**MC448: Projeto e Análise de Algoritmos I**

Prof. Cid C. de Souza – 3ª Prova – (30/06/2009)

**Nome:**

**RA:** **Turma:**

| Questão | frac | Peso | Nota |
|---|---|---|---|
| 1 | | | |
| 2 | | | |
| 3 | | | |
| 4 | | | |
| Total | | 10,0 | |

**Observação:** o peso das questões será decidido pelo docente da seguinte forma: as duas questões que você responder melhor terão peso 3 e as demais terão peso 2. Portanto, uma mesma questão pode ter peso 2 para um aluno e peso 3 para um outro aluno.

**Instruções:**

1. A duração da prova é de 110 minutos.
2. Não é permitido usar qualquer material de consulta durante a prova.
3. **Questões mal justificadas serão consideradas erradas !**
4. As provas podem ser feitas a lápis porém, neste caso, ficará a critério do docente aceitar ou não eventuais pedidos de revisão de nota.
5. Use as folhas de papel almaço para responder as questões.
6. Coloque o seu nome, RA e turma em **todas as folhas de papel almaço**.
7. O uso de calculadoras, **aparelhos celulares** e *pagers* está proibido durante a prova.
8. Não desgrampeie o caderno de questões.
9. Você pode usar um algoritmo $A$ visto em sala de aula como **subrotina** de algoritmos que você esteja projetando para responder a uma questão da prova **exceto**, é claro, se o que for pedido for na questão for a própria descrição do algoritmo $A$.
10. Em todas as questões, se nada for dito em contrário, o grafo passado na entrada é dado por suas *listas de adjacência*.
11. **Nenhum aluno poderá deixar a sala de prova antes de terem transcorridos 60 minutos desde o início da mesma.**
12. **O aluno que deixar a sala por qualquer motivo deverá entregar a prova em definitivo, ou seja, não poderá mais retornar**.

**1.** Seja $G = (V, E)$ um grafo **orientado** e $w : E \to \mathbb{R}$ uma função que atribui pesos aos arcos de $E$, sendo alguns deles **negativos**. Seja $s$ um vértice de $V$. Responda os itens a seguir:

(a) Defina o que seria um **ciclo negativo** em $G$.

(b) Suponha que $G$ não tenha ciclos negativos. Mostre através de um exemplo que, mesmo nesse caso, o algoritmo de Dijktra não calcula corretamente os caminhos mínimos a partir de $s$. Em sua resposta você deve desenhar o grafo $G$ identificando: o pesos dos arcos, o vértice $s$ e os comprimentos dos caminhos mínimos computados pelo algoritmo de Dijkstra para esta instância.

(c) Para o caso em que $G$ não tem ciclos negativos, Mané sugeriu um algoritmo para encontrar caminhos mínimos de $s$ para todos os vértices de $G$ cujos passos são resumidos a seguir: (1) Seja $k = 1 + \max\{e \in E : w_e\}$; (2) Para todo arco $e$ em $E$, atribua o custo $\overline{w}_e = w_e + k$; (3) com os pesos $\overline{w}$ associados aos arcos de $G$, use o algoritmo de Dijkstra para calcular caminhos mais curtos a partir de $s$; (4) Retorne os caminhos encontrados no passo anterior.

Prove que este algoritmo está correto ou dê um contra-exemplo mostrando que ele está errado.

**2.** Seja $G = (V, E)$ um **grafo direcionado acíclico** (DAG). Dado um vértice $u$ em $V$, define-se o *grau de entrada de* $u$, denotado por $ge(u)$, como sendo o número de arcos que entram em $u$. Responda os itens a seguir:

(a) Explique o que é uma **ordenação topológica** de $G$.

(b) Mostre que, por ser um DAG, $G$ sempre tem um vértice $u$ tal que $ge(u) = 0$.

(c) Sendo o grafo $G$ dado por suas listas de adjacência, dê um pseudo-código em alto nível de um algoritmo que calcula $gs(u)$ para todo $u \in V$ em tempo linear no tamanho do grafo. (*Nota:* não esqueça de mostrar que o seu algoritmo tem a complexidade exigida no enunciado).

(d) Uma forma de computar uma **ordenação topológica** é usar recursivamente o resultado do item (b) acima. Os passos básicos deste algoritmo seriam: (1) calcular $ge(u)$ para todo $u \in V$ (conforme o item (c)); (2) encontrar um vértice $u$ com $ge(u) = 0$; (3) imprimir $u$; (4) "remover" $u$ e todos os arcos saindo de $u$ de $G$; (5) se ainda houver vértices em $G$, voltar ao passo (2). *[Nota: a "remoção" do passo (4) não precisa ser executada de fato. Por quê ?]*

Dê um pseudo-código que implementa o algoritmo acima com uma complexidade $O(|V|+|E|)$. O nível de detalhamento do seu algoritmo deve ser suficiente para que você consiga explicar porque a complexidade desejada foi alcançada.

(e) Suponha que **não** se saiba de antemão se o grafo $G$ passado na entrada do algoritmo do item anterior é acíclico. Como o algoritmo poderia ser adaptado para identificar a presença de um ciclo sem ter sua complexidade alterada ?

**3.** A seguir encontra-se o pseudo-código do algoritmo de Floyd-Warshall dado em aula para calcular o comprimento dos caminhos mínimos entre todos os pares de vértices de um grafo **orientado** com pesos nos arcos com sinais arbitrários. Neste algoritmo, supõe-se os vértices rotulados de 1 a $|V|$ (sem repetições). Responda os itens a seguir.

**Floyd-Warshall($d$)**
1. inicializar `dist` e `pred`;
2. **para** $k$ **de** 1 **até** $n$ **faça**
3.   **para** $i$ **de** 1 **até** $n$ **faça**
4.     **para** $j$ **de** 1 **até** $n$ **faça**
5.       **se** $\texttt{dist}[i,j] > \texttt{dist}[i,k] + \texttt{dist}[k,j]$ **então**
6.         $\texttt{dist}[i,j] \leftarrow \texttt{dist}[i,k] + \texttt{dist}[k,j]$
7.         $\texttt{pred}[i,j] \leftarrow \texttt{pred}[k,j]$
8. **retorne** $(\texttt{dist}, \texttt{pred})$.

(a) diga como são feitas as inicializações na linha 1 do algoritmo e complete as informações abaixo.

"O algoritmo computa duas matrizes, cujas células na iteração $k$ ao final do laço da linha 2 contém as seguintes informações:
• $\texttt{dist}[i,j]$**:** ... (complete na folha de respostas);
• $\texttt{pred}[i,j]$**:** ... (complete na folha de respostas);"

(b) O algoritmo de Floyd-Warshal é um exemplo de uso de programação dinâmica no projeto de algoritmos. Este paradigma tem como uma de suas principais características a existência de uma **fórmula de recorrência** que evidencia a existência de subestrutura ótima no problema. Denotando por $\texttt{dist}[i,j]^k$ o valor da célula $[i,j]$ de `dist` ao final da $k$-ésima execução do laço da linha 2, escreva qual a fórmula de recorrência que norteia o algoritmo de Floyd-Warshall.

(c) Foi dito em sala de aula que a partir das matrizes de saída do algoritmo de Floyd-Warshall é possível determinar a existência de um ciclo negativo no grafo de entrada, caso exista algum.

Usando um pseudo-código de alto-nível, descreva um algoritmo de complexidade polinomial no tamanho do grafo de entrada que encontra **um** ciclo negativo em $G$ caso ele exista. Faça a análise de complexidade do seu algoritmo. Lembre-se que o seu código deve ser suficientemente detalhado (inclusive a estrutura de dados) para que se possa compreender a análise.

**4.** Seja $G = V, E)$ um grafo **orientado** dado por suas listas de adjacência. Considere os algoritmos de busca em grafos **vistos em aula** e responda os itens abaixo **justificando** as suas respostas.

(a) Seja $r$ um vértice de $G$ e $T$ uma árvore contendo caminhos entre $r$ e todos os vértices alcançáveis a partir de $r$ em $G$. Agora, para todo vértice $u$ em $T$, seja $P_{ru}$ o (único) caminho de $r$ para $u$ em $T$. Se nenhum outro caminho de $r$ para $u$ em $G$ tem menos arcos que $P_{ru}$, diz-se que $T$ satisfaz à *propriedade dos camínhos mínimos* (PCM) para o vértice $r$.

Sabe-se que toda árvore BFS construída por uma busca em **largura** tendo o vértice $r$ como raiz satisfaz à PCM para este vértice.

Mostre um exemplo de um grafo $G$ em que uma árvore $\overline{T}$ satisfaz à PCM para um vértice $r$, porém, independente da ordem em que os registros estejam organizados nas listas de adjacência, **não** é possível executar uma busca em **largura** tendo $r$ como raiz cuja árvore BFS seja dada por $\overline{T}$.

(b) Sejam $u$ e $v$ vértices de $G$ tais que existe um caminho de $u$ para $v$ e suponha que uma busca em **profundidade** em $G$ resultou em $d[u] < d[v]$. Zé afirma que, nestas circunstâncias, $v$ é sempre um descendente de $u$ na floresta resultante da busca. Ele está certo ou está errado ?



Pseudo-código do algoritmo de busca em profundidade visto em aula

**DFS($G$)**
**para** cada vértice $u \in V$ **faça**
  $\texttt{cor}[u] \leftarrow \texttt{branco}$; $\texttt{pred}[u] \leftarrow \texttt{NULO}$;
$\texttt{tempo} \leftarrow 0$;
**para** cada vértice $u \in V$ **faça**
  **se** $\texttt{cor}[u] = \texttt{branco}$ **então** DFS-AUX($u$)

**DFS-AUX($u$)** $\triangleright$ $u$ acaba de ser descoberto
$\texttt{cor}[u] \leftarrow \texttt{cinza}$;
$\texttt{tempo} \leftarrow \texttt{tempo} + 1$; $d[u] \leftarrow \texttt{tempo}$;
**para** $v \in \texttt{Adj}[u]$ **faça** $\triangleright$ explora aresta $(u,v)$
  **se** $\texttt{cor}[v] = \texttt{branco}$ **então**
    $\texttt{pred}[v] \leftarrow u$; DFS-AUX($v$);
$\texttt{cor}[u] \leftarrow \texttt{preto}$; $\triangleright$ $u$ foi explorado
$\texttt{tempo} \leftarrow \texttt{tempo} + 1$; $f[u] \leftarrow \texttt{tempo}$;