Reconhecimento de um grafo cordal

Nome: Matheus da Silva Oliveira
DRE: 118178020

A entrada do algoritmo é um grafo $G = (V, E)$ conexo, não orientado e sem pesos.

O algoritmo constrói um grafo a partir do arquivo *grafo.txt*. Na construção do grafo, o algoritmo checa se o grafo em questão é não orientado e se possui vértices com IDs inteiros maiores do que 0. Estamos supondo na entrada do algoritmo que o grafo não possui pesos nas arestas, então não há nenhum teste para checar isto. O critério para saber se o grafo é orientado é o seguinte:

Um grafo G é não orientado se, e somente se, ∀ vértice $v_i \in V$, se v_j é vizinho de v_i, então v_i deve ser vizinho de v_j.

Em outras palavras, podemos dizer que se um grafo em questão é não direcionado, então se existe a aresta (v_i, v_j), então deve existir a aresta (v_j, v_i).

Basicamente, para saber se um dado grafo é cordal, primeiro computamos uma busca em largura lexicográfica (lexBFS) para descobrirmos um possível esquema de eliminação perfeita (EEP) do grafo. A seguir, checamos para cada vértice v_i na ordem inversa da lista retornada pelo método lexBFS se é simplicial. Se for, retiramos este vértice do grafo e analisamos o próximo na lista. Se todos forem simpliciais, o grafo se torna vazio, ou seja, não possui vértices nem arestas, então o grafo é cordal. Se isso não ocorrer, o grafo não é cordal. O teorema abaixo nos permite utilizar o resultado acima:

Teorema:
Um grafo G é cordal, se, e somente se, possui um EEP

No algoritmo lexBFS, testamos no início se o grafo é conexo. Para isso, utilizamos um algoritmo de busca em largura modificado para reconhecer se o dado grafo é conexo ou não. Caso não seja conexo, o grafo é inválido e terminamos. Caso seja conexo, o algoritmo lexBFS é executado. Caso o lexBFS seja executado, precisamos saber a seguir se cada vértice v_i é simplicial no possível EEP. Dessa forma, checamos se $N(v_i)$ induz uma clique. Caso induza, retiramos v_i do grafo, pois se G é cordal, continuaremos a possuir um grafo cordal, ou seja, $G \setminus \{v_i\}$ será cordal. O lema abaixo nos concede este resultado

Lema:
Se G é cordal, $|V| > 1$, então $G \setminus \{v_i\}$ é cordal.

Se $N(v_i)$ não induz uma clique, então mostramos o vértice v_i e o subgrafo restante que não é cordal, ou seja, o subgrafo sem retirar v_i, já que G não é cordal. O critério para saber se $N(v_i)$ induz uma clique é o seguinte:

$N(v_i)$ *induz uma clique se* $\forall$ v_j *e* v_k *vizinhos de* v_i, v_j *e* v_k *são vizinhos, onde* $v_i \neq v_j$.

Os detalhes da implementação podem ser vistos nos comentários do próprio código fonte. Além disso, foi utilizado um arquivo makefile para facilitar a compilação, então basta rodar o comando *make* no terminal para o algoritmo compilar e rodar automaticamente.

A classe ***Grafo*** é responsável por gerar o grafo que será analisado através dos IDs dos vértices, a classe ***Vértice*** é utilizada para guardar as informações de um vértice, por exemplo quem são seus vizinhos e seu ID. O enum ***Cor*** é utilizado para facilitar a implementação da busca em largura modificada. A classe ***GrafoInvalidoException*** é utilizada para o tratamento de grafos inválidos para o problema em questão.

Referências

1. *Livro: Teoria computacional de grafos: os Algoritmos*
2. *Livro do Cormen*
3. *Lexicographic breadth-first search - Wikipedia*
4. *Day 7 - Busca em Largura Lexicográfica*
5. *Day 16 - Reconhecimento de grafos cordais - Aula Completa*
6. *Lexicographic Breadth First Search*
7. *Clique – Wikipédia, a enciclopédia livre*
8. *Relatório de Resumo das aulas Aula 02*
9. *Chordal graph - Wikipedia*
10. *Conectividade (teoria dos grafos) – Wikipédia, a enciclopédia livre*
11. *UNIVERSIDADE FEDERAL DO PARANÁ MATHEUS VINICIUS CORREA BUSCA EM LARGURA LEXICOGRÁFICA E ALGORITMOS DE SOLUÇÃO EXATA PARA OP*