Implementação 2 - Grafos

Samuel Correia , Vínicius Ferrer , Daniel Salgado , Arthur Martinho

17/08/2024

1 Gerador de todos os subgrafos de um grafo em C++

```cpp
#include <iostream>
#include <vector>
#include <cmath>

using namespace std;

// Funcao para imprimir o conjunto de vertices
void printVertices(const vector<int>& vertices) {
    cout << "Vertices: { ";
    for (size_t i = 0; i < vertices.size(); ++i) {
        cout << vertices[i];
        if (i < vertices.size() - 1) cout << ", ";
    }
    cout << " }";
}

// Funcao para imprimir o conjunto de arestas
void printArestas(const vector<pair<int, int>>& edges) {
    if (edges.empty()) {
        cout << " Arestas: { }";
        return;
    }
    cout << " Arestas: { ";
    for (size_t i = 0; i < edges.size(); ++i) {
        cout << "(" << edges[i].first << ", " << edges[i].second << ")";
        if (i < edges.size() - 1) cout << ", ";
    }
    cout << " }";
}

// Funcao para gerar todos os subgrafos
void gerarSubgrafos(int n) {
    int totalSubgrafos = 0;
    int vertexSubsets = pow(2, n);

    // Itera por todos os subconjuntos de vertices (excluindo o conjunto vazio)
    for (int vertice_masc = 1; vertice_masc < vertexSubsets; ++vertice_masc) {
        vector<int> vertices;

        // Identifica quais vertices estao no subconjunto atual
        for (int i = 0; i < n; ++i) {
            if (vertice_masc & (1 << i)) {
                vertices.push_back(i + 1);
            }
        }

        int k = vertices.size();
        int totalArestasPossivel = k * (k - 1) / 2;
        int totalEdgeSubsets = pow(2, totalArestasPossivel);

        // Mapeia cada par de vertices para uma posicao para geracao de arestas
        vector<pair<int, int>> possiveisAresta;
        for (int i = 0; i < k; ++i) {
            for (int j = i + 1; j < k; ++j) {
```

```
                possiveisAresta.push_back({vertices[i], vertices[j]});
            }
        }

        // Itera por todos os subconjuntos de arestas
        for (int e_mask = 0; e_mask < totalEdgeSubsets; ++e_mask) {
            vector<pair<int, int>> edges;

            for (int idx = 0; idx < totalArestasPossivel; ++idx) {
                if (e_mask & (1 << idx)) {
                    edges.push_back(possiveisAresta[idx]);
                }
            }

            // Exibe o subgrafo atual
            printVertices(vertices);
            printArestas(edges);
            cout << endl;
            totalSubgrafos++;
        }
    }

    cout << "\nNumero total de subgrafos gerados: " << totalSubgrafos << endl;
}

int main() {
    int n;

    cout << "Informe o numero de vertices do grafo completo: ";
    cin >> n;

    if (n <= 0) {
        cout << "O numero de vertices deve ser maior que 0." << endl;
        return 1;
    }

    gerarSubgrafos(n);

    return 0;
}
```

Listing 1: Exemplo C++

1.1 Como Funciona

Função gerarSubgrafos: Iteração sobre subconjuntos de vértices: Usa uma máscara de bits para gerar todos os subconjuntos possíveis de vértices. Determinando vértices e arestas: Para cada subconjunto de vértices, determina quais arestas existem entre esses vértices,Após isso imprime cada subgrafo e faz a contagem total.

Função printArestas: Esta função recebe um vetor de pares que representam arestas entre vértices. for loop: Itera sobre todas as arestas armazenadas no vetor e as exibe na forma (vértice1, vértice2). cout: Exibe a lista de arestas como um conjunto.

Função printVertices: Imprime os vértices de um subgrafo, Ela recebe como parâmetro uma referência constante para um vetor de inteiros chamado vertices, que contém os números dos vértices que compõem o subgrafo atual.

Função main: Demonstra o uso das funções

1.2 Teste do Codigo

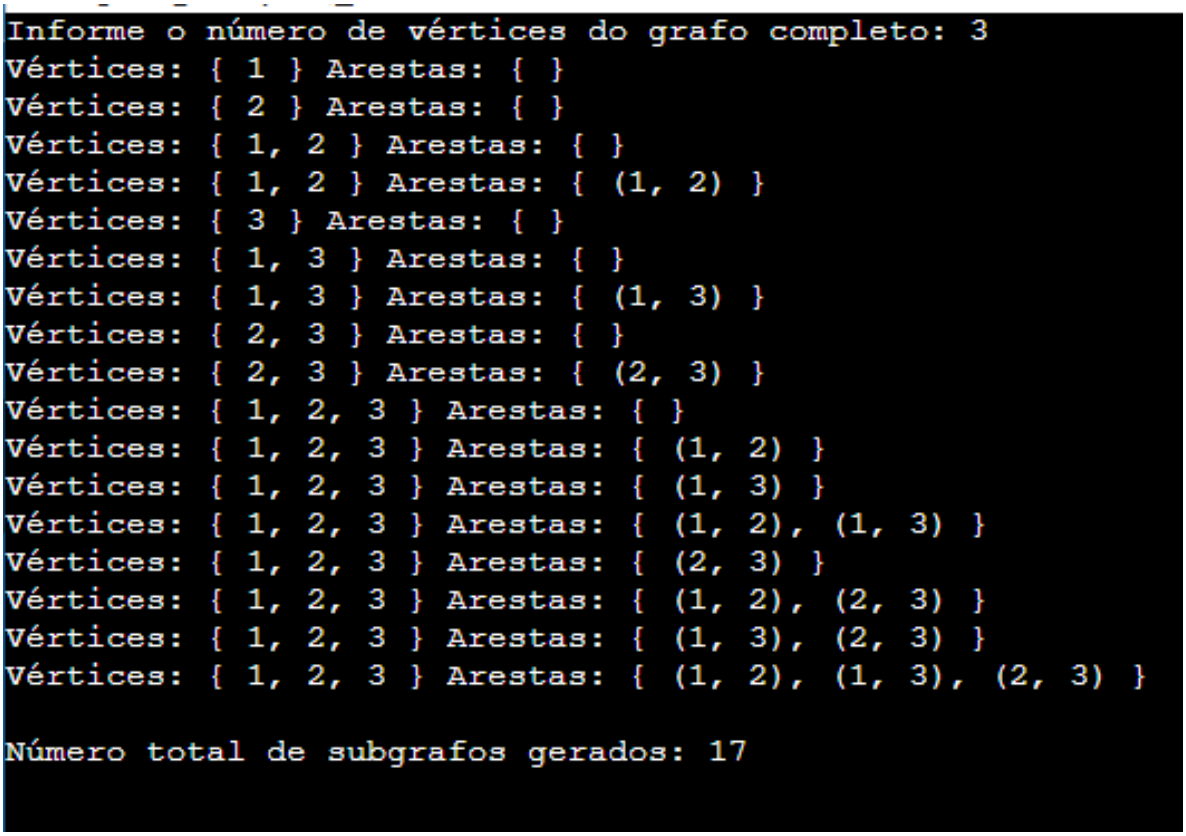

Figure 1: Teste Gerador Subgrafos