Filippo Manini Matricola: 4798004

Relazione Monte Carlo MinCut

TESTO:
Compito 2.1. Implementazione di MCMinCut Genera il grafo di Fritsch in Figura (4) e verifica la frequenza empirica con la quale ottieni un taglio minimo applicando *MCMinCut* 10^5 volte.
Confronta il risultato ottenuto con la stima p ≈ 2/n^2.

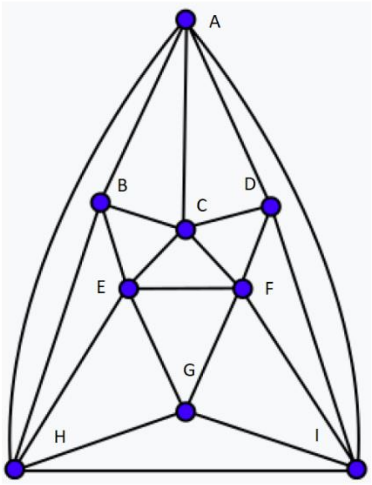

Figura 4: Il grafo di Fritsch ha n = 9 vertici e 21 archi

ESECUZIONE:
g++ -std=c++17 MCMinCut.cpp

FUNZIONAMENTO:

la **struttura** è così composta:

```
typedef string Label;

struct vertexNode
{
    Label label;
    bool visited;
    vector <Label> adjList;

}typedef Graph;
```

Input: G, un multigrafo connesso e non orientato con n vertici

- all'interno del ciclo for che parte dal numero di vertici, si continua decrementando fino ad arrivare a 2
- imposto il vertice da cui partire; passando un vertice casuale tramite la funzione *verticeCasuale(…)*
- per ogni ciclo del for eseguo la funzione taglioMinimo(…) dove controllo che i vertici esistano realmente
- setto a true il vertice come visitato tramite la funzione *setVerticeVisitato(..)*
- controllo se tutti i vertici adiacenti di quel vertice sono visitati tramite un ciclo while all'interno della funzione *verticeNonVisistato(…)* che controlla se tutta la lista di adiacenza è stata visitata tramite la funzione *listaDiAdiacenzaVisistata(…)*
    - finché tutta la lista di adiacenza non è stata visitata a ogni ciclo ottengo un mezzo arco e se non è stato visitato lo restituisco.
- confronto il Label da cancellare con il vertice casuale non ancora visitato e se il vertice casuale e' diverso dal vertice da cancellare rieseguo il taglioMinimo altrimenti eseguire la funzione *taglio(…)* e setto i vertici come non visitati tramite la funzione *setNonVisitato(..)*
- posso tagliare tramite la funzione *taglio(…)*
    - mi salvo la lista di adiacenza del vertice da eliminare
    - cancello archi e vertici da eliminare tramite la funzione *cancella(…)*
    - unisco la lista di adiacenza del vertice che vado a tagliare al nuovo vertice tramite la funzione *unisciArci(…)*
    - infine rinomino la nuova label tramite la funzione *rinominoLabel(…)*

A ogni iterazione la cardinalità dell'insieme dei vertici del multigrafo diminuisce di un'unità mentre il grado del nuovo vertice non è mai inferiore al grado dei vertici precedenti.

esempio di un primo ciclo

```
---------------------------------
vCasuale: D
DOPO TAGLIO:
numVertice: 8
numArchi: 40
A (0): G; I; B; CE; D; G; I; B; CE; D;
B (0): A; CE; G; A; CE; G; CE; CE;
CE (0): A; B; D; F; A; B; D; F; B; F; G; H; B; F; G; H
;
D (0): A; CE; I; F; A; CE; I; F;
F (0): CE; D; H; I; CE; D; H; I; CE; CE;
G (0): A; B; H; I; A; B; H; I; CE; CE;
H (0): F; G; I; F; G; I; CE; CE;
I (0): A; D; F; G; H; A; D; F; G; H;
---------------------------------
```

Terminato il ciclo for ottengo cosi il taglio minimo.
Proprio per la sua natura randomica Monte Carlo MinCut per ottenere dei dati interessanti bisogna eseguire l'intero procedimento spiegato in precedenza per un numero soddisfacente di volte come 10^5 e per ognuna ricreare il Grafo Fritsch tramite la funzione *creazioneGrafoFritsch(…)*
Si salva poi in un vettore il MinCut ritornato per ogni run.
terminato anche quest'ultimo ciclo posso così calcolare la stima p calcolata come: $2/n^2$
e confrontarla con la frequenza empirica utilizzando la funzione ausiliaria *frequenza(…)* che viene divisa per il #run cosicché si calcola il numero di volte che ho ottenuto un MinCut conoscendo però questa volta il minimo MinCut dopo #run.
noto che effettivamente la stima *p* si avvicina alla frequenza empirica del taglio minimo.

Se eseguiamo MCMinCut per m volte, dalla formula per la probabilità p(m) di non ottenere un taglio minimo in m run otteniamo:

$$p(m) < \left(1 - \frac{2}{n^2}\right)^{m-1}$$

Che, al crescere di *m,* si avvicina a 0.

Output finale:

```
---------------------------------
Grafo Fritsch iniziale:
numVertice: 9
numArchi: 21
A (0): G; I; B; C; D;
B (0): A; C; G; E;
C (0): A; B; D; E; F;
D (0): A; C; I; F;
E (0): B; C; F; G; H;
F (0): C; D; E; H; I;
G (0): A; B; E; H; I;
H (0): E; F; G; I;
I (0): A; D; F; G; H;
---------------------------------
MinCut del grafo dopo 100000 run e' 4 archi
stima p (2/n^2): 0.0246914
Frequenza empirica del taglio minimo: 0.02
---------------------------------
```