MCbizantino

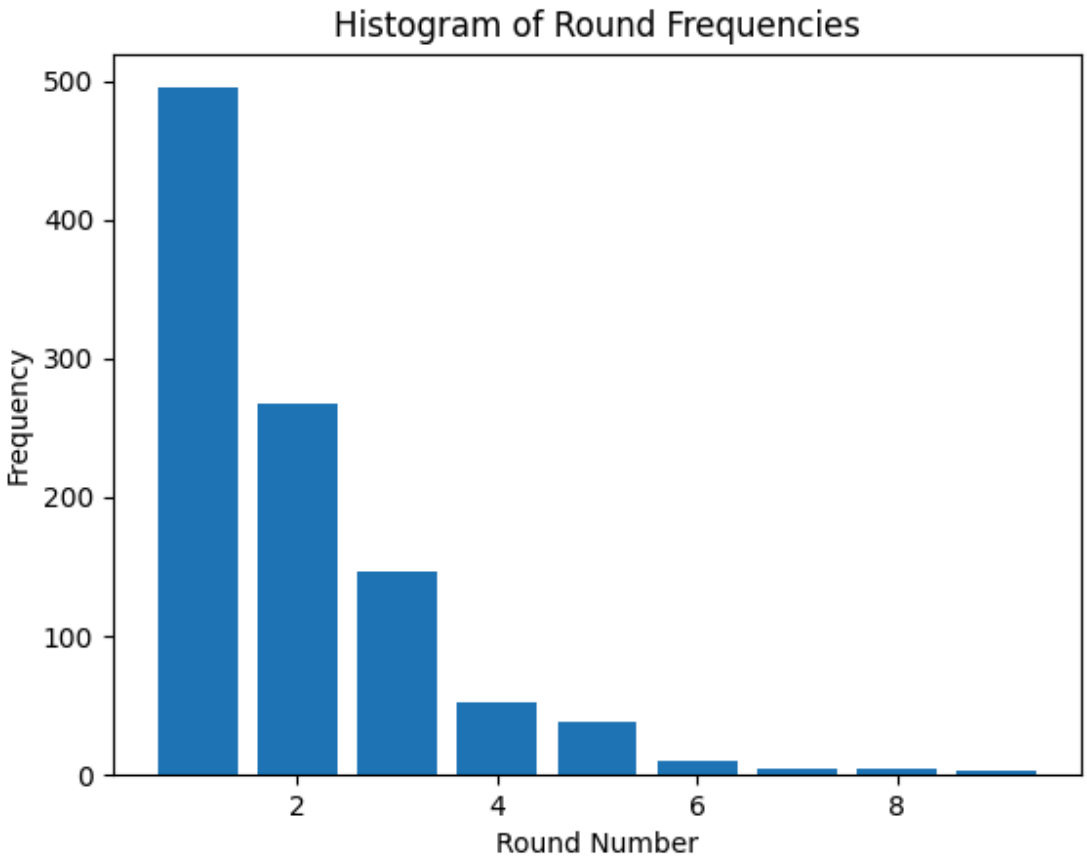
Implementazione in python

```python
import random
import math
import matplotlib.pyplot as plt
from collections import Counter

def maj(v):
    # ritorna il valore maggioritario (1 o 0) nella lista v
    c1 = sum(v)
    c0 = len(v) - c1
    return int(c1 > c0)

def tally(v):
    # ritorna il conteggio del valore maggioritario nella lista v
    c1 = sum(v)
    c0 = len(v) - c1
    return max(c1, c0)

# seme per la generazione di numeri casuali
random.seed()

# parametri
num_rounds = int(math.pow(2, 10))  # numero di round massimi
num_malicious = 1  # numero di nodi maliziosi
total_nodes = 4  # numero totale di nodi
reliable_transmissions = 2 * num_malicious + 1  # numero di trasmissioni affidabili
rounds_counter = Counter()  # contatore dei round
```

```python
for _ in range(num_rounds):
    current_round = 0  # numero di round correnti
    # matrice di trasmissione
    # 4 righe e vuota perche e il processo malizioso
    matrix = [
        [1, 1, 0, 0],
        [1, 1, 0, 0],
        [1, 1, 0, 1]
    ]

    while True:
        # tirare una moneta
        coin = random.randint(0, 1)
        current_round += 1

        bits = [None] * total_nodes
        for j in range(reliable_transmissions):
            # calcola il valore maggioritario e il conteggio del valore maggioritario
            majority = maj(matrix[j])
            tally_count = tally(matrix[j])
            # in base al conteggio del valore maggioritario, assegna il valore maggioritario o il valore della moneta
            bits[j] = majority if tally_count >= reliable_transmissions else coin

        # Impostazione valori del malizioso
        bits[-1] = random.randint(0, 1)

        # aggiorna la matrice di trasmissione
        for v in matrix:
            v[:] = bits
        for j in range(reliable_transmissions):
            matrix[j][-1] = 1 - bits[j]

        # controlla se tutti i nodi hanno lo stesso valore
        if all(bits[0] == bits[k] for k in range(reliable_transmissions)):
            break

    rounds_counter[current_round] += 1
```

```python
# stampa il contatore dei round
rounds_list = list(rounds_counter.items())
rounds_list.sort()
print(f"Final rounds counter: {rounds_list}")

# costruzione dell'istogramma
round_numbers = [item[0] for item in rounds_list]
frequencies = [item[1] for item in rounds_list]

plt.bar(round_numbers, frequencies)
plt.xlabel('Round Number')
plt.ylabel('Frequency')
plt.title('Histogram of Round Frequencies')
plt.savefig('round_frequencies_histogram.png')
plt.show()
```

Print Final rounds counter: [(1, 495), (2, 268), (3, 147), (4, 53), (5, 38), (6, 10), (7, 5), (8, 5), (9, 3)]



Grazie al protocollo protocollo Monte Carlo si riesce molto spesso a ottenere un accordo prima dell round 10