Programming Project I

An Analysis Tool for Water Supply Management

DA 2024 Instructors Team

Departamento de Engenharia Informática (DEI)/Departamento de Ciências de Computadores (DCC)
Faculdade de Eng. da Univ. do Porto (FEUP)/Faculdade de Ciências da Univ. do Porto (FCUP)

Spring 2024

Due Date: April 7, 2024 at midnight (PT time)

1. Objectives

This first programming assignment aims to expose you to realistic implementations of algorithmic solutions presented in class, and in particular to the greedy algorithmic approach in the context of maximum-flow problems. In addition, you will be required to work in teams of 2 or 3 students (3 is preferred) and hence will have to develop interpersonal and project management skills. This first programming project is not very complex, but you are still required, at the end, to prepare a short demo of the functionalities and interfaces you have developed. It is also an objective of this project to prepare you for short presentations where you must be succinct and time-conscientious, focusing on what is essential and what is not.

This document describes the motivation of this project and the expected interface, followed by a description of the problem statement and a description of the demo you are expected to present. The problem statement includes a description of each task (alongside the corresponding grading). Lastly, we provide specific turn-in instructions you need to follow. **Recall that the deadline is April 7, 2024, at midnight.**

2. Problem Motivation

In this first programming project assignment, you are asked to implement an analysis tool that can support the Management Team of a water supply network in Portugal to make informed decisions about how to best allocate its resources, both financial as well as physical (e.g., procurement of new pumping stations or installing new pipelines). Such a tool will allow management to decide on aspects such as how to best assign selected pumping stations to supply sources and how to identify more sensitive sections of its network to failures to anticipate service disruption or at least to mitigate its nefarious effects.

3. Problem Data and Interface

To make your problem realistic, you are given an idealized dataset describing the *Portuguese continental water supply network*. The dataset contains the indication of existing dams or water reservoirs, pipelines, pumping stations, and, finally, consumer sites (i.e., urban locations with residential customers). This is obviously a simplification, as there are other uses, such as agricultural uses. The entire network is thus structured as a **directed graph** with the various types of **elements,** each of which includes the following data:

1. **Water Reservoir (WR)**, for example, a dam or artesian wells. These nodes of the network provide water to the entire network and have a maximum delivery capacity of X *m³/sec*. This is a hard limit, so under regular operation, the water rate drawn from each reservoir must be less than X. These nodes are directly connected to pumping stations (PS) via pipelines with a maximum capacity of Y *m³/sec*.

2. **Pumping Stations (PS)**. These nodes direct or redirect flow from other pumping stations or water reservoirs to the final customer sites or delivery sites (DS).
3. **Delivery Sites (DS)**. These nodes directly connect the water supply network to the final consumer. Each site has a specific rate demand or need of *Z* m^3/sec.

A dataset that represents an instance of this problem is available in the ***Project1Data.zip*** file annexed to this project description, whose illustration is depicted in Figure 1 below. The cities (delivery sites) are represented by red circles, water reservoirs by blue circles, and pumping stations by black dots. Note that water reservoirs never directly connect to cities or other water reservoirs.

Figure 1. Illustration of Portugal's continental water supply network described in the *Project1Data.zip* file.

The provided dataset comprises four (4) CSV (comma-separated values) files, as described below.

- **File Reservoir.csv:** contains the information regarding water reservoirs (WR).
  The first line includes the headers: Reservoir (name of the water reservoir), Municipality (municipality where the water reservoir is located), Id (unique identifier of water reservoir), Code (code of water reservoir), and MaxDelivery (maximum delivery of water reservoir in m^3/sec).

```
Reservoir,      Municipality,       Id,    Code,  MaxDelivery
Ermida,         Aveiro,             1,     R_1,   2750
Aguieira,       Coimbra,            2,     R_2,   2080
```

- **File Stations.csv:** contains the simple reference regarding pumping stations (PS).
  The first line includes the headers: Id (unique identifier of pumping station) and Code (code of pumping station).

```
Id,Code
1, PS_1
2, PS_2
```

- **File Cities.csv:** contains the information regarding delivery sites (DS).
  The first line includes the headers: City (city of the delivery site), Id (unique identifier of the delivery site), Code (code of the delivery site), Demand (rate demand or need of delivery site in m3/sec), and Population (total population of the delivery site).

```
City,           Id,    Code,  Demand,      Population
Alcacer_do_Sal, 1,     C_1,   52,          7733
Aveiro,         2,     C_2,   515,         77241
```

- **File Pipes.csv:** contains the information regarding pipelines.
  The first line includes the headers: ServicePointA (source service that can be a water reservoir, a pumping station, or a delivery site), ServicePointB (target service that can be a water reservoir, a pumping station, or a delivery site), Capacity (maximum capacity of the pipe) and Direction (boolean indicating whether the connection between source service and target service is undirected, i.e., 1 - unidirectional and 0 - bidirectional).

```
ServicePointA,  ServicePointB,      Capacity,    Direction
R_1,            PS_21,              1000,        0
PS_22,          C_2,                260,         1
```

[T1.1: 1.0 point] Develop a Water Supply Management Analysis System Menu. Obviously, the first task will be to create a simple interface menu exposing some basic functionalities implemented in the most user-friendly way possible. This menu will also be instrumental for you to showcase the work you have developed in a short demo to be held at the end of the project.

NOTE: Please do not spend a lot of effort into making a fancy menu. This is just to facilitate the demo and does not significantly contribute to your grade.

[T1.2: 1.0 point] Read and Parse the Input Data. Similarly, you must develop some basic functionality (accessible through your menu) to read and parse the provided data set files. This functionality will enable you (and the eventual user) to select alternative water supply networks for analysis. With the extracted information, you must create one (or more) appropriate graph(s) for the requested tasks. The modeling of the graph is entirely up to you, so long as it is a sensible representation of the water supply network and enables the correct application of the required algorithms.

NOTE: Often times files are copied across different system and have different line terminating characters such as CRLF or simply a LF. This is the case of ASCII files and some of the CSV files exhibit some issues when parsed in different systems.

[
T1.3: 2.0 points] Documentation and Time Complexity Analysis. In addition, you should also include documentation of all the implemented code using Doxygen, indicating the corresponding time complexity for each of the most relevant implemented algorithms.

4. Problem Statement and Organization of Work

Given the water supply network system provided in this project, there is a crucial set of questions about managing it, which the Management Team needs your help with. To facilitate your work, we have structured the functionalities you are expected to develop so that they answer the Management Team's questions. They are divided into two sets, as follows.

4.1. Basic Service Metrics

[T2.1: 4.0 points] Determine the maximum amount of water that can reach each or a specific city (to be selected via the Menu) and display it in a natural graphical format, such as by listing one pair (city code, value) per line on the output as well as to a file. One possible way to derive this information is by using a maximum-flow algorithm (Edmonds-Karp) to find the optimal flow through the network. For this, you can set water reservoirs as sources, delivery sites as sinks, and pumping stations as intermediate nodes.

[T2.2: 1.0 point] Can an existing network configuration meet the water needs of its customer? In other words, can all the water reservoirs supply enough water to all its delivery sites? To verify this, you can simply add up all the actual water capacity being delivered to each site. For the cities that cannot be supplied by the desired water rate level, you should list them using the pairs (city code, value), where the value indicates the amount of water flow in deficit.

[T2.3: 3.0 points] Use the results from T2.1. and develop a simple algorithm or heuristic to balance the load across the network so that, as much as possible, you can minimize the differences of flow to capacity on each pipe across the entire network. Note that in some cities where the cumulative delivery capacity is very close to (but still larger than) the city's needs, it is likely that every single pipe has a flow close to its full capacity. First, you should compute some initial metrics, such as the average and variance of the differences between capacity and flow for each pipe, as well as the maximum difference. After executing your balancing algorithm, you should show that these metrics have improved.

4.2. Reliability and Sensitivity to Failures

[T3.1: 2.0 points] You might be concerned about the network's resiliency. In this context, you should evaluate what happens in terms of the delivery capacity of the network if one specific water reservoir is out of commission. How are the various delivery sites affected? Your system must allow it to take as input the water reservoir that will be out of service and determine which cities are affected by having their water supply not being met. Your interface should thus allow the user to selectively remove one reservoir and list the affected cities whose water supply does not meet its demand. Could you think about an algorithm that only sometimes needs to run the entire Max-Flow algorithm from scratch to evaluate the impact of removing all the reservoirs, one at a time? Discuss this in your presentation.

[T3.2: 2.0 points] As periodic maintenance is required, some of the pumping stations often need to be temporarily removed from the network. Can any pumping station be temporarily taken out of service without affecting the delivery capacity to all the cities? If not, which cities are most affected? For each examined pumping station, list the affected cities displaying their codes and water supply deficits.

[T3.3: 2.0 points] Sometimes, pipeline failures can occur. For each city, determine which pipelines, if ruptured, i.e., with a null flow capacity, would make it impossible to deliver the desired amount of water to a given city. For each examined pipeline, list the affected cities displaying their codes and water supply in deficit.

5. Demo & Presentation

[T4.1: 2.0 points] Giving a "short-and-to-the-point" presentation is increasingly essential. As such, you are required to structure a short 15-minute demo of your work, where you can highlight some of the aspects of your implementation, specifically:

- Present your water supply management system, using illustrative examples to demonstrate the several results requested in this project;
- Highlight your graph class, explaining the conceptual decisions that were made;
- Highlight the most important and challenging aspects of your implementations.

You should also elaborate a PowerPoint presentation to support your presentation. Instructions for preparing the PowerPoint presentation are available on Moodle.

6. Turn-In Instructions & Deadline

Submit a zip file named **DA2024_PRJ1_G<GN>.zip** on Moodle, where **GN** stands for your group number, with the following content:

- **Code** folder (contains program source code)
- Documentation folder (contains html documentation, generated using **Doxygen**)
- Presentation file (**PDF format**) that will serve as a basis for the demonstration.

Late submissions, up to 24 hours and 48 hours, will incur a penalty of 10% and 25% of the grade, respectively. No submissions will be accepted 48 hours after the deadline. Exceptions apply for justified and documented technical submission issues.

Recall that the deadline is April 7, 2024, at midnight.