# Accordo Bizantino MC

Lorenzo Livio Vaccarecci (matr. 5462843)

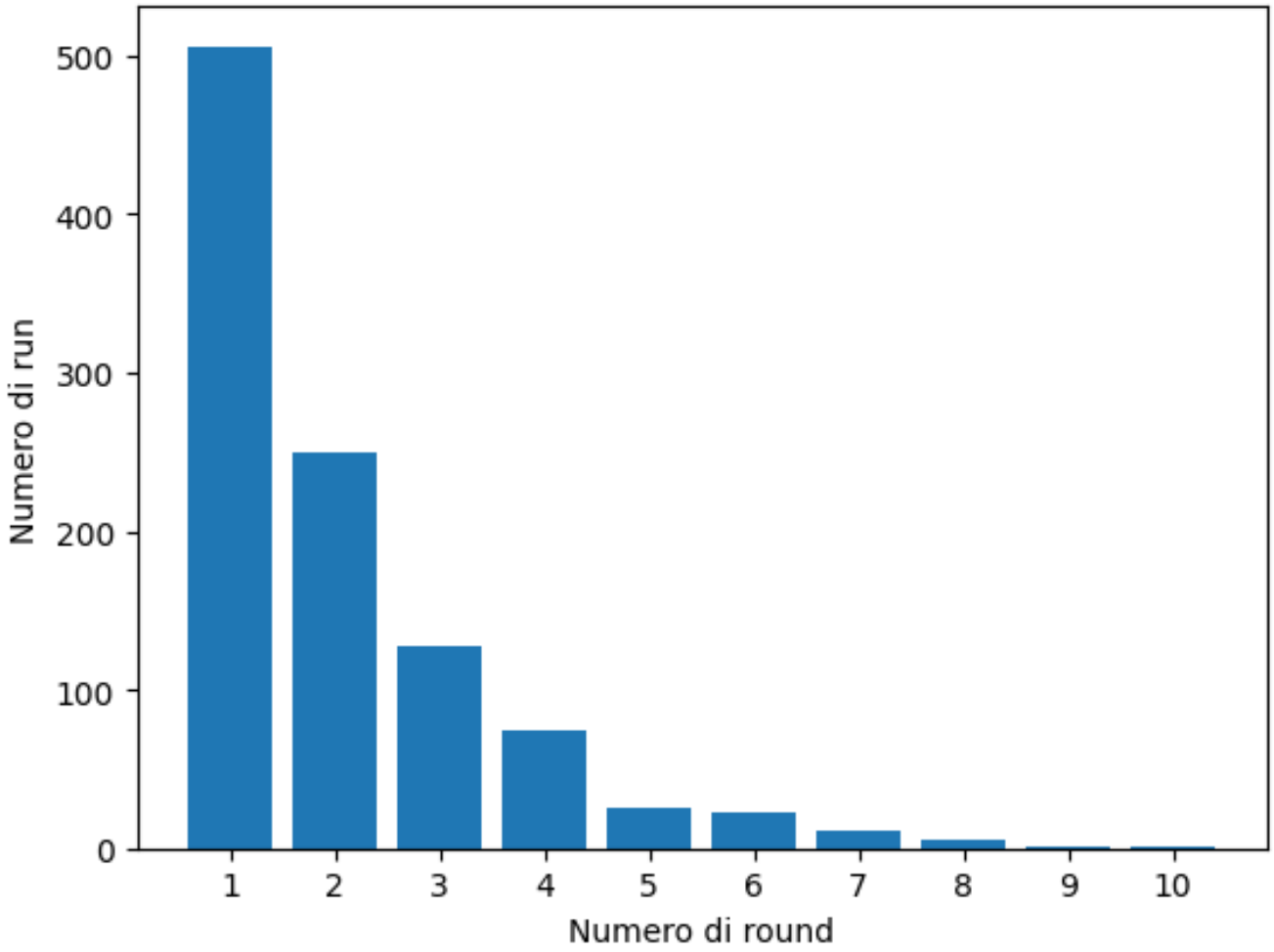
A.A. 2023/2024

## 1 Codice

```
import random
import matplotlib.pyplot as plt

def tally(j, maj, matrix):
    eq = 0
    for i in range(0, n):
        if matrix[j][i] == maj:
            eq += 1
    return eq

def maj(j, matrix):
    max = [0,0]
    for i in range(0, n):
        if matrix[j][i] == 0:
          max[0]+=1
        else:
          max[1]+=1
    return 0 if max[0] > max[1] else 1

def coin():
    return random.randint(0, 1)

def MCByzantineGeneral(matrix):
    matrix_copy = [row[:] for row in matrix]
    for r in range(10):
      coinFlip = coin()

      for j in range(n):
        for i in range(n):
          if i != j:
            matrix_copy[i][j] = matrix_copy[j][j]

      for j in range(n):
        for i in range(n):
          if i != j:
            matrix_copy[j][i] = matrix_copy[i][j]
```

```
        success = False
        for j in range(n):
            for i in range(n):
                majVar = maj(j, matrix_copy)
                if tally(j, majVar, matrix_copy) >= T:
                    matrix_copy[i][j] = majVar
                    if majVar == coinFlip:
                      success = True
                    break
                else:
                    matrix_copy[i][j] = coinFlip

        if success:
            roundSucc[r] += 1
            return

R=2**10
n=4
f=1
T=(2*f)+1

bitSndRcv = [[0 for _ in range(n)] for _ in range(n)]
for i in range(0, n):
    for j in range(0, n):
        if i == 3:
            bit = 1 - bitSndRcv[j][i]
        else:
            bit = random.randint(0, 1)
        bitSndRcv[i][j]=bit

roundSucc = [0 for _ in range(10)]
for i in range(R): # Run
    MCByzantineGeneral(bitSndRcv)

plt.bar(list(range(1,len(roundSucc)+1)), roundSucc)
plt.xlabel("Numero di round")
plt.ylabel("Numero di run")
plt.show()
```

## 2 Osservazioni




Il grafico suggerisce che la maggior parte delle $2^{10}$ esecuzioni dell'accordo bizantino viene completata nel primo round, con un dimezzamento del successo ad ogni round successivo. Questo è dovuto al fatto che non tutti i processi affidabili concordano e che per qualche processo $p$ avremo $tally(p) \geq T$, almeno $f + 1$ processi affidabili il bit che verrà spedito è il maggioritario di $p$, e quindi non può esserci un altro processo affidabile che abbia il tally maggiore o uguale a $T$ con i bit maggioritari diversi da quello di $p$. Quindi con probabilità $\frac{1}{2}$, il lancio della moneta, tutti i processi affidabili convergono al bit maggiorante di $p$. **Da finire!**