U.PORTO
FEUP FACULDADE DE ENGENHARIA
UNIVERSIDADE DO PORTO

Portugal Water Supply System

David Carvalho up202208654
João Torres up202205576
Leonardo Magalhães up202208726

“A arte de programar consiste em organizar e dominar a complexidade.”

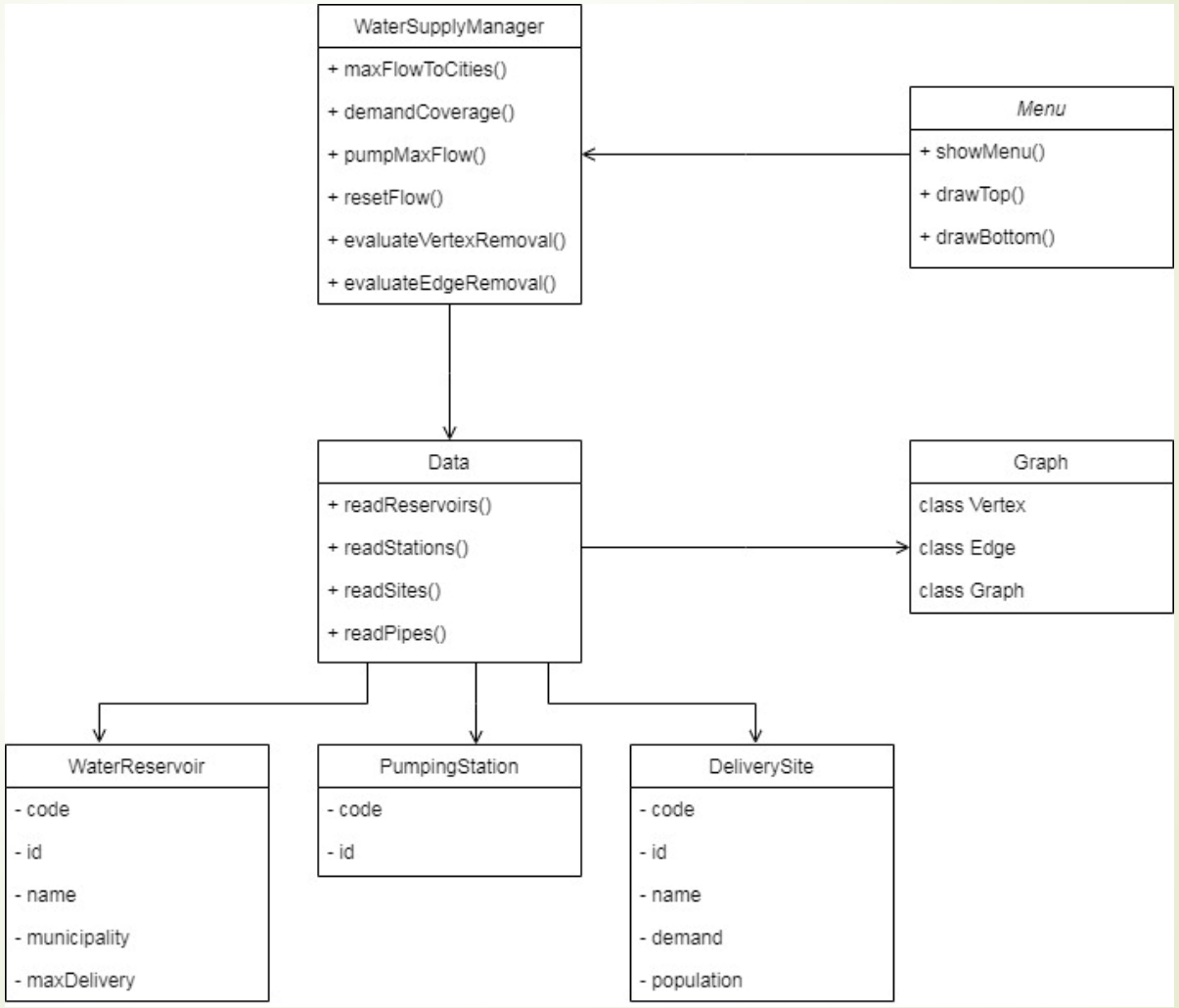
Edsger W. Dijkstra

Índice

Introdução

- No contexto da disciplina de Desenho de Algoritmos, empenhámo-nos na criação de um projeto que demonstra o nosso compromisso com a aplicação prática de conceitos avançados nesta área.
- Utilizamos o eficiente método de Edmonds-Karp do algoritmo de Ford-Fulkerson, para maximizar a eficácia e a precisão do nosso projeto.
- Estamos confiantes de que o nosso projeto corresponderá e atingirá os objetivos propostos.

Diagrama de Classes



Leitura do Dataset

- readReservoirs(const string &filename):

Lê os dados e cria objetos WaterReservoir.

Insere os objetos WaterReservoir em um mapa chamado reservoirs, usando o código do reservatório como chave.

- readStations(const string &filename):

Lê os dados e cria objetos PumpingStation.

Insere os objetos PumpingStation em um mapa chamado stations, usando o código da estação como chave.

Leitura do Dataset

- readSites(const string &filename):

Lê os dados de e cria objetos DeliverySite.

Insere estes objetos em um mapa chamado sites, usando o código do local como chave.

- readPipes(const string &filename):

Lê os dados de pipes e cria arestas no grafo, representando as conexões entre os locais.

Define o tipo de cada vértice no grafo com base no prefixo do código do vértice (reservoir, pumping station ou city).

Estrutura do Grafo Utilizado

- Vértices (Water Reservoirs, Pumping Stations e Delivery Sites):

Cada vértice no grafo corresponde a um destes.

As propriedades associadas a cada vértice incluem código, seletor, cidade e fluxo de água.

```
template<class T>
class Vertex {
    T code;                // code node
    int sel;  //1(water reservoir) ,2(pumping station) or 3(delivery site)
    int flow;
    std::vector<Edge<T> *> adj;  // outgoing edges

    // auxiliary fields
    bool visited = false; // used by DFS, BFS, Prim ...
    bool processing = false; // used by isDAG (in addition to the visited attribute)
    unsigned int indegree; // used by topsort
    double dist = 0;
    Edge<T> *path = nullptr;

    std::vector<Edge<T> *> incoming; // incoming edges

    int queueIndex = 0;        // required by MutablePriorityQueue and UFDS

    void deleteEdge(Edge<T> *edge);
```

Estrutura do Grafo Utilizado

- Arestas (Pipes):

As arestas conectam os vértices e representam os pipes entre os diferentes tipos de reservatórios de água.

As arestas têm um peso associado, representando a capacidade de cada pipe. Esse peso pode ser usado para otimizar fluxos de água.

```
template<class T>
class Edge {
protected:
    Vertex<T> *dest; // destination vertex
    double weight; // edge weight, can also be used for capacity

    // auxiliary fields
    bool selected = false;

    // used for bidirectional edges
    Vertex<T> *orig;
    Edge<T> *reverse = nullptr;

    double flow; // for flow-related problems
```

Estrutura do Grafo Utilizado

- Este grafo fornece uma estrutura eficiente para modelar e analisar a conexão entre todos estes vértices, permitindo a implementação de algoritmos para planejamento e gerenciamento de fluxos de água e outras funcionalidades relevantes para a gestão de um sistema de água.

```
template<class T>
class Graph {
protected:
    std::vector<Vertex<T> *> vertexSet;    // vertex set

    double **distMatrix = nullptr;   // dist matrix for Floyd-Warshall
    int **pathMatrix = nullptr;   // path matrix for Floyd-Warshall

    /*
     * Finds the index of the vertex with a given content.
     */
    int findVertexIdx(const T &in) const;
```

Funcionalidades e Algoritmos Implementados

```
void maxFlowToCities();

void demandCoverage();

int pumpMaxFlow();

void resetFlow();

void evaluateVertexRemoval(const std::string& vertex);

void evaluateEdgeRemoval(const std::string &source, const std::string &destination);

void printCitiesDetails();

void printReservoirsDetails();

void printStationsDetails();
```

Interface com o Utilizador

Visualização do Menu de Seleção Inicial

```
Loading ...

_____________________________________________________
|====================== Menu ======================|
|_____________________________________________________|
| 1. Portugal's continental water supply network   |
| 2. Madeira water supply network                  |
|_____________________________________________________|
|=====================================================|
|_____________________________________________________|
Choose an option:
```

Interface com o Utilizador

Visualização do Menu Inicial

```
 ------------------------------------------------
|====================== Menu ====================|
|________________________________________________|
| 1. Basic Service Metrics                       |
| 2. Reliability and Sensitivity to Failures     |
| 3. Reset Water Supply System                   |
| 4. Print Network Details                       |
| Q. Exit                                        |
|________________________________________________|
|================================================|
|________________________________________________|
Choose an option:
```

Interface com o Utilizador

Visualização do Menu de Estatísticas

```
 ____________________________________________________
|====================== Menu ======================|
|__________________________________________________|
| 1. Maximum amount of water that reaches each city|
| 2. View cities that have less water than demand  |
| Q. Exit                                          |
|__________________________________________________|
|==================================================|
|__________________________________________________|
Choose an option:
```

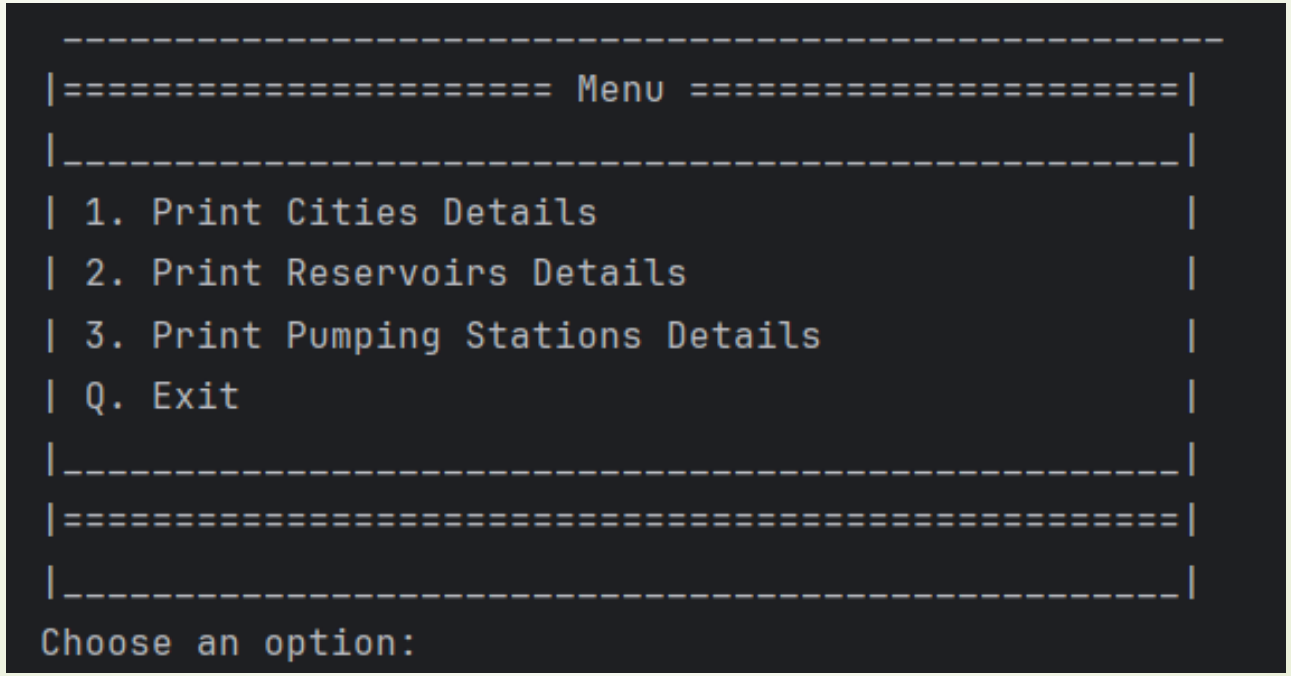
Interface com o Utilizador

Menu de Remoção de Elementos do Sistema

```
 ------------------------------------------------
|====================== Menu ====================|
|________________________________________________|
| 1. Evaluate the impact of a water reservoir    |
|  that is temporarily unavailable               |
| 2. Evaluate the impact of a pumping station    |
|  that is temporarily unavailable               |
| 3. Evaluate the impact of a pipe that is       |
|  temporarily unavailable                       |
| Q. Exit                                        |
|________________________________________________|
|================================================|
|________________________________________________|
Choose an option:
```

Interface com o Utilizador

Menu de Detalhes do Sistema



Algoritmo de Balanceamento

- Cálculo das Métricas Iniciais:

Calcular as diferenças entre a capacidade e o fluxo para cada pipe.

Calcular a média, variância e diferença máxima.

- Algoritmo de Balanceamento:

Começar com um estado inicial onde cada pipe possui um fluxo.

Identificar pipes com fluxos próximos à capacidade.

Redistribuir o excesso de fluxo dessas pipes para pipes com proporções de fluxo/capacidade mais baixas, verificando as capacidades máximas dos vértices e a necessidade de cada cidade.

Repetir esse processo iterativamente até que as diferenças nas proporções de fluxo/capacidade entre as pipes sejam minimizadas, mantendo o flow de cada cidade inalterado.

Algoritmo de Remoção

- Calcular o fluxo máximo inicial:

Usar o algoritmo Max-Flow normal para toda a rede com todos os vértices.

Guardar o valor do fluxo em cada vértice da rede.

Remover o vértice pretendido.

- Atualizar o fluxo parcialmente:

Em vez de refazer o Max-Flow, ver o impacto do vértice removido:

Identificar os pipes entrando e saindo do vértice e como o vértice já não existe, essas pipes são desconectadas da rede.

Colocar o valor de fluxo a 0 pois não contribuem mais.

Reutilizar os valores de fluxo calculados antes para as outras ligações (as que não mudam).

Algoritmo de Remoção

- Verificação final:

Depois de mudar o fluxo das ligações do vértice removido, comparar o novo fluxo em cada ligação com a sua capacidade.

Se o novo fluxo for maior que a capacidade, é um sinal de que a rede pode ter problemas de capacidade após remover o vértice.

Destaque de Funcionalidades

- Sentimos um grande orgulho na eficiente leitura e processamento dos dados, resultando na criação de um grafo dinâmico de voos com informações cruciais. O controlo da complexidade temporal evidencia a nossa atenção meticulosa aos detalhes.
- Neste projeto, orgulhamo-nos também de ter feito todos os parâmetros pedidos.
- Para além disso, ainda incluímos a remoção em cadeia de elementos na Water Supply System e a possibilidade de dar um reset geral a este para voltar ao estado inicial. Consideramos que isso revela a maior rigidez e complexidade do nosso sistema.

Principais Desafios e Esforço do Grupo

- Enfrentámos desafios iniciais na interpretação dos dados, que foram superados com esforço conjunto. A implementação precisa do grafo dinâmico e a otimização da complexidade temporal exigiram colaboração e a escolha cuidada de algoritmos, destacando a nossa capacidade de superar obstáculos no desenvolvimento deste sistema.
- Todos os elementos contribuíram para os muitos âmbitos deste projeto e o esforço de cada um foi equilibrado e essencial para o resultado desenvolvido e apresentado.

Conclusão

- No âmbito do projeto proposto, dedicamo-nos totalmente à aplicação de conceitos avançados de desenho de algoritmos e estruturas de dados de forma eficaz e eficiente.
- Empenhámo-nos profundamente na análise de algoritmos, selecionando cuidadosamente as estruturas de dados mais apropriadas para assegurar o desempenho eficaz do projeto.
- Procurámos, igualmente, otimizar a complexidade temporal e espacial do código, conduzindo análises de complexidade e realizando testes de desempenho.

U.PORTO
FEUP FACULDADE DE ENGENHARIA
UNIVERSIDADE DO PORTO

Obrigado!