Algorithms for Water Leakage Control Systems

Rúben Filipe Martins Marques

Thesis to obtain the Master of Science Degree in Information Systems and Computer Engineering

Supervisors: Professor Vasco Miguel Gomes Nunes Manquinho
Professor Maria Inês Camarate de Campos Lynce de Faria

Examination Committee
Chairperson: Professor Nuno João Neves Mamede
Advisor: Professor Vasco Miguel Gomes Nunes Manquinho
Members of the Committee: Professor José Rui De Matos Figueira

May, 2018
Acknowledgments

I would like to thank my supervisors, Professor Vasco Manquinho and Professor Inês Lynce, for all the availability, guidance, support and for helping me everytime I need. Without your help I would never finish this dissertation.

I want to thank Professor Dídia Covas and Professor Laura Monteiro for all the patience to explain me all I needed to know about hydraulics, which was very helpful.

I would like to thank Professor José Figueira for the feedback and suggestions during the development of first part of this dissertation and for providing me with written support.

I would like to thank my parents for the opportunity they give me to have a college degree.

Finally I would like to thank all my friends for not forgetting me during my absence while I was writing this dissertation, specially Ana Sofia because she always believed in me and helped me when I was down.
Abstract

Water leakages are present in every Water Distribution Network (WDN). This problem can cause economic losses and health problems that affect the population and the environment. Hence, the necessity of having automatic and accurate techniques to reduce and detect these leakages opened a challenging area of research. Despite all the work already done trying to minimize the amount of ware losses, the seeking for improvement will always continue.

In this dissertation, the problem of water leakages is presented, followed by an overview on the most relevant work done in this area. This work includes methodologies for leakage detection with sensor placement and leakage reduction with usage of valves. Moreover, a new tool called WATimizer is presented. This tool is based on the algorithm Biogeography-based Optimization (BBO) and it was built to handle multi criteria optimization problems. In this case the tool will be used to minimize to distinct problems, one is to minimize pressure reduction in a WDN with the installation of Pressure Reducing Valves (PRVs), the second is minimize the costs associated with installation. This multiobjective problem is handled by two different methods, the Pareto approach and the weighted sum approach. At the end this two methods are compared, and the most efficient one is used to solve a study case that has been used to test the performance of the solutions in various researches.

Keywords: WATimizer, Epanet2.0, Multi-objective Optimization, Biogeography-based Optimization, Pareto Front, Weighted Sum.
Resumo

As Fugas de água estão presentes em todas as redes de distribuição. Este problema pode causar perdas económicas e problemas de saúde que afectam tanto a população como o ambiente. Assim, a necessidade de criar mecanismos automáticos que permitam detetar e reduzir estas fugas de água deu origem a uma grande e desafiante área de investigação. Apesar de todo o trabalho já realizado na tentativa de minimizar as perdas de água, a procura por soluções mais eficientes irá continuar.

Nesta dissertação é apresentado o problema das fugas de água, seguido de uma apresentação sobre todo o trabalho já realizado nesta área de investigação. Este trabalho inclui metodologias que permitem detetar fugas de água, através da utilização de sensores, e metodologias que permitem reduzir essas mesmas fugas através da utilização de válvulas. De seguida é apresentada uma nova ferramenta chamada WATimizer. Esta ferramenta tem por base o algoritmo BBO e foi construída para lidar com problemas de otimização multi-objetivo. Neste caso a ferramenta será usada para minimizar dois problemas distintos, o problema da minimização das pressões em redes de distribuição, com a instalação de válvulas redutoras de pressão, e o problema de minimização dos custos associados a essas instalações. Para lidar com este problema multi-objectivo, foram utilizadas duas abordagens distintas, a abordagem da fronteira de Pareto e a abordagem da soma de pesos. No final estas duas abordagens são comparadas em termos de eficiência, e a abordagem mais eficiente será utilizada para resolver um caso de estudo que tem sido utilizado em vários trabalhos anteriores como método de testar a eficiência.

Palavras-chave: WATimizer, Epanet2.0, Otimização Multi-Objetivo, Biogeography-based Optimization (BBO), Fronteira de Pareto, Soma de Pesos.
# Contents

<table>
<thead>
<tr>
<th>List of Tables</th>
<th>viii</th>
</tr>
</thead>
<tbody>
<tr>
<td>List of Figures</td>
<td>ix</td>
</tr>
</tbody>
</table>

## 1 Introduction
1.1 Motivation .................................................. 3
1.2 Contributions ................................................. 4
1.3 Document Organization ......................................... 4

## 2 Preliminaries
2.1 Water Leakage Problem ......................................... 5
2.2 Biogeography-based Optimization ............................. 7
2.3 Multi-Objective Optimization ................................. 7
2.3.1 Pareto Optimality Set ..................................... 8
2.3.2 Weighted Sum Method ..................................... 9

## 3 Related Work
3.1 Leakage Detection and Location ............................... 11
3.1.1 Sensor Placement .......................................... 12
3.1.2 Leakage Detection Procedure .............................. 12
3.2 Leakage Reduction ............................................ 15
3.2.1 Valve Setting Optimization ................................ 15
3.2.2 Valve Location and Setting Optimization ................ 16

## 4 WATimizer
4.1 Model .......................................................... 21
4.2 Hydraulic Simulator .......................................... 23
4.3 Nondominated Set .............................................. 23
4.4 Metrics Module ................................................ 24
4.5 Optimization Module ......................................... 25
4.5.1 Operators ................................................... 27
4.5.2 Ranking Solutions ......................................... 28
4.5.3 Elitism ....................................................... 29

## 5 Experimental Results
5.1 Network Instances ............................................. 32
5.2 Choosing the Best Configuration ............................. 33
5.3 WATimizer vs Non-Dominated Sorting Genetic Algorithm (NSGAII) ........................................... 34
5.4 Pareto efficiency vs Weighted sum efficiency .............. 36

vii
5.5 Case Study .......................................................... 38

6 Conclusion 43
  6.1 Achievements ......................................................... 43
  6.2 Limitations .......................................................... 43
  6.3 Future Work ......................................................... 44

Bibliography 45

A Additional Data 51
  A.1 Leakage detection literature ........................................ 52
  A.2 Leakage reduction literature ........................................ 53
  A.3 Pareto approach Results ........................................... 54
  A.4 Comparison Weighted sum approach vs Pareto approach ....... 57
# List of Tables

5.1 Network Configurations ................................................................. 32  
5.2 WATimizer algorithm. ................................................................. 33  
5.3 NSGAII algorithm. ................................................................. 34  
5.4 Emitter coefficient of each node of the network ................................ 39  
5.5 Results from different authors for Jowitt and Xu (1990) network. ........ 39  
5.6 Best solutions Generated by WATimizer ......................................... 40  
5.7 Diameter of each pipe where a PRV could be installed and the respective cost. 41  

A.1 Leakage detection bibliography ..................................................... 52  
A.2 Leakage reduction bibliography ..................................................... 53
## List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>Difference between field values (green dots) and simulated values (red line) of pressures at node 4.</td>
<td>6</td>
</tr>
<tr>
<td>2.2</td>
<td>Species model of a single habitat (Simon, 2008)</td>
<td>7</td>
</tr>
<tr>
<td>2.3</td>
<td>Representation of a set of criterion vectors in a feasible criterion space. The orange region is the criterion space region. The blue vectors are the dominated vectors. The red vectors are the supported nondominated vectors and the green vector is the unsupported nondominated vector.</td>
<td>9</td>
</tr>
<tr>
<td>3.1</td>
<td>Two Child Staggered Average Crossover Example (Vítkovský, Simpson, &amp; Lambert, 2000)</td>
<td>14</td>
</tr>
<tr>
<td>3.2</td>
<td>Relationship between pressure and leakage (Germanopoulos &amp; Jowitt, 1989)</td>
<td>16</td>
</tr>
<tr>
<td>3.3</td>
<td>Locations of pipe-set for valve location (Liberatore &amp; Sechi, 2009)</td>
<td>17</td>
</tr>
<tr>
<td>3.4</td>
<td>(a) Comparison of the performance of the analyzed hydraulic criteria; (b) Comparison of the effect of the reducing percentage over the solutions obtained. Both variables in USD (J. Saldarriaga &amp; Salcedo, 2015)</td>
<td>18</td>
</tr>
<tr>
<td>3.5</td>
<td>Representation of the genes of the i-th individual(J. Saldarriaga &amp; Salcedo, 2015)</td>
<td>18</td>
</tr>
<tr>
<td>3.6</td>
<td>Harmony search procedure for the optimization problem of PRVs location and setting(Paola, Galdiero, &amp; Giugni, 2017)</td>
<td>19</td>
</tr>
<tr>
<td>3.7</td>
<td>Logic flux of the instructions to carry out for each solution of the GA (Covelli, Cimorelli, et al., 2016)</td>
<td>20</td>
</tr>
<tr>
<td>4.1</td>
<td>Flux Diagram of WATimizer algorithm procedure.</td>
<td>22</td>
</tr>
<tr>
<td>4.2</td>
<td>Hypervolume for a set of Nondominated solutions and a reference point.</td>
<td>25</td>
</tr>
<tr>
<td>4.3</td>
<td>Inverted Generational Distance of a nondominated set concerning a best Pareto front approximation.</td>
<td>26</td>
</tr>
<tr>
<td>5.1</td>
<td>CaseStudy network with 25 nodes and 37 pipes.</td>
<td>32</td>
</tr>
<tr>
<td>5.2</td>
<td>Results for the Hypervolume and IGD for each of the algorithms. The upper plot, shows the evolution of the hypervolume over the 30 minute run, taken into account the median values of 6 different runs, where the orange line is the evolution for NSGAII and the blue line is the evolution for WATimizer. The lower plot shows the evolution of the IGD over the 30 minutes run, taken into account the median values of 6 different runs, where the orange line is the evolution for NSGAII and the blue line is the evolution for WATimizer (WATimizer).</td>
<td>35</td>
</tr>
<tr>
<td>5.3</td>
<td>Results for the Hypervolume and IGD for each of the algorithms for every 13 networks. The upper plot shows the Hypervolume of the final solution of each algorithm. The orange bares are the Hypervlume of NSGAII and the blue bars are the Hypervolume of WATimizer. The lower plot shows the IGD of the final solution of each algorithm. The orange bares are the IGD of NSGAII and the blue bars are the IGD of WATimizer.</td>
<td>36</td>
</tr>
</tbody>
</table>
5.4 Comparison between the solutions of WATimzer with a weighted sum approach and the solutions of NSGAII with a Pareto approach for every 13 networks. The upper plot shows the Hypervolume of the final solution of each algorithm. The orange bares are the Hypervolume of NSGAII and the blue bars are the Hypervolume of WATimzer. The lower plot shows the IGD of the final solution of each algorithm. The orange bares are the IGD of NSGAII and the blue bars are the IGD of WATimzer.

5.5 Comparison of the solutions generated from the Pareto approach and the weighted sum approach. The left plot shows the solutions for WATimzer, where the blue dots are the solutions generated by the weighted sum approach and the orange dots are the solutions generated by the Pareto approach. The right plot shows the solutions for NSGAII, where the blue dots are the solutions generated by the weighted sum approach and the orange dots are the solutions generated by the Pareto approach.

5.6 Comparison of the solutions generated from the Pareto approach and the weighted sum approach for the case study network introduced by Jowitt and Xu (1990). The blue dots correspond to the solutions generated with the weighted sum approach. The orange dots correspond to the solutions generated with a Pareto front approach.

A.1 Evolution for network 1 and 2 over 30 minutes. network 1 on the left, network 2 on the right

A.2 Evolution for network 3 and 4 over 30 minutes. network 3 on the left, network 4 on the right

A.3 Evolution for network 5 and 6 over 30 minutes. network 5 on the left, network 6 on the right

A.4 Evolution for network 7 and 9 over 30 minutes. network 7 on the left, network 9 on the right

A.5 Evolution for network 10 and 11 over 30 minutes. network 10 on the left, network 11 on the right

A.6 Evolution for network 12 and 13 over 30 minutes. network 12 on the left, network 13 on the right

A.7 Left plot shows the results generated by WATimzer. Right plot shows the solutions generated by NSGAII

A.8 Left plot shows the results generated by WATimzer. Right plot shows the solutions generated by NSGAII

A.9 Left plot shows the results generated by WATimzer. Right plot shows the solutions generated by NSGAII

A.10 Left plot shows the results generated by WATimzer. Right plot shows the solutions generated by NSGAII

A.11 Left plot shows the results generated by WATimzer. Right plot shows the solutions generated by NSGAII

A.12 Left plot shows the results generated by WATimzer. Right plot shows the solutions generated by NSGAII

A.13 Left plot shows the results generated by WATimzer. Right plot shows the solutions generated by NSGAII

A.14 Left plot shows the results generated by WATimzer. Right plot shows the solutions generated by NSGAII

A.15 Left plot shows the results generated by WATimzer. Right plot shows the solutions generated by NSGAII

A.16 Left plot shows the results generated by WATimzer. Right plot shows the solutions generated by NSGAII

A.17 Left plot shows the results generated by WATimzer. Right plot shows the solutions generated by NSGAII
A.18 Left plot shows the results generated by WATimizer. Right plot shows the solutions generated by NSGAII.
Chapter 1

Introduction

Water is one of the most valuable resources in the world (Henry, 2009) and so it is essential to take care of the infrastructures responsible for its distribution. As the population of the Earth grows, the consumption of water increases proportionally, not only due to the water consumed by the community but also due to Agricultural and Industrial usage (Vítkovský et al., 2000). Despite water being incredibly abundant (it covers about 70% of the Earth’s surface), the amount of water available for consumption is limited, which makes the problem of water losses a critical problem to solve.

Pipes are the most economical and safest mode of transportation for water. If properly maintained, they can last indefinitely with only negligible leaks. However, most of the water infrastructures are old, and at the time when they were built, the technology was limited. Therefore, the infrastructures were poorly designed and maintained. Nowadays the Water Distribution Network (WDN) infrastructures are damaged, and leakages occur in all of them.

Losses of water in the urban supply system reach values of about 50% at a global level, which is half of the water volume supplied. These losses occur due to the aging of the infrastructures which cause pipe bursts, design errors in the network, inadequate maintenance, and unauthorized water consumption (Nasirian, Maghrebi, & Yazdani, 2013).

One way to detect leaks is searching for them using various equipment and techniques, such as mass balance technique, acoustic equipment, thermography, ground-penetration-radar, trace-gas, and video. However, these methodologies have some problems associated. The equipment is expensive, the methodologies require specialized human labor, time, and the worst thing is that they have a limited accuracy (Nasirian et al., 2013).

With the advances in technology, it has become possible to monitor parts of the WDN, using field instrumentation like sensors. The information collected by these sensors can be used to compare the actual behavior of the network with the predicted behavior and, thus find irregularities caused by losses of water. The creation of computational models of the network is the fastest way to detect leakages, but they are extremely dependent on the data collected from the real network.

1.1 Motivation

Until detected and repaired, small water leakages can cause serious problems for both individuals and communities, such as environmental, economic and health problems. These problems will always grow as time passes.

The consumption of water increases each year due to the population growth. With this constant increase in water consumption, the population will end up suffering from lack of drinkable water, which
is a problem that will arise sooner if we let that this amount of losses of drinkable water happen every year. At an economic level, water losses represent a real threat to the organizations responsible for its distribution. With losses of 30%, it means a decrease of 30% in the profit of the organization which can lead to the decrease in salaries and an increase in water price.

As people become aware of water leakages problem, the answer to the question: “How can we mitigate the problem of water losses?” became a significant area of research.

### 1.2 Contributions

In this dissertation, the tool WATimizer (WATimizer) based on the Biogeography-based Optimization (BBO) algorithm is introduced as a new metaheuristic algorithm to solve the water leakage reduction problem. An evaluation of the BBO performance in solving this problem is made, alongside with a comparison of performance between WATimizer and Non-Dominated Sorting Genetic Algorithm (NSGAII). To evaluate the performance, a Pareto Front is generated, two metrics are calculated, the HyperVolume and the Inverted Generational Distance. To finish, a comparison between the performance of a Pareto Optimal approach and a Weighted Sum Method approach is made.

### 1.3 Document Organization

Chapter 2 introduces all the important concepts that will be used throughout this dissertation. It explains what is the water leakage problem and how the BBO algorithm works, as well as what is a multiobjective problem.

In Chapter 3 is presented some previous work. The first part is to present an overview of the work about water leakage detection. This part is divided into two subproblems. The first is to choose the best place to place sensors in order to improve the reliability of the data collected on the field, that is used to detect leakages. The second is the actual leakage detection part, which used data collected on the field to detect the leakages. The second part presents the problem of water leakage reduction which is divided into two parts. The first part is an overview of methodologies which the goal is to choose the best opening degree of each valve already present in the WDN in order to reduce pressures. The second part presents methodologies to choose the best locations to install new valves and optimize its opening degree.

In Chapter 4 is explained how the tool WATimizer is implemented. The tool is divided into four modules. One module which is the hydraulic simulator, used to get the values that will be evaluated. One module to calculate the nondominated set of solutions. One module with the optimization structure and one module to calculate the metrics that will be used to show the performance of the algorithm.

Chapter 5 presents the experimental results of applying WATimizer to a set of different problems with two different approaches, the Pareto approach and the weighted sum approach. The same problems were solved the well known algorithm NSGAII and the results were compared. In the end of this chapter the WATimizer tool is applied to a case study.

Chapter 6 gives the final conclusion about this work.
Chapter 2

Preliminaries

In this chapter a review of a few concepts that will be used throughout this dissertation is made. In Section 2.1, the water leakage problem and its relation to pressure is introduced. Then, in the following sections, the algorithm Biogeography-based Optimization (BBO) for multiobjective optimization and the concept of dominance are explained.

2.1 Water Leakage Problem

A Water Distribution Network (WDN) is a network responsible for supplying water to all the consumers. This network can be seen as an undirected graph, composed of pipes, which are the edges of the network, and junctions, which can be consuming nodes or distribution nodes, and those are the nodes of the network.

Definition 2.1.1 (Undirected graph). An undirected graph \( G \) is defined by a pair of finite sets \( G = (V, E) \), where \( V \) is a non-empty countable set of elements, called vertices or nodes, and \( E \) is a set of unordered pairs of different vertices \( v_1, v_2 \in V \), called edges or links.

All the components of a WDN have physical properties that change over time and are responsible for the good operation of the network. The pipes have an internal diameter, a length, and usually, these variables are well known from construction plans, but they also have an internal roughness coefficient which is a coefficient that affects the variation of energy of the fluid when it flows through a pipe. The energy conservation is given by (Lansey, 1996):

\[
\sum H_i - \sum H_j = \Delta H = \frac{4.73 L Q^{4.73}}{C^{1.852} D^{1.87}}
\]

(2.1)

where \( H_i \) and \( H_j \) are the energy at the extremity nodes of the pipe, \( L \) is the length of the pipe, \( Q \) is the pipe flow, \( C \) is the Hazen-Williams pipe roughness coefficient, and \( D \) is the pipe diameter.

The roughness coefficient of each pipe is not static, it varies over time as deterioration occurs (Lansey, 1996). The water that leaves the network, through some junction nodes, is typically controlled by devices that keep track of the amount of water consumed. The mass conservation in each junction is given by (Lansey, 1996):

\[
\sum Q_{in} - \sum Q_{out} = Q_{demand} + Q_{leak}
\]

(2.2)

where \( Q_{in} \) is the amount of water entering a junction, \( Q_{out} \) is the amount of water exiting a junction, \( Q_{demand} \) is the amount of water delivered to consumers and \( Q_{leak} \) is the amount of water lost due to untracked losses. In perfect networks, the last coefficient is negligible.
With Equation 2.1 and Equation 2.2 it is possible to create a model of a network to predict the values of pressure at each node and flow velocity at each edge. This model is created by a hydraulic simulator, such as EPANET (Rossman, 2000), that handles all the calculation and gives us the values of pressures and flows at each junction and pipe, respectively. Most of the times, the values collected on the field and the simulated values do not match. This discrepancy of values may occur due to the unknown roughness coefficients of some pipes, or there are losses of water somewhere in the network, which could cause a pressure drop. Figure 2.1 shows the difference between simulated pressure values and field pressure values of a given network. This difference can occur due to misleading data used to create the model, such as the roughness coefficient of each pipe, or the undetected water losses that lead to pressure drop.

The red line corresponds to the pressure values of node 4 given by the hydraulic simulator and the green dots correspond to the values of pressure measured on the field. The goal is that the red line matches the green dots so that the model corresponds to reality. This match can be done, manually, by a trial and error process, in which we add an extra demand to each node to reduce the pressure of the model. This additional demand added can be seen as a leak in that node or an adjacent pipe. The values of pressure and the amount of leakage are proportional (Germanopoulos & Jowitt, 1989) so as we reduce the pressure, we also reduce the leakage. To lessen the pressure on delivery points, it is necessary the installation of Pressure Reducing Valve (PRV). These valves have a cost of installation that varies according to the diameter of each valve. Hence a trade-off between the installation costs and the reduction of the amount of water lost must be made. These valves have different configurations that allow different values of pressure downstream. The set of configured valves must obey the constraint of minimum pressures at each node. This minimum pressure corresponds to the pressure necessary to deliver water to consumers.
2.2 Biogeography-based Optimization

Biogeography-based Optimization (BBO) (Simon, 2008) is a metaheuristic algorithm for solving optimization problems, and it will be used to solve the problem of pressure reduction. This algorithm was motivated by biogeography, which is the process of evolution and geographical distribution of species through time and space. A geographically isolated habitat is called *island*. These islands are composed of independent conditions which can be, temperature, rainfall, amount food, among others, that define that habitat suitability. To a vector of these independent conditions is called Suitability Index Variables (SIVs) and to the goodness of an island for species to migrate is called High Suitable Index (HSI), which is similar to the fitness of a Genetic Algorithm (GA). The goal of BBO algorithm is to simulate the behavior of migration of species, which includes the extinction and the appearance of new species. Habitats with high HSI tend to have many species. Hence, the rate of immigration of species to that habitat is low, whereas the emigration rate is high. Habitats with a low HSI tend to have a much lower amount of species, and so, the immigration rate is high, and the emigration rate is low. The relationship between the number of species and rate of immigration and emigration is presented Figure 2.2.

**Definition 2.2.1 (Migration).** The migration operator \( M_{ig} : I \times I \rightarrow I \) selects two islands \( I_j, I_k \) from an archipelago \( A \), \( I_j \) being the emigration island with a SIV vector \([I_{j1}, I_{j2}, ..., I_{jn}]\) and \( I_k \) being the immigration island with a SIV vector \([I_{k1}, I_{k2}, ..., I_{kn}]\) and migrate an SIV value \( I_{jm} \) to \( I_{km} \), \( 1 \leq m \leq n \).

**Definition 2.2.2 (Mutation).** The mutation operator \( M_{ut}(p) : I \rightarrow I \) is a probabilistic operator that modifies a SIV of an island \( I \). The mutation operator is described as follows:

1. Choose whether to mutate an island according to the probability \( p \);
2. Choose uniformly the SIV to mutate.

2.3 Multi-Objective Optimization

A multi-objective optimization is an area of multiple criteria decision making with the main goal of optimizing a set of objective functions, that are mutually contradictory, this is, as we improve one objective value, we deteriorate, at least, the value of another objective. There are various approaches to deal with
this problem. In this dissertation we will introduce two very different approaches, the Pareto Optimality Method and the Weighted Sum Method and a comparison between them, regarding efficiency, is done.

The Pareto Optimality Method and the Weighted Sum Method are very different from each other. While the former gives a set of possible solutions which correspond to the solutions in the nondominated set, the latter convert a multi-objective problem into a single-objective problem using a scalarization function. Thus, only one optimal solution can be found.

For the sake of the following definitions we assume that we want to minimize all the objective functions.

**Definition 2.3.1 (Multi-Objective optimization Problem).** Let \( \mathbb{R}^n \) be the decision space, and \( X \subset \mathbb{R}^n \) a feasible region in the decision space. Let \( \mathbb{R}^k \) be the criterion space, and \( Z \subset \mathbb{R}^k \) a feasible region in the criterion space.

Given a feasible region on the criterion space \( Z \) and a decision space \( X \), a Multi-Objective optimization Problem consists of finding a solution vector \( \bar{x} \in X \) that optimizes the vector function \( \bar{f}(\bar{x}) \in Z \).

### 2.3.1 Pareto Optimality Set

To understand what is a Pareto Optimal Set it is important to understand the definition of dominance and Nondominated criterion vector.

**Definition 2.3.2 (Dominance).** Let \( \bar{z}, \bar{z}' \in Z \) denote two solution vectors. Then, \( \bar{z} \) dominates \( \bar{z}' \) if and only if:

1. \( f_i(\bar{z}) \leq f_i(\bar{z}') \) for all functions \( i \) in \( \bar{f} \);
2. There is at least one \( i \) such that \( f_i(\bar{z}) < f_i(\bar{z}') \).

**Definition 2.3.3 (Nondominated Criterion Vector).** Let \( \bar{z} \in Z \). Then, \( \bar{z} \) is nondominated if and only if there does not exist another \( z \in Z \) such that \( z \leq \bar{z} \) and \( z \neq \bar{z} \). Otherwise, \( \bar{z} \) is a dominated criterion vector.

**Definition 2.3.4 (Nondominated Set).** The nondominated set is the set of all nondominated criterion vectors.

The nondominated set of solutions is composed of two types of vectors:

- Supported Nondominated Vectors
- Unsupported Nondominated Vectors.

**Definition 2.3.5 (Supported Nondominated Vector).** Vector \( \bar{z} \in Z \) is a supported nondominated criterion vector if and only if \( \bar{z} \in N \) and \( \bar{z} \in \text{bd}Z \leq \), where \( \text{bd}Z \leq \) corresponds to the border of the feasible region in the criterion space \( Z \).

**Definition 2.3.6 (Unsupported Nondominated Vector).** Vector \( \bar{z} \in Z \) is an unsupported nondominated criterion vector if and only if \( \bar{z} \in N \) and \( \bar{z} \in \text{int}Z \leq \), where \( \text{int}Z \leq \) corresponds to the interior of the feasible region in the criterion space \( Z \).

The supported nondominated vectors correspond to the solutions that belong to the border of the criterion space \( Z(\text{bd}Z \leq) \), which corresponds to the most efficient solutions whereas the unsupported nondominated vectors belong to the interior region of the criterion space \( Z(\text{int}Z \leq) \). Figure 2.3 shows an orange criterion space and a set of dominated and supported and unsupported nondominated vectors.
A Pareto optimal set is a set of criterion vectors that are nondominated with respect to all other vectors in the search space. Due to the computationally challenging task of computing the true Pareto set, the nondominated set of solutions obtained, correspond to an approximation of the true Pareto front (Figueira, Greco, & Ehrgott, 2005).

**Definition 2.3.7** (Pareto Set). A Pareto set \( P \in Z \) is a set of all \( z \in Z \) that are nondominated.

### 2.3.2 Weighted Sum Method

The simplest approach for solving multi-objective optimization problems is by scalarization, which consists in creating a scalarization function that is based on all objective functions that are being optimized. The weighted sum method is a scalarization method, commonly used to solve multi-criteria problems (Odu & Charles-Owaba, 2013).

**Definition 2.3.8** (Scalarizing Function). Let \( F \) be a scalarizing function, and let \( f = [f_1, ..., f_n] \) be a set of objective functions. \( F \) is given by

\[
\sum_{i=1}^{n} w_i f_i
\]

where \( w_i \) is the weight assigned to objective function \( f_i \), and \( n \) is the number of objective functions.

The usage of different weights for each optimization function has the advantage of assigning different degrees of importance to each one of them, instead of all having the same importance.
Chapter 3

Related Work

Water Leakage is a problem that most of the time passes unnoticed to the public because all the infrastructures are underground, which makes the problem of water loss invisible.

There are several negative impacts of water losses for both, the water providers and society itself such as economic losses, environmental damage, energy losses, and also social consequences such as property damage, destruction of crops, deterioration of health.

These water leakages can be classified as apparent losses which are non-physical losses, and they are related to customer’s meter inaccuracies and illegal consumption, or real losses which include water storage overflow and leakages due to bursts in the water infrastructures (American Water Work Association, 2012; McKenzie & Seago, 2005; Lambert, 2002). Leakage control can be classified as passive leakage control or active leakage control. Passive leak control implies repairing or replacing parts of the physical water network only for visible and reported leaks by customers. Active leakage control is the regular testing of the network infrastructure in order to find irregularities that can be caused by invisible leakages. In order to reduce real losses in water distribution networks, it is important to adopt an active leakage control policy. However, this is not a simple task because sometimes, minor losses can be difficult to locate.

The problem of leakage control has been investigated since the 80’s. This investigation includes work on leakage detection and leakage reduction. The development of an efficient computational procedure to control the amount of non-revenue water is the main goal of these investigations.

This Chapter is divided into two Sections that describe two different methodologies to control the amount of non-revenue water delivered. The first Section is called Leakage Detection and Location for which the main goal is to develop computational and mathematical models to locate/detect the existence of a leak in a network. The second Section is called Leakage Reduction and focus on developing computational models to improve the design of a water network distribution in order to minimize the amount of leakage.

3.1 Leakage Detection and Location

The traditional approach to detect leakages relies on a passive methodology, but this only works when the leak is visible (Casillas, Puig, Garza-Castanon, & Rosich, 2013), which most of the times is not the case. More recently, the development of acoustic techniques (Khulief, Khalifa, Mansour, & Habib, 2012) has allowed the detection of invisible leakage and consequently its reparation. The problem with this methodology is that water distribution networks are large-scale networks and so the application of these acoustic techniques is very expensive, time-consuming and requires specialized human labor (Casillas
et al., 2013). All these disadvantages encourage the development of automatic and accurate leakage location techniques.

To locate leakages is necessary to use network monitoring. The monitoring of pressure and flow around the network is extremely common and important because the data collected by these type of sensors represent the state of the real network on specific points which will be used to create computational models. Thus, the importance of having an appropriate location for sensors to collect reliable information from the network (Behzadian, Kapelan, Savic, & Ardeshir, 2009) is the difference between a successful leakage detection or a failure.

The data collected from the real Water Distribution Network (WDN) is an essential factor that determines if an accurate computational model of the network can be built with success or not. In fact, the accuracy of the computational model is quite dependent on the number and location of sensors in the network (Ribeiro, Sousa, Sá Marques, & Simões, 2015; Sousa, Ribeiro, Muranho, & Sá Marques, 2015).

3.1.1 Sensor Placement

Goulet, Coutu, and Smith (2013) showed that the performance of leakage detection improves if we increase the number of sensors on the network because with more reliable data the model of the network will be much more accurate. However, sensors are expensive, so a trade-off between cost and data reliability must be taken into account; (Candelieri, Conti, & Archetti, 2014) proposed an approach for selecting the best node locations for sensor placement by identifying the best trade-off between model reliability and deployment costs. The proposed solution relies on clustering a dataset of leakage scenarios. This clustering is applied to the variations in pressure and flow at each junction and each pipe. The medoids of each cluster, which corresponds to the center of a cluster, are chosen to be the monitoring points; Rosich, Sarrate, and Nejjari (2012) proposed an iterative technique with the main goal of identifying essencial sensors in the network in order to minimize the costs.

Christodoulou et al. (2013) used an entropy-based approach for efficient and economically viable water loss incident; (Casillas et al., 2013) proposed a new approach to the problem formulating the sensor placement as an integer optimization problem concerning the minimization of the number of non-isolable leaks, according to the “isolability criteria” introduced; (Steffelbauer, Neumayer, Günther, & Fuchs-Hanusch, 2014) used a Monte Carlo simulation.

Ribeiro et al. (2015) used an adaptation of the TrustRank algorithm (traditionally used to rate the quality of websites), for choosing the best places to locate pressure transducers. This algorithm selects the junctions based on the water demand. This water demand is like a trust transmission which allows the selection of the best node for pressure transducer location. The trust transmission is given by

$$T(d) = \frac{R(s)}{N}$$

where $T(d)$ is the trust transmission of the upstream node $s$ to the downstream node $d$, $R(s)$ is the trust score of the upstream node and $N$ is the number of out ways from the node $s$.

3.1.2 Leakage Detection Procedure

Different methodologies were developed in order to solve this water leakage problem, and they are, regularly, based on the analysis of flow and pressure measurements. Some of these methodologies include artificial neural networks (Gabrys & Bargiela, 1999), Bayesian networks (Poulakis, Valougeorgis, & Papadimitriou, 2003; Qi et al., 2014), model-falsification methodology (Goulet et al., 2013); non-linear Kalman filter (Jung & Lansey, 2014). Some of these approaches can handle uncertainties arising from
modeling errors and measurement noises.

The data set of flows and pressures acquired from the real network and the laws of mass and energy conservation provide the necessary conditions to calibrate a computational model of the network through a hydraulic simulator. It is possible to use the hydraulic simulator to locate leakages in steady-state or unsteady-state conditions. The inverse transient analysis is an unsteady-state analysis for leak detection (Covas & Ramos, 2010; Soares, Covas, & Reis, 2011; Vítkovský et al., 2000) and it consists in introducing transients in the network model and analyze the variations in pressure to locate the leaks. The advantages of this technique over the steady-state analysis is that it provides much more data (Vítkovský et al., 2000). A more detailed review of transient-based leakage detection methods can be found in (Colombo, Lee, & Karney, 2009).

The usage of a classification learning, like support vector machines, together with a hydraulic simulator has also been developed to identify leaks at junctions. Candelieri, Soldi, Conti, and Archetti (2014) used spectral clustering and a support vector machine to solve the problem of leakage detection/location. The idea is to generate various leakage scenarios and use a network-base spectral clustering to group together similar scenarios, regarding variations in pressure and flow. After that, a support vector machine is used to discover the relation between variations in pressure and flow in a limited set of probably leaky pipes. This work reports a comparison between three different implementations of support vector machines.

The problem of leakage detection is, in fact, a model calibration problem and this problem consists in adjusting the model characteristics and parameters so that the model predicted flows and pressures approximate as much as possible the values observed in the field (Wu et al., 2002). With the changes in correlation between pressure data from the field and the simulated one, it may be possible to detect leakages. Usually, this calibration is obtained by minimizing an objective function of the absolute values of the sum of differences like:

\[
F(x) = \sum_{i=1}^{NPT} \left| \frac{H_{obs_i} - H_{sim_i}}{H_{obs_i}} \right| \tag{3.2}
\]

where \(NPT\) denotes the number of observation points, \(H_{obs_i}\) is the i-th observed hydraulic head and \(H_{sim_i}\) is the i-th model simulated hydraulic head.

Approaches that use evolutionary algorithms to solve the network calibration problem use a variation of the optimization function described above.

One of the most used evolutionary algorithms is the genetic algorithm (Vítkovský et al., 2000; Wu et al., 2002; Behzadian, Kapelan, Savic, & Ardeshir, 2007; Behzadian et al., 2009; Covas & Ramos, 2010; Soares et al., 2011; Lijuan, Hongwei, & Hui, 2012; Mambretti & Orsi, 2012; Casillas et al., 2013; Nasirian et al., 2013).

Lijuan et al. (2012) proposed a solution based on a genetic algorithm to detected leakages with EPANET as the hydraulic simulator. To detect a leak is necessary to create a computational hydraulic model that represents the real network. This model has a set of parameters \(\theta\) which is divided into two subsets, \(\theta_1\) and \(\theta_2\), that denote the locations of leakage in the network (junctions) and the leakage volume, respectively. The objective function receives a set of simulated pressures and compares it to a set of pressures measured on the field. \(\theta_1\) can be seen as a set of simulated pressures on each monitored junction, and these values are correlated with the set of values of \(\theta_2\), which is the leakage amount on each junction. To minimize the objective function is necessary adjusting the values of \(\theta_2\) so that the values of \(\theta_1\) match the set of pressures measured in the field. The genetic algorithm from Toolbox of MATLAB is used to find these optimal values of \(\theta_2\) as it updates the values of \(\theta_1\) after each generation.

Vítkovský et al. (2000) used a genetic algorithm technique together with an inverse transient method
string is usually by discrete values within a continuous range. For a discrete coding scheme, a lookup table relating a code (either binary or integer) with a corresponding discrete value (for example, \( f = 0.015, 0.016, \ldots \)) is necessary. To suit the continuous nature of the friction factor and lumped leak coefficient values, a continuous coding scheme has been developed for implementation of the GA in this paper. The initial population of strings of unknown friction factors and leaks in the GA process is generated randomly using values across the continuous range between the selected lower and upper bounds. The ranges for the seeding of the friction factors and the lumped leak coefficients were from 0.01 to 0.05 and from 0.0 to 0.001 respectively.

A New Crossover Operator

A new crossover operator referred to as two child staggered average crossover (Vítkovský and Simpson, 1997) has been developed to exploit the continuous nature of the variables being represented by the string. This operator is based upon a crossover operator first used by Savic and Walters (1995) named “one child average crossover”. The schematic of this new crossover operator with three crossover points is shown in Fig. 3.1. This operator averages the friction factor value from a pair of strings for corresponding bits in the same positions of the parent strings while the original bits are used at the other locations. This operator differs from the Savic and Walters’ scheme in that it maintains some of the original genetic information in each parent string.

\[
\text{gene}_{\text{NEW}} = \text{gene}_{\text{OLD}} + \text{step}\_\text{size} \times (2 \times \text{RND} - 1)
\]

where \( \text{RND} \) is a random value between 0 and 1.

A methodology for leakage location based on simulated annealing was proposed by (Ribeiro et al., 2015). The objective function to minimize is the following one:

\[
F(x) = \sum_{i=1}^{NPT} |(\frac{P}{\gamma})_i^{\text{measured}} - (\frac{P}{\gamma})_i^{\text{simulated}}|
\]

where \( NPT \) is the number of monitored nodes, \( (\frac{P}{\gamma})_i^{\text{measured}} \) is the pressure per unit volume measured in the field at node \( i \) and \( (\frac{P}{\gamma})_i^{\text{simulated}} \) is the pressure per unit volume calculated. The pseudo code of the simulated annealing algorithm is presented in Algorithm 3.1.

The algorithm begins with an initial solution (line 1) and an initial temperature. The initial solution is generated splitting the total leakage flow into different parts, and assigning each part to a selected pipe. After the initial solution is created, the other candidate solutions are generated by one of these two processes: diversification mechanism or concentration mechanism (line 4). Diversification mechanism consists in assigning a percentage \( p \) of the total leakage of one pipe to one of its adjacent pipe of the network; concentration mechanism consists in concentrate in one pipe all the leaks of its adjacent pipes. After this, the hydraulic simulator is used to simulate the behavior of the model, and the pressure values returned by the simulator are used to test the objective function given by Equation 3.5. The next solution is chosen based on the Metropolis criterion: if the candidate solution is better the current solution (line 6) or it satisfies the condition in line 10, the candidate solution becomes the current solution, otherwise, it

\[
\text{*} = \text{the average of the parents corresponding genes}
\]

Figure 3.1: Two Child Staggered Average Crossover Example (Vítkovský et al., 2000)

in order to detect leaks and friction factors in water distribution systems. The goal of the genetic algorithm is to minimize the following equation:

\[
-\sum_{i=1}^{M} |H_i^* - H_i|
\]

where \( H_i^* \) is the measured hydraulic grade line and the \( H_i \) is the hydraulic grade line of the model.

This genetic algorithm uses a new crossover operator named “one child average crossover”. This new crossover is represented in Figure 3.1. The crossover operator takes two strings, one of each parent and, either averages the friction factor of two corresponding values of each string or keep the original values of one of the parents for the new individual.

The mutation operator is also new. It alters the gene to a new value within a range of a \textit{step\_size}. Equation 3.4 shows how the new gene value is calculated.

\[
\text{gene}_{\text{NEW}} = \text{gene}_{\text{OLD}} + \text{step}\_\text{size}(2 \times \text{RND} - 1)
\]

where \( \text{RND} \) is a random value between 0 and 1.

A methodology for leakage location based on simulated annealing was proposed by (Ribeiro et al., 2015). The objective function to minimize is the following one:

\[
F(x) = \sum_{i=1}^{NPT} |(\frac{P}{\gamma})_i^{\text{measured}} - (\frac{P}{\gamma})_i^{\text{simulated}}|
\]

where \( NPT \) is the number of monitored nodes, \( (\frac{P}{\gamma})_i^{\text{measured}} \) is the pressure per unit volume measured in the field at node \( i \) and \( (\frac{P}{\gamma})_i^{\text{simulated}} \) is the pressure per unit volume calculated. The pseudo code of the simulated annealing algorithm is presented in Algorithm 3.1.

The algorithm begins with an initial solution (line 1) and an initial temperature. The initial solution is generated splitting the total leakage flow into different parts, and assigning each part to a selected pipe. After the initial solution is created, the other candidate solutions are generated by one of these two processes: diversification mechanism or concentration mechanism (line 4). Diversification mechanism consists in assigning a percentage \( p \) of the total leakage of one pipe to one of its adjacent pipe of the network; concentration mechanism consists in concentrate in one pipe all the leaks of its adjacent pipes. After this, the hydraulic simulator is used to simulate the behavior of the model, and the pressure values returned by the simulator are used to test the objective function given by Equation 3.5. The next solution is chosen based on the Metropolis criterion: if the candidate solution is better the current solution (line 6) or it satisfies the condition in line 10, the candidate solution becomes the current solution, otherwise, it
### Algorithm 3.1: Simulated Annealing

<table>
<thead>
<tr>
<th>Line</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data:</td>
<td>$max_{it}, max_{tmp}$</td>
</tr>
<tr>
<td>Result:</td>
<td>$S_{best}$</td>
</tr>
<tr>
<td>1</td>
<td>$S_{current} \leftarrow \text{init_population}();$</td>
</tr>
<tr>
<td>2</td>
<td>$S_{best} \leftarrow S_{current};$</td>
</tr>
<tr>
<td>while Criteria not met do</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>$S_i \leftarrow \text{choose_neighbour_solution}(S_{current});$</td>
</tr>
<tr>
<td>5</td>
<td>$\text{temp}<em>{current} \leftarrow \text{calculate_temperature}(S_i, max</em>{tmp});$</td>
</tr>
<tr>
<td>6</td>
<td>if $\text{cost}(S_i) \leq \text{cost}(S_{current})$ then</td>
</tr>
<tr>
<td>7</td>
<td>$S_{current} \leftarrow S_i;$</td>
</tr>
<tr>
<td>8</td>
<td>if $\text{cost}(S_i) \leq \text{cost}(S_{best})$ then</td>
</tr>
<tr>
<td>9</td>
<td>$S_{best} \leftarrow S_i;$</td>
</tr>
<tr>
<td>10</td>
<td>if $\text{Exp}(\frac{\text{Cost}<em>{S</em>{current}} - \text{Cost}<em>{S_i}}{\text{temp}</em>{current}}) &gt; \text{Rand}()$ then</td>
</tr>
<tr>
<td>11</td>
<td>$S_{current} \leftarrow S_i;$</td>
</tr>
</tbody>
</table>

is rejected. This process repeats until a stopping criterion is met. More recently (Huang, Lin, & Yeh, 2016) proposed another approach based on a combination of transient flow simulation and simulated annealing.

## 3.2 Leakage Reduction

The best way to mitigate the problem of leakage in a water distribution network is to reduce the pressure at junctions. The relationship between pressure and leakage is represented in Figure 3.2. To reduce the pressure at junctions, the usage of certain types of valves are fundamental and different problems must be solved:

- Optimize the number of valves used;
- Optimize the location of each valve in the network;
- Optimize the opening of each valve (setting).

The pressure at each junction is responsible for delivering water to consumers, and thus, a minimum pressure at each junction is required. This minimum pressure varies from region to region.

### 3.2.1 Valve Setting Optimization

The problem of leakage reduction is addressed since 1989 when Germanopoulos and Jowitt (1989), Jowitt and Xu (1990) and Vairavamoorthy and Lumbers (1998) proposed a solution based on the optimal control valve settings. These control valves are used to reduce the pressure in the system and consequently minimizing the leakages; Vairavamoorthy and Lumbers (1998) proposed a solution based on sequential quadratic programming to find the optimal valve setting.

For pressure regulation, Pressure Reducing Valve (PRV) are used. This type of valve has various different settings that regulate the pressure downstream. (Karadirek, Kara, Yilmaz, Muhammetoglu, & Muhammetoglu, 2012) proposed an optimization methodology to optimize the setting of each pressure reduction valves to reduce water losses in a water distribution network in Turkey, using Epanet as the hydraulic simulator and considering only minimum night flows, which is the best condition to detect leakages.
In 2006 Geem (2006) used a harmonic search algorithm for the first time to solve a water distribution design problem. This metaheuristics algorithm shows a competitive performance in comparison with other traditional metaheuristics algorithms such as the genetic algorithm, simulated annealing and tabu-search (De Paola, Galdiero, & Giugni, 2016). The harmonic search algorithm was developed by Geem, Kim, and Loganathan (2001) and it uses the experience of the Jazz musicians improvisation as the evolutionary approach to minimize or maximize a specific objective function. The main idea is that we have a set of variables and each one represents a musician. Those variables have a range of possible discrete values that represents the music instrument’s pitch, and the music harmony corresponds to the set of values of each variable at a certain time step (Geem, 2010).

### 3.2.2 Valve Location and Setting Optimization

Other research works propose a more complex algorithm that not only solves the setting optimization for each valve but also optimize their location in the network.

Reis, Porto, and Chaudhry (1997) proposed a solution for optimization of valve positioning and setting using a genetic algorithm for optimizing both problems.
Araujo, Ramos, and Coelho (2006) used EPANET as the hydraulic simulator for its optimization procedure. The authors proposed a genetic algorithm based on a two-phase method:

1. Optimize the number and location of valves;
2. Adjustment of valve opening degree.

The genetic algorithm was slightly changed and the application of elitism was introduced. In order to optimize the valve locations, pseudo-valves in each pipe are considered. These pseudo-valves are obtained by the simulation of an additional roughness on each pipe in order to minimize the pressure at each junction of the network. The location of each pseudo-valve is generated randomly for the system to reach the minimum possible pressure. An optimization procedure to optimize valve settings begins when the best solution for valve locations is found. (Liberatore & Sechi, 2009) also divided his solution into two phases. In the first phase, the number of valves and a set of pipes were chosen as potential candidates to locate the valves, based on the pressure of its extremity junctions. An example of the best junctions to locate the valves is given in Figure 3.3.

This first phase reduces the search space to only a restricted set of pipes which reduces the complexity of the second phase. In the second phase, an optimization procedure is used to find the optimal location for valves. This optimization procedure is based on a metaheuristic optimization approach called Scatter-Search. The Scatter-Search metaheuristic algorithm is used for global optimization like genetic algorithms. While the latter uses a population and a randomized recombination of existent solutions to evolve over time, the former uses the adaptive memory and associated memory-exploiting mechanisms to create new solutions. It accepts worst moves to avoid being stuck in a local minimum or maximum, and it keeps track of prohibitions to discourage the search from coming back to previously-visited solutions.

Creaco and Pezzinga (2014) proposed a Hybrid multi-objective algorithm that has pipe diameters and valves positions and settings as decision variables. The goal is to find the best location for the valves and pipe replacements for minimizing the maximum undelivered water. The authors also explain the advantages of considering the existence of isolation valves in the pipe and how they improve the control valve regulation.

One year later, (Creaco & Pezzinga, 2015) showed that the combination of a local search algorithm and a genetic algorithm can be used to divide the problem into two sub-spaces. The first sub-space is made up of discontinuous integer variable that represents the location where the valves can be installed. This problem is solved by a genetic algorithm. The second sub-space is made up of continuous real variables which create an infinite search space, and thus, it is a hard problem for a genetic algorithm to solve, so and iterated linear programming algorithm can be used to solve this problem.

(J. Saldarriaga & Salcedo, 2015) proposed a solution based on a well-known meta-heuristic algorithm NSGA-II with a search space reduction (among all pipes choose the best minimal set of pipes to install the valves) methodology in order to improve the performance of the procedure. In order to choose the
In the presented approach, three hydraulic criteria were tested in order to establish which one of them reduces the solution space and reaches the best results, a theoretical and practical study. For the selection of the hydraulic criteria that reduces the solution space, the required computational times were implemented in previous researches, and they are:

• Sectorization Criterion (J. Saldarriaga & Salcedo, 2015);
• Specific Power, measuring the energy dissipation in the network (J. G. Saldarriaga, Ochoa, Moreno, Romero, & Cortés, 2010);
• Pipe-Index (Vairavamoorthy & Ali, 2005);

The performance of these three methodologies is represented in Figure 3.4(a). In Figure 3.4(b) the difference between reducing the number of possible pipes for possible valve location of about 50% and 90% is shown. Both sectorization and specific power have good trade-offs between costs of the valves and profit for leakage reduction, but due to the complexity of the former, the chosen criterion to be used was the specific power. This criterion is based on the loss of energy when water flows through a pipe.

After choosing the best methodology for space reduction, the NSGA-II from MATLAB R2013 with a crossover operator based on the Simulated Binary Crossover and a mutation operator based on the Polynomial mutation is used to determine the optimal location and setting for each valve (J. Saldarriaga & Salcedo, 2015). The representation of genes for the genetic algorithm optimization is represented in Figure 3.5.

The encoding is divided into three parts: $C_{i,1}$ represents the pipe number where valve $i$ is located, $\Delta D_{i,2}$ represents an increment on the setting of valve $i$ during the day time and $\Delta N_{i,2}$ represents an increment on the setting of valve $i$ during the night time.

This year (Paola et al., 2017) proposed a double harmonic component for solving both the location and the setting of each valve in the network. The steps are presented in Figure 3.6.
(Covelli, Cimorelli, et al., 2016) proposed a solution to obtain the optimal trade-off between the cost of PRV installation and the benefits of leakage reduction. It tries to minimize the number of valves installed in the network to optimize its positioning and setting; (Covelli, Cozzolino, Cimorelli, Della Morte, & Pianese, 2016) proposed a methodology to find the optimal number, location, and setting of PRVs. The proposed methodology uses a genetic algorithm together with a physical modeling of leakage from joints and a simplified and realistic hydraulic simulation. The genetic algorithm to solve the valve positioning and setting is based on the binary Gray encoding for the properties of the individuals; the selection procedure is based on an exponential rank selection; the crossover is uniform with a predefined probability; the mutation is a bitwise mutation process with a given probability. A partial elitism technique is also used, and this allows up to two elements to be preserved for the next step of the solution (Covelli, Cimorelli, et al., 2016). The fitness function, given by Equation 3.6 used by (Covelli, Cozzolino, et al., 2016) was already used in (Covelli, Cimorelli, et al., 2016) and is divided into two parts.

\[ FF = C + P \]  \hspace{1cm} (3.6)

where \( C \) represents the sum of the costs associated with the loss of earnings as a result of leakage in pipes and the costs of purchasing and installation of PRVs. \( P \) represents a penalty in costs if the min-
imum pressure constraint is not satisfied for all the simulated pressures. The procedure is represented in Figure 3.7.

Figure 3.7: Logic flux of the instructions to carry out for each solution of the GA (Covelli, Cimorelli, et al., 2016).

More recently the harmonic search algorithm was proposed by (De Paola, Giugni, & Portolano, 2017) to solve the pressure reducing valve setting optimization. The goal is to minimize an objective function subject to a set of constraints described in (De Paola et al., 2017). A penalty function is introduced to increase the efficiency of the algorithm.

A summary of all the work done and the methodologies used is present in Appendix A.1 and Appendix A.2.
Chapter 4

WATimizer

In this chapter, a description of the architecture of the proposed tool is presented. The optimization procedure consists in finding the edges of a network to install Pressure Reducing Valves (PRVs), with the main goal of minimizing both the pressure all over the network and the costs of equipment and installation. This architecture includes four distinct modules: an optimization module, that uses the algorithm described in section 2.2; a hydraulic simulator, to calculate the new pressures values at each iteration; a module to calculate the nondominated set of solutions for each run of the algorithm; a module to calculate the Hypervolume and the Inverted Generational Distance (IGD) which are the commonly used measures to compare the output provided by algorithms for multi-objective optimization problems.

4.1 Model

Let $S = A_1, A_2, ..., A_k$ denote a system of Archipelagos. $A_i = \{[I_{i1}, I_{i2}, ..., I_{ij}], O, C\}$ represents an Archipelago, which is composed of a vector of Islands $I_{i1}, ..., I_{ij}$, an objective function $O$, and a constraint $C$.

The Islands are vectors of real values which are called Suitability Index Variable (SIV) values. These values correspond to the roughness coefficient of each pipe of the network. The roughness coefficient is a physical characteristic of pipes which are measured by the smoothness of its internal area. The pressure on an adjacent node decreases when the value of the roughness coefficient of its pipe decreases too. Hence, decreasing the values of roughness coefficient can be used to decrease the values of pressure in the network.

The alteration of the roughness coefficient of a pipe can be used to simulate a valve installed in that same pipe, with an opening degree. As the roughness coefficient decreases, even more, the opening degree of the installed valve also decreases and so, the pressure of the downstream node, decreases too.

In this work, we consider that all the pipes can have a valve installed. The opening degree of each valve can be adjusted by varying the roughness coefficient. This can be expressed as a percentage of the initial roughness coefficient of each pipe. For example, for a given pipe, if the percentage of the roughness coefficient is 100%, it corresponds to having a valve fully opened in that pipe and so, in that case, that valve is not considered. If the percentage of the roughness coefficient is 20% of the original roughness, it corresponds to having a valve 20% opened in that pipe, and so, it is a good approach to consider a valve in that pipe.

To choose what pipes should have a valve, it was considered a fixed threshold, that represents the
boundary values for what should be considered valves or not. In WATimizer (WATimizer) this threshold is called VALVE_THRESHOLD. The VALVE_THRESHOLD is a roughness coefficient percentage in which, pipes with lower roughness than its initial roughness coefficient multiplied by the threshold are pipes with valves, and pipes with roughness higher than that are non valve pipes. Only pipes with a roughness coefficient with 50% or less than its original roughness, are considered to be real valves, because having valves which are open, more than 50%, will increase the cost and then they are not effective in decreasing the pressure.

Figure 4.1 shows the flow of the optimization procedure of the algorithm.

Step 1: Initialize Archipelagos
The optimization procedure begins with the initialization of each archipelago, which corresponds to assign randomly values of new roughness coefficient to each island. The range of these values are between 0 and the initial value of the roughness.

Given an archipelago $A$, and a vector $R = [r_1, r_2, ..., r_n]$ of initial roughness coefficients, for island $i$, an island is a vector of SIV values $[SIV_1, SIV_2, ..., SIV_n]$, with $SIV_k \in [0, \frac{r_k}{2}] \cup \{r_k\}$, $k \in [1, n]$.

Step 2: Simulate Pressures
In this step, will be calculated a vector of pressures from the corresponding SIV values of each island. Given a vector $SIV = [SIV_1, SIV_2, ..., SIV_n]$ and a hydraulic simulator $S : SIV \rightarrow P$, the hydraulic simulator receive a vector of $SIV$ values and return a vector $P$ of pressures for each node of the network.

Step 3: Calculate Fitness
In this step, each island is evaluated for the two objective functions presented in Section 4.5.

Step 4: Calculate Constraints Violated
This step is used to calculate the number of contraints that are violated by each island. These number of constraints will be important to calculate the ranking of each island. Given a vector $P$ of pressures, for each position of vector $p$, if the values does not obey the contraint of minimum pressure presented in Equation 4.1 the number of constraints violated will increase by 1.
Step 5: **Calculate Ranking**
In this step will be calculated the ranking of each solution according to Algorithm 4.4 that will be presented in Section 4.5.1.

Step 6: **Calculate HSI**
The calculation of the High Suitable Index (HSI) takes into account the ranking calculated on previous step, and this calculation is handled by Equation 4.4.

Step 7: **Apply Elitism**
This step in handled by Algorithm 4.1 which will be presented in Section 4.3. The algorithm receives a set of islands and return the mark the islands that are part of the nondominated set.

Step 8: **Migration & Mutation**
In this step, the islands within the same archipelago, share information between them. Islands with higher HSI tend to accept values more often from islands with lower HSI. At each share of information, the shared value can be mutated with a certain probability. Both algorithms are described in Section 4.5.1.

Step 9: **Cross Migration & Mutation**
This step is similar to the previous step, but instead of sharing information within each Archipelago, the islands share information with islands in other Archipelago.

These steps are repeated at each iteration, until a SIGTERM signal is sent to the algorithm. The SIGTERM signal is when a given timeout is reached. This signal activates an exit function that applies Algorithm 4.1, in order to calculate the last nondominated set of solutions. Then, this set of solutions is provided in the output.

### 4.2 Hydraulic Simulator

The hydraulic simulator used in this dissertation was the Epanet 2.0 (Rossman, 2000). Epanet is an integrated environment used to edit and create networks, and can also be used to simulate the hydraulic behavior of a network for an extended period of analysis. As a result, Epanet provides the values of the pressure, the flow at each node and pipe at each hour of the day.

Epanet provides a library with endpoints that allow developers to integrate the hydraulic simulator with external algorithms in order to optimize the network regarding efficiency.

### 4.3 Nondominated Set

The nondominated set is the set of solutions that are nondominated. The definition of dominance and nondominated set are presented in section 2.3.1. The set is calculated based on the algorithm introduced by Kung, Luccio, and Preparata (1975), and its pseudocode is presented in Algorithm 4.1.

Algorithm 4.1 is a divide-and-conquer algorithm which means it works by breaking down the problem into small sub-problems, recursively. The algorithm receives as input a vector of islands that is sorted from the best to the worst fitness related to the first objective function. Then, the vector is divided into two sub vectors, the topSet and the bottomSet being the topSet always better than the bottomSet in first fitness value(line 4, line 5). Then, each element of the topSet is compared to the first element of the bottomSet. If the first fitness value of the solution in the topSet is equal to the solution in the bottomSet (line 13), then we compare the second fitness (line 14), if the bottomSet solution has a better
Algorithm 4.1: Nondominated Set

Data: islands
Result: nondominated island set
1 size ← size(islands);
2 if size = 1 then
3 return islands;
4 topSet ← nonDominatedSet(islands[0 : (size/2) − 1]);
5 bottomSet ← nonDominatedSet(islands[(size/2) : size − 1]);
6 while size(bottomSet) > 0 do
7 isDominated ← false;
8 replaced ← false;
9 for i ∈ range(0, size(topSet)) do
10 if topSet[i].fitness(1) ≤ bottomSet[0].fitness(1) then
11 isDominated ← true;
12 break;
13 else if topSet[i].fitness(0) == bottomSet[0].fitness(0) then
14 if topSet[i].fitness(1) > bottomSet[0].fitness(1) then
15 topSet[i] ← bottomSet[0];
16 replaced ← true;
17 break;
18 if replaced == false then
19 if isDominated == false then
20 topSet.append(bottomSet[0]);
21 bottomSet[0].delete();
22 return topSet;

second fitness than the solution in topSet, then the solution in topSet is replaced with the solution in the bottomSet (line 15).

4.4 Metrics Module

This module is responsible for calculating the Hypervolume and the Inverted Generational Distance of the final population of the algorithm. The calculation of the Hypervolume was handled by a library called shark-ml which is used for machine learning and optimization problems (Igel, Heidrich-Meisner, & Glasmachers, 2008). The calculation of IGD was handled by a metaheuristics framework called jmetalcpp (López-Camacho, García Godoy, Nebro, & Aldana-Montes, 2014).

In order to calculate the Hypervolume, we must have a nondominated set of points and a reference point. In this case, as there are two objective functions with values between 0 and 1, with 0 being the optimum value, the reference point will be the point (1, 1) in a bi-objective space with both objectives being minimized. A graphical example of a given nondominated set of points and a reference point is illustrated in Figure 4.2. The Hypervolume value corresponds to the pink area above the nondominated set.

The Inverted Generational Distance is a metric for assessing the quality of approximations to the Pareto front obtained by a multiobjective optimization algorithm (Bezerra, López-Ibáñez, & Stützle, 2017).

In order to calculate the Inverted Generational Distance, it is necessary to have the Pareto front of a multiobjective optimization problem and the nondominated set of solutions generated by the algorithm. In most cases it is not possible to get the complete Pareto front of a multi-objective optimization problem.
because of its computational complexity. Hence, an approximation must be used. For this, we group all nondominated solutions of each independent run of the algorithm and calculate a new nondominated set of these solutions. This new nondominated set is the best approximation to the Pareto front that was found, and it is used as the reference to calculate the Inverted Generational Distance against each of the nondominated set solutions of each independent run of the algorithm.

A graphical example of the inverted generational distance is represented in Figure 4.3. The Inverted Generational Distance corresponds to the sum of each Euclidean distance between each point in the Pareto front and its closest solution in the nondominated set. The final value, which is the sum of every Euclidean distance is normalized by being divided by the number of solutions in the Pareto front.

4.5 Optimization Module

A brief explanation of the Biogeography-based Optimization (BBO) algorithm was already made in Section 2.2. In this section, the optimization criteria of the algorithm is described.

The optimization procedure is done regarding two objective functions. The first objective function is given by:

$$\text{Minimize} : \sum_{i}^{NN} \frac{|\text{MIN\_PRESSURE} - p_i|}{\text{MIN\_PRESSURE} \ast NN}$$  \hspace{1cm} (4.1)

where $\text{MIN\_PRESSURE}$ is the minimal pressure that each node should have, $p_i$ is the value of the pressure at node $i$ and $NN$ is the number of nodes in the network.

The second objective function to optimize is given by:

$$\text{Minimize} : \sum_{i}^{PP} \text{diameter}(\text{pipe}_i) \ast \phi(\text{pipe}_i)$$  \hspace{1cm} (4.2)

where $PP$ is the number of pipes in the network, $\text{diameter}(\text{pipe}_i)$ is the value of the diameter of $\text{pipe}_i$, and $\phi(\text{pipe}_i)$ is an activation function that returns 1 if in that step of the algorithm is considered that $\text{pipe}_i$ has a valve installed and returns 0 otherwise. The final value given by the previous equation is then divided by a $\text{MAX\_COST}$ which is the cost associated with having valves installed in every pipe. This
allows the values to range between 0 and 1.

The BBO algorithm has four subsystems which are called *Archipelagos*. Each *Archipelago* optimizes according to one of the objective functions previously described. Also, each *Archipelago* must obey a set of constraints. In this case, only one constraint is considered, which is the minimum pressure necessary to deliver water to consumers. The constraint is given by Equation 4.3:

\[ p_i \geq \text{MIN}_\text{PRESSURE}, \quad i \in [0, NN] \]  

(4.3)

where \( p_i \) is the pressure of node \( i \) and \( NN \) is the number of nodes in the network.

The fitness of each island is calculated based on the objective function of the archipelago it belongs to.

The fitness value of each island will be used to calculate its ranking. A detailed description how the ranking mechanism works will be made in section 4.5.2.

The ranking of each solution is used to calculate its HSI, which is a probabilistic value. The HSI is responsible for whether or not an island will receive new SIV values from other islands. For example, if an island \( i \) has an HSI value of 0.7 it will receive SIV values from other islands with a probability of 0.7 and will share its SIV values with a probability of 1 – 0.7. Hence, islands with an higher HSI will receive SIVs from other islands more often than islands with lower HSI.

Next we describe the implementation of the ranking mechanism, which is calculated regarding the fitness value and the number of constraints violated of each solution. In section 4.5.1 will be described the implementation the migration and mutation mechanisms that are responsible for the optimization process. In Subsection 4.5.3 is described the algorithm to choose the best set of solutions to be kept for the next iteration; In Subsection 4.1 is presented a flux diagram with all the steps of the optimization procedure.
Algorithm 4.2: Mutation Operator

Data: island

for $i \in [0, \text{ISLAND\_SIZE}]$ do
  $r \leftarrow \text{random}(0,1)$;
  if $r < \text{MUTATION\_RATE}$ then
    if island.getSIV($i$) == initialRoughness[$i$] then
      newRoughness $\leftarrow \text{random}(0,1) \times \text{VALVE\_THRESHOLD} \times \text{initialRoughness}[i]$;
      island.setSIV($i$, newRoughness);
    else
      island.setSIV($i$, initialRoughness[$i$]);

4.5.1 Operators

For the optimization, there are three essential operators: Migration, Cross Migration, and Mutation. The Migration operator is responsible for sharing SIVs values, with a certain probability, between different islands within the same subsystem. This mechanism allows islands with higher HSI to receive values from other islands with a higher probability, thus increasing the fitness of all islands. The Cross Migration operator is similar to the Migration operator. The difference is that, while in the Migration operator the SIVs values are shared between islands within the same subsystem, in the Cross Migration operator the SIVs values are shared between islands of different subsystems. This allows all the archipelagos to evolve concerning all the objective functions. The Mutation operator is used to maintain the diversity of values in each solution. Hence, it keeps the diversity of solutions that helps to optimize all the candidate solutions.

The pseudocode for the Mutation and the Migration are given by Algorithm 4.2, and Algorithm 4.3, respectively.

Algorithm 4.2 receives as input the SIV values of an island, which correspond to the roughness coefficient of each pipe. These values could be mutated taking into account a probability. In line 2 a random number between 0 and 1 is generated to verify whether to mutate a SIV value of an island or not. If the SIV value is chosen to be mutated, there are two conditions:

1. the value corresponds to the initial roughness of a pipe (line 4), and so, the value will be mutated to a value which corresponds of having a valve installed in that pipe with a opening degree between 0 and VALVE\_THRESHOLD.

2. the value of roughness coefficient is already considered a valve with a certain opening degree and it will be mutated to a non-valve value which is the value of the initial roughness of the pipe (line 8).

VALVE\_THRESHOLD is a constant that represents the maximum opening degree that a valve must have to be installed. In this case the value corresponds to 0.5.

In Algorithm 4.3 all the islands will suffer immigration proportionally to its ranking. The ranking mechanism will be described in Section 4.5.2. For each island in the Archipelago (line 1) and then, for each solution in the island (line 3) a random number between 0 and 1 is generated in order to accept or not emigrating values from other islands. The emigrating value is chosen according to the emigration probability which is $1 - \text{HSI}$. Both the mutation operator and the migration operator are destructive methods, this is, the input they receive are pointers to the data structures and the values are replaced instantaneously, and so, there is no need for the operators to return a value.
Algorithm 4.3: Migration Operator

Data: Archipelago
1 for island ∈ Archipelago do
2 immigration = island.getHSI();
3 for i ∈ [0, ISLAND_SIZE] do
4 r = random(0, 1);
5 if r ≤ immigration then
6 emigratingIsland = chooseProbabilistically(Archipelago);
7 newSIV = emigratingIsland.getSIV[i];
8 island.setSIV(i, newSIV);

Algorithm 4.4: Ranking Algorithm

Data: islands
1 for i in range(0, size(islands)) do
2 for j in range(i, size(islands)) do
3 if island[i].getConstraintsViolated() > island[j].getConstraintsViolated() then
4 island[i].addRanking(1);
5 else if island[i].getConstraintsViolated() < island[j].getConstraintsViolated() then
6 island[j].addRanking(1);
7 else
8 F1 = island[i].getFitness();
9 F2 = island[j].getFitness();
10 if F1 < F2 then
11 island[j].addRanking(1);
12 else if F1 > F2 then
13 island[i].addRanking(1);

4.5.2 Ranking Solutions

In order to rank all the solutions, a ranking mechanism was implemented. This ranking mechanism is based on the one presented by Zheng et al. (2016). The pseudocode is presented in Algorithm 4.4.

To calculate the ranking of each solution, both the number of constraints that are violated and the fitness of each solution are taken into account. The solution with the lowest ranking will be the solution with the best fitness and with no constraints violated. For this, we compare each pair of islands of the same archipelago (line 1, line 2), verify which one has more constraints violated (line 3) and add 1 to its ranking. In case the number of constraints violated is equal, the measure of quality used to rank the solutions is the fitness. The island with the highest fitness will have its ranking increased by 1.

This ranking is used to calculate the immigration probability which corresponds to the HSI value of each island based on Equation 4.4:

\[
\lambda = \frac{\sum_{i=1}^{R} i}{\sum_{i=1}^{T} i}
\]  

(4.4)

where \( R \) is the ranking of the island whose immigration rate is being calculated and \( T \) is the total number of islands in a subsystem.
4.5.3 Elitism

In multiobjective optimization problems, the elitism is a way to preserve the best solutions found so far. This process of elitism helps the algorithm to evolve faster because, in evolutionary computation, the evolution process consists in the changing of information between all the candidate solutions. If good solutions are preserved for the next iterations, they will share its information with all other solutions. In each iteration of the algorithm, all the islands are submitted to a process to choose if they will be part of the elite or not. In WATimizer the process of election is based on the nondominated set of solutions. All the solutions in the nondominated set are considered elite solutions. For each archipelago, at the end of each iteration, there will be a set of islands that are part of the nondominated set, these islands are elected and they will be immutable on the next generation. The algorithm to find the nondominated set was presented in Algorithm 4.1.
Chapter 5

Experimental Results

The purpose of this Chapter is to present the results obtained by applying the WATimizer tool to the problem of reducing the leakage in water distribution systems. The amount of water leakage reduced is proportional to the reduction of the overall pressure in the network according to the following Equation:

\[ Q = K_f \times P^b \]  

(5.1)

where \( Q \) is the leakage amount, \( K_f \) is the leakage coefficient which is estimated as a function of the pipe characteristics, \( P \) is the average zonal pressure and \( b \) is the leakage exponent. These calculations are all handled by EPANET2. The leakage coefficients are known as emitter coefficients in the EPANET2 simulator and they can be seen as a hole where the leakage happens.

To evaluate WATimizer, thirteen different networks were generated with the WaterNetGen tool (Muranho, Ferreira, Sousa, Gomes, & Marques, 2012). These generated networks have not a pattern of leakages and so, it is not possible to present quantitative values of how much the leakage was reduced by reducing the pressures. Also a case study network was considered and in this case, the network have a well known leakaged pattern associated, and so it is possible to present quantitative values of how much the leakage was reduced by reducing the pressures. This case study network is presented in 5.5.

This multi-objective problem was also modelled to work with the Non-Dominated Sorting Genetic Algorithm (NSGAII) in order to compare the results obtained by WATimizer and NSGAII. The NSGAII algorithm used was developed at KanGAL, which is a research laboratory at the Indian Institute of Technology Kanpur. The problems to be solved are WDNs with different configurations.

This Chapter is divided into four Sections. In Section 5.1 is explained how the networks used to test the algorithm were generated. In this Section is also presented a well known network first used by Jowitt and Xu (1990) and later used in many other studies (Savic & Walters., 1995; Reis et al., 1997; Alve, Kalanithy, & Lumbers, 1998; Liberatore & Sechi, 2009; Dai & Li, 2014; J. Salazar & Salcedo, 2015; Araujo et al., 2006; Dai & Li, 2016; Paola et al., 2017). In Section 5.2 is explained how the size of the population and the mutation rate values for both algorithms, WATimizer and NSGAII, were setup to solve this particular problem. In Section 5.3 a qualitative evaluation of WATimizer against the algorithm NSGAII is done. The tool WATimizer is described in Chapter 4 and it is compared with NSGAII in terms of efficiency, taken into consideration the Hypervolume and the Inverted Generational Distance of the final solutions that were generated in a limited period of time. In Section 5.4 is presented a comparison between two methodologies to deal with multiobjective problems: the Pareto approach and the weighted sum approach. The goal is to analize which approach is more efficient. To finish, in Section 5.5, the WATimizer tool is used to solve a problem which is a well known network used in previous works.
Table 5.1: Network Configurations

<table>
<thead>
<tr>
<th>Network</th>
<th>Number Nodes</th>
<th>Number Edges</th>
<th>K-nearest Neighbours</th>
</tr>
</thead>
<tbody>
<tr>
<td>Net1</td>
<td>40</td>
<td>40</td>
<td>1</td>
</tr>
<tr>
<td>Net2</td>
<td>60</td>
<td>60</td>
<td>1</td>
</tr>
<tr>
<td>Net3</td>
<td>80</td>
<td>80</td>
<td>1</td>
</tr>
<tr>
<td>Net4</td>
<td>40</td>
<td>51</td>
<td>3</td>
</tr>
<tr>
<td>Net5</td>
<td>60</td>
<td>78</td>
<td>3</td>
</tr>
<tr>
<td>Net6</td>
<td>80</td>
<td>102</td>
<td>3</td>
</tr>
<tr>
<td>Net7</td>
<td>100</td>
<td>129</td>
<td>3</td>
</tr>
<tr>
<td>Net8</td>
<td>150</td>
<td>194</td>
<td>3</td>
</tr>
<tr>
<td>Net9</td>
<td>40</td>
<td>78</td>
<td>5</td>
</tr>
<tr>
<td>Net10</td>
<td>60</td>
<td>118</td>
<td>5</td>
</tr>
<tr>
<td>Net11</td>
<td>80</td>
<td>151</td>
<td>5</td>
</tr>
<tr>
<td>Net12</td>
<td>100</td>
<td>194</td>
<td>5</td>
</tr>
<tr>
<td>Net13</td>
<td>150</td>
<td>297</td>
<td>5</td>
</tr>
</tbody>
</table>

Figure 5.1: CaseStudy network with 25 nodes and 37 pipes.

results obtained by WATimizer (WATimizer) will is compared to the results obtained by in previous works.

5.1 Network Instances

To evaluate the performance of WATimizer, a set of networks were generated with the tool WaterNet-Gen (Muranho et al., 2012). This tool allows the creation of different networks with different configurations, varying the number of nodes, the number of edges and the K-Nearest Neighbours (KNN). Default values were used for all other parameters. Table 5.1 shows the configuration of each of the networks that were generated with WaterNetGen tool. These 13 networks are different in number of nodes, edges and KNN. The nodes are the components of the networks where the pressure must be minimized. The edges are the pipes where the valves could be installed. The K-Nearest Neighbours is the number of nodes that each node will connect to. Hence, a network with KNN \( \leq 2 \) is, approximately, a spanning-tree and a network with KNN \( = N \) is, approximately a clique.

In addition to the networks presented in Table 5.1, another network was used to evaluate the performance of the proposed algorithm. The configuration of the network is presented in Figure 5.1. The network was introduced as a research problem by Jowitt and Xu (1990) and it has been used in numerous research works addressing the problem of minimizing leakage in the installation of valves. The network has 3 reservoirs, 25 nodes and 37 pipes and a set of leakage coefficients that will be described in Section 5.5.
5.2 Choosing the Best Configuration

To choose the best configuration for each algorithm (WATimizer and NSGAII) in terms of mutation rate and population size, 6 different networks were used. Net1, Net2, Net4, Net5, Net9 and Net10 were chosen because they are the easiest problems to converge in a limited period of time of 300 seconds, which was the timeout given to each algorithm to evolve.

Each algorithm ran several combinations of different population sizes and mutation rates. For each combination of population sizes and mutation rates were used 6 different seeds in order to stabilize the results. For each run it was given a limited time of 300 seconds which was controlled by runsolver (Roussel, 2011). Runsolver is a tool developed by Roussel (2011) and it was designed to control the execution of solvers for the pseudo-boolean competition in 2005. In this case the runsolver was used to send a signal to the algorithm as soon as the 300 seconds of execution ended in order for the algorithm calculate the best solution found so far and exit. Both the algorithms ran in an AMD Opteron 6276 2.3 GHz Processor.

Some of the results are presented in Table 5.2 and Table 5.3. Each row of the table presents a combination of population sizes and mutation rate as well as its corresponding Hypervolume and Inverted Generational Distance (IGD) with the mean, median and standard deviation of each of them taken into account the seeds used.

<table>
<thead>
<tr>
<th>PopSize</th>
<th>MutRate</th>
<th>HV mean</th>
<th>HV median</th>
<th>HV std</th>
<th>IGD mean</th>
<th>IGD median</th>
<th>IGD std</th>
</tr>
</thead>
<tbody>
<tr>
<td>40</td>
<td>0.0001</td>
<td>0.6228</td>
<td>0.6464</td>
<td>0.0092</td>
<td>0.0944</td>
<td>0.091</td>
<td>0.0242</td>
</tr>
<tr>
<td>40</td>
<td>0.0005</td>
<td>0.624</td>
<td>0.641</td>
<td>0.0112</td>
<td>0.1027</td>
<td>0.1069</td>
<td>0.0191</td>
</tr>
<tr>
<td>40</td>
<td>0.001</td>
<td>0.628</td>
<td>0.6434</td>
<td>0.0181</td>
<td>0.099</td>
<td>0.107</td>
<td>0.0245</td>
</tr>
<tr>
<td>40</td>
<td>0.005</td>
<td>0.6326</td>
<td>0.6507</td>
<td>0.0081</td>
<td>0.0881</td>
<td>0.0821</td>
<td>0.0238</td>
</tr>
<tr>
<td>80</td>
<td>0.0001</td>
<td>0.5902</td>
<td>0.6098</td>
<td>0.0203</td>
<td>0.0954</td>
<td>0.0821</td>
<td>0.0167</td>
</tr>
<tr>
<td>80</td>
<td>0.0005</td>
<td>0.5928</td>
<td>0.6111</td>
<td>0.0198</td>
<td>0.0898</td>
<td>0.0897</td>
<td>0.0175</td>
</tr>
<tr>
<td>80</td>
<td>0.001</td>
<td>0.5875</td>
<td>0.6006</td>
<td>0.0225</td>
<td>0.1001</td>
<td>0.0956</td>
<td>0.0171</td>
</tr>
<tr>
<td>80</td>
<td>0.005</td>
<td>0.5741</td>
<td>0.6034</td>
<td>0.0377</td>
<td>0.0981</td>
<td>0.086</td>
<td>0.0182</td>
</tr>
<tr>
<td>160</td>
<td>0.0001</td>
<td>0.5699</td>
<td>0.6162</td>
<td>0.0262</td>
<td>0.103</td>
<td>0.0894</td>
<td>0.0137</td>
</tr>
<tr>
<td>160</td>
<td>0.0005</td>
<td>0.5667</td>
<td>0.6677</td>
<td>0.0085</td>
<td>0.0812</td>
<td>0.0773</td>
<td>0.0276</td>
</tr>
<tr>
<td>160</td>
<td>0.001</td>
<td>0.5665</td>
<td>0.6696</td>
<td>0.0126</td>
<td>0.0859</td>
<td>0.0847</td>
<td>0.0278</td>
</tr>
<tr>
<td>160</td>
<td>0.005</td>
<td>0.5719</td>
<td>0.6725</td>
<td>0.0123</td>
<td>0.0946</td>
<td>0.0949</td>
<td>0.0156</td>
</tr>
<tr>
<td>160</td>
<td>0.01</td>
<td>0.5606</td>
<td>0.6629</td>
<td>0.0161</td>
<td>0.0836</td>
<td>0.0698</td>
<td>0.0194</td>
</tr>
<tr>
<td>280</td>
<td>0.0001</td>
<td>0.5792</td>
<td>0.6026</td>
<td>0.0113</td>
<td>0.1161</td>
<td>0.0857</td>
<td>0.0162</td>
</tr>
<tr>
<td>280</td>
<td>0.0005</td>
<td>0.5753</td>
<td>0.5794</td>
<td>0.016</td>
<td>0.1152</td>
<td>0.0904</td>
<td>0.0177</td>
</tr>
<tr>
<td>280</td>
<td>0.001</td>
<td>0.5748</td>
<td>0.5715</td>
<td>0.0145</td>
<td>0.1204</td>
<td>0.0863</td>
<td>0.0124</td>
</tr>
<tr>
<td>280</td>
<td>0.005</td>
<td>0.5689</td>
<td>0.5874</td>
<td>0.0169</td>
<td>0.1158</td>
<td>0.0884</td>
<td>0.0117</td>
</tr>
<tr>
<td>280</td>
<td>0.01</td>
<td>0.5623</td>
<td>0.5963</td>
<td>0.0116</td>
<td>0.1777</td>
<td>0.0926</td>
<td>0.0114</td>
</tr>
<tr>
<td>400</td>
<td>0.0001</td>
<td>0.5542</td>
<td>0.5633</td>
<td>0.02</td>
<td>0.1299</td>
<td>0.0867</td>
<td>0.0122</td>
</tr>
<tr>
<td>400</td>
<td>0.0005</td>
<td>0.5555</td>
<td>0.5618</td>
<td>0.0174</td>
<td>0.1281</td>
<td>0.0968</td>
<td>0.0148</td>
</tr>
<tr>
<td>400</td>
<td>0.001</td>
<td>0.5532</td>
<td>0.5649</td>
<td>0.0127</td>
<td>0.127</td>
<td>0.0944</td>
<td>0.0111</td>
</tr>
<tr>
<td>400</td>
<td>0.005</td>
<td>0.5333</td>
<td>0.5618</td>
<td>0.0167</td>
<td>0.1279</td>
<td>0.0982</td>
<td>0.012</td>
</tr>
<tr>
<td>400</td>
<td>0.01</td>
<td>0.5203</td>
<td>0.5423</td>
<td>0.0167</td>
<td>0.1356</td>
<td>0.1039</td>
<td>0.009</td>
</tr>
</tbody>
</table>

In Table 5.2 the best result for WATimizer is undoubtedly the one with a population size of 160 and a mutation rate of 0.0005 because all the values are better than for other configurations. Hence, the values chosen for population size and mutation rate were 160 and 0.0005, respectively. For WATimizer, the size of the population presented in the table is the size of an isolate archipelago. Since the WATimizer algorithm was implemented with 4 different archipelagos, the total number of candidate solutions is 160×4 which is 640 candidate solutions.

In Table 5.3 the best results correspond to a population size of 400 individuals and a mutation rate of 0.01, because almost every metric calculated is better in this configuration. For the NSGAII there is
Table 5.3: NSGAII algorithm.

<table>
<thead>
<tr>
<th>PopSize</th>
<th>MutRate</th>
<th>HV mean</th>
<th>HV median</th>
<th>HV std</th>
<th>IGD mean</th>
<th>IGD median</th>
<th>IGD std</th>
</tr>
</thead>
<tbody>
<tr>
<td>160</td>
<td>0.0005</td>
<td>0.626</td>
<td>0.6733</td>
<td>0.0313</td>
<td>0.0807</td>
<td>0.0726</td>
<td>0.0248</td>
</tr>
<tr>
<td>160</td>
<td>0.001</td>
<td>0.6565</td>
<td>0.6775</td>
<td>0.0296</td>
<td>0.0816</td>
<td>0.0876</td>
<td>0.0264</td>
</tr>
<tr>
<td>160</td>
<td>0.005</td>
<td>0.6889</td>
<td>0.6995</td>
<td>0.0229</td>
<td>0.0912</td>
<td>0.0993</td>
<td>0.0182</td>
</tr>
<tr>
<td>160</td>
<td>0.01</td>
<td>0.6576</td>
<td>0.6911</td>
<td>0.0168</td>
<td>0.0712</td>
<td>0.0754</td>
<td>0.0179</td>
</tr>
<tr>
<td>160</td>
<td>0.1</td>
<td>0.5591</td>
<td>0.6748</td>
<td>0.0082</td>
<td>0.0434</td>
<td>0.0409</td>
<td>0.0119</td>
</tr>
<tr>
<td>160</td>
<td>0.15</td>
<td>0.625</td>
<td>0.639</td>
<td>0.0067</td>
<td>0.0683</td>
<td>0.0546</td>
<td>0.0079</td>
</tr>
<tr>
<td>280</td>
<td>0.0005</td>
<td>0.6469</td>
<td>0.6804</td>
<td>0.0318</td>
<td>0.0692</td>
<td>0.0645</td>
<td>0.0264</td>
</tr>
<tr>
<td>280</td>
<td>0.001</td>
<td>0.6376</td>
<td>0.6772</td>
<td>0.0257</td>
<td>0.0695</td>
<td>0.0647</td>
<td>0.0284</td>
</tr>
<tr>
<td>280</td>
<td>0.005</td>
<td>0.6509</td>
<td>0.6911</td>
<td>0.0168</td>
<td>0.0723</td>
<td>0.0815</td>
<td>0.0299</td>
</tr>
<tr>
<td>280</td>
<td>0.01</td>
<td>0.6574</td>
<td>0.6889</td>
<td>0.0152</td>
<td>0.0728</td>
<td>0.0824</td>
<td>0.0335</td>
</tr>
<tr>
<td>280</td>
<td>0.05</td>
<td>0.6699</td>
<td>0.6884</td>
<td>0.0167</td>
<td>0.0585</td>
<td>0.0591</td>
<td>0.0202</td>
</tr>
<tr>
<td>280</td>
<td>0.1</td>
<td>0.657</td>
<td>0.6713</td>
<td>0.0074</td>
<td>0.045</td>
<td>0.0284</td>
<td>0.0126</td>
</tr>
<tr>
<td>280</td>
<td>0.15</td>
<td>0.6228</td>
<td>0.6299</td>
<td>0.009</td>
<td>0.0672</td>
<td>0.0502</td>
<td>0.0092</td>
</tr>
<tr>
<td>400</td>
<td>0.0005</td>
<td>0.6548</td>
<td>0.6898</td>
<td>0.0206</td>
<td>0.0559</td>
<td>0.0482</td>
<td>0.0204</td>
</tr>
<tr>
<td>400</td>
<td>0.001</td>
<td>0.6507</td>
<td>0.6758</td>
<td>0.0181</td>
<td>0.0671</td>
<td>0.0642</td>
<td>0.0197</td>
</tr>
<tr>
<td>400</td>
<td>0.005</td>
<td>0.6556</td>
<td>0.6852</td>
<td>0.0203</td>
<td>0.0632</td>
<td>0.0651</td>
<td>0.0276</td>
</tr>
<tr>
<td>400</td>
<td>0.01</td>
<td>0.6649</td>
<td>0.6923</td>
<td>0.0138</td>
<td>0.0594</td>
<td>0.0571</td>
<td>0.0225</td>
</tr>
<tr>
<td>400</td>
<td>0.05</td>
<td>0.6637</td>
<td>0.6903</td>
<td>0.0188</td>
<td>0.0558</td>
<td>0.0539</td>
<td>0.018</td>
</tr>
<tr>
<td>400</td>
<td>0.1</td>
<td>0.6539</td>
<td>0.6666</td>
<td>0.0079</td>
<td>0.0485</td>
<td>0.0384</td>
<td>0.0094</td>
</tr>
<tr>
<td>400</td>
<td>0.15</td>
<td>0.6183</td>
<td>0.6327</td>
<td>0.0064</td>
<td>0.0717</td>
<td>0.053</td>
<td>0.0061</td>
</tr>
<tr>
<td>560</td>
<td>0.0005</td>
<td>0.6562</td>
<td>0.6801</td>
<td>0.0166</td>
<td>0.0662</td>
<td>0.0421</td>
<td>0.0218</td>
</tr>
<tr>
<td>560</td>
<td>0.001</td>
<td>0.6538</td>
<td>0.6853</td>
<td>0.0175</td>
<td>0.0556</td>
<td>0.0469</td>
<td>0.0231</td>
</tr>
<tr>
<td>560</td>
<td>0.005</td>
<td>0.6626</td>
<td>0.6901</td>
<td>0.0201</td>
<td>0.0555</td>
<td>0.0495</td>
<td>0.0179</td>
</tr>
<tr>
<td>560</td>
<td>0.01</td>
<td>0.6668</td>
<td>0.6792</td>
<td>0.0188</td>
<td>0.0593</td>
<td>0.0581</td>
<td>0.0194</td>
</tr>
<tr>
<td>560</td>
<td>0.05</td>
<td>0.6588</td>
<td>0.6807</td>
<td>0.014</td>
<td>0.0495</td>
<td>0.0466</td>
<td>0.0158</td>
</tr>
<tr>
<td>560</td>
<td>0.1</td>
<td>0.6478</td>
<td>0.6633</td>
<td>0.0063</td>
<td>0.0556</td>
<td>0.0419</td>
<td>0.0085</td>
</tr>
<tr>
<td>560</td>
<td>0.15</td>
<td>0.6102</td>
<td>0.627</td>
<td>0.0092</td>
<td>0.0812</td>
<td>0.0617</td>
<td>0.0077</td>
</tr>
</tbody>
</table>

an additional parameter that was set to 1. This parameter is the crossover mutation. This crossover mutation is used after a tournament where two individual are chosen to cross with each other. Since this tournament is already probabilistic and selects the best solutions with more probability, there is no need to have another probabilistic parameter to decide whether to cross or not.

The results presented in both Tables 5.2 and 5.3 are some combinations of population sizes and mutation rates that where calculated, but much more combinations were tried. The results for values of population sizes greater than the ones presented in each Table have the advantage of having a good initial fitness but they take much more time to evolve, specially for larger networks. For mutation rates with values lower than the values presented on each table do not allow the algorithms to evolve properly. After a few iterations they converge. For values of mutation rates greater that the values showed on each Table, the algorithms take a lot of time to evolve and at the end the results are worse than the ones presented on the tables.

5.3 WATimizer vs NSGAII

The WATimizer and the NSGAII algorithm were tested using every network presented in table 5.1. Each algorithm was run, with its best configuration for 1800 seconds, which corresponds to half of an hour. Moreover, for each algorithm and network, 6 different seeds were used.

The results collected from each algorithm were the nondominated set of solutions at each second of execution. At the end of every execution, the best approximation to the Pareto front was calculated taken into account all nondominated sets generated by every execution of both the WATimizer and the NSGAII. This calculation was handled by Algorithm 4.1. After the best Pareto front approximation has been calculated, the hypervolume and IGD metrics were calculated for each nondominated set outputted by each algorithm in each second. The goal was to observe the evolution of these two metrics over time.
Figure 5.2: Results for the Hypervolume and IGD for each of the algorithms. The upper plot shows the evolution of the hypervolume over the 30 minute run, taken into account the median values of 6 different runs, where the orange line is the evolution for NSGAII and the blue line is the evolution for WATimizer. The lower plot shows the evolution of the IGD over the 30 minutes run, taken into account the median values of 6 different runs, where the orange line is the evolution for NSGAII and the blue line is the evolution for WATimizer.

The results show that for networks with number of edges less than 80, the WATimizer algorithm converges after few iterations, and stop evolving until the end of the 1800 seconds, but the NSGAII algorithm increases slightly until the end of the 1800 seconds. This happens because the mutation rate of the WATimizer algorithm is much lower than the mutation rate of the NSGAII and so, it is much more probable of being stuck in a local minima. Figure 5.2 shows the evolution of the Hypervolume and the IGD over the 1800 seconds for both the algorithms in Net8 which is a network big enough for the algorithms to evolve during some time before converging. Net 8 was chosen as the example network because is one of the largest networks used to test both the algorithms and is the largest network with the best diversity of solutions. The results for the other networks are presented in the Appendix chapter in Section A.3.

For all the 13 networks, the NSGAII has a slightly better performance than WATimizer, for both the Hypervolume and the IGD, except in network 10 where WATimizer have a better IGD than the NSGAII. This difference is around 5% to 10%. The results for each of the networks are showed in Figure 5.3.
Section 5.3 presents a Pareto approach for the multiobjective problem. In this Section the same problem will be solved with a weighted sum approach.

The weighted sum approach differs from the Pareto approach in the sense that, instead of optimizing each objective function individually, we create a global optimization function that is a scalarization of all objective functions. So, all the solutions that belong to the Pareto front can be generated by a weighted sum approach in which, the value generated by the scalarization function is the sum of each individual fitness multiplied by a number, which is the weight associated to that fitness. A brief explanation of the two mechanisms are presented in Section 2.3 where the concepts of dominance and scalarization function are introduced.

To test the efficiency of a weighted sum approach, we considered a set of weights to each of the objective functions. The goal is that the sum of the weights attributed to each of the objective functions be equal to 1. The weights used were 0.1, 0.25, 0.50, 0.75 and 0.9. For the two objective functions $f_1$ and $f_2$, the scalarization function $F$ is given by:

$$f_1 \ast \text{weight}_i + f_2 \ast (1 - \text{weight}_i), \quad \text{weight}_i \in \{0.1, 0.25, 0.5, 0.75, 0.9\}$$

where $f_1$ is the function for pressure minimization and $f_2$ is the function for cost minimization.

For each solution generated with the weighted sum approach, the values for each individual objective function were kept in order to be possible to plot the results and compare them to the results of the Pareto approach.

The results show an improvement in performance for the tool WATimizer, while using the weighted sum approach.

Figure 5.3: Results for the Hypervolume and IGD for each of the algorithms for every 13 networks. The upper plot shows the Hypervolume of the final solution of each algorithm. The orange bars are the Hypervolume of NSGAII and the blue bars are the Hypervolume of WATimizer. The lower plot shows the IGD of the final solution of each algorithm. The orange bars are the IGD of NSGAII and the blue bars are the IGD of WATimizer.

### 5.4 Pareto efficiency vs Weighted sum efficiency

The weighted sum approach differs from the Pareto approach in the sense that, instead of optimizing each objective function individually, we create a global optimization function that is a scalarization of all objective functions. So, all the solutions that belong to the Pareto front can be generated by a weighted sum approach in which, the value generated by the scalarization function is the sum of each individual fitness multiplied by a number, which is the weight associated to that fitness. A brief explanation of the two mechanisms are presented in Section 2.3 where the concepts of dominance and scalarization function are introduced.

To test the efficiency of a weighted sum approach, we considered a set of weights to each of the objective functions. The goal is that the sum of the weights attributed to each of the objective functions be equal to 1. The weights used were 0.1, 0.25, 0.50, 0.75 and 0.9. For the two objective functions $f_1$ and $f_2$, the scalarization function $F$ is given by:

$$f_1 \ast \text{weight}_i + f_2 \ast (1 - \text{weight}_i), \quad \text{weight}_i \in \{0.1, 0.25, 0.5, 0.75, 0.9\}$$

where $f_1$ is the function for pressure minimization and $f_2$ is the function for cost minimization.

For each solution generated with the weighted sum approach, the values for each individual objective function were kept in order to be possible to plot the results and compare them to the results of the Pareto approach.

The results show an improvement in performance for the tool WATimizer, while using the weighted sum approach.
sum approach, for every network. This improvement includes better results in both the objective functions and in time of execution. For the largest network tested, WATimizer output the best result for one run in approximately 20 minutes.

For the NSGAII it is the opposite. The weighted sum approach decreases slightly the performance of the algorithm. The results for the 13 different networks are presented in Figure 5.4.

In Figure 5.4 can be seen the comparison between the tool WATimizer using the weighted sum approach and the NSGAII using the Pareto approach, which are the best approaches for each algorithm respectively. It can be seen that WATimizer improved its performance in terms of Hypervolume for every network. For networks 3, 7, and 8 WATimizer got better results than NSGAII. In terms of IGD it got a little worse results than using the Pareto approach. This happens because with the weighted sum approach, for each run we only consider one solution which is the one with the best value given by the scalarization function. So, for each seed used, there will be atmost 5 possible solutions, which corresponds to the solutions generated by each combination of weights. When the nondominated set of solutions is calculated it will have atmost 5 possible solutions which is much less than the number of solutions in a nondominated set generated by a Pareto approach. Given the way how IGD is calculated, the IGD value get worse if few solutions are presented in the nondominated set.

The best improvement can be seen in network 8 with and increase of about 20% on the Hypervolume. In Figure 5.5 is presented two scatter plots with the solutions generated with the Pareto approach and the weighted sum approach for each algorithm. The results for the other networks are presented in Appendix chapter in Section A.4.

Figure 5.5 presents two different plots. The left plot shows the solutions obtained with the Pareto approach and with the weighted sum approach for WATimizer, and the right plot shows the same infor-
Figure 5.5: Comparison of the solutions generated from the Pareto approach and the weighted sum approach. The left plot shows the solutions for WATimizer, where the blue dots are the solutions generated by the weighted sum approach and the orange dots are the solutions generated by the Pareto approach. The right plot shows the solutions for NSGAII, where the blue dots are the solutions generated by the weighted sum approach and the orange dots are the solutions generated by the Pareto approach.

5.5 Case Study

This section presents the results of applying WATimizer to a well known network, shown in Figure 5.1. The network was first used by Jowitt and Xu (1990) and after that, was used in various different works (Savic & Walters., 1995; Reis et al., 1997; Alve et al., 1998; Liberatore & Sechi, 2009; Dai & Li, 2014; J. Saldarriaga & Salcedo, 2015; Araujo et al., 2006; Dai & Li, 2016; Paola et al., 2017).

This network is composed by 3 reservoirs, 37 pipes and 25 junctions. For this network there is a leakage coefficient, denoted as $K_f$, associated with each node in order to be able to calculate the amount of leakage that can be reduced. The calculation of the amount of leakage is handled by Equation 5.1 introduced early in this chapter.

The value chosen for the leakage exponent $b$ was 1.18 because it is the value chosen being in various previous researches. The $K_f$ for each node is given by a well known pattern of leakage coefficients that were used in this network by other researchers in order to consider water losses during the simulation. This leakage coefficients are known as emitter coefficients in the EPANET2 simulator and they can be seen as a hole where the leakage happens. A table with the emitter coefficients for each node of the network is presented in Table 5.4.

The network with this distribution of emitter coefficients has a total of water losses of $27.3 L/s$. Considering the price of water of Empresa Portuguesa de Águas Livres (EPAL) of 0.8 €/m³, the amount of losses corresponds to 689, 203.26€ of lost water per year.

To preserve the constraints of minimum pressure, a median value of high demand period was considered during the simulation. The high demand period is a period when the global pressure drop. This is due to the fact that everyone using a Water Distribution Network (WDN) at the same time cause a great amount of water leaving the network because there will be more opened holes where the water
Table 5.4: Emitter coefficient of each node of the network

<table>
<thead>
<tr>
<th>Node ID</th>
<th>Emitter($K_f$) $L_s^{-1} m^{-1/2}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.012055</td>
</tr>
<tr>
<td>2</td>
<td>0.033956</td>
</tr>
<tr>
<td>3</td>
<td>0.032088</td>
</tr>
<tr>
<td>4</td>
<td>0.005562</td>
</tr>
<tr>
<td>5</td>
<td>0.018383</td>
</tr>
<tr>
<td>6</td>
<td>0.019238</td>
</tr>
<tr>
<td>7</td>
<td>0.005300</td>
</tr>
<tr>
<td>8</td>
<td>0.018853</td>
</tr>
<tr>
<td>9</td>
<td>0.003532</td>
</tr>
<tr>
<td>10</td>
<td>0.016842</td>
</tr>
<tr>
<td>11</td>
<td>0.006270</td>
</tr>
<tr>
<td>12</td>
<td>0.024410</td>
</tr>
<tr>
<td>13</td>
<td>0.016842</td>
</tr>
<tr>
<td>14</td>
<td>0.019490</td>
</tr>
<tr>
<td>15</td>
<td>0.026884</td>
</tr>
<tr>
<td>16</td>
<td>0.019567</td>
</tr>
<tr>
<td>17</td>
<td>0.005286</td>
</tr>
<tr>
<td>18</td>
<td>0.009203</td>
</tr>
<tr>
<td>19</td>
<td>0.010819</td>
</tr>
<tr>
<td>20</td>
<td>0.020118</td>
</tr>
<tr>
<td>21</td>
<td>0.034997</td>
</tr>
</tbody>
</table>

can exit, thus, the overall pressure will decrease.

As previously mentioned, this network was used in many research works, and so there are several results which can be used to compare the results of applying WATimizer to this network. The previous results will be used to validate the results obtained in this dissertation and they are presented in Table 5.5. The column Pipe ID shows the id of each pipe where a valve was installed. The column Valve Open corresponds to a percentage of the opening degree of the corresponding valve and the Total Leakage column shows the values in liters per second of water lost after the optimization procedure. In the final column is represented the percentage of improvement related to the initial amount of water leakage which was 27.3 L/s.

Table 5.5: Results from different authors for Jowitt and Xu (1990) network.

<table>
<thead>
<tr>
<th>Authors</th>
<th>Pipe ID</th>
<th>Valve Open</th>
<th>Total Leakage flowrate (L/s)</th>
<th>Improvement</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reis et al. (1997)</td>
<td>1-11-29</td>
<td>6%-13%-0%</td>
<td>22.78L/s</td>
<td>16.6%</td>
</tr>
<tr>
<td>Vairavamoorthy and Lumbers (1998)</td>
<td>11-21-29</td>
<td>30%-90%-0%</td>
<td>20.78L/s</td>
<td>23.9%</td>
</tr>
<tr>
<td>Araujo et al. (2006)</td>
<td>11-21-29</td>
<td>-</td>
<td>22.1L/s</td>
<td>19%</td>
</tr>
<tr>
<td>Nicolini and Zovatto (2009)</td>
<td>1-11-20-21-27</td>
<td>6%-30%-0%-63%-37%</td>
<td>22.89L/s</td>
<td>16.2%</td>
</tr>
<tr>
<td>Creaco and Pezzinga (2014)</td>
<td>1-11-20</td>
<td>-</td>
<td>23.09L/s</td>
<td>15.4%</td>
</tr>
<tr>
<td>Ali (2015)</td>
<td>1-11-20</td>
<td>6%-29%-0%</td>
<td>21.72L/s</td>
<td>20.4%</td>
</tr>
</tbody>
</table>

The results for this network were obtained using WATimizer with both a Pareto approach and a weighted sum approach. The WATimizer tool ran in an AMD Opteron 6276 2.3 GHz Processor for 30 minutes with 6 different seeds and 5 different combination of weights which were, 0.9 – 0.1, 0.75 – 0.25, 0.5 – 0.5, 0.25 – 0.75 and 0.1 – 0.9.

The number of solutions obtained with the weighted sum approach was 30 because there is a single solution for each combination of seed and weights.

The results obtained were similar to the results obtained for the other networks, which corresponds to a better performance while using a weighted sum approach. Figure 5.6 shows the solution generated by the weighted sum and the Pareto approach.

After the WATimizer finish running, all the solutions found by the tool were filtered and the nondominated set was calculated. The values for all the solutions in the nondominated set are shown in
Figure 5.6: Comparison of the solutions generated from the Pareto approach and the weighted sum approach for the case study network introduced by Jowitt and Xu (1990). The blue dots correspond to the solutions generated with the weighted sum approach. The orange dots correspond to the solutions generated with a Pareto front approach.

Table 5.6: Best solutions Generated by WATimizer

<table>
<thead>
<tr>
<th>Number of Valves</th>
<th>Pipe ID</th>
<th>Valve Open</th>
<th>Total Leakage Flowrate (L/s)</th>
<th>Improvement</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>11</td>
<td>13%</td>
<td>22.0L/s</td>
<td>19.4%</td>
</tr>
<tr>
<td>3</td>
<td>11-20-21</td>
<td>16%-20%-3%</td>
<td>21.35L/s</td>
<td>21.8%</td>
</tr>
<tr>
<td>4</td>
<td>5-11-20-21</td>
<td>23%-27%-18%-8%</td>
<td>21.08L/s</td>
<td>22.7%</td>
</tr>
<tr>
<td>4</td>
<td>11-20-21-27</td>
<td>16%-14%-10%-6%</td>
<td>20.94L/s</td>
<td>23.3%</td>
</tr>
<tr>
<td>6</td>
<td>1-5-11-20-21-29</td>
<td>5%-27%-24%-4%-39%-25%</td>
<td>20.06L/s</td>
<td>25.5%</td>
</tr>
<tr>
<td>7</td>
<td>1-5-11-20-21-27-29</td>
<td>5%-14%-27%-7%-33%-29%-33%</td>
<td>19.95L/s</td>
<td>25.9%</td>
</tr>
</tbody>
</table>

Table 5.6 shows 6 different solutions with different number of valves installed. In terms of minimizing leakages, all the solutions are capable of a minimization of at least 19.4%, which corresponds to a saving of 133,705, 43\$ per year. In terms of investment, the cost of installing Pressure Reducing Valve (PRV) increase with the diameter of the pipe where the valve is to be installed. For the solutions mentioned above, the installations may occur on pipes 1, 5, 11, 20, 21, 27 and 29. Table 5.7 shows the diameter of each pipe where a PRV can be installed as well as the cost of installation and maintenance. These costs were taken from Covelli, Cimorelli, et al. (2016).

The first column of Table 5.7 shows the id of each pipe where the valves are to be installed. The second column shows the values of the pipe diameters which is the characteristic responsible for the cost of valve installation. The third column shows the initial cost of each PRV installation which include the purchase and the human labor cost. The fourth column shows the total price of $N$ years of additional loan repayment after the installation (Covelli, Cimorelli, et al., 2016). This value is obtained by the following expression:

$$FC_{PRV, inst} = IC_{PRV, inst} \cdot (1 + r)^N$$  \hspace{1cm} (5.3)$$

where $FC_{PRV, inst}$ is the final cost of installation, $IC_{PRV, inst}$ is the initial cost of installation, and $(1 + r)^N$ is the annual growth rate of the additional money used for the repayment of the loan received for purchasing the valve. A loan is considered because we are thinking of prices in the worst case scenario. In this case considering a loan mean that the entity has not enough money to buy the valves to be installed. The fifth column shows the initial price of maintenance that corresponds to the maintenance of the first year. The last column shows the total price of $N$ years of maintenance after the installation (Covelli,
Cimorelli, et al., 2016). The expression used to calculate the total cost of maintenance is given by:

$$FC_{PRV,\text{maint}} = IC_{PRV,\text{maint}} \sum_{i=1}^{N} (1 + r)^{i-1}$$  \hspace{1cm} (5.4)

where $FC_{PRV,\text{maint}}$ is the final cost of maintenance, $IC_{PRV,\text{maint}}$ is the initial cost of maintenance, $(1 + r)^{i-1}$ is the annual growth rate of the maintenance costs.

To calculate both the final installation cost and the final maintenance cost it was used the values $N = 5$ and $r = 0.01$, where the $N$ is the number of years considered and $r$ is the interest rate.

For the cheapest solutions which is the first presented in Table 5.6, the total price is 23,479.77€ including installation and maintenance during the next 5 year after the installation. It was already demostrated for this solution that the amount of saving of reducing the the water leakages was of 133,722.89€ per year. For the most expensive solution, the final cost is 110,493.89€, which includes installation and maintenance for next 5 years. The amount of saving due to the reduction of water leakages was 185,395.67€ per year.

There is always a tradeoff between efficiency and cost. This tradeoff must be evaluated by an expertise that knows how much money can be invested and how serious the problem is.

<table>
<thead>
<tr>
<th>Pipe ID</th>
<th>Diameter (mm)</th>
<th>$IC_{PRV,\text{inst}}$</th>
<th>$FC_{PRV,\text{inst}}$</th>
<th>$IC_{PRV,\text{maint}}$</th>
<th>$FC_{PRV,\text{maint}}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>457</td>
<td>16,316.60</td>
<td>17,148.91</td>
<td>1,241.10</td>
<td>6,330.86</td>
</tr>
<tr>
<td>5</td>
<td>305</td>
<td>9,571.20</td>
<td>10,059.43</td>
<td>596.00</td>
<td>3,040.2</td>
</tr>
<tr>
<td>11</td>
<td>457</td>
<td>16,316.60</td>
<td>17,148.91</td>
<td>1,241.10</td>
<td>6,330.86</td>
</tr>
<tr>
<td>20</td>
<td>229</td>
<td>7,582.80</td>
<td>7,969.8</td>
<td>407.40</td>
<td>2,078.15</td>
</tr>
<tr>
<td>21</td>
<td>229</td>
<td>7,582.80</td>
<td>7,969.8</td>
<td>407.40</td>
<td>2,078.15</td>
</tr>
<tr>
<td>27</td>
<td>381</td>
<td>14,241.60</td>
<td>14,968.06</td>
<td>1,043.60</td>
<td>5,323.41</td>
</tr>
<tr>
<td>29</td>
<td>229</td>
<td>7,582.80</td>
<td>7,969.8</td>
<td>407.40</td>
<td>2,078.15</td>
</tr>
</tbody>
</table>
Chapter 6

Conclusion

Water leakage is a global problem that affects everyone around the world, so something needs to be done in order to stop or at least mitigate this problem. The Dissertation has two main goals:

1. Provide an overview of different methodologies implemented to control leakages;

2. Introduce the tool WATimizer, built to deal with multiobjective problems, in this case the pressure minimization, and costs minimization problems.

WATimizer is based on a Biogeography-based Optimization (BBO) algorithm with a ranking system and two types of elitism, an elitism based on the nondominated set of solutions that is used with the Pareto front approach and a simple elitism based on the ranking of each individual that is used with the weighted sum approach.

6.1 Achievements

The tool presented in this dissertation proposes to introduce the BBO algorithm in the area of research of water leakage problems. The main goal of this tool is to find the best combination of valves installed in pipes of a Water Distribution Network (WDN) in order to minimize both the leakages and the investment costs. Two different approaches were used to test the efficiency of the tool. The Pareto front based approach, in which at each iteration of the algorithm the nondominated sort of solutions is calculated based on both fitnesses and excluding the solutions that violate constraints. The weighted sum approach, in which at each iteration the algorithm keeps the $n$ solutions that have the lower ranking.

The results have shown that with a weighted sum approach this tool can achieve a trade off between both objective function better that some of the algorithms presented in previous works.

6.2 Limitations

Despite of the positive results shown in this dissertation there are a few limitations that should be considered.

The value of roughness coefficient of each pipe is used as a pseudo valve installed in that pipe. When the value of the original roughness coefficient of a pipe is unchanged, we consider no valve in that pipe. When the value of the roughness coefficient is less than the original value we consider a valve in that pipe with an opening degree proportional to the new roughness coefficient, since the domain of each roughness coefficient is between 0 and its original value. In this dissertation it was considered a
threshold in which only pipes with a roughness coefficient less than 50% of its original roughness are considered to have a valve installed. This limitation only allows opening degrees of valves between 0 and 50 percent.

Other limitation of this methodology arises when the size of the WDN grows. With larger WDNs, the search space for the optimization algorithm grows due to the increase in the number of pipes in the network. For a WDN with one hundred pipes, the search space will be $100^n$ where $n$ is the number of different possible values for roughness coefficient. To surpass this limitation, a deeper research about techniques to reduce the search space must be made.

6.3 Future Work

As future work we propose to test our tool with more networks with larger sizes.

The networks have many variables and only three were tested, the number of pipes, the number of nodes and the cluster coefficient. Variables like the elevation of the nodes, number of reservatories and already installed valves should be considered in future work.

Furthermore, the tool should be improved with a module to detect a portion of pipes which are the best pipes to install the valves. This module will reduce the search space and is more likely to produce better results. The creation of this module is possible in two different ways:

- Detect the pipes that have end nodes with a high difference on its elevations, and if the water flows from upper node to the lower node, the lower node will have an increase in pressure and so, that pipe is a probably a good pipe to install a valve;
- Give preference to install valves in pipes with a larger internal diameter. Pipes with a large internal diameter have a decrease in the flow velocity and consequently an increase on pressure;


Appendix A

Additional Data

This Chapter is divided into four Sections. In the first section is presented a table with the methodologies used to detect leakages in water distribution systems. In the second section is presented a table with the methodologies used to reduce leakages in water distribution systems. In Section A.3 are presented the evolution in terms of Hypervolume and Inverted Generational Distance (IGD) over 30 minutes for each of the 13 different networks. In Section A.4 are present all the solutions generated with the Pareto approach and the weighted sum approach for both algorithms, WATimizer (WATimizer) and Non-Dominated Sorting Genetic Algorithm (NSGAII).
## A.1 Leakage detection literature

<table>
<thead>
<tr>
<th>Problem</th>
<th>Procedures</th>
<th>Variables</th>
<th>Simulator</th>
</tr>
</thead>
<tbody>
<tr>
<td>Leak Detection</td>
<td>Network Calibration</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sensor Placement</td>
<td>Levenberg-Marquardt</td>
<td>Simulated Annealing</td>
<td></td>
</tr>
<tr>
<td>Genetic Algorithm</td>
<td>Entropy based</td>
<td>Neural Networks</td>
<td></td>
</tr>
<tr>
<td>Evolving Networks</td>
<td>Spectral Clustering</td>
<td>Bayesian Networks</td>
<td></td>
</tr>
<tr>
<td>SVM</td>
<td>Inverse Transient Method</td>
<td>Monte Carlo Simulation</td>
<td></td>
</tr>
<tr>
<td>Depth-First Search</td>
<td>Pressure</td>
<td>Flow</td>
<td></td>
</tr>
<tr>
<td>Emitter Coefficient</td>
<td>EPANET</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Table A.1: Leakage detection bibliography

<table>
<thead>
<tr>
<th>Reference</th>
<th>Techniques</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ribeiro et al., 2015</td>
<td>✓ ✓ ✓ ✓ ✓</td>
</tr>
<tr>
<td>Candelieri, Conti, &amp; Archetti, 2014</td>
<td>✓ ✓ ✓ ✓ ✓</td>
</tr>
<tr>
<td>Vítkovský et al., 2000</td>
<td>✓ ✓ ✓ ✓ ✓</td>
</tr>
<tr>
<td>Candelieri, Soldi, et al., 2014</td>
<td>✓ ✓ ✓ ✓ ✓</td>
</tr>
<tr>
<td>Sousa et al., 2015</td>
<td>✓ ✓ ✓ ✓ ✓</td>
</tr>
<tr>
<td>Huang et al., 2016</td>
<td>✓ ✓ ✓ ✓ ✓</td>
</tr>
<tr>
<td>Nasirian et al., 2013</td>
<td>✓ ✓ ✓ ✓ ✓</td>
</tr>
<tr>
<td>Qi et al., 2014</td>
<td>✓ ✓ ✓ ✓ ✓</td>
</tr>
<tr>
<td>Soares et al., 2011</td>
<td>✓ ✓ ✓ ✓ ✓</td>
</tr>
<tr>
<td>Goulet et al., 2013</td>
<td>✓ ✓ ✓ ✓ ✓</td>
</tr>
<tr>
<td>Behzadian et al., 2009</td>
<td>✓ ✓ ✓ ✓ ✓</td>
</tr>
<tr>
<td>Covas &amp; Ramos, 2010</td>
<td>✓ ✓ ✓ ✓ ✓</td>
</tr>
<tr>
<td>Lijuan et al., 2012</td>
<td>✓ ✓ ✓ ✓ ✓</td>
</tr>
<tr>
<td>Mambretti &amp; Orsi, 2012</td>
<td>✓ ✓ ✓ ✓ ✓</td>
</tr>
<tr>
<td>Poulakis et al., 2003</td>
<td>✓ ✓ ✓ ✓ ✓</td>
</tr>
<tr>
<td>Wu et al., 2002</td>
<td>✓ ✓ ✓ ✓ ✓</td>
</tr>
<tr>
<td>Casillas et al., 2013</td>
<td>✓ ✓ ✓ ✓ ✓</td>
</tr>
<tr>
<td>Christodoulou et al., 2013</td>
<td>✓ ✓ ✓ ✓ ✓</td>
</tr>
<tr>
<td>Rosich et al., 2012</td>
<td>✓ ✓ ✓ ✓ ✓</td>
</tr>
<tr>
<td>Steffelbauer et al., 2014</td>
<td>✓ ✓ ✓ ✓ ✓</td>
</tr>
<tr>
<td>Colombo et al., 2009</td>
<td>✓ ✓ ✓ ✓ ✓</td>
</tr>
</tbody>
</table>

Review of transient-based leakage
## A.2 Leakage reduction literature

<table>
<thead>
<tr>
<th>Problem</th>
<th>Variables</th>
<th>Optimization procedures</th>
<th>Hydraulic Simulator</th>
<th>System linearization</th>
</tr>
</thead>
<tbody>
<tr>
<td>Water Losses reduction</td>
<td>Pressure Reducing Valves</td>
<td>Control Valves</td>
<td>Pipe diameter configuration</td>
<td>Genetic Algorithm</td>
</tr>
<tr>
<td>(Araujo et al., 2006)</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>(Covelli, Cimorelli, et al., 2016)</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>(Covelli, Cozzolino, et al., 2016)</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>(Creaco &amp; Pezzinga, 2015)</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>(Creaco &amp; Pezzinga, 2014)</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>(J. Saldarriaga &amp; Salcedo, 2015)</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>(Liberatore &amp; Sechi, 2009)</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>(De Paola et al., 2016)</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>(Paola et al., 2017)</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>(De Paola et al., 2017)</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>(Reis et al., 1997)</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>(Germanopoulos &amp; Jowitt, 1989)</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>(Jowitt &amp; Xu, 1990)</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>(Vairavamoorthy &amp; Lumbers, 1998)</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>(Geem, 2006)</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>
A.3 Pareto approach Results

The results presented below are the evolution of Hypervolume and IGD over time for both the WATimizer and NSGAII, using a Pareto approach.

Figure A.1: Evolution for network 1 and 2 over 30 minutes. network 1 on the left, network 2 on the right

Figure A.2: Evolution for network 3 and 4 over 30 minutes. network 3 on the left, network 4 on the right
Figure A.3: Evolution for network 5 and 6 over 30 minutes. network 5 on the left, network 6 on the right

Figure A.4: Evolution for network 7 and 9 over 30 minutes. network 7 on the left, network 9 on the right

Figure A.5: Evolution for network 10 and 11 over 30 minutes. network 10 on the left, network 11 on the right
Figure A.6: Evolution for network 12 and 13 over 30 minutes. network 12 on the left, network 13 on the right
A.4 Comparison Weighted sum approach vs Pareto approach

The results presented below are the solutions generated by WATimizer and NSGAII, using both the weighted sum approach and the Pareto approach.

![Figure A.7](image1.png)

Figure A.7: Left plot shows the results generated by WATimizer. Right plot shows the solutions generated by NSGAII

![Figure A.8](image2.png)

Figure A.8: Left plot shows the results generated by WATimizer. Right plot shows the solutions generated by NSGAII
Figure A.9: Left plot shows the results generated by WATimizer. Right plot shows the solutions generated by NSGAII

Figure A.10: Left plot shows the results generated by WATimizer. Right plot shows the solutions generated by NSGAII

Figure A.11: Left plot shows the results generated by WATimizer. Right plot shows the solutions generated by NSGAII
Figure A.12: Left plot shows the results generated by WATimizer. Right plot shows the solutions generated by NSGAII

Figure A.13: Left plot shows the results generated by WATimizer. Right plot shows the solutions generated by NSGAII

Figure A.14: Left plot shows the results generated by WATimizer. Right plot shows the solutions generated by NSGAII
Figure A.15: Left plot shows the results generated by WATimizer. Right plot shows the solutions generated by NSGAII

Figure A.16: Left plot shows the results generated by WATimizer. Right plot shows the solutions generated by NSGAII

Figure A.17: Left plot shows the results generated by WATimizer. Right plot shows the solutions generated by NSGAII
Figure A.18: Left plot shows the results generated by WATimizer. Right plot shows the solutions generated by NSGAII.