Departamento de Electrónica, Telecomunicações e Informática da Universidade de Aveiro

Algoritmos e Estruturas de Dados

2024/2025 — 1º Semestre

2º Trabalho — O TAD GRAPH

Data limite de entrega: 3 de janeiro de 2025, às 18 horas

O tipo de dados **GRAPH** foi apresentado nas aulas, e permite representar e operar sobre **grafos e grafos orientados**, com ou sem pesos associados às suas arestas. A estrutura de dados usada é constituída por uma **lista de vértices** e, para cada vértice, pela sua **lista de adjacências**. Estas listas são definidas usando o tipo de dados genérico **SORTED-LIST**.

O tipo **GRAPH** fornece apenas as **operações básicas** sobre grafos. Outros **algoritmos** são implementados em **módulos autónomos**.

OBJETIVOS

Este trabalho vai permitir desenvolver **algoritmos** sobre **grafos sem pesos associados aos arcos**, e efetuar a **análise da eficiência computacional** de algumas das estratégias desenvolvidas.

Assim, é necessário:

1. **TAD GRAPH:** desenvolver a função que, dado um grafo orientado, constrói o correspondente **grafo orientado transposto**.

2. **Módulo BELLMAN-FORD:** desenvolver a função que, dado um grafo, sem pesos associados às arestas, e um vértice inicial, constrói a **árvore dos caminhos mais curtos** entre esse vértice inicial e cada um dos outros vértices alcançáveis, usando o **algoritmo de Bellman-Ford**. Consultar, por exemplo, a Wikipédia.

3. **Módulo TRANSITIVE-CLOSURE:** desenvolver a função que permite, dado um grafo orientado, sem pesos associados aos arcos, construir o grafo orientado que é o seu **fecho transitivo**. Esse grafo tem os mesmos vértices que o grafo original, existindo um arco orientado entre os vértices u e v se, no grafo original, v for alcançável a partir de u, i.e., existe um caminho orientado entre esses vértices. Os **vértices alcançáveis**, a partir de um dado vértice de um grafo orientado, deverão ser determinados usando o módulo **BELLMAN-FORD**.

4. **Módulo ALL-PAIRS-SHORTEST-DISTANCES:** desenvolver as funcionalidades que permitam, dado um grafo orientado, sem pesos associados aos arcos, construir a **matriz de distâncias** que, para cada **par de vértices**, contém a distância associada ao correspondente **caminho mais curto**, caso exista. Os caminhos mais curtos deverão ser determinados usando o módulo **BELLMAN-FORD**.

5. **Módulo extra – Valorização adicional**

   Módulo ECCENTRICITY-MEASURES: desenvolver as funcionalidades que permitam, dado um grafo orientado, sem pesos associados aos arcos, calcular características do grafo e dos seus vértices, que são baseadas nos valores de distância associados a caminhos mais curtos. Assim devem ser determinados: 1) **a excentricidade de cada vértice** de um grafo (i.e., a maior distância entre esse vértice e cada um dos outros vértices do grafo), 2) **o raio de um grafo** (i.e., o menor valor de excentricidade de todos os seus vértices), 3) **o diâmetro de um grafo** (i.e., o maior valor de excentricidade de todos os seus vértices), e 4) **o conjunto dos vértices centrais** de um grafo (i.e., o conjunto dos vértices cuja valor de excentricidade é igual ao raio do grafo). As distâncias entre pares de vértices deverão ser determinadas usando o módulo **ALL-PAIRS-SHORTEST-DISTANCES**.

- **Análise da Complexidade:** Caracterizar a complexidade algorítmica das soluções implementadas para 1) o **algoritmo de Bellman-Ford**, e para 2) o **algoritmo de construção do fecho transitivo** de um grafo.

TAREFAS BÁSICAS

- **Completar o desenvolvimento das funções pedidas nos vários ficheiros `.c`**
- **Não devem ser alterados os correspondentes ficheiros `.h`**
- Assegurar que essas funções executam corretamente paras os grafos orientados (muito simples) dos ficheiros de exemplo: `Test*.c`
- Testar os algoritmos com grafos orientados mais complexos.

ANÁLISE DA COMPLEXIDADE – PARA NOTAS SUPERIORES A 16 VALORES

- Escrever um **relatório sucinto** (máx. 6 págs.), com a caracterização da **complexidade** das soluções implementadas para 1) o **algoritmo de Bellman-Ford**, e para 2) o **algoritmo de construção do fecho transitivo** de um grafo.
- O relatório deverá incluir uma breve descrição da(s) **métrica(s) adotada(s)** para medir a complexidade e **tabelas (gráficos) com os resultados dos testes** efetuados.
- **A entrega de um relatório não significa por si só que a nota do final do trabalho venha a ser superior a 16.**

Atenção – Desenvolvimento do código

- Os vértices de um grafo estão sequencialmente numerados: 0, 1, 2, …
- Deve respeitar os protótipos das funções definidos nos vários ficheiros cabeçalho.
- Pode criar **funções auxiliares (static)** sempre que achar útil.

- O **código** desenvolvido deverá ser **claro** e **comentado** de modo apropriado: os identificadores escolhidos para as variáveis e a estrutura do código, bem como os eventuais comentários, deverão ser suficientes para a sua compreensão.

- No final do trabalho deverá ser entregue um **ficheiro ZIP** com o código desenvolvido e um **ficheiro PDF** com o relatório da análise da complexidade.

- Não é necessário entregar qualquer relatório relativo ao desenvolvimento do código.

Critérios de Avaliação

- **Desenvolvimento e teste das funções pedidas (16 valores)**
  - Qualidade do código (eficiência, legibilidade, clareza, robustez)
  - Teste do código e verificação de fugas de memória

- **Relatório (4 valores)**
  - Aspetos Gerais/Apresentação/Conclusão
  - Complexidade dos dois algoritmos analisados
    - Dados experimentais

Atenção:

- O trabalho deve ser realizado em grupos de 2 alunos, mantendo-se os grupos do trabalho anterior, sempre que possível.

- A entrega do trabalho (código + relatório opcional) será feita através da plataforma eLearning.