Algoritmo para verificar existência de circuito euleriano em grafo não direcionado

Lucas Guimarães Batista
10 de maio de 2021

1 INTRODUÇÃO

Queremos definir se um dado grafo não direcionado possui um circuito euleriano. Consideremos um grafo não direcionado G. Um circuito euleriano de G é um passeio fechado de G tal que toda aresta de G aparece exatamente uma vez nesse passeio. Caso exista tal passeio, podemos dizer que G possui um circuito euleriano. Com a utilização de alguns argumentos teóricos aliados a algoritmos, podemos escrever um programa que define se em um grafo não direcionado dado como entrada existe circuito ou caminho euleriano e que caminho é esse.

2 CONCEITOS UTILIZADOS

Primeiramente, a forma de representação interna escolhida para o grafo de entrada é por meio de uma lista de adjacências. Assim como Euler, utilizamos conceitos de conectividade em grafos e paridade que foram utilizados para formular originalmente a resolução do problema das pontes de Königsberg. Além disso foram empregados algoritmos auxiliares como o de busca em profundidade, utilizado para percorrer o grafo, quando necessário, de forma ótima. Na hora de fazer a impressão do caminho euleriano, foi feita uma implementação baseada no algoritmo de Fleury.

3 IMPLEMENTAÇÃO

3.1 INTERPRETANDO E ARMAZENANDO ENTRADA

A primeira parte do programa deve tratar essa entrada de forma que possamos manipular seus elementos. Para isso foi criado um *parser* simples, que analisa cada linha do arquivo de entrada. Ele ignora espaços em branco e divide a linha ao meio utilizando o símbolo "="(igual) como separador. O lado esquerdo é definido como vértice da vez e ligado a cada vértice do lado direito antes de prosseguir para a linha seguinte. Essas informações são guardadas em uma lista com tamanho igual à quantidade de vértices do grafo de entrada. Cada entrada i da lista é também uma lista que contém o índice de cada vértice da vizinhança de i.

3.2 MANIPULANDO O GRAFO

Depois de interpretado o arquivo de entrada, o programa realiza uma sequência de testes das propriedades do grafo passado para determinar se ele é euleriano, semi-euleriano ou não euleriano. Primeiro é feito um teste de conectividade com auxílio de um algoritmo de busca em profundidade. É feita em sequência a checagem da classificação do grafo quanto a sua qualidade euleriana. Essa checagem é baseada em paridade de vértices e será explicada adiante. Para que ela funcione, computamos quantos e quais são os vértices ímpares do grafo.

3.3 INTERPRETANDO O RESULTADO

A saída do programa é dada em forma de uma afirmação sobre a qualidade do grafo de entrada. Se no teste de conectividade for detectado que o grafo recebido é desconexo, obteremos como saída que o grafo é **não euleriano**. Caso conexo, sua qualidade será dada em função da quantidade de vértices ímpares. Caso não haja vértice de grau ímpar, afimamos que no grafo **há circuito euleriano**. Caso haja dois vértices de grau ímpar, **há caminho euleriano**. Caso haja mais de dois vértices ímpares, o grafo é **não euleriano**.

3.4 IMPRIMINDO O CAMINHO

Depois de definir que existe um caminho, precisamos definir qual é esse caminho. O algoritmo de Fleury nos permite definí-lo facilmente. Partimos de um vértice de grau ímpar quando possível e sempre que encontrarmos um vértice com mais de uma aresta, escolhemos sempre aquela aresta que não é ponte, ou seja, que não torna o grafo desconexo ao ser removida. Ao realizar esse procedimento após definir que o grafo é euleriano, sempre teremos como saída um caminho/ciclo válido.

4 BIBLIOGRAFIA

1. Complexidade de Algoritmos I - Prof. Pedro J. de Rezende
2. Rangel, Socorro. Teoria do Grafos, Notas de aula, IBILCE, Unesp, 2002-2013.
3. Fleury's Algorithm, C Implementation