Optimal Valve Placement in Water Distribution Networks with CLP(FD)*

Massimiliano Cattafi
Imperial College London, UK

Marco Gavanelli, Maddalena Nonato, Stefano Alvisi and **Marco Franchini**
University of Ferrara, Italy
name.surname@unife.it

Abstract

This paper presents a new application of logic programming to a real-life problem in hydraulic engineering. The work is developed as a collaboration of computer scientists and hydraulic engineers, and applies Constraint Logic Programming to solve a hard combinatorial problem. This application deals with one aspect of the design of a water distribution network, i.e., the valve isolation system design.

We take the formulation of the problem by Giustolisi and Savić [2008] and show how, thanks to constraint propagation, we can get better solutions than the best solution known in the literature for the Apulian distribution network.

1 Introduction

A water distribution network is often represented by hydraulic engineers as a labeled graph, in which pipes are represented as undirected edges. In the network, there is at least one special node that is the source of the water (node 0 in Figure 1). Each user has a demand (in litres per seconds) and is connected to some edge of the graph. Each edge is labelled with the total demand of the users linked to it. For example, in Figure 1, the edge connecting nodes 2 and 5 (let us name it $e_{2,5}$) has a demand of $15l/s$ (that may be due, e.g., to five clients each requesting $3l/s$ on average).

When designing a water distribution network, one of the steps is designing the isolation system: in case a pipe has to be repaired (e.g., because of a break), a part of the network has to be disconnected from the rest of the network, in order to allow workers to fix the broken pipe. The isolation system consists of a set of *isolation valves*, that are placed in the pipes of the network, usually near one of the two endpoints; this means that in each pipe at most two valves can be placed. If in some pipe there are two valves, this means that this single pipe can be isolated by closing both the valves. In the example of Figure 1, the edge $e_{2,3}$ connecting nodes 2 and 3 has two valves, so in case this pipe is damaged, valves $v_{2,3}$ and $v_{3,2}$ will be closed, isolating only $e_{2,3}$.

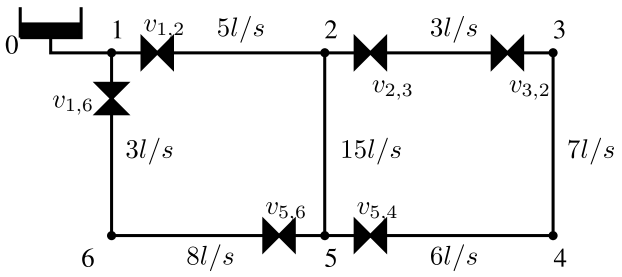

Figure 1: A schematic water distribution system with valves

However, placing two valves in each pipe is often not a viable option, because each valve has a cost; the cost is not only due to the manufacturing and physical placing of the valve, but also to the fact that the pipe is more fragile and deteriorates more quickly near valves. In case there are not two valves in each pipe (the usual case in real distribution networks), de-watering a pipe can imply the closure of more than two valves and the isolation of more than one pipe. In this case, more users other than those connected to the broken pipe will remain without service during pipe substitution. Suppose that the pipe $e_{3,4}$ connecting nodes 3 and 4 is damaged. In order to de-water it, workers have to close valves $v_{3,2}$ and $v_{5,4}$; as a result edge $e_{5,4}$ will be de-watered as well, and the clients that take water from it will have no service as well. Valves partition the network in the so-called *sectors*, that are those parts of the distribution network enclosed by a set of valves: edges $e_{3,4}$ and $e_{4,5}$ are in the same sector, so they cannot be de-watered independently one from the other.

The usual measure of the disruption in the service is the *undelivered demand*, i.e., the demand (in litres per second) that is not fulfilled during the repair operations; in the case there is need to de-water edge $e_{3,4}$, the disruption is the demand of the edges $e_{3,4}$ and $e_{4,5}$, i.e., $7 + 6 = 13l/s$. However, the undelivered demand does not always coincide with the sector the damaged pipe belongs to. For example, pipe $e_{2,5}$ belongs to the sector consisting of the edges $e_{1,2}$ and $e_{2,5}$, that is surrounded by valves $v_{1,2}$, $v_{2,3}$, $v_{5,4}$, and $v_{5,6}$; however by closing these four valves, edges $e_{2,3}$, $e_{3,4}$, and $e_{4,5}$ will be de-watered as well, even though they are not in the same sector of the broken pipe. This effect is called *unintended isolation* [Jun and Loganathan, 2007].

*The paper on which this extended abstract is based was the recipient of the best paper award of the 2011 International Conference on Logic Programming (ICLP) [Cattafi *et al.*, 2011].

The design of the isolation system consists of placing in the distribution network a given number of valves such that, in case of damage, the disruption is "minimal". Of course, the level of disruption depends on which pipe has to be fixed. In Figure 1 we have four sectors: if $e_{2,3}$ is damaged, the undelivered demand is $3l/s$, if one of $\{e_{3,4}, e_{4,5}\}$ is broken, the undelivered demand is $13l/s$, if the broken pipe is $e_{1,6}$ or $e_{5,6}$ the undelivered demand is $3 + 8 = 11l/s$, while for sector $\{e_{1,2}, e_{2,5}\}$ the undelivered demand is $36l/s$, corresponding to the demand of $\{e_{1,2}, e_{2,5}, e_{2,3}, e_{3,4}, e_{4,5}\}$. A usual measure [Giustolisi and Savić, 2010] is to take the worst case, and assign to the placement shown in Figure 1 (characterised by 6 valves) the effect of the maximal possible disruption: $36l/s$.

Giustolisi and Savić [2010] address the design of an isolation valve system as a two-objective problem: one objective is minimizing the number of valves in the isolation system, and the other is the minimization of the (maximum) undelivered demand. They adopt a genetic algorithm that is able to provide near-Pareto-optimal solutions, and apply it to the Apulian distribution network. The genetic algorithm provides good solutions in a very short time, but it is incomplete, so it does not provide, in general, Pareto-optimal solutions, but only solutions that are hopefully near to the Pareto front. The real optimal Pareto front remains unknown.

We believe that a complete search algorithm could provide better solutions, although at the cost of a higher computation time. Since the problem should be solved during the design of the valve system, there is no need to have a solution in real-time, and an algorithm providing a provably Pareto-optimal solution may be preferable with respect to incomplete algorithms, even with higher computation times.

In this paper, we address the same two-objective problem studied by Giustolisi and Savić [2010] as a sequence of single-objective ones; this is always possible when one of the objectives is integer [Van Wassenhove and Gelders, 1980; Gervet *et al.*, 1999; Gavanelli, 2002]. Given the number of valves, we model the design of the isolation valve system as a two-player game, and solve it with a minimax approach [Russell and Norvig, 2003]. As the game has an exponential number of moves, we reduce the search space by pruning redundant branches of the search tree, implementing the minimax algorithm in Constraint Logic Programming (CLP) [Jaffar and Maher, 1994] on Finite Domains (CLP(FD)) [Marriot and Stuckey, 1998; Frühwirth and Abdennadher, 2003; Dechter, 2003], in particular we used ECL^{i}PS^e [Schimpf and Shen, 2012]. Our algorithm is complete, so it is able to find the optimal solutions and prove optimality; we show improvements on the best solutions known in the literature, up to 10% of the objective function value.

2 Problem description

A water distribution network is a weighted undirected graph $G \equiv (N, E)$, where $N = \{1, \ldots, n\}$ is a set of nodes, $E = \{e_{ij}\}$ is a set of edges, and each edge e_{ij} has a weight $w(e_{ij})$ called *demand*. In the network, there is at least a *source* node.

Valves can be positioned near one of the ends of a pipe; we will refer to valve on edge e_{ij} near to node i as v_{ij}, while v_{ji} is a valve on the same edge, but close to node j.

Given a number N_v of valves, the objective is to position the valves in the network such that:

1. it is possible to isolate any pipe in the network. Formally, given an edge e_{ij}, it is possible to identify a set of valves $C(e_{ij})$ to be closed such that there is no path from any source node to e_{ij} that does not contain a valve $v \in C(e_{ij})$. Note that there is only one reasonable set $C(e_{ij})$ of valves to be closed given a broken edge e_{ij}: only the valves directly reachable from e_{ij} will be closed. For example, in Figure 1 if $e_{3,4}$ is broken, then $C(e_{3,4}) = \{v_{3,2}, v_{5,4}\}$ and it does not make sense to close farther valves, such as $v_{2,3}$, because in order to reach $v_{2,3}$ from $e_{3,4}$ we have to overpass other valves.
2. the objective is to minimize the maximum undelivered demand (UD). Let $D(C(e_{ij}))$ be the set of edges that do not receive water when the valves in $C(e_{ij})$ are closed; the function to be minimized is

$$UD = \max_{e_{ij} \in E} \sum_{e_{kl} \in D(C(e_{ij}))} w(e_{kl}).$$

3 Game model

The problem can be considered as a two-player game, consisting of the following three moves:

- the first player places N_v valves in the network;
- the second player selects one pipe to be damaged;
- the first closes a set of valves isolating the damaged pipe.

The cost for the first player (and reward for the second) is the undelivered demand (l/s): the total demand of all users that remain without service when the broken pipe is de-watered.

Given this formalization, the well-known minimax algorithm is applicable [Russell and Norvig, 2003].

As we said, choosing the last move is very easy, as there is only one reasonable solution: close all valves that are reachable from the broken pipe, without overpassing other valves.

Clearly the first step of the first player is the most sensitive, because it can generate a wide number of alternatives. However, some of the moves are not very interesting, for three reasons detailed in the next section. First, some solutions are clearly non-optimal. Second, some are symmetric, and provide valve placements that, although different, represent equivalent solutions. Third, after some solution is known, there is no point in looking for worse solutions: as soon as the current search branch cannot lead to solutions better than the incumbent, we can stop the search, backtrack, and continue from a more promising branch.

Each of these three cases provides a possible pruning of the search space, that can exponentially speed-up the computation with respect to a naive approach. The first two cases can be thought of as *constraints*, while the third can be though of as a *bound*: all of them can be simply cast in Constraint Logic Programming on Finite Domains (CLP(FD)).

4 Constraint Logic Programming model

We associate a Boolean variable to each possible position of a valve (so we have two Boolean variables for each edge in

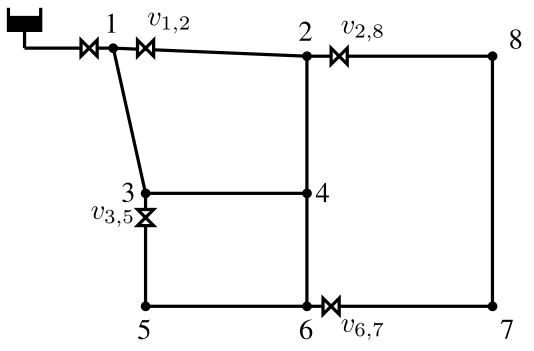

Figure 2: A network with redundant valves

the graph); if the variable takes value 1, then the given end of the edge hosts a valve, otherwise, if the variable takes value 0, there is no valve in such location. In the following, the list of these variables is called *Valves*.

The two-player game can be implemented as follows:

```
solve(Valves,$N_v$):-
  impose_constraints(Valves,$N_v$),
  **minimize**(
  ( assign_valves(Valves),
   **maximize**(
   ( break_pipe(Broken),
     close_valves(Valves,Broken,Closed),
     undelivered_demand(Valves,Closed,UD)
   ), UD, MaxUD)
  ),MaxUD,MinMaxUD).
```

$minimize/3$ and $maximize/3$ are predefined in CLP(FD) languages; declaratively, $minimize(G, F, V)$ selects, amongst the solutions of goal G (bindings to the variables that make true the goal G), the solution that provides the minimum value for variable F [Marriott and Stuckey, 1994; Fages, 1996]; such minimal value is bound to variable V.

Operationally, $minimize$ implements a form of branch-and-bound: it calls goal G and, if it succeeds providing some binding F/F^*, it imposes a new unbacktrackable constraint $F < F^*$; then it continues the search. The unbacktrackable constraint is considered in the constraint store of all the nodes of the search tree, and prunes every node that cannot possibly provide a lower value than F^*. When G fails, the last F^* obtained value is provided as the optimum [Prestwich, 1996].

Predicate $impose\_constraints$ posts all the constraints of the model to the constraint solver. It contains the constraint stating that there are N_v valves in the distribution network; other constraints will be described in Section 4.1.

$assign\_valves$ starts the search on the $Valves$ variables.

After finding a tentative valves positioning that satisfies all constraints, a maximisation phase tries the moves of the opponent player: it searches (predicate $break\_pipe$) the pipe that, if damaged, leads to a maximum disruption of the service. When we know the $Broken$ pipe, we can compute the valves that will be closed and the undelivered demand.

4.1 Reducing the number of moves

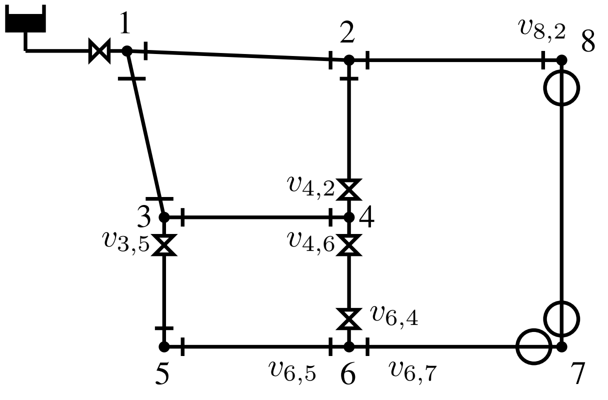

Figure 3: A partial assignment: circles mean absence of valve, strokes are variables not assigned yet

Redundant valves and symmetries Consider the network in Figure 2; just by looking at the topology, we can tell that some of the valves are redundant. Valve $v_{1,2}$ cannot be used to identify a sector; in fact, there is a closed path going from one side of the valve to opposite side: starting from node 1, to node 3, then 4, then 2.

In any closed path of the network there cannot be exactly one valve. No valves means that the whole path will be contained in a sector, which is sensible. Two valves or more can mean that the path is divided into two or more sectors. So, for each closed path, one could impose a constraint saying that the number of valves in such path cannot be equal to 1.

Indeed, the number of paths is exponential in the size of the network, however we can choose to impose such constraint only for a limited number of closed paths. We decided to impose such constraint only for (boundaries of) *faces*. When drawing the graph on a plane, each of the regions surrounded by edges of the graph is called a *face*. The number of faces of a planar graph is always polynomial, as proven by Euler.

When a node is connected to exactly two edges, we have a symmetry. Consider node 8 in Fig. 2: in one assignment, we could have a valve $v_{8,2}$, while another assignment could be identical but with a valve in $v_{8,7}$. These two solutions are symmetric, because the fact that node 8 is in the same sector as edge $e_{2,8}$ or as $e_{7,8}$ is irrelevant, since nodes do not have a contribution to the objective function. So, we can impose the symmetry breaking constraint $v_{8,7} = 0$. This simple observation can provide a notable speedup, because real networks often have this situation.

Bounding Consider a node in the search tree that selects the move for the first player (predicate $assign\_valves$): in a generic node, some of the v_{ij} variables will be assigned value 1 (meaning that some valves have already been placed), some variables will have value 0 (meaning that in such position there is no valve), and some will still be unassigned.

In Figure 3: circles represent positions in which there is no valve, while strokes are variables still unassigned. Even though we do not have a complete placement, we can already say that there is a sector containing at least edges $e_{7,8}$, and

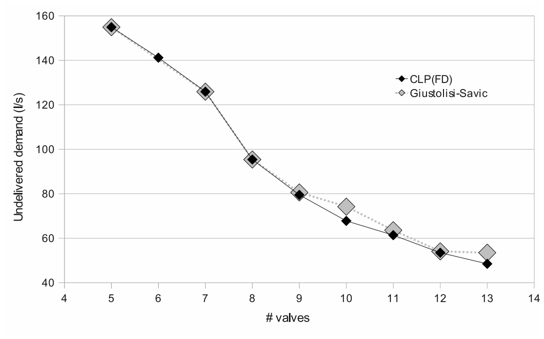

Figure 4: Comparison between the approximate Pareto front computed by Giustolisi-Savić and the optimal Pareto front obtained in CLP(FD)

$e_{6,7}$. The opponent player will have the option of damaging, e.g., pipe $e_{7,8}$, causing an undelivered demand that is no less than $w(e_{7,8}) + w(e_{6,7})$. So if the cost of such sector is worse than the current best solution found by the first player (i.e., $w(e_{7,8}) + w(e_{6,7}) > UD^{best}$), there is no point in continuing the search on the current branch.

We can also reason as in reduced costs pruning [Focacci *et al.*, 1999; 2002]. Suppose that $w(e_{7,8}) + w(e_{6,7}) < UD^{best}$ but adding $w(e_{2,8})$ makes the total higher than UD^{best}: this means that we cannot afford to include edge $e_{2,8}$ in the same sector, and the only possibility to get a solution better than UD^{best} is to separate the two sectors, placing a valve in $v_{8,2}$.

5 Experimental results

We compare our results with those reported by Giustolisi and Savić [2008], and we apply our CLP(FD) algorithm on the Apulian water distribution network reported in that paper. Both the software and the instance are available on the web [Cattafi and Gavanelli, 2011]. Giustolisi and Savić [2008] adopt a multi-objective genetic algorithm, that minimizes both the number of valves and the undelivered demand. The aim is to find the so-called Pareto frontier [Gavanelli, 2002]; a solution belongs to the frontier if there is no way to improve one objective without worsening the other. The genetic algorithm, however, is not able to prove that a solution is indeed Pareto-optimal, and provides an approximation of the Pareto frontier. Moreover, Giustolisi and Savić [2008] use a simplifying assumption: *"in order to reduce greatly the search space of the optimizer, the constraint of a maximum of one valve for each pipe was tested"*. They report the best solutions obtained with a number of valves from 5 to 13.

We computed the true Pareto-optimal frontier by varying the number of valves from 5 to 13, and computing for each value the best placement. The comparison of the near-Pareto-optimal frontier and the true Pareto-optimal frontier obtained with our CLP(FD) program is shown in Figure 4. Giustolisi and Savić [2008] do not provide a solution with 6 valves, possibly because their algorithm was not able to find a solution with undelivered demand lower than that obtained with 5 valves. We proved, instead, that such a solution exists and adding a valve reduces the damage. Excluding this case, when the number of valves is low (up to 8 valves) their algorithm found the real optimum, probably due to the fact that the search space is still not very wide. When the number of valves increases, their algorithm gets farther from the real optimum, with a gap of about 10% with 10 and 13 valves. Also, we found a solution with 12 valves that gives the same undelivered demand that Giustolisi and Savić [2008] compute with 13 valves: in this sense, we were able to save one valve (out of 13) maintaining the same undelivered demand.

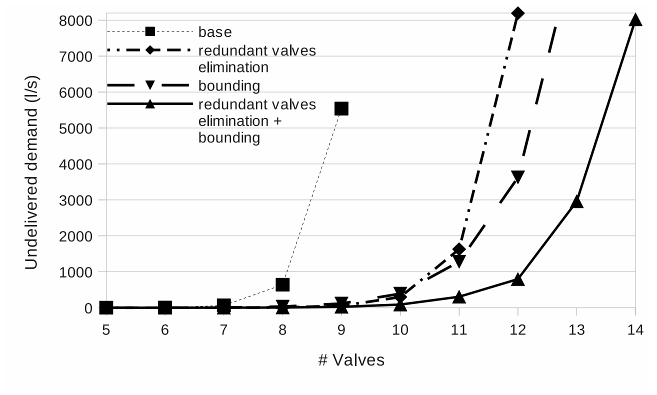

Figure 5: Computation time of the algorithms including different optimizations

In Figure 5, we show the computing time of the basic algorithm, and of the improved versions (Section 4.1) varying the number of valves. All experiments were done on an Intel Core 2 Duo T7250 2GHz computer (using only one core) with 4GB of RAM.

6 Related work

In the literature of hydraulic engineering, Giustolisi and Savić [2010] presented a multi-objective genetic algorithm for the near-optimal placement of isolation valves. Creaco *et al.* [2010] use different objective functions: the total cost of the set of valves, and the weighted average unsupplied demand associated with the segments. Both works use genetic algorithms, that cannot ensure that the found solution is the real optimum.

The valve placement problem has some similarities with the graph partitioning problem, in which the goal is to partition a graph into (almost) equal-size parts by removing the minimal number of edges or the total weight of removed edges. Most works in the literature deal with heuristics or approximation algorithms.

The valve-placement problem was also addressed in Integer Programming [Peano *et al.*, 2012] and in Answer Set Programming [Gavanelli *et al.*, 2013], but the best results are currently achieved by the CLP(FD) formulation.

7 Conclusions and future work

We presented a new application, taken from the hydraulic domain, for logic programming. We proposed an algorithm based on CLP(FD), and found solutions better than the best solutions known in the literature. The computation time can be high when the number of valves is high, but in many cases it is still acceptable since it is applied during the design of a water distribution network.

When the number of valves is very high, we could apply incomplete methods, that try to get quickly good solutions sacrificing the proof of optimality. One very promising approach would be to use Large Neighbourhood Search, that has been implemented in Prolog in previous works [Dal Palù *et al.*, 2010]; in future work we will explore this possibility.

References

[Cattafi and Gavanelli, 2011] Massimiliano Cattafi and Marco Gavanelli. A CLP(FD) program for the optimal placement of valves in a water distribution network. Source code available at `http://www.ing.unife.it/docenti/MarcoGavanelli/software/vp/`, April 2011.

[Cattafi *et al.*, 2011] Massimiliano Cattafi, Marco Gavanelli, Maddalena Nonato, Stefano Alvisi, and Marco Franchini. Optimal placement of valves in a water distribution network with CLP(FD). *Theory and Practice of Logic Programming*, 11(4-5):731–747, 2011.

[Creaco *et al.*, 2010] E. Creaco, M. Franchini, and S. Alvisi. Optimal placement of isolation valves in water distribution systems based on valve cost and weighted average demand shortfall. *Water Resources Management*, 24(15):4317–4338, 2010.

[Dal Palù *et al.*, 2010] Alessandro Dal Palù, Agostino Dovier, Federico Fogolari, and Enrico Pontelli. CLP-based protein fragment assembly. *Theory and Practice of Logic Programming*, 10(4-6):709–724, 2010.

[Dechter, 2003] Rina Dechter. *Constraint processing*. Morgan Kaufmann, 2003.

[Fages, 1996] F. Fages. From constraint minimization to goal optimization in CLP languages. In Eugene C. Freuder, editor, *Second International Conference on Principles and Practice of Constraint Programming CP'96*, volume 1118 of *Lecture Notes in Computer Science*, pages 537–538. Springer, 1996.

[Focacci *et al.*, 1999] Filippo Focacci, Andrea Lodi, and Michela Milano. Cost-based domain filtering. In Joxan Jaffar, editor, *Principles and Practice of Constraint Programming - CP'99*, volume 1713 of *Lecture Notes in Computer Science*, pages 189–203. Springer, 1999.

[Focacci *et al.*, 2002] Filippo Focacci, Andrea Lodi, and Michela Milano. A hybrid exact algorithm for the TSPTW. *INFORMS Journal on Computing*, 14(4):403–417, 2002.

[Frühwirth and Abdennadher, 2003] Thom Frühwirth and Slim Abdennadher. *Essentials of Constraint Programming*. Springer, 2003.

[Gavanelli *et al.*, 2013] Marco Gavanelli, Maddalena Nonato, and Andrea Peano. An ASP approach for the valves positioning optimization in a water distribution system. *Journal of Logic and Computation*, 2013.

[Gavanelli, 2002] Marco Gavanelli. An algorithm for multi-criteria optimization in CSPs. In Frank van Harmelen, editor, *ECAI 2002. Proceedings of the 15th European Conference on Artificial Intelligence*, pages 136–140, Lyon, France, July 21-26 2002. IOS Press.

[Gervet *et al.*, 1999] C. Gervet, Y. Caseau, and D. Montaut. On refining ill-defined constraint problems: A case study in iterative prototyping. In *Practial Application of Constraint Technologies and Logic Programming (PACLP) 1999*, pages 255–275, London, 1999.

[Giustolisi and Savić, 2008] Orazio Giustolisi and Dragan A. Savić. Optimal design of isolation valve system for water distribution networks. In *Proc. 10th Annual Water Distribution Systems Analysis Conf. WDSA*, 2008.

[Giustolisi and Savić, 2010] Orazio Giustolisi and Dragan A. Savić. Identification of segments and optimal isolation valve system design in water distribution networks. *Urban Water Journal*, 7(1):1–15, 2010.

[Jaffar and Maher, 1994] Joxan Jaffar and Michael J. Maher. Constraint logic programming: A survey. *Journal of Logic Programming*, 19/20:503–581, 1994.

[Jun and Loganathan, 2007] Hwandon Jun and G. V. Loganathan. Valve-controlled segments in water distribution systems. *Journal of Water Resources Planning and Management*, 133(2):145–155, March/April 2007.

[Marriot and Stuckey, 1998] K. Marriot and P.J. Stuckey. *Programming with constraints: An introduction*. 1998.

[Marriott and Stuckey, 1994] K. Marriott and P.J. Stuckey. Semantics of constraint logic programs with optimization. In H. Aït-Kaci, M. Hanus, and J.J. Moreno-Navarro, editors, *ICLP Workshop: Integration of Declarative Paradigms*, pages 23–35, 1994.

[Peano *et al.*, 2012] Andrea Peano, Maddalena Nonato, Marco Gavanelli, Stefano Alvisi, and Marco Franchini. A Bilevel Mixed Integer Linear Programming Model for Valves Location in Water Distribution Systems. In Stefan Ravizza and Penny Holborn, editors, *3rd Student Conference on Operational Research*, volume 22 of *OpenAccess Series in Informatics (OASIcs)*, pages 103–112, Dagstuhl, Germany, 2012. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik.

[Prestwich, 1996] S. Prestwich. Three implementations of branch-and-bound in CLP. In *Proceedings of Fourth Compulog-Net Workshop on Parallelism and Implementation Technologies*, Bonn, September 1996.

[Russell and Norvig, 2003] Stuart J. Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach*. Prentice Hall, 2 edition, 2003.

[Schimpf and Shen, 2012] Joachim Schimpf and Kish Shen. Ecl$^{\mathrm{i}}$ps$^{\mathrm{e}}$ - from LP to CLP. *Theory and Practice of Logic Programming*, 12(1-2):127–156, 2012.

[Van Wassenhove and Gelders, 1980] L.N. Van Wassenhove and L.F. Gelders. Solving a bicriterion scheduling problem. *European Journal of Operational Research*, 4(1):42–48, 1980.