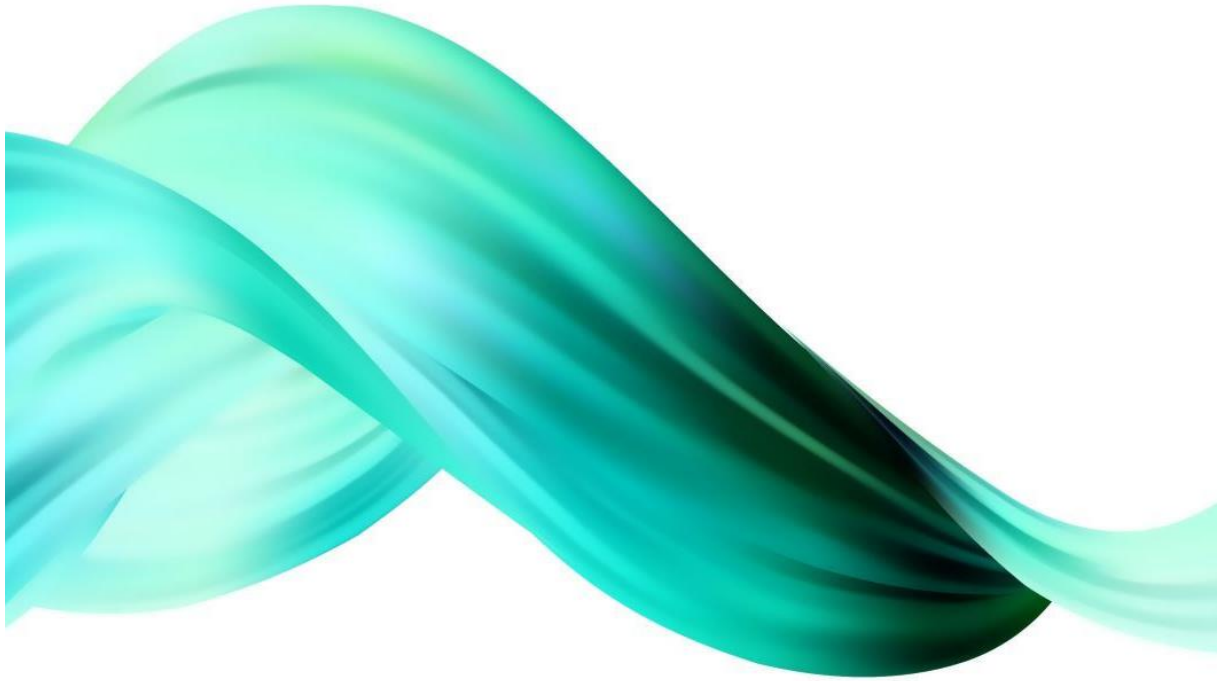
PROJETO 1 DA

- Realizado por :

- Rafael Campeão - up202207553
- Lucas Faria - up202207540
- Pedro Borges - up202207552



INTRODUÇÃO

- Neste trabalho desenvolvemos uma ferramenta de análise com o objetivo de ajudar a fazer as melhores escolhas para a distribuição de água em Portugal. Atráves desta ferramenta o utilizador consegue saber como seria a melhor distribuição de água em Portugal e também identificar partes sensíveis da rede de abastecimento para antecipar estragos no serviço ou pelo menos diminuir ao máximo as consequências.
- Neste trabalho implementamos algorítmos e estruturas de dados apresentadas durante as aulas de forma a obter o melhor sistema possível.

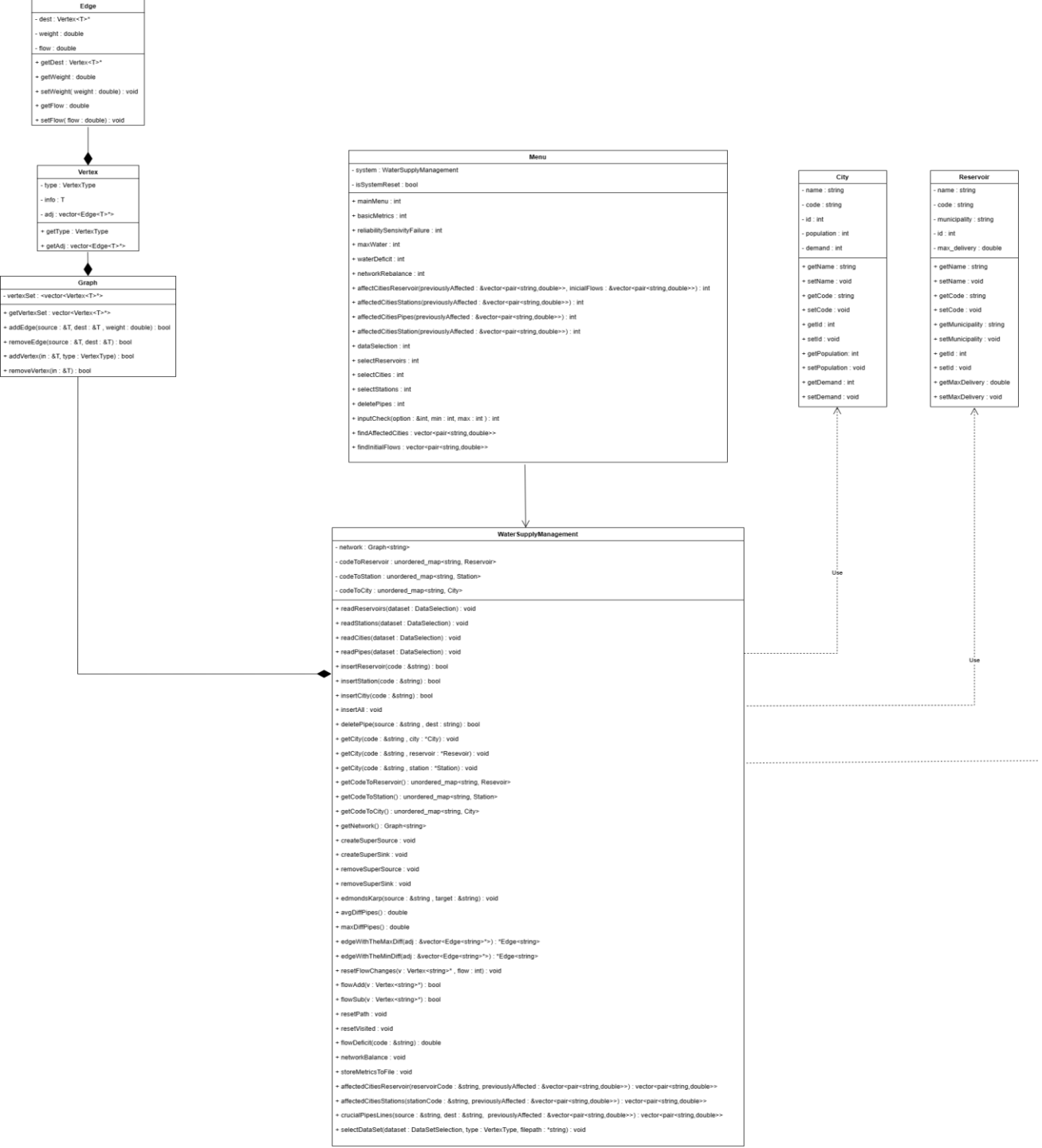


DIAGRAMA DE CLASSES

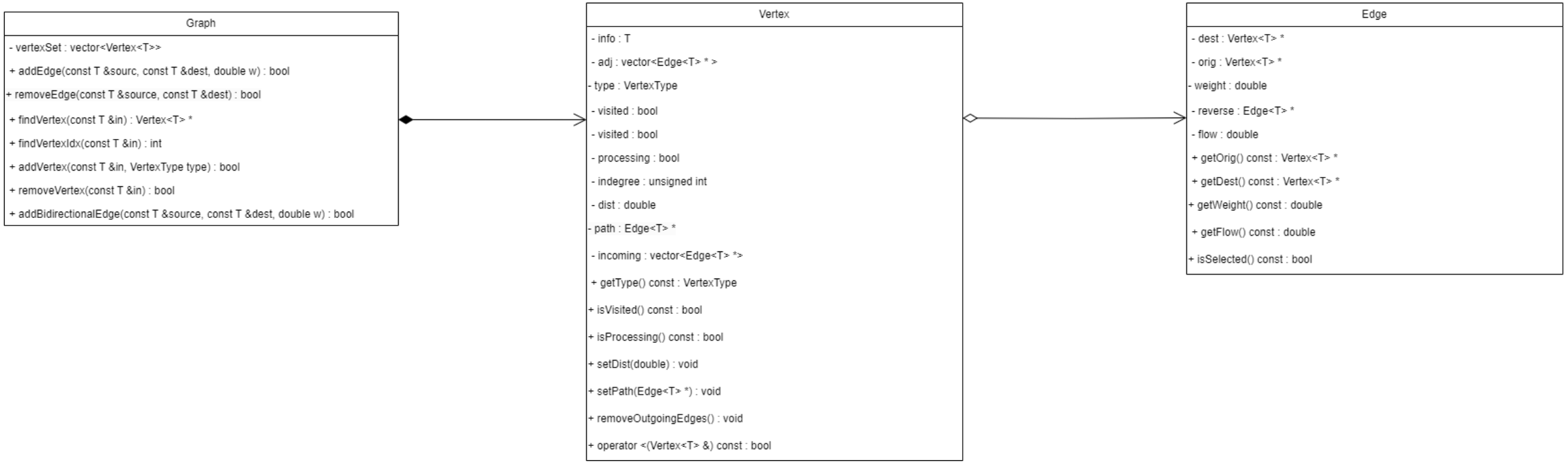
- Menu: Menu.h, Menu.cpp
- **WaterSupplyManagment:** WaterSupplyManagment.h,
- WaterSupplyManagment.cpp
- **City:** City.h
- **Reservoir:** Reservoir.h
- **Station:** Station.h
- **Graph:** Graph.h
- **Vertex:** Graph.h, VertexType.h
- **Edge:** Graph.h

LEITURA DO DATASET

- Primeiramente, é lido o ficheiro com as Cities.
- Estas são guardadas numa Hash Table (unordered_map), acedidas através do seu código.
- De seguida, é lido o ficheiro com os Reservoirs.
- Estes são guardados também numa Hash Table e também sao acedidos através do seu código.
- Posteriormente, é lido o ficheiro com as Stations.
- Estes também são guardados da mesma forma que os anteriores.
- Todos estes elementos anteriormente mencionados compõem o nosso VertexSet.
- Por fim, é lido o ficheiro com os Pipes.
- Estes são as nossas arestas (Edges) entre dois vertíces.

UTILIZAÇÃO DE GRAFOS

- Foi utilizado um grafo neste trabalho formado por 5 classes:
  - **Graph.h** - correspondente à classe Graph utilizada nas aulas, com a adição de algumas funções como getters e setters, sendo a maior diferença a adição de um atributo chamado VertexType.
  - **VertexType.h** - corresponde à distinção dos diversos Vertex enumerados em seguida.
    - **Station.h** - corresponde a uma parte dos Vertex da network.
    - **Reservoir.h** - corresponde a uma parte dos Vertex da network.
    - **City.h** - corresponde a uma parte dos Vertex da network.



UTILIZAÇÃO DE GRAFOS

FUNCIONALIDADES

- As nossas funcionalidades podem ser divididas em 2 partes:
    - **Métricas básicas de serviço;**
    - **Sustentabilidade e Sensibilidade a falhas.**

FUNCIONALIDADES - MÉTRICAS BÁSICAS DE SERVIÇO

- Algorítmo de Edmonds Karp, subdivido em várias funções:
    - edmondsKarp – Complexidade O(V E^2), onde V é o número de vertíces e E de arestas do grafo.
    - testAndVisit - Complexidade O(1);
    - findAugmentingPath – Complexidade O(E), onde E é o número de arestas do grafo;
    - findMinResidualAlongPath – Complexidade O(n) onde n é o número de arestas do caminho (path);
    - augmentFlowAlongPath – Complexidade O(n) onde n é o número de arestas do caminho (path);
    - createSuperSource - Complexidade O(n) onde n é o número de Reservoirs do grafo;
    - createSuperSink - Complexidade O(n) onde n é o número de Cidades do grafo;
- Cálculo do flow deficit (diferença entre demanda de água de uma cidade e o que realmente recebe) - Complexidade O(V) onde V é o número de vértices do grafo.
- Guardar métricas de cada Cidade em um ficheiro – Complexidade O(n) onde n é o número de cidades.
- Balanceamento da network – Complexidade O(VE^2 D) onde V é o número de vértices, E o número de arestas e D é o número de vezes que o loop corre (depende do balanço original do grafo, no pior caso é exponencial).

FUNCIONALIDADES - MÉTRICAS BÁSICAS DE SERVIÇO

- Funções auxiliares para o balanço da network -
  - flowAdd – Adiciona flow nos pipes que têm a maior diferença entre capacidade e flow, O(E) onde E é o número de pipes (edges).
  - flowSub – Subtrai flow nos pipes que têm a menor diferença entre capacidade e flow, O(E) onde E é o número de pipes.
  - resetFlowChanges - Reseta as mudanças anteriormente feitas ao adicionar ou subtrai flow, O(E) onde E é o número de pipes.
  - edgeWithMinDiff - Encontra o pipe com a menor diferença (entre capacidade e flow) num conjunto de pipes, O(n) onde n é o tamanho do vetor de pipes.
  - edgeWithMaxDiff - Encontra o pipe com maior diferença num conjunto de pipes, O(n) onde n é o tamanho do vetor de pipes.
  - resetPath – Reseta o atributo path de todos os vertíces do grafo, O(v) onde v é o número de vertíces do grafo.
  - resetVisited – Reseta o atributo visited de todos os vertíces do grafo, O(v) onde v é o número de vertíces do grafo.

FUNCIONALIDADES - MÉTRICAS BÁSICAS DE SERVIÇO

- Outras métricas auxiliares:
  - avgDiffPipes – Calcula a diferença média entre capacidade e flow de cada pipe, O(VE) onde V é o número de vertíces (sem ser cidades) e E é o numero de pipes (edges).
  - maxDiffPipes – Calcula a maior diferença entre capacidade e flow de cada pipe, O(VE) onde V é o número de vertíces (sem ser cidades) e E é o numero de pipes (edges).

FUNCIONALIDADES - SUSTENTABILIDADE E SENSIBILIDADE A FALHAS

- affectedCitiesReservoir – Encontra as cidades que foram afetadas ao remover um dado Reservoir, O(V E^2) onde V é o número de vertíces e E é o número de pipes do grafo.
- affectedCitiesStations - Encontra as cidades que foram afetadas ao remover uma dada Station, O(V E^2) onde V é o número de vertíces e E é o número de pipes do grafo.
- crucialPipelines – Encontra as cidades que foram afetadas ao remover um dado Pipe, O(V E^2) onde V é o número de vertíces e E é o número de pipes do grafo.

- Outra funcionalidade sobre a network em si:
  - resetSystem – Reenicia o sistema (vertíces e as arestas), Complexidade O(1).

INTERFACE DE UTILIZADOR

- A nossa User Interface (UI) é composta por um conjunto de menus e submenus, permitindo assim ao utilizador usar o nosso sistema com facilidade. Existem 3 menus principais divididos depois em submenus:
  - Menu Principal – Primeiro Menu que aparece quando o User dá load ao sistema, permite escolher que tipo de data quer utilizar (toda ou personalizada).
  - Menu das Métricas Básicas - Menu onde o User pode escolher que tipo de métricas deseja executar ou também guardar num ficheiro (metrics.csv).
  - Menu da Sustentabilidade e Sensibilidade a Falhas – Menu onde o User pode escolher que tipo de rutura pretende causar para analisar as suas consequências.

INTERFACE DE UTILIZADOR - DEMONSTRAÇÃO

Vamos ver alguns exemplos da utilização da UI.

ALGORITMO MELHORADO PARA RETIRAR RESERVOIRS E VERIFICAR CITIES AFETADAS

- Usamos inicialmente o algoritmo de Edmonds Karp para calcular os valores iniciais de flow;
- Removemos um reservoir;
- Eliminamos completamente o flow de pipes que apenas recebem água do reservoir que retiramos;
- Para verificar que o pipe apenas recebe água do reservoir que retiramos, percorremos o pipe no sentido reverso ao do seu flow (o flow não pode ser 0) até encontrarmos outro reservoir. Caso não encontremos outro reservoir, então esse pipe depende inteiramente do reservoir retirado;
- Caso o pipe também seja abastecido por outros reservoirs, temos que verificar a quantidade de água que lá chega pertencente a outros reservoirs;
- Em seguida, reduzimos o flow no pipe para aquele que anteriormente verificamos que provem de outros reservoirs;
- Por fim, usamos uma versão simplificada do Edmonds Karp (sem restaurar os valores de flow para 0) para verificar se sobrou algum augmenting path.
- Devemos seguir este procedimento para cada reservoir retirado, sendo que devemos ter guardados os dados originais do grafo antes de retirar um reservoir para poderem ser restaurados antes de podermos retirar outro reservoir.
- Pode não compensar para grafos mais pequenos ou com poucos reservoirs porque acabaremos por percorrer o grafo todo, talvez mais que uma vez.

ALGORITMO DE REBALANCEAMENTO DA REDE

- Estratégia greedy:
  - Escolher pipe com menor diferença para retirar flow e com maior diferença para adicionar flow.
- Fazemos a escolha greedy para cada station e reservoir do grafo (outgoing edges).
- Apenas fazemos alterações no flow de um pipe se encontrarmos um caminho que permita adicionar/retirar uma unidade de flow sucessivamente até encontrarmos uma cidade.
- Apenas passamos para outra station ou reservoir se não existirem novos caminhos ou a média das diferenças entre capacity e flow em cada pipe não melhorar.
- Complexidade: O(VE^2 F) em que F é desconhecido (depende da média das diferenças e pode ser no pior dos casos exponencial)

FUNCIONALIDADES DESTACADAS

- A funcionalidade que mais destacamos no nosso trabalho é o do **rebalanceamento da rede.**
- Esta era a função mais complicada de implementar.
- Exigia um certo nível de estudo do problema e atenção.
- Conseguimos com êxito implementar esta funcionalidade no sistema.

MAIORES DIFICULDADES E PARTICIPAÇÃO

- As maiores dificuldades foram:
  - Adaptar a estrutura do grafo usado nas aulas ao problema que o projeto propunha.
  - Implementar alguns algoritmos mais complicados como, por exemplo, o de rebalanceamento da rede.

- As dificuldades foram superadas com a contribuição e esforço homogéneo de todos os elementos do grupo.
- A distribuição de trabalho foi equilibrado por todos os elementos sendo que cada elemento participou em um pouco de tudo (desenvolver algoritmos, trabalhar no menu, trabalhar na apresentação, etc..)

CONCLUSÃO

1

Neste projeto tivemos a oportunidade de desenvolver um sistema de gestão de uma rede de abastecimento de água.

2

Além disso, tivemos a oportunidade de aplicar os diversos conceitos aprendidos nas aulas, em especial os algoritmos de maximum flow.

3

Por isso, podemos concluir que este projeto foi de grande proveito para a aprendizagem neste unidade curricular.