MAC0323 ALGORITMOS E ESTRUTURAS DE DADOS II

FOLHA DE SOLUÇÃO

Nome: Leonardo Heidi Almeida Murakami **NUSP:** 11260186

Exercício: Exercício Teórico II - MAC0323

SOLUÇÃO

1. Exercício 2

1.1. Algoritmo Proposto.

(1) **Escolha de um vértice raiz arbitrário:** Selecione um vértice arbitrário $s \in V$. Este vértice será o último na ordenação de remoção, ou seja, $x_n = s$.
(2) **Busca em Profundidade (DFS) e Pós-Ordem:** Realize uma DFS em G iniciando pelo vértice s. Durante a travessia, compute uma ordenação de pós-ordem (post-order traversal) dos vértices. Denotaremos esta sequência como $P_1, P_2, \ldots, P_n$. Pela definição de pós-ordem em uma DFS iniciada em s, temos $P_n = s$.
(3) **Definição da ordenação de remoção:** A ordenação de remoção desejada $x_1, x_2, \ldots, x_n$ é definida diretamente pela sequência de pós-ordem: $x_i = P_i$ para $i = 1, \ldots, n$.

1.2. **Demonstração.** Precisamos demonstrar que para a ordenação $x_1, \ldots, x_n$ gerada, os grafos $G_k = G - \{x_1, \ldots, x_k\}$ são conexos para $k = 0, 1, \ldots, n-1$. O grafo G_k é o subgrafo de G induzido pelo conjunto de vértices $V'_k = V \setminus \{x_1, \ldots, x_k\}$. Utilizando a definição $x_i = P_i$, temos $V'_k = \{P_{k+1}, \ldots, P_n\}$.

- Para $k = 0$, $G_0 = G[V \setminus \emptyset] = G[V] = G$. O problema estipula que G é conexo, então a condição é satisfeita para G_0.
- Para $k \in \{1, \ldots, n-1\}$, consideremos o grafo $G_k = G[V'_k]$. Seja T a árvore DFS gerada pela busca em profundidade iniciada em $s = P_n$.

  Para qualquer vértice $v \in V'_k$:
  - Se $v = s$ (ou seja, $v = P_n$), ele é a raiz da árvore DFS T e pertence a V'_k.
  - Se $v \neq s$, então $v = P_j$ para algum $j \in \{k+1, \ldots, n-1\}$. Na árvore DFS T, v possui um pai, que denotaremos pai(v). De acordo com as propriedades da travessia em pós-ordem, um nó é registrado na sequência de pós-ordem somente após todos os seus descendentes na árvore DFS terem sido registrados. Isso implica que o índice de pós-ordem de qualquer nó é menor que o índice de pós-ordem de seu pai. Assim, se $v = P_j$, seu pai pai(v) será P_l para algum $l > j$. Como $j \geq k+1$, segue que $l > j \geq k+1$, e portanto $l \in \{k+1, \ldots, n\}$. Logo, pai$(v) \in V'_k$.

  Isso demonstra que cada vértice $v \in V'_k \setminus \{s\}$ está conectado ao seu pai pai(v) (que também está em V'_k) através de uma aresta da árvore T. Como $s = P_n$ é a raiz de T e $s \in V'_k$, todos os vértices em V'_k estão conectados a s dentro de $G[V'_k]$ por meio de

Date: 13 de Abril de 2025.

caminhos formados por arestas de T (que são um subconjunto das arestas de $G[V'_k]$). Consequentemente, o grafo $G[V'_k]$ é conexo.

Esta argumentação é válida para $k = 1, \ldots, n-1$. Note que para $k = n-1$, $V'_{n-1} = \{P_n\} = \{s\}$. O grafo $G[\{s\}]$ consiste em um único vértice e é, por definição, conexo.

Portanto, o algoritmo produz uma ordenação que satisfaz as condições do problema.

1.3. **Análise da Eficiência.** A complexidade do algoritmo é determinada pelas suas etapas principais:

(1) **Escolha de um vértice raiz arbitrário:** A seleção de um vértice arbitrário pode ser feita em $O(1)$ (assumindo que os vértices são acessíveis de forma eficiente, por exemplo, como o primeiro elemento de uma lista de adjacência ou através de um índice).
(2) **Busca em Profundidade (DFS) e Pós-Ordem:** Uma DFS padrão em um grafo com n vértices e m arestas tem complexidade $O(n+m)$. A computação da sequência de pós-ordem é uma parte natural da DFS (um vértice é adicionado à lista de pós-ordem quando a chamada recursiva da DFS para esse vértice é concluída) e não adiciona sobrecarga significativa à complexidade da DFS.
(3) **Definição da ordenação de remoção:** Atribuir a sequência de pós-ordem $P_1, \ldots, P_n$ à ordenação $x_1, \ldots, x_n$ envolve a manipulação de uma lista de n vértices, o que leva $O(n)$ tempo.

Somando as complexidades das etapas, a complexidade total do algoritmo é dominada pela etapa da DFS, resultando em $O(n+m)$.