University of Padova

Department of INFORMATION ENGINEERING

Master Thesis in INGEGNERIA DELL'AUTOMAZIONE

Water distribution network control via adaptive consensus

Supervisor
Prof. Angelo Cenedese
University of Padova

Co-supervisor
Dr. Marco Fabris
University of Padova

Master Candidate
Marco Davide Bellinazzi

Student ID
1242063

Academic Year
2021-2022

Abstract

This thesis deals with the analysis of water management in a network of canals, focusing on the Cavallino di Venezia drainage network. In particular, it is proposed an improvement for the water distribution of this site. Aiming to find a feasible solution to this issue, it is derived a nonautonomous difference-equation-based model that considers the constraints imposed on a general water network.
The presented approach primarily leverages graph-theoretical tools and, by means of a modified version of the distributed consensus protocol accounting for the physical restrictions within the network, the water exchange among nodes is regulated at a common level given by the average of the initial heights.
The main contribution of this thesis is thus devoted to the adaptation of the classic consensus protocol to this specific framework, by introducting a time-variant adjustment to cope with different water regimes occurring at each canal loading or draining. In addition, the proposed solution is shown to have general application properties, namely it is designed to be suitable for many frameworks in which the consensus protocol needs to take into account restricted capacity of information exchange. This aspect is explored through a theoretical study on the convergence of the developed distributed algorithm towards the state agreement and, consequently, an analytical metric for the convergence rate is suggested. Finally, to support the obtained theoretical results, numerical simulations using MATLAB are also reported and the devised distributed algorithm is applied to the Cavallino di Venezia drainage network in order to show its performances in a real scenario.

Abstract

Questa tesi si occupa della gestione dell'acqua facente parte di un sistema di canali e, in dettaglio, viene preso in considerazione il sistema di canali della città di Cavallino di Venezia. In particolare, viene proposto un miglioramento nella redistribuzione delle acque di questo sito. Con l'obbiettivo di trovare una soluzione per questo tipo di problema, è proposto un modello non autonomo di equazioni alle differenze in grado di considerare i limiti fisici di un generico sistema di canali. L' approccio descritto fa leva sulla teoria dei grafi, e mediante una versione modificata del già noto protocollo di consensus distribuito, viene tenuto conto delle restrizioni fisiche del sistema, e l' altezza dell'acqua nei vari nodi del sistema viene portata verso la media delle altezze iniziali.
Il contributo principale di questa tesi è l'adattamento del protocollo del consensus a questo campo specifico. Ciò viene raggiunto introducendo parametri tempo varianti calcolati in base alla capacità del sistema di caricare e scaricare acqua da un canale all'altro. Inoltre, la soluzione trovata presenta applicabilità generica per tutti i sistemi riconducibili a grafi che risentono di ridotte capacità di trasmissione tra i propri nodi. Questo aspetto è approfondito tramite uno studio teorico sulla convergenza dell' algoritmo distribuito proposto e, conseguentemente, viene descritta una metrica analitica per la velocità di convergenza dello stesso. Infine, a supporto dei risultati teorici ottenuti, vengono effettuate delle simulazioni numeriche in ambiente MATLAB testando l'algoritmo proposto per gestire le acque della rete di canali di Cavallino di Venezia in modo da valutarne le prestazi- oni in uno scenario reale.

Contents

Abstract iii

List of figures x

List of tables xii

1 Introduction 1
- 1.1 State of the art . . . 3
- 1.2 Thesis contributions . . . 7
- 1.3 Thesis outline . . . 8
- 1.4 Notation . . . 9

2 Theoretical background 13
- 2.1 Basics on graph theory . . . 13
  - 2.1.1 General definitions . . . 13
  - 2.1.2 Matrices defined over graphs . . . 15
  - 2.1.3 Properties of undirected graphs . . . 18
  - 2.1.4 Adjoint graph . . . 22
- 2.2 Elements of consensus theory . . . 23
  - 2.2.1 Consensus dynamics . . . 24
  - 2.2.2 Properties of the consensus matrix . . . 25
  - 2.2.3 Elements of Perron-Frobenius theory . . . 27
  - 2.2.4 Convergence properties of the consensus protocol 28
  - 2.2.5 Convergence rate of consensus . . . 30
  - 2.2.6 Consensus over networks . . . 30

3 Model design 33
- 3.1 Water allocation via distributed consensus . . . 34
  - 3.1.1 Design of the consensus matrix . . . 35
  - 3.1.2 Regular bipartite graphs: convergence . . . 36
- 3.2 Handling of the network constraints . . . 37
  - 3.2.1 Derivation of parameter $\eta(k)$ . . . 39

3.2.2 Tuning of parameter $\eta(k)$ 42

4 Water distribution algorithm **45**
4.1 Premise 45
4.2 Description of the main algorithm 47
4.3 Execution example on a regular bipartite graph 49

5 Convergence analysis **55**
5.1 Max-min disagreement 55
5.2 Convergence over time-varying digraphs connected over time 58
5.3 Induced convergence metric 61

6 Numerical results **65**
6.1 Real scenario simulation with identical constraint values 65
6.2 Real scenario simulation with different constraint values 73
6.3 Real scenario simulation with variable constraint values 76

7 Conclusions **81**

A Appendix **83**
A.1 On the minimum entry of the Metropolis-Hasting matrix 83
A.2 Distributed computation of $\left\|x(k)\right\|_\infty$ 86

References **89**

Listing of figures

1.1 Satellite view of Cavallino-Treporti and its canal network 2
1.2 Water Distribution Network, time frames division [1] . 4

2.1 Graph and digraph used as examples in the following . . 15
2.2 Gershgorin disks theorem: eigenvalues for matrix A lay in the grey area 20
2.3 Eigenvalues of the Laplacian matrix lay in the grey area . 21
2.4 Left:Graph G-Right: J_G, the adjoint graph of G 23
2.5 Relations among non-negative matrices 27
2.6 Graph example. In blue and green are highlighted the path from node 1 to 2 of length 2. 31

4.1 4-canal graph F from Table 4.1 data (initial state) 49
4.2 4-canal adjoint graph G from Table 4.1 data 50
4.3 Adaptive consensus algorithm 4.1 applied on the proposed example. 51
4.4 Graphical comparison between $c_{max}(k)$ and $||x(k+1) - x(k)||_\infty$. 52
4.5 comparison between $V_{max-min}(x(k))$ and γ 52
4.6 Adjoint graph G result from Table 4.1 53
4.7 Table 4.1 graph F result 53

6.1 Satellite view of Cavallino-Treporti with highlighted canal network 66
6.2 Simplified model of the Cavallino-Treporti canal network with canal heights from Table 6.1 69
6.3 Adaptive consensus algorithm applied on the Cavallino-Treporti canal network 69
6.4 Adaptive consensus algorithm applied on the Cavallino-Treporti canal network, highlight on iteration 11 70
6.5 Graphical comparison between c_{max} and $||x(k+1) - x(k)||_\infty$. 71

6.6 Simplified model of the Cavallino-Treporti canal network with final states 71
6.7 comparison between $V_{max-min}(x(k))$ and γ 73
6.8 Adaptive consensus algorithm applied on the Cavallino-Treporti canal network with $c_{max,download} > c_{max,upload}$ . . . 74
6.9 Graphical comparison between $c_{max,upload}$, $c_{max,download}$ and $||x(k+1) - x(k)||_\infty$. 75
6.10 comparison between $V_{max-min}(x(k))$ and γ 75
6.11 Simplified model of the Cavallino-Treporti canal network with final states: different constraints case 76
6.12 Adaptive consensus algorithm applied on the Cavallino-Treporti canal network with variable constraints 77
6.13 comparison between $V_{max-min}(x(k))$ and γ 78
6.14 Simplified model of the Cavallino-Treporti canal network with final states: variable constraints case 78
6.15 Graphical comparison between $c_{max,upload}(k)$, $c_{max,download}(k)$ and $||x(k+1) - x(k)||_\infty$. 79

Listing of tables

4.1 Initial states of the given 4-canal network 49
4.2 Final states of the given 4-canal network 53

6.1 Initial informations of the Cavallino-Treporti water network 68
6.2 η selection 70
6.3 Final states of the Cavallino-Treporti water network . . 72
6.4 η selection in $c_{max,download} > c_{max,upload}$ case 74

1

Introduction

This thesis focuses on the Cavallino Treporti drainage water network. Cavallino Treporti is a long and thin peninsula that separates the north-eastern Venetian lagoon from the Adriatic Sea. The peninsula is also separated from the land by the river Sile on the north-east side and it extends toward south-west into the Venetian lagoon. Figure 1.1 shows this piece of land highlighting the water network that will be under analysis.

Figure 1.1: Satellite view of Cavallino-Treporti and its canal network

Note that the network is composed of open canals and pipelines. Indeed, it is crossed by the Saccagnana canal, which divides the network into two parts: the main subnetwork is the one of the peninsula facing onto the Adriatic Sea and a second subnetwork extends amidst a urban area located further inland in the lagoon. The two networks are connected through underground pipelines that lies on the Saccagnana canal. Both subnetworks communicate through hydraulic structures (gates) with the lagoon, rivers and sea surrounding the peninsula.
The peculiarity of this territory is that it is located under the sea level and it is protected by an elevated coastline with gates that can isolate the network from the external water system. Sharing water junctions with the sea, affects the lagoon surrounding the Cavallino peninsula by tides. In addition to the normal fluctuation of the water level generated by tides, the entire lagoon area is subject to a phenomenon called Acqua alta [2]. This phenomenon is caused by a combination of astronomical, meteorological and geological events. This problem often occurs during rainy periods and requires the operators to isolate the Cavallino network from the outside environment to prevent flooding in the Urban area of the

Cavallino territory. However, this strategy does not allow the possibility for the network to drain part of the water outside. The Cavallino water network is not designed to withstand such extreme phenomena; indeed, the previously mentioned events in combination with violent rainfalls cause flooding on the Urban area of the Cavallino territory.
The aforementioned fact depicts a system which, in the worst case scenario, has the issue to stay isolated from the external water network and, possibly, with an excess of water load in at least some of the canals. The main objective of this thesis is to avoid this kind of issue by designing an algorithm that controls and distributes the water evenly throughout the network, so to be able to reach the largest admissible water capacity.

1.1 State of the art

In this section, the state of the art about water distribution problem is presented. In general, there are many different approaches to water distribution problem. This is due to the fact that there are many aspects eventually to consider, such as cost reduction or optimization, control optimization, supply or allocation problem, leaks management. Also the time variable could be key in this problem, as in [1], where it is presented a detailed review of the management problems and essential mathematical models that are divided into short (STP)-medium (MTP)-long (LTP) term framework categories. Figure 1.2 gives a wide picture about the variability and difference that this kind of problem in function of the time can offer.

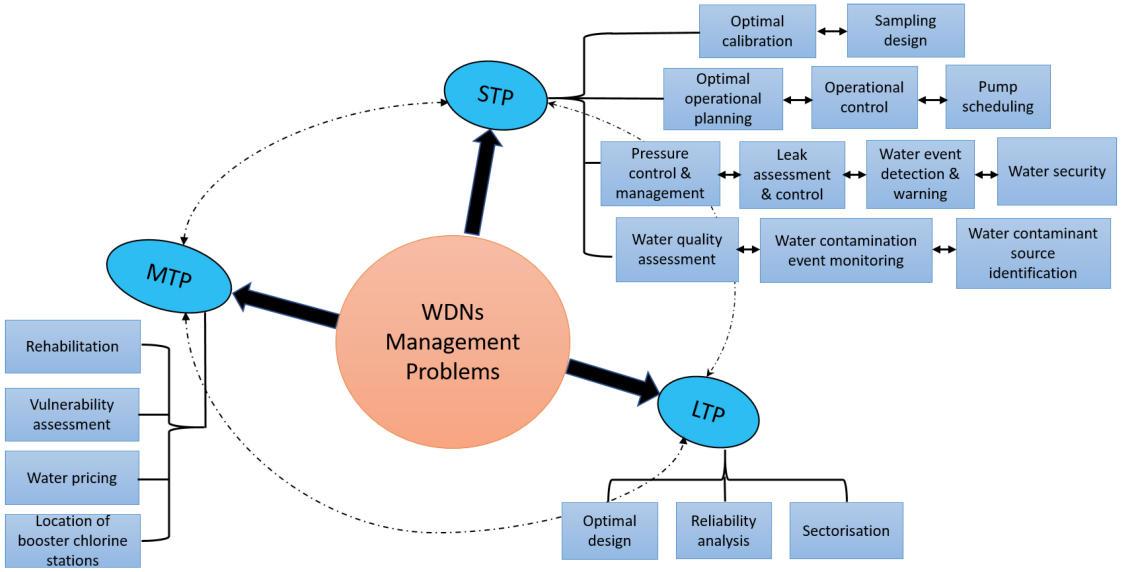

Figure 1.2: Water Distribution Network, time frames division [1]

All these aspects are usually taken into account separately since their nature is very different between one another. For this reason, different kind of solutions to the problem are requested. Network design is a good example of a long term framework application and could esemplify well the starting point addressing such a problem. In [3], the implementation of a smart water grid system is presented and supports the public utilities board's mission to supply good water 24/7 to its customers; another solution is presented in [4] where a stochastic formulation validated through Monte Carlo simulation is applied in order to fix uncertainty in demand in designing framework. Together with the design purpose comes the reliability of the system as main object of the LTP and models leveraging stochastic solutions are preferred. An example of this approach is yielded by [5] where a new approach for reliability-based optimization of water distribution networks is presented and is capable of recognizing the uncertainty in nodal demands and pipe capacity as well as the effects of mechanical failure of system components.
MTP framework categories include for example vulnerability assessment as in [6], where some fundamental concepts are highlighted from graph theory for vulnerability assessment of water distribution networks.

More than the previous two cases, the STP framework gives us a better picture to what this thesis is about. This category includes the majority of the aspects than can be studied with automated systems, and usually have an immediate and adaptive contribute to the controlled parameters. The vast majority of solutions in this category comprehends machine learning tools and graph-based algorithms.
Machine learning tools are usually employed for leak and infiltration assessment as in [7], where it is shown a comparison between statistical and machine learning models for pipe failure modeling prediction or, as in [8] where machine learning is exploited in order to determine and predict water contamination.
As machine learning is often exploited for quality issues, graph theory is best employed for more operational planning problems. Examples of graph theory applied to water network distribution issues are very common and used to solve different tasks; for example, in [9], graph theory is used to create an algorithm to manage the scheduling of the water network. In particular their algorithm returns an optimal minimal cost pump-scheduling pattern; whereas, in it is [10] it is introduced a holistic analysis framework to support water utilities on the decision making process for an efficient supply management. Furthermore, within graph theory, it's common opinion [11],[12] that theoretical computer science is in a vantage point to achieve significant advances for what concerns the understanding of key emergent properties in complex systems.
From this perspective, we can certainly model water distribution problems by using distributed networked systems. Given this premise and willing to dive more deeply into graph theory related to water distribution problem, we found consensus algorithm as one of the most applied solution.
In networks of agents, "consensus" means to reach an agreement regarding a certain quantity of interest that depends on the state of all agents. A consensus algorithm (or protocol) is an interaction rule that specifies the information exchange between an agent and all of its neighbors on the network [13]. Consensus can be applied, in general, for multiple purposes, as for example in [14], where it is provided a theoretical frame-

work for the analysis of consensus algorithms concerning multi-agent networked systems. This work emphasizes the role of directed information flow and discusses the connections between consensus problems in networked dynamic systems.
More relevant to our specific case we find [15], where a consensus-based control strategy for a water distribution system is proposed. The latter enables a water system to continuously supply the demand minimizing the impact of faulty equipment within the water distribution system facilities. Within the consensus literature, it is possible to find several applications, as in [16] where it is proposed a study of average consensus problem of multi-agent systems for general network topologies with unidirectional information flow. They propose two linear distributed algorithms, deterministic and gossip, respectively for the cases where the inter-agent communication is synchronous and asynchronous. In both cases, the developed algorithms guarantee state averaging on arbitrary strongly connected digraphs.
Discrete consensus in a dynamic networks is significantly similar to what we are describing in this thesis. Some prior examples are given also in [17], where it is proposed a class of discrete-time dynamic average consensus algorithms that allow a group of agents to track the average of their reference inputs. The corrisponding convergence results rely on the input-to-output stability properties of static average consensus algorithms and require that the union of communication graphs over a bounded period of time be strongly connected.
Another work that has very similar scope to this thesis is presented in [18] through a multi-agent assignment problem, in which a group of agents has to reach a consensus on an optimal distribution of tasks under communication and assignment constraints. In this paper, starting from any unfeasible solution, it is developed an extended version of the gossip algorithm able to iteratively find an initial feasible assignment state in order to reach the consensus state among the agents. Constraints are another aspect that consensus algorithm theory takes into account, as in [19] where it is considered the global consensus problem for discrete-time multi-agent systems with input saturation constraints under fixed undi-

rected topologies. Firstly, necessary conditions are given for achieving global consensus via a distributed protocol based on relative state measurements of the agent itself and its neighboring agents. Then, the focus is is directed towards two special cases, where the agent model is either neutrally stable or a double integrator.
In presence of constraints consensus algorithm usually need to be adapted, as in [20] in which it is presented a study of consensus for a team of agents with continuous dynamics in the presence of state constraints. Due to the existence of state constraints, most existing consensus algorithms cannot be applied directly and, thus, a novel consensus algorithm is proposed to deal with state constraints. The novel consensus algorithm is shown to guarantee consensus when a few conditions are satisfied.
Even though the literature is full of examples in which consensus algorithm is exploited and modified in order to adapt to a particular case, there are still some gaps to fill. In particular, speaking about constrained consensus algorithm, it is missing the case where the constraints are applied to the capability of the system to exchange information between one another.

1.2 Thesis contributions

The objective of this thesis is the control of the water height in every canal of the Cavallino network.
In particular, as it is desired to obtain the maximum water capacity throughout the considered hydraulic system, we devise a distributed algorithm to set the water height of each canal at the same level. Such an approach is expected to cope with the physical constraints of the network involved.
This thesis mainly deals with the well known consensus protocol. The consensus protocol, in general, could be interpreted as a graph based protocol that, under appropriate hypotesis, ensures the convergence of the system nodes to a common value also called agreement value. We adapted this protocol in order to satisfy the main physical limits that a real mechanical system can exhibit. The rationale presented in this thesis is ori-

ented toward finding a solution to this issue, so to proper describe and simulate the application of consensus protocol in a physical environment, such as that of water channels. In particular, we manage to adapt the classic consensus protocol introducing non-linear constraints that can be satisfied thanks to the proposed tunable parameter [21][22][23]. This parameter regulates the rate of information exchanged between nodes and, thus, allows to satisfy requirements on limited water flow capacity. Clearly, the approach proposed can be also applied to any other situation in which the same framework is recognized. On top of that, we provide a convergence metric study of the proposed protocol to evaluate the performance of this new approach. Finally, to support the theoretical results obtained, we report some numerical simulations on a real nominal and faulty scenarios. The simulations show both the correctness, robustness and the versatility of the devised adaptive algorithm to the different proposed frameworks.

1.3 Thesis outline

This thesis aims at proposing a solution to the water management of Cavallino Treporti drainage water network.
In doing so, Section 2 presents some basic theoretical tools that will be exploited in the subsequent chapters.
In particular basics of graph theory and consensus theory are reported.
Chapter 3 describes how we derive a feasible model that takes into account the main features and limits that a structure like a water network could intrinsically have.
In chapter 4, the derived model and constraints are exploited into the proposed algorithm to control the water network. On top on that, a simple example of the algorithm execution is presented.
Chapter 5 introduces a convergence analysis of the algorithm previously mentioned. In particular, a convergence metric is then computed in order to evaluate the algorithm performances in relation to the constraints applied.

In Chapter 6, exploiting the MATLAB environment, the algorithm is applied to a simplified version of the Cavallino Treporti drainage water network in order to simulate how the algorithm behaves in a real case scenario. Furthermore, some numerical consideration on the convergence metric are discussed.
Chapter 7 draws the conclusions of the entire work.
Finally, in Appendix A one can find useful computations in order to fully understand the mathematical derivations.

1.4 Notation

In this section, the notation used in this thesis is presented.

- G: graph
- J_G: adjoint graph of G
- V: set of nodes
- E: set of edges
- v_i: i^{th} vertex or node
- e_{ij}: edge connecting v_i and v_j
- d_i: degree of v_i
- k: iteration counter variable
- $N(v_i)$ or N_i: neighborhood of v_i
- $|\cdot|$: absolute value or cardinality
- n: number of nodes
- m: number of edges
- Δ_G: degree matrix of G
- A_G: adjacency matrix of G

- E_G: incidence matrix of G
- P_{ij}: element in the i^{th} raw and j^{th} column of P
- P: consensus matrix
- P_J: Jordan form of P
- L_G: Laplacian of G
- I_n: identity matrix of dimension n
- $1_n = \begin{bmatrix} 1 \\ 1 \\ ... \\ 1 \end{bmatrix}$: n dimensional vector of ones
- $||x||_2 = (\sum_{i=1}^{n} |x_i^2|)^{\frac{1}{2}}$
- $||x||_\infty = max_i |x_i|$
- λ_i^L: i^{th} eigenvalue of L
- $\mathbb{R}^{n \times n}$: $n \times n$ domain real numbers
- $\mathbb{Z}$: integer number domain
- $\mathbb{N}$: natural number domain
- $\sigma(L_G)$: spectrum of L_G
- $x(k)$: state vector of n dimension at iteration k
- $x_i(k)$: i^{th} state at iteration k
- x_{max}: maximum element of vector x
- x_{min}: minimum element of vector x
- T: change of basis matrix
- w^T: transposed form of vector w
- w_0: eigenvector of P and first raw of T
- v_0: eigenvector of P and first column of T^{-1}

- V_T: $n \times n-1$ dimensional matrix representing T^{-1} without v_0
- W_T: $n-1 \times n$ dimensional matrix representing T without w_0^T
- Λ: $n-1 \times n-1$ diagonal matrix containing the eigenvalue of P without λ_0

2 Theoretical background

This chapter is aimed to give the theoretical background needed to create and implement the solutions and simulations proposed. Starting from basic graph theory, a list of the main definitions about graphs is given. Secondly, the matrix representations of graphs are briefly explained with an overview on the principal characterizing properties. Then, the consensus problem is introduced and discussed within the presented graph-matrix framework.

2.1 Basics on graph theory

This section presents a list of the basic notions and definitions on graph theory, since the latter provides several tools for studying the consensus theory.

2.1.1 General definitions

Definition 1 (Graph) *An (undirected) graph is a pair $G = (V, E)$, where $V = \{v_1, \ldots, v_n\}$ is the set of the nodes endowed with a state x and $E \subseteq V \times V$ is the set of edges, i.e. connections between vertices.*

Definition 2 (Neighbor) *v_i, v_j are said to be neighbors iff $e_{ij} \in E$.*

Definition 3 (Neighborhood) *Given a vertex v_i, its neighborhood is defined as the set $N(v_i) = N_i = \{v_j : e_{ij} \in E\}$*

Definition 4 (Degree) *The number of neighbors of node i is called degree of node i.*

Definition 5 (Regular Graph) *A graph is said to be regular if all its nodes have the same degree.*

Definition 6 (Path) *A path is a sequence of neighbouring nodes. If it is closed it is called a cycle.*

Definition 7 (Connected Graph) *A graph is connected if given $\forall v_i, v_j$ there exists a path connecting v_i e v_j.*

Definition 8 (Subgraph) *A graph whose vertices and edges are subsets of another graph.*

Definition 9 (Spanning Tree) *A spanning tree is a subgraph that connects all the vertices in V with some of the edges in E, without creating cycles.*

Definition 10 (Weighted Graph) *Let $w : E \to \mathbb{R}$ be a real positive function, a weighted graph is defined as the triplet $G_w = (V, w, E)$.*
This function allow us to define the length of a path as $\sum_{i \in path} w_i$.

Definition 11 (Geodesic) *A geodesic is the path between two nodes with minimum length.*

Definition 12 (Diameter) *The length of the longest geodesic is called diameter. In particular, let G be a graph, φ_G is the diameter of G.*

Definition 13 (Directed graph/Digraph) *A directed graph (also called digraph) is a graph where the edges have a starting and an ending node called tail and head, respectively.*

Definition 14 (Connected Digraph) *A digraph is said to be strongly connected if there exists an oriented path between any pair of nodes, weakly connected if the non-oriented version is connected.*

In the case of digraphs we distinguish between in-degree, i.e. the number of incoming edges of a node, and out-degree, i.e. the number of outgoing edges of a node. These distinction brings to define in-neighbours and out-neighbours

In order to model situations in which the nodes have a memory of their own state we introduce the following:

Definition 15 (Self Loops) *Self-loops are edges where tail and head coincide.*

Definition 16 (Simple Graph) *A simple graph, also called a strict graph, is an unweighted, undirected graph containing no graph loops or multiple edges. A simple graph may be either connected or disconnected.*

2.1.2 Matrices defined over graphs

In this section, we define the matrices used to characterize a graph (directed and undirected) and their properties. The graph will be denoted as $G = (V, E)$ with $|V| = n$ number of nodes and $|E| = m$ number of edges.

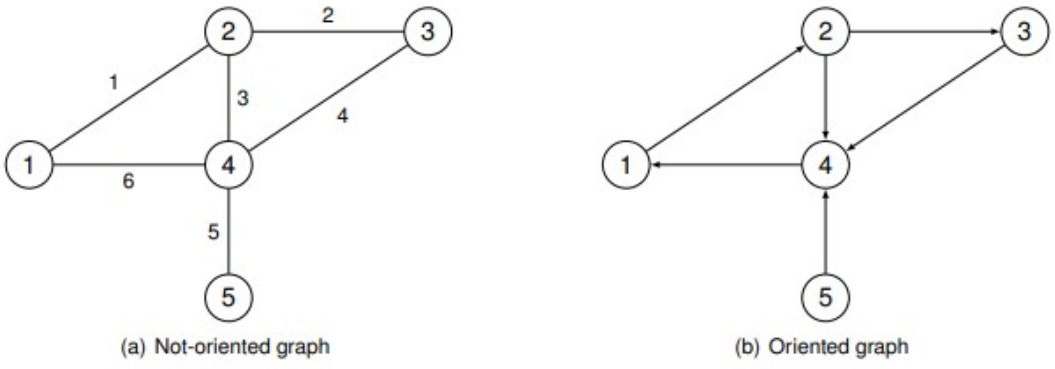

Figure 2.1: Graph and digraph used as examples in the following

Definition 17 (Node Degree Matrix) *The node degree matrix of a (non-oriented) graph G is the diagonal matrix $\Delta_G \in \mathbb{R}^{n \times n}$ with entries the degrees of the nodes. For oriented graphs, the diagonal matrices $\Delta_{G_{IN}}$ and $\Delta_{G_{OUT}}$ are defined, whose elements are, respectively, the in-degrees and out-degrees of the nodes.*

Consider, for example, the non-oriented and oriented graphs in Figure 2.1. The node degree matrices for the graphs are:

$$\Delta_G = \begin{bmatrix} 2 & 0 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 4 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \quad \text{(non-oriented)}$$

$$\Delta_{G_{OUT}} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \text{(oriented)} \quad \Delta_{G_{IN}} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \text{(oriented)}$$

Now, we define a few matrices that represent the relations among nodes in the graph.

Definition 18 (Adjacency Matrix) *The adjacency matrix $A_G \in \mathbb{R}^{n \times n}$ of an undirected graph without self-loops is defined as:*

$$A_G(i,j) = \begin{cases} 1 & \text{if } (i,j) \in E \\ 0 & \text{otherwise} \end{cases}$$

$$A_G = \begin{bmatrix} 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix} \text{(non-oriented).}$$

Whereas for a directed graph without self-loops it is:

$$A_G(i,j) = \begin{cases} 1 & if \quad (i \to j) \in E \\ 0 & otherwise \end{cases}$$

$$A_G = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix} \text{(oriented).}$$

Note that, in the undirected case matrix A_G is symmetric and, in the directed case it is, in general, non-symmetric.

So far, we have not considered self-loops. If the graph has self-loops, we can extend the definition by choosing:

$$A_G(i,i) = 1 \qquad if \quad (i,i)/(i \to i) \in E$$

By means of the incidence matrix, we describe the relations between nodes and edges in the graph.

Definition 19 (Incidence Matrix) *The incidence matrix $E_G(i,k) \in \mathbb{R}^{n \times m}$ of an directed graph is defined as follows:*

$$E_G(i,k) = \begin{cases} +1 & if \quad v_i \quad tail \quad of \quad e_k \\ -1 & if \quad v_i \quad head \quad of \quad e_k \\ 0 & otherwise \end{cases}$$

$$E_G(i,k) = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & -1 \\ -1 & 1 & 1 & 1 & 0 & 0 \\ 0 & -1 & 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & -1 & -1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} \textit{(oriented)}$$

Note that the sum of the elements in a column is 0.
In order to describe graph theory we also need the following definitions.
A first simple classification of nonnegative matrices is as follows:

Definition 20 *Given a generic matrix $P \in \mathbb{R}^{n\times n}$*

- *if we have P such that $P_{ij} > 0$, then we say that P is positive: $P > 0$;*
- *if we have P such that $P_{ij} \geq 0$, then we say that P is non negative: $P \geq 0$.*

And more in detail:

Definition 21 *A generic non negative matrix $P \in \mathbb{R}^{n\times n}$ is:*

- *irreducible if and only if*
$$\sum_{k=0}^{n-1} p^k > 0;$$
- *primitive if and only if $\exists \bar{k} \in \mathbb{N}$ such that:*
$$P^{\bar{k}} > 0.$$

2.1.3 Properties of undirected graphs

Consider now only the case of undirected graphs without self-loops.

Definition 22 (Laplacian of a Graph) *The Laplacian of a graph* $L_G \in \mathbb{R}^{n\times n}$ *is:*

$$L_G = \Delta_G - A_G$$

$$L_G = \begin{bmatrix} 2 & -1 & 0 & -1 & 0 \\ -1 & 3 & -1 & -1 & 0 \\ 0 & -1 & 2 & -1 & 0 \\ -1 & -1 & -1 & 4 & -1 \\ 0 & 0 & 0 & -1 & 1 \end{bmatrix} \text{(non-oriented).}$$

If we choose an arbitrary orientation of the edges, we can further characterize the Laplacian through its incidence matrix as follows:

$$L_G = E_G E_G^T.$$

Proposition 1 (Properties of the Laplacian Matrix) *Here, some notable properties of the Laplacian matrix are given.*

- *if* L_G *is symmetric, then its eigenvalues* $\lambda_i^{L_G}$ *are real.*
- *The sum by rows and columns is equal to zero.*
- $L_G 1_n = 0 1_n$*, i.e.* 0 *is eigenvalue of the Laplacian with eigenvector* 1_n*.*
- L_G *is positive semidefinite. Indeed, by the characterization of the Laplacian through the incidence matrix, and denoting the state of the graph with* x*, it follows that:*

  $$x^T L_G x = x^T E_G E_G^T x = ||E_G^T x||_2^2 \geq 0$$

  herefore the eigenvalues are all non-negative.
- *By combining the above properties it follows that:*

  $$0 = \lambda_0^{L_G} \leq \lambda_1^{L_G} \leq \ldots \leq \lambda_{n-1}^{L_G}$$

The spectrum of the Laplacian $\sigma(L_G)$ will be also denoted as $\sigma(G)$ to characterize the eigenvalues of the graph.

We present, now, some results for the spectral analysis of the Laplacian eigenvalues.

Theorem 1 (Gershgorin Disks) *Given any square matrix $A \in \mathbb{R}^{n \times n}$, it holds that:*

$$\sigma(A) \subseteq \bigcup_{i=1,\ldots,N} \{z \in \mathbb{C} : |z - a_{ii}| \leq \sum_{j \neq i} |a_{ij}|\}$$

i.e. as shown in Figure 2.2 the the spectrum of A is included in the union of all circles centered in $(a_{ii}, 0)$ of radius $\sum_{j \neq i} |a_{ij}|$

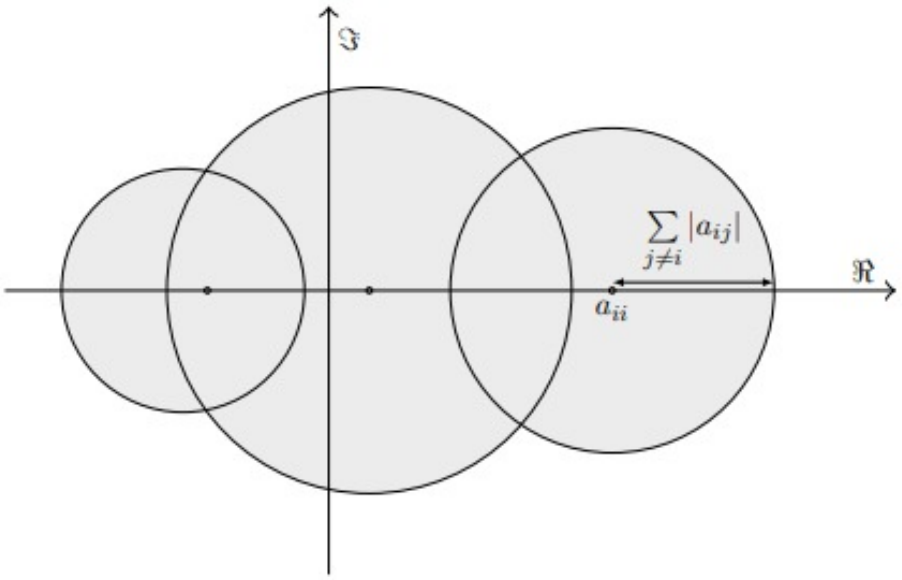

Figure 2.2: Gershgorin disks theorem: eigenvalues for matrix A lay in the grey area

Let's see an application of the Gershgorin Theorem to the Laplacian matrix. We know that the sum by rows for the Laplacian matrix is equal to zero. Therefore the circles defined in the Gershgorin Theorem are centered in $(d_i, 0)$, where d_i is the degree of the i-th node, and have radius equal to d_i. It follows that the eigenvalues of the Laplacian are all in the

interval $[0, 2d_{MAX}]$, where $d_{MAX} = max\{d_i, i = 1, \ldots, N\}$; i.e. they satisfy the relation:

$$0 = \lambda_0 \leq \ldots \leq \lambda_{n-1} \leq 2d_{MAX}. \tag{2.1}$$

For the example in Figure 2.3, we have $\sigma(L_G) = \{0, 1, 2, 4, 5\}$ and $2d_{MAX} = 8$. The placement of eigenvalues is shown in Figure 2.3. The red crosses indicate the Laplacian eigenvalues.

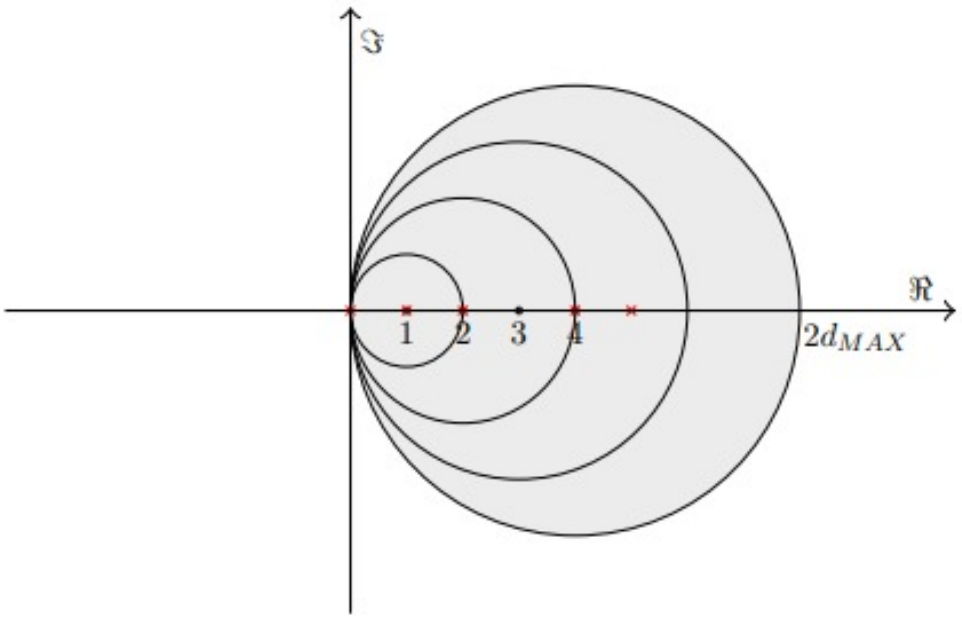

Figure 2.3: Eigenvalues of the Laplacian matrix lay in the grey area

With the following theorem we state a condition for the connectivity of a graph.

Theorem 2 (Connectivity of a Graph) *A non-oriented graph G is connected if and only if $\lambda_1^G > 0$, where λ_1 denotes the second lowest eigenvalue of the graph and it is called the Fiedler's value.*

From the theorem above it follows that, if a graph is connected, then it has only one zero eigenvalue. Furthermore, it can be proven as a corollary that the number of null eigenvalues is equal to the number of connected

components in the graph. In the case of a graph with more than one connected component, the Laplacian matrix can be put into a block-diagonal form, following a permutation of the nodes.

Remark 1 *In addition, it is possible to yield some connectivity bounds of particular interest. In a connected graph*

$$\lambda_1^G \leq k_V^G \leq k_\varepsilon^G \leq min\{d_i\}$$

where:

- λ_1^G*: is the first non-null eigenvalue;*
- $k_V(G)$*: is the node connectivity, namely the minimum number of nodes that can be removed to make the graph disconnected;*
- $k_\varepsilon(G)$*: is the edge connectivity, namely the minimum number of edges that can be cut to make the graph disconnected;*
- $min\{d_i\}$*: is the minimum node degree.*

2.1.4 Adjoint graph

Definition 23 (Adjoint graph) *An adjoint graph J_G (also called a line graph, conjugate, covering, derivative, derived, edge, edge-to-vertex dual, interchange, representative, or theta-obrazom graph) of a simple graph G is obtained by associating a vertex to each edge of the graph and connecting two vertices with an edge if and only if the corresponding edges of G have a vertex in common [24].*

Figure 2.4 show an example of a graph G and its adjoint graph J_G

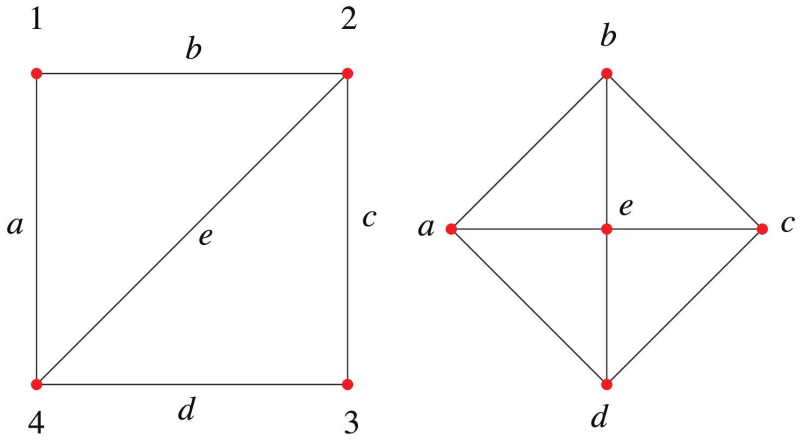

Figure 2.4: Left:Graph G-Right: J_G, the adjoint graph of G

The adjoint graph of a graph with n nodes, m edges, and vertex degrees d_i contains $n' = m$ nodes and

$$m' = \frac{1}{2}\sum_{i=1}^{n} d_i^2 - m$$

edges [25][26]. The incidence matrix E of a graph and adjacency matrix E of its adjoint graph are related by

$$J_G = E^T E - 2I_{n'}.$$

Trivially, the connectivity is mantained in J_G if previously present in G.

2.2 Elements of consensus theory

We introduce the problem of consensus with reference to one specific applications properly modified to our purpose: convergence to the mean of the initial conditions. We want to exploit consensus theory and apply it to our specific case of study. We consider an autonomous system of the form

$$x(k+1) = Px(k) \tag{2.2}$$

and we want the state to converge to a unique solution.
In this model, $x(k) \in \mathbb{R}^n$ stands for the state vector, $k \in \mathbb{N}$ is the iteration or time variable and $P \in \mathbb{R}^{n\times n}$ is a square matrix that allow us to control

the system.
The main idea behind this algorithm is the following:

- exchange state information between nodes;
- update current state.

2.2.1 Consensus dynamics

Let's consider the update rule:

$$x_i(k+1) = p_{i1}x_1(k) + p_{i2}x_2(k) + ... + p_{in}x_n(k)$$

where

$$p_{ij} \geq 0 \quad \text{and} \quad \sum_{j=1}^{n} p_{ij} = 1 \tag{2.3}$$

and $x_i(k)$ is the state of the agent i at time k. Note that p_{ij} can be null if the agent i does not exchange information with the agent j. Considering the state x_i in the system, we can notice that

$$\begin{aligned} x_i(k+1) &= \sum_j p_{ij}x_j(k) \\ &= p_{ii}x_i + \sum_{j\neq i} p_{ij}x_j(k) \\ &= (1 - \sum_{j\neq i} p_{ij})x_i(k) + \sum_{j\neq i} p_{ij}x_j(k) \\ &= x_i(k) + \sum_{j\neq i} p_{ij}(x_j(k) - x_i(k)) \end{aligned}$$

where the left hand side after the equality symbol is the state and the right hand side is the state dependent input $u_i(k) = \sum_{j\neq i} p_{ij}(x_j(k) - x_i(k))$. If we consider the problem in matrix notation

$$x(k+1) = \begin{bmatrix} x_1(k) \\ x_2(k) \\ * \\ x_n(k) \end{bmatrix} + \underbrace{\begin{bmatrix} (-p_{12} - p_{13}...) & p_{12} & p_{13} & * & p_{1n} \\ p_{21} & (-p_{21} - p_{23}...) & p_{23} & * & p_{2n} \\ * & * & * & * & * \\ p_{n1} & p_{n2} & p_{n3} & * & (-p_{n1} - p_{n2}...) \end{bmatrix}}_{F} \begin{bmatrix} x_1(k) \\ x_2(k) \\ * \\ x_n(k) \end{bmatrix}$$

From this, we end up having

$$x(k+1) = x(k) + Fx(k) = (I_n + F)\, x(k) = Px(k)$$

which is the autonomous system we presented in the beginning of Section (2.2), with P provided by

$$P = \begin{bmatrix} (1 - p_{12} - p_{13}....) & p_{12} & p_{13} & * & p_{1n} \\ p_{21} & (1 - p_{21} - p_{23}...) & p_{23} & * & p_{2n} \\ * & * & * & * & * \\ p_{n1} & p_{n2} & p_{n3} & * & (1 - p_{n1} - p_{n2}...) \end{bmatrix}$$

We recognize that P is a stochastic matrix, that means that all its entries are non-negative and all its rows sum to 1. Then, since P is stochastic, it follows

$$P1_n = 11_n$$

which means that 1_n is the right eigenvector relative to the eigenvalue 1. In this case, we would like that the consensus problem converges to a unique solution $\bar{P}x(0) = \alpha 1_n$, and in particular, we want α to be the average of the initial states.

2.2.2 Properties of the consensus matrix

We consider the consensus protocol (in discrete time) presented in Section 2.2 and before stating the theorem that characterizes its convergence properties, we provide some intuition behind that, by given the following desiderata: we want P such that there is a dominant eigenvalue λ_0 with a Jordan block of dimension 1. Note that in general P is non-negative,

meaning that all the entries are greater or equal than 0.
Being $\sigma(P)$ the spectrum of P, $\rho(P)$ is the spectral radius of P ($\rho(P) = max\{\,|\lambda|\; s.t. \lambda \in \sigma(P)\}$).
We note that if $\rho(P) \leq 1$ and 1 is associated to block of dimension 1, all modes related to the non-dominant eigenvalue vanish as the number of iterations increases.

We ask for a dominant λ_0 with a Jordan block of dimension 1 because, in such a case, the Jordan decomposition results to be:

$$P = T^{-1} \underbrace{\begin{bmatrix} \lambda_0 & 0 \\ 0 & \Lambda \end{bmatrix}}_{P_J} T,$$

where $T \in \mathbb{R}^{n \times n}$ is the change of basis matrix to the Jordan normal form P_J composed by the generalized autovectors of P. Indeed we can notice in P_J the first eigenvalue of $P\,\lambda_0$ and Λ that is a Jordan matrix containing all the other eigenvalues. It follows that:

$$P^k = T^{-1} \begin{bmatrix} \lambda_0^k & 0 \\ 0 & \Lambda^k \end{bmatrix} T = \begin{bmatrix} v_0 & V_T \end{bmatrix} \begin{bmatrix} \lambda_0^k & 0 \\ 0 & \Lambda^k \end{bmatrix} \begin{bmatrix} w_0^T \\ W_T \end{bmatrix}.$$

Note

$$T^{-1}T = I_n \rightarrow \begin{bmatrix} v_0 & V_T \end{bmatrix} \begin{bmatrix} w_0^T \\ W_T \end{bmatrix} = \begin{bmatrix} w_0^T \\ W_T \end{bmatrix} \begin{bmatrix} v_0 & V_T \end{bmatrix} = \begin{bmatrix} w_0^T v_0 & w_0^T V_T \\ W_T v_0 & W_T V_T \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & I_{n-1} \end{bmatrix}$$

hence

$$P^k = \begin{bmatrix} v_0 \lambda_0^k & V_T \Lambda^k \end{bmatrix} \begin{bmatrix} w_0^T \\ W_T \end{bmatrix} = v_0 \lambda_0^k w_0^T + V_T \Lambda^k W_T \rightarrow_{k \rightarrow \infty} \bar{P} \approx v_0 \lambda_0^k w_0^T = \lambda_0^k (v_0 w_0^T).$$

Definition 24 (Stochastic Matrix) *A square matrix is stochastic if all of its entries are non negative, and the entries of each raw sum to* 1.
It is said to be doubly stochastic if all of its entries are non negative, and the entries of both each column and raw sum to 1.

With respect to the stochastic matrices, that appear to be our choice for

the consensus law, we have the following

Lemma 3 *If P is a stochastic matrix then:*

- 1 *is an eigenvalue with the associated eigenvector* 1_n
- $\sigma(P)$ *is a subset of the unit disk with* $\rho(P) = 1$.

2.2.3 Elements of Perron-Frobenius theory

Keeping in mind Definition 20 and Definition 21 we will show some facts.
In particular, we will see that if P is primitive then it is also irreducible but not viceversa.
The important inclusion relation among these sets of matrices is shown in Figure 2.5.

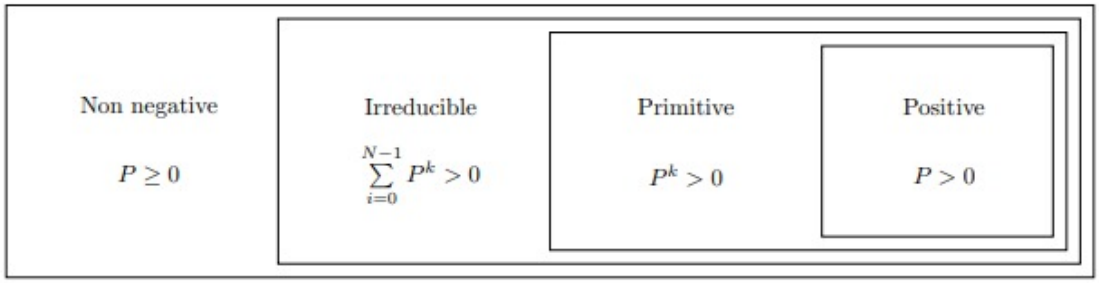

Figure 2.5: Relations among non-negative matrices

The fundamental theorem that provides a spectral characterization for these matrices is the following:

Theorem 4 (Perron-Frobenius) *Consider a matrix $P \in \mathbb{R}^{n \times n}$ with eigenvalues $\lambda_0 \geq \lambda_1 \geq \ldots \geq \lambda_{n-1}$:*

- *If P is non negative: there exists $\lambda_0 \geq 0$ such that:*

  - $\lambda_0 \geq |\lambda_i|, \forall i = 1, ..., n-1$*;*
  - *Right / left eigenvectors of P,* (v_0, w_0) *can be selected as non negative vectors,* $v_0, w_0 \geq 0$*.*

- *If P is irreducible:*
  - λ_0 *is strictly positive, i.e.* $\exists! \lambda_0 > 0$ *such that* $\lambda_0 \geq |\lambda_i|$*;*
  - *Right / left eigenvectors of P,* (v_0, w_0) *are unique and positive,* $v_0, w_0 > 0$*.*

- *If P is primitive:*
  - $\exists! \lambda_0 > 0$ *such that* $\lambda_0 > |\lambda_i|$*;*
  - *Right / left eigenvectors of P,* (v_0, w_0) *are unique and positive,* $v_0, w_0 > 0$*.*

Some observations about Theorem 4:

- λ_0 is the dominant eigenvalue and defines the spectral radius of matrix P;
- if the property of primitivity holds then it implies all the other properties it also entails that λ_0 is unique.

2.2.4 Convergence properties of the consensus protocol

Consider the linear system (2.2).

Theorem 5 *If P is primitive, with* λ_0 *dominant eigenvalue and* v_0, w_0 *eigenvectors (normalized such that* $w_1^T v_1 = 1$*), then*

$$\lim_{k \to \infty} \frac{P^k}{\lambda_0^k} = v_0 w_0^T.$$

In practice, Theorem 5 has been proved when we stated λ_0 as dominant eigenvalue and we decomposed P in Jordan form ending up with a matrix $v_0\lambda_0^k w_0^T$ as the convergence matrix. A direct consequence of this theorem is the following corollary:

Corollary 1 *If P is primitive and stochastic (by rows)*

- $\lambda_0 = \rho(P) = 1$ *is simple,* $\lambda_0 > |\lambda_i|$*, then we have semi-convergence;*
- $\bar{P} = \lim_{k\to\infty} P^k = 1_n w_0^T$ *where* w_0 *is normalized according to* $\sum_{i=1}^n w_{0i} = 1$
- *given the update rule of the model, we have* $\lim_{k\to\infty} P^k x(0) = \alpha 1_n$*, where* $\alpha = w_0^T x(0)$ *is a linear combination of the initial conditions with coefficients given by* w_0 *(consensus);*
- *If P is doubly stochastic, then* $w_0 = \frac{1}{n}$ *and* $\lim_{k\to\infty} P^k x(0) = \hat{x}(0)1_n$ *where* $\hat{x}(0) = w_0^T x(0)$ *is the average of the initial condition (average consensus).*

Remark 2 *The following considerations are in order:*

- *Note that we have* $1_n w_1^T x(0) = (w_1^T x(0))1_n$*. The value* α *is a scalar number and thus we can say that, when k tends to infinity, the state converges to* $\alpha 1_n$ *which is referred to as the agreement vector. Thinking about our problem, this is the condition we have in all cases when the consensus is attained.*
- *We stress that, from the point of view of iterative algorithms,* $x(0)$ *is the initial condition. Moreover, we think of the model as the estimation of a common variable that is the measurement we take at the sampling instant. An interesting role is played by* w_0*: it is the weighting factor of the measurements we take.*

2.2.5 Convergence rate of consensus

How fast is the state converging to consensus? We have that:

$$P^k = 1_n w_0^T + V\Lambda^k W \to \bar{P} = 1_n w_0^T$$

We can define a convergence gap as

$$e^2(k) = ||x(k) - \bar{x}||_2^2 = ||(1_n w_0^T + V\Lambda^k W)x(0) - (1w_0^T)x(0)||_2^2 = ||V\Lambda^k W)x(0)||_2^2$$

This convergence is ruled by Λ^k: it depends on the spectrum of Λ and in particular on the slowest of its modes (second largest eigenvalue of P).

2.2.6 Consensus over networks

In this section, we give some details on how the graph theory and the consensus theory are strongly related; in particular, we are going to consider a multiagent system ruled by the consensus protocol characterized by P as in Equation (2.2) and whose agents are connected through a network characterized by an adjacency matrix A.
Assume that the consensus matrix P is positive. Then, the adjacency matrix A is positive and it has the same support of the matrix P where the support of a matrix is defined as the set of the couples of indices corresponding to entries that are non-zero. More in general, we know that P positive is a particular case of P primitive. We can see that:

- $P > 0 \Rightarrow A > 0$, where A is the adjacency matrix and the two matrixes have the same support. $P > 0$ means that P is full, that is any node in the graph is connected to another node. A gives us the number of paths (sequences of edges) of length 1;
- $P^k > 0 \Rightarrow A^k > 0$. This matrix gives the number of paths of length k among nodes.

Consider the following example:

Example 1 *Consider the graph in Figure 2.6, in which* $m = |E| = 6$ *and* $n = |V| = 5$.

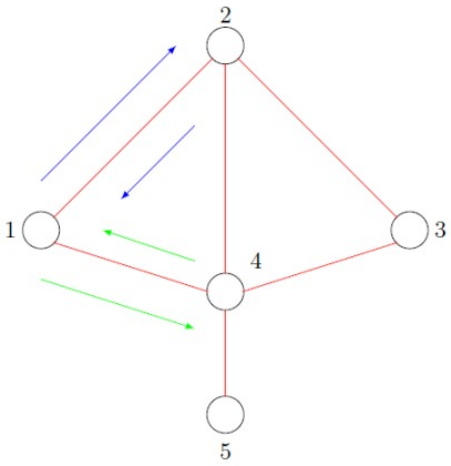

Figure 2.6: Graph example. In blue and green are highlighted the path from node 1 to 2 of length 2.

Consider the adjacency matrix A

$$A = \begin{bmatrix} 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix} \quad \rightarrow \quad A^2 = \begin{bmatrix} 2 & 1 & 2 & 1 & 1 \\ 1 & 3 & 1 & 2 & 1 \\ 2 & 1 & 2 & 1 & 1 \\ 1 & 2 & 1 & 4 & 0 \\ 1 & 1 & 1 & 0 & 1 \end{bmatrix} \ldots \rightarrow A^4 > 0$$

Note that, in matrix A^2 , the entry (i,j) gives the number of paths of length 2 in the graph G, from the node i to the node j.
For example $A(1,1) = 2$ means that there are 2 different paths that go from node 1 to node 1 in two steps (blue and green arrows in Fig. 1.2) and $A(4,2) = 2$ means that there are 2 different paths that go from node 4 to node 2 in two steps.
By iterating, A^3 has as entry (i,j) the number of paths of length 3 that go from node i to node j and, in general, A^k has as entry (i,j) the number of paths of length k that go from node i to node j.
Computing the subsequent powers of matrix A, we find out that the first value of k for which A^k is full (i.e. has all its entries different from zero) is $k = 4$.
This result implies that in 4 steps we can reach any other node from an ar-

bitrary node of the graph G. Note that, since $A^4 > 0$, then, by definition, the matrix A_G is primitive. If the matrix is full, the graph will spread information on all the nodes, thus every node in the graph is reached starting from any node.
This concept can be also applied to the P matrix. So, we can state that if P is primitive then there exists an integer number k such that A^k is full.

We also have the following properties:

- Primitive A: any node is connected to any other by a k-length path (connected/strongly connected).
- Irreducible A: any node is connected to another by some k-length path (connected/strongly connected).
- If starting from an irreducible A we can find a place where to stand and wait, we can equalize all paths.

The properties just presented allow the formalization of the following theorems:

Theorem 6 *G is strongly connected if and only if A is irreducible*

Theorem 7 *A is irreducible and there is a self-loop in the graph G described by $A \Rightarrow$ A is primitive*

3

Model design

In this chapter the model that will be exploited is presented. In particular, since our aim is to control the height of the water in the network, we will consider as state of the height of canals.
Differently from the standard version of the consensus protocol, we will consider constrained capacity of state variation during an iteration of the algorithm. This choice is made in order to represent what could be considered as a limited capacity actuator in a real scenario. Imagine we want to move an amount of water from canal A to canal B, we are going to compute the information needed to set the power of the actuator output. For this reason, we need to know and manage the maximum capacity of the actuator we are exploiting in order to tune the speed of the system in relation with the iterations of the algorithm.
The following sections describe how we manage to find a feasible solution to this problem.

3.1 Water allocation via distributed consensus

From [21], a distributed model can be constructed by introducing some memory in the system and adopting the model (2.2) to provide only a weighted correction to current estimate, thus leading to

$$\begin{aligned} x(k+1) &= \eta(k)x(k) + (1-\eta(k))Px(k) \\ &= (\eta(k)I_n + (1-\eta(k))P)x(k) \\ &= P_{\eta(k)}(k)x(k) \end{aligned} \tag{3.1}$$

where $\eta(k) \in (0,1)$ and I_n is the identity matrix of dimension n. The matrix $P_{\eta(k)} \in \mathbb{R}^{n\times n}$ is still row-stochastic but with eigenvalues in the range $(-1+2\eta(k),1]$. In this kind of model the i^{th} component of $x(k)$ represents the height of the water, expressed in meters, of the i^{th} canal. In this model the following mild assumption is adopted.

Assumption 1 (Basic assumptions on the system)

1. $\min_i x_i(0) > 0$
2. $\forall i,j \in E$: $p_{ij} \in (0,1)$

Assumption 1 states that two different facts needs to hold. The first means that the system is not empty. The second is a weak assumption on the weights of matrix P to ensure certain convergence properties (detailed in Section 3.1.1).
We can observe that, exploiting the linearity of the spectrum, it holds that $\lambda_i^{P_{\eta(k)}} = \eta(k) + (1-\eta(k))\lambda_i^P, i = 0 \dots n-1$.
The presence of self-loops in the model, controlled by parameter $\eta(k)$, allows to modify the eigenvalues domain from the unit circle to the set $\Upsilon \cup 1$ where Υ is a circle centered in $(\eta(k),0)$ with radius $1-\eta(k) < 1$, ruling out the possible presence of the critical eigenvalue $\lambda = -1$. More interestingly, the $\eta(k)$ parameter can be tuned to control the convergence

speed, governed by the second largest (in modulus) eigenvalue of $P_{\eta(k)}$. Remarkably, this behavior can be seen as dependent on the control parameter $\eta(k)$. A good and viable strategy is to select the parameter $\eta(k)$ as

$$\eta(k)^* = \arg\min_{\eta(k)\in[0,1)}\{\max_{i=1\dots n-1}|\lambda_i^{P_{\eta(k)}}|\}$$

to minimize the convergence rate.

3.1.1 Design of the consensus matrix

In general, given a consensus problem we want to design P such that:

- P is adapted to the graph $G = (V, E)$ (same support as A);
- the eigenvalues of P (except $\lambda_0^P = 1$) are located as close as possible to 0. By doing so, we are locating also the second eigenvalue λ_1^P, which determines the speed of convergence.
- we want to reach average consensus

We can draw a feasible solution for P matrix exploiting the Metropolis-weights-based-design that defines its coefficients as follows:

$$p_{ij} = \begin{cases} \frac{1}{1+\max\{d_i,d_j\}} & i,j \in E \\ 1 - \sum_j p_{ij} & i = j \\ 0 & otherwise \end{cases} \tag{3.2}$$

This design provides a doubly stochastic matrix considering second-order information as the degrees of the neighboring nodes. Notice that this kind of selection for matrix P ensures that the second point of Assumption 1 is satisfied, i.e. $\forall i,j \in E$: $p_{ij} \in (0,1)$.
From [27], we know that this kind of design ensures convergence on average unless the underlying network is a regular bipartite graph.
For this particular case, computations will be shown in the next section in order to ensure convergence. We can observe that if the graph G representing the given system is connected, P has n real eigenvalues $\lambda_0^P \geq \lambda_1^P \dots \geq$

λ_{n-1}^P in the range $[-1, 1]$.
The value of η^* can be analytically computed as shown in the next propositions that are referring to the two different choices of P previously explained.

Proposition 2 (Metropolis weights based design) *Given a multi-agent network represented by graph G, the optimal value η^* is univocally determined as*

$$\eta^* = \begin{cases} \frac{\varsigma_{\mathcal{M}}}{\varsigma_{\mathcal{M}}-1} & \varsigma_{\mathcal{M}} < 0 \\ 0 & \textit{otherwise} \end{cases} \tag{3.3}$$

where $\varsigma_{\mathcal{M}} := \frac{1}{2}(\lambda_1^P + \lambda_{n-1}^P) \in \mathbb{R}$ and P is the matrix, related to G, computed as in Equation (3.2).

Note that the proposition just presented holds when applied to constant topologies. If applied to time-varying models additional computation is necessary in order to ensure stability and convergence.

3.1.2 Regular bipartite graphs: convergence

In Section 3.1.1 it is mentioned that the convergence is not ensured in case of regular bipartite graph topology. The following reasoning is aimed to show that the distributed spectral based solution proposed allows convergence also in case of this particular topology.
Regular bipartite graphs refers to a topology that does not ensure convergence because, differently from the others, the consensus matrix P induced by these networks has at least two eigenvalues lying on the boundary of the unit circle. Under the hypotesis of designing the matrix P through the Metropolis weights, we have that P is symmetric and for this reason it has only real eigenvalues $\lambda_i^P \in [-1, 1]$. This fact tells us that, not only the eigenvalues belong to the unit circle, but, in particular, they all lie on the real axis. Furthermore, exploiting the distributed spectral based

solution and the linearity of the spectrum we have that:

$$\lambda_i^{P_{\eta(k)}} = \eta(k) + (1 - \eta(k))\lambda_i^P, i = 0 \dots n - 1. \tag{3.4}$$

This is a convex combination that, by definition, gives us as result an absolute value for each one of the eigenvalues of $P_\eta(k)$ smaller than the corresponding of matrix P. In particular, we obtain $|\lambda_i^{P_\eta(k)}| < 1$ for all $i = 1...n - 1$, thus ensuring convergence for this particular kind of topology.

3.2 Handling of the network constraints

In this section, it is presented the computation made in order to introduce constraints on the consensus model just obtained in Equation (3.1). We want our system to recognize non-linear limits in information exchange capacity. $P_\eta(k)$ is in the form:

$$P_\eta(k) = \begin{bmatrix} \eta(k) + (1 - \eta(k))P_{11} & (1 - \eta(k))P_{12} & * & (1 - \eta(k))P_{1n} \\ (1 - \eta(k))P_{21} & \eta(k) + (1 - \eta(k))P_{22} & * & (1 - \eta(k))P_{2n} \\ * & * & * & * \\ (1 - \eta(k))P_{n1} & (1 - \eta(k))P_{n2} & * & \eta(k) + (1 - \eta(k))P_{nn} \end{bmatrix}.$$

This fomulation allows us to exploit the consensus protocol and the parameter $\eta(k)$ in order to obtain consensus on the average value. What the model (3.1) does not take into account is the limits of information exchange between two connected nodes during every single iteration of the algorithm. However, for this kind of algorithm, the exchanged information may exceed the maximum flow capacity w.r.t. a single connection. This issue is due to the fact that the algorithm is conceived to manage and, possibly, to share all the state values during every iteration. This is the reason why we need to introduce nonlinear constraints that are necessary to take care of the water channel network under analysis. Such a strategy is implemented through the network by exploiting the presence of the tuning parameter $\eta(k)$.

In reality, the network links cannot physically exchange more than a predefined amount of water; thus, the following condition will be considered as a violation for the proposed water distribution protocol:

$$||x(k) - x(k+1)||_\infty > c_{max}(k) \tag{3.5}$$

for some $k \in \mathbb{N}$ and where $c_{max} : \mathbb{N} \to \mathbb{R}_+ : k \mapsto c_{max}(k)$ such that $c_{max}(k) > 0$ for all $k = 0, 1, 2, \ldots$ is a piecewise constant function that represents the maximum exchange capacity among all canals at each iteration. We do not consider the case where $c_{max}(k) = 0$ which would describe a static system, incapable of exchanging information between its nodes. Consequently, we impose the following set of complementary constraints:

1. when in download condition, meaning the value of the state is decreasing in the next iteration, i.e. $x_i(k) - x_i(k+1) > 0$,

$$x_i(k) - x_i(k+1) \leq c_{max}(k) \quad \text{download constraint} \tag{3.6}$$

2. when in upload condition, meaning the value of the state is increasing in the next iteration $x_i(k+1) - x_i(k) > 0$,

$$x_i(k+1) - x_i(k) \leq c_{max}(k) \quad \text{upload constraint} \tag{3.7}$$

3. when in equilibrium, meaning that there is no difference in consecutive iterations, we simply let:

$$x_i(k+1) - x_i(k) = 0 \quad \text{equilibrium regime} \tag{3.8}$$

The download constraint regulates the capacity of a node to exchange information from its state to its neighbors. Constraint (3.6) holds for all time instants k in which the state value at the next iteration $x_i(k+1)$ decreases w.r.t. $x_i(k)$. On the other hand, the upload constraint does the opposite, so the capacity to receive water of a node from its neighbourhood, for this reason constraint (3.7) holds for all time instants k in which the state value at the next iteration $x_i(k)$ decreases w.r.t. $x_i(k+1)$.
For the sake of completeness, it is a must to specify the case relative to

the equilibrium regime, i.e. $x_i(k+1) = x_i(k)$. The fact that this equality holds means that state i is not changing from the current iteration to the next one. This fact is true if and only if the protocol has reached an equilibrium point. This said, it is fundamental to remind that the consensus protocol has a singular equilibrium point that is reached as the number of iterations grows: this means that the equality holds if and only if consensus is reached.

3.2.1 Derivation of parameter $\eta(k)$

In deriving parameter $\eta(k)$ it is important to remind that the model deals with three different regimes. In particular, download and upload regimes are meaningful for our purpose of derive this parameter. On the other hand, equilibrium regime is reached only for $k \to \infty$, so the derivation in this particular case will not be treated.
Download regime: we start the derivation of parameter $\eta(k)$ substituting $x_i(k+1)$ as in Equation (3.1) into equation (3.6) obtaining:

$$x_i(k) - \eta(k)x_i(k) - \sum_{j=1}^{n} p_{ij}x_j(k) + \eta(k)\sum_{j=1}^{n} p_{ij}x_j(k) \leq c_{max}(k). \tag{3.9}$$

$$\eta(k)(x_i(k) - \sum_{j=1}^{n} p_{ij}x_j(k)) \geq x_i(k) - \sum_{j=1}^{n} p_{ij}x_j(k) - c_{max}(k). \tag{3.10}$$

Now, we want to isolate $\eta(k)$ in order to obtain the minimum value to satisfy the inequality. To do this, some considerations follow.
In order to divide by the factor $x_i(k) - \sum_{j=1}^{n} p_{ij}x_j(k)$ we need to make sure this value is positive. Assuming that condition in (3.6) holds, one has the following proposition:

Proposition 3 $x_i(k) - \sum_{j=1}^{n} p_{ij}x_j(k) > 0$ *when in download case.*

Proof 1 *Recalling the download case:*

$$x_i(k) > x_i(k+1) \tag{3.11}$$

where

$$x_i(k+1) = \eta(k)x_i(k) + (1-\eta(k))\sum_{j=1}^{n} p_{ij}x_j(k) \tag{3.12}$$

and substituting (3.12) *in* (3.9) *we obtain*

$$x_i(k) > \eta(k)x_i(k) + (1-\eta(k))\sum_{j=1}^{n} p_{ij}x_j(k) \tag{3.13}$$

$$(1-\eta(k))x_i(k) > (1-\eta(k))\sum_{j=1}^{n} p_{ij}x_j(k). \tag{3.14}$$

Recalling that $\eta(k) \in (0,1) \Rightarrow (1-\eta(k)) > 0$*, we get*

$$x_i(k) > \sum_{j=1}^{n} p_{ij}x_j(k) \tag{3.15}$$

$$x_i(k) - \sum_{j=1}^{n} p_{ij}x_j(k) > 0. \quad \square \tag{3.16}$$

Now we can isolate $\eta(k)$ in (3.10) obtaining

$$\eta(k) > \frac{x_i(k) - \sum_{j=1}^{n} p_{ij}x_j(k) - c_{max}(k)}{x_i(k) - \sum_{j=1}^{n} p_{ij}x_j(k)} = 1 - \frac{c_{max}(k)}{x_i(k) - \sum_{j=1}^{n} p_{ij}x_j(k)} \tag{3.17}$$

from which it follows that $\eta(k)$ can be chosen through

$$\eta(k) > 1 - \frac{c_{max}(k)}{||x(k) - Px(k)||_\infty}. \tag{3.18}$$

In an effort to pave the way for the calculation of next section, we set $\eta(k) = \eta_{download}(k)$ where $\eta_{download}(k)$ is such that

$$\eta_{download}(k) \geq 1 - \frac{c_{max}(k)}{||I_n - P||_\infty ||x(k)||_\infty} \tag{3.19}$$

where we exploit the fact that

$$||I_n - P||_\infty ||x(k)||_\infty \geq ||x(k) - Px(k)||_\infty \tag{3.20}$$

The parameter $\eta_{download}(k)$ just derived ensures the node not to exceed the download maximum capacity, since it satisfies (3.18).
Upload regime: the derivation procedure of the upload parameter is similar. By substituting as in the previous case, we obtain

$$\eta(k)x_i(k) + \sum_{j=1}^{n} p_{ij}x_j(k) - \eta(k)\sum_{j=1}^{n} p_{ij}x_j(k) - x_i(k) < c_{max}(k), \tag{3.21}$$

isolating $\eta(k)$ we get

$$\eta(k) > \frac{\sum_{j=1}^{n} p_{ij}x_j(k) - x_i(k) - c_{max}(k)}{\sum_{j=1}^{n} p_{ij}x_j(k) - x_i(k)} = 1 - \frac{c_{max}(k)}{\sum_{j=1}^{n} p_{ij}x_j(k) - x_i(k)} \tag{3.22}$$

where, by simmetry with the download case, we exploited the fact that in the upload case it is true that $\sum_{j=1}^{n} p_{ij}x_j(k) > x_i(k)$.
We apply again the infinity norm to obtain

$$\eta(k) > 1 - \frac{c_{max}(k)}{||Px(k) - x(k)||_\infty}. \tag{3.23}$$

Finally, it is set $\eta(k) = \eta_{upload}(k)$ where $\eta_{upload}(k)$ is such that

$$\eta_{upload}(k) \geq 1 - \frac{c_{max}(k)}{||P - I_n||_\infty ||x(k)||_\infty}. \tag{3.24}$$

The parameter $\eta_{upload}(k)$ just derived ensure the system not to exceed the upload maximum capacity, since it satisfies (3.23).

3.2.2 Tuning of parameter $\eta(k)$

Assuming to deal with a general multi-agent network we may not have access to global system information. This could be due to uncertainties, in general or, more in particular, noises affecting the system. Clearly, this fact leads to computational problems w.r.t. the parameter $\eta(k)$. In this case, we can manage this kind of issue by designing a decentralized calculation that generalizes formulation in Subsection 3.2.1. We now have $\eta(k)$ as in inequality (3.19) that requires all the information about P to solve $||I_n - P||_\infty$. The value of $||I_n - P||_\infty$ is obtained with the computation presented in the following Lemma.

Lemma 8 *Let $\omega := 2d_M/(1+d_M)$. Then, for a matrix P raw-stochastic, it holds that $||I_n - P||_\infty \leq \omega$.*

Proof 2 *One has*

$$||I_n - P||_\infty \leq \max_i\{|1 - p_{ii}| + \sum_{j\neq i}|p_{ij}|\} = \max_i\{1 - p_{ii} + \sum_{j\neq i} -p_{ij}\} \tag{3.25}$$

$$= \max_i\{1 - 2p_{ii} + 1\} = \max_i\{2 - 2p_{ii}\} \tag{3.26}$$

$$= 2\max_i\{1 - p_{ii}\} = 2(1 - \min_i\{p_{ii}\}) = \omega, \tag{3.27}$$

since $\min_{i\in V}\{p_{ii}\} = \dfrac{1}{1+d_M}$, *as shown in Equation* (A.4). □

Lemma 8 applied on (3.19) leads to a generalized and decentralized form of $\eta(k)$

$$\eta_{download}(k) \geq 1 - \frac{c_{max}(k)}{\omega||x(k)||_\infty}. \tag{3.28}$$

Noticing that, in general, the value of $1 - \frac{c_{max}(k)}{\omega||x(k)||_\infty}$ might be negative, we define:

$$\eta_L = \begin{cases} \eta^* & \text{if } \eta^* > 0 \\ \zeta & \text{otherwise} \end{cases} \tag{3.29}$$

where η^* is chosen as in (3.3) and $\zeta \in (0,1)$ is an arbitrarily small constant. Exploiting this new parameter we can now define:

$$\eta^*_{download}(k) = \max\left\{1 - \frac{c_{max}(k)}{\omega||x(k)||_\infty}, \eta_L\right\} \tag{3.30}$$

By simmetry we get the same result for $\eta^*_{upload}(k)$ if constant $c_{max}(k)$ is employed for both download and upload conditions, i.e. $\eta^*_{upload}(k) = \max\{1 - \frac{c_{max}(k)}{\omega||x(k)||_\infty}, \eta_L\}$.
In this particular case we can define:

$$\eta^*_{constrained}(k) := \eta^*_{download}(k) = \eta^*_{updload}(k) = \max\left\{1 - \frac{c_{max}(k)}{\omega||x(k)||_\infty}, \eta_L\right\} \tag{3.31}$$

On the other hand, more in general, we can define the two parameters as

$$\eta^*_{constrained,download}(k) = \max\left\{1 - \frac{c_{max,download}(k)}{\omega||x(k)||_\infty}, \eta_L\right\} \tag{3.32}$$

$$\eta^*_{constrained,upload}(k) = \max\left\{1 - \frac{c_{max,upload}(k)}{\omega||x(k)||_\infty}, \eta_L\right\} \tag{3.33}$$

where we highlight the different values for $c_{max,download}(k) > 0$ and $c_{max,upload}(k) > 0$ can be chosen with the same definition as $c_{max}(k)$.
This differentiation allows us to have a more versatile model able to recognize differences in capacity, respectively, in download or upload case.
Finally it is worth to note that, since the chosen model is structured as a convex combination of the previous state, we obtain the following result:

$$\max_k\{||x(k)||_\infty\} = ||x(0)||_\infty \tag{3.34}$$

From the computation of $\eta^*_{constrained}(k)$ just given and Equation (3.31), we obtain a more precise domain for this parameter that results to be as shown in the following:

Proposition 4 *One has*

$$\eta^*_{constrained}(k) \in [\eta_L, \eta_U(k)] \subseteq (0,1), \tag{3.35}$$

where $\eta_U(k) = \max\left\{\eta_L, 1 - \frac{\min\{c_{max,download}(k), c_{max,upload}(k)\}}{\omega||x(0)||_\infty}\right\}$

Proof 3 *From the definition of* $\eta^*_{constrained}(k)$ *given in Equation* (3.31) *we can deduce that:*

$$\eta^*_{constrained}(k) \geq \eta_L > 0 \tag{3.36}$$

Regarding the upper-bound, Equation (3.34) *implies*

$$\eta^*_{constrained}(k) \leq 1 - \frac{\min\left\{c_{max,download}(k), c_{max,upload}(k)\right\}}{\omega||x(0)||_\infty} < 1 \quad \forall k = 1,2,3\ldots \square \tag{3.37}$$

This is a direct consequence of the domain of $c_{max}(k)$ being chosen to represent a physical limit, meaning that this parameter is strictly positive. Lastly, the only information needed to obtain $\eta^*_{constrained}(k)$ is the value of $||x(k)||_\infty$. The general solution to get the value of $||x(k)||_\infty$ is presented in Appendix A.2.

4 Water distribution algorithm

In this chapter, we present the main contribution of this thesis, namely a time-varying consensus-based water distribution algorithm, and illustrate an application example.

4.1 Premise

This chapter is aimed to give the rationale that leads to the algorithm we implement and solve the water distribution problem. The aim of this approach is to evenly distribute and manage an amount of water in case of uneven distribution in the given water channel system. Even distribution among states could be measured and described in many ways, one of which is by applying the following technique.
Define the max-min disagreement function: $V_{max-min} : \mathbb{R}^n \to \mathbb{R}_{\geq 0}$ by

$$V_{max-min}(x(k)) = \underbrace{\max x_i(k)}_{i=1\ldots n} - \underbrace{\min x_i(k)}_{i=1\ldots n} = \underbrace{\max x_i(k) - x_j(k)}_{i,j=1\ldots n}. \tag{4.1}$$

$V_{max-min}(x(k)) \geq 0$ and $V_{max-min}(x(k)) = 0$ if and only if $x(k)$ is a consensus vector. We will impose $V_{max-min}(x(k))$ to decrease beneath an ar-

bitrary small threshold $\gamma > 0$. Clearly our aim is to reach an agreement on the mean of the network states in order to ensure even water distribution. This minimization will be put in place by the execution of the algorithm and will be stopped when the threshold is reached. Such an approach can be generalized and applied for every graph-like system that has even distribution of its states as main objective. The starting point of this method is to collect information about the system topology that we want to control.
This thesis deals with a water network, so what we need are descriptions for the network of interest. In particular, we want to run the proposed algorithm by first collecting the following data:

- $x(0)$ or initial state: this vector describes the initial state of the system. In our case, we want to know a measure of water filling about the network canals. For instance, we could have $x_i(k) = 10$ meaning that the i^{th} canal at iteration k has height of 10 m. Each entry of this vector is paired with the coordinates of its location;

- canal junctions (CJ) or location coordinates: CJ is a vector containing couples of coordinates describing the starting and ending points of each canal. This feature allows us to associate the measures taken with the relative position in the network. Moreover, the position vector also describes the connections among the canals, meaning that whenever the ending point of a canal matches with the starting point of another one we identify a connection. This information is fundamental to generate a graph by coupling the initial states and their position;

- $c_{max}(k)$ or maximum capacity: this information gives us the upper bound limit in exchange capacity among nodes. This is an intrinsic feature of the given system. Its unit of measurement reflects what it is limiting, so for us it is a scalar value expressed in meter/iteration $[\frac{m}{s}]$, since what we are limiting is the height variation from an iteration to the next one. We highlight the fact that, for the sake of simplicity, we adopt; $c_{max}(k) = c_{max,download}(k) = c_{max,upload}(k)$

- γ or agreement threshold: this feature represents the threshold that

has to be reached in order to satisfy the given specification about water distribution. This value will be used as termination condition in the proposed algorithm.

4.2 Description of the main algorithm

Having these four requirements, we illustrate and analyze the proposed algorithm in this section, as shown in Algorithm 4.1.

Algorithm 4.1 Adapted consensus for water distribution

Require: connectivity of G
Ensure: consensus on average
Input: CJ, $x(0)$, $c_{max}(k)$, γ
Output: $x(k+1)$
1: compute graph F induced by CJ and $x(0)$
2: compute G, i.e. the adjoint graph of F
3: compute η_L as in (3.29)
4: compute φ_G
5: $k \leftarrow 0$
6: **while** $V(x(k)) > \gamma$
7: $x_{max}(k) \leftarrow Maxconsensus(G, x(k), \varphi_G)$
8: compute η as in (3.31)
9: $x(k+1) \leftarrow P_\eta x(k)$
10: $k \leftarrow k+1$
11: **end while**

As visible on the top of the pseudocode, we state an important requirement on our system. We require the system to be connected so to be able to apply the consensus protocol properly for our aim.
Given this requirement coupled with Corollary 1, our protocol ensures the system to achieve consensus on average on its final states.

The first six lines of the pseudocode describe the initialization of the given input. Pairing the initial states $x(0)$ and their position data, it is computed the given graph F, where F is a graph that represents the canals as edges with the relative water height associated. In order to apply the consensus protocol, a representation for the dual version of the given topology is indeed required. Since the goal is balancing the water levels within the network, we associate each canal height to the state component x_i and treat each node i associated with x_i as a vertex in the adjoint graph G whose incident edges are determined by the canal junctions in F. For this reason, what we do is to compute G, the adjoint graph of F.
At line 3 we compute the optimal value η_L, so to ensure the maximum rate of convergence in a static scenario, i.e. with constant η.
The last step of the initialization is to compute the diameter of graph G and set the counter variable k.
The algorithm is then characterized by a main while cycle whose termination condition is the mentioned agreement threshold γ, that is the variable representing the value to be reached in order to determine when we are satisfied with the states heights.
It is worth to observe that $V(x(t))$ can be computed in a distributed fashion leveraging the max-consensus protocols. After computing x_{max} exploiting the Max-consensus algorithm (see Algorithm A.1 presented in Appendix A.2), we compute $\eta^*_{constrained}(k)$. The algorithm then attempts to update the model with the optimal* value η_L and by checking if the limits are exceeded. If not, the execution proceeds by updating the states; otherwise, the ensemble state is updated by exploiting P_η. At each iteration, the updated state $x(k+1)$ can be returned. This is the information needed in order to set the actuators in a real scenario.
The following section provides a numerical example to help the understanding of this algorithm.

*from the point of view of convergence rate in a static scenario

4.3 Execution example on a regular bipartite graph

Here, we present a simple example of how the algorithm works. Table 4.1 reports the data of the state, starting and ending connections of a 4-canal network representing a regular bipartite graph. The choice of this topology is made in order to show the correctness of the protocol also in this particular case (see Section 3.1.2)

	STATE (m)	START	END
1	10	1	2
2	4	1	4
3	3	2	3
4	4	3	4

Table 4.1: Initial states of the given 4-canal network

Table 4.1 shows a simple example of a 4-canal network where initial states were randomly chosen. The aim of this example is to check whether the algorithm converges to the average of the initial states that is $5.25m$. The value of $c_{max}(k)$ employed in this example is constant and set at $c_{max}(k) = 1m/s$ and the threshold used as ending condition is $\gamma = 10^{-4}$. From this data we can draw graph F as in Figure 4.1.

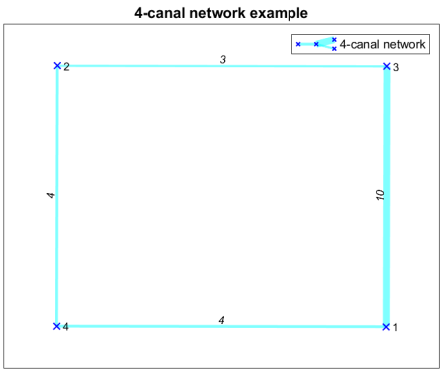

Figure 4.1: 4-canal graph F from Table 4.1 data (initial state)

Figure 4.1 shows how the chosen 4-canal network looks like. We can appreciate that the thickness of each edge reflects the height of the water contained in the canal that is also specified just above each edge. Next is the representation of G, the adjoint graph of F.

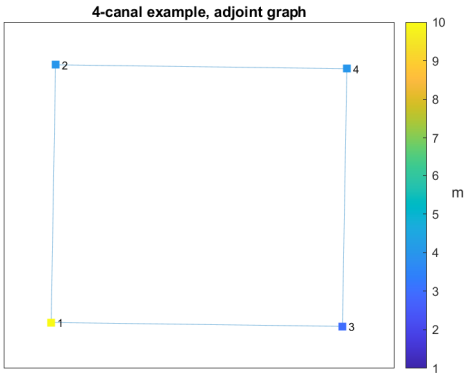

Figure 4.2: 4-canal adjoint graph G from Table 4.1 data

Figure 4.2 has nodes representing canals and highlights the height of water present in each canal with different color. With this configuration we can apply the consensus protocol. Within the while cycle, the first instruction invokes max-Consensus algorithm (see Algorithm A.1 presented in Appendix A.2) to find the variable $x_{max}(k)$, representing the current maximum state value. The computation of $x_{max}(k)$ is necessary to get the value of $\eta^*_{constrained}(k)$ that is changing throughout the iterations. After this computation, there is an attempt to update the system with η_L by checking whether the download or upload constraints are not exceeded: if not, the system is updated through η_L; otherwise, it is updated throught $\eta^*_{constrained}(k)$, so to ensure suitable exchange capacity for the network.
We highlight that the constraint check in Algorithm 4.1 is an ad hoc verification for the particular case in which it is set $c_{max} = c_{max,download} = c_{max,upload}$. If different values for $c_{max,download}$ and $c_{max,upload}$ are employed, it is necessary to inspect which one of the two constraints is exceeded and the possibilities are the following:

- if both limits, shown in Equations (3.6) and (3.7), are exceeded we

need to select

$$\eta^*_{constrained}(k) = \max\{\eta^*_{constrained,download}(k), \eta^*_{constrained,upload}(k)\} \quad (4.2)$$

- if only one of the constraints, shown in Equations (3.6) and (3.7), is exceeded we select the corresponding constraint
- if none of them is exceeded η_L is selected.

The state trajectories computed through the iterations of the proposed example are depicted in Figure 4.3.

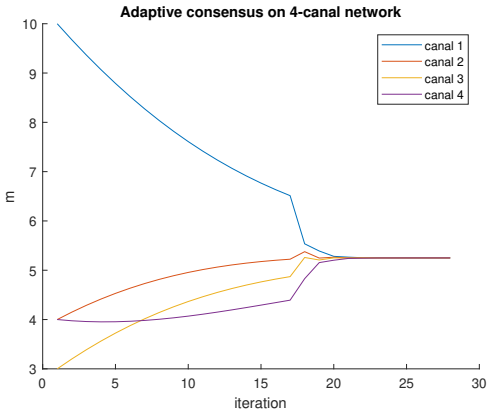

Figure 4.3: Adaptive consensus algorithm 4.1 applied on the proposed example.

Figure 4.3 shows the variation of the water height for each canal until the agreement threshold γ is reached so that the algorithm stops. It is also possible to notice a similar pattern for all the state trajectories; in particular, the trajectories are seemingly split into two parts: the first part, characterized with a slower rate of convergence, is where $\eta^*_{constrained}(k)$ is employed and the second part, that begins on the 17^{th} iteration until the end, characterized with a faster rate of convergence, is where η_L is employed.
In Figure 4.4, it is worthy to notice how the constraint (3.6) and (3.7) are satisfied for all the iteration of the algorithm, as the value of $||x(k+1) - x(k)||_\infty$ is always less than $c_{max}(k)$. This is a simulation result that supports the correctness of the algorithm even in case of regular and bipartite topology.

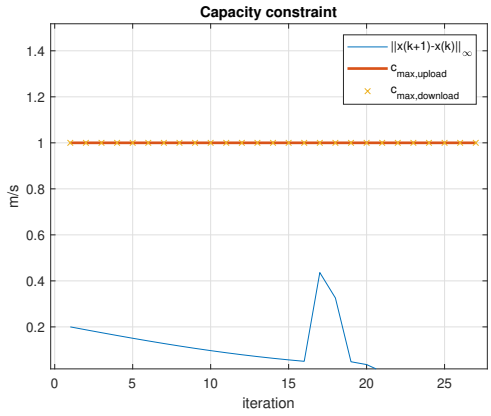

Figure 4.4: Graphical comparison between $c_{max}(k)$ and $||x(k+1) - x(k)||_\infty$.

The following two figures, Figures 4.6 and 4.7, depict the fulfillment of the consensus as the termination of the algorithm occurs. Figure 4.6 is a representation of the adjoint graph G that has reached the requested agreement threshold γ; whereas, Figure 4.7 shows the starting graph F with the final values written on edges. Note that, the final values are $x_i(k)$ with $k = 28$, having a maximum dispersion $V_{max-min}(x(28)) = (5.250006 - 5.249992) = 0.000014m$ around the average value of the initial state. The comparison between $V_{max-min}(x(k))$ and γ is reported in Figure 4.5.

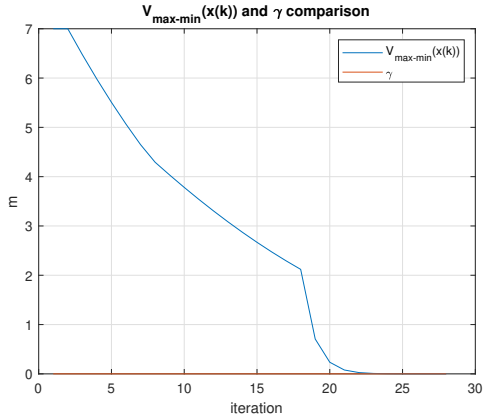

Figure 4.5: comparison between $V_{max-min}(x(k))$ and γ

The nodes color in Figure 4.6 and the edges thickness in Figure 4.7 are almost equally depicted for this exact reason.

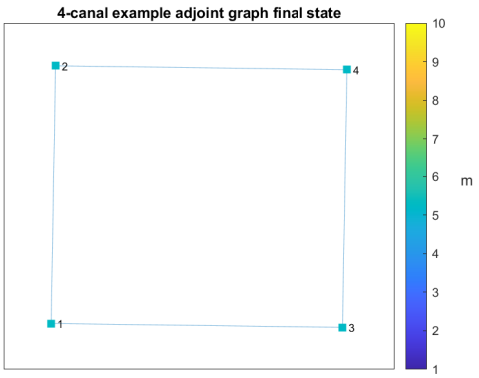

Figure 4.6: Adjoint graph G result from Table 4.1

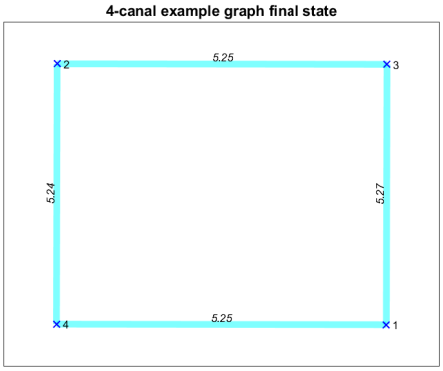

Figure 4.7: Table 4.1 graph F result

In addition, the final states are reported in Table 4.2.

	STATE (m)	START	END
1	5.25	1	2
2	5.25	1	4
3	5.25	2	3
4	5.25	3	4

Table 4.2: Final states of the given 4-canal network

From Table 4.2 we can appreciate that the final state values, rounded up to the second decimal, are all very close to the average.

5 Convergence analysis

In this section, a convergence analysis of the algorithm proposed is presented. All the theretical tools used in this chapter refer to [28]. In particular, the convergence is proven by means of Lyapunov-based theorems and a formulation of a convergence rate is then proposed.

5.1 Max-min disagreement

In this chapter, the max-min disagreement function of Equation (4.1) is recalled and its convergence proved by Lemma 9. This function will be then exploited in Theorem 10 in order to propose a convergence rate metric for system (3.1).

Definition 25 (1-coefficient of ergodicity) *Given a row-stochastic matrix P, let us define the 1-coefficient of ergodicity of P by one of the following*

equal expressions:

$$\min_{i,j=1\dots n} \tau_1(P) = \underbrace{\max ||P^T y||_1}_{||y||_1=1, y\perp 1_n}$$
$$= \frac{1}{2} \max_{i,j=1\dots n} \sum_{h=1}^{n} |p_{ih} - p_{jh}| = 1 - \min_{i,j=1\dots n} \sum_{h=1}^{n} \min\{p_{ih}, p_{jh}\}.$$

The coefficient just defined allows us to express Lemma 9 that ensures the convergence of the max-min disagreement function.

Lemma 9 (Convergence of max-min disagreement) *Given a row-stochastic primitive matrix P with associated digraph G,*

1. *for all $x \in \mathbb{R}^n$, the max-min function satisfies*

$$V_{max-min}(Px) \leq \tau_1(P)\, V_{max-min}(x); \tag{5.1}$$

2. *$\tau_1(P) < 1$ if and only if P is scrambling, i.e., any two nodes have a common out-neighbor in G;*

3. *if G contains a node that is globally reachable in h steps, then P^h is scrambling and any solution to $x(k+1) = Px(k)$ satisfies*

$$V_{max-min}(x(k)) \leq \underbrace{\tau_1(P^h)^{\lfloor k/h \rfloor}}_{<1} V_{max-min}(x(0)) \quad \textit{for all } k \in \mathbb{N} \tag{5.2}$$

Note 1: under the conditions stated in the theorem, the max-min disagreement diminishes monotonically along each solution and we say that the function $V_{max-min}$ is a Lyapunov function for $x(k+1) = Px(k)$.
Note 2: G contains a node that is globally reachable in h steps, for some h (i.e., there exists h such that from each node there exists a directed path of length h to the specific node) if and only if G contains a globally reachable node and the strongly connected component of globally reachable nodes is aperiodic.
Note 3: if P is scrambling and each of its non-zero entries is lower bounded

by $p_{min} > 0$, then statement 2 can be strenghtened to state that $\tau_1(P) \leq 1 - p_{min}$.

Proof 4 (Proof of Lemma 9) *Statement 1 is trivial if* $\max_{j\in 1\dots n} \min_{j\in 1\dots n} p_{ij} = 0$ *e.g. . Hence let us prove the statement when* $\max_{j\in 1\dots n} \min_{j\in 1\dots n} p_{ij} > 0$. *We compute*

$$\begin{aligned} V_{max-min}(Px) &= \max_i \sum_{m=1}^{n} p_{im}x_m - \min_i \sum_{m=1}^{n} p_{im}x_m \\ &= \max_i\Big(\sum_{m=1,m\neq j}^{n} p_{im}x_m + a_{ij}x_j\Big) - \min_i\Big(\sum_{m=1,m\neq j}^{n} p_{im}x_m + p_{ij}x_j\Big) \\ &\leq \max_i\Big(\sum_{m=1,m\neq j}^{n} p_{im}x_{max} + p_{ij}x_j\Big) - \min_i\Big(\sum_{m=1,m\neq j}^{n} p_{im}x_{min} + p_{ij}x_j\Big), \end{aligned}$$

where, after using the bounds $x_{min} \leq x_m \leq x_{max}$, *we minimize the right hand side as a function of j. From the latter equation we obtain*

$$V_{max-min}(Px) = \min_j \max_i((1-p_{ij})x_{max} + p_{ij}x_j) - \min_j \max_i((1-p_{ij})x_{min} + p_{ij}x_j) \tag{5.3}$$

and noting that $\min_j \max_i (1-p_{ij}) = 1 - \max_i \min_j p_{ij} = 1 - p_{ij}$,

$$\begin{aligned} V_{max-min}(Px) &= ((1-p_{ij})x_{max} + p_{ij}x_j) - ((1-p_{ij})x_{min} + p_{ij}x_j) \\ &= (1-p_{ij})(x_{max} - x_{min}) = (1-p_{ij})V_{max-min}(x) \end{aligned}$$

Statement 2 is an immediate consequence of the definition of $\tau(P)$. *Indeed if each column j as an entry equal to* 0 *the the quantity* $b_j = \min p_{ij} = 0$ *and, in turn* $\tau(P) = \max_j b_j = 0$. *Regarding statement 3, if the* j^{th} *node in G that is globally reachable in h steps, then the* j^{th} *column of* P^h *is strictly positive. Therefore* P^h *is scrambling, since each node has the node j as common out-neighbor in the digraph associated to* P^h. *The final bound follows from the previous statements*

5.2 Convergence over time-varying digraphs connected over time

The theorems presented in this chapter allow us to ensure convergence for our model (3.1) and paves the way for the definition of a convergence metric.

Theorem 10 (Consensus for time-varying algorithms) *Let* $\{P(k)\}_{k\in\mathbb{Z}\geq 0}$ *be a sequence of row-stochastic matrices with associated digraphs* $\{G(k)\}_{k\in\mathbb{Z}\geq 0}$. *Assume that*

- *A1 each digraph* $G(k)$ *has a self-loop at each node;*
- *A2 each non-zero edge weight* $p_{ij}(k)$*, including the self-loops weights* $p_{ii}(k)$*, is larger than a constant* $\varepsilon > 0$*; and*
- *A3 there exists a duration* $\delta \in \mathbb{N}$ *such that, for all times* $k \in \mathbb{Z} \geq 0$*, the union digraph* $G(k) \bigcup \ldots \bigcup G(k+\delta-1)$ *contains a globally reachable node.*

Then

1. *there exists a non-negative vector* $w \in \mathbb{R}^n$ *normalized to* $w_1+\ldots+w_n = 1$ *such that* $\lim_{k\to\infty} P(k)P(k-1)\ldots P(0) = 1_n w^T$*;*
2. *the solution to* $x(k+1) = P(k)x(k)$ *converges exponentially fast to* $(w^T x(0))1_n$
3. *if additionally each matrix in the sequence is doubly-stochastic, then* $w = \frac{1}{n}1_n$ *so that*

$$\lim_{k\to\infty} x(k) = \text{average}(x(0))1_n \tag{5.4}$$

Note 1: In a sequence with property 2, edges can appear and disappear, but the weight of each edge (that appears an infinite number of times) does not go to zero as $k \to \infty$.
Note 2: The existence of a globally reachable node is the connectivity requirement.

Note 3: Assumption $A3$ is a uniform connectivity requirement, that is, any interval of length δ must have the connectivity property. In equivalent words, the connectivity property holds for any contiguous interval of duration δ.

Lemma 11 (Global reachability over time) *Given a sequence of digraphs $\{G(k)\}_{k\in\mathbb{Z}_{\geq}}$ such that each digraph $G(k)$ has a self-loop at each node, the following two properties are equivalent:*

- *there exists a duration $\delta \in \mathbb{N}$ such that, for all times $k \in \mathbb{Z}_{\geq}$, the union digraph $G(k) \bigcup \cdots \bigcup G(k+\delta-1)$ contains a directed spanning tree;*
- *there exists a duration $\delta \in \mathbb{N}$ such that, for all times $k \in \mathbb{Z}_{\geq}$, there exists a node $j = j(k)$ that reaches all nodes $i \in 1,\ldots,n$ over the interval $\{k, k+\delta-1\}$ in the following sense: there exists a sequence of nodes $\{j, h_1, \ldots, h_{\delta-1}, i\}$ such that (j, h_1) is an edge at time k, (h_1, h_2) is an edge at time $k+1,\ldots,(h_{\delta-2}, h_{\delta-1})$ is an edge at time $k+\delta-2$, and $(h_{\delta-1}, i)$ is an edge at time $k+\delta-1$;*
  or, equivalently, for the reverse digraph,
- *exists a duration $\delta \in \mathbb{N}$ such that, for all times $k \in \mathbb{Z} \geq 0$, the union digraph $G(k) \bigcup \ldots \bigcup G(k+\delta-1)$ contains a globally reachable node;*
- *there exists a duration $\delta \in \mathbb{N}$ such that, for all times $k \in \mathbb{Z} \geq 0$, there exists a node j reachable from all nodes $i \in 1,\ldots,n$ over the interval $\{k, k+\delta-1\}$ in the following sense: there exists a sequence of nodes $\{j, h_1, \ldots, h_{\delta-1}, i\}$ such that $(j, h1)$ is an edge at time k, (h_1, h_2) is an edge at time $k+1,\ldots,(h_{\delta-2}, h_{\delta-1})$ is an edge at time $k+\delta-2$, and $(h_{\delta-1}, i)$ is an edge at time $k+\delta-1$;*

Proof 5 (Theorem 10: the max-min function is exponentially decreasing) *We start by noting that Assumptions $A1$ and $A3$ imply property of Lemma 11 about the existence of a duration δ with certain properties. Next, without loss of generality, we assume that at some time $h\delta$, for some $h \in \mathbb{N}$, the solution $x(h\delta)$ is not equal to a multiple of 1_n and, therefore, satisfies*

$V_{max-min}(x(h\delta)) > 0$. *Clearly,*

$$\begin{aligned} x((h+1)\delta) &= P((h+1)\delta - 1)\dots P(h\delta+1)P(h\delta)x(h\delta) \\ &= Px(h\delta). \end{aligned}$$

By Assumption A3, we know that there exists a node j reachable from all nodes i over the interval $\{h\delta, (h+1)\delta - 1\}$ in the following sense: there exists a sequence of nodes $\{j, h_1, \dots, h_{\delta-1}, i\}$ such that all following edges exist in the sequence of digraphs: (h_1, j) at time $h\delta$, (h_2, h_1) at time $h\delta+1, \dots, (i, h\delta-1)$ at time $(h+1)\delta - 1$. Therefore, assumption A2 implies

$$p_{h_1,j}(h\delta) \geq \varepsilon, p_{h_2,h_1}(h\delta+1) \geq \varepsilon \dots p_{h_i,h_{\delta-1}}((h+1)\delta-1) \geq \varepsilon, \tag{5.5}$$

and therefore their product satisfies

$$p_{h_i,h_{\delta-1}}((h+1)\delta-1)p_{h_{\delta-1},h_{\delta-2}}((h+1)\delta-2)\dots p_{h_2,h_1}(h\delta+1)p_{h_1,h_j}(h\delta) \geq \varepsilon^\delta. \tag{5.6}$$

Remarkably, this product is one term in the (i,j) entry of the row-stochastic matrix $P = P((h+1)\delta-1)\dots A(h\delta)$. In summary, Assumption 3 implies that there exists a node j such that, for all i, $P_{ij} \geq \varepsilon^\delta$ or, in other words, the row-stochastic matrix P has a positive column lower bounded by $\varepsilon^1\delta 1_n$.
We now invoke Lemma 9 to obtain that the row-stochastic matrix P is scrambling with $\tau_1(P) \leq 1 - \varepsilon^\delta$ and that the max-min disagreement function decreases according to

$$V_{max-min}(x((h+1)\delta)) \leq (1-\varepsilon^\delta)V_{max-min}(x(h\delta)) \tag{5.7}$$

This inequality proves exponential convergence of the cost function $k \to V_{max-min}(x(k))$ to zero and, together with the positive definiteness property of the $V_{max-min}$ function, convergence of $x(k)$ to a multiple of 1_n. We leave the other statements in Theorem 10 to the reader and refer to [29] and [30] for further details. □

5.3 Induced convergence metric

In this section, we exploit Theorem 10 in order to define a convergence metric for Algorithm A.1. To do so, we need to check if all the assumptions needed are satisfied by our model.
Assumptions $A1$ and $A3$ of Theorem 10 are are trivially satisfied by model in Equation (3.1), respectively by having a self-loop at each node and for $\delta = 1$. Assumption $A2$ needs some reasoning to be checked.
The non-zero edges in our model can only be of two different types:

1. $\eta^*_{constrained}(k) + (1 - \eta^*_{constrained}(k))p_{ii}$ for the diagonal entries;
2. $(1 - \eta^*_{constrained}(k))p_{ij}$ for all the other entries.

Let us define the following set of functions:

Definition 26 *Let be* $g_{ij} : (0,1) \to (0,1), \eta^*_{constrained}(k) \mapsto [P_\eta(k)]_{ij}$ *to all elements* $(i,j) \in E$. *In particular,* $\forall (i,j) \in E$

$$g_{ij}(\eta^*_{constrained}(k)) = \begin{cases} \eta^*_{constrained}(k) + (1 - \eta^*_{constrained}(k))p_{ii} & i = j \\ (1 - \eta^*_{constrained}(k))p_{ij} & i \neq j \end{cases} \tag{5.8}$$

Note that

$$g'_{ij}(\eta^*_{constrained}(k)) = \begin{cases} 1 - p_{ii} & i = j \\ -p_{ij} & i \neq j \end{cases} \tag{5.9}$$

hence

$$\min_k\{g_{ij}(\eta^*_{constrained}(k))\} = \begin{cases} \min_k\{\eta^*_{constrained}(k)\}(1 - p_{ii}) + p_{ii} & i = j \\ (1 - \max_k\{\eta^*_{constrained}(k)\})p_{ij} & i \neq j \end{cases} \tag{5.10}$$

and

$$\max_k\{g_{ij}(\eta^*_{constrained}(k))\} = \begin{cases} \max_k\{\eta^*_{constrained}(k)\}(1 - p_{ii}) + p_{ii} & i = j \\ (1 - \min_k\{\eta^*_{constrained}(k)\})p_{ij} & i \neq j \end{cases}. \tag{5.11}$$

Recall that by Proposition 3.35 for all k it holds:

$$\eta_L \leq \min_k\{\eta^*_{constrained}(k)\} \leq \max_k\{\eta^*_{constrained}(k)\} \leq \eta_U(k) \tag{5.12}$$

therefore, defining $\forall(i,j) \in E$

$$p_{L,D} = \min_i p_{ii} \tag{5.13}$$

$$p_{U,D} = \max_i p_{ii} \quad \text{with } 0 < p_{L,D} \leq p_{U,D} < 1 \tag{5.14}$$

$$p_{L,O} = \min_{i\neq j} p_{ij} \tag{5.15}$$

$$p_{U,O} = \max_{i\neq j} p_{ij} \quad \text{with } 0 < p_{L,O} \leq p_{U,O} < 1 \tag{5.16}$$

one has

$$[P_\eta(k)]_{ii} \in [\eta_L(1-p_{L,D}) + p_{L,D}, \eta_U(k)(1-p_{U,D}) + p_{U,D}] \tag{5.17}$$

$$[P_\eta(k)]_{ij} \in [(1-\eta_U(k))p_{L,O}, (1-\eta_L)p_{U,O}] \text{ with } i \neq j. \tag{5.18}$$

Considering that

$$\eta_L(1-p_{L,D}) + p_{L,D} > (1-\eta_U(k))p_{L,O} \tag{5.19}$$

$$\underbrace{\eta_L(1-p_{L,D})}_{>0} + \underbrace{p_{L,D} - p_{L,O}}_{=0} > \underbrace{-\eta_U(k)p_{L,O}}_{<0}, \tag{5.20}$$

where, for $p_{L,D} = p_{L,O}$, we exploited the computation presents in Appendix A.1. Therefore the minimum entry of P_η all over k is such that

$$\min_k [P_\eta(k)]_{ij} \geq (1-\bar{\eta}_U)p_{L,O} \quad \forall(i,j) \in E, \tag{5.21}$$

where $\bar{\eta}_U := \max_k \eta_U(k)$, which is computed by leveraging $\min_k c_{max}(k)$. Therefore, we get a minimum entry greater than zero for matrix P_η. Having a lower bound greater than zero, also Assumption $A2$ is satisfied.
Now, we can exploit Theorem 10 and define a convergence metric r as follows:

$$r := 1 - \varepsilon, \tag{5.22}$$

where ε turns out to be $\varepsilon = (1 - \bar{\eta}_U)p_{L,O}$ by exploiting Equation (5.21). More precisely, ε can be written in the following form:

$$\begin{aligned}\varepsilon &= \xi\left(1 - \max\left\{\eta_L, 1 - \frac{\bar{c}}{\omega||x(0)||_\infty}\right\}\right) \\ &= \xi \min\left\{\frac{\bar{c}}{\omega||x(0)||_\infty}, 1 - \eta_L\right\},\end{aligned} \tag{5.23}$$

where $\xi = 1/(1 + d_M)$ is defined as in (A.3) and $\bar{c} = \min_k\{c_{max,download}(k), c_{max,upload}(k)\}$. Finally, we can express the value of r depending on the parameters present as:

$$r = 1 - (1 - \eta_L)p_{L,O}, \tag{5.24}$$

in the case where $\eta_L > 1 - \frac{\bar{c}}{\omega||x(0)||_\infty}$, or, alternatively:

$$r = 1 - \frac{\bar{c}}{\omega||x(0)||_\infty} \min_{i,j} p_{ij}, \tag{5.25}$$

in the case where $1 - \frac{\bar{c}}{\omega||x(0)||_\infty} > \eta_L$. The two cases, depicted by Equations (5.24) and (5.25), are representative of the model in both constrained and unconstrained cases. This is expressed by the presence of the term η_L in the unconstrained case and the term $1 - \frac{\bar{c}}{\omega||x(0)||_\infty}$ in the constrained case. In a more compact form we can express r as follows.

$$r = 1 - \xi \min\left\{\frac{\bar{c}}{\omega||x(0)||_\infty}, 1 - \eta_L\right\} \tag{5.26}$$

It is worth to notice that, reasonably, the convergence rate r is increasing* as the capacity value $\bar{c}$ diminishes in the constrained case. On the other hand, r is depending only on topological factors in the unconstrained case.

*meaning the consensus convergence is slowing down

6

Numerical results

In this chapter, we provide some numerical results from the simulation resting on a real case scenario. In particular, we apply Algorithm A.1 on a graph-based model of the Cavallino Treporti drainage water network and some consideration on the results and the convergence metric are given.

6.1 Real scenario simulation with identical constraint values

In this section, we apply Algorithm A.1 to a simplified graph-based model of the Cavallino Treporti drainage water network. In Figure 6.1 we can appreciate a satellite view of the area of interest where we have highlighted the canals interested by our study.

Figure 6.1: Satellite view of Cavallino-Treporti with highlighted canal network

In Figure 6.1 it is also interesting to notice that each canal junction is enumerated to satisfy one of the four requirements, i.e. the CJ vector, needed to run the algorithm. So, we model as canal every connection between two junctions present in the vector CJ. In particular, this representation accounts for 22 junctions that are composing a total of 26 canals. This just described is a semplification of the Cavallino Treporti actual canal network. Here, an equivalent topological framework is taken into account, meaning that we consider as a single canal for each couple of consecutive canals connected to other canals both at the beginning of the first one and at the end of the second one, without any other junction, if not between them. This choice was made for the sake of simplicity and without loss of generality w.r.t. the applied algorithm. After getting the CJ vector, we need the vector representing the initial state of the system, $x(0)$. This vector, that in a real life scenario is meant to be measured, is randomly initialized in MATLAB environement and has values included between 1 and 10. This satisfies the first of the two requirements of Assumption 1, that require the system not to be empty and gives us a random starting point. The values for the parameter $c_{max}(k)$, selected in order to represent a reasonable exchange capacity limit, is then set to $c_{max}(k) = c_{max} = 1m/s$, meaning that the system actuators are allowed to increase or decrease the height of each canal for each iteration at most by $1m$. The agreement threshold γ, that can be selected arbitrarily small, is set to $\gamma = 10^{-1}$, so to obtain a difference of height between canals smaller than $10cm$. All the information just mentioned are contained in the following in Table 6.1:

CANAL	STATE (m)	START	END	$c_{max}(k)$	γ
1	7	1	2	1	0.1
2	2	2	3		
3	2	2	22		
4	5	3	4		
5	10	3	5		
6	4	3	14		
7	6	5	6		
8	3	5	13		
9	8	6	7		
10	3	6	10		
11	6	7	8		
12	7	7	9		
13	9	7	10		
14	10	10	11		
15	6	11	12		
16	2	11	13		
17	2	13	14		
18	3	14	15		
19	9	15	16		
20	3	15	17		
21	9	16	18		
22	3	17	18		
23	10	18	19		
24	4	19	20		
25	2	19	22		
26	3	21	22		

Table 6.1: Initial informations of the Cavallino-Treporti water network

Figure 6.2 shows how the simplified model of the Cavallino water network looks like. The information on canal heights are also reported just above each canal representation that is proportional to its thickness.

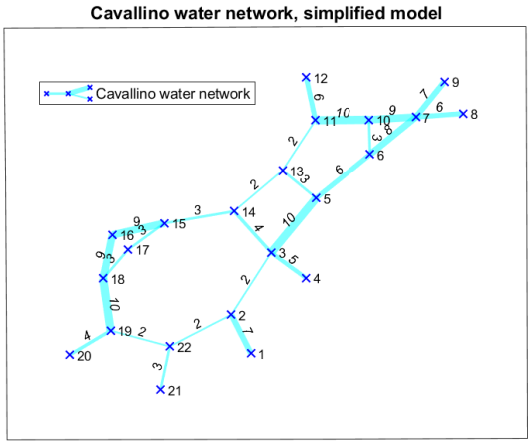

Figure 6.2: Simplified model of the Cavallino-Treporti canal network with canal heights from Table 6.1

Running the water distribution algorithm with this setup gives us the following result, depicted in Figure 6.3, that reports the variation of the canal heights over all the iterations:

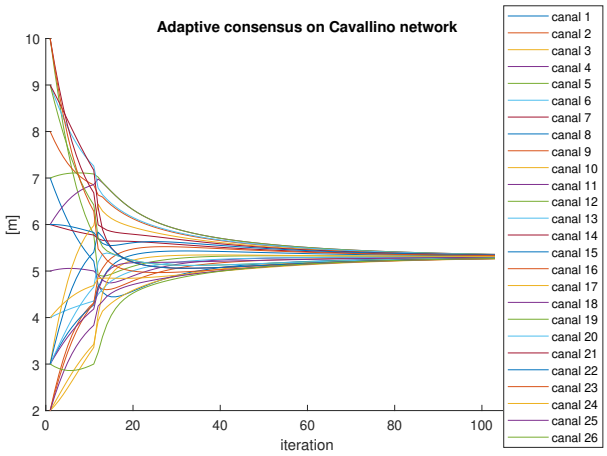

Figure 6.3: Adaptive consensus algorithm applied on the Cavallino-Treporti canal network

In Figure 6.3, it is possible to observe that the consensus protocol converges, according to the agreement threshold γ, in 103 iterations. We highlight the fact that, at iteration 11, it is possible to notice that all the trajectories converge faster to the average. This is due to the fact that from

iteration 11, the algorithm is selecting $\eta = \eta_L$, which ensures a higher rate of convergence. What just mentioned is depicted in Figure 6.4, where are highlited the trajecories around iteration 11, and Table 6.2, that summarizes the choices of η selected over all the iterations.

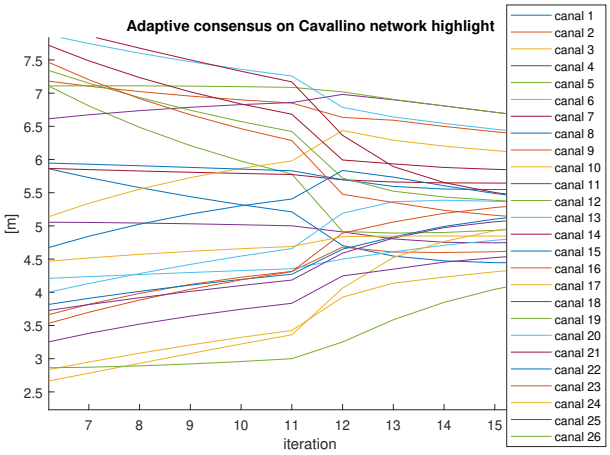

Figure 6.4: Adaptive consensus algorithm applied on the Cavallino-Treporti canal network, highlight on iteration 11

η selected over the iterations							
iter	1	2	3	4	5	6	7
η	0.863	0.856	0.848	0.840	0.835	0.831	0.827
iter	8	9	10	11	12	...	103
η	0.822	0.818	0.815	ζ	ζ	...	ζ

Table 6.2: η selection

As expected, before being selected until the end as ζ, the value of η is decreasing throughout the first iterations confirming the convex combination property of the model structure. Also, note that ζ has to be used as $\varsigma_{\mathcal{M}} = 0.3845 \geq 0$, see Equations (3.29) and (3.3). After iteration 11, the value assigned to η is $\eta = \eta_L = \zeta = 0.001$. In this particular numerical simulation the rate of convergence, computed as in Equation (5.22), is $r = 1 - \varepsilon = 0.9958$. It is worthy to observe that this value of r, being close to 1 describes a slow rate of convergence. This is due to the fact that the system, in the particular case taken as example, has strict limits of exchange capacity that slow down the convergence rate. For this reason, we

can say that, this high value of r, supports the correctness of the solution presented in this thesis. To underline the latter statement, in Figure 6.5 it is shown the comparison between $||x(k+1)-x(k)||_\infty$ and c_{max}. As can be notice in Figure 6.5, the value of $||x(k+1)-x(k)||_\infty$ remains below c_{max} for all the simulation time. On top of that, it is possible to notice a peak at iteration 11 that represents the point in time where η_L is chosen and keeps to be chosen until the algorithm stops.

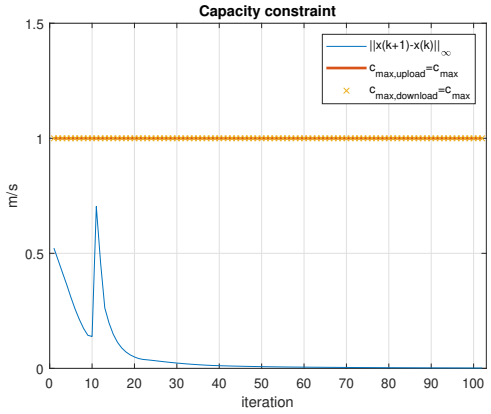

Figure 6.5: Graphical comparison between c_{max} and $||x(k+1)-x(k)||_\infty$.

The final result of this simulation is reported in Figure 6.6

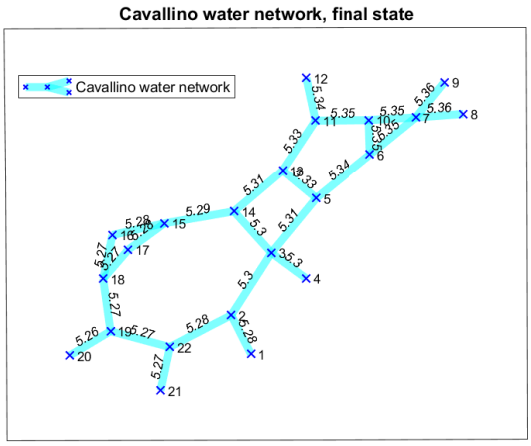

Figure 6.6: Simplified model of the Cavallino-Treporti canal network with final states

The final states are summarized in Table 6.3. The maximum dispersion $V_{max-min}(x(103)) = (5.359 - 5.263) = 0.096m$ around the average value of the initial state that is $5.31m$. The comparison between $V_{max-min}(x(k))$ and γ is reported in Figure 6.7.

CANAL	STATE (m)	START	END	$c_{max}(k)$	γ
1	5.28	1	2	1	0.1
2	5.30	2	3		
3	5.28	2	22		
4	5.30	3	4		
5	5.31	3	5		
6	5.30	3	14		
7	5.34	5	6		
8	5.33	5	13		
9	5.35	6	7		
10	5.35	6	10		
11	5.36	7	8		
12	5.36	7	9		
13	5.35	7	10		
14	5.35	10	11		
15	5.34	11	12		
16	5.33	11	13		
17	5.31	13	14		
18	5.29	14	15		
19	5.28	15	16		
20	5.28	15	17		
21	5.27	16	18		
22	5.27	17	18		
23	5.27	18	19		
24	5.26	19	20		
25	5.27	19	22		
26	5.27	21	22		

Table 6.3: Final states of the Cavallino-Treporti water network

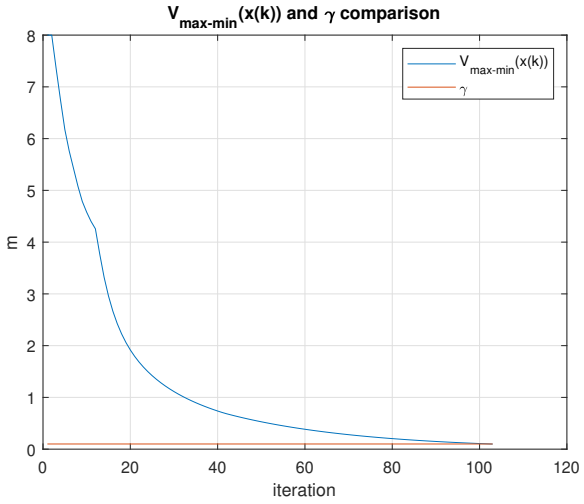

Figure 6.7: comparison between $V_{max-min}(x(k))$ and γ

6.2 Real scenario simulation with different constraint values

In this section, we apply Algorithm A.1 within the same framework as in Section 6.1 with constant but different values for the parameters $c_{max,download}$ and $c_{max,upload}$. We set the same initial information given in Table 6.1 with the exception for $c_{max}(k)$. In this simulation we will consider $c_{max,download} = 1$ and $c_{max,upload} = 0.5$. Running the algorithm with this setup gives us the following result, depicted in Figure 6.8, that reports the variation of the canal heights over the iterations.

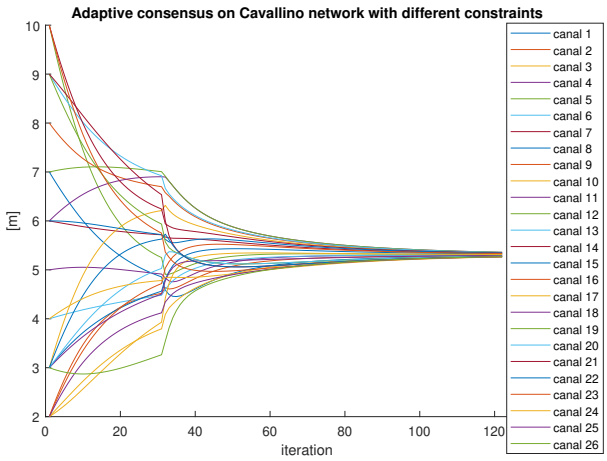

Figure 6.8: Adaptive consensus algorithm applied on the Cavallino-Treporti canal network with $c_{max,download} > c_{max,upload}$

In Figure 6.8, it is shown how the trajectories converge in case of different values for $c_{max,download}$ and $c_{max,upload}$. The shape of the trajectories, as the previous case, highlight that the selection changes from $\eta^*_{constrained}(k)$ to η_L that happens at iteration 31. In particular, since for this particular simulation $c_{max,download} > c_{max,upload}$, $\eta^*_{constrained,upload}(k)$ is selected as η because this represents the tightest constraint applied. This just mentioned is summarized in Table 6.4, where the choice of η throughout all the iterations are reported.

η selected through the iterations			
iter	1	...	30
η	$\eta^*_{constrained,upload}(1)$ =0.932	...	$\eta^*_{constrained,upload}(30)$ =0.902
iter	31	...	122
η	ζ	...	ζ

Table 6.4: η selection in $c_{max,download} > c_{max,upload}$ case

In Figure 6.9, it is possible to appreciate that $||x(k+1) - x(k)||_\infty$ satisfies the tightest constraint, represented by $c_{max,upload}$, throughout all the iterations.

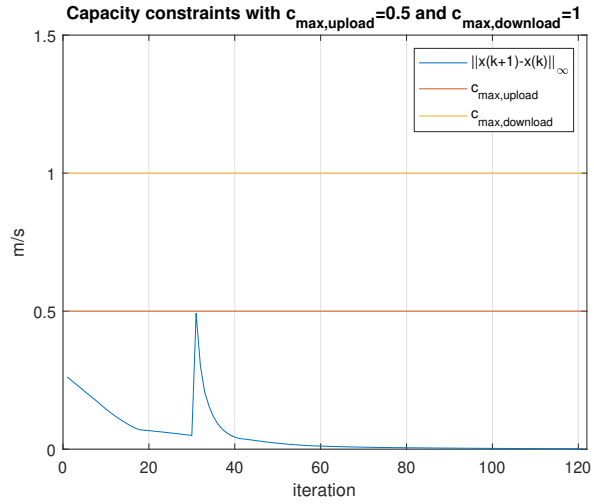

Figure 6.9: Graphical comparison between $c_{max,upload}$, $c_{max,download}$ and $||x(k+1)-x(k)||_\infty$.

Also, in this case, the agreement is reached and the final states situation is depicted in Figure 6.11. The maximum dispersion $V_{max-min}(x(122)) = (5.358 - 5.263) = 0.095m$ around the average value of the initial state that is $5.31m$. The comparison between $V_{max-min}(x(k))$ and γ is reported in Figure 6.10.

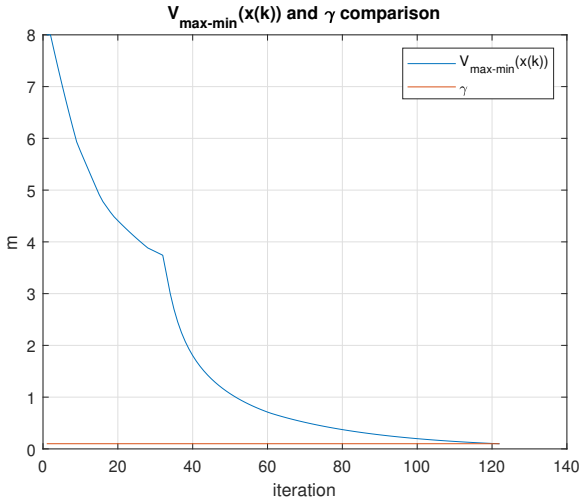

Figure 6.10: comparison between $V_{max-min}(x(k))$ and γ

The convergence rate for this simulation results to be $r = 1-\varepsilon = 0.9989$. It is interesting to notice that it is a higher value that in Section 6.1, con-

firming the dependence on the parameter $c_{max,upload}$ as, for this particular case, it is set at a smaller value than Section 6.1. Remarkably, the comparison is meaningful just because we voluntarily adopted the same initial conditions for the two case considered in Sections 6.1 and 6.2.

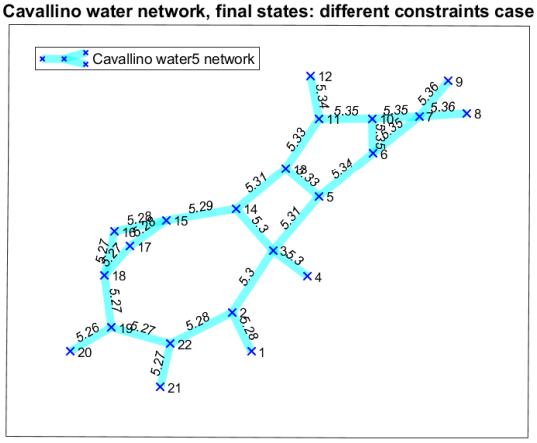

Figure 6.11: Simplified model of the Cavallino-Treporti canal network with final states: different constraints case

6.3 Real scenario simulation with variable constraint values

In this section, we apply Algorithm A.1 within the same framework as in Section 6.1 with variable values for the parameters $c_{max,download}(k)$ and $c_{max,upload}(k)$. We consider the same initial information given in Table 6.1 with the exception for the constraints. In this simulation we will consider the following values for $c_{max,download}(k)$ and $c_{max,upload}(k)$:

- for $0 \leq k \leq 49$, $c_{max,download}(k) = 0.1$ and $c_{max,upload}(k) = 0.1$
- for $50 \leq k \leq 99$, $c_{max,download}(k) = 0.15$ and $c_{max,upload}(k) = 0.2$
- for $100 \leq k \leq 349$, $c_{max,download}(k) = 0.1$ and $c_{max,upload}(k) = 0.1$
- for $350 \leq k \leq 399$, $c_{max,download}(k) = 0.05$ and $c_{max,upload}(k) = 0.075$

- for $400 \leq k \leq 486$, $c_{max,download}(k) = 0.1$ and $c_{max,upload}(k) = 0.1$.

The values of k that determine the constraints are chosen are just for explanation purposes and without loss of generality. In particular, considering 0.1 as standard constraint value for this particular case, we want to show how the algorithm behaves both with higher and smaller capacity values during one application. This could describe a real case scenario in which, for some reason (e.g. an actuator fault, Acqua alta phenomenon, tides...), it occurs for the system increasing the capability of exchanging information (for $50 \leq k \leq 99$), or decreasing the capability of exchanging information (for $350 \leq k \leq 399$). Running the algorithm with this setup gives us the following result, depicted in Figure 6.12, that reports the variation of the canal heights over the iterations.

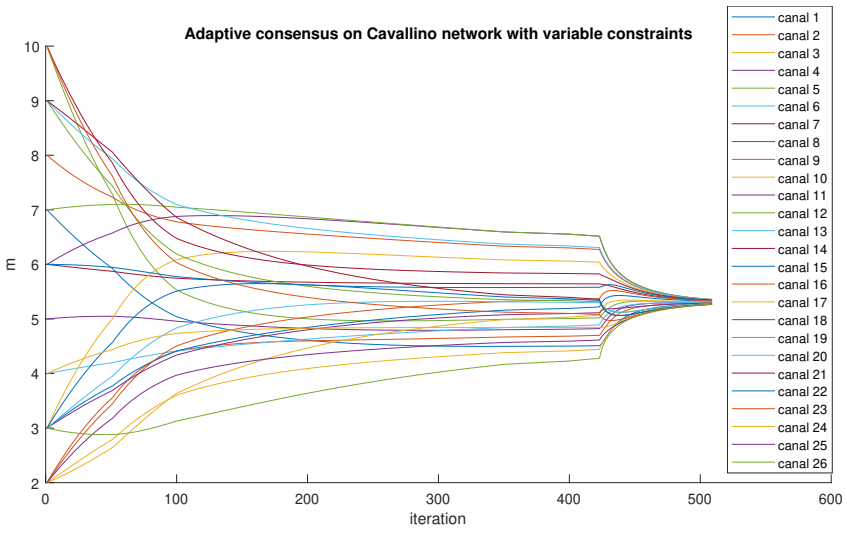

Figure 6.12: Adaptive consensus algorithm applied on the Cavallino-Treporti canal network with variable constraints

In Figure 6.12 it is shown how the trajectories converge in case of variable values for $c_{max,download}(k)$ and $c_{max,upload}(k)$. The agreement is reached also in this case, at iteration $k = 509$, and the final states situation is depicted in Figure 6.14. The maximum dispersion is $V_{max-min}(x(509)) = (5.358 - 5.263) = 0.095m$ around the average value of

the initial state that is, as previous cases, $5.31m$. The comparison between $V_{max-min}(x(k))$ and γ is reported in Figure 6.13.

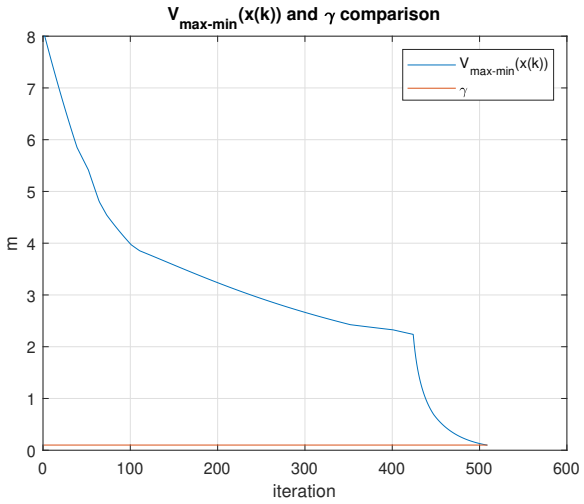

Figure 6.13: comparison between $V_{max-min}(x(k))$ and γ

The convergence rate for this simulation results to be $r = 1-\varepsilon = 0.99995$. Again, this value is related to the choice made for $c_{max,download}(k)$ and $c_{max,upload}(k)$, which are set relatively small w.r.t. to 1.

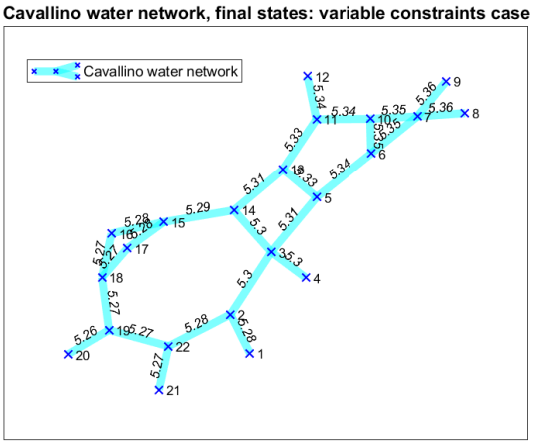

Figure 6.14: Simplified model of the Cavallino-Treporti canal network with final states: variable constraints case

In Figure 6.15 it is possible to appreciate that, also in this case with variability of parameters, $||x(k+1) - x(k)||_\infty$ satisfies for all $k = 0, 1, 2, \ldots$ the tightest constraint throughout all the iterations. It is also worthy to notice that this simulation supports the robustness of the solution proposed. In fact, the system successfully managed to increase or decrease its exchange capability relatively to different requirements given as input.

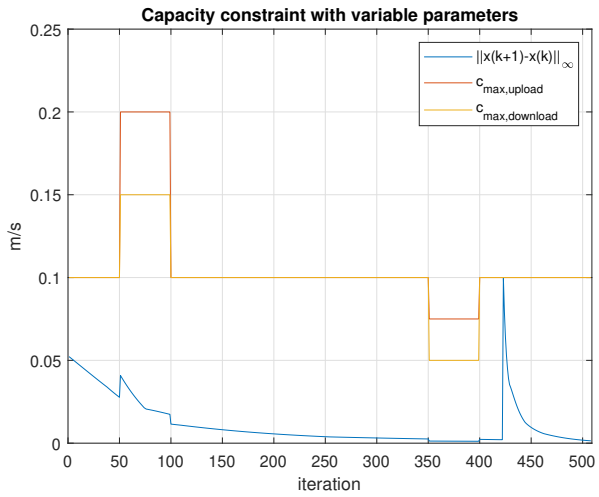

Figure 6.15: Graphical comparison between $c_{max,upload}(k)$, $c_{max,download}(k)$ and $||x(k+1) - x(k)||_\infty$.

7 Conclusions

In this thesis, it is proposed an automated solution to water management in a water network. In particular, focusing on the Cavallino di Venezia drainage network, our aim is to distribute evenly the water throughout the whole network with the presence of water exchange capacity limits. We have interpreted the network as a graph, and designed a model taking into account the main physical constraints that intrinsically characterize a generic water network. Then, we have derived the expression of a time-varying parameter that can control the capacity of water exchanged between nodes and can be autonomously set to satisfy possible limits in capacity flow. The approach used allows us to provide a distributed protocol with some degree of versatility, meaning that it can be applied on every network modelled as connected graph having the same kind of requirements to satisfy. The presented algorithm is iterative and after obtaining some initialization information as input, it provides the information relative to the feasible variation of the system states according to the imposed flow constraints. This information can be used, for example, to set the actuators that are responsible for the mechanical actions needed. In addition, we propose a convergence analysis resting upon Lyapunov-based theorems, which is presented in order to suggest a convergence metric. The latter has been used as measure of how quick the protocol and

the constraints enforced allow the states to converge towards the mean of the initial values. Finally, some numerical simulations are proposed to test a simplified model of the Cavallino di Venezia drainage network. The simulations take into consideration different constraint conditions: equal and constant constraint values, different and constant constraint values and variable constraint values. In each case the protocol successfully reached an agreement value for all the states, within the given relative agreement threshold. Furthermore, the proposed convergence metric indicates a slow rate of convergence, confirming the correctness, robustness and reliability of the protocol that is ensured not to overcome the imposed limits.

The next research direction could be pointed to various aspects inherent to this thesis. Clearly, the main problem that can be faced from our perspective is evaluating the physical and mechanical implementation issues that the proposed protocol requires. However, this solution is not claimed to be unique relatively to the assessed water distribution problem. Furthermore, some improvements can be achieved also in the same framework of the solution proposed in this thesis. For example, one might consider also negative values for the parameter η^*, defined in Equation (3.3), and try to derive one feasible solution for time-varying models.

A Appendix

A.1 On the minimum entry of the Metropolis-Hasting matrix

Here we provide more details on the computations for the minimum entry of a Metropolis-Hasting matrix.

Lemma 12 *Function* $f : \mathbb{R} \to \mathbb{R} : x \mapsto \dfrac{x}{1+x}$ *is strictly increasing for all*

Proof 6 *The first derivative of* $f(x)$ *is computed as*

$$f'(x) = \frac{1}{(1+x)^2}. \tag{A.1}$$

Since $f'(x) > 0$ *for all* $x \in \mathbb{R} \setminus \{-1\}$, *the proof is concluded.* □

Proposition 5 *Let us consider an undirected and connected graph* $G = (V, E)$. *Denote with* $N_i = \{j \in V \mid (i,j) \in E\}$ *and* $d_i = |N_i|$ *the i-th neighborhood and i-th degree, respectively, for all vertices* $i = 1, \dots, n$ *in* V. *Let us also define the Metropolis-Hasting matrix* $P \in \text{stoch}^2(\mathbb{R}^{n \times n})$ *associated to* G

as

$$[P]_{ij} = p_{ij} = \begin{cases} \dfrac{1}{1+\max(d_i, d_j)}, & \forall j \in N_i; \\ 0, & \forall j \notin N_i \cup \{i\}; \\ 1 - \sum\limits_{j \in N_i} p_{ij}, & \textit{otherwise}. \end{cases} \tag{A.2}$$

Then, the smallest nonzero entry $\xi = \min\limits_{p_{ij}>0}\{p_{ij}\}$ *of* P *is yielded by*

$$\xi = \min_{(i,j)\in E}\{p_{ij}\} = \min_{i \in V}\{p_{ii}\} = \frac{1}{1+d_M}, \tag{A.3}$$

where $d_M = \max\limits_{1,\ldots,n}\{d_i\}$ *denotes the maximum degree of* G.

Proof 7 *The proof is divided into two parts, dealing with the off-diagonal elements and diagonal elements of* P *in (i) and (ii), respectively.*

(i) Firstly, we show that

$$\min_{i \in V}\{p_{ii}\} = \frac{1}{1+d_M}. \tag{A.4}$$

Equality in (A.4) *can be verified by recalling that, for all* $(i,j) \in E$, $p_{ij} = (1+\max(d_i, d_j))^{-1}$ *is defined as in* (A.2). *Noting that, generally, one has*

$$\max(d_i, d_j) \leq d_M, \quad \forall (i,j) \in E, \tag{A.5}$$

and observing that there exist some $(i,j) \in E$ *for which the equality in* (A.5) *holds, then claim* (A.4) *is trivially proven.*

(ii) Secondly, we show that

$$\min_{(i,j)\in E}\{p_{ij}\} = \frac{1}{1+d_M}. \tag{A.6}$$

Consider the following chain of inequalities for all $i = 1, \dots, n$*:*

$$\sum_{j \in N_i} \frac{1}{1 + \max(d_i, d_j)} \leq \sum_{j \in N_i} \frac{1}{1 + \min\limits_{j \in N_i} \Big\{ \max(d_i, d_j) \Big\}} \leq \frac{d_i}{1 + \min\limits_{j \in N_i} \Big\{ \max(d_i, d_j) \Big\}}. \tag{A.7}$$

From the last expression of (A.7)*, assign* $\delta_i := \min\limits_{j \in N_i} \Big\{ \max(d_i, d_j) \Big\}$ *and consider the following inequality:*

$$\frac{d_i}{1 + \delta_i} \leq \frac{\dfrac{d_i + \delta_i}{2}}{1 + \dfrac{d_i + \delta_i}{2}}. \tag{A.8}$$

After easy calculations, expression in (A.8) *can be rewritten as*

$$d_i + d_i^2 \leq \delta_i + \delta_i^2. \tag{A.9}$$

Since $1 \leq d_i \leq \delta_i$ *holds true by the connectivity assumption and the definition of* δ_i*, it can be concluded that* (A.8)-(A.9) *is an identity for all* $i = 1, \dots, n$*.*
From the r.h.s. of (A.8)*, let us also assign* $\bar{d}_i = (d_i + \delta_i)/2$*. Recalling* (A.7)*, the following inequality can be now deduced:*

$$\sum_{j \in N_i} \frac{1}{1 + \max(d_i, d_j)} \leq \frac{\bar{d}_i}{1 + \bar{d}_i}, \quad \forall i = 1, \dots, n. \tag{A.10}$$

We now resort to the fact that $\forall (z_1, z_2)$*, with* $z_1, z_2 \neq -1$*, if* $z_1 \leq z_2$ *then it holds that* $z_1/(1 + z_1) \leq z_2/(1 + z_2)$*. In particular, observing that* $\bar{d}_i \leq d_M$*, as both* $d_i \leq d_M$ *and* $\delta_i \leq d_M$*, this allows us to find an upper bound for the r.h.s. of* (A.10)*, implying that*

$$\sum_{j \in N_i} \frac{1}{1 + \max(d_i, d_j)} \leq \frac{d_M}{1 + d_M}, \quad \forall i = 1, \dots, n. \tag{A.11}$$

From (A.11) *and the definition of the diagonal entries* p_{ii} *in* (A.2) *one has*

$$p_{ii} = 1 - \sum_{j \in N_i} \frac{1}{1 + \max(d_i, d_j)} \geq 1 - \frac{d_M}{1 + d_M} = \frac{1}{1 + d_M}, \quad \forall i = 1, \ldots, n. \tag{A.12}$$

Noting that there exists some $i \in V$ *for which equality in* (A.12) *holds, claim* (A.6) *is satisfied and thus the proof is concluded.* □

A.2 Distributed computation of $\left\|x(k)\right\|_\infty$

To compute $\left\|x(k)\right\|_\infty$ in a distributed fashion the max-consensus protocol is generally exploited. In particular, this graph-based algorithm ensures the distributed computation of the maximum magnitude of the state entries $x_i(k)$ within a connected graph in finite time. The hypothesis of connectivity is fundamental to guarantee correct outcome (see Theorem 2), due to the fact that such a protocol is based on the information exchange among neighboring nodes.

In Algorithm A.1, the pseudocode of the protocol in question, exploited to compute $\left\|x(k)\right\|_\infty$, is reported. This protocol converges over any given undirected and connected topology G and returns the maximum value in at most a time proportional to the diameter φ_G. This fact occurs because the value of each auxiliary state z_i is updated through its highest neighbor's value $z_j, j \in N_i$, and backing variables sat_i are employed. Thus, in the worst case scenario*, the propagation of the information being exchanged about the maximum value $\max_i\{|x_i(k)|\}$ clearly spreads among all nodes within at most φ_G steps. It is also worth to note that, since in this peculiar setting we are dealing with only positive values of the state vector x, line 2 can be actually simplified and rewritten as $z_i(k) \leftarrow x_i(k)$. Lastly, under the assumption that Algorithm A.1 can be invoked and terminated within the time interval $[k, k+1]$, the computation of $\left\|x(k)\right\|_\infty$

*Represented by the line graph.

is returned as x_{max} and can be used to guarantee the correct execution of the main algorithm presented in Section 4.2. To this aim, each update of z_i is regularly spaced in time of an interval $\Delta_{\varphi_G} \in (0, \varphi_G^{-1})$.

Algorithm A.1 Distributed computation of $\left\|x(k)\right\|_\infty$

Require: $G = (V, E)$, connectivity of G, φ_G, $\Delta_{\varphi_G} \in (0, \varphi_G^{-1})$, $x(k)$
Ensure: distributed value of $\left\|x(k)\right\|_\infty$

```
for each i ∈ V
    z_i(k) ← |x_i(k)|
end for
for t = 0, ..., φ_G − 1
    for each i ∈ V
        sat_i ← z_i(k + tΔ_{φ_G})
        for each j ∈ N_i
            if z_j(k + tΔ_{φ_G}) > sat_i
                sat_i ← z_j(k + tΔ_{φ_G})
            end if
        end for
        z_i(k + (t + 1)Δ_{φ_G}) ← sat_i
    end for
end for
return z_i(k + φ_G Δ_{φ_G}), for any i ∈ V
```

References

[1] O. Bello, A. M. Abu-Mahfouz, Y. Hamam, P. R. Page, K. B. Adedeji, and O. Piller, "Solving management problems in water distribution networks: A survey of approaches and mathematical models," *Water*, vol. 11, no. 3, 2019. [Online]. Available: https://www.mdpi.com/2073-4441/11/3/562

[2] C. ISMAR. (2022) Acqua alta a venezia. [Online]. Available: http://www.ismar.cnr.it/infrastrutture/previsioni/acqua-alta-venezia

[3] P. U. B. Singapore, "Managing the water distribution network with a smart water grid," *Smart Water*, vol. 1, no. 1, 2016. [Online]. Available: https://doi.org/10.1186/s40713-016-0004-4

[4] A. Babayan, Z. Kapelan, D. Savic, and G. Walters, "Least-cost design of water distribution networks under demand uncertainty," *JOURNAL OF WATER RESOURCES PLANNING AND MANAGEMENT-ASCE*, vol. 131, pp. 375–382, 09 2005.

[5] C. Xu and I. C. Goulter, "Reliability-based optimal design of water distribution networks," *Journal of Water Resources Planning and Management*, vol. 125, no. 6, pp. 352–362, 1999.

[6] R. Sheikholeslami and A. Kaveh, "Vulnerability assessment of water distribution networks: Graph theory method," *International Journal of Optimization in Civil Engineering*, vol. 5, pp. 283–299, 01 2015.

[7] M. M. Giraldo-González and J. P. Rodríguez, "Comparison of statistical and machine learning models for pipe failure modeling in water distribution networks," *Water*, vol. 12, no. 4, 2020. [Online]. Available: https://www.mdpi.com/2073-4441/12/4/1153

[8] L. Grbčić, L. Kranjčević, and S. Družeta, "Machine learning and simulation-optimization coupling for water distribution network contamination source detection," *Sensors*, vol. 21, no. 4, 2021. [Online]. Available: https://www.mdpi.com/1424-8220/21/4/1157

[9] E. Price and A. Ostfeld, "Graph theory modeling approach for optimal operation of water distribution systems," *Journal of Hydraulic Engineering*, vol. 142, no. 3, p. 04015061, 2016.

[10] A. Di Nardo, C. Giudicianni, R. Greco, M. Herrera, and G. F. Santonastaso, "Applications of graph spectral techniques to water distribution network management," *Water*, vol. 10, no. 1, 2018. [Online]. Available: https://www.mdpi.com/2073-4441/10/1/45

[11] O. Feinerman and A. Korman, "Theoretical distributed computing meets biology: A review," in *Distributed Computing and Internet Technology*, C. Hota and P. K. Srimani, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 1–18.

[12] S. Navlakha and Z. Bar-Joseph, "Distributed information processing in biological and computational systems," *Commun. ACM*, vol. 58, no. 1, p. 94–102, dec 2014. [Online]. Available: https://doi.org/10.1145/2678280

[13] M. Herlihy, N. Shavit, V. Luchangco, and M. Spear, "Chapter 6 - universality of consensus," in *The Art of Multiprocessor Programming (Second Edition)*, second edition ed., M. Herlihy, N. Shavit, V. Luchangco, and M. Spear, Eds. Boston: Morgan Kaufmann, 2021, pp. 129–143. [Online]. Available: https://www.sciencedirect.com/science/article/pii/B978012415950100015X

[14] R. Olfati-Saber, J. A. Fax, and R. M. Murray, "Consensus and cooperation in networked multi-agent systems," *Proceedings of the IEEE*, vol. 95, no. 1, pp. 215–233, 2007.

[15] N. H. M. Yusof, N. A. M. Subha, F. S. Ismail, and N. Hamzah, "Logical consensus-based control for water distribution system with physical faults," in *2020 IEEE 10th International Conference on System Engineering and Technology (ICSET)*, 2020, pp. 118–122.

[16] K. Cai and H. Ishii, "Average consensus on general strongly connected digraphs," *Automatica*, vol. 48, no. 11, pp. 2750–2761, 2012. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0005109812004049

[17] M. Zhu and S. Martínez, "Discrete-time dynamic average consensus," *Automatica*, vol. 46, no. 2, pp. 322–329, 2010. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0005109809004828

[18] M. P. Fanti, M. Franceschelli, A. M. Mangini, G. Pedroncelli, and W. Ukovich, "Discrete consensus in networks with constrained capacity," in *52nd IEEE Conference on Decision and Control*, 2013, pp. 2012–2017.

[19] T. Yang, Z. Meng, D. V. Dimarogonas, and K. H. Johansson, "Global consensus for discrete-time multi-agent systems with input saturation constraints," *Automatica*, vol. 50, no. 2, pp. 499–506, 2014. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0005109813005256

[20] Y. Cao, "Consensus of multi-agent systems with state constraints: a unified view of opinion dynamics and containment control," in *2015 American Control Conference (ACC)*, 2015, pp. 1439–1444.

[21] M. Fabris, G. Michieletto, and A. Cenedese, "On the distributed estimation from relative measurements: a graph-based convergence analysis," in *2019 18th European Control Conference (ECC)*. IEEE, 2019, pp. 1550–1555.

[22] ——, "A proximal point approach for distributed system state estimation," *IFAC-PapersOnLine*, vol. 53, no. 2, pp.

2702–2707, 2020, 21st IFAC World Congress. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2405896320306984

[23] ——, "A general regularized distributed solution for system state estimation from relative measurements," *IEEE Control Systems Letters*, vol. 6, pp. 1580–1585, 2022.

[24] J. Gross and J. Yellen, "Graph theory and its application," *The Mathematical Gazette*, vol. 84, 03 2000.

[25] S. Skiena, *Implementing Discrete Mathematics: Combinatorics and Graph Theory with Mathematica*. USA: Addison-Wesley Longman Publishing Co., Inc., 1991.

[26] E. Weisstein, *CRC Concise Encyclopedia of Mathematics*. CRC Press, 2002. [Online]. Available: https://books.google.it/books?id=aFDWuZZslUUC

[27] V. Schwarz, G. Hannak, and G. Matz, "On the convergence of average consensus with generalized metropolis-hasting weights," in *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2014, pp. 5442–5446.

[28] F. Bullo, *Lectures on network systems*. Kindle Direct Publishing Santa Barbara, CA, 2019, vol. 1.

[29] L. Moreau, "Stability of multiagent systems with time-dependent communication links," *IEEE Transactions on Automatic Control*, vol. 50, no. 2, pp. 169–182, 2005.

[30] J. M. Hendrickx and V. Blondel, "Graphs and networks for the analysis of autonomous agent systems." Ph.D. dissertation, Catholic University of Louvain, Louvain-la-Neuve, Belgium, 2008.