Las Vegas QuickSort – Cattaneo Kevin – S4944382

Per il corrente esercizio si è ritenuto necessario usufruire del linguaggio di programmazione C++ per l'esecuzione dell'algoritmo LVQS nelle sue 10^5 run, per poi passare a Python per il plotting dei dati e la loro rappresentazione su un grafico, mediando i dati attraverso un file contenente il numero di iterazioni per run, fornito in output dal C++ e in input a Python.

Di seguito il codice C++

```
#include <iostream>
#include <algorithm>
#include <random>
#include <chrono>
#include <fstream>

const int DIM = pow(10,4);
const int NUMITER = pow(10,5);
int k = 0; // var ausiliaria per tenere in memoria il numero di confronti

int partition(std::vector<int>& seq, int start, int end){
    int p = seq[end];
    int i = start;
    int j = end - 1;
    k += end-start;

  while (true){
    while (i <= j && seq[j] >= p) j--;
    while (i <= j && seq[i] <= p) i++;
    if (i > j) break;
    int tempy = seq[i];
    seq[i] = seq[j];
    seq[j] = tempy;
  }

  // Riporto il pivot in posizione
  int temp = seq[i];
  seq[i] = seq[end];
  seq[end] = temp;
  return i;
}

void qsort(std::vector<int>& seq, int start, int end){
  if (start < end){
    int p = partition(seq, start, end);
    qsort(seq, start, p - 1);
    qsort(seq, p + 1, end);
  }
}

void lvqs(std::vector<int>& seq){
  unsigned int seed = std::chrono::system_clock::now().time_since_epoch().count();
  std::shuffle(seq.begin(), seq.end(), std::default_random_engine(seed));
  qsort(seq,0,DIM-1);
}
```

```cpp
int main(){
  std::vector<int> S(DIM);
  std::vector<int> X(NUMITER); // confronti per run

  // Inserisco valori casuali nella sequenza
  for (int i=0; i<DIM; i++) S[i] = std::rand()%DIM;

  // LVQS
  for (int i=0; i<NUMITER; i++){
    lvqs(S);
    X[i] = k;
    std::cout << "Numero iterazioni run " << i+1 << ": " << k << std::endl;
    k = 0;
  }
  std::cout << std::endl;

  // Calcolo valore medio
  double medio = accumulate(X.begin(),X.end(),0.)/NUMITER;
  std::cout << "Valore medio: " << medio << std::endl;

  // Calcolo varianza
  double somma = 0;
  for(int i=0; i<NUMITER; i++)
    somma += pow(X[i] - medio, 2);
  double var = somma/(NUMITER-1);
  std::cout << "Varianza: " << var << std::endl;


  // Stampa su file
  std::ofstream f("confronti.txt");
  if (f.is_open())
  {
    for(int i=0; i<NUMITER; ++i)
        f << X[i] << "\n";
    f.close();
  }
  else std::cout << "Errore nell'apertura del file";
  return 0;
}
```

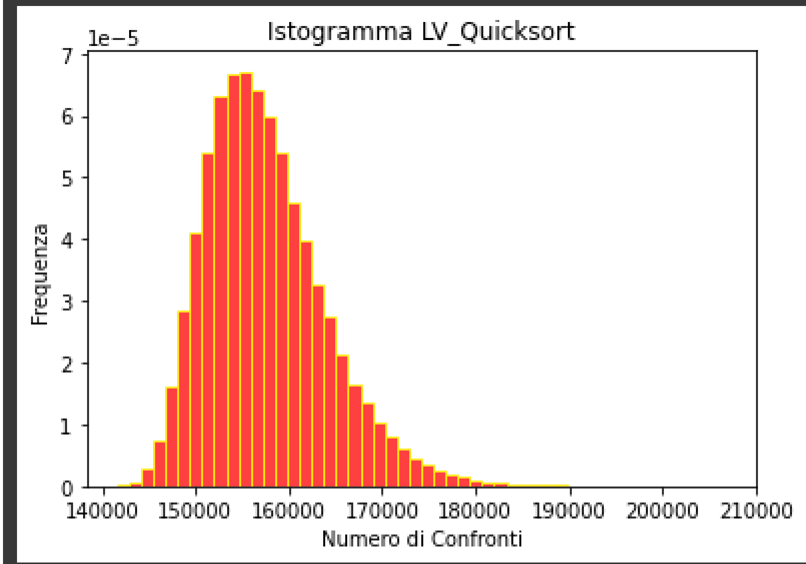
Valore medio: 157516

Varianza: 4.19369e+07

Di seguito il codice in Python

```
import matplotlib.pyplot as plt # libreria per i grafici
from scipy.stats import norm
import numpy as np

# Elaboro il file in input
confronti = "/content/confronti.txt"
myfile = open(confronti, "r")
FileContent = [int(k) for k in myfile.readlines()]

medio = sum(FileContent) / len(FileContent) # valore atteso (o medio)
somma = 0
for i in range(len(FileContent)):
  somma += pow(FileContent[i] - medio, 2)
var = somma/(len(FileContent)-1) # varianza
print("Valore medio: " + str(medio))
print("Varianza: " + str(var))

# Creazione istogramma
# larghezza bins = (maxval - minval) / 50, i valori vengono raggruppati per range
plt.hist(FileContent, bins = 50, density = True, edgecolor="yellow", color ='red',
alpha = 0.75)
plt.title("Istogramma LV_Quicksort")
plt.xlabel("Numero di Confronti")
plt.ylabel("Frequenza")

# Limitazione dall'alto con disuguaglianza di Markov (a = vu e v = 2 (doppio), dove
 u = valore atteso E (medio))
Mk2 = 1/2
# Limitazione dall'alto con disuguaglianza di Markov (a = vu e v = 3 (triplo))
Mk3 = 1/3
# Limitazione dall'alto con disuguaglianza di Chebyshev (e = (v-1)u e v = 2)
Ch2 = var / (4 * medio**2)
# Limitazione dall'alto con disuguaglianza di Chebyshev (e = (v-1)u e v = 3)
Ch3 = var / (9 * medio**2)

print("Disuguaglianza di Markov con v = 2: " + str(Mk2))
print("Disuguaglianza di Markov con v = 3: " + str(Mk3))
print("Disuguaglianza di Chebyshev con v = 2: " + str(Ch2))
print("Disuguaglianza di Chebyshev con v = 3: " + str(Ch3))

# Calcolo della frequenza empirica di ottenimento del doppio e triplo del valore at
teso
f2 = f3 = 0
for e in FileContent:
  if e >= (2 * medio): f2+=1
  if e >= (3 * medio): f3+=1
print("Frequenza empirica di ottenimento del doppio di E: " + str(f2))
print("Frequenza empirica di ottenimento del triplo di E: " + str(f3))
myfile.close()
print("\n")
plt.show()
```

```
Valore medio: 157515.8354
Varianza: 41936856.827294536
Disuguaglianza di Markov con v = 2: 0.5
Disuguaglianza di Markov con v = 3: 0.3333333333333333
Disuguaglianza di Chebyshev con v = 2: 0.00042255908540542555
Disuguaglianza di Chebyshev con v = 3: 0.00018780403795796692
Frequenza empirica di ottenimento del doppio di E: 0
Frequenza empirica di ottenimento del triplo di E: 0
```



Il grafico mostra una distribuzione somigliante ad una Gaussiana; si osserva che al crescere del numero di run, il numero medio di operazioni eseguite tende all’espettazione del QuickSort nel caso medio, cioè nlog(n) che per n = 10^4 tende a 132877 confronti. Dal grafico si desume inoltre la difficoltà nel raggiungere valori doppi o tripli per il valore di espettazione, per i quali infatti la frequenza si attesta a zero.
Il maggior numero di operazioni lo si osserva nell’intervallo compreso fra le 150.000 e le 160.000 operazioni, con una deviazione attorno alle 6476 operazioni fra una run e l’altra.
A differenza della disuguaglianza di Markov, si nota una maggiore precisione nel limite di maggiorazione dato dalla disuguaglianza di Chebyshev, conoscendo a priori la varianza.