U.PORTO
FEUP FACULDADE DE ENGENHARIA UNIVERSIDADE DO PORTO

Design of Algorithms
First Project

Group G15_2
Bruno Oliveira up202208700
Rodrigo Silva up202205188
Vítor Pires up202207301

TABLE OF CONTENTS

Introduction

Problem:

- Creating a *Water Supply Management System;*

Tools:

- Implementation of a directed graph;
- Standard Template Library (STL);
- Appropriate data structures;
- Efficient algorithms.

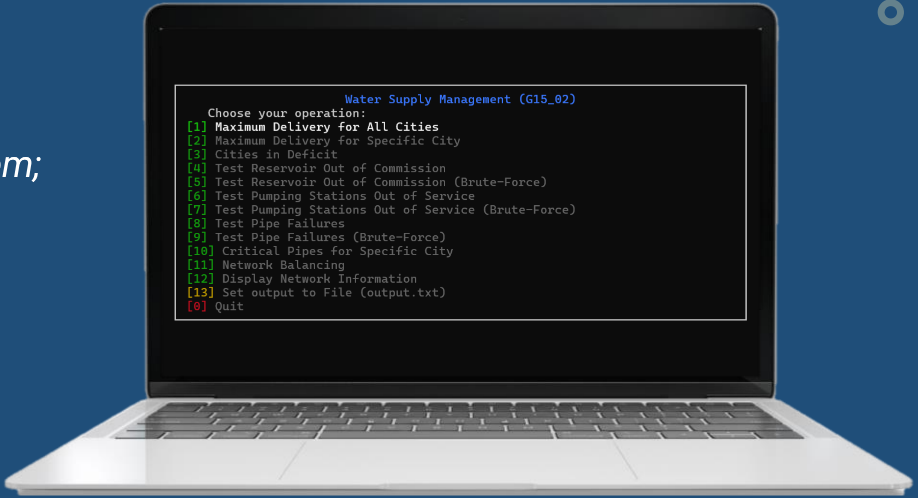


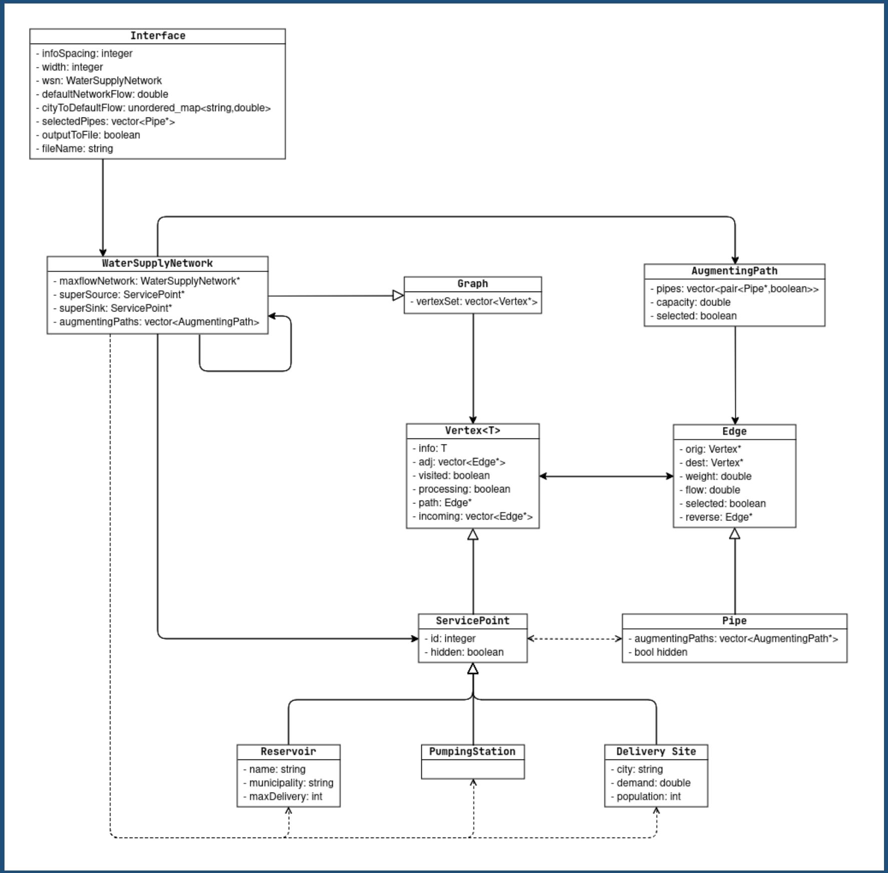
TABLE OF CONTENTS

Class Diagram

Each class:

- Is declared in a `.h` file with the same name (inside `include/`)
- Has its functionality defined in the respective `.cpp` file (inside `src/`, with the exception of the generic classes).

The app logic (main algorithms) is implemented as methods of `WaterSupplyNetwork`, and the `Interface` is responsible for the UI.



Graph Representation

Vertices — **Service Points**

Reservoir (R)
- Contains reservoir information
(name, municipality, id, code, max. delivery)

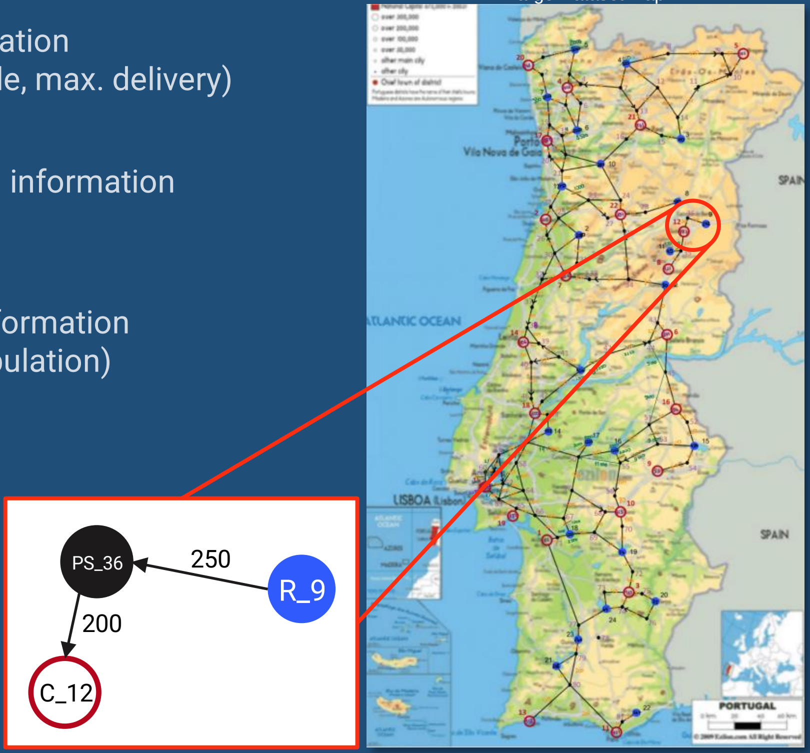
Pumping Stations (PS)
- Contains pumping station information
(id, code)

Delivery Sites (C)
- Contains delivery sites information
(city, id, code, demand, population)

Edges

Pipes
- Connection between two service points
(origin and destination)
- Undirected connections are represented
using two (directed) pipes

Large Dataset Map



Graph Representation

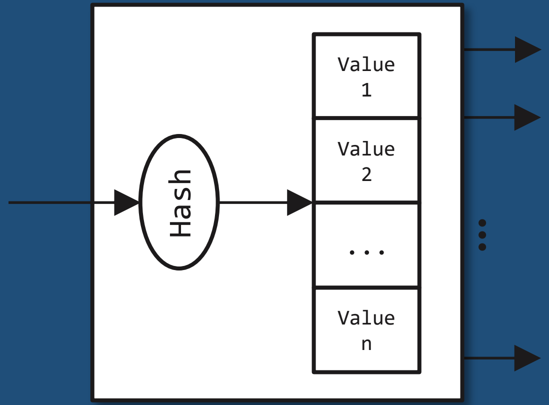
We used a graph **adjacency list representation**, similar to the one used in the practical classes, which we kept as a generic graph.

Refinement:

- The vertices are stored in a **hash table** instead of a vector.

Advantages:

- Constant time for search, removal and insertion;
- Therefore, lower-complexity algorithms.



Simplified representation of a **hash table**

TABLE OF CONTENTS

Reading the Dataset

```
                Water Supply Management (G15_02)
   Choose the Dataset
[1] Large Data Set (Portugal)
[2] Small Data Set (Madeira)
[3] Custom Data Set (Check README)
[0] Quit
```

Reservoir.csv
Stations.csv
Cities.csv
Pipes.csv

parseData(“Reservoir.csv”, “Stations.csv”, “Cities.csv”, “Pipes.csv”)

- → parseReservoir(“Reservoir.csv”)
- → parseStations(“Stations.csv”)
- → parseCities(“Cities.csv”)

} addVertex()

- → parsePipes(“Pipes.csv”)

} addEdge() or addBidirectionalEdge()

Parse the data files → Initialize data structures

Implemented Features

Display Network Info

Delivery Site (City) Display

Code	Id	City	Demand	Population
C_10	10	Calheta	76	10915
C_7	7	Câmara de Lobos	225	32162
C_6	6	Funchal	740	105782
C_4	4	Machico	137	19593
C_9	9	Ponta do Sol	59	8360
C_2	2	São Vicente	34	4865
C_5	5	Santa Cruz	295	42168
C_1	1	Porto Moniz	18	2517
C_3	3	Santana	46	6553
C_8	8	Ribeira Brava	89	12680

< Press **ENTER** to continue >

For delivery sites, reservoirs, pumping stations and pipes

Calculate Max Flow For All

Normal Max Flow (All Cities)

Code	Id	City	Demand	Flow	Population
C_10	10	Calheta	76	76	10915
C_7	7	Câmara de Lobos	225	225	32162
C_6	6	Funchal	740	664	105782
C_4	4	Machico	137	137	19593
C_9	9	Ponta do Sol	59	59	8360
C_2	2	São Vicente	34	34	4865
C_5	5	Santa Cruz	295	295	42168
C_1	1	Porto Moniz	18	18	2517
C_3	3	Santana	46	46	6553
C_8	8	Ribeira Brava	89	89	12680

Total Network Flow: 1643

< Press **ENTER** to continue >

Maximizes the total flow to all cities

$O(VE^2)$

Calculate Max Flow For City

Focused Max Flow (Porto Moniz)

Code	City	Demand	Normal	Focused	Population
C_1	Porto Moniz	18	18	18	2517

< Press **ENTER** to continue >

Redirects the flow to only one city as much as possible

$O(VE^2)$

Implemented Features

Check Cities in Deficit

Cities with Demand Deficits (Normal Flow considered)

Code	City	Demand	Flow	Deficit	Deficit %
C_6	Funchal	740	664	76	10.27

There is 1 city in deficit!

< Press ENTER to continue >

$O(VE^2)$

Test Failure

Max Flow without Pumping Station PS_1

Code	City	Normal	Affected	Delta	Delta %
C_10	Calheta	76	50	26	34.211
C_6	Funchal	664	625	39	5.8735
C_5	Santa Cruz	295	278	17	5.7627
C_1	Porto Moniz	18	0	18	100

There are 4 cities affected!
Total Network Flow: 1543 / 1643

< Press ENTER to continue >

See effects of putting a reservoir, pumping station or pipe out of commission

$O(VE^2)$

Implemented Features

Critical Pipes For City

Cities with Demand Deficits (Normal Flow considered)

Code	City	Demand	Flow	Deficit	Deficit %
C_6	Funchal	740	664	76	10.27

There is 1 city in deficit!

< Press ENTER to continue >

Get pipes that affect the city flow

$O(VE^3)$

Balance Network

Balancing Metrics (this may take a while)

Maximum Difference	Mean Difference	Variance
561	96.833	24769.2
421	67.119	10283.7
344	67.19	10082.5
344	67.19	10082.5

< Press ENTER to continue >

See effects of putting a reservoir, pumping station or pipe out of commission

$O(VE^3C)$

Balancing Algorithm

```
function balance(value):
  calculate max flow;
  while there is no pipe that can be changed do
    store current max flow in a copy;
    for each edge in increasing order of (capacity – flow) do
      decrease pipe capacity by value;
      recalculate max flow;
      if flow value decreased then
        restore max flow;
      else
        end inner for loop;
  restore capacities;
```

- The parameter value affects the **precision** and **efficiency** of the algorithm.
- Smaller values lead (in general) to better results, while making the algorithm run slower.

- For every capacity decrement, the algorithm can run the Edmonds-Karp for every pipe in the network, resulting in a worst-case complexity of $O(VE^3C)$, where C is the sum of capacities of all pipes.

Failure Algorithm

```
function recalculateMaxFlowWithoutPipes(pipes):
  get previous max flow and augmenting paths;
  for each pipe in pipes
    select augmenting paths that pass through the pipe;
    set pipe as hidden;
  removeSelectedAugmentingPaths();
  calculate remaining flow;

Function removeSelectedAugmentingPaths():
  for selected augmenting paths in the order they were found
    remove path flow from pipe;
    if the pipe is directed and flow is negative then
      select augmenting paths that pass through pipe;
```

- This algorithm is used for all **failure algorithms**, including for reservoirs and pumping stations (one just needs to hide the pipes incident on the service point).
- A **brute-force alternative is also provided**.

- **Note**: It sometimes redistributes the flow, resulting in different results compared to those obtained with a brute-force strategy, while still achieving a maximum flow.

Failure Algorithm

- **Removing the flow of an augmenting path** results in a **valid flow**, however **the direction constraint of a pipe can be violated** (if the pipe flow becomes negative). This results from following augmenting paths that go against the pipe's normal direction. To resolve this, we just need to **remove the next augmenting paths** that pass through the pipe.

- **Previous augmenting paths can be ignored**: the sum of their flows through that pipe, by the algorithm, must be non-negative, so removing them would result in a lower flow through the pipe (zero, to be exact).

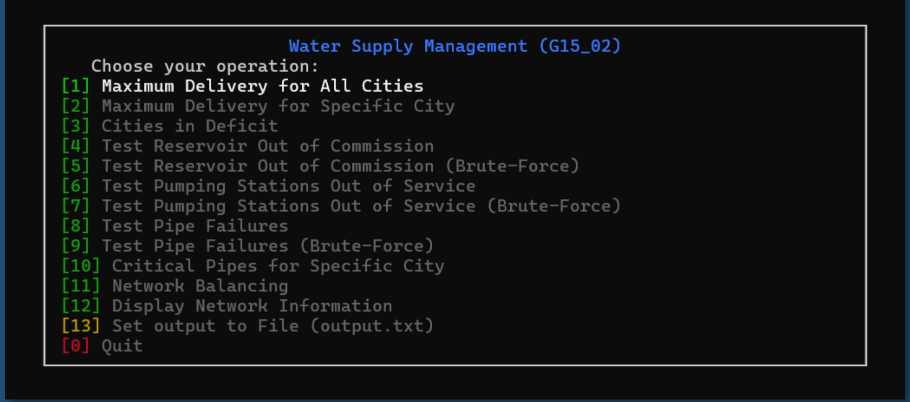
- Since it can happen that the algorithm removes all the augmenting paths, starting the Edmonds-Karp from scratch, the **worst-case complexity** of the algorithm is still $O(VE^2)$. However, since the removed flow value will be much smaller than the total flow, and thinking that rerunning the Edmonds-Karp will have, from a pseudo-polynomial standpoint, complexity $O(Ef)$, with f being the removed flow value, **the average case will be much more efficient**.

Interface

- **Terminal User Interface** (TUI) that can be controlled with and Return and Arrow keys.
- There are input capture functions defined in input.h.

- To make the menus, box-drawing characters are used and to change the colors, we've used macros with ANSI escape sequences (library created in ansi.h).

Main Menu



Interface

City Selection Menu

```
Water Supply Management (G15_02)
   Choose a city:
[1] Alcacer do Sal
[2] Évora
[3] Faro
[4] Guarda
[5] Lagos
[6] Leiria
[7] Lisboa
[8] Portalegre
[9] Porto
[10] Santarém
[0] Back
                    1 / 3   >
```

```
Water Supply Management (G15_02)
   Choose a city:
[21] Covilhã
[22] Estremoz
[0] Back
                <   3 / 3
```

Reservoir Out of Commission

(e.g. Castelo do Bode)

Max Flow without Reservoir R_13

Code	City	Normal	Affected	Delta	Delta %
C_14	Leiria	406	204	202	49.754
C_18	Santarém	200	105	95	47.5
C_6	Castelo Branco	230	100	130	56.522
C_15	Lisboa	12250	7164	5086	41.518
C_22	Viseu	330	310	20	6.0606

There are 5 cities affected!
Total Network Flow: 18630 / 24163

< Press **ENTER** to continue >

TABLE OF CONTENTS

Highlighted Functionalities

The implementation of the **more advanced** and theoretical **algorithms**:

- **Balancing** Function maintaining the max flow
- **Testing the resiliency** without recalculating the Edmonds-Karp for every iteration

Responsive interface with relatively quick responses from the algorithms and relevant metrics/comparisons. Although not as important, it's always better to view the work and the information in a pleasant and organized manner.

Main difficulties

- **Creating** and **implementing solutions** to the more theoretical questions of the project.
- **Debugging** and **testing code** was sometimes hard due to the complex algorithms and solutions to analyze.

Contributions

- Each team member contributed significantly to different phases of the project, collectively trying to overcome these challenges.

U.PORTO

FEUP FACULDADE DE ENGENHARIA
UNIVERSIDADE DO PORTO

Bruno Oliveira – Rodrigo Silva – Vítor Pires
GROUP G15_2

April, 2024