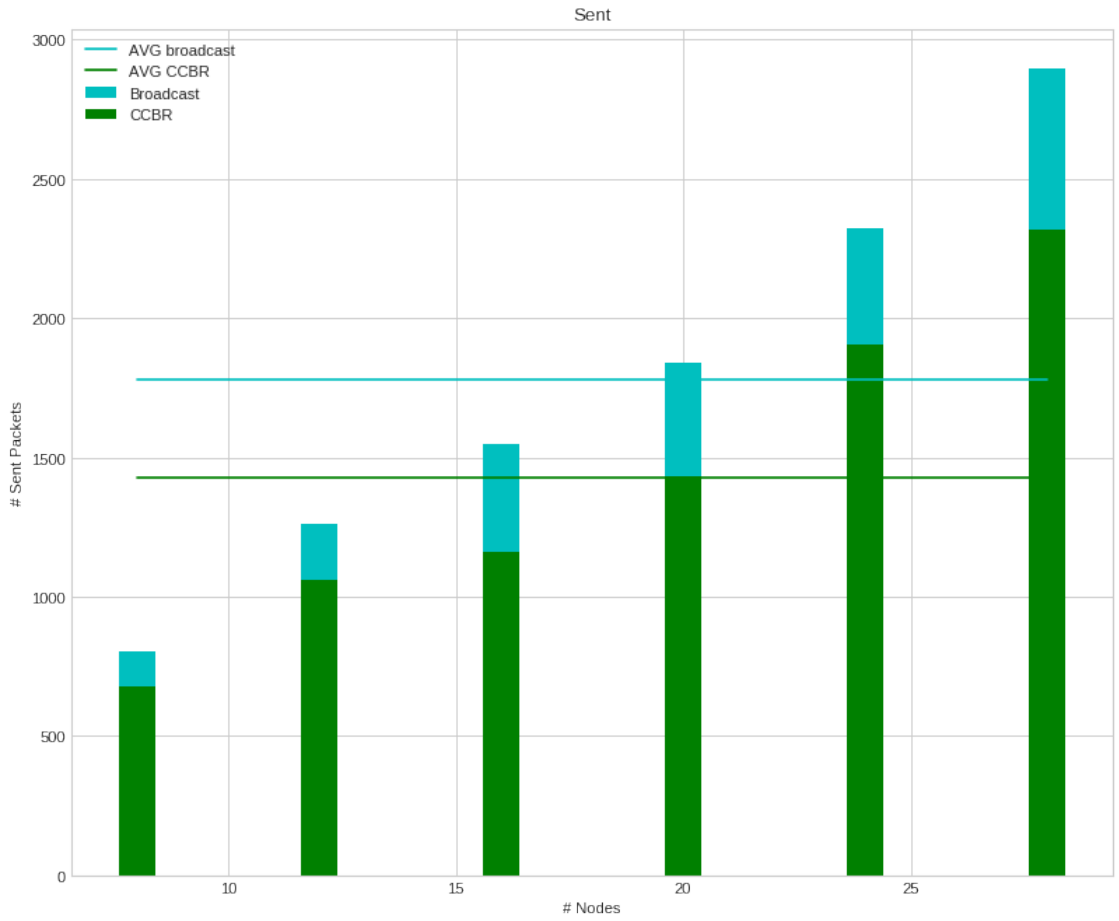
```
1
8 nodes + 1 sink
2
12 nodes + 1 sink
3
16 nodes + 1 sink
4
20 nodes + 1 sink
5
24 nodes + 1 sink
6
28 nodes + 1 sink
```

```
In [14]: algorithms = ["broadcast", "ccbr"]
         packet_size = 2
         packets = 100
         nodes_number = [8,12,16,20,24,28]
         distance = 30

         sent = {}
         received = {}

         for algorithm in algorithms:
             print("\n\t", bcolors.BOLD + algorithm + bcolors.ENDC)
             sent[algorithm] = list()
             received[algorithm] = list()

             for nodes in nodes_number:
                 filename = "results/nodes/" + algorithm + "_" + str(packet_size) + "_" + str(di
                 file = open(filename, "r")

                 received_by_nodes = list()
                 sent_by_nodes = list()
                 sent_by_sink = list()
                 received_by = []

                 for i in range(0, nodes+1):
                     received_by.append(list())
                 for line in file:
                     if ('SinkC: Broadcasting packet 101' in line):
                         break
                     if ('SinkC: Packet sent' in line):
                         sent_by_sink.append(line)
                     if ('NodeC: Packet sent' in line):
                         sent_by_nodes.append(line)
                     if ('NodeC: Received the new packet' in line):
                         received_by_nodes.append(line)
```

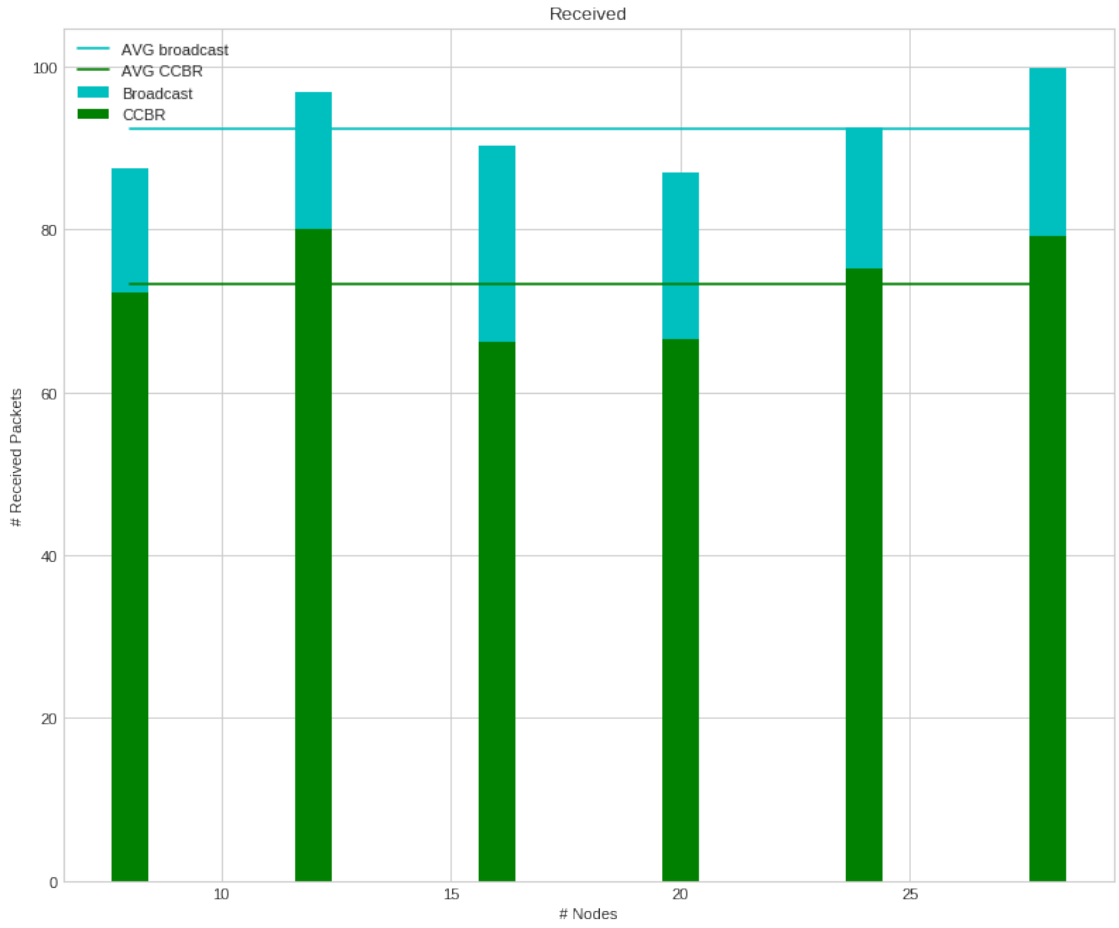
```
            print("Number of nodes: %d" % nodes)
            print("\tPackets sent by sink: %d" % len(sent_by_sink))
            print("\tPackets sent:received by nodes: %d:%d" % (len(sent_by_nodes),len(recei
            print("\tReceived packets total: %d/%d, %.2f%%\n" % (len(received_by_nodes), (p

            sent[algorithm].append(len(sent_by_sink) + len(sent_by_nodes))
            received[algorithm].append(len(received_by_nodes)/(packets*nodes)*100)

print("------------ AVG sent Broadcast VS CCBR ------------")
print("Sent AVG broadcast: %.2f" % (sum(sent["broadcast"])/len(sent["broadcast"])))
print("Sent AVG ccbr: %.2f" % (sum(sent["ccbr"])/len(sent["ccbr"])))

print("\n------------  sent Broadcast VS CCBR ------------")
d_sent = []
for i in range(0, len(sent["broadcast"])):
    d_sent.append(sent["broadcast"][i] - sent["ccbr"][i])
    print(" sent %d: %d" % (nodes_number[i], d_sent[i]))
print("\n sent AVG: %.2f" % (sum(d_sent)/len(d_sent)))

x1 = nodes_number
y1 = sent["broadcast"]

x2 = nodes_number
y2 = sent["ccbr"]

plt.figure(figsize=(fig_width, fig_height), dpi= 80)
plt.bar(x1, y1, color = 'c', align = 'center', label="Broadcast")
plt.bar(x2, y2, color = 'g', align = 'center', label="CCBR")
plt.plot(x1, [(sum(sent["broadcast"])/len(sent["broadcast"]))]*len(nodes_number), color
plt.plot(x1, [(sum(sent["ccbr"])/len(sent["ccbr"]))]*len(nodes_number), color = 'g', la
plt.title('Sent')
plt.ylabel('# Sent Packets')
plt.xlabel('# Nodes')
plt.legend(loc='upper left')
plt.show()


print("------------ AVG received Broadcast VS CCBR ------------")
print("Received AVG broadcast: %.2f%%" % (sum(received["broadcast"])/len(received["broa
print("Received AVG ccbr: %.2f%%" % (sum(received["ccbr"])/len(received["ccbr"])))

print("\n------------  received Broadcast VS CCBR ------------")
d_received = []
for i in range(0, len(received["broadcast"])):
    d_received.append(received["broadcast"][i] - received["ccbr"][i])
    print(" received %d: %.2f%%" % (nodes_number[i], received["broadcast"][i] - receive
print("\n received AVG: %.2f%%" % (sum(d_received)/len(d_received)))
```

```
        x1 = nodes_number
        y1 = received["broadcast"]

        x2 = nodes_number
        y2 = received["ccbr"]

        plt.figure(figsize=(fig_width, fig_height), dpi= 80)
        plt.bar(x1, y1, color = 'c', align = 'center', label="Broadcast")
        plt.bar(x2, y2, color = 'g', align = 'center', label="CCBR")
        plt.plot(x1, [(sum(received["broadcast"])/len(received["broadcast"]))]*len(nodes_number
        plt.plot(x1, [(sum(received["ccbr"])/len(received["ccbr"]))]*len(nodes_number), color =
        plt.title('Received')
        plt.ylabel('# Received Packets')
        plt.xlabel('# Nodes')
        plt.legend(loc='upper left')
        plt.show()


        broadcast
Number of nodes: 8
        Packets sent by sink: 100
        Packets sent:received by nodes: 701:701
        Received packets total: 701/800, 87.62%

Number of nodes: 12
        Packets sent by sink: 100
        Packets sent:received by nodes: 1163:1163
        Received packets total: 1163/1200, 96.92%

Number of nodes: 16
        Packets sent by sink: 100
        Packets sent:received by nodes: 1446:1446
        Received packets total: 1446/1600, 90.38%

Number of nodes: 20
        Packets sent by sink: 100
        Packets sent:received by nodes: 1740:1740
        Received packets total: 1740/2000, 87.00%

Number of nodes: 24
        Packets sent by sink: 100
        Packets sent:received by nodes: 2221:2221
        Received packets total: 2221/2400, 92.54%

Number of nodes: 28
        Packets sent by sink: 100
        Packets sent:received by nodes: 2794:2794
        Received packets total: 2794/2800, 99.79%
```

```
         ccbr
Number of nodes: 8
        Packets sent by sink: 100
        Packets sent:received by nodes: 578:578
        Received packets total: 578/800, 72.25%

Number of nodes: 12
        Packets sent by sink: 100
        Packets sent:received by nodes: 960:960
        Received packets total: 960/1200, 80.00%

Number of nodes: 16
        Packets sent by sink: 100
        Packets sent:received by nodes: 1060:1060
        Received packets total: 1060/1600, 66.25%

Number of nodes: 20
        Packets sent by sink: 100
        Packets sent:received by nodes: 1330:1330
        Received packets total: 1330/2000, 66.50%

Number of nodes: 24
        Packets sent by sink: 100
        Packets sent:received by nodes: 1807:1807
        Received packets total: 1807/2400, 75.29%

Number of nodes: 28
        Packets sent by sink: 100
        Packets sent:received by nodes: 2218:2218
        Received packets total: 2218/2800, 79.21%

------------ AVG sent Broadcast VS CCBR ------------
Sent AVG broadcast: 1777.50
Sent AVG ccbr: 1425.50

------------  sent Broadcast VS CCBR ------------
 sent 8: 123
 sent 12: 203
 sent 16: 386
 sent 20: 410
 sent 24: 414
 sent 28: 576

 sent AVG: 352.00
```




```
------------ AVG received Broadcast VS CCBR ------------
Received AVG broadcast: 92.37%
Received AVG ccbr: 73.25%

------------  received Broadcast VS CCBR ------------
 received 8: 15.38%
 received 12: 16.92%
 received 16: 24.12%
 received 20: 20.50%
 received 24: 17.25%
 received 28: 20.57%

 received AVG: 19.12%
```



2.0.1 Resulting plots analysis

Given the Plot above we can observe that **the number of packet used to flood the network increases linearly with the number of nodes present on the network**, while **the percentage of nodes that actually received the flooded data does not change with the number of nodes**.

We can also notice that CCBR on avarage uses **20%** less packets to flood the network that the Broadcast version, while the percentage of nodes that actually receive the flooded data is **19%** less that the Broadcast version.

3 3. Variable Distance

- distance = [10, . . . , 40]
- packet size = 2 bytes
- number of nodes: 8 nodes + 1 sink
- TX ratio = 80%, RX ratio = 0%
- topologies:

1

```
distance = 10
2
distance = 15
3
distance = 20
4
distance = 25
5
distance = 30
6
distance = 35
7
distance = 40
8
distance = 45
```

```
In [4]: algorithms = ["broadcast", "ccbr"]
        packet_size = 2
        packets = 100
        nodes = 8
        distances = [10,15,20,25,30,35,40,45]

        sent = {}
        received = {}

        for algorithm in algorithms:
            print("\n\t", bcolors.BOLD + algorithm + bcolors.ENDC)
            sent[algorithm] = list()
            received[algorithm] = list()

            for distance in distances:
                filename = "results/distances/" + algorithm + "_" + str(packet_size) + "_" + str
                file = open(filename, "r")

                received_by_nodes = list()
                sent_by_nodes = list()
                sent_by_sink = list()
                received_by = []

                for i in range(0, nodes+1):
                    received_by.append(list())
                for line in file:
                    if ('SinkC: Broadcasting packet 101' in line):
                        break
                    if ('SinkC: Packet sent' in line):
                        sent_by_sink.append(line)
                    if ('NodeC: Packet sent' in line):
                        sent_by_nodes.append(line)
```

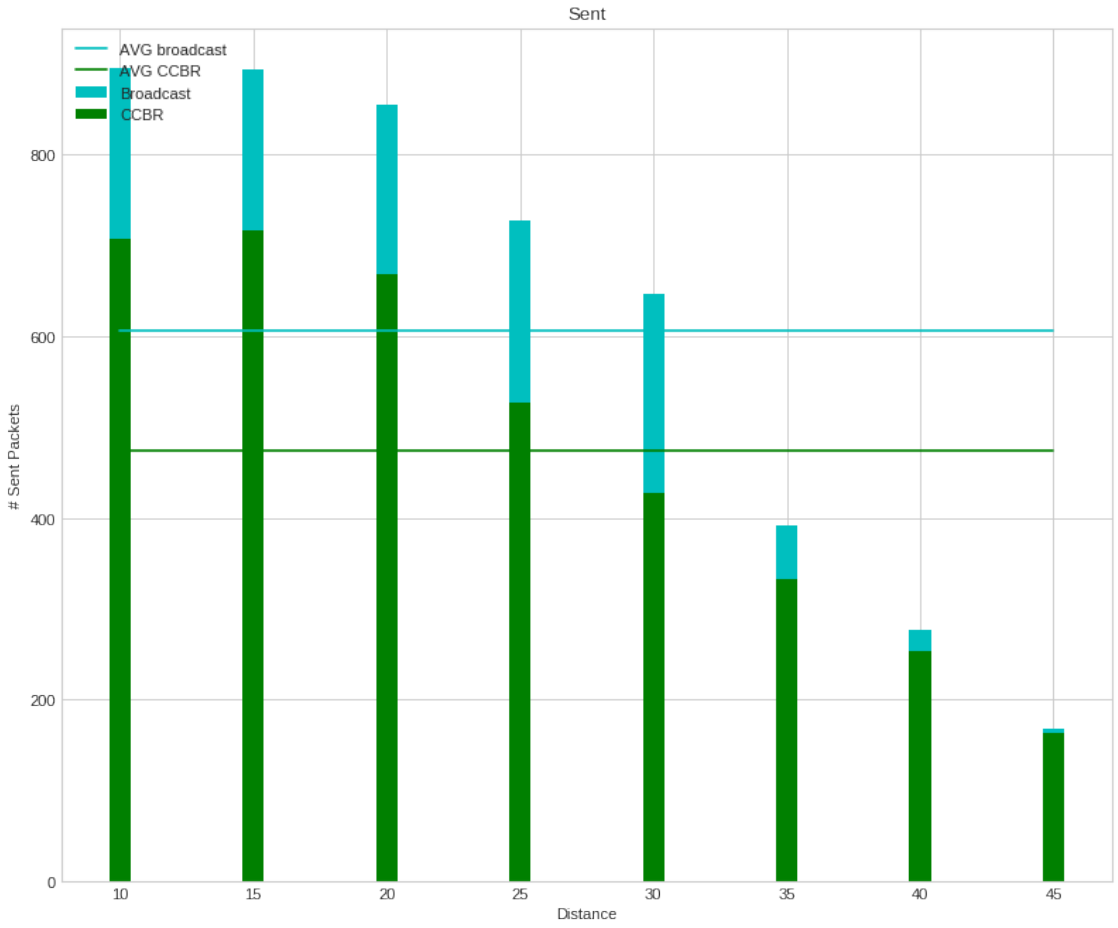
```
            if ('NodeC: Received the new packet' in line):
                received_by_nodes.append(line)

        print("Distance: %d" % distance)
        print("\tPackets sent by sink: %d" % len(sent_by_sink))
        print("\tPackets sent:received by nodes: %d:%d" % (len(sent_by_nodes),len(receiv
        print("\tReceived packets total: %d/%d, %.2f%%\n" % (len(received_by_nodes), (pa

        sent[algorithm].append(len(sent_by_sink) + len(sent_by_nodes))
        received[algorithm].append(len(received_by_nodes)/(packets*nodes)*100)


print("------------ AVG sent Broadcast VS CCBR ------------")
print("Sent AVG broadcast: %.2f" % (sum(sent["broadcast"])/len(sent["broadcast"])))
print("Sent AVG ccbr: %.2f" % (sum(sent["ccbr"])/len(sent["ccbr"])))

print("\n------------  sent Broadcast VS CCBR ------------")
d_sent = []
for i in range(0, len(sent["broadcast"])):
    d_sent.append(sent["broadcast"][i] - sent["ccbr"][i])
    print(" sent %d: %d" % (distances[i], d_sent[i]))
print("\n sent AVG: %.2f" % (sum(d_sent)/len(d_sent)))

x1 = distances
y1 = sent["broadcast"]

x2 = distances
y2 = sent["ccbr"]

plt.figure(figsize=(fig_width, fig_height), dpi= 80)
plt.bar(x1, y1, color = 'c', align = 'center', label="Broadcast")
plt.bar(x2, y2, color = 'g', align = 'center', label="CCBR")
plt.plot(x1, [(sum(sent["broadcast"])/len(sent["broadcast"]))]*len(distances), color = '
plt.plot(x1, [(sum(sent["ccbr"])/len(sent["ccbr"]))]*len(distances), color = 'g', label=
plt.title('Sent')
plt.ylabel('# Sent Packets')
plt.xlabel('Distance')
plt.legend(loc='upper left')
plt.show()


print("------------ AVG received Broadcast VS CCBR ------------")
print("Received AVG broadcast: %.2f%%" % (sum(received["broadcast"])/len(received["broad
print("Received AVG ccbr: %.2f%%" % (sum(received["ccbr"])/len(received["ccbr"])))

print("\n------------  received Broadcast VS CCBR ------------")
d_received = []
for i in range(0, len(received["broadcast"])):
```

```
            d_received.append(received["broadcast"][i] - received["ccbr"][i])
            print(" received %d: %.2f%%" % (distances[i], received["broadcast"][i] - received["c
        print("\n received AVG: %.2f%%" % (sum(d_received)/len(d_received)))

        x1 = distances
        y1 = received["broadcast"]

        x2 = distances
        y2 = received["ccbr"]

        plt.figure(figsize=(fig_width, fig_height), dpi= 80)
        plt.bar(x1, y1, color = 'c', align = 'center', label="Broadcast")
        plt.bar(x2, y2, color = 'g', align = 'center', label="CCBR")
        plt.plot(x1, [(sum(received["broadcast"])/len(received["broadcast"]))]*len(distances), c
        plt.plot(x1, [(sum(received["ccbr"])/len(received["ccbr"]))]*len(distances), color = 'g'
        plt.title('Received')
        plt.ylabel('# Received Packets')
        plt.xlabel('Distance')
        plt.legend(loc='upper left')
        plt.show()
```

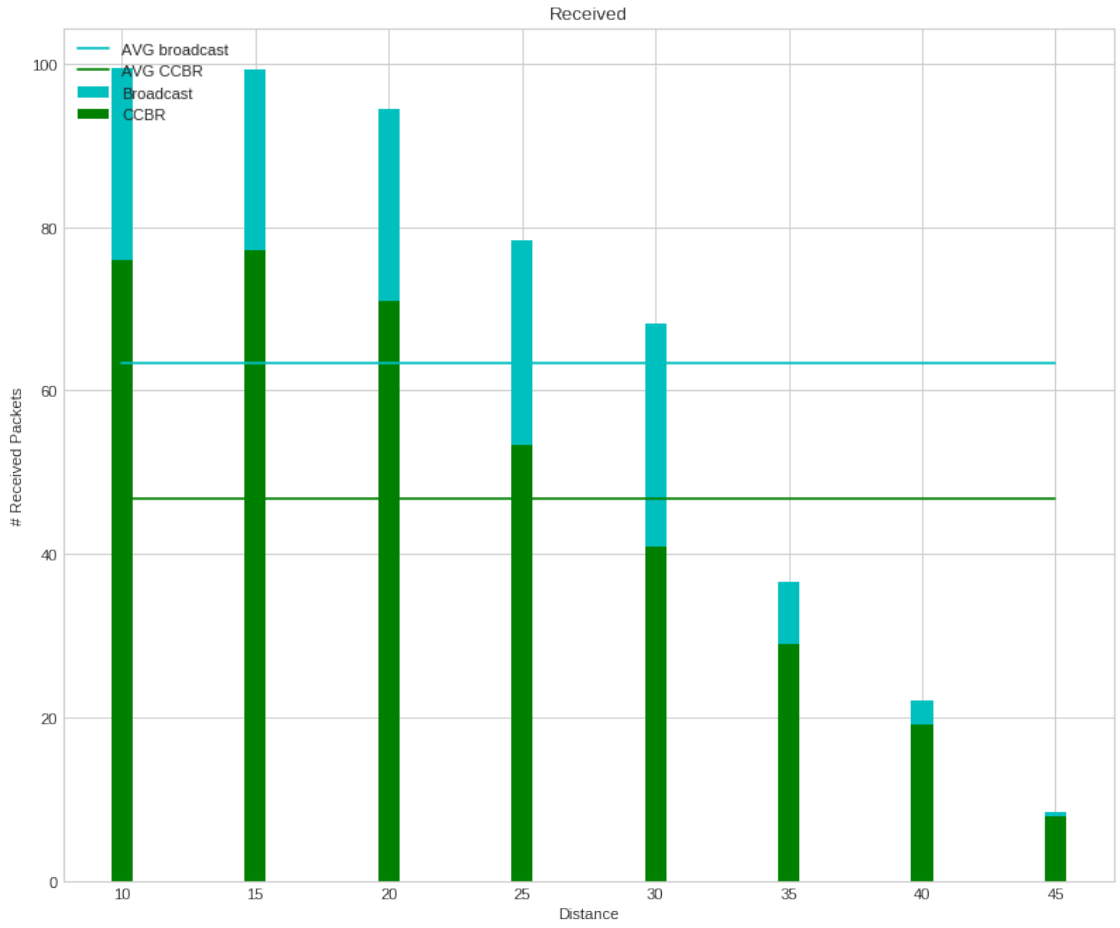
```
 broadcast
Distance: 10
        Packets sent by sink: 100
        Packets sent:received by nodes: 795:795
        Received packets total: 795/800, 99.38%

Distance: 15
        Packets sent by sink: 100
        Packets sent:received by nodes: 794:794
        Received packets total: 794/800, 99.25%

Distance: 20
        Packets sent by sink: 100
        Packets sent:received by nodes: 755:755
        Received packets total: 755/800, 94.38%

Distance: 25
        Packets sent by sink: 100
        Packets sent:received by nodes: 627:627
        Received packets total: 627/800, 78.38%

Distance: 30
        Packets sent by sink: 100
        Packets sent:received by nodes: 546:546
        Received packets total: 546/800, 68.25%
```

```
Distance: 35
        Packets sent by sink: 100
        Packets sent:received by nodes: 292:292
        Received packets total: 292/800, 36.50%

Distance: 40
        Packets sent by sink: 100
        Packets sent:received by nodes: 177:177
        Received packets total: 177/800, 22.12%

Distance: 45
        Packets sent by sink: 100
        Packets sent:received by nodes: 67:67
        Received packets total: 67/800, 8.38%


         ccbr
Distance: 10
        Packets sent by sink: 100
        Packets sent:received by nodes: 608:608
        Received packets total: 608/800, 76.00%

Distance: 15
        Packets sent by sink: 100
        Packets sent:received by nodes: 617:617
        Received packets total: 617/800, 77.12%

Distance: 20
        Packets sent by sink: 100
        Packets sent:received by nodes: 568:568
        Received packets total: 568/800, 71.00%

Distance: 25
        Packets sent by sink: 100
        Packets sent:received by nodes: 427:427
        Received packets total: 427/800, 53.37%

Distance: 30
        Packets sent by sink: 100
        Packets sent:received by nodes: 327:327
        Received packets total: 327/800, 40.88%

Distance: 35
        Packets sent by sink: 100
        Packets sent:received by nodes: 232:232
        Received packets total: 232/800, 29.00%

Distance: 40
```

```
        Packets sent by sink: 100
        Packets sent:received by nodes: 153:153
        Received packets total: 153/800, 19.12%

Distance: 45
        Packets sent by sink: 100
        Packets sent:received by nodes: 63:63
        Received packets total: 63/800, 7.88%

------------ AVG sent Broadcast VS CCBR ------------
Sent AVG broadcast: 606.62
Sent AVG ccbr: 474.38

------------  sent Broadcast VS CCBR ------------
 sent 10: 187
 sent 15: 177
 sent 20: 187
 sent 25: 200
 sent 30: 219
 sent 35: 60
 sent 40: 24
 sent 45: 4

 sent AVG: 132.25
```




```
------------ AVG received Broadcast VS CCBR ------------
Received AVG broadcast: 63.33%
Received AVG ccbr: 46.80%

------------  received Broadcast VS CCBR ------------
 received 10: 23.38%
 received 15: 22.12%
 received 20: 23.38%
 received 25: 25.00%
 received 30: 27.38%
 received 35: 7.50%
 received 40: 3.00%
 received 45: 0.50%

 received AVG: 16.53%
```



3.0.1 Resulting plots analysis

Given the plot above we can observe that **by increasing the distance between the nodes, both the number of packet used to flood the network and the percentage of nodes that actually received the flooded data decrease linearly**.

We can also notice that CCBR on avarage uses **21%** less packets to flood the network that the Broadcast version, while the percentage of nodes that actually receive the flooded data is **16%** less that the Broadcast version.

3.1 ———

Final Thoughts For each of the three configurations we compared, we observed that the two protocols had the same general behaviour. While the only thing differentiating the two is that CCBR has a lower rate of received packets it uses a less packets to flood the network. This can be useful on some applications like for example TinyDB that treats the sensor network as a database.