Metodologia Ativa – 06/11/2019 – Pesquisa e debate

Encontre respostas para os itens a seguir.

1. O que é um clique em um grafo? É um componente de um grafo onde todos os vértices formam um grafo completo, adjacentes entre si.
2. O que é um clique maximal? É um clique que não é sub-clique de nenhum outro clique no grafo.
3. O que é um clique máximo? É o clique dentro de um grafo com o maior número de elementos.
4. Um clique máximo é maximal e vice-versa? Explique.

Sim, um clique máximo é maximal, mas a recíproca não é verdadeira. Pois um clique máximo é o maior clique do grafo, e um clique maximal não é necessariamente, o clique máximo do grafo pois pode haver vários cliques maximais em um grafo.

1. Escreva o pseudo-código de um algoritmo que verifica se um conjunto de vértices forma um clique

\\ Função para grafos não-direcionados.

boolean checarSeHáClique(Conjunto conjunto){

para i de 0 até conjunto.length-1 passo 1

int[] contador = new int[2], lock = 0;

para j de 0 até conjunto.length-1 passo 1

para k de 0 até conjunto.length-1 passo 1 se i != j e k != j e k != i

se existe adjacência entre i e j

contador[0] = j;

lock++;

fim\_se

se existe adjacência entre i e k

contador[1] = k;

lock++;

fim\_se

fim\_se

se lock == 2

se existe adjacência entre contador[0] e contador[1]

retorne true;

senão

lock = 0;

contador = new int[2];

fim\_se

senão

lock = 0;

contador = new int[2];

fim\_se

fim\_para

fim\_para

fim\_para

retorne false;

}

1. Escreva o pseudo-código de um algoritmo que verifica se um conjunto de vértices forma um clique maximal

boolean checarSeHáCliqueMaximal(Conjunto conjunto){

Lista lista = new Lista;

para i de 0 até conjunto.length-1 passo 1

int[] contador = new int[2], lock = 0;

para j de 0 até conjunto.length-1 passo 1

para k de 0 até conjunto.length-1 passo 1 se i != j e k != j e k != i

se existe adjacência entre i e j

contador[0] = j;

lock++;

fim\_se

se existe adjacência entre i e k

contador[1] = k;

lock++;

fim\_se

fim\_se

se lock == 2

se existe adjacência entre contador[0] e contador[1]

lista.adicionar(contador);

contador = new int[2];

fim\_se

fim\_para

fim\_para

fim\_para

Lista c\_maximal = new Lista;

para i de 0 até lista .length-1 passo 1

para j de 0 até lista.length-1 passo 1

se i != j

se lista(i) não é subset de lista(j)

c\_maximal.adicionar(lista(i));

fim\_se

fim\_se

fim\_para

fim\_para

retorne (c\_maximal.length == 0);

}