# Gossip-based Communication for Distributed Energy Management

Geert Maan

*September 26, 2024*

University of Twente

UNIVERSITY OF TWENTE.

Faculty of Electrical Engineering, Mathematics and Computer Science (EEMCS)
Computer Architecture for Embedded Systems (CAES)

Master's Thesis

# Gossip-based Communication for Distributed Energy Management

Geert Maan

*Chair* Dr.ir. M.E.T. Gerards
EEMCS
University of Twente

*Supervisor* Dr.ir. G. Hoogsteen
EEMCS
University of Twente

*Supervisor* Ir. A. Pappu
EEMCS
University of Twente

*External* Dr. S. Bayhan
EEMCS
University of Twente

September 26, 2024

**Geert Maan**
*Gossip-based Communication for Distributed Energy Management*
Master's Thesis, September 26, 2024
Supervisors: Dr.ir. M.E.T. Gerards, Dr.ir. G. Hoogsteen, Ir. A. Pappu, and Dr. S. Bayhan

**University of Twente**
*Computer Architecture for Embedded Systems (CAES)*
Faculty of Electrical Engineering, Mathematics and Computer Science (EEMCS)
Drienerlolaan 5
P.O. Box 207
7500 AE, Enschede
The Netherlands

# Abstract

Current-day developments in the energy distribution domain, such as large-scale electrification and increasing distributed generation of electricity, have resulted in a congested electricity grid. A major challenge that has to be resolved is the supply-demand mismatch, for which many Demand-Side Management (DSM) approaches are proposed in literature. DSM requires the coordination energy generation, storage, and usage of the participants in a microgrid, and thus communication between these participants. This is the reason why the communication layer is becoming an integral part of energy distribution systems.

This thesis focuses on the communication layer and investigates how a distributed network architecture and gossip-based communication can aid the implementation of DSM. A distributed network architecture has the advantage of lacking single point of failures and communication bottlenecks, making it a robust, reliable, and scalable solution for the implementation of a communication network. However, the application of a distributed network architecture also comes with challenges, one being network convergence. Gossip-based communication is a promising solution for the network convergence process, where all nodes in a network cooperate to become aware of the global state of the network. The convergence speed of a communication algorithm is important, because the proper operation of DSM applications often has certain timing constraints. Hence, this thesis investigates how the convergence speed of gossip-based communication can be improved with the design of the communication algorithm, and how knowledge about network properties can be utilized for this purpose. The central research question of this thesis is: *How can communication mechanisms and network properties be utilized to improve convergence speed, to aid the implementation of Demand-Side Management in distributed energy networks?*

This thesis investigates several aspects related to this research question. Firstly, this thesis puts forward a framework to characterize gossip protocols. This framework is used to differentiate several directions in the domain of gossip protocols, and to discusses multiple different gossip protocols proposed in literature.

Secondly, in this thesis, unidirectional gossip is used to investigate how the design of the communication protocol and network properties affect the convergence speed

of a network. For this purpose, we propose two gossip protocols based on Weighted Gossip (WG) and evaluate their performance in random geometric graphs (RGGs) with different radii. These gossip protocols attempt to improve convergence speed by applying two forms of prioritization: degree-based diffusion matrix shares and degree-based neighbor selection. Both of these protocols show equal or worse performance than standard Push-Sum (PS). However, the analysis of the individual convergence speed of nodes with respect to node degrees gives valuable insights in the operation of gossip protocols. From this analysis we conclude that lower-degree nodes converge significantly slower than higher-degree nodes, which is also the reason why the proposed protocols converge slower.

In addition, this thesis uses a second approach to further investigate the influence of diffusion matrix shares and the influence of node degrees on the convergence speed. For this investigation a conceptual gossip protocol is proposed. This is a modification of PS in which different static shares are applied. This protocol is evaluated with multiple static shares in path graphs, RGGs, fully-connected graphs, and $k$-regular graphs. The major result from this experiment is that the static share $\frac{1}{2}$, as used in PS, is not necessarily always leading to the fastest network convergence speed in sparsely-connected networks. Besides, from the experiment with $k$-regular graphs we conclude that degree centrality is insufficient for the purpose of improving convergence speed.

Lastly, this thesis also investigates the influence of the variation of input value set, by applying several input sets that are generated randomly, using normal distributions with different variations. The experiments show that a larger variation in the input value set results in larger convergence times.

The contributions of this work can be summarized as follows. Firstly, this thesis establishes a framework to classify gossip protocols. Secondly, this thesis helps to understand the operation of unidirectional gossip protocols, and provides a deeper insight in how the diffusion matrix shares influence the convergence speed of a network. Furthermore, this thesis implies that degree centrality is not sufficient to improve the convergence speed of a network. Lastly, this thesis identifies significant factors that affect the convergence speed of a network, which are important for the design of communication mechanisms operating with timing restrictions in future distributed energy networks. These factors are the network density, the expected variation of the input value set and the correlation with the optimization algorithm.

# Contents

# List of Figures

# List of Tables

# Acronyms

**DER** Distributed Energy Resource. 1, 2, 3, 4, 13

**DES** distributed energy storage. 17

**DG** Distributed Generation. 2, 3, 17

**DR** Demand Response. 4, 20, 21

**DSM** Demand-Side Management. v, 4, 5, 6, 11, 12, 13, 19, 20, 22, 73, 118, 119, 121, 124

**DWG** Degree-based Weighted Gossip. x, xii, 70, 77, 78, 80, 82, 83, 84, 85, 86, 88, 89, 90, 91, 92, 93, 120, 131

**DWG*** Degree-based Weighted Gossip with degree-based neighbor selection. x, 77, 82, 83, 85, 86, 88, 89, 90, 91, 92, 93, 131

**EMS** energy management system. xii, 17, 18, 19, 20

**EV** electric vehicle. 2, 4

**HV** high-voltage. 3

**LV** low-voltage. 3, 123

**MG** microgrid. 3, 14, 17, 18, 19, 20, 21, 22, 118, 123

**MV** medium-voltage. 3

**P2P** peer-to-peer. 10, 20, 21

**PS** Push-Sum. vi, xi, xii, 21, 22, 28, 30, 33, 34, 36, 37, 40, 45, 48, 52, 55, 61, 67, 70, 74, 80, 82, 83, 85, 86, 87, 88, 89, 90, 92, 93, 94, 95, 96, 118, 120, 121, 122

**PS*** Push-Sum with degree-based neighbor selection. x, xii, 77, 80, 82, 83, 84, 85, 86, 88, 89, 90, 91, 92, 93, 120, 131, 132

**PV** Photo-Voltaic. 1, 3

**RES** renewable energy source. 1, 2, 4, 17

**RGG** random geometric graph. vi, xi, xii, 15, 53, 54, 65, 70, 77, 81, 82, 83, 84, 85, 86, 87, 88, 89, 92, 95, 97, 99, 100, 101, 102, 104, 105, 106, 107, 108, 111, 112, 113, 114, 115, 116, 117, 120, 121, 131, 132

**SG** Smart Grid. 4, 5, 6, 9, 11, 21, 22, 23

**SWG** Static-share Weighted Gossip. xii, 95, 96, 97, 98, 99, 100, 101, 102, 104, 105

**WG** Weighted Gossip. vi, 15, 29, 30, 36, 37, 38, 40, 45, 52, 55, 56, 57, 59, 60, 62, 78, 80, 81, 96, 119

# 1 Introduction

## 1.1 Energy management

For a long time, the most significant supply of electricity came from large power plants, where fossil fuels like gas and coal are used to produce electricity. Most of the current-day electricity grids are designed around these centrally located power plants as the main electricity supplier. The production of electricity in these power plants can be adjusted according to the (expected) demand of the electricity users. This demand-driven energy distribution has resulted in a stable energy grid for a long amount of time.

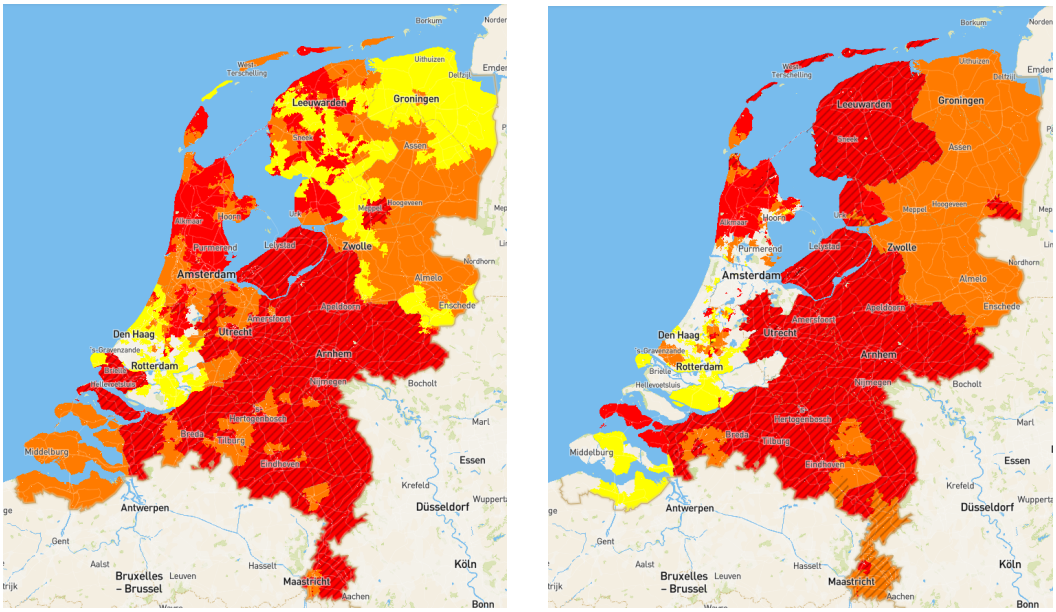
With increasing awareness of global warming and climate change in recent years, a focus on sustainability has become the standard in many sectors. It has become a strong motivator for research, industry, and governmental activities. In addition, geopolitical movements have raised the awareness of energy security, resulting in governments striving to become more self-sufficient regarding energy production and distribution. This has made sustainability a driving force in policymaking. As a result, a worldwide energy transition has been put in motion, with the aim to reduce the emission of greenhouse gasses. This is done by means of the reduction of fossil fuel consumption, and the stimulation of renewable energy such as solar energy, wind power, and hydropower in order to meet the energy demand.

### 1.1.1 Decentralization of the grid

The increasing deployment of renewable energy sources (RESs) has led to increased distributed energy generation. Various reasons are core to this development. For example, technical innovations, governmental stimulation, and decreasing prices of RESs, such as Photo-Voltaic (PV) panels and wind turbines, contribute to the increasing deployment of RESs. Distributed Energy Resources (DERs), the devices that produce, consume, or store energy, are increasingly being adopted by households and non-residential users. As a result, electricity is no longer only generated at central power plants, but is also generated on a local level. In other words, we see a decentralization of the electricity grid. Part of the electricity generation

from centralized power plants is shifted towards areas close to the end-user. This development is referred to as Distributed Generation (DG).

Unfortunately, the deployment of RESs also brings challenges, one being the dependence on weather conditions. For example, the sources sunlight and wind are intermittent and variable. The electricity generation from these sources depends on the weather, time of day, and season of the year, which makes the energy production variable and uncertain. As the generation from these sources cannot be controlled in the same way as fossil-based power plants, energy generation is becoming less flexible. In addition, the electricity demand increases due to worldwide electrification as a result of the energy transition. Moreover, the moments of high production from renewable sources do not always align with the moments users want to consume the electricity. An electric vehicle (EV) owner might for example want to charge during the night, while solar panels supply energy during the day. This supply-demand mismatch results in higher peaks in the power profile of the electricity grid, which is becoming a problem for conventional grids.




**(a)** Grid congestion on the demand side.

**(b)** Grid congestion on the supply side.

**Fig. 1.1:** Grid congestion for large-scale power connections in the Netherlands (7 December 2023) [1].

Most of current-day electricity grids are designed with predictable production and consumption in mind. However, with higher penetrations of DERs and the variability and uncertainty of electricity generation, the conventional grid is pressured, and the limited capacity becomes an issue. This overloading of the electricity grid is known as grid congestion. Grid congestion can cause asset overloading, voltage fluctuations

in the grid, and even grid blackouts in some cases. In other words, the electricity grid becomes less reliable and unstable due to the increased power demand and DG. Consider the current situation of grid congestion in the Netherlands in Fig. 1.1. The red and orange zones refer to 'No transport capacity available' and 'No transport capacity available for the moment' respectively. For these zones, no new or upgraded grid connections are possible, preventing industry from growing or meeting the sustainability goals. The largest part of the Netherlands being colored red or orange emphasizes how critical and urgent the current situation of grid congestion is.

### 1.1.2 Microgrids

With the decentralization and diversification of the electricity grid, the management of energy distribution becomes more complex. That is why it is meaningful to divide the grid into separable parts for the analysis of the operation of the grid. This is where we introduce the concept of microgrids (MGs). In literature, multiple definitions for microgrids are used. According to [2] microgrids are defined as:

> *"Microgrids comprise LV distribution systems with Distributed Energy Resources (DERs) (microturbines, fuel cells, PV, etc.) together with storage devices (flywheels, energy capacitors and batteries) and flexible loads. Such systems can be operated in a non-autonomous way, if interconnected to the grid, or in an autonomous way, if disconnected from the main grid. The operation of microsources in the network can provide distinct benefits to the overall system performance, if managed and coordinated efficiently."*

In other words, the grid can be considered as a collection of separable units, which we call MGs. An MG can operate autonomously (islanded mode), and has the ability to interact with the main grid (interconnected mode). From the point of view of the main grid, an MG can be considered as a single controllable entity [3]. Because an MG consists of a subset of controllable entities (DERs), concepts applying to MGs are potentially also applicable to higher levels of the distribution grid, such as medium-voltage (MV) or high-voltage (HV) grids. For instance, an MV grid containing multiple MGs, which act as single controllable entities, is similar to an MG containing DERs. This scalability potential makes the concept of MGs a useful tool for the development of energy distribution approaches.

### 1.1.3 Demand-Side Management

An apparent solution for solving grid congestion is grid reinforcement. While it seems inevitable to extend the grid's capacity, it is not a complete solution, especially not on short-term. Grid reinforcement is a costly and time-consuming operation. In addition, a shortage of skilled people is also a limiting factor in the realization of a reinforced grid. As a result, the pace of reinforcing the grid is unable to meet the growing power supply and demand. Therefore, other ways of solving the congestion problem are necessary. One of them is reducing the supply-demand mismatch, which is more promising to form a solution to the grid congestion problem.

As said before, the proliferation of RESs significantly contributes to the supply-demand mismatch because it decreases the flexibility on the production side due the dependence on the weather. However, with the deployment of DERs, more flexibility arises on the demand side. The reason for this is that the usage of DERs often comes with some inherent flexibility. For example, an EV might be plugged in at night and must be charged before the next morning. It does not matter when exactly the EV is being charged and at what power, as long as the EV is fully charged when the owner needs to use it again. Another example is the commissioning of battery storage systems. Generated energy that is not directly used could intermediately be stored and fed back to the grid when needed. This flexibility can be exploited to improve the matching of supply and demand, which results in flattened peaks in power consumption and effectively balances the grid. This utilizing of the flexibility on the demand-side of the energy distribution is known as Demand-Side Management (DSM) or Demand Response (DR) [3]. Many approaches to DSM can be found in literature, which we discuss in Section 2.1.

### 1.1.4 Smart Grids

To effectively coordinate the matching of supply and demand, knowledge of intended consumption and production is needed. As opposed to conventional grids, active monitoring and controlling of grid assets is needed to exploit the flexibility on the demand side. This is where the Smart Grid (SG) concept is introduced. The International Energy Agency (IEA) defines a Smart Grid as [4]:

> *"A Smart Grid is an electricity network that uses digital and other advanced technologies to monitor and manage the transport of electricity from all generation sources to meet the varying electricity demands of end-users. Smart Grids co-ordinate the needs and capabilities of all generators, grid*

*operators, end-users and electricity market stakeholders to operate all parts of the system as efficiently as possible, minimising costs and environmental impacts while maximising system reliability, resilience and stability."*

The fundamental principle of an SG is the incorporation of an ICT layer in the grid infrastructure. This layer facilitates the communication between different actors in the network, and can be used to collect measurements from the grid or to manage grid assets. The communication between actors in an SG is an essential part of the implementation of DSM in SGs and is the main topic of this thesis.

## 1.2 Communication network

### 1.2.1 DSM elements

DSM approaches typically consist of three inter-dependent elements, which are the *1) optimization algorithm*, *2) operative structure*, and *3) communication network*:

1. *optimization algorithm:* the optimization algorithm that is used to obtain the DSM objective. For instance, congestion solving could be the DSM objective. Examples of optimization algorithms include the gradient method, the push-sum method, the fast gradient method, the alternating direction method of multipliers, average consensus, and the distributed proportional integral control algorithm [5].

2. *Operative structure:* the way different roles are assigned to nodes in a network and the associated tasks of these roles. Consider for example a hierarchical operative structure: In such a network, parent and child nodes exist, where the parents process information from their child nodes, and a child node acts based on instructions it receives from its parent.

3. *Communication network:* the way information is exchanged between participants of a network. This entails the dissemination of information through a network, but also the interconnection of nodes. Nodes could, for example, be limited to communicating with 1-hop neighbors, or communicate with any node in the network by allowing message relaying.

The investigations of this thesis take place in the area of the communication network. Network convergence is important in distributed networks. This is discussed in the following section.

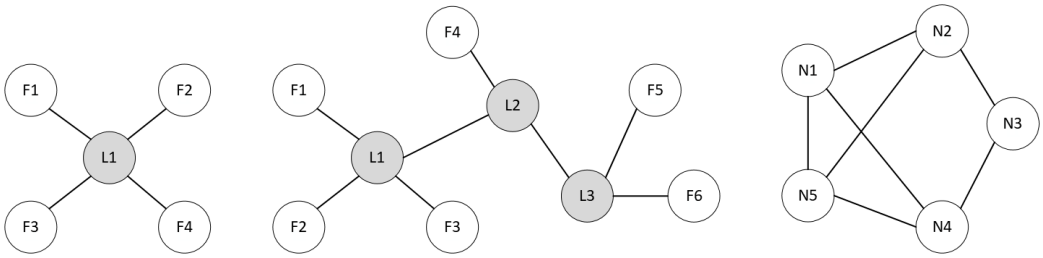
### 1.2.2 Network convergence

A communication network facilitates the implementation of DSM in a SG. Such a network consists of a number of participants that are called nodes. Nodes represent autonomous operating entities in a network which, in the communication context, mostly contains information exchange with other nodes and local processing of information. These nodes in the network must cooperate to reach a common objective, which we refer to as the global objective. Global objectives of DSM could for instance be load balancing or maximizing economic efficiency due to mutual power exchange. Each node in the network has access to local information, which could for example be a local measurement. For convenience, we refer to this local information as the local value of a node. Note that this could easily be interchanged by a vector of values. A requisite to perform the steering of a network towards the global objective (achieving the DSM objective) is that each node in the network has the same view of the network, i.e. each node has the same knowledge. If, for example, the global objective is to balance the load on the network, each node should at least know the expected global load profile of the network before it can be optimized. Hence, a mechanism to form a global view of the network, based on local available information at individual nodes, must be in place. This process is known as network convergence, which is essentially the convergence from partial (local) views of the network towards a global view of the network. A network is converged when all nodes have the same knowledge of the network, i.e. all nodes have obtained the global view. Convergence to a global view of the network is part of the communication network, while network steering towards a global objective is part of the optimization algorithm. This research does not necessarily focus on the optimization algorithm, but rather on network convergence.

### 1.2.3 Network architectures

A network architecture refers to how different nodes in the network are interconnected, and thus what nodes are able to directly communicate with each other. This section discusses three important architectures: centralized, decentralized, and distributed networks. Fig. 1.2 provides visualizations of these architectures. For the discussion of the different architectures, we distinguish two types of nodes that have a different role: coordinator nodes and follower nodes. Coordinator nodes have the task of receiving and combining data from other nodes and coordinating their followers. This means that the follower nodes act upon instructions received from coordinator nodes. Furthermore, two tasks are important in a communication

network, namely the storage of data and the aggregation of local values. A proper mechanism to store data in the network is needed because the latest version of certain data must be available to current nodes, as well as new nodes joining the network. The aggregation can be seen as a summarizing action, such as an average, sum, or maximum of local values. In essence, the aggregation of data is the formation of the global view. Section 1.3 treats the topic of data aggregation in more detail. The remainder of this section discusses how these roles and tasks apply inside the different network architectures.




**(a)** Centralized **(b)** Decentralized **(c)** Distributed

**Fig. 1.2:** Overview of network architectures.

### 1.2.4 Centralized networks

A centralized network architecture is a network built around one central coordinator node (Fig. 1.2a). This central coordinator acts as a fusion center, meaning that the local values from all the follower nodes are sent to the coordinator. The coordinator does all the computations to calculate the group aggregate. Once this aggregate is computed, the follower nodes can retrieve it from the central coordinator. When all nodes retrieved the group aggregate, the network converged because all nodes have the same (global) view. In a centralized network, most of the data is stored at the central coordinator, and only communication between a follower node and the central coordinator exists.

A major problem of this network architecture is the single point of failure it creates. In case the central coordinator goes down, the complete network will become dysfunctional due to the high dependence on the central coordinator for data storage and communication. Besides, a single coordinator node will also become a communication bottleneck in the network, because this node is involved with all communications. This becomes a major issue in especially large-scale networks, because the bandwidth needed at the central coordinator scales with the number of nodes in the network.

### 1.2.5 Decentralized networks

An alternative network architecture is a decentralized network (Fig. 1.2b). The concept of a decentralized network is similar to a centralized network, but in this architecture, the coordination is performed by multiple coordinator nodes instead of one. Each coordinator node has a set of follower nodes. The follower nodes send their local value to their coordinator node, which calculates the group aggregate. Subsequently, all the coordinator nodes share their group aggregate with each other so that each coordinator node can obtain the global view by combining all group aggregates. After this exchange, all follower nodes can retrieve the global view from their coordinator nodes. Because the data storage and the communication are now divided over multiple coordinators, the single point of failure and communication bottleneck problems are reduced compared to centralized networks.

However, decentralization is not a complete solution for the single point of failure problem, because the failure of one coordinator node can still result in a partial blackout in the network. In addition, much communication is still going through the coordinator nodes, so the number of follower nodes connected to a coordinator is limited in order to prevent communication bottlenecks. A possible solution is scaling up the number of coordinator nodes, but this will result in additional communication overhead between coordinator nodes. Hence, there is always a trade-off to be made between the number of coordinators and convergence speed.

In conclusion, a decentralized architecture can make a good improvement in reducing the impact of single point of failures and communication bottlenecks. However, a decentralized architecture can only solve these problems partially.

### 1.2.6 Distributed networks

The third alternative, the distributed network architecture (Fig. 1.2c), offers a complete solution for the described problems of centralized networks. The hierarchy as seen in centralized and decentralized networks differs from the hierarchy in a distributed network; all the nodes in a distributed network are placed on the same hierarchical level. The distributed architecture has no pure coordinators or pure followers. The nodes in a distributed network all fulfill both a coordinating role and a following role. As a consequence, communication between any set of nodes is possible, as long as nodes are connected. This is the reason why a distributed network architecture does not contain a single point of failure. In case one of the nodes goes down, the other nodes are able to continue network operation because

they are not solely depending on individual nodes. Furthermore, the distributed architecture is resistant to communication bottlenecks because nodes communicate with various other nodes. This is beneficial, as it makes the network robust, reliable, and scalable.

However, making a network distributed also comes with challenges. First is the challenge of data storage. Where almost all data is contained in the coordinator node in the centralized architecture, the way of storing data in a distributed network is not directly easily solved. A distributed network does not have a central node and therefore leaves the question of where and how the data in the network should be stored. Two commonly used possibilities of data storage in distributed systems are data sharding and data replication. With data sharding, data is split into smaller pieces and distributed over the participants of the network. In other words, nodes become responsible for the storage of a particular part of the data. The other possibility is the replication of data. This entails storing copies of the complete data set in each node of the network. Also, a combination of these data storage possibilities exists. Both data storing techniques have their advantages and disadvantages.

Second is the challenge of network convergence. In larger networks, individual nodes might lack a complete view of the network and the partial views of nodes might vary from node to node. The network must converge to a shared global view. In addition, the convergence speed is also an important aspect, next to the correct convergence of a network. Since the proper operation of SGs has certain requirements on timing, a distributed network should converge in time. An interesting consideration is the needed accuracy of network convergence. For example, the optimization algorithm might still operate properly if the deviation of the partial views of nodes stay within certain limits. Hence, the trade-off between convergence speed and accuracy is an interesting subject.

Another challenge is uncertainty and variation in a network. New nodes might join the network, or nodes might disappear from the network at any time. In addition, messages can get lost or nodes can crash at random times. Hence, a communications approach is required to achieve network convergence in distributed networks despite network disruptions.

Concluding, a distributed network architecture is well-suited to create robust, reliable, and scalable communication networks. This architecture offers a promising solution for the single point of failure and communication bottleneck problems, seen by centralized and decentralized networks. That is why this research takes a distributed approach to the implementation of a communication network.

## 1.3 Gossip-based communication

An attractive type of communication suitable for distributed network architectures that recently has received a surge of attention, is the family of gossip algorithms [6]. Gossip algorithms, also called epidemic or rumor-spreading protocols, are based on how information spreads through a community of gossiping humans. Gossip algorithms have several interesting properties for distributed systems: simplicity, speed, robustness, a lack of central management, and the lack of bottlenecks [7]. This is why gossip protocols can play a critical role in realizing robust, scalable, and self-operating energy distribution networks in the future.

The important applications of gossip communication inside a network are *information dissemination* and *data aggregation*. Information dissemination can be characterized as spreading information through a network by communicating between nodes. A good example where information is disseminated using gossip-based communication, is the distributed database system Cassandra [8]. In this peer-to-peer (P2P) network, nodes gossip with each other. In a gossip between two nodes, the nodes check how up-to-date their database records are. Subsequently, the latest versions of database records are exchanged between the two nodes so that both nodes end up with the newest information from each other [9, 10].

Data aggregation is an extension of the information dissemination application. Not only information is spread through the network, but the information is also processed 'on the fly'. Aggregation can be seen as a summarizing action that is performed on data. This could for example be calculating an average, a maximum, or extracting some attribute from nodes. Even more complicated functions, like fitting a model on data, are possible. With data aggregation, each gossip interaction consists of a data exchange between two nodes, followed by a local processing step. The processed information is exchanged in the next gossip interaction with another node. The processing step is calculating the aggregate of the data from both nodes. This iterative process of exchanging data and computing aggregates leads to a network collectively moving towards computing a global aggregate. The global aggregate could for example be a network-wide average of local measurements. This movement towards the state of the network where each node has the same and the correct value of this aggregate is what is explained in Section 1.2.2 as network convergence.

## 1.4 Research definition

This section defines the research. Section 1.4.1 and Section 1.4.2 discuss the research area and the scope of the research. Subsequently, the research questions are presented in Section 1.4.3. Section 1.4.4 explains the used research approach to answer the research questions. This introductory chapter is concluded by giving the outline of this thesis in Section 1.5.

### 1.4.1 Research area

This research is done approached from an energy management perspective. The electricity grid and the communication network become more and more intertwined with SGs, which is likely to become a significant technology used in future energy management. As a result, the proper operation of the electricity grid also becomes dependent on the communication network. The centralized and decentralized network architecture have its drawbacks with the central points of failure and communication bottlenecks. That is the reason why this research investigates network convergence, and follows an approach that uses distributed network architectures and gossip-based communication.

### 1.4.2 Research scope

Section 1.2.1 introduced and explained three structural elements of DSM approaches: The optimization algorithm, operative structure, and communication network. This thesis focuses on the communication network, and more specifically on network convergence. The optimization algorithm and operative structure are closely related, but out of the scope of this research. We investigate the operation of gossip protocols in distributed network architectures, which inherently have an operative structure where there is no hierarchy and each node fulfills both a coordinating and following role.

An interesting topic in the domain of gossip protocols is the matter of how individual nodes know when the network has converged to the global view. Nevertheless, this is out of scope of this research. This research analyzes network convergence from a network point of view, and assumes that in a real-life situation some mechanism is in place that enables nodes to determine whether the network has converged, if this required for their operation. In some cases nodes do not need to know whether the

network has converged. The reliability of their view of the network just increases during the gossip process, which could be sufficient for the proper operation of the optimization algorithm.

In addition, this research makes the following explicit assumptions:

- All nodes in the network can be trusted. This means that all nodes behave according to a defined algorithm and cannot deviate from this.
- The network is connected, i.e. each node in the network can be reached directly or via other nodes.
- Communication is synchronous. This means that the used communication algorithms contain rounds with simultaneous communication between nodes.
- At the initial state of the network, the network topology is known. This means that nodes know their direct neighbors from the beginning. Initial network discovery is out of scope.

### 1.4.3 Research questions

The following research question is central to this thesis:

> *How can communication mechanisms and network properties be utilized to improve convergence speed, to aid the implementation of Demand-Side Management in distributed energy networks?*

To form an answer to the main research question, several relevant topics will be investigated. These topics are guaranteeing network convergence, algorithm design, network topology, and the input value set.

Firstly, as discussed in Section 1.2.2, the application of Demand-Side Management in a distributed network requires a network state in which each node has the same knowledge about the network: the global view. To execute an optimization process, such as load balancing, the nodes must know the initial global load profile of the network. Guaranteeing this convergence from a partial view of the network towards a global view of the network is a central problem in distributed communication networks. Hence, the following question must be answered:

1. *How could a network converge from different partial views to a shared global view of the network?*

Secondly, the rate at which a network converges is a performance metric often as important as guaranteeing network convergence. Real-life networks, like energy networks, need to find a solution to a certain problem within a certain time period, which puts requirements on the convergence speed of algorithms. For example the planning of DERs production and consumption in DSM, must be finished before prosumers start executing the planning. The design of the communication algorithm can play an important part in the convergence speed of a network. Hence, the second subquestion focuses on the design of algorithms:

2. *How can the convergence speed of a network be improved by the design of communication algorithms?*

Thirdly, real-life networks will often not be fully connected (this means that each node does not have a 1-hop connection to every other node in the network, see Section 3.2), and network topologies vary between different use cases of energy networks. According to [11], communication topology is a factor influencing the convergence speed of a network. We hypothesize that the graph density might be a significant factor for the convergence speed in a network. Therefore, this research investigates how the density of a graph relates to the convergence speed. The related subquestion for this is:

3. *How does the density of a network relate to the convergence speed of the network?*

The fourth topic targets the input values given to the communication algorithm. The variation in values of an input set might affect how fast a network converges. If properties of this input set relate to the convergence speed, this can be helpful information in the estimation of convergence times in energy networks. Hence, the fourth question is:

4. *How does the variation in input values influence the convergence speed of a network?*

Up to now, the subquestions are formulated in a general sense to make the findings of these subquestions generally applicable. However, this research is approached from an energy management point of view. This means that it is necessary to make explicit how the outcomes can be used in the implementation of DSM in distributed energy networks. This formulates the last subquestion:

5. *How can the results of the researched topics of guaranteeing network convergence, algorithm design, network topology, and the input value set be used in the implementation of Demand-Side Management in distributed energy networks?*

### 1.4.4 Research approach

This thesis uses the following approach to form answers to the main research question and the subquestions. Firstly, Chapter 2 discusses related literature for this research. Subsequently, Chapter 3 introduces a framework to characterize different gossip protocols and explains a number of fundamental gossip protocols and variations of them in Chapter 3. In this discussion, the advantages and disadvantages of different gossip protocols become clear, and we explain how network convergence can be guaranteed. One of the gossip protocols is selected to be used in the experiments of this thesis, which is discussed in Section 3.1.9. Chapter 4 contains an analysis of individual convergence in small-scale scenarios using the selected gossip protocol. This analysis is used to gain an intuition for how the convergence speed can be improved by the design of the gossip protocol, and forms a basis for the experiments with larger networks, which is the topic of Chapters 5, 6, 7, and 8. The general simulation setup that is realized to perform the different experiments is presented in Chapter 5. Chapter 6 and 7 both investigate the influence of algorithm design and network architecture on the convergence speed of a network. Chapter 8 focuses on how the variation in the input value set affects the convergence speed of a network. This thesis is concluded with a general discussion, conclusion and future work in Chapter 9.

## 1.5 Thesis outline

**Chapter 2** discusses literature about energy management that is relevant for this thesis. Multiple works on energy management approaches and microgrids are discussed. This chapter ends with a number of cases where gossip-based communication is applied in an energy management context.

**Chapter 3** elaborates on the wide variety of existing gossip protocols. First a definition, terminology, and framework to classify gossip protocols are given. Subsequently, multiple fundamental gossip protocols and variations to these protocols are explained, in order to present an overview of existing gossip protocols.

**Chapter 4** contains an analysis of three small-scale scenarios to investigate diffusion matrix shares and how these influence the individual convergence speed of nodes. The analysis shows that there is optimization possible, and this is the basis for the experiments with larger networks in Chapters 6 and 7.

**Chapter 5** presents the used research methodology for the experiments with larger networks. This chapter describes the requirements and the design of the general simulation setup. Furthermore, the implementation architecture is presented and specifics about how the simulations are executed and evaluated are given.

**Chapter 6** focuses on the topics of algorithm design and network density, and how they affect the convergence speed. Chapter 6 proposes two algorithms based on Weighted Gossip that both use a form of prioritization based on node degrees, to attempt to improve the convergence speed of a network. These protocols are evaluated in random geometric graphs (RGGs), with different network densities.

**Chapter 7** continues the investigation of algorithm design and network topology, but uses a different approach. Chapter 7 uses $k$-regular graphs, RGGs, path graphs, and fully connected graphs in combination with so-called static shares to obtain a better understanding of the influence of the diffusion matrix shares and the network topology.

**Chapter 8** focuses on how the variance of the input values of the gossip protocol affect the convergence speed of the network. The experiment uses normal distributions with different standard variations to evaluate the convergence speed in RGGs.

**Chapter 9** concludes this thesis by discussing the limitations and implications of the research, deriving conclusions from the findings of this research and offering directions for future research.

# Related work

# 2

This chapter discusses relevant literature for this thesis. Firstly, research focused on microgrids is discussed in Section 2.1, followed by several studies that contain different energy management approaches in Section 2.2. Finally, Section 2.3 discusses studies that apply gossip-based communication in an energy management context.

Related work about gossip protocols is discussed in Chapter 3. The reader can refer to Section 3.1 for an elaborate discussion and explanation of several types of gossip protocols that are related to this thesis.

## 2.1 Microgrids

The microgrid concept has become an integral subject in the research and development of energy grids. Su and Wang [12] state that a microgrid is a promising technology that improves the reliability and economics of energy supply. Furthermore, they elaborate on several aspects of microgrids including MG components, functionalities of MGs, architecture and control philosophies, and expected future trends in MGs control and energy management systems (EMSs). Next to the opportunities MGs offer for dynamic energy supply (DG and distributed energy storage (DES)) and better utilization of renewable energy sources, Su and Wang also point out the need to include spatial and temporal requirements in the design of EMSs for microgrids. Besides, communication is a critical aspect of MGs, and one should consider among others requirements on the reliability of communication, network latencies, time synchronization, and cybersecurity.

In energy management systems for MGs, three main control architectures can be distinguished: centralized, decentralized, and distributed [13]. In a centralized EMS, the energy grid is managed from one central point in the network. The decentralized EMS is not coordinated from a single point, but relies on multiple local coordination points. These local coordinators contain intelligence to make operational decisions for a part of the MG. In the distributed architecture, there are no central nodes and there exists no hierarchy between nodes. The nodes in the network must cooperate

to manage the energy flows properly. These three architectures have advantages and disadvantages, as summarized in Table 2.1.

**Tab. 2.1:** Overview of advantages and disadvantages for centralized, decentralized and distributed control of EMSs, based on [12, 13].

| Centralized control | |
|---|---|
| **Advantages:**<br>• Low implementation difficulty.<br>• Easy maintainability.<br>• Widely used and operated.<br>• Wide control over the entire system. | **Disadvantages:**<br>• Single point of failure.<br>• Bandwidth scales with network size.<br>• Computational burden (central node).<br>• Low flexibility/expandability.<br>• Low reliability.<br>• High computational costs. |

| Decentralized control | |
|---|---|
| **Advantages:**<br>• Reduced impact of single points of failure.<br>• Higher flexibility/expandability.<br>• Moderate computational burdens at local controllers.<br>• Suitable for large-scale systems.<br>• Potentially most robust.<br>• Potentially fastest. | **Disadvantages:**<br>• Local controllers are single points of failure for subparts of the network.<br>• Higher implementation difficulty.<br>• Synchronization is needed between the local controllers.<br>• May need more time to reach a consensus.<br>• Convergence rates are affected by communication network topology. |

| Distributed control | |
|---|---|
| **Advantages:**<br>• No single point of failure.<br>• High reliability.<br>• Highest flexibility/expandability.<br>• Scalable.<br>• Low computational burdens.<br>• Robust. | **Disadvantages:**<br>• High implementation difficulty.<br>• May need more time to reach a consensus.<br>• Convergence rates are affected by communication network topology. |

A recent survey paper by Raya-Armenta *et al.* [13] presents a clear overview of the current state of the art of EMS optimization in islanded microgrids. This work includes a discussion of various optimization frameworks, optimization algorithms,

and future prospects. One of the statements in the paper related to architecture is that optimization frameworks based on game theory and multiagent systems have been shown most effective in achieving a decentralized configuration. Nonetheless, some problems like high complexity, high reliance on conventional distributed energy resources (like diesel generators), or always requiring a communication link, are still unsolved. A reliable and fully effective decentralized EMS for islanded MGs has yet to be developed.

There is an interesting contradiction between the work of Su and Wang [12] and Raya-Armenta *et al.* [13]. Su and Wang state that in a decentralized configuration, it might be more time-consuming for local nodes to reach consensus, which holds equally true for a distributed configuration. By contrast, Raya-Armenta *et al.* claim that it is well-known that a distributed configuration can be faster compared to a centralized architecture and that a decentralized configuration is potentially the fastest. Unfortunately, supporting evidence for this statement is not given. However, this contradiction does point out that it is not directly clear which of the configurations is fastest in solving a problem. This probably depends on the convergence speed of a distributed network, together with the improved computational costs due to the possibility of nodes to compute aggregates in parallel. Hence, the area of convergence speed of distributed or decentralized networks is an interesting topic for research.

## 2.2 Energy Management Systems

A lot of research is done focusing on the development of energy management systems and Demand-Side Management. The three control structures discussed in the previous section, return in different approaches to DSM. For instance, the centralized EMSs are applied in [14, 15], decentralized approaches for local congestion control can be found in [16], a game-theory-based DSM is presented in [17], and [18] focuses on a distributed DSM algorithm.

Other examples of work related to distributed and decentralized operations in energy networks is the work by Mendel, mostly focusing on the security of Supervisory Control and Data Acquisition systems and underlying communication. This includes research on distributed monitoring of neighborhoods of field stations [19], local intrusion detection systems based on power-flows of the electricity grid [20], and the evaluation of these intrusion detection systems on distributed hardware [21].

Another study presents a fully distributed P2P-architecture for DR control in a community of smart buildings that contain energy generation and storage capabilities [22]. The implemented communication in this work is gossip-based.

The work from Pappu *et al.* [23] is a related study to our research, focusing on the communication layer in distributed energy networks. This work uses blockchain concepts Proof of Work and Proof of Stake as consensus mechanisms to implement DSM in a distributed way. Pappu *et al.* use a DSM optimization algorithm called Profile Steering, as proposed by Gerards *et al.* [15]. While the Profile Steering algorithm normally involves a hierarchical operational structure with central coordinators, Pappu *et al.* [23] implement it in a fully distributed manner. The study deals with reaching consensus in a network, even if each node cannot be trusted by default. The network is assumed to be fully connected, known, and reliable, and no node failures appear. Our research is different because it focuses on network convergence, it considers multiple network types other than fully connected networks, and it assumes that there is no malicious behavior inside the network.

The works mentioned in the previous paragraphs outline the application field of this research and form a background for this research. While much research focuses on the intelligence part of EMSs, the communication side and topology of the networks are also important aspects. Especially in a distributed architecture, communication is an important factor in the performance and robustness of the EMS.

A source that also emphasizes the importance of network topology for distributed optimization is the work of Nedić *et al.* [11]. The authors provide an overview of state-of-the-art multiagent optimization methods. They discuss several relevant topics in decentralized optimization, such as network architectures; optimization in directed, undirected, and time-varying graphs; and the combination of optimization and averaging algorithms.

## 2.3 Gossip communication in the energy context

Gossip-based communication has also been applied in communication networks for energy management. Early work is contained by Brabandere *et al.* [24] where gossip algorithms are applied in the control of microgrids in a fully distributed way. The study defines three levels of control: primary control to ensure reliable local operation; secondary control for voltage and frequency profile optimization; and tertiary control for economic optimization, i.e. power can be exchanged between the peers in the network at equal marginal cost. The control algorithms on all

these levels operate in a completely distributed way. From these, the control levels requiring communication between the nodes, secondary and tertiary control, are implemented using gossip protocols. However, specifics on how gossip communication is implemented in this research are missing.

A study that focuses more on gossip communication itself and its application in future power systems, is the work from Krkoleva *et al.* [25]. This paper is mostly explanatory and shows a very basic approach to gossip communication. Other work from the same authors concentrates on the requirements for implementing gossip-based schemes in future power systems [26]. ICT technologies and infrastructure, which enable the deployment of gossip-based communication in power systems, are discussed.

Campos *et al.* [27] acknowledge the importance of reliable and scalable communication systems for Smart Grids, but bring up the problem of compatibility and maintainability of gossip-based P2P-communication and coordination protocols with existing systems. That is why they propose a communication framework providing gossip-based dissemination and coordination built on top of Web Services. They implement gossiping and membership management using low-resource-consuming UDP. This is possible due to the inherent scalability and reliability of gossip protocols.

In more recent work, Croce *et al.* [22] propose a fully distributed DR algorithm called 'Overgrid'. In this work, gossip-based communication is applied for multiple purposes: topology detection (heartbeat protocol), information dissemination (power demand constraints), and data aggregation (average power demand estimation). Croce *et al.* use a flow updating algorithm that guarantees fast convergence for the data aggregation. Although the study promises a good performance of the gossip protocol, and that it is suitable for the Demand Response, it is not compared to a centralized DR in the same environment. This leaves the question of the relative performance improvement of the distributed DR.

Other recent works apply gossip communication for energy management based on Push-Sum from Kempe *et al.* [28], which also has a central role in our research. Chapter 3 explains Push-Sum and related terminology like gossip rounds, diffusion matrices, weights, and shares in more detail. The work of Koukoula *et al.* [29] and the work of Engels *et al.* [30] both present a modification of the Push-Sum algorithm for distributed voltage control inside a microgrid.

Koukoula *et al.* [29] attempt to accelerate the convergence speed of Push-Sum by constructing an optimal weights' matrix. They approach the construction of the

weights' matrix as a semi-definite problem and solve it using a sub-gradient method. Furthermore, the authors show that their proposed gossip protocol is able to address overvoltages fast and effectively. The weights used by Koukoula *et al.* have similar meaning as what is referred to as 'shares' in this thesis. However, the work of Koukoula *et al.* differs from this research in a couple of ways. Koukoula *et al.* let nodes communicate with all 1-hop neighbors in one gossip round, and use a fixed and doubly stochastic diffusion matrix in all gossip rounds. Our research assumes that each node contacts only one neighbor per round, and uses a (varying) single stochastic diffusion matrix, which is different in each gossip round.

Engels *et al.* [30] adapt the Push-Sum algorithm differently. They adjust the algorithm such that each node in the network keeps track of the state variable of each individual node in the network, instead of only the node's own state variable. The authors apply the gossip communication to implement a novel voltage control algorithm that operates in a distributed manner, without centralized control. They also present a case study with rapidly changing load profiles in a microgrid and show that their algorithm is able to keep the voltage within tight limits.

In addition, the work of Bisceglie *et al.* [31] explorers the Push-Sum gossip protocol in a grid monitoring system. The authors highlight that modern trends in SGs move towards the employment of advanced monitoring architectures and cooperation between entities inside the SG. Bisceglie *et al.* propose a decentralized and non-hierarchical monitoring architecture and compare two information spreading algorithms. These algorithms are Push-Sum and a Kuramoto algorithm, which are used to let the monitoring sensors converge to the true average voltage. The authors conclude a faster convergence of the Kuramoto algorithm, but at the cost of a larger communication complexity compared to Push-Sum.

Gossip-based communication has been a recurring subject in multiple research areas, and now also found its way to the research domain of energy management. The attractive properties of gossip-based communication, such as simplicity, robustness, a lack of central management, and a lack of bottlenecks, can be beneficial for distributed network architectures in energy distribution applications. Therefore, it could become an integral part of future energy distribution systems. The optimization process in DSMs receives much attention in research, while the communication network is a less-discussed topic. This research contributes to the body of work about the communication side of energy distribution systems, by investigating how the convergence speed in distributed network architectures is affected by the design of the communication mechanism and network properties. This thesis gives deeper

insight in the convergence in different networks, and identifies important factors to be considered in future communication networks used in SGs.

# 3 Background

This chapter discusses relevant background information for this thesis. Section 3.1 presents a framework to characterize gossip protocols and discusses different types of gossip protocols. Section 3.2 briefly explains the different types of graphs that are used in this research.

## 3.1 Gossip algorithms

This section gives an overview of different types of gossip protocols and explains several variations of gossip protocols found in the literature. Firstly, a general definition and ways to characterize gossip protocols are given in Section 3.1.1. Subsequently, Section 3.1.2 introduces the terminology used to explain gossip protocols and gives an overview of the gossip protocols discussed in the Sections 3.1.3 to 3.1.8. Section 3.1.9 concludes this section and specifies which type of gossip communication is used in the experiments of this research.

### 3.1.1 Characterization of gossip protocols

A wide variety of gossip-based communication algorithms exist. Literature is not always clear about the exact definition of gossip algorithms. This research defines a gossip protocol as follows:

**Definition 3.1.1 (Gossip protocol)** *An algorithm is called a gossip algorithm if it at least adheres to the following propositions:*

- *A gossip algorithm has the aim to diffuse information through a network or reach network convergence in a distributed manner.*
- *A gossip algorithm complies with the mass conservation property (Definition 3.1.2), which guarantees network convergence.*
- *A gossip algorithm is iterative. In each iteration:*
  - *Nodes select one or a few other nodes to exchange information with.*

- *Information is exchanged unidirectionally or bidirectionally.*
- *Nodes perform some local processing with the received information (optional).*

The following set of properties are means to characterize the many gossip-based communication protocols:

- *Network objective:* information dissemination or network convergence.
- *Casting type:* unicasting, multicasting, or broadcasting.
- *Directionality:* unidirectional or bidirectional.
- *Partner selection:* deterministic, random, quasi-random, or hybrid.

**Network objective:** An important property distinguishing two broadly used types of gossip protocols is the network objective. The two common objectives are information dissemination and network convergence. While a gossip protocol is in general meant to reach a network state where each node has the same global view of the network, the definition of this global view of the network is different between these two types of protocols. In a gossip protocol to disseminate information, a node $i$ with the global view means that node $i$ knows the local information of all the nodes in the network. In case of information dissemination, we refer to this local information as a rumor. Hence, the information dissemination objective is obtained when each node in the network knows all rumors inside the network. The second network objective is network convergence. In a gossip protocol with this objective, a node $i$ with the global view means that node $i$ knows the network aggregate, which is built up from all local information. With this type of gossip, nodes have a local state variable which gets updated every time a node receives information from another node. To emphasize the difference between these two types of protocols the exchanged information between nodes is referred to differently; in the case of information dissemination nodes exchange *rumors* and in the case of network convergence nodes exchange *state variables*. Note that *rumors* or *state variables* are defined by the gossip protocol in use, but can easily represent the same initial local information.

**Casting type:** The casting type refers to how many recipients are involved in gossip interactions. We distinguish three types of casting: unicasting, multicasting, and broadcasting. A unicast is a communication between one sender and one recipient. A multicast involves one sender and multiple targeted recipients. A broadcast is a communication from one sender to one or more untargeted recipients, i.e. all the nodes in the communication range overhearing the broadcast.

**Directionality:** Directionality refers to communication going one way (unidirectional) or two ways (bidirectional). Bidirectional communication is when two nodes exchange their current values. For example, in pairwise gossip a node $i$ sends its local value to node $j$ and node $j$ directly reacts by sending its local value to node $i$. In the case of unidirectional communication, this direct reply does not occur. So node $i$ sends its value to $j$, without expecting a reply.

**Partner selection:** The type of partner selection is an important characteristic of gossip protocols. In literature, four types of partner selection are used: random, quasi-random, deterministic, and hybrid versions of selection. With a random neighbor selection, gossip partners are selected randomly, i.e. each node selects partners during runtime by making a random choice. With quasi-random neighbor selection, a list of neighbors to contact is generated randomly once at the beginning of the gossip communication. This is a cyclic list, so after the last neighbor from this list a node continues at the beginning of the list again. We say that the selection of a neighbor is deterministic when the selection is based on a set of rules, without the usage of a random choice. A hybrid neighbor selection is the combination of two or more of the selection types described above.

### 3.1.2 Terminology

For the discussion of gossip protocols, it is useful to introduce terminology to describe the operation of the protocols. In multi-agent optimization problems, a network of $N$ nodes is considered, where each node $i \in \{1, 2, ..., N\}$ is associated with some local initial value $x_i(0)$. The objective is for each node to converge to the network average $\overline{x} = \frac{1}{N} \sum_{i=1}^{N} x_i(0)$. To not overcomplicate the analysis of the protocols, we consider this local value to be one-dimensional. However, the values could easily be interchanged by multi-dimensional data, such as time-stamped vectors. During an iterative process of gossip communication, each node maintains a local variable that represents its current local view (or local estimate) of the network. Gossip algorithms consist of rounds, in which gossip interactions take place. We follow the conventional notation to denote rounds, or iterations, of gossip algorithms found in literature with $t \in \mathbb{N}$. It should be noted that $t$ is used as a discrete-time variable. This is different from the continuous variable representing time, which is often denoted with $t$ as well. The local estimate of node $i$ in round $t$ is written as $x_i(t) \in \mathbb{R}$ and is referred to as the state variable or the value of node $i$.

In the field of (synchronous) distributed algorithms, two measures of complexity are commonly used: time complexity and communication complexity [32, p. 21-22].

The *time complexity* is measured as the number of rounds until all the required outputs are produced. In the context of gossip communication, this means the number of rounds needed to let all nodes converge to the true average, i.e. the convergence time. The *communication complexity* is usually measured as the number of messages that are sent to reach the point of convergence. In practice, the time complexity is the more important measure because the communication complexity is only important if it causes enough congestion on the communication channel to slow down the execution of the algorithm. Because it is often more interesting to know how the time or communication complexity scales with the network size $N$, the big-O notation and several related notations are used to characterize the growth rates in research.

Throughout this thesis, the terms 'gossip partner' and 'neighbor' are used interchangeably. The set of neighbors that can be selected by a node differs from protocol to protocol. Hence, when the gossip partner selection is restricted to nodes having a 1-hop distance, this is referred to as 1-hop neighbors. When no further specification is used, or when the 1-hop restriction is not clear from context, 'neighbor' and 'gossip partner' can be interpreted as any node in a network.

For the analysis of gossip algorithms on a network level, we define the vector containing the state variables associated with the nodes in the network as $\boldsymbol{X}(t) = [x_1(t), x_2(t), ..., x_N(t)]^\mathsf{T}$. This means that vector $\boldsymbol{X}(0)$ contains all the initial local values $x_i(0)$, for all $i \in \{1, 2, ..., N\}$. A way to describe the interactions between nodes in one round is through a diffusion matrix $\boldsymbol{D}(t) \in \mathbb{R}^{N \times N}$. We refer to the components $D_{ij}$ with $i, j \in \{1, 2, ..., N\}$ as shares. These shares determine how the initial values are diffused through the network. In each round $t$ the local estimates in the network are updated according to:

$$\boldsymbol{X}(t+1)^\mathsf{T} = \boldsymbol{X}(t)^\mathsf{T} \boldsymbol{D}(t). \tag{3.1}$$

If the diffusion matrix meets certain properties, then some guarantees on the behavior of the gossip protocol, such as mass conservation and stable convergence, can be given (Section 3.1.3).

With the characteristics to differentiate gossip protocols and the necessary terminology introduced, we discuss some examples of gossip protocols in the next sections:

- Uniform Gossip (Section 3.1.3)
- Push-Sum (Section 3.1.3)

- Weighted Gossip (Section 3.1.3)
- Pairwise Gossip (Section 3.1.4)
- Standard pairwise randomized gossip (Section 3.1.4)
- Geographic pairwise randomized gossip (Section 3.1.4)
- Deterministic Gossip (Section 3.1.5)
- Deterministic request-based gossip (Section 3.1.5)
- Quasi-random gossip protocol (Section 3.1.6)
- Broadcast Gossip (Section 3.1.7)
- Hybrid gossip protocol (Section 3.1.8)

**Tab. 3.1:** Overview of gossip protocols and their characteristics.

| | | Uniform Gossip | Push-Sum [28] | Weighted Gossip [33] | Pairwise Gossip | Standard pairwise randomized gossip | Geographic pairwise randomized gossip | Deterministic Gossip, Liu *et al.* [34] | Deterministic Gossip, Haeupler [35] | Quasi-random gossip [36, 37] | Broadcast gossip [38] | Hybrid gossip [39] |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Networkobjective** | Convergence | | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | | | ✗ | |
| | Dissemination | ✗ | | | | | | | ✗ | ✗ | | ✗ |
| **Casting type** | Unicast | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | | ✗ |
| | Multicast | | | ✗ | | | | | | | | |
| | Broadcast | | | | | | | | | | ✗ | |
| **Directionality** | Unidirectional | ✗ | ✗ | ✗ | | | | | ✗ | ✗ | ✗ | |
| | Bidirectional | | | | ✗ | ✗ | ✗ | ✗ | | ✗ | | ✗ |
| **Partner selection** | Deterministic | | | | | | | ✗ | ✗ | | ✗ | ✗ |
| | Random | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | | | | | ✗ |
| | Quasi-random | | | | | | | | | ✗ | | |

Table 3.1 shows an overview of the characteristics of each of these gossip protocols. The characteristics presented in Table 3.1 can also be found in the frame included at the beginning of all the discussed gossip protocols in the following sections. In addition, Fig. 3.1 presents a flowchart placing the different types of gossip protocols in context to each other.

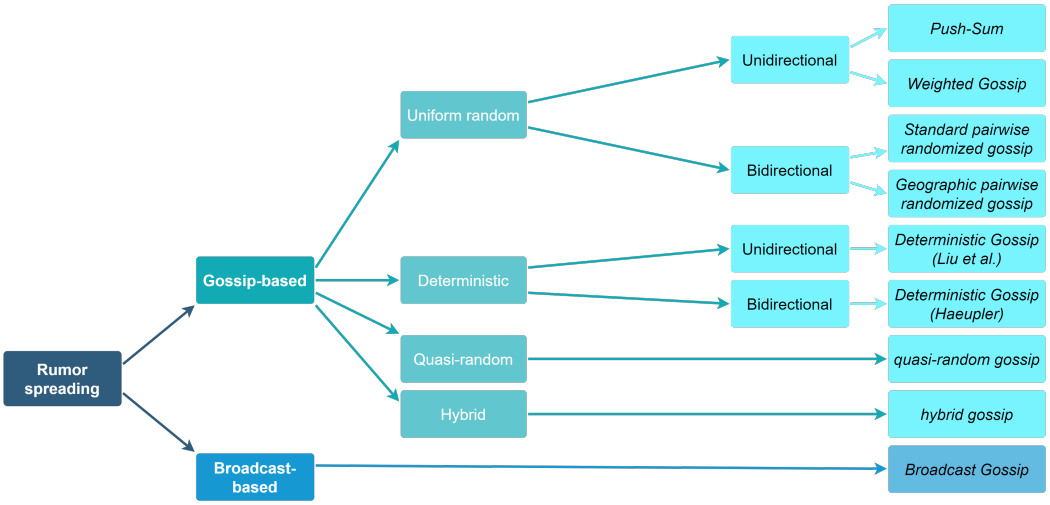



**Fig. 3.1:** Overview of discussed gossip algorithms.

### 3.1.3 Unidirectional randomized gossip protocols

Most gossip algorithms rely on randomized behavior for robustness and fast network convergence. A parallel research direction to randomized gossip protocols concentrates on the necessity of this randomization in gossip algorithms [35]. The general results of this line of research indicate that at least some randomization is needed for the efficient operation of rumor-spreading. Hence, we start the discussion with the most common version of gossip: randomized gossip protocols. This section introduces the unidirectional gossip protocols Uniform Gossip, Push-Sum (PS), and Weighted Gossip (WG).

#### Uniform Gossip

| | |
|---|---|
| **Network objective:** | Dissemination |
| **Casting type:** | Unicast |
| **Directionality:** | Unidirectional |
| **Partner selection:** | Random |

Uniform Gossip is the most fundamental gossip protocol, forming the basis for many other gossip protocols. Frieze and Grimmet [40] introduced a rumor-spreading model in 1985, which later received the name 'Uniform Gossip' and has become the common term to refer to this rumor-spreading model in literature. The model

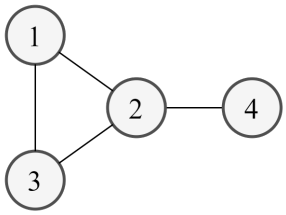



**Fig. 3.2:** Example network.

considers a population of $N$ people in which a single rumor is being spread. One person starts a rumor by selecting one other person from the population uniformly at random and informing this person about the rumor. In each subsequent step, all persons knowing the rumor will select one other person from the population uniformly at random and pass on the rumor. This continues until all $N$ persons in the population know the rumor.

The rumor-spreading model from Frieze and Grimmet considers a single rumor being spread. If node $i$ starts a rumor at time $0$, we describe this as $x_i(0) = c$, where $c$ represents the rumor (as a constant value), and $x_j(0) = 0$ for all $j \neq i$. In each round the nodes that know the rumor select another node at random to send the value to. This means that $D_{ij} = 1$ for all $i$ knowing the rumor and $j$ being the selected neighbor. In addition, $D_{ii} = 1$ indicates self-knowledge, i.e. node $i$ knows the rumor.

Consider node $1$ from the example network in Fig. 3.2 starting a rumor in Uniform Gossip and selecting node 2 in round 0. This process can be described as:

$$\boldsymbol{X}(0) = \begin{bmatrix} c \\ 0 \\ 0 \\ 0 \end{bmatrix}, \boldsymbol{D}(0) = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}. \tag{3.2}$$

Hence, after one round nodes 1 and 2 know the rumor:

$$\boldsymbol{X}(1)^{\mathsf{T}} = \boldsymbol{X}(0)^{\mathsf{T}}\boldsymbol{D}(0) = \begin{bmatrix} c & c & 0 & 0 \end{bmatrix}. \tag{3.3}$$

If in the next round node 1 selects node 3 at random and node 2 selects node 4, we get:

$$\boldsymbol{X}(2)^{\mathsf{T}} = \boldsymbol{X}(1)^{\mathsf{T}}\boldsymbol{D}(1) = \begin{bmatrix} c & c & 0 & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} = \begin{bmatrix} c & c & c & c \end{bmatrix}. \tag{3.4}$$

A couple of remarks must be made about the analysis of Uniform Gossip using a diffusion matrix. Firstly, the given matrix representation of the Uniform Gossip model is not mathematically correct and is only intended to explain the fundamental protocol to the reader with the terminology we use with the other gossip protocols. For instance, the matrix description does not cover the case when a node is selected by two other nodes in the same round. According to the described model by Frieze and Grimmet, a node would simply be marked as 'knowing the rumor' if a node receives a rumor twice in one round. However, in the matrix notation, the nodes' state variable would become $2c$. Hence, the analysis with the diffusion matrix can only be used to get an idea of the operation of Uniform Gossip. Secondly, the Uniform Gossip model only describes the spreading of a single rumor. In many network convergence problems, nodes have a common goal of finding a global average view of the network which is constructed from all local (initial) values. This translates to all nodes starting a rumor, instead of only a single node. However, the added value of Uniform Gossip is the fundamental concepts such as random neighbor selection and interacting with at most one node at a time that are introduced. These concepts form the basis of most gossip-based communication protocols. We will see the network-wide diffusion of local information through the network in the description of the other gossip protocols in the remainder of this section.

### Mass conservation and convergence stability

For further discussion of gossip protocols, it is useful to introduce properties playing an important role in the design of gossip protocols. Firstly, we present the definition of mass conservation property for gossip protocols [33, 41]:

**Definition 3.1.2 (Mass conservation)** *Let $\boldsymbol{D}(t)$ be the diffusion matrix associated with a gossip protocol. $\boldsymbol{D}(t)$ is right stochastic if*

$$\boldsymbol{D}(t)\mathbf{1} = \mathbf{1}, \forall t. \tag{3.5}$$

*If (3.5) holds, then the sum of local estimates is conserved:*

$$\boldsymbol{X}(t)^{\mathsf{T}}\mathbf{1} = \boldsymbol{X}(0)^{\mathsf{T}}\mathbf{1}, \forall t. \tag{3.6}$$

Equation (3.6) is called the mass conservation property because it ensures that if a network has converged, it has converged to the true average.

Secondly, we present the stable average property which is related to left stochasticity [33, 41]:

**Definition 3.1.3 (Stable average)** *Let $\boldsymbol{D}(t)$ be the diffusion matrix associated with a gossip protocol. $\boldsymbol{D}(t)$ is left stochastic if*

$$\mathbf{1}^\mathsf{T}\boldsymbol{D}(t) = \mathbf{1}^\mathsf{T}, \forall t. \tag{3.7}$$

*If (3.7) holds and $\forall i : x_i(t) = \overline{x}$, then:*

$$\boldsymbol{x}(t+1) = \boldsymbol{x}(t). \tag{3.8}$$

When a matrix is both right stochastic and left stochastic, i.e. (3.5) and (3.7) hold, a matrix is called doubly stochastic.

Definition 3.1.2 and 3.1.3 are important properties for gossip protocols because they ensure network convergence to the true average and a stable state of the network after convergence. These guarantees are essential to the objective of using gossip protocols to let all the nodes in a network converge to a shared global view of the network. While bidirectional gossip protocols often encompass doubly stochastic diffusion matrices, unidirectional gossip protocols do not necessarily use this. The following sections discuss two unidirectional gossip protocols with right stochastic diffusion matrices and a different stability constraint to compensate for the absence of double stochasticity.

### Push-Sum (PS)

| | |
|---|---|
| **Network objective:** | Convergence |
| **Casting type:** | Unicast |
| **Directionality:** | Unidirectional |
| **Partner selection:** | Random |

Another fundamental type of gossip protocol is the Push-Sum algorithm proposed by Kempe *et al.* [28]. The Push-Sum algorithm is based on the idea of how Uniform

Gossip disseminates information through a network. That is, in every iteration each node in the network selects one neighbor uniformly at random, to send information to. Push-Sum does not diffuse the local state variables $x_i(t)$ directly through the network but makes use of two auxiliary state variables called sum $s_i(t)$ and weight $w_i(t)$ where $i \in \{1, 2, ..., N\}$. If the objective is to compute the network average, these variables are initialized to $s_i(0) := x_i(0)$ and $w_i(0) := 1$. In addition, each node sends the pair $(s_i(0), w_i(0))$ to itself in round $t = 0$. This self-loop is needed to start the Push-Sum algorithm (Algorithm 1), which is performed in each subsequent round $t \geq 1$. We define $((\hat{s}_{r,i}(t), \hat{w}_{r,i}(t)))$ as the vector containing all the received sum-weight pairs of node $i$ in round $t$. Each node $i$ follows Algorithm 1 and thus performs a set of actions in each round $t$. Firstly, all sums and weights are summed to compute $s_i(t)$ and $w_i(t)$ (step 1). Next, node $i$ selects a random node $j$ uniformly at random (step 2). Subsequently, half of the computed $s_i(t)$ and $w_i(t)$ are sent to $j$ and the other half is sent to node $i$ itself (step 3). At any round $t$, the current state variable of node $i$ is $x_i(t) = \frac{s_i(t)}{w_i(t)}$.

**Algorithm 1** Push-Sum protocol [28]

1: Compute $s_i(t) = \sum_r \hat{s}_{r,i}(t-1)$ and $w_i(t) = \sum_r \hat{w}_{r,i}$.
2: Choose a target $j$ uniformly at random.
3: Send pair $(\frac{1}{2}s_i(t), \frac{1}{2}w_i(t))$ to $j$ and $i$ (yourself).

When considering the algorithm on the network level, we define the vector of sums and the vector of weights similar as $\boldsymbol{X}(t)$, namely: $\boldsymbol{S}(t) = [s_1(t), s_2(t), ..., s_N(t)]^\mathsf{T}$ and $\boldsymbol{W}(t) = [w_1(t), w_2(t), ..., w_N(t)]^\mathsf{T}$, where all the components $s_i(t)$ and $w_i(t)$ represent the sum and weight of node $i$ at time $t$. The sum and weight values are initialized to the initial local values and a vector of ones respectively, i.e. $\boldsymbol{S}(0) = \boldsymbol{X}(0)$ and $\boldsymbol{W}(0) = \mathbf{1}$.

The sums and weights vectors are updated according to:

$$\boldsymbol{S}(t)^\mathsf{T} = \boldsymbol{S}(t-1)^\mathsf{T}\boldsymbol{D}(t), \tag{3.9}$$

$$\boldsymbol{W}(t)^\mathsf{T} = \boldsymbol{W}(t-1)^\mathsf{T}\boldsymbol{D}(t), \tag{3.10}$$

in which $\boldsymbol{D}(t)$ is the diffusion matrix. The exchange of values in the network (step 3 from Algorithm 1) is described by this matrix. If node $i$ randomly selects node $j$ in round $t$, then $D_{ij} = 1/2$ and $D_{ii} = 1/2$. In addition, because Push-Sum restricts nodes to contact at most one node per round $D_{ik} = 0, \forall k \notin \{i, j\}$.

Consider again the situation in the example network (Fig. 3.3) where in round $t = 1$ the gossip partners are selected as follows:

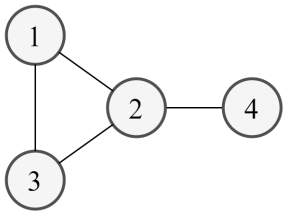



**Fig. 3.3:** Example network.

- node $1 \to$ node $2$
- node $2 \to$ node $4$
- node $3 \to$ node $1$
- node $4 \to$ node $2$

Defining $\boldsymbol{S}(0) = \boldsymbol{X}(0) := \begin{bmatrix} a & b & c & d \end{bmatrix}^\mathsf{T}$ the updates in round $t = 1$ is performed as:

$$\begin{aligned} \boldsymbol{S}(1)^\mathsf{T} &= \boldsymbol{S}(0)^\mathsf{T}\boldsymbol{D}(0) \\ &= \begin{bmatrix} a & b & c & d \end{bmatrix} \begin{bmatrix} 1/2 & 1/2 & 0 & 0 \\ 0 & 1/2 & 0 & 1/2 \\ 1/2 & 0 & 1/2 & 0 \\ 0 & 1/2 & 0 & 1/2 \end{bmatrix} \\ &= \begin{bmatrix} \frac{a+c}{2} & \frac{a+b+d}{2} & \frac{c}{2} & \frac{b+c}{2} \end{bmatrix}, \end{aligned} \tag{3.11}$$

$$\begin{aligned} \boldsymbol{W}(1)^\mathsf{T} &= \boldsymbol{W}(0)^\mathsf{T}\boldsymbol{D}(0) \\ &= \begin{bmatrix} 1 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} 1/2 & 1/2 & 0 & 0 \\ 0 & 1/2 & 0 & 1/2 \\ 1/2 & 0 & 1/2 & 0 \\ 0 & 1/2 & 0 & 1/2 \end{bmatrix} \\ &= \begin{bmatrix} 1 & \frac{3}{2} & \frac{1}{2} & 1 \end{bmatrix}. \end{aligned} \tag{3.12}$$

Hence, the local estimates of the nodes become:

$$\boldsymbol{X}(1) = \frac{\boldsymbol{S}(1)}{\boldsymbol{W}(1)} = \begin{bmatrix} \frac{a+c}{2} \\ \frac{a+b+d}{2} \\ \frac{c}{2} \\ \frac{b+c}{2} \end{bmatrix} / \begin{bmatrix} 1 \\ \frac{3}{2} \\ \frac{1}{2} \\ 1 \end{bmatrix} = \begin{bmatrix} \frac{a+c}{2} \\ \frac{a+b+d}{3} \\ c \\ \frac{b+c}{2} \end{bmatrix}. \tag{3.13}$$

Note that since the nodes send half of the state variables to a gossip partner, and keep half themselves, all the rows of the diffusion matrix sum to one. The diffusion matrix of Push-Sum is right stochastic and the mass conservation property applies to Push-Sum (Definition 3.1.2). The diffusion matrix is not necessarily left stochastic because of the unidirectionality of Push-Sum. Although Push-Sum does not comply with the stable convergence property of (3.7), stable convergence can still be guaranteed with a different stability condition introduced by Bénézit *et al.* [33]. The next section introduces a generalization of Push-Sum and elaborates on this stability condition.

Uniform Gossip and the Push-Sum algorithm are closely related, and it should be noted that these protocols are sometimes confused in literature. For clarity, we make the distinction between the two protocols, based on the network objective. Uniform Gossip has the objective of disseminating information. This means that rumors are spread through the network. Push-Sum has the objective of network convergence and uses the state variables sum and weight to reach convergence. This means that in each gossip interaction, a node updates its state variables. Note that Uniform Gossip forms the basis for many gossip protocols, which make use of the uniform random neighbor selection of Uniform Gossip (whereof Push-Sum is an example).

**Tab. 3.2:** Advantages and disadvantages of Push-Sum, based on [28, 34, 39]

| Push-Sum | |
|---|---|
| **Advantages:** | **Disadvantages:** |
| • Simple implementation. | • Vulnerable to packet loss. |
| • No deadlocks. | • Vulnerable to bottlenecks. |

### Weighted Gossip (WG)

| | |
|---|---|
| **Network objective:** | Convergence |
| **Casting type:** | Unicast/Multicast |
| **Directionality:** | Unidirectional |
| **Partner selection:** | Random |

The Push-Sum protocol described in the previous section, can be considered as a special case of a more general class of gossip protocols called Weighted Gossip. A generalization of the Push-Sum protocol was already mentioned by Kempe *et al.*but

remained unnamed until Bénézit *et al.* proposed Weighted Gossip [33]. This class of gossip protocols is also inspired by Uniform Gossip and targets the design of unidirectional gossip protocols. Bénézit *et al.* remark that most of the presented gossip algorithms require doubly stochastic diffusion matrices to guarantee network convergence. Bénézit *et al.* state that double stochasticity is not needed to accomplish network convergence, and they prove correct network convergence for a broad set of gossip protocols, i.e. Weighted Gossip protocols. Dropping the double-stochasticity requirement allows more freedom in designing gossip protocols for specific applications and situations. The downside of the lack of double-stochasticity, however, is a more complicated convergence analysis. Nevertheless, Weighted Gossip is shown to converge with a simple stability condition for any connected, directed, or undirected graph.

Weighted Gossip follows the same concept as Push-Sum where each node maintains two local variables; a sum $s_i(t)$ and weight $w_i(t)$. While in Push-Sum the shares of nodes contributing to a gossip interaction are restricted to be $1/2$ for the gossip partner and $1/2$ for the current node, Weighted Gossip allows more freedom. Bénézit *et al.* show that as long as the mass convergence property holds (Definition 3.1.2) the network converges to the true average. Following this definition, the only requirement for Weighted Gossip is the diffusion matrix being right stochastic. This allows nodes, for instance, to communicate with multiple nodes in the same round and use different shares than $1/2$, used in Push-Sum, as long as all the rows of the diffusion matrix sum to 1. Hence, this offers opportunities for the optimization of gossip protocols concerning certain characteristics like convergence speed, time complexity, robustness, etc.

Weighted Gossip requires the following rules regarding the diffusion matrix $\boldsymbol{D}(t)$. The shares for 1-hop neighbors can be arbitrarily chosen, so $D_{ij} = \alpha_{ij,t}$, where $\alpha_{ij,t}$ is a scalar value in round $t$ with $0 \leq \alpha_{ij,t} \leq 1$. For the other nodes in the network $D_{ij} = 0$ for all $j$ not being a 1-hop neighbor of node $i$. In addition, the diffusion matrix has to be right stochastic, so each row has to sum up to one: $\sum_{i=1}^{N} \alpha_{ij,t} = 1$ (Mass conservation). Besides, Bénézit *et al.* [33] introduce a simple stability condition for unidirectional gossip protocols, replacing the right stochasticity requirement from (3.7). This is the requirement that the diagonal of the diffusion matrix must be positive. Hence, $D_{ii} > 0$ for each node $i$.

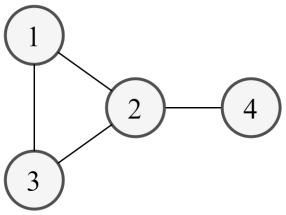



**Fig. 3.4:** Example network.

Using the same example network as before (Fig. 3.4), the diffusion matrix in Weighted Gossip looks the following:

$$\boldsymbol{D}(t) = \begin{bmatrix} \alpha_{11,t} & \alpha_{12,t} & \alpha_{13,t} & 0 \\ \alpha_{21,t} & \alpha_{22,t} & \alpha_{23,t} & \alpha_{24,t} \\ \alpha_{31,t} & \alpha_{32,t} & \alpha_{33,t} & 0 \\ 0 & \alpha_{42,t} & 0 & \alpha_{44,t} \end{bmatrix}. \tag{3.14}$$

Now consider the following gossip interactions in round $t_a$:

- node $1 \rightarrow$ node $2$
- node $2 \rightarrow$ node $1$, $3$ and $4$
- node $3 \rightarrow$ node $2$
- node $4$ not participating in this round

The diffusion matrix of round $t_a$, where all shares $\alpha_{ij,t_a}$ represent non-negative values is:

$$\boldsymbol{D}(t_a) = \begin{bmatrix} \alpha_{11,t_a} & \alpha_{12,t_a} & 0 & 0 \\ \alpha_{21,t_a} & \alpha_{22,t_a} & \alpha_{23,t_a} & \alpha_{24,t_a} \\ 0 & \alpha_{32,t_a} & \alpha_{33,t_a} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{3.15}$$

The shares $\alpha_{ij,t_a}$ can be defined by the specific design of a Weighted Gossip protocol. Nevertheless, when the stability condition of positive diagonals is satisfied, the network converges to the true global average.

Tab. 3.3: Advantages and disadvantages of Weighted Gossip, based on [33, 34, 39]

| Weighted Gossip | |
|---|---|
| **Advantages:** | **Disadvantages:** |
| • Simple implementation. | • Vulnerable to packet loss. |
| • No deadlocks. | • Vulnerable to bottlenecks. |
| • Optimization possibilities. | • Complex convergence analysis. |

### 3.1.4 Bidirectional randomized gossip protocols

So far this chapter has focussed on unidirectional randomized gossip protocols, but a second version of randomized gossip is a prominent topic in gossip communication research: bidirectional randomized gossip. The following sections focus on bidirectional randomized gossip protocols. Firstly, the general concept of Pairwise Gossip is explained, which is followed by the discussion of two versions of pairwise gossip.

#### Pairwise gossip

| | |
|---|---|
| **Network objective:** | Convergence |
| **Casting type:** | Unicast |
| **Directionality:** | Bidirectional |
| **Partner selection:** | Random |

A fundamental bidirectional gossip protocol is called Pairwise Gossip [33, 42]. The term pairwise gossip covers the collection of gossip protocols where state variable exchanges happen in pairs and in a bidirectional manner. This means that in each round multiple pairs of nodes are formed that mutually exchange values. During these pairwise interactions, nodes update their local state variable to the average of their previous value and the value of the gossip partner.

This pairwise averaging can be described with a diffusion matrix. When node $i$ interacts with node $j$, $D_{ij} = 1/2$, $D_{ii} = 1/2$, $D_{ji} = 1/2$, $D_{jj} = 1/2$, and $D_{kl} = 0$ for all $k, l \notin \{i, j\}$. Nodes that do not participate in a round of the gossip algorithm,

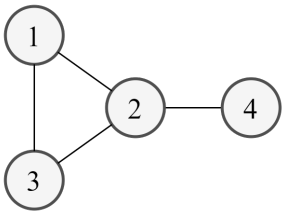



**Fig. 3.5:** Example network.

preserve their value, i.e. $D_{ii} = 1$ for all nodes $i$ not participating. Take the situation in the example network from Fig. 3.5 with the following gossip interactions:

- node $0 \leftrightarrow$ node $2$
- node $1$ and $3$ not participating in this round

Defining $\boldsymbol{X}(t)^\mathsf{T} := \begin{bmatrix} a & b & c & d \end{bmatrix}$, the update in round $t$ is performed like:

$$\begin{aligned} \boldsymbol{X}(t+1)^\mathsf{T} &= \boldsymbol{X}(t)^\mathsf{T}\boldsymbol{D}(t) \\ &= \begin{bmatrix} a & b & c & d \end{bmatrix} \begin{bmatrix} 1/2 & 0 & 1/2 & 0 \\ 0 & 1 & 0 & 0 \\ 1/2 & 0 & 1/2 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \\ &= \begin{bmatrix} \frac{a+c}{2} & b & \frac{a+c}{2} & d \end{bmatrix}. \end{aligned} \tag{3.16}$$

Hence, (3.16) shows that in round $t+1$ nodes 0 and 2 have avaraged their values from the previous round, and nodes 1 and 3 have kept their values, as intended.

Important properties of pairwise gossip protocols are conformance to the mass conservation and stable average properties (Definition 3.1.2 and 3.1.3). The bidirectionality ensures that each agent $i$ sending to $j$ also receives a response from $j$. Given that the used shares are 1/2, all the pairwise communications in one round result in the diffusion matrix being doubly stochastic. See for example the diffusion matrix in (3.16). As a result, the execution of pairwise gossip protocols ensures the convergence to the true average of the network and remains stable after the network has converged. Note the difference with unidirectional gossip protocols, such as Push-Sum and Weighted Gossip, which are only right stochastic (and use a different stability constraint).

In pairwise gossip protocols, gossip partners are often selected uniformly at random (Uniform Gossip). However, pairwise gossip is characterized by bidirectional com-

munications and unicast transmissions between nodes in a network. Pairwise gossip knows many variations, from which we discuss two: standard pairwise randomized gossip and geographic pairwise randomized gossip.

### Standard pairwise randomized gossip

| | |
|---|---|
| **Network objective:** | Convergence |
| **Casting type:** | Unicast |
| **Directionality:** | Bidirectional |
| **Partner selection:** | Random |

The difference between the standard version and the geographic version of pairwise randomized gossip is the freedom in neighbor selection. Both protocols are a combination of Uniform Gossip and Pairwise Gossip: neighbors are selected uniformly at random and interactions take place in pairs communicating bidirectionally.

With *standard pairwise randomized gossip*, neighbors are selected like most gossip protocols: only nodes in the communication range can be selected for gossip interactions. The communication range of a node is the set of nodes that have a 1-hop distance to the current node. In each round of the algorithm, a node has three options: it selects a 1-hop neighbor randomly and sends a gossip request; it is selected by one of its 1-hop neighbors and responds to the gossip request; or it remains silent during this round.

Next to the diffusion matrix description of pairwise gossip described in Section 3.1.4, one additional restriction applies for standard pairwise randomized gossip: $D_{ij} = 0$ for each node $j$ that is not a 1-hop neighbor of node $i$. This ensures that only nodes in a node's communication range can be selected.

Fig. 3.6 shows the communication ranges of two nodes. In this example, node 1 can only select nodes 2 and 3 to gossip with, and node 4 can only select nodes 2, 5, and 6.

The major advantages of standard pairwise randomized gossip are the simplicity of implementation and the convergence analysis of the protocol. This protocol is fundamental for other versions of pairwise gossip. However, this protocol also has downsides. Firstly, storage and computational resources grow linearly with the size of the network and the protocol has a relatively high communication complexity. In

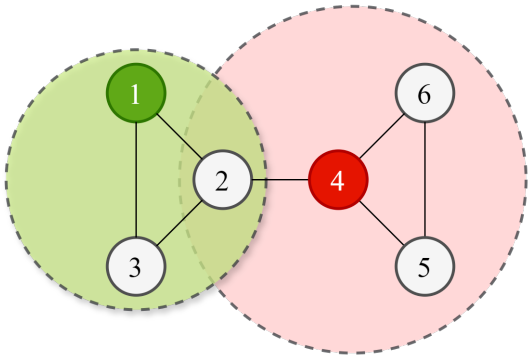



**Fig. 3.6:** Example network with marked communication ranges. The green and red areas mark the communication ranges of nodes 1 and 4 respectively.

addition, pairwise gossip is vulnerable to deadlocks (when no specific precautions are implemented). This occurs when a node sends a gossip request, but also receives a gossip request from another node in the same round. Resolving such a conflict, without affecting convergence time guarantees is a challenge [34]. Another downside is the vulnerability to package loss.

**Tab. 3.4:** Advantages and disadvantages of standard pairwise randomized gossip, based on [34, 38]

| Standard pairwise randomized gossip | |
|---|---|
| **Advantages:** | **Disadvantages:** |
| • Simple implementation. | • Vulnerable to packet loss. |
| • Simple convergence analysis. | • Relatively high communication complexity. |
| | • Storage and computational resources grow linearly with network size. |
| | • Vulnerable to deadlocks. |

### Geographic pairwise randomized gossip

| | |
|---|---|
| **Network objective:** | Convergence |
| **Casting type:** | Unicast |
| **Directionality:** | Bidirectional |
| **Partner selection:** | Random |

In *geographic pairwise randomized gossip* geographic routing is included in the gossip algorithm. This means that nodes are not restricted to selecting a gossip partner in the 1-hop neighborhood but they can select any node in the network. When node $i$ selects node $j$, which is not in the 1-hop neighborhood of node $i$, other nodes will act as a relay, and forward the communication between node $i$ and node $j$. So if node 3, from the example network in Fig. 3.6, selects node 6 as gossip partner, nodes 2 and 4 will forward the messages between 3 and 6. This extension to the standard version has an increased diversity of pairwise averaging due to the network-wide neighbor selection. As a result, geographic pairwise randomized gossip has a lower communication complexity compared to standard pairwise randomized gossip. Next to the general downsides of pairwise gossip that are discussed in the previous sections, geographic pairwise randomized gossip also incurs overhead as a result of multi-hop routing, overhead due to location discovery of nodes, and needs a relatively complex routing scheme.

**Tab. 3.5:** Advantages and disadvantages of geographic pairwise randomized gossip, based on [38]

| Geographic pairwise randomized gossip | |
|---|---|
| **Advantages:** | **Disadvantages:** |
| • Low communication complexity.<br>• Simple convergence analysis. | • Overhead due to location discovery.<br>• Relatively complex routing.<br>• Increased vulnerability to packet loss.<br>• Storage and computational resources grow linearly with network size.<br>• Vulnerable to deadlocks. |

### 3.1.5 Deterministic gossip protocols

While many gossip protocols rely on inherent randomized behavior for robustness and fast convergence, deterministic gossip protocols are also proposed in literature. This section discusses two deterministic gossip protocols, one with the network objective of information dissemination and one with the objective of network convergence.

#### Deterministic Gossip

| | |
|---|---|
| **Network objective:** | Dissemination |
| **Casting type:** | Unicast |
| **Directionality:** | Unidirectional |
| **Partner selection:** | Deterministic |

Haeupler [35] presents a deterministic gossip algorithm as an efficient solution for rumor-spreading called Deterministic Gossip. Haeupler claims that the algorithm is more robust and faster than its randomized pendant. An important underlying idea of Deterministic Gossip is that rumors spread most rapidly if nodes talk more frequently to nodes seen before than to newly selected neighbors.

The deterministic gossip protocol Haeupler describes relies on a bidirectional flooding sub-routine. The basic idea of deterministic gossip can be explained as follows: Each node keeps a list of known initial values (or rumors) and a list of activated links. Initially, all links are not activated and each node only knows its initial value. When two nodes interact for the first time, both nodes add the other node to their list of activated links. The iterative algorithm alternates between two actions: 1) contacting a new 1-hop neighbor (activating the link), and 2) a sequence of flooding steps where each node exchanges its known rumors with all its neighbors bidirectionally. The flooding subroutine ensures that whenever a node gets to know a new rumor, all activated neighbors will also get to know this rumor quickly.

Take again the example network (Fig. 3.7), we consider two iterations of the protocol. During the first iteration, each node activates a link:

- Node 1 $\rightarrow$ node 3
- Node 2 $\rightarrow$ node 1

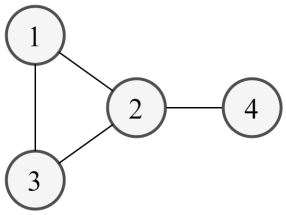



**Fig. 3.7:** Example network.

- Node 3 $\rightarrow$ node 1
- Node 4 $\rightarrow$ node 2

This means that all nodes have one node in their activated neighbors list.

The vector with initial state variables is defined similarly as before: $\boldsymbol{X}(0)^{\mathsf{T}} := \begin{bmatrix}(a) & (b) & (c) & (d)\end{bmatrix}$. It is important to note that Deterministic Gossip has the objective of disseminating information through the network (rumor spreading), rather than letting the network converge to the global view. As a result, each node in the network maintains a vector of all initial values from other nodes, instead of calculating a new local average with newly received information. As a consequence, the update rule cannot be interpreted as a mathematical matrix multiplication. Diffusion matrices of dissemination protocols only describe the interaction between nodes. The matrices only contain zeros and ones, in which a 1 refers to a node sending its vector to another node. When a node receives such a vector, it will add the unknown values to its vector, instead of averaging the values like Push-Sum and Weighted Gossip. Note that the mass conservation and stable convergence properties do not apply to diffusion matrices describing protocols with a dissemination objective.

Haeupler defines an iteration where each node $i$ sends its vector to all nodes that contacted $i$ or were contacted by $i$. Note that this results in more than one message per iteration, when one or more other nodes contact node $i$. This is a consequence of the bidirectionality.

After the neighbor activation step, a sequence of $\lceil 2log(n) \rceil = 4$ flooding iterations are performed. The first iteration has the following diffusion matrix:

$$\boldsymbol{D}(0) = \begin{bmatrix} 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \end{bmatrix}. \tag{3.17}$$

Hence, the gossip vector after the first iteration becomes:

$$\boldsymbol{X}(1) = \begin{bmatrix} (a,b,c) \\ (a,b,d) \\ (a,c) \\ (b,d) \end{bmatrix}. \tag{3.18}$$

In the second flooding iteration, each node contacts the same list of activated neighbors, but with an updated vector of known rumors. This means that $\boldsymbol{D}(0) = \boldsymbol{D}(1)$, and the gossip vector results in:

$$\boldsymbol{X}(2) = \begin{bmatrix} (a,b,c,d) \\ (a,b,c,d) \\ (a,b,c) \\ (a,b,d) \end{bmatrix}. \tag{3.19}$$

In the next flooding iteration ($t = 3$), all nodes know all rumors in the network because node 3 will receive the vector of node 1, and node 4 will receive the vector of 2 (and nodes 1 and 2 know all rumors). It could be possible that a sequence of flooding steps does not result in all rumors being known by all nodes. In that case, the next iteration all nodes again activate one other link and continue with another flooding sequence.

An important property of Deterministic Gossip is that the partner selection for new gossip interactions could be done arbitrarily. In this context, arbitrary means that it does not matter how the choice of partner selection is made for the convergence analysis. Whether the partner selection is random or deterministic, it does not change the bounds of the convergence speed. The beneficial consequence of this is that the choice can be made deterministically. As a result, guarantees for network convergence hold with certainty instead of with high probability (which is the case for random selection).

The flooding mechanism increases the robustness of the protocol because nodes sending all their known rumors in any gossip results in a lot of replication. The major drawback of this is the relatively large communication complexity of a flooding protocol. Congestion on the communication channel could be a potential consequence of this.

Tab. 3.6: Advantages and disadvantages of Deterministic Gossip, based on [35]

| Deterministic Gossip | |
|---|---|
| **Advantages:**<br>• Robustness.<br>• Simplicity.<br>• Faster diffusion compared to randomized pendent.<br>• Network guarantees are certain. | **Disadvantages:**<br>• High communication complexity.<br>• Message size grows linearly with the network size. |

#### Deterministic request-based gossip

| | |
|---|---|
| **Network objective:** | Convergence |
| **Casting type:** | Unicast |
| **Directionality:** | Bidirectional |
| **Partner selection:** | Deterministic |

Other interesting deterministic protocols, having a different network objective and directionality, are proposed by Liu *et al.* [34]. Liu *et al.* bring up the underappreciated idea in literature that it is difficult to design provably correct bidirectional gossip algorithms which are guaranteed to avoid deadlocks. They propose three deterministic request-based protocols that are guaranteed to not deadlock and still ensure exponentially fast convergence. We do not discuss these protocols in detail, the reader is referred to [34] for an explanation of these gossip protocols.

### 3.1.6 Quasi-randomized gossip protocols

| | |
|---|---|
| **Network objective:** | Dissemination |
| **Casting type:** | Unicast |
| **Directionality:** | Unidirectional/bidirectional |
| **Partner selection:** | Quasi-random |

Quasi-random gossip is a rumor-spreading model where each node in the network has a (cyclic) list of neighbors to contact. In contrast to Uniform Gossip, where nodes select a neighbor at random in each round, quasi-random gossip determines the order of neighbors once, at the start of algorithm. When a node receives or starts a rumor for the first time, it randomly selects a starting position in the list of neighbors, and in each subsequent round, it contacts the next neighbor from the list.

An interesting result from the work of Doerr *et al.* [36] is that quasi-random gossip succeeds in spreading a rumor in $O(\log(n))$ rounds for many network topologies, which is the same bound that holds for Uniform Gossip. In some cases, the quasi-random gossip achieves even better bounds than Uniform Gossip. An important implication of this result is that these bounds hold regardless of which neighbor lists are used. A downside of quasi-random gossip is that it is more susceptible for communication bottlenecks. Doerr *et al.* give the example of two (sufficiently large) subgraphs connected with a single connection. In that situation, the classic Push-Sum model converges faster.

Berenbrink *et al.* [37] present another advantage of quasi-random gossip. They proof that quasi-random gossip has significantly smaller communication complexity in comparison to Uniform Gossip in random graphs. In other words, quasi-random gossip requires fewer messages to reach network convergence, compared to Uniform Gossip.

**Tab. 3.7:** Advantages and disadvantages of quasi-randomized gossip, based on [36, 37]

| Quasi-randomized gossip | |
|---|---|
| **Advantages:** | **Disadvantages:** |
| • Same convergence speed as PS in many network topologies.<br>• Smaller communication complexity then PS in random graphs. | • Susceptible for communication bottlenecks. |

### 3.1.7 Broadcast-based gossip protocols

The gossip protocols described in the previous sections rely on the selection of one or more gossip partners per round. Broadcast-based gossip communication is built on a different idea. Instead of selecting individual neighbors for unicast transmission,

nodes send out a broadcast message from time to time. Any node within the physical or artificial communication range overhearing the transmission will process the broadcasted information to update its local state variable.

### Broadcast Gossip

| | |
|---|---|
| **Network objective:** | Convergence |
| **Casting type:** | Broadcast |
| **Directionality:** | Unidirectional |
| **Partner selection:** | Deterministic |

Aysal *et al.* [38] propose a broadcast-based gossip algorithm called Broadcast Gossip. A broadcast-based gossip algorithm does not suffer from disadvantages such as complex routing, location discovery overhead, linearly growing computational and storage resources, and deadlocks of standard pairwise randomized gossip and geographic pairwise randomized gossip (Section 3.1.4). In a broadcast-based communication model, one transmission can reach multiple other nodes, resulting in better communication complexity and faster diffusion of information since multiple simultaneous updates can be done at once. Broadcast-based algorithms do not need routing, resulting in less communication overhead and fewer additional resources needed. Furthermore, [38] states that broadcast-based algorithms are especially well-suited for wireless networks because broadcast-based algorithms exploit the way of communicating in a wireless medium (broadcasts).

The broadcast-based gossip protocol proposed by Aysal *et al.* has a so-called mixing parameter associated with the algorithm. Each node $j$ overhearing a broadcast transmission from node $i$ updates its local value using this mixing parameter:

$$x_j(t+1) = \gamma x_j(t) + (1-\gamma)x_i(t),$$

where $\gamma \in (0,1)$ is the algorithm-defined mixing parameter. When a node broadcasts a message, this means that all neighbors in a node's communication range will receive the information. Hence, the diffusion matrix describing a broadcast of node $i$ in round $t$ has the following properties. For each node $j$ that is a 1-hop neighbor of $i$: $D_{ij} = \gamma$, $D_{jj} = (1-\gamma)$. In addition, the broadcasting node and nodes out of range preserve their value, i.e. $D_{ii} = 1$ and $D_{kk} = 1$ for all nodes $k$ not in the 1-hop neighborhood of $i$. All other components are equal to zero.

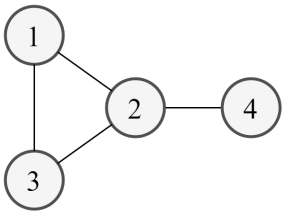



**Fig. 3.8:** Example network.

If in the example network of Fig. 3.8 node 3 broadcasts its current state variable in round $t$, the diffusion matrix will be:

$$\boldsymbol{D}(t) = \begin{bmatrix} (1-\gamma) & 0 & 0 & 0 \\ 0 & (1-\gamma) & 0 & 0 \\ \gamma & \gamma & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{3.20}$$

A major drawback of broadcast-based communication is that mass conservation cannot be guaranteed. To guarantee mass convergence, one should implement an acknowledgment structure, so that a broadcasting node knows which nodes have received the transmission. However, implementing such a structure would nullify the benefits of lower communication complexity and less communication overhead completely. Hence, Aysal *et al.* have to relax the mass conservation property for broadcast-based gossip, and can only make the statement that network convergence will be reached almost surely at a value within the neighborhood of the average of the initial local values of the nodes.

**Tab. 3.8:** Advantages and disadvantages of Broadcast Gossip, based on [38].

| Broadcast Gossip | |
|---|---|
| **Advantages:**<br>• Low communication complexity.<br>• Fast information diffusion.<br>• Low communication overhead.<br>• No routing needed. | **Disadvantages:**<br>• No mass conservation.<br>• Less effective in wired media. |

### 3.1.8 Hybrid gossip protocols

| | |
|---|---|
| **Network objective:** | Dissemination |
| **Casting type:** | Unicast |
| **Directionality:** | Bidirectional |
| **Partner selection:** | Random and deterministic |

Hybrid gossip protocols are algorithms that utilize the benefits of deterministic and randomized gossip algorithms. An example of a hybrid algorithm alternating between deterministic and random choices is proposed by Censor-Hillel and Shachnai [39]. The authors tackle the problem of uniform gossip protocols, which tend to repeatedly communicate between well-connected neighbors, while information is not communicated well across bottlenecks. The proposed hybrid gossip algorithm exploits connectivity within components of a network, to overcome the communication bottlenecks in a network.

**Tab. 3.9:** Advantages and disadvantages of hybrid gossip protocols

| hybrid gossip protocols | |
|---|---|
| **Advantages:** | **Disadvantages:** |
| • Utilizes the best characteristics of randomized and deterministic protocols. | • Complex implementation. |

### 3.1.9 Conclusion and research direction

This section has presented an overview of different types of gossip protocols and has established a framework to characterize different gossip protocols. This framework classifies gossip protocols based on network objective, casting type, directionality and partner selection. The field of gossip protocols has many branches, including unidirectional randomized gossip, bidirectional randomized gossip, deterministic gossip, quasi-random gossip, broadcast-based gossip, and hybrid versions of gossip communication. All the discussed gossip protocols have their advantages and disadvantages. The protocol that is the best solution depends on the requirements of the intended application. For any gossip protocol the compliance with the mass

conservation property is essential to guarantee network convergence. In addition, a gossip protocol should adhere to the stable average property or ensure positive diagonals of the diffusion matrix to achieve stable convergence. Most of the existing gossip protocols are based on a Uniform Gossip model and Push-Sum and pairwise gossip are the most elementary protocols for the unidirectional and bidirectional branches of gossip communication respectively.

The gossip protocols used in this research to investigate the convergence speed in networks are based on the unidirectional gossip protocols Weighted Gossip and Push-Sum. Weighted Gossip is presented like a general framework for the design of gossip protocols, which allows room for customization. The authors specify the requirements for the proper operation of WG, namely right stochasticity to guarantee network convergence and positive diagonals of the diffusion matrices to guarantee stable convergence. Other specifics, such as the type of neighbor selection, number of neighbors to select per round, and the definition of diffusion matrix shares, can be chosen by the designer. Hence, this gossip protocol offers ways for optimization. Besides, we consider the unidirectional communication as a beneficial property, because it does not suffer from deadlocks like pairwise gossip [34]. In addition, the unidirectionality makes the implementation of PS and WG simple in comparison to bidirectional protocols, which is beneficial for when the gossip protocols are applied in real-life, for example in distributed energy networks. Chapter 6 and 7 present the experiments with customizations of WG and the comparison with PS.

## 3.2 Graphs

This section first introduces relevant network properties for this research, and subsequently explains the types of networks used in this research.

### 3.2.1 Graph density and connectivity

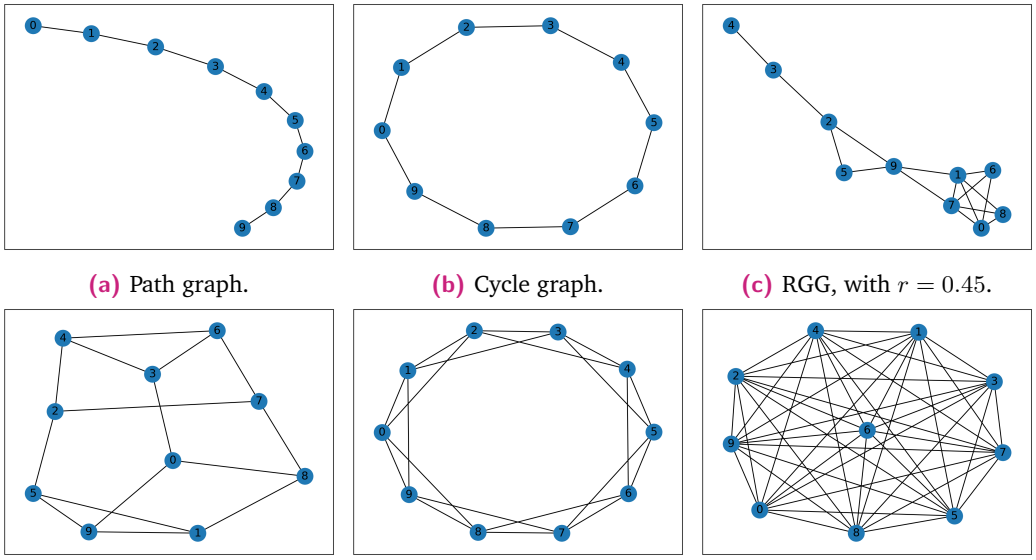
An important network property used in this thesis is graph density. This is defined as the ratio of the number of connections in a graph with respect to the maximum possible number of connections in a graph. The definition of sparsely-connected graphs and densely-connected graphs is often not strictly defined and depends on the context of the application. That is why we follow the definition given by the National Institute of Standards and Technology (NIST) [43, 44]: A sparsely-connected graph refers to a graph in which the number of connections is much less than the possible

number of edges, and a densely-connected graph refers to a graph in which the number of connections is close to the maximum possible number of edges.

A similar concept used in literature is graph connectivity. The connectivity of a graph refers to the minimum number of elements, either nodes or connections, that need to be removed to disconnect a connected graph. Vertex connectivity refers to the minimum number of vertexes, or nodes, to be removed, and edge connectivity refers to the minimum number of edges, or connections, to be removed [45, 46]. The graph connectivity is often used as a measure of the resilience of a network.

### 3.2.2 Graph types

This subsection defines and explains graph types used in this research. Fig. 3.9 gives a typical example for each graph type.



**(a)** Path graph. **(b)** Cycle graph. **(c)** RGG, with $r = 0.45$.

**(d)** $k$-regular graph, with $k = 2$. **(e)** Circulant graph, with $Q = \{1, 2\}$. **(f)** Fully connected graph.

**Fig. 3.9:** Examples of relevant graph types with $N = 10$.

**Path graph:** A path graph of size $N$ is a graph where the nodes can be numbered such that each node $i$ has a connection to node $i + 1$, for all $i \in \{1, 2, \ldots, N - 1\}$. This means that each node is connected to two other nodes, except for the two utmost nodes. This graph type has the lowest possible graph density.

**Cycle graph:** A cycle graph is a graph that consists of a single cycle. In other words, a cycle graph of size $N$ is a graph where the nodes can be numbered such that each node $i$ has a connection to node $i + 1 \mod N$, for all $i \in \{1, 2, \ldots, N\}$.

**Random geometric graph (RGG):** This research only considers 2-dimensional RGGs. In a 2-dimensional RGG, $N$ nodes are placed on the unit square uniformly and independently at random. Given a radius $r$, nodes in the network are connected when the Euclidean distance is less or equal than $r$. Due to the placement of nodes in a 2-dimensional plane and the distance-based connections, RGGs have a close resemblance with real-life wireless networks, as the reachability in such a network is mainly determined by the transmission range of nodes [47].

***k*-Regular graph:** $k$-Regular graphs are defined as graphs in which each node has the same degree $k$. In such a graph we refer to $k$ as the network degree.

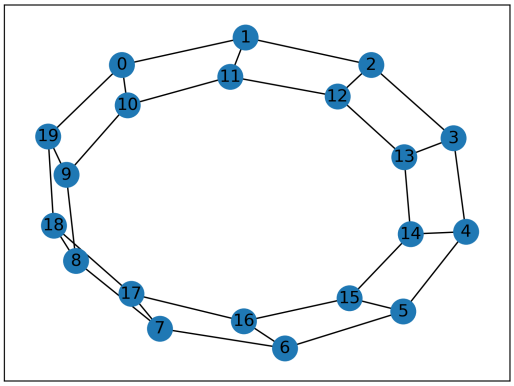



**Fig. 3.10:** A 3-regular circulant graph, with $Q = \{1, 10\}$.

**Circulant graph:** A circulant graph is a special type of $k$-regular graph. A circulant graph is defined by the number of nodes $N$ and a set of jumps $Q = \{q_1, \ldots, q_M\}$, where $M$ is the number of jumps and $Q \subseteq \{1, 2, \ldots, N-1\}$. Circulant graphs are denoted with $C_N^Q$. Fig. 3.10 shows an additional example of a 3-regular circulant graph $C_{20}^{\{1,10\}}$. A circulant graph is constructed as follows: For each jump $q_m \in Q$, each node $i$ has a connection with node $i + q_m \mod N$ and node $i - q_m \mod N$. For instance, node $i = 0$ in the circulant graph $C_{20}^{\{1,10\}}$ of Fig. 7.1, has a connection to node 1 and node 19 due to the jump $q_1 = 1$. The second jump $q_2 = 10$ results in a connection with node 10. Since $0+10 \mod 20 = 10$ and $0-10 \mod 20 = 10$, jump $q_2 = 10$ results in only one additional connection. Similarly, node 1 has connections to node 0, node 2, and node 12, due to the jumps $q_1$ and $q_2$.

**Fully connected graph:** A fully connected graph is a graph in which each node has a connection with every node in the network. This type of network has the largest possible graph density.

# 4 Analysis of individual convergence speed of nodes

This chapter presents an analysis of three scenarios focussing on the optimization of individual convergence of nodes. The first scenario is a single unidirectional gossip interaction and the second scenario has two unidirectional gossip interactions in the same round. These first two scenarios assume no prior knowledge about the accuracy of local estimates of nodes. The third scenario is the same as the second scenario, except that one of the nodes knows the true average.

This research builds upon the generalization of the unidirectional Push-Sum protocol, called Weighted Gossip [33]. Kempe *et al.* [28] and Bénézit *et al.* [33] do bring up the idea of customizing shares in the diffusion matrix. However, they do not elaborate on this topic and they do not discuss if and how the diffusion matrix shares affect the convergence speed of a network. This chapter explores how the diffusion matrix shares influence the convergence speed of individual nodes in a network.

## 4.1 Methodology

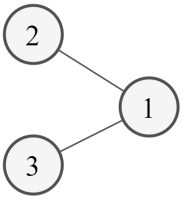



**Fig. 4.1:** The 3-node network for the analysis of individual convergence.

Both scenarios use a simple 3-node network for the analysis of the individual convergence of nodes, which is shown in Fig. 4.1. The analysis considers a Weighted Gossip protocol. The way Weighted Gossip is defined gives a lot of freedom in design [33]. We consider a protocol with the following characteristics:

- The objective is network convergence.
- Nodes use unicast messages to transmit information.

- The gossip protocol is unidirectional.
- A node can receive multiple messages, but will initiate at most one gossip interaction per round itself.

The unidirectionality of Weighted Gossip has the consequence that after a gossip interaction, the two interacting nodes can have different local values depending on the used shares. This is different from bidirectional gossip protocols, such as pairwise gossip, where two nodes by definition have the same local values after a gossip interaction. From this, we hypothesize that there exist opportunities to utilize the used shares in the diffusion matrix to optimize the convergence speed in a network.

The initial values of the nodes in the network are defined as follows:

$$\boldsymbol{X}(0) = \begin{bmatrix} x_1(0) \\ x_2(0) \\ x_3(0) \end{bmatrix} := \begin{bmatrix} a \\ b \\ c \end{bmatrix}. \tag{4.1}$$

The initial values $a, b, c$ are uniformly distributed random variables, with $a, b, c \sim \mathcal{U}[10, 40]$. The bounds on the range of the random variables are chosen arbitrarily, but can be interchanged by any other values while the results of the analysis remain valid.

The analysis of each scenario considers only a single gossip round in which we want to optimize the convergence speed of node 1. In a general network convergence problem, all nodes have to find the network average. Hence, optimizing the convergence speed of node 1 is the same as minimizing the error between the state variable $x_1(t)$ and the network average $\bar{x}$. The sections that follow analyze individual convergence speed optimization in three scenarios. The first scenario considers a single gossip interaction, the second scenario explores two gossip interactions, and the third scenario analyzes two gossip interactions, where one of the nodes knows the true average.

## 4.2 Analysis of Scenario 1

The first scenario targets the analysis of shares on the individual convergence of 1-hop neighbors. Consider the network of Fig. 4.2, where the arrow indicates the gossip interaction. In Scenario 1 a single gossip communication takes place in round $t = 1$, which is node 2 sending to node 1.

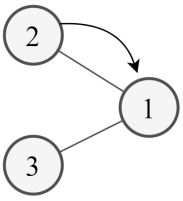



**Fig. 4.2:** Scenario 1: a single gossip interaction.

Following the definition of the Weighted Gossip protocol (Section 3.1.3), the sums and weights matrices are

$$\boldsymbol{S}(0) = \boldsymbol{X}(0) = \begin{bmatrix} a \\ b \\ c \end{bmatrix}, \boldsymbol{W}(0) = \begin{bmatrix} w_1(0) \\ w_2(0) \\ w_3(0) \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}. \tag{4.2}$$

The single gossip communication in round $t = 1$ is described by the diffusion matrix

$$\boldsymbol{D}(0) = \begin{bmatrix} 1 & 0 & 0 \\ \alpha_{21} & \alpha_{22} & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{4.3}$$

The components $\alpha_{21}$ and $\alpha_{22}$ are the shares referring to the transmission from node 2 to node 1 and the 'transmission' of node 2 to itself respectively. For the described scenario, $\alpha_{21}$ is called the outward share of node 2 and $\alpha_{22}$ is called the inward share of node 2. Scenario 1 concerns itself with optimizing the convergence of node 1 through these shares in the diffusion matrix. In other words, the following analysis uses $\alpha_{21}$ and $\alpha_{22}$ as parameters to optimize the convergence of node 1. The following conditions apply to $\alpha_{21}$ and $\alpha_{22}$:

1. $\alpha_{22} > 0$ due to the stability condition of Bénézit *et al.* (Section 3.1.3).
2. $\alpha_{21} > 0$ because node 2 transmits to node 1.
3. $\alpha_{21} + \alpha_{22} = 1$ to comply with the mass conservation property (right stochasticity).

### Optimal outward share

Following the update rules for the sums and weights matrices (3.9) and (3.10), the estimate of node 1 in round $t = 1$ is defined by:

$$x_1(1) = \frac{s_1(1)}{w_1(1)} = \frac{a + \alpha_{21} b}{1 + \alpha_{21}}. \tag{4.4}$$

The optimization of the convergence of node 1 is equal to minimizing the expected error of node 1 to the global average. The expected error of node 1 in the first round $e_1(1)$ is calculated by integrating the squared error of node 1 to the global average over all possible values of $a$ and $b$:

$$e_1(1) = \int_{10}^{40} \int_{10}^{40} (x_1(1) - E[\bar{x}])^2 \, da \, db, \tag{4.5}$$

where $E[\bar{x}]$ denotes the expected value of the global average. The calculation of $E[\bar{x}]$ is trivial, because the ranges of possible initial values $a, b, c$ are known:

$$E[\bar{x}] = E\left[\frac{a + b + c}{3}\right] = \frac{1}{3} E[a + b + c] = \frac{1}{3}(E[a] + E[b] + E[c]). \tag{4.6}$$

Because $a, b, c$ are all uniformly distributed on the same range, their expected values are the same, which we define as $\mu$:

$$\mu := E[a] = E[b] = E[c] = 10 + \frac{40 - 10}{2} = 25.$$

This means that (4.6) reduces to

$$E[\bar{x}] = \frac{1}{3}(E[a] + E[b] + E[c]) = \frac{1}{3}(\mu + \mu + \mu) = \mu = 25. \tag{4.7}$$

Hence, in the considered scenario, the expected value of the global average $\bar{x}$ is equal to the expected values of the initial values $a, b, c$.

Substituting (4.4) and (4.7) in (4.5) yields:

$$e_1(1) = \int_{10}^{40} \int_{10}^{40} \left(\frac{a + \alpha_{21} b}{1 + \alpha_{21}} - 25\right)^2 \, da \, db. \tag{4.8}$$

The double integral can be solved for $a$ and $b$. This reduces (4.8) to:

$$e_1(1) = 67500 \left( \frac{1}{(1+\alpha_{21})^2} + \frac{{\alpha_{21}}^2}{(1+\alpha_{21})^2} \right), \tag{4.9}$$

where $0 < \alpha_{21} < 1$. The function (4.9) on the interval $(0, 1)$ is a convex function, which reaches its minimum as $\alpha_{21} \to 1$.

#### Optimal inward share

Similar to (4.4), the estimate of node 2 in round $t = 1$ can be derived using the update rules of Weighted Gossip:

$$x_2(1) = \frac{s_2(1)}{w_2(1)} = \frac{\alpha_{22} b}{\alpha_{22}} = b. \tag{4.10}$$

Because $\alpha_{22}$ cancels out in (4.10), the convergence of node 2 does not depend on the inward share $\alpha_{22}$. Note, however, that the inward share should still comply with condition 1, i.e. $\alpha_{22} > 0$.

### 4.2.1 Conclusion Scenario 1

Considering a single gossip communication in a scenario as depicted in Fig. 4.2, the individual convergence of a 1-hop neighbor is optimal if the outward share approaches 1. Because of mass conservation (condition 3) this results in the inward share approaching 0. However, this does not influence the convergence of the transmitting node under the condition that the inward share is larger than 0. Hence, in general, a large share improves the convergence speed of the node receiving the gossip variables.

## 4.3 Analysis of Scenario 2

The second scenario considers two gossip communications in the first round. This scenario regards the analysis of the shares of two transmitting nodes, to optimize the individual convergence of one of these two nodes. The arrows in Fig. 4.3 indicate the two simultaneous gossip interactions in Scenario 2. Node 2 initiates a gossip interaction with node 1 and node 1 initiates a gossip interaction with node 3. The objective is again to optimize the convergence speed of node 1. The initial value

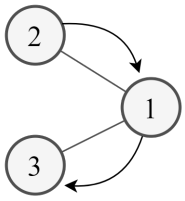



**Fig. 4.3:** Scenario 2: two gossip interactions.

vector $\boldsymbol{X}(0)$, the sums vector $\boldsymbol{S}(0)$, and the weights vector $\boldsymbol{W}(0)$ are the same as in Scenario 1, refer to (4.2).

The two gossip communications in Scenario 2 result in the following diffusion matrix:

$$\boldsymbol{D}(0) = \begin{bmatrix} \alpha_{11} & 0 & \alpha_{13} \\ \alpha_{21} & \alpha_{22} & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{4.11}$$

Similar to Scenario 1, the update of the local estimate of node 1 can be derived using the update rules of Weighted Gossip. Consequently, the estimate of node 1 at $t = 1$ becomes:

$$x_1(1) = \frac{\alpha_{11}a + \alpha_{21}b}{\alpha_{11} + \alpha_{21}} \tag{4.12}$$

Using (4.5) the error to the global average is calculated with:

$$e_1(1) = \int_{10}^{40} \int_{10}^{40} \left( \frac{\alpha_{11}a + \alpha_{21}b}{\alpha_{11} + \alpha_{21}} - 25 \right)^2 da\, db. \tag{4.13}$$

Solving this integral results in:

$$e_1(1) = 67500 \left( \frac{\alpha_{11}^2 + \alpha_{21}^2}{(\alpha_{11} + \alpha_{21})^2} \right) \tag{4.14}$$

Inspection of the 3D-plot of this formula, where the error is plotted against $\alpha_{11}$ and $\alpha_{21}$, shows that the minimum of $e_1(1)$ is at $\alpha_{11} = \alpha_{21}$. This means that the individual convergence rate of node 1 is optimal when the outward share of node 2 $\alpha_{21}$ is equal to the inward share of node 1 $\alpha_{11}$.

### 4.3.1 Conclusion Scenario 2

In a situation like Scenario 2, in which some node $i$ is contacted by node $h$ and node $i$ contacts node $j$, the individual convergence of node $i$ can be optimized using shares of the diffusion matrix. The individual convergence of node $i$ is optimal when the inward share of node $i$ is equal to the outward share of node $h$. Note that this does not mean that these two shares have to be $\frac{1}{2}$ like in the Push-Sum algorithm.

## 4.4 Analysis of Scenario 3

Up to now all the analyzed scenarios considered nodes where it is unknown how well the local estimates represent the true average. Scenario 3 is different, as it is now given that node 2 knows the true average. Except for this given, Scenario 3 is equal to Scenario 2. The expected error of node 1 can be derived similarly to the two previous scenarios. We skip the beginning steps and continue from (4.12).

The fact that node 2 knows the true average means $b = 25$, hence the estimate of node 1 at $t = 1$ becomes:

$$x_1(1) = \frac{\alpha_{11}a + \alpha_{21} \cdot 25}{\alpha_{11} + \alpha_{21}} \tag{4.15}$$

The error between the local estimate of node 1 and the global average is now calculated with a single integral:

$$e_1(1) = \int_{10}^{40} \left( \frac{\alpha_{11}a + \alpha_{21} \cdot 25}{\alpha_{11} + \alpha_{21}} - 25 \right)^2 da. \tag{4.16}$$

Solving this integral for $a$, gives the expected error of node 1:

$$e_1(1) = 2250 \left( \frac{\alpha_{11}^2}{(\alpha_{11} + \alpha_{21})^2} \right) \tag{4.17}$$

From (4.17) we can deduce that $e_1(1)$ is minimal, when $\alpha_{11} \to 0$ and $\alpha_{21} \to 1$. In other words, node 1 converges fastest when the inward share of node 1 is minimal (close to zero) and the outward share of node 2 is maximal (close to 1).

### 4.4.1 Conclusion Scenario 3

The analysis of Scenario 3 shows that the diffusion matrix shares can be used to optimize the convergence speed in a scenario with prior knowledge about the estimate of nodes. In the case with prior knowledge the optimal shares are different from the optimal shares in a scenario without prior knowledge. If a node $i$ is contacted by node $h$, node $i$ contacts node $j$, and it is known that node $j$ knows the true average, the individual convergence of node $i$ is optimal when the inward share of node $i$ approaches 0 and the outward share of node $j$ approaches 1. This result suggests that when two nodes interact, the outward share of the node with the local estimate closest to the true average being close to one results in the fasted individual convergence of the other node.

## 4.5 Discussion

The analysis of Scenarios 1 and 2 is limited in the sense that only a low number of gossip interactions are considered in a single round. In most of the existing unidirectional gossip protocols, all nodes in a network will contact one other node in each round. This analysis only focuses on single interactions and the convergence of the involved nodes. The results do not say anything about the optimization of convergence of all nodes in the network. In addition, the analysis of the optimal convergence of a single node could not directly be extended to the convergence of a network of nodes in general. The optimal convergence of all nodes in a network does not imply the optimal convergence of a network as a whole. In a (larger) network of nodes other dynamics and the architecture of the network could play a significant role in the convergence of the network.

The results of the analysis of Scenarios 1 and 2 are, however, valuable for the insight into the role of diffusion matrix shares in the convergence of nodes. The work introducing Weighted Gossip [33] does not fully explain the effect of the diffusion matrix shares on the operation of a gossip protocol. The results from the analysis in the previous sections do contribute to the clarification of this matter.

An interesting question related to the analysis presented in this chapter is how the number of gossip interactions in a round is related to the convergence speed of a network. This analysis considers the case where nodes only initiate contact with one other node in a round. However, the Weighted Gossip framework gives the designer the freedom to let nodes contact multiple nodes in the same round.

## 4.6 Conclusion

The analysis of the three scenarios in this chapter identify how the individual convergence of nodes can be optimized using shares in a diffusion matrix of a unidirectional gossip protocol. In Scenario 1, where node $h$ contacts node $i$, the convergence of node $i$ is optimal if the outward share of node $h$ approaches 1. The convergence of node $h$ in this situation is not affected by the outward share, under the condition that the stability condition (positive diagonal) is met. In Scenario 2, where node $i$ is contacted by node $h$ and contacts node $j$ itself, the convergence of node $i$ is optimal when the outward share of node $h$ equals the inward share of node $i$. Scenario 3 is equal to Scenario 2 except that now node $h$ knows the true average, leads to different optimal shares. In that scenario the convergence of node $i$ is optimal if the outward share of node $h$ approaches 1 and the inward share of node $i$ approaches zero.

These findings contribute to this research in three ways. Firstly, from this analysis, we conclude that the diffusion matrix shares influence the individual convergence speed of nodes. This means that there are opportunities to utilize these shares to improve the convergence speed of a network. Secondly, this analysis provides more insight into how the shares can be used to utilize the convergence speed. Lastly, Scenario 3 suggests that prior knowledge about the accuracy of the estimates of nodes, can be used to let other nodes converge faster. This finding implies that if certain nodes in a network converge faster, i.e. they have a local estimate closer to the true average than other nodes, this can be used to improve the convergence of other nodes. This result is the basis for our investigation of prioritizing nodes in a network to improve network convergence, which is discussed in Chapter 6.

# 5 Methodology for network convergence

This chapter describes the main application that is developed to execute the simulations presented in Chapter 6, 7 and 8. This main application is the general simulation architecture used for all simulations. However, additional adjustments or extensions were needed for the specific experiments. The methodology sections of the respective experiments following this chapter, describe these additions. Chapter 5 consists of two parts. Section 5.1 introduces the general simulation setup. Section 5.2 explains how simulation runs are performed and gives specifics about the implementation.

## 5.1 Simulation setup

This section is organized as follows: Section 5.1.1 gives a high level description of the main application. Subsequently, Section 5.1.2 describes the requirements of the simulation setup, which is followed by the tools used to realize the setup in Section 5.1.3. Lastly, an overview of the simulation architecture and the main building blocks are discussed in Section 5.1.4.

### 5.1.1 High level description of the main application

The application to execute the experiments presented in Chapter 6, 7 and 8 is intended to simulate the operation of gossip protocols in larger networks. Besides, the application monitors the operation of gossip protocols such that data like local state variables, sent messages, network structure, and network convergence are tracked. In the experiments parameters of this main application are varied to examine the influence on network convergence. Such parameters are for example the radius $r$ of RGGs (Chapter 6), the network degree $k$ of $k$-regular graphs (Chapter 7), the gossip protocol (Chapter 6), the network type (Chapter 7 and 8), or the initial value set (Chapter 8).

The main application contains the basic building blocks to perform the experiments. This application consists of the setup of a network of nodes that communicate to each other by sending messages. The used network topology determines which communications can take place, because nodes are only allowed to communicate with 1-hop neighbors. In addition, the main application contains a logging structure, which keeps track of events taking place during simulation runtime. These events are for example the selection of neighbors, sending and receiving messages, and updating local state variables. After the simulations, an analyzing structure produces figures to visualize the simulation data properly. The next section describes the requirements for the main application.

### 5.1.2 Requirements

The following paragraphs explain the requirements, assumptions, and design choices made for the general research setup.

**Unidirectional gossip:** First of all, this research uses unidirectional gossip protocols for the experiments, as Section 3.1.9 already mentioned. The used protocols are based on Weighted Gossip. Hence, the nodes in the simulation environment should be able to send messages to other nodes and should be able to process messages received from other nodes. Because of the unidirectionality, nodes do not have to be able to reply in the same iteration of the gossip algorithm when they receive a message. Bidirectional protocols would require such functionality and, moreover, a strategy to overcome potential deadlocks that come with it.

**Modularity:** The focus of the simulation application is to gain an understanding of the operation of existing gossip protocols and variations of these protocols on a network level. In addition, the simulation is intended to investigate the influence of design parameters on the convergence speed. This investigation requires multiple experiments. That is why the main application is designed in a modular way and aimed to support multiple experiments. This modularity, with a clear separation of building blocks that are all responsible for certain tasks, helps the extension of the application during the research.

**Network knowledge:** Although we analyze the operation of gossip protocols on a network level, the simulation application is designed with a node perspective in mind. This means that actions performed by individual nodes in the network are based solely on locally available data at these nodes. This approach is chosen because it matches real-life networks where nodes act as autonomous operating entities, and

have to make decisions based on what is known at each node. The full network topology or size is often not known unless a specific network discovery mechanism is implemented as part of the initialization of the communications network. Such a network discovery mechanism is out of scope for this research and, besides, is not required for most of the existing gossip protocols that inherently fit well into the distributed network setting. Therefore, this research implements the realistic perspective of nodes only knowing the basic information, namely the 1-hop neighbor connections, the values retrieved from these neighbors, and initial local values.

**Connected graphs:** This research only considers connected graphs because this is required to reach network convergence. If a graph is disconnected, it means that at least a part of the nodes in the network cannot communicate with another part of the network. Without this ability to communicate, all nodes will lack the knowledge of at least a part of the network, which makes it unlikely that all nodes converge to the same network average. The only possibility to reach convergence in a disconnected graph is the case where the averages of all the connected subgraphs are equal by coincidence. Hence, all graphs used in the simulations are connected.

**Synchronous communication rounds:** Many works about gossip protocols consider synchronous communication rounds. The main reason for this is that it allows a simpler analysis of protocols. In addition, the results obtained in the analysis of synchronous protocols can be extended to apply to asynchronous versions, like the Weighted Gossip generalization of Push-Sum [33]. These asynchronous versions have more meaning in practice because they are guaranteed to run correctly in networks with arbitrary timing behavior [32, p. 197]. Hence, this research follows the commonly used assumption of synchronous rounds in the gossip protocols.

**Reliable data transfer:** We assume no message loss in the communication between nodes of networks. Reliable data transfer is often ensured by implementing an acknowledgment structure. Such an acknowledgement structure is part of the transport layer, which is out of the scope of this research. This research focuses on the algorithm and assumes that reliable data transfer is implemented in the transport layer.

**Static network topology** As this research focuses on the convergence of gossip protocols, the considered network topologies are static. An interesting aspect is the tolerance of gossip protocols to deal with network disruptions. This means the ability to handle a topology that can change during operation. However, this is out of the scope of this research.

### 5.1.3 Tools

The simulations of the current research are set up using a discrete-event simulator package based on standard Python: SimPy [48]. This simulation framework provides basic discrete-event simulation building blocks such as processes, events, and shared resources. The SimPy simulator is compatible with Python, an object-oriented programming language, which is helpful in the objective of creating a modular simulation application. This is a motivation for choosing SimPy for this research setup.

Next to the simulation package, the NetworkX [49] module is used for setting up networks in Python. NetworkX offers functionality to generate many types of network topologies and to obtain properties of (randomly generated) networks. The seamless integration between SimPy and NetworkX, because they are implemented in Python, is the second motivation for using SimPy over alternatives like SimEvents [50] from the Matlab & Simulink environment. The analysis and monitoring of the simulation in the Simulink environment is restricted to the tools given by the producer. On the other hand, simulation analysis and monitoring in Python can be developed from scratch and tailored to the needs of the current research. SimPy in combination with the generation and analysis of networks provided by NetworkX forms the suitable tool set for the convergence analysis of gossip protocols.

### 5.1.4 Implementation architecture

The simulation environment is set up in a modular manner. Fig. 5.1 shows the modular design and interrelations between classes of the simulation setup. The solid lines show which submodules deliver input for the initialization and operation of other submodules. The dotted lines show how `Messages` are passed between nodes. The implementation consists of six submodules:

- `Main`: the module processing user input, starting the initialization of the other modules, and starting the simulations.
- `Graph`: the module that generates graphs with the NetworkX package. `Network` uses the generated graph to set up the agents, using the interconnections defined by the graph.
- `Network`: the module responsible for setting up a network and initializing the agents in it.

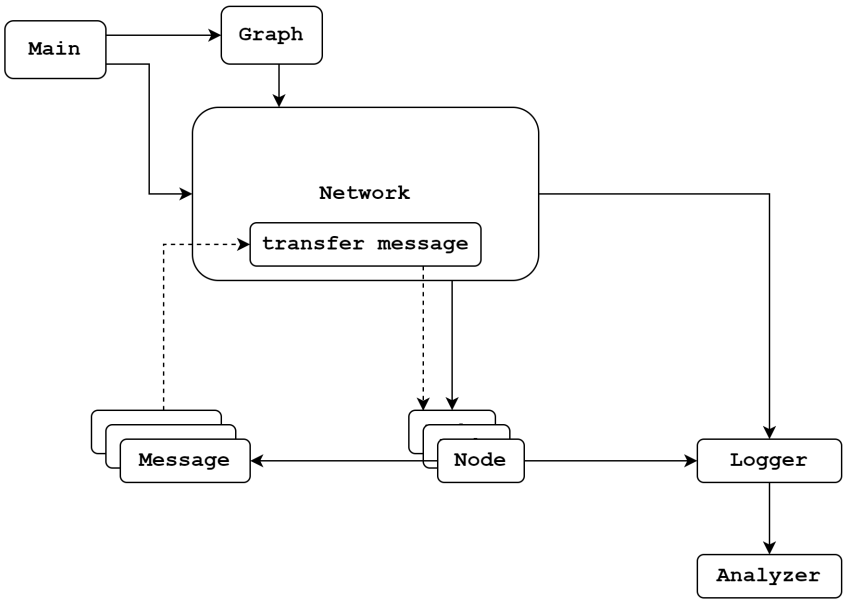



**Fig. 5.1:** Simulation architecture.

- `Node`: the module that provides the entities operating in the network. Objects of this module start a process in the SimPy environment, which communicates with other processes to reach the common objective of finding the global average. Each agent is initialized with an initial value, initial weight, and a list of 1-hop neighbors.
- `Message`: the module that defines the structure and fields of the messages sent between nodes.
- `Logger`: the module that monitors all the events in the simulation.
- `Analyzer`: the module where all functions to analyze the results of the simulations are implemented.

The sections that follow describe the functions and relations of these submodules in more detail and list the most important input parameters per submodule.

### Main

`Main` is the main file of the simulations. This file processes user input in the form of parameters specific to certain simulations, such as network size, simulation time, type of protocol, type of neighbor selection, and the number of runs. `Main` initializes `Graph` and sets up the network using `Network`.

The most important input parameters:

- Simulation time: the number of simulation steps.
- List of network sizes $[N_1, N_2, ...]$.
- List of neighbor selection types, options: random, degree-based.
- List of gossip algorithms to simulate, options: Push-Sum, Degree-based Weighted Gossip.
- Number of runs $p$.

### Graph

`Graph` defines the network topology by generating graphs using NetworkX. This graph defines the connections between nodes. The `Network` module uses this to initialize the list of neighbors at each node, such that the nodes know to which are allowed to send. Supported graphs include random geometric graphs (RGGs), path graphs, cycle graphs, circulant graphs, and fully connected graphs. `Graph` module also has the functionality to guarantee connectedness of the used graphs.

Most important input parameters are:

- Graph type, options: RGG, path graph, complete graph, cycle graph, circulant graph.

### Network

The `Network` submodule is the mainframe for the initialization of nodes, communication between the nodes, and initialization of the `Logger` module. `Network` sets up a network of $N$ nodes and initializes all these nodes with an ID, a list of neighbors, and an initial state value and weight. The `Network` also facilitates the transmission of messages. Nodes that send a message, pass it to the `Network` layer, which delivers it to the destination node.

The most important input parameters are:

- Network size $N$.
- Type of neighbor selection, options: random, degree-based.
- Gossip algorithm, options: Push-Sum, Degree-based Weighted Gossip.
- Initial values $[x_1, x_2, ..., x_N]$ and weights $[w_1, w_2, ..., w_N]$ of the nodes.
- The graph generated by `Graph`.

### Node

The `Node` instances represent the operating entities in the gossiping network. All the nodes in the network start a process in the SimPy environment. This process entails the gossip communications of a node. Each node has an internal clock. Clock ticks denote the transition from one round to the next. When the clock ticks, the node processes the messages received in the previous round and updates its local state value accordingly. In addition, it selects a gossip partner and determines the shares for sending its local state value to this partner and itself. Subsequently, it sends two `Messages` to `Network` which delivers it to the right agents. The determination of shares and selection of neighbors is dependent on the gossip protocol in use. The `Node` instances also have the functionality to select and manage the 1-hop neighbors of a node. As last, all instances of `Node` will report all events to the central `Logger` submodule. The events that are logged are: finishing initialization, selecting a neighbor, sending a message, receiving a message, processing a message, and updating a state variable.

The most important input parameters are:

- Node-ID: the identical identifier $i$ of a node.
- Initial value $x_i(0)$.
- Initial weight $w_i(0)$.
- Type of neighbor selection, options: random, degree-based.

### Message

The `Message` module defines the blueprint of messages sent between nodes. `Message` is implemented as a submodule to be able to develop extensions easily. When a new gossip protocol requires additional fields in the messages, it can be implemented easily. The used messages in the simulations of this research have the following fields: source ID, destination ID, timestamp of generation, value, weight, and message ID.

The most important input parameters are:

- Source: the ID $i$ of the sending node.
- Destination: the ID $j$ of the receiving node.
- Timestamp: time $t$ when the message is sent.
- Value: state variable $x_i(t)$.
- Weight $w_i(t)$.

### Logger

Instances of `Network` and `Node` send logging information to `Logger`. This logging information includes all relevant events and related data taking place during the simulation. The `Logger` collects all the event-information, post-processes it, and stores it in a suitable data table format for `Analyzer`. The post-processing contains the calculation of the true average, error bounds, statistics of individual nodes (such as the final value and individual convergence time), network convergence time, and convergence timelines. In addition, the `Logger` module has the functionality to plot the values of one or more individual nodes over time, which can be used to inspect the operation of gossip protocols.

The most important input parameter is:

- Error: error margin $a$ to calculate the error bounds.

### Analyzer

`Analyzer` uses the data table containing the simulation results, produced by `Logger`. The analyzer module is separated from the other modules such that it can perform the analysis after the simulations. This has the advantage that simulation results can easily be compared, combined, or analyzed at a later time, using `Analyzer`. `Analyzer` contains the functions to generate the figures to visualize the performance of gossip protocols. This includes the comparison of convergence of different protocols with confidence intervals or heatmaps, the plotting of graph properties, and individual convergence timelines.

The most important input parameter is:

- List of records with events of nodes.

## 5.2 Executing simulations

In the chapters that follow, all experiments use a certain number of runs because of the involved randomization in the simulation of gossip protocols. This section clarifies what is meant by a 'simulation run' and explains the parameters and metrics used in these runs.

### 5.2.1 Simulation setting

In an energy management context, a particular use case for communication between nodes is the exchange of power schedules between nodes to find the aggregate power schedule. For example, a DSM approach like Profile Steering of Gerards *et al.* [15] uses aggregate power profiles to achieve peak shaving. Our research, on the contrary, uses 1-dimensional values to study network convergence. However, the results of 1-dimensional values easily generalizes to the convergence of networks in which vectors of values, like power schedules, are disseminated. Besides, this research is also intended to be more broadly applicable than only the energy management domain. Hence, the simulations are based on a generally used example in distributed algorithm research, which is a sensor network that measures temperatures. Consider a network of $N$ nodes containing a temperature sensor. Each node locally measures the temperature, which is its initial value, and each node wants to know the average of all the temperature measurements inside the network. We define all measured temperatures to be in the range from 10 to 40 °C. To simulate such a setting, each node $i$ in the network starts with a uniformly random initial value $x_i(0) \backsim \mathcal{U}[10, 40]$. This research approaches network convergence in an abstract way. That is why this research assumes no prior knowledge about the initial values, and each initial value is even likely to occur, which is achieved with the uniform distribution. Note that the bounds of the uniform distribution are chosen arbitrarily, and could be interchanged by any other combination of values without changing the findings of this research.

The nodes in the network have the objective of finding the average of all the initial values $\bar{x}(0) = \frac{1}{N}\sum_{i=1}^{N} x_i(0)$. This is the true average of the network. In practice, a node has found the true average when its local estimate approaches the true average sufficiently. Therefore, we define a network to be converged when all the local estimates of the nodes in the network are within an error margin $a$ relative to the true average. This means that when

$$\frac{|x_i(t) - \bar{x}(0)|}{\bar{x}(0)} \leq a, \forall i \in \{1, 2, ..., N\}, \tag{5.1}$$

the network is said to be converged at iteration $t$. The convergence time for a network is defined as $t_c = \min t$ for which (5.1) holds. Since the error margin $a$ is relative to the true average, the simulation results do not depend on the chosen bounds for the initial value distribution. In general, 10%, 5%, or 1% are commonly used values in many domains that use a similar error margin. This research uses an error margin of 5% in all simulations, so $a = 0.05$.

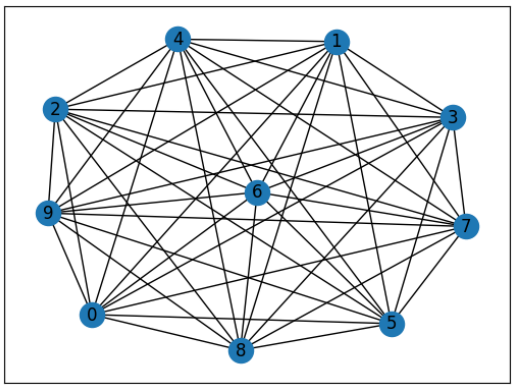



**Fig. 5.2:** Fully connected network of size $N = 10$.

Fig. 5.2 presents a fully connected network of ten nodes. Fig. 5.3 shows a typical simulation run of Push-Sum in this network. This figure shows the local value updates of nodes, and how the nodes converge to the true average. The true average is highlighted in Fig. 5.3, and the upper and lower error bounds mark the error margin. At $t = 8$ all local estimates of the average lie inside the error margin and thereafter stay within this error margin. Hence, the convergence time is $t_c = 8$ in the example of Fig. 5.3.

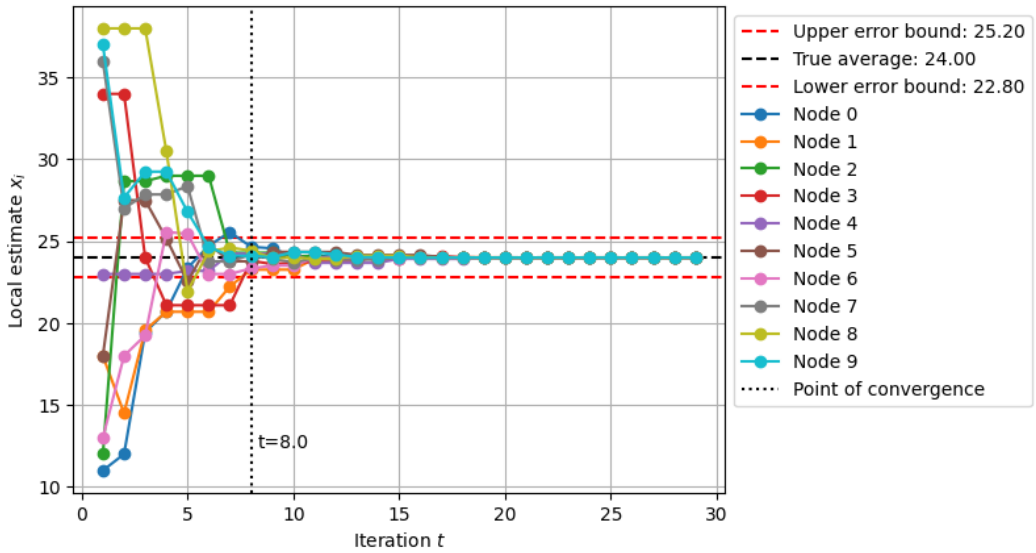



**Fig. 5.3:** Local estimate updates of individual nodes in a fully connected network (Fig. 5.2).

### 5.2.2 Statistical evaluation of measurements

The test setup involves multiple factors of randomization, namely in the graph generation, neighbor selection process, and initial value generation. This randomization causes different behavior each time the experiment is repeated, and the results will vary from run to run. This means that a single simulation run cannot be used to reliably estimate a parameter, like the average convergence time. Every experiment must be repeated sufficiently many times to be able to give a reliable estimate of the parameter for that experiment. That is why this thesis approaches all experiments as statistical evaluations. In these statistical evaluations we estimate the average convergence time based on multiple runs and present confidence bounds to quantify the certainty of the estimate. All experiments in this research apply a commonly used confidence level of 95%. This means that when the experiment is repeated multiple times, each time constructing a confidence interval according to the measurements, this confidence interval will encompass the true value of the parameter in 95% of the cases. The smaller the confidence interval, the more certain the estimate is. Hence, these confidence intervals are presented in most of the experiments in the following chapters, which help to quantify the certainty of the obtained result, in order to make a fair comparison between protocols.

As listed in the section about the `Main` module of the simulation environment under Section 5.1.4, different configurations can be supplied to the simulation environment. These configurations specify the network size $N$, the neighbor selection type, and the gossip algorithm. `main` performs $p$ simulation runs for all the supplied configurations to obtain a set of samples. The average convergence speed and the confidence interval are computed by `Analyzer` from this sample set later. In each simulation run, the simulation application performs the following (simplified) steps:

- Generate a graph $G$ of size $N$.
- Generate an initial value set $X(0)$.
- For each combination of the given neighbor selection types and gossip algorithms, use $G$ and $X(0)$ to set up a network.
- In all these (equal) networks, simulate the gossip process until the network has converged.
- Add the obtained simulation data of each network to the sample set.

This sequence of steps ensures that all the different configurations are performed on the same (randomly) generated graphs and the same (randomly) generated initial value sets. The following chapter presents the first experiments that use the simulation application.

# 6 Network convergence speed with node prioritization

This chapter switches from the individual perspective on convergence in Chapter 4 to the analyses of convergence in a network-wide context. The analytical results applying to the limited scenarios presented in Chapter 4 do not guarantee optimal convergence in larger networks. New dynamics might be introduced due to the architecture of such a network. Analyzing a larger network in the same manner as before significantly increases the scope of the analysis. That is why this chapter investigates the convergence speed in a network empirically.

We hypothesize that the prioritization of high-degree nodes in a network can lead to a lower network convergence time. This chapter proposes two algorithms that attempt to improve the network convergence speed using node prioritization based on node degrees. The first algorithm we propose utilizes diffusion matrix shares for this prioritization. This protocol is called Degree-based Weighted Gossip (DWG) and is explained in more detail in Section 6.1. The second gossip protocol we propose utilizes the probability that neighbors get selected for the prioritization. This protocol is referred to as Push-Sum with degree-based neighbor selection (PS*)[1] and is described in Section 6.2. In addition, we test a third gossip algorithm, which is the combination of DWG and PS*. This protocol does not apply other concepts but is solely added to the experiment to see how the combination of degree-based shares of DWG and the degree-based neighbor selection of PS* affects the convergence speed. This protocol is referred to as Degree-based Weighted Gossip with degree-based neighbor selection (DWG*).

The proposed gossip protocols are evaluated in RGGs with different radii. The results of the experiments do not show faster convergence of the proposed protocols. However, our analysis of individual convergence speeds per node, gives valuable insights in the operation of the gossip protocols. A major finding is that lower-degree nodes are a limiting factor for the convergence speed of a network. In addition, the results show a correlation between the convergence speed of a network, and the network density.

[1]This thesis uses '*' to denote gossip protocols that use degree-based neighbor selection.

The remainder of this chapter is organized as follows. Firstly, the two proposed protocols are introduced in Section 6.1 and 6.2. Subsequently, the simulation setup is presented in the Methodology in Section 6.3. Section 6.4 presents the results of the experiment. Section 6.5 discusses the interpretations, implications and limitations of the findings. The conclusions of this chapter can be found in Section 6.6.

## 6.1 Degree-based Weighted Gossip (DWG)

The first algorithm we propose, Degree-based Weighted Gossip, uses node degrees to determine shares in a diffusion matrix. The degree of a node is the number of 1-hop neighbors it has. DWG builds on the Weighted Gossip algorithm, so the requirements for a Weighted Gossip algorithm described in Section 3.1.3 apply.

The analysis of Chapter 4 showed that in the two scenarios without prior knowledge (Scenario 1 and 2) the convergence speed could be improved by letting an outward share approach 1 and a corresponding inward share approach 0. From this, we hypothesize that higher outward shares might improve the convergence speed of nodes receiving the outward shares. That is what is investigated with Degree-based Weighted Gossip.

In a network consisting of nodes with varying degrees, we can intuitively expect that the dissemination of information from high-degree nodes is faster than the dissemination of low-degree nodes. This is because high-degree nodes can reach more nodes in one hop. In general, the number of hops for node $i$ to reach a node $j$ strongly relates to how much time is needed to send a message from $i$ to $j$. Hence, a node having more 1-hop neighbors means that more nodes can be reached quickly, while having a few 1-hop neighbors means the opposite. That is why we expect information of high-degree nodes to diffuse faster through a network. DWG prioritizes higher-degree nodes, by using shares proportional to the degree of nodes. This results in high-degree nodes receiving the larger outward shares, which is intended to let the higher-degree nodes converge faster. If these nodes have individually converged, this might then result in faster convergence of the neighbors of the high-degree nodes.

The determination of degree-based shares of DWG works as follows. Each node in the network knows its own degree and the degree of its 1-hop neighbors. The degree of node $i$ is denoted as $d_i$. The shares are constructed as normalized ratios between the two node degrees in the following way. If node $i$ randomly selects node $j$, then $D_{ij} = \frac{d_j}{d_i+d_j}$ and $D_{ii} = \frac{d_i}{d_i+d_j}$. For all other nodes $k \neq \{i, j\}$ the share is

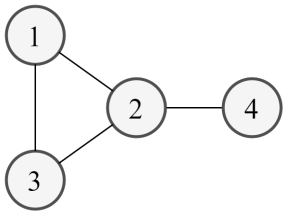



**Fig. 6.1:** Example network.

$D_{ik} = 0$. The ratios are normalized to ensure mass conservation. With these shares $D_{ij} + D_{ii} = 1, \forall i$. This makes the diffusion matrix right stochastic, which implies mass conservation (Definition 3.1.2).

Consider the example network in Fig. 6.1 and the following gossip interactions in a round $t$:

- node 1 $\rightarrow$ node 3
- node 2 $\rightarrow$ node 3
- node 3 $\rightarrow$ node 2
- node 4 $\rightarrow$ node 2

Note that from Fig. 6.1 we know $d_1 = d_3 = 2$, $d_2 = 3$, and $d_4 = 1$. This means that the diffusion matrix for this round is given by:

$$\boldsymbol{D}(t) = \begin{bmatrix} 1/2 & 0 & 1/2 & 0 \\ 0 & 3/5 & 2/5 & 0 \\ 0 & 3/5 & 2/5 & 0 \\ 0 & 3/4 & 0 & 1/4 \end{bmatrix}. \tag{6.1}$$

Note that each column $m$ of the diffusion matrix is a measure of how much a node $m$ receives in this round. Hence, we can observe that node 2 receives relatively much, which is in line with the objective of prioritizing high-degree nodes in the network.

## 6.2 Push-Sum with degree-based selection (PS*)

The second algorithm we propose is intended to improve the convergence speed by applying a non-uniform random neighbor selection, and uses node degrees for this

purpose. This algorithm is based on the Push-Sum algorithm and is referred to as Push-Sum with degree-based neighbor selection (PS*).

The randomized unidirectional protocols described before (Push-Sum, Weighted Gossip, and Degree-based Weighted Gossip) all use the Uniform Gossip model, which means that all neighbors have an equal probability of being selected. In PS*, this probability depends on a neighbor's degree. Each node $i$ assigns a weight to all of its neighbors $j$, which is the degree of the neighbor $j$. At the selection of a gossip partner, a node makes a weighted random choice between its neighbors using these weights. As a result, with PS* the probability of a neighbor being selected is proportional to its degree. Because higher-degree nodes have a higher probability to be selected, higher-degree nodes will be selected more often compared to lower-degree nodes. We expect that this might lead to a faster convergence of the higher-degree nodes, because these nodes will get contacted more often. If the higher-degree nodes are converged, this would then also improve the convergence speed of the neighbors based on the same argumentation of DWG.

## 6.3 Methodology

This section describes the methodology used in the experiment. The requirements of the experiment setup are first given in Section 6.3.1. Subsequently, the simulation setup is discussed in Section 6.3.3.

### 6.3.1 Requirements

In addition to the general requirements for this research setup, given in Section 5.1.2, the experiment of this chapter has a couple more requirements. These are described in the following paragraphs.

**Each node contacts at most one neighbor per round:** Many existing gossip protocols follow the idea of each node selecting at most one other node per round. While the Weighted Gossip algorithm allows nodes to contact multiple nodes in one round, we explicitly restrict nodes to contact at most one other agent per round to follow the conventional idea of gossip protocols. Note that this does not preclude nodes from receiving multiple messages from other nodes, because multiple nodes may select the same node to send their state variable to in a round. This is a result of the unidirectionality of the gossip protocol, which allows the processing of multiple

received messages in a round because a direct reply to nodes is not needed (like in a bidirectional gossip protocol).

**Varying degrees:** To test the idea of node prioritization based on node degrees, the used network should consist of nodes with varying degrees. If a network does not contain enough variation in the degrees of the nodes, the effect of degree-based shares and neighbor selection might not become clear.

**Varying densities:** Next to the investigation of how node degrees can be utilized for the convergence speed, this experiment has a second objective of investigating the influence of the network density on the convergence speed. Hence, this experiment uses various graphs with different densities, to investigate this relation.

**Resemblance with reality:** As this research aims at the development of gossip protocols for implementation in distributed energy networks, it is meaningful to explore the behavior of gossip protocols using networks that resemble real-life networks. Therefore, this experiment setup incorporates network architectures that reflect real-life networks sufficiently.

### 6.3.2 Random geometric graphs

The experiments in this chapter use random geometric graphs (RGGs) to explore the convergence rates of gossip protocols based on Weighted Gossip. RGGs have been a well-studied and popular model for large networks, such as ad-hoc networks and sensor networks [47]. As discussed in Section 3.2, RGGs have a close resemblance with real-life wireless networks and this network type is also widely used in network-related research, for instance [33]. That is why RGGs were chosen to explore the convergence rates of gossip protocols in networks with different densities.

The NetworkX package offers a function to generate random geometric graphs, given the network size $N$ and a radius $r$. Refer to Section 6.3.3 for the visualization of several graphs generated with NetworkX.

The context of gossip protocols operating in an RGG, brings up the issue that the graph must be connected. If a graph is not connected, the graph is unable to converge to the global view. In such a graph, certain nodes are disconnected from the rest of the graph, which makes it impossible to receive and process state variables from at least a part of the network. Hence, the assumption of a connected graph is made. However, the RGGs are not by definition connected, because of the random placement of nodes and the distance-based connections can easily result in isolated nodes, especially for small values of $r$.

To generate connected RGGs we make use of a so-called connectivity threshold [47]. This connectivity threshold is defined as: $r_c = \sqrt{\frac{logn}{\pi n}}$. At the connectivity threshold, it can be said that generated graphs are connected with high probability [47]. Diaz *et al.* [47] use a slightly different definition of RGG, as the nodes are placed on a unit torus instead of a unit square. However, the exact definition is of minor importance, as our objective is to find a practical way to generate connected RGGs within a reasonable amount time. Therefore, with this connectivity threshold, we performed a quick exploration to determine how many retries are needed to obtain a connected RGG for the first time. At the threshold connectivity $r_c$, given by Diaz *et al.*, sometimes more than one hundred retries were needed to get a connected graph. Hence, we introduce a scaling parameter $c$, such that $r_l = cr_c$ and experimented with values $c \in \{1.1, 1.2, 1.3, 1.4, 1.5\}$ to find a suitable lower bound radius $r_l$ for the simulations. Based on observations of this exploration, a radius of $r_l = 1.4r_c$ is decided to be used as the lower bound of radii in the following experiments. At this radius most of the time at most eight retries were needed to obtain a connected graph. This number of retries is acceptable for the simulation runtime, while the obtained graphs are sufficiently sparsely-connected.

A major benefit of this approach is that it does not influence the process of generating RGGs. An alternative approach would be to manually connect nodes after the generation of the RGG. However, this would conflict with the connections being made based on the Euclidean distance, and it might introduce artifacts or biases that reduce the resemblance of the graph with real-life networks. Another option would be to relocate isolated nodes to a new random position after the RGG generation until the graph is connected. Still, this needs a relatively complex implementation, taking additional time to implement, while it is not sure to result in lower runtimes. That is why the simple yet effective approach with retries of generating graphs is used to produce connected RGGs.

### 6.3.3 Simulation setup

The simulations are set up as follows. Four gossip protocols are examined in the simulations:

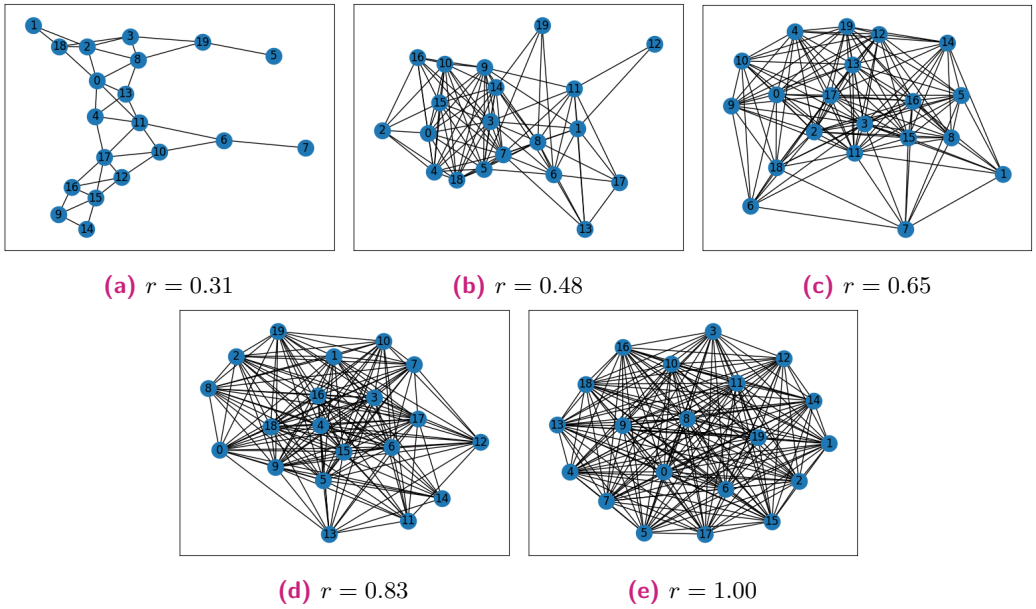
- Push-Sum (PS)
- Degree-based Weighted Gossip (DWG)
- Push-Sum with degree-based neighbor selection (PS*)
- Degree-based Weighted Gossip with degree-based neighbor selection (DWG*)

The first two gossip protocols, PS and DWG, use the standard way of selecting neighbors that is also used in the papers proposing the algorithms, namely a uniformly random selection. The other two protocols, PS* and DWG*, use a degree-based selection of neighbors.

We are interested in the convergence of these four protocols in RGGs with different radii. The simulations use a network size of $N = 20$ and five radii for the generation of RGGs to search for a pattern between density and convergence behavior. The lower bound radius is

$$r_l = 1.4 \cdot r_c = 1.4 \cdot \sqrt{\frac{log(n)}{\pi n}} = 1.4 \cdot \sqrt{\frac{log(20)}{\pi \cdot 20}} \approx 0.31$$

and the upper bound is set at $r_u = 1$. Four incremental steps between these bounds are taken, so the used $r \in \{0.31, 0.48, 0.65, 0.83, 1.00\}$. Fig. 6.2 presents one example of a generated RGG for each used radius to give an idea about what the simulated networks look like.



**(a)** $r = 0.31$ **(b)** $r = 0.48$ **(c)** $r = 0.65$

**(d)** $r = 0.83$ **(e)** $r = 1.00$

**Fig. 6.2:** Examples of generated RGGs for each radius.

The simulations consist of $p = 1000$ runs for each protocol at all five different radii $r$. This number simulation runs results in sufficient confidence intervals to compare the performance of the gossip protocols. In each run, one connected RGG is generated as described in Section 6.3.2, and all four algorithms are simulated one by one using this graph.

### 6.3.4 Evaluation of the simulations

We evaluate the gossip protocols and their convergence times in three different ways. The first evaluation is a comparison of the network convergence time for each of the protocols in the different RGGs. This evaluation is presented using confidence intervals in Section 6.4.1. The second evaluation is intended to investigate the observed results from the first evaluation in more detail, to explain the behavior. This is an analysis of individual convergence rates per node degree. This evaluation is presented in Section 6.4.2 and uses boxplots to visualize the variation of observed individual convergence times. The third evaluation considers the gossip protocols similarly as the first evaluation, but uses reversed prioritization in the gossip protocols. In DWG this means that when node $i$ selects node $j$ the shares $D_{ii}$ and $D_{ij}$ are swapped. This means that $D_{ij} = \frac{d_i}{d_i+d_j}$ and $D_{ii} = \frac{d_j}{d_i+d_j}$. In PS* the assigned weights are reversed such that higher-degree nodes get low weights and lower-degree nodes get high weights. The reversed weights are defined as the neighbors' degree subtracted from the highest observed node degree, which is 15 (Fig. 6.4).

## 6.4 Simulation results

This section presents the simulation results and is subdivided into three parts. Section 6.4.1 presents the results regarding the convergence times of the four gossip protocols in RGGs. Section 6.4.2 presents results to analyze the individual convergence of nodes per node degree. Finally, Section 6.4.3 presents the results of simulations that apply a reversed prioritization in the proposed protocols.

### 6.4.1 Network convergence

Fig. 6.3 shows the convergence times for the four gossip protocols of RGGs with different radii. From Fig. 6.3 a couple of observations can be made.

Firstly, we make three observations in the comparison of the individual protocols:

1. The relative differences between the protocols are most significant at the lowest radius. The highest two radii do not show significant differences between all gossip protocols.

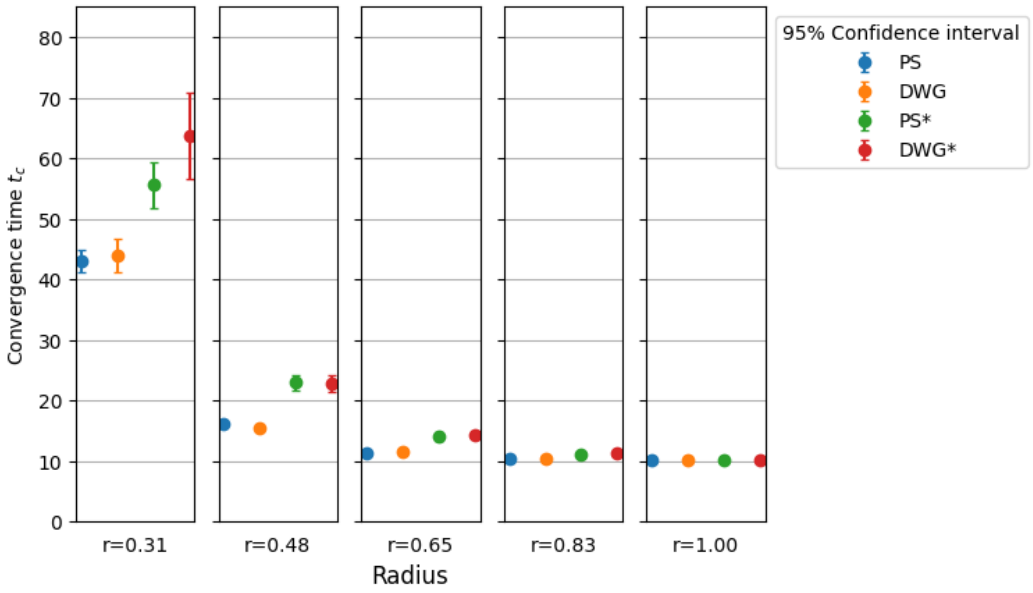



**Fig. 6.3:** The convergence times of gossip protocols with node prioritization and Push-Sum for RGGs with different radii.

2. At the lowest three radii $r \in \{0.31, 0.48, 0.65\}$, PS and DWG show a similar average convergence time, while PS* and DWG* show significantly larger average convergence times.
3. With the increasing average convergence time, also the confidence interval increases. This is visible at the lowest radius, and to a lesser extent at the radii $r \in \{0.48, 0.65\}$.

Secondly, considering how the results of the four protocols in general change over the range of radii, we see two trends in Fig. 6.3:

1. At the lowest radius the average convergence times of the four protocols are relatively large with $t_c \in [40, 65]$, whereas this reduces towards low convergence times around $t_c = 10$ as the radius increases.
2. The confidence intervals are large at the lowest radius and diminish towards such small confidence intervals that they are not visible in the figure at higher radii. The larger confidence intervals refer to a larger variation in measured convergence times in individual runs.

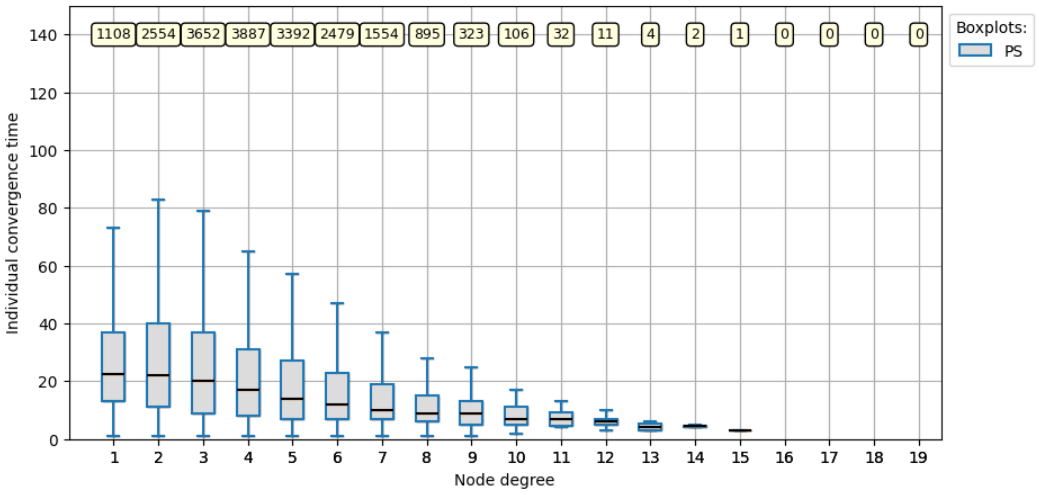



**Fig. 6.4:** Box plots of individual convergence times per node degree of PS in RGGs ($r = 0.31$). The results are obtained with $p = 1000$ simulation runs.

### 6.4.2 Individual convergence per degree

Fig. 6.4 presents box plots of the individual convergence times per node degree of PS in RGGs with $r = 0.31$. At each degree $d_i$ the total number of samples (the total number of observed nodes having degree $d_i$ in all generated RGGs) is denoted in the yellow box at the top of the figure. Given $p = 1000$, we only consider degrees $d_i \in \{1, 2, ..., 8\}$ to have sufficiently many samples to make deductions. Still, the results of higher degrees than this range are included in the figure to give a complete overview of observed convergence times. From Fig. 6.4, we make the following observations:

1. With the exception of $d_i = 1$, in general holds: the lower the node degree, the larger the individual convergence times are. Nodes with $d_i = 1$ converge on average faster than nodes with $d_i = 2$.
2. With the exception of $d_i = 1$, the lower the node degree, the larger the variance in individual convergence times. Nodes with $d_i = 1$ show less variation than $d_i = 2$.
3. The convergence time of nodes with lower degrees are significantly larger than nodes with higher degrees.

The protocols DWG, PS*, and DWG* show similar results for individual convergence per node degree. The reader can refer to Appendix A.1 for the results of these protocols over the full range of node degrees.

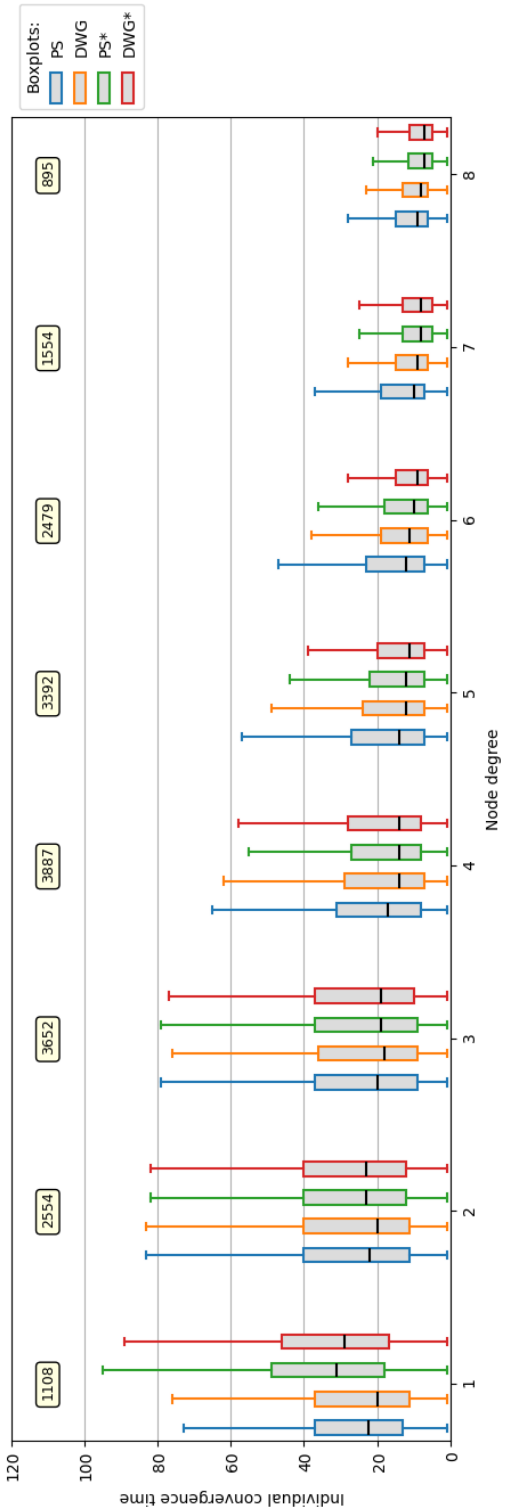



**Fig. 6.5:** Comparison of convergence times of individual nodes for the gossip protocols with node prioritization and Push-Sum in RGGs with $r = 0.31$.

Fig. 6.5 presents a comparison of the protocols on the range $d_i \in \{1, 2, ..., 8\}$, which we consider to have sufficiently many observations. From Fig. 6.5 we make the following observations:

1. DWG has similar results as PS at degrees $d_i \in \{1, 2, 3, 4\}$. DWG shows slightly lower convergence times for $d_i \in \{5, 6, 7, 8\}$ compared to PS.
2. PS* and DWG* show significant larger convergence times at $d_i = 1$. At $d_i \in \{2, 3\}$ the convergence times are comparable to PS. At $d_i \in \{5, 6, 7, 8\}$ the convergence times are slightly lower than PS.
3. Node degrees $d_i \in \{3, 4, 5\}$ have the most samples, with each degree having more than 3300 samples. The degrees $d_i \in \{2, 6\}$ follow, with approximately 2500 samples.

### 6.4.3 Network convergence with reversed priorities

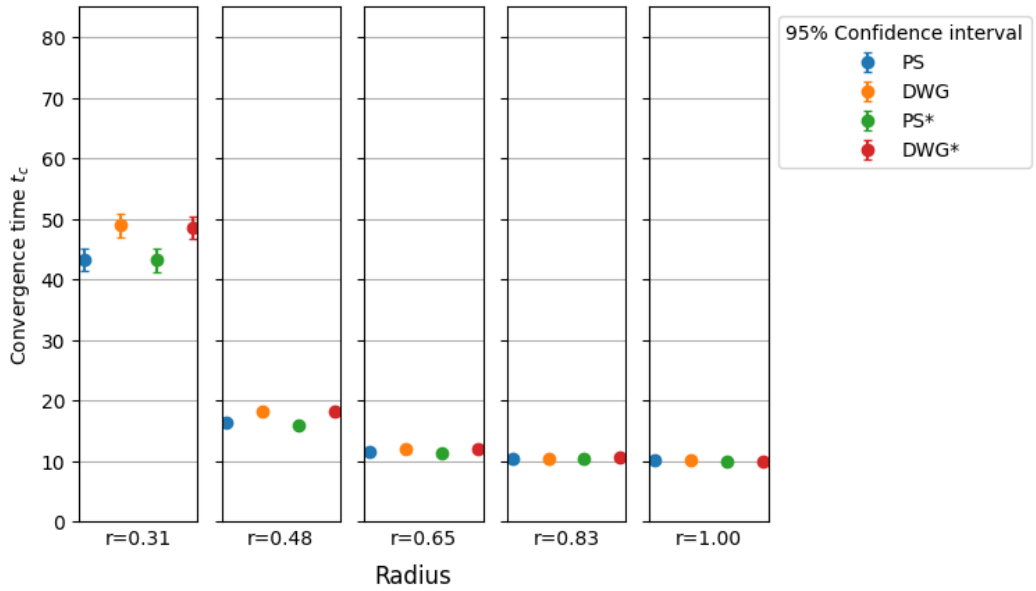



**Fig. 6.6:** The convergence times of gossip protocols with reversed node prioritization and Push-Sum for RGGs with different radii.

Fig. 6.6 shows the comparison of convergence times of gossip protocols with reversed node prioritization. From Fig. 6.6 we make the following observations:

1. At all radii PS* has the same average convergence time and confidence interval as PS.

2. At all radii DWG* and DWG have the same average convergence time and confidence interval.
3. At $r \in \{0.31, 0.48\}$ DWG and DWG* show slightly larger convergence times than PS and PS*. At the other radii, all protocols show similar convergence times and confidence intervals.

## 6.5 Discussion

This section discusses the interpretation and implications of the results in the same order as they were presented in Section 6.4. Section 6.5.1 discusses the interpretation and implications of the first evaluation, which considers network convergence. Section 6.5.2 discusses the interpretation and implications of the second evaluation about individual convergence per degree. Section 6.5.3 discusses the interpretation and implications of the third evaluation, which considers network convergence of the protocols with reversed priorities. Lastly, the limitations of the experiments are discussed in Section 6.5.4.

### 6.5.1 Interpretation and implications network convergence

#### Algorithm design and convergence speed

Firstly, the results presented in Fig. 6.3 show that the most difference in convergence times and confidence intervals between the protocols appears at the lowest radius, and that the least variation is found at the highest radius. Moving from a low to high radius, the differences between the used gossip protocols decrease and become insignificant at the highest two radii. Hence, from these results, we can conclude that sparsely-connected graphs have the most potential for optimizing the convergence speed of gossip protocols, based on the network architecture. In addition, towards fully connected network architectures the differences between the four tested protocols become negligible.

This finding can be attributed to how the symmetry of a network evolves as it becomes more connected. The larger the radius used in the generation of an RGG, the more connections each node will have. Each node having more connections factually means that each node has a higher degree. Nodes having higher degrees, result in the relative difference between the degrees becoming less. Recall how the normalized ratios between nodes $i$ and $j$ are defined: $D_{ij} = \frac{d_j}{d_i+d_j}$ and $D_{ii} = \frac{d_i}{d_i+d_j}$.

If nodes $i$ and $j$ in a sparsely-connected network have degrees $d_i = 1$ and $d_j = 2$, their normalized ratios become $D_{ij} = \frac{2}{3}$ and $D_{ii} = \frac{1}{3}$. However, if the nodes $i$ and $j$ in a densely-connected network have degrees $d_i = 18$ and $d_j = 19$, the normalized ratios become $D_{ij} = \frac{19}{37} \approx \frac{1}{2}$ and $D_{ii} = \frac{18}{37} \approx \frac{1}{2}$. While in both cases the nodes only have a difference in degree of one, it results in a significantly larger share difference in the sparsely-connected case, compared to the densely-connected case. The shares in the densely-connected example are close to $\frac{1}{2}$, which is the share used in all communications in Push-Sum. Similarly, this holds for the degree-based neighbor selection of PS* and DWG* because the relative difference between weights decreases as node degrees become higher. This results in a more uniform probability distribution for the neighbor selection. This explains the differences between the four protocols becoming negligible at high radii.

Secondly, the proposed algorithms show the unexpected result of larger convergence times compared to PS. At all radii used in the experiment, the proposed protocols perform worse or equal to PS concerning the convergence time. The explanation for this is found with the analysis of individual convergence times per node degree, which we discuss in the Section 6.5.2.

Thirdly, the proposed protocols, DWG, PS*, and DWG*, show larger the confidence intervals than PS. This is visible at the lowest radius in Fig. 6.3. A larger confidence interval implies a larger variance in the measured convergence times. This is finding suggests that the adjustments to PS make the gossip protocols more susceptible to the specific network topology in use, while PS performs relatively stable on all the generated graphs with the same radius. Thus, based on these results, PS is more robust in terms of susceptibility to network topology, compared to adjustments of PS that use a form of node prioritization.

### Network density and convergence speed

If we now turn to the general performance of the gossip protocols in relation to the density of the graphs, we conclude that sparsely-connected graphs result in significantly larger convergence times in general. This is based on the observation of all convergence times $t_c \in [40, 65]$ at radius $r = 0.31$ and all convergence times $t_c \approx 10$ at radius $r = 1.00$. This can be explained by the fact that sparsely-connected graphs in general lead to a higher communication complexity. The more connections in a network, the more nodes can reach other nodes with a single hop. As the number of 1-hop connections increases, and the number of multi-hop connections inherently decreases, the number of message transmissions reduces, lowering the

communication complexity. The lower convergence time of the network is a result of the lower communication complexity.

The result of sparsely-connected graphs leading to significantly larger convergence times in general, is a reason to focus on sparser connected graphs in the remainder of this research. Since the investigated protocols do not show significant differences and all the protocols have a significantly small convergence time at higher radii, less improvement can be achieved in that area.

### 6.5.2 Interpretation and implications individual convergence per degree

The observations in Section 6.4.2 can be summarized as follows: the lower the node degree, the larger convergence times are observed and the more variation in convergence time is observed, with the exception of $d_i = 1$. In addition, the convergence times of lower-degree nodes are significant larger than higher-degree nodes. This explains why the proposed algorithms do not give the intended convergence time improvements, as discussed in Section 6.5.1. The basis of both protocols was to prioritize higher-degree nodes, with the intention to let them converge faster, which hopefully results in the other nodes converging faster as well. However, the findings of individual convergence times per node degree show that on average the lower-degree nodes converge significantly slower than higher-degree nodes. Based on these results, it is makes more sense to focus on decreasing the convergence times of the low-degree nodes, rather than focussing on the high-degree nodes, as the low-degree nodes seem to be the limiting factor for faster network convergence.

Turning now to the findings of the comparison of individual convergence times per degree between the four protocols in Fig. 6.3. We have seen that DWG, PS*, and DWG* showed slightly lower convergence times at higher degrees ($d_i \in \{5, 6, 7, 8\}$). This suggests that the prioritization of proposed protocols indeed lowers the convergence time of higher-degree nodes, to a small extent. However, this minimal improvement is at the cost of larger convergence times of nodes with $d_i = 1$ for PS*, and DWG*. From this we conclude that degree-based shares or degree-based neighbor selection, as defined in Sections 6.1 and 6.2, can lead to faster (individual) convergence of higher-degree nodes to a very limited extend. However, this does not result in low-degree nodes converging faster. This explains why the proposed protocols do not improve the network convergence speed.

### 6.5.3 Interpretation and implications network convergence with reversed priorities

The first and second observation are that the PS and PS* perform the same at all radii, and DWG and DWG* perform the same at all radii. From this we conclude that neighbor selection with reversed priorities, does neither improve, nor worsen the convergence speed of the network. Secondly, at the lowest radii DWG and DWG* with reversed priorities show larger convergence times, compared to these protocols with normal priorities. This means that reversing the degree-based shares does not lead to faster convergence of a network.

### 6.5.4 Limitations

The experiment described in this chapter also has its limitations. Firstly, the proposed algorithms rely on the presence of sufficient variation in node degrees in a network, as that is used in the prioritization of nodes. Based on our inspection of the generated graphs, we assume that the generated RGGs, especially the RGGs with a small radius ($r = 0.31$), have enough variation in node degrees to investigate the proposed algorithms. However, this assumption is not assessed explicitly.

Secondly, the proposed algorithms in this chapter use node degrees for prioritization, which might be a too simplistic centrality measure. The idea of using node degrees is to create a ranking of the centrality of nodes, which serves as a measure of how influential nodes are in a network. One of the expectations is that information of high-degree nodes diffuses faster through a network than information of low-degree nodes. The presented simulation results of the setup in this chapter did not imply such behavior. However, there exist more sophisticated forms to measure the centrality of nodes in a network, which could potentially rank the nodes in a better way for our purpose of improving the convergence speed. For example, the well-known centrality types closeness centrality, betweenness centrality, and eigenvector centrality are options that might be better suitable to improve the convergence speed. Degree centrality is the simplest form of centrality, and might therefore be too limited for node prioritization.

## 6.6 Conclusion

The first objective of this experiment was to investigate if the convergence speed of a network can be improved by the design of the gossip protocol. The proposed algorithms, DWG, PS*, and DWG*, do not result in faster network convergence compared to standard Push-Sum. A variation of the proposed algorithms, where the prioritization is reversed, is also evaluated in this chapter, but also did not show faster network convergence times. Therefore, the experiments in this chapter show that the prioritization of higher-degree nodes by means of share determination or neighbor selection does not improve the network convergence time. Another result of the comparison between the protocols is that the proposed protocols showed increased confidence intervals compared to the basic PS protocol. This means that DWG, PS*, and DWG* show more fluctuating results, while PS performs relatively stable in all simulation runs. In this sense, the expected performance of PS in a real-life network can be given with more certainty than the proposed protocols.

A second objective of this experiment was to assess how the density of a network affects the convergence speed. All evaluated protocols show the existence of a relationship between the density of the network and the convergence speed: higher network densities result in significant lower convergence times. Moreover, the higher the network density, the more all protocols show equal performance regarding the convergence speed. Hence, we conclude that applications with sparsely-connected graphs can benefit most from optimizing the gossip algorithms. In densely-connected graphs standard Push-Sum is most likely a better option than similar gossip protocols with adjustments, because of its simplicity and equal performance.

The analysis of individual convergence times per node degrees gives valuable insights in the convergence of individual nodes, and explains why the proposed protocols do not result in lower convergence times. The prioritization in the proposed protocols affect the individual convergence speed minimally, because DWG, PS*, and DWG* show a slight decrease in individual convergence time of higher-degree nodes. However, we consider this as an insignificant difference, and it does not improve the overall network convergence. Moreover, the analysis of individual convergence times per node degree analysis gave an interesting result, namely the correlation between individual convergence time and node degree. With the standard PS protocol, higher-degree nodes already show significant lower convergence times, than lower-degree nodes. This means that the intention of the proposed protocols, namely trying to achieve faster convergence of higher-degree nodes, is misdirected, because the higher-degree nodes already converge significantly faster. From these

results we conclude that lower-degree nodes are a limiting factor for the network convergence speed. Therefore, focussing on improving the individual convergence speed of lower-degree nodes has more potential to improve the overall network convergence.

An interesting idea to explore in future research, might be an algorithm consisting of a phase with standard PS followed by a phase where lower-degree nodes converge faster. With the PS phase the higher-degree nodes can converge relatively fast. The higher-degree nodes having the true average, can then be used to let the lower-degree nodes converge quickly, by using large outward shares, as we concluded from Scenario 3 in Chapter 4.

The findings in this chapter are the reason why the remainder of this research focuses more on sparsely-connected graphs, because these graphs have the most optimization potential. Besides, the correlation between network density and convergence speed is interesting to investigate further, which is attempted in Chapter 7 with $k$-regular graphs.

# 7 Network architecture and convergence speed

This chapter continues the investigation of Chapter 6, and investigates two topics further: diffusion matrix shares and network architecture.

Chapter 6 showed that the customized diffusion matrix shares have a minimal influence on the individual convergence of nodes, but that it does not have a positive effect on overall network convergence speed. Chapter 7 has the objective to further investigate this and gain insight in how diffusion matrix shares can be utilized in a larger network to improve the network convergence speed. To do this, Chapter 7 introduces a different (conceptual) gossip protocol called Static-share Weighted Gossip (SWG), that applies a fixed static share which is used in all gossip interactions. This is similar to PS of Kempe *et al.* [28], where the static share is $\frac{1}{2}$. We hypothesize that a static share of $\frac{1}{2}$ might not result in the fastest convergence in any network. The experiments in this chapter investigate this by applying several static shares in graphs with different densities. The performance of SWG is evaluated in path graphs, RGGs, fully connected graphs, and $k$-regular graphs.

The second objective of this chapter is to investigate the influence of network architecture on the convergence speed. The results of Chapter 6 show that higher-degree nodes converge significantly faster than lower-degree nodes inside RGGs. This chapter investigates the influence of node degrees further by using $k$-regular graphs, where all nodes have degree $k$, to see how this affects the network convergence speed. In addition, the results of this chapter contribute to answering the third sub-question about the influence of network density. The used graphs in the experiments have different densities. This chapter evaluates how this affects the convergence speed is.

A main finding of this chapter is that the static share $\frac{1}{2}$, as used in Push-Sum, does not lead to the fastest convergence in all graph types. The investigated sparsely-connected graphs (path graphs, RGGs with a small radius, $k$-regular circulant graphs with a small network degree) convergence faster with a static share in the range $[0.6, 0.8]$. This chapter also concludes that node degrees are not sufficient as predictors for the convergence speed of a network. In addition, the results of this

chapter support the trend observed in Chapter 6: Densily-connected graphs result in significantly lower convergence times than sparsely-connected graphs.

This chapter is organized as follows. SWG is introduced in Section 7.1. Subsequently, simulation setup and used concepts are explained and motivated in the Methodology in Section 7.2. Section 7.3 presents the simulation results. We discuss these results in Section 7.4 and draw conclusions in Section 7.5.

## 7.1 Static-share Weighted Gossip (SWG)

The Push-Sum protocol of Kempe *et al.* [28] uses shares of $\frac{1}{2}$ in all gossip interactions. However, Kempe *et al.* do not elaborate on why $\frac{1}{2}$ is used. In the experiments of this chapter, we further investigate the diffusion matrix shares by using a similar approach as Kempe *et al.*, namely by using static shares. For this purpose, we propose the conceptual gossip protocol called Static-share Weighted Gossip (SWG). SWG is a version of Weighted Gossip in which all nodes use a fixed constant $\alpha_s$ as outward shares, which is called the static share.

Recall from the analysis of Chapter 4, that when a node $i$ contacts node $j$, it uses an inward and outward share. The local state variables sum $s_i(t)$ and weight $w_i(t)$ are multiplied by these shares when they are exchanged. The inward share determines the fraction of the sum and weight that node $i$ keeps, and the outward share determines the fraction of the sum and weight that is sent to node $j$. With SWG, all nodes use a fixed (network-wide) constant $\alpha_s$ as outward share. Consequently, all nodes also use the same inward share, because the inward share is equal to $1 - \alpha_s$ due to the mass conservation property. In the remainder of this chapter, the $\alpha_s$-parameter is called the static share, which refers to the outward share that all nodes use.

SWG serves as a conceptual tool to gain insight into the influence of diffusion matrix shares on network convergence. In the experiments the static share is a parameter that is varied, to evaluate how this affects network convergence. The conclusion of the analysis of Scenario 2 in Chapter 4 is that optimal convergence of node $i$ is reached when the inward share of node $i$ is equal to the outward share of node $j$, which is sending to node $i$. In the case of static shares, only $\frac{1}{2}$ accomplishes an equal inward and outward share. Although we made this conclusion for the individual convergence, we hypothesize that in a larger network a static share of $\frac{1}{2}$ might not result in the fastest convergence for any network type, and that a gossip protocol

might perform better in certain network architectures when a different static share is used.

## 7.2 Methodology

This section consists of two subsections. Section 7.2.1 describes the used graphs, how the graphs are guaranteed to be connected, and used parameters in the simulation setup. Section 7.2.2 discusses how the simulation runs are evaluated.

### 7.2.1 Simulation setup

#### Graph types

Several graph types are used in the experiments: RGG, path graphs, fully connected graph and $k$-regular graphs. These graphs are chosen to investigate SWG in graphs with different densities. The path graph has the lowest possible density and the fully connected graph has the largest possible density. Furthermore, RGGs with a radius of $r = 0.31$ are also evaluated, because that radius showed the most difference between gossip protocols in Chapter 6. The uniformity in degrees in $k$-regular graphs make it is a suitable way to investigate the influence of node degrees on the convergence speed of a network. That is why this test setup applies $k$-regular graphs with different values of $k$ to assess the influence of node degrees.

#### Guaranteeing connected graphs

A gossip protocol can only be guaranteed to converge if the graph is connected. The implementation of connected path graphs, fully connected graphs and RGGs is trivial. The path graph and fully connected graph are connected by definition, and the generation of RGGs is implemented similar as in Chapter 6, as described in Section 6.3.2. However, $k$-regular graphs are not connected by definition, because a graph consisting of two separated subgraphs, in which all nodes have degree $k$, does comply with the definition of a $k$-regular graph. To accomplish this, we use a special type of $k$-regular graph called a circulant graph, which is connected by definition.

The generation of $k$-regular circulant graphs of size $N = 20$ for each possible value of $k$ requires an explicit construction of set of jumps $Q$. Recall the definition of a circulant graph $C_N^Q$, given in Section 3.2, where $N$ is the network size and

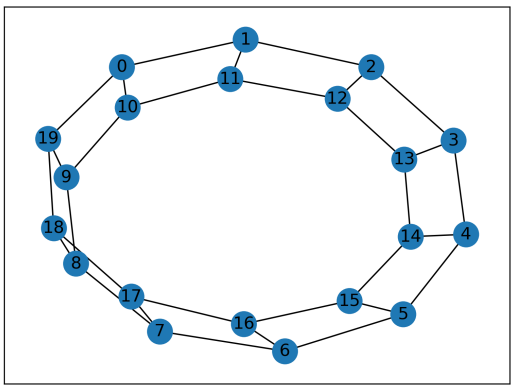



**Fig. 7.1:** A 3-regular circulant graph with the set of jumps $Q = \{1, 10\}$.

$Q \subseteq \{1, 2, \ldots, N-1\}$ is the set of jumps. For convenience, the example circulant graph $C_{20}^{\{1,10\}}$ is presented again in Fig. 7.1. In this example circulant graph, the jump $q_1 = 1$ results in *two* additional connections for each node, and the jump $q_2 = 10$ results in *one* additional connection for each node. This is a general property: adding a jump $q_M$ to a set of jumps $Q$ increases the network degree of the circulant graph by one or two.

Whether this added jump $q_M$ increments the network degree by one or by two can be generalized as follows: Consider a circulant graph $C_N^Q$, with $Q = \{q_1, \ldots, q_{M-1}\}$ and $\frac{N}{2} \notin Q$, which has network degree $k$. If the jump $q_M$, with $q_M \in \{1, 2, \ldots, N-1\} \setminus Q \setminus \frac{N}{2}$, is added to the set of jumps $Q$, then the network degree becomes $k+2$. In the other case, when $q_M = \frac{N}{2}$ is added to $Q$, the network degree becomes $k+1$. These properties are used to generate $k$-regular graphs for each possible value of $k$ in the experiment setup.

In the experiment setup, all possible network degrees for $k$-regular networks with size $N = 20$ are evaluated, namely $k \in \{2, 3, \ldots, 19\}$. Each value of $k$ is accomplished by a set of jumps $Q_k$, which we define as follows: $Q_2 = \{1\}$, $Q_3 = \{1, 10\}$, $Q_4 = \{1, 2\}$, $Q_5 = \{1, 2, 10\}$, $Q_6 = \{1, 2, 3\}$, etc.

### Simulation parameters

SWG is applied in the graph types described in the previous two subsections. Because of the restrictions for diffusion matrix shares given in Section 3.1.3, only diffusion matrix shares in the range $(0, 1)$ can be used. We want to examine this full range, and therefore evaluate SWG for all $\alpha_s \in \{0.1, 0.2, \ldots, 0.9\}$ in the experiments.

Each used configuration in the experiments is repeated with $p = 100$ runs. A configuration in the experiments of this chapter is defined by the combination of graph type and the used static share $\alpha_s$ in SWG. Each $k$-regular graph counts as one graph type. $p = 100$ runs resulted in confidence intervals which are sufficiently small to compare the different configurations and draw conclusions.

In the simulations with RGGs, a new RGG is generated in each simulation run, similar as in Chapter 6. The other graph types (path graph, fully connected graphs, and $k$-regular graphs) are by definition not generated randomly, and are therefore the same for all simulation runs.

### 7.2.2 Evaluation of simulations

The network convergence time of SWG in path graphs, fully connected graphs, and RGGs are evaluated using comparisons of confidence intervals, similar as in Chapter 6. The results of the $k$-regular graphs are evaluated using a heat-map, as this creates a clear overview of the average convergence time for each used static share in each $k$-regular graph.

## 7.3 Simulation results

This section presents the simulation results in the following order:

- RGGs in Section 7.3.1.
- Path graphs in Section 7.3.2.
- Fully connected graphs in Section 7.3.3.
- $k$-regular graphs in Section 7.3.4.

### 7.3.1 RGGs

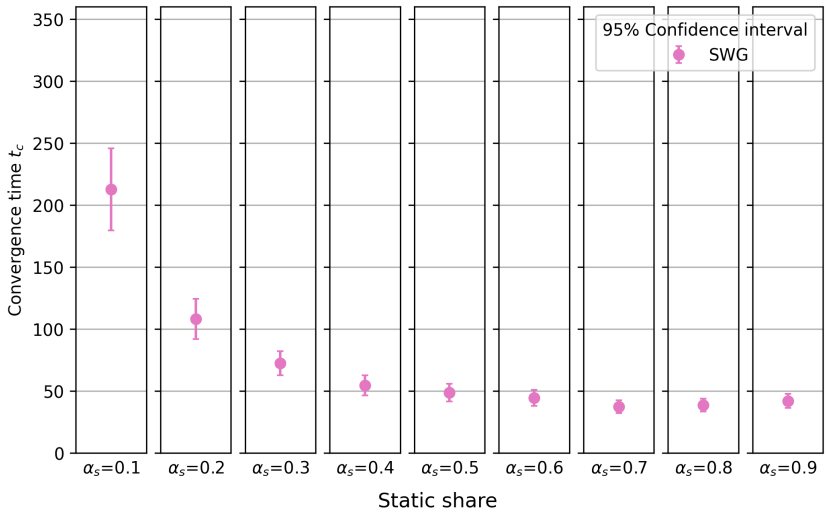



**Fig. 7.2:** Convergence times of SWG in RGGs for different static shares.

Fig. 7.2 shows the convergence times of SWG in RGGs for different static shares applied. A couple of observations can be made from Fig. 7.2:

1. The smaller shares (0.3 and smaller) have significantly higher convergence times than the shares of 0.4 and higher.
2. From the evaluated constants, static share $\alpha_s = 0.7$ has the lowest mean convergence time.
3. The relation between convergence time and size of the static share seems to be inversely proportional. At low static shares, the convergence time shows a significant increase, and at high static shares, the convergence time seems to reach a minimum.

### 7.3.2 Path graphs

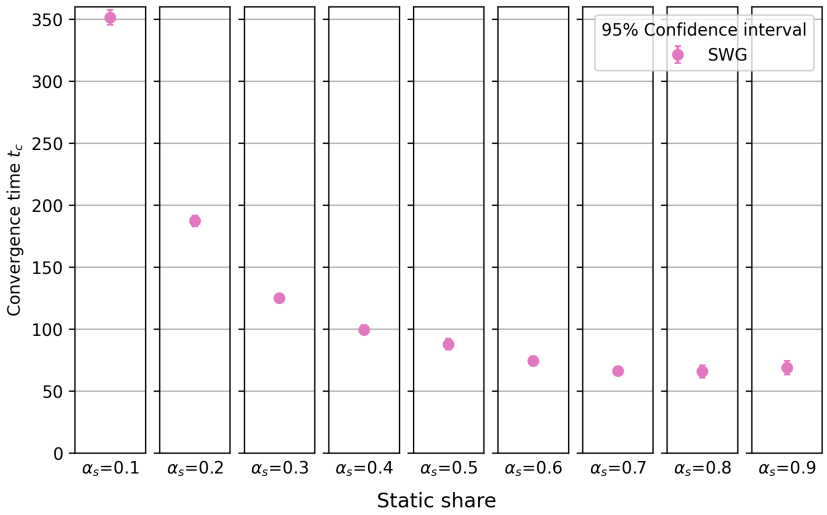



**Fig. 7.3:** Convergence times of SWG in path graphs for different static shares.

Fig. 7.3 presents the comparison of convergence times for a path graph. In this figure, we observe the following:

1. A similar inverse relationship as observed at RGGs is visible.
2. All measured convergence times are, to a limited extent, larger than corresponding convergence times of RGGs.
3. The static share $\alpha_s = 0.7$ has the lowest mean convergence time.
4. The confidence intervals are small in comparison to confidence intervals observed in sparsely-connected RGGs so far.

### 7.3.3 Fully connected graphs

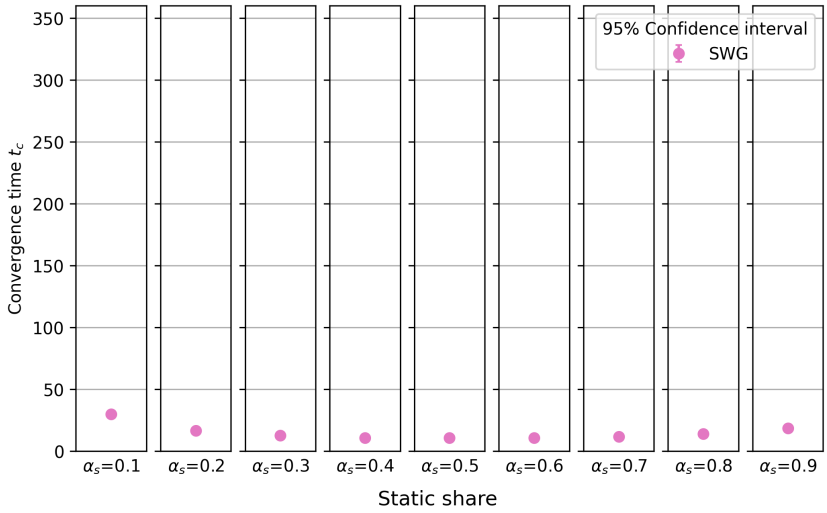



**Fig. 7.4:** Convergence times of SWG in fully connected graphs for different static shares.

Fig. 7.4 shows convergence times of SWG with different static shares in a fully connected graph on the same scale as used with the RGGs and the path graph. When we compare Fig. 7.4 with the results of RGGs and the path graph one thing stands out: All convergence times in the fully connected graph are significantly lower than the convergence times in RGGs and the path graph.

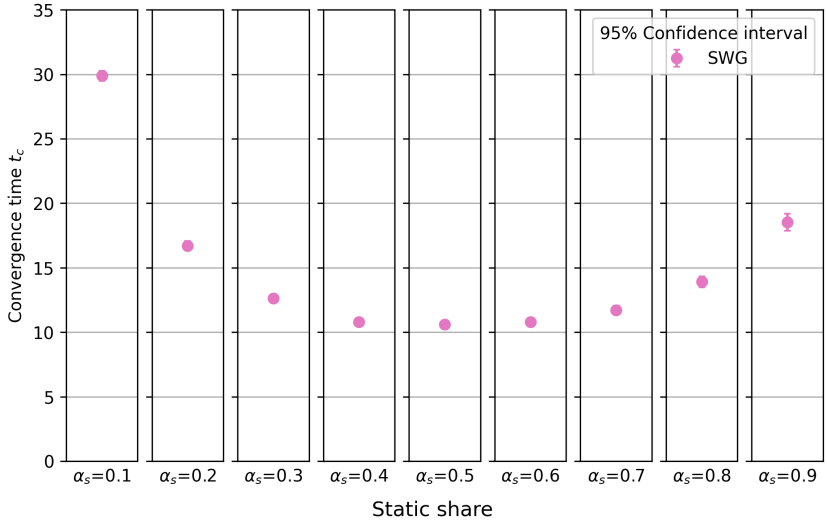



**Fig. 7.5:** Results of Fig. 7.4 on a smaller scale.

For a better analysis of the differences inside fully connected graphs, Fig. 7.5 presents the same results as Fig. 7.4 on a smaller scale. In this Fig. 7.5, the following things can be observed:

1. The relation between convergence time and size of the static share seems to approach a parabolic relationship.
2. The static share $\alpha_s = 0.5$ has the lowest mean convergence time.
3. The figure is not line-symmetric at $\alpha_s = 0.5$. From $\alpha_s = 0.5$ to $\alpha_s = 0.1$, the convergence times increase faster than from $\alpha_s = 0.5$ towards $\alpha_s = 0.9$.
4. All confidence intervals are significantly small.

### 7.3.4 $k$-Regular graphs

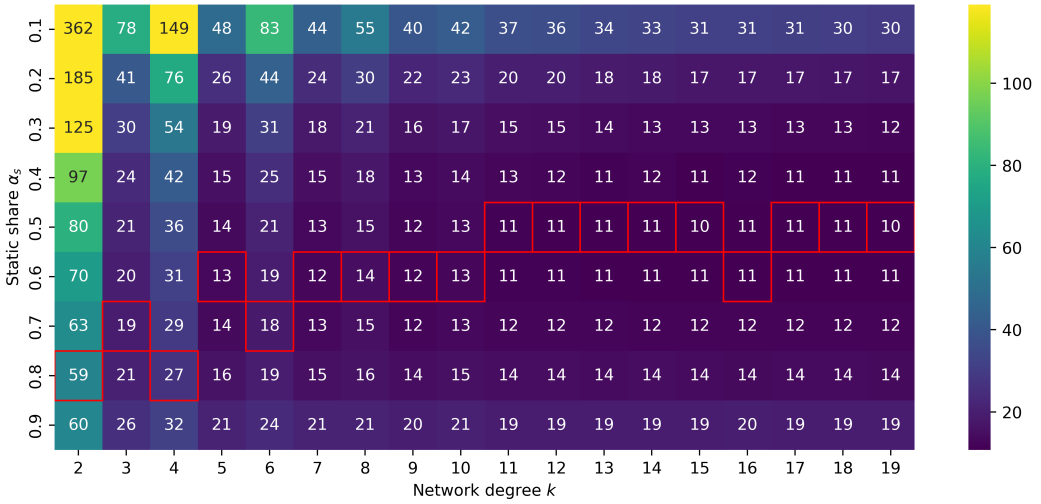



**Fig. 7.6:** Heatmap of the average convergence time of SWG in $k$-regular circulant graphs, for different static shares.

Fig. 7.6 provides a heatmap showing the average convergence times for all possible combinations of static share $\alpha_s$ and network degree $k$. For each network degree $k$, the lowest convergence time is highlighted with a red edge. We make the following observations from Fig. 7.6:

1. At low network degrees ($k \leq 10$), the static shares $\alpha_s \in [0.6, 0.8]$ lead to the lowest convergence time. At high network degrees ($k \geq 11$) static share $\alpha_s = 0.5$ results in the lowest convergence times in almost all cases. Hence, for increasing $k$, the share resulting fastest convergence gradually decreases from $\alpha_s = 0.9$ to $\alpha_s = 0.5$.
2. Low network degrees ($k \leq 11$) result in general in significantly higher convergence times, compared to higher network degrees.
3. At low network degrees ($k \leq 11$), each even value for $k$ shows significantly higher convergence times compared to adjacent odd values of $k$.

## 7.4 Discussion

This section discusses the results presented in the previous section and is organized as follows. The interpretation and implications of RGGs, path graphs, and fully connected graphs are discussed in Section 7.4.1. Subsequently, the interpretation

and implications of the $k$-regular graphs are discussed in Section 7.4.2. Finally, Section 7.4.3 elaborates on the limitations of the experiments of this chapter.

### 7.4.1 Interpretation and implications RGGs, path graphs, and fully connected graphs

#### Confidence intervals

The path graphs and fully connected graphs show significantly smaller confidence intervals than the simulation results of RGGs for the same number of simulation runs $p$. This result can be attributed to the fact that in each simulation run with the RGGs a new RGG is generated, while the path graphs and fully connected graphs by definition are the same for all runs. As a result, the RGG simulations have an additional factor of randomization, compared to the other simulations. This explains the larger variation of convergence times in RGGs. Besides, this finding further supports the idea that network architecture influences the convergence speed, because the different RGGs lead to variation in convergence times, while the path graph and fully connected graph show relatively stable results across all simulation runs.

#### Relation between convergence time and static shares

The results of Fig. 7.2 (RGGs) and Fig. 7.3 (Path graph) show inversely proportional relations between the convergence time and the static share. This relationship could be a consequence of the used inward shares with SWG. A low static share $\alpha_s$ results in a large inward share, because the inward share is $1 - \alpha_s$. Large inward shares enlarges preservation of local state values, because a large inward share of node $i$ results in a larger contribution of the local state value $x_i(t)$ to the calculation of $x_i(t+1)$. More preservation of local state values counteracts diffusion of state values through the network, and thus decreases the convergence speed. In addition, the unidirectionality of the gossip protocol might amplify this effect. A general consequence of unidirectional gossip is that nodes can contact at most one neighbor per gossip round. On the other hand, nodes can receive messages from multiple neighbors. When a node receives multiple messages in one gossip round, it increases the probability the node converges individually because it receives more 'new' information.

Moreover, the fact that $\alpha_s = 0.7$ shows a slightly lower convergence time for RGGs and path graphs, compared to the other static shares, implies that 0.5 is not necessarily always the optimal share for network convergence speed. A path graph as the connected graph with the lowest density possible and an RGG with $r = 0.31$ and $N = 20$ can be classified as a relatively sparsely-connected graph. In both of these (sparsely-connected) graphs $\alpha_s = 0.7$ results in the smallest convergence time, while all static shares $\alpha_s = 0.5, 0.6, \ldots, 0.9$ result in convergence times in the same range.

On the other hand, in the most densely-connected graph possible, the fully connected graph, the static share $\alpha_s = 0.5$ results in the lowest convergence time. In addition, the relation between the convergence time and static share looks to approach a parabolic relationship for the fully connected graph. The fully connected graph does not show similar convergence times for $\alpha_s = 0.5, 0.6, \ldots, 0.9$, as we observed with the sparsely-connected graphs.

From these findings we conclude the following: $\alpha_s = 0.5$ is not necessarily always the optimal share resulting in the lowest convergence speed, depending on the density of a graph. In sparsely-connected graphs, the convergence times do not vary much when static shares in the range $[0.5, 0.9]$ are used. In densely-connected graphs, the lowest convergence times are obtained around $\alpha_s = 0.5$.

### Relation graph type and convergence time

When looking at the general performance on convergence speed, we see that the convergence times obtained in the fully connected graph are significantly lower than the results of RGGs and path graphs. This finding confirms the conclusions made in Chapter 6: graphs with higher densities converge faster than graphs with lower densities.

In addition, the differences in results of the path graphs and RGGs are small. The path graph lowest possible density for a connected graph. Nevertheless, the nodes in the used RGGs mostly have a degree in the range between 1 and 9. This makes these graphs more densely connected compared to path graphs where all nodes have degree 2, except for the two utmost nodes, which have degree 1. Hence, it is surprising that despite the larger density of the used RGGs, there is not a significant improvement regarding convergence times compared to path graphs.

### 7.4.2 Interpretation and implications $k$-regular graphs

This subsection discusses the observations made about $k$-regular graphs in Section 7.3.4. The first observation (the optimal static share resulting in the lowest convergence time gradually decreases from $\alpha_s = 0.9$ to $\alpha_s = 0.5$) is in line with the results obtained for the RGGs, path graphs, and fully connected graphs. That is, the optimal static share for sparsely-connected graphs is larger than 0.5, and the optimal share for densely-connected graphs is 0.5.

The second observation is that for $k$-regular graphs, in general, low network degrees result in significantly higher convergence times, compared to high network degrees. This is in accord with what one can expect based on the results of the RGGs, path graphs, and fully connected graphs. These findings imply that networks with high density converge faster, compared to networks with low density.

The third observation is a surprising result. In the first half of the tested network degrees ($k \leq 10$), even numbered values of the network degree $k$ show significantly higher convergence times in comparison with the adjacent odd values of $k$. This outcome is most likely caused by the specific experiment setup that is used. The pattern used to generate the circulant graphs with even values of $k$ is by adding a new jump $q_M = q_{M-1} + 1$ to the set of jumps $\{q_1, \ldots, q_M - 1\}$ of the circulant graph with degree $k - 2$. This means that new 1-hop connections are made with relatively nearby neighbors. On the other hand, all the circulant graphs with an odd value of $k$, are obtained by adding jump $q_M = 10$ to the set of jumps used for the graph with (even) network degree $k - 1$ and $N = 20$. This means that all neighbors in the network get a new 1-hop connection to the neighbor that was the most number of hops away in the previous circulant graph.

A possible explanation for the increased convergence speeds is that the jump $q_M = 10$ at the odd network degrees, forms for each node a shortcut to the other side of the network. This shortcut allows that values are sent to the other side of the network with a single hop. Once node $j$, which is on the other side of the network, has received a value $x_i(t)$, this value also continues to spread on the other side of the network. This in contrast to the even network degrees, where such a shortcut does not exist. In that case, the value $x_i(t)$ can only spread from one side of the network and it takes more hops to reach node $j$ and the nodes close to $j$. The added jump $q_M = 10$, at the odd network degrees, evidently increases the convergence speed of network significantly, according to the results presented in Fig. 7.6.

From this third result, we conclude that node degrees are not factors that strongly influence the convergence speed. Over the whole range of values for $k$, in general,

higher degrees lead to larger convergence speeds. However, the observed alternating pattern at $k \leq 10$ suggests that degree centrality does not capture enough information of the network architecture to sufficiently predict the network convergence speed.

### 7.4.3 Limitations

The results of the experiments in this chapter are limited in two ways. Firstly, as discussed in the previous section, we observed a pattern of decreased convergence times for odd values of network degree $k$. This is a direct result of the choice of jumps defining the circulant graphs. As multiple graphs can comply with the $k$-regular requirement, the obtained results are only representative for the specific circulant graphs used in our experiment setup. Secondly, the confidence intervals in the heatmap in Fig. 7.6 are not assessed. The confidence intervals are assumed to be sufficiently small for $p = 100$, because all other simulated graphs with $p = 100$ also resulted in sufficient confidence intervals.

## 7.5 Conclusion

This experiment was set out to investigate the influence of network architecture on the convergence speed and to gain insight into how diffusion matrix shares influence the convergence speed. A major finding of this experiment is that a static share of 0.5, as used by Kempe *et al.* [28], is not for all networks the optimal share that maximizes convergence speed. This experiment showed that in path graphs, RGGs with a small radius, and $k$-regular circulant graphs with small values of $k$, a static share in the range $[0.6, 0.8]$ results in the lowest convergence time. In addition, in the case of fully connected networks and $k$-regular circulant graphs with a high value of $k$, a static share of $\alpha_s = 0.5$ results in the lowest convergence times. From this, we conclude that sparsely-connected graphs benefit the most from using tailored diffusion matrix shares, which are larger than 0.5. When using static shares, densely-connected graphs perform best with a static share of $\alpha_s = 0.5$.

A second major finding of this experiment is that the network architecture is an important factor influencing the convergence speed of a network. Throughout the different used graphs in this experiment, we observe a general trend that sparsely-connected graphs result in significantly lower convergence times, compared to densely-connected graphs. Moreover, the node degrees in a network, which are

related to the network density, cannot be used as predictors for the convergence speed. The convergence speed of the assessed gossip protocols showed irregular increases in convergence time when the network degree $k$ increases. From this we conclude that degree centrality is too limited for the purpose of predicting the convergence speed. However, other types of centrality might still be suitable for this purpose.

For future research we propose an alternative experiment setup for the $k$-regular graphs, which expectedly does not result in the artifacts as observed in Fig. 7.6. Such an experiment could be set up as follows. Start with a connected 2-regular graph, which is by definition a circle graph. For each subsequent value $k+1$, divide the graph with network degree $k$ in two equally sized subgraphs. Let each node in one of the subgraphs form randomly a new connection with a node in the other subgraph, to get the next graph with network degree $k+1$. This process must fulfill three conditions: Each node $i$ in both equally sized subgraphs has at least one node in the other subgraph that is not connected to $i$; Each node only forms a single new connection per iteration; and the network size $N$ must be even. This experiment setup should reduce the observed artifacts, because nodes making a new connection to a nearby node is equally likely as making a new connection with a node further away. Evaluating sufficiently many of these (randomly generated) graphs per network degree $k$, would gain more insight in how the network degree affects the convergence speed on average.

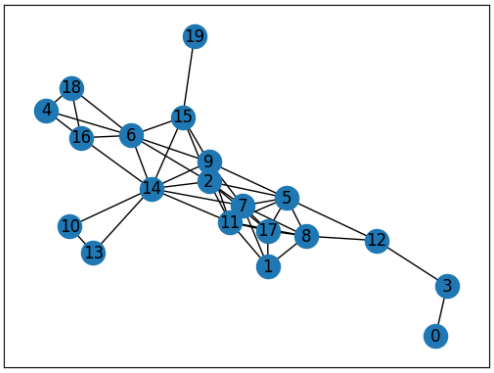
# 8 Initial value variance and convergence speed

This chapter investigates the influence of the variance of the input values of a gossip algorithm on the convergence speed. During the execution of the experiments of Chapter 7, a couple of seemingly similar RGGs showed significant different convergence times. This observation suggests that the network architecture is not the only significant factor for the convergence speed, and led to the idea that the variance of initial values might also be of influence. We hypothesize that regardless of the node degree distribution, the convergence time in a network is related to the variance of the initial values. The lower the variance, the lower the convergence time. This chapter assesses this hypothesis by applying normal distributions with different variances to generate input values for the gossip algorithm. Two types of networks are evaluated: a sparsely-connected RGG and a fully connected graph. The main result of this chapter is an existing positive relationship between convergence speed and the input values set variance. The remainder of this chapter is organized as follows: The used methodology is first explained in Section 8.1. Section 8.2 present the results of the experiments and these are discussed in Section 8.3. Lastly, Section 8.4 presents the conclusions of the experiments.

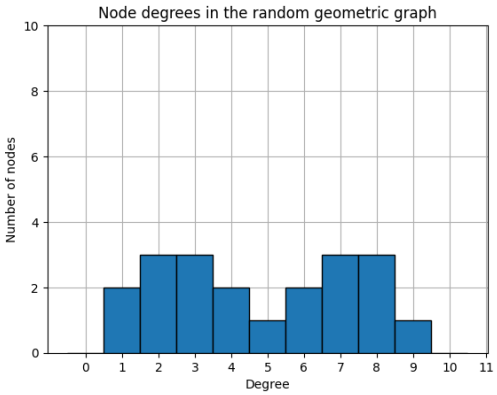
## 8.1 Methodology

Since the objective is to determine whether the convergence time is related to the variance of the input values regardless of the node degree distribution, two types of networks are assessed. The first network is an RGG, the type of network from which the hypothesis of this chapter arose. The second network is to verify whether similar results are obtained in a network with a monotonous degree distribution, namely a fully connected graph. If both graphs show a similar trend between the input value variance and the convergence speed, it suggests that the variance in the input values is a factor influencing the convergence speed of a network.

Fig. 8.1 shows the RGG used in this experiment and the distribution of the node degrees in this RGG. This RGG is chosen because of the relatively large variation




(a) RGG, with $r = 0.31$.




(b) The distribution of the node degrees.

**Fig. 8.1:** RGG and its distribution of node degrees.

in node degrees, which means that it has a differently architecture than the fully connected network with a monotonic degree distribution. Both networks have size $N = 20$.

The initial value sets are generated by using normal distributions. Each normal distribution can be characterized by two parameters: the mean value $\mu$ and a standard deviation $\sigma$. The mean value is the expected global average $E[\bar{x}]$ used in the previous experiments for convenience, i.e. $\mu = 25$. The variance of a set $X$ is related to the standard deviation as: $\mathrm{Var}[X] = \sigma^2$. The uniform distributions used in the previous experiments (Chapter 6 and 7) allowed setting upper bounds and lower bounds of the generated values. However, a normal distribution does not have such bounds, as it only gives probabilities of a generated value to lie in a certain range. Nevertheless, it is known that 68% of the values draw from a normal distribution lie in the range $(\mu - \sigma, \mu + \sigma)$. Similarly, 95% lie in the range $(\mu - 2\sigma, \mu + 2\sigma)$, and 99.7% lie in the range $(\mu - 3\sigma, \mu + 3\sigma)$. The experiments in this chapter follows the example of the temperature sensor network used in the previous experiments in Chapter 6 and 7. We consider 95% lying in a certain range is sufficient approximation of the bounds of the initial value set. This experiment evaluates value sets that have (approximate) bounds significant smaller than $[10, 40]$ used before, up to value sets with a larger variance than these bounds. The used standard deviations are $\sigma = 1, 3, 5, \ldots, 15$. This means that at the smallest variance ($\sigma = 1$) approximately 95% of the values lie within $(23, 27)$, and at the largest variance ($\sigma = 15$) approximately 95% of the values lie within $(-5, 55)$. For each assessed standard deviation $p = 100$ simulation runs are performed, as this results in sufficiently small confidence intervals.

## 8.2 Simulation results

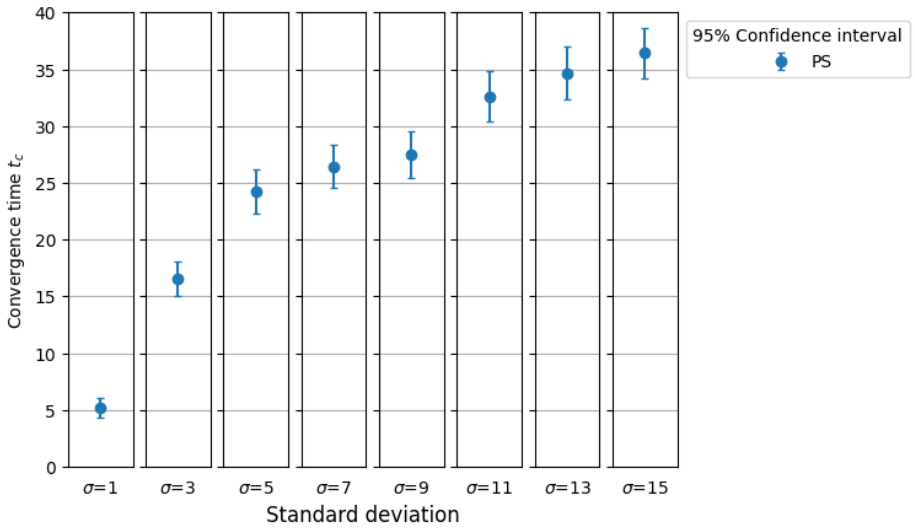



**Fig. 8.2:** The convergence time of the RGG in Fig. 8.1 with initial value sets generated using normal distributions with different standard deviations.

Fig. 8.2 shows the convergence times in an RGG for input sets generated with normal distributions that have different standard deviations. From this figure, we make the following observations:

1. For an increasing standard deviation the convergence time increases steadily, but not at a constant rate.
2. At lower standard deviations ($\sigma \leq 5$), the convergence time increases more rapidly than at higher standard deviations.

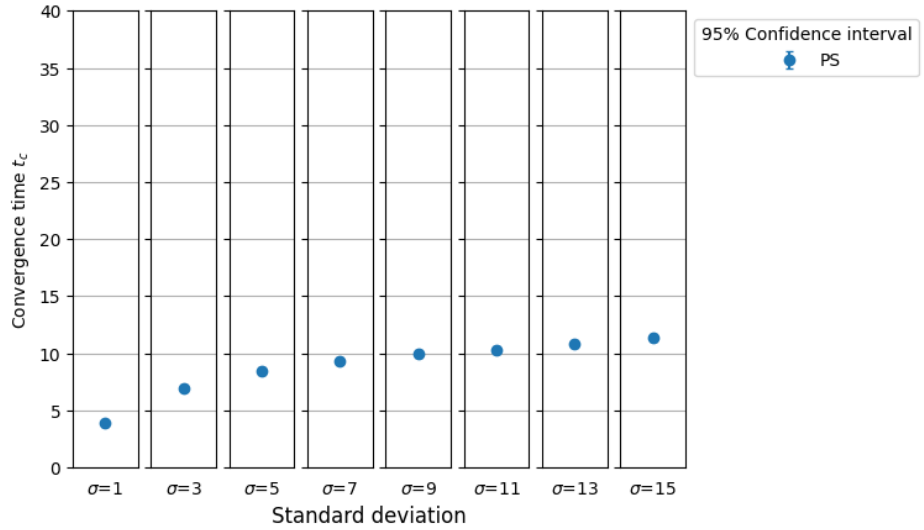



**Fig. 8.3:** The convergence time of a fully connected graph with $N = 20$ with initial value sets generated using normal distributions with different standard deviations.

In Fig. 8.3 the simulation results of initial value sets with different standard deviations in fully connected graphs are presented. The following aspects of this figure stand out:

1. For an increasing standard deviation, the convergence time increases steadily, but not at a constant rate.
2. As the standard deviation increases, the growing rate of the convergence time decreases.
3. All measured convergence times are significantly lower than the convergence times measured in the RGG.
4. The confidence intervals are not visible, and thus significantly smaller than the confidence intervals of the measurements in the RGG.

## 8.3 Discussion

This discussion section consists of two parts. Section 8.3.1 first discusses the interpretation and implications of the results. Subsequently, Section 8.3.2 elaborates on the limitations of the experiments in this chapter.

### 8.3.1 Interpretation and implications

The results of Fig. 8.2 and Fig. 8.3 confirm the hypothesis of this experiment. Both cases found a positive relation between the standard deviation of the initial value set and the convergence time of the network. In addition to this finding, note that an initial value set with a standard deviation of $\sigma = 0$ in this experiment would lead to a convergence time of $t_c = 0$ because all nodes have the true average at the start. This is in line with the positive relationship observed in the experiment. The found positive relationship supports the idea that the variance of the initial value set is a significant factor influencing the convergence speed of a network.

Although the two experiments in this chapter showed a similar relationship, the results also showed significantly lower convergence times in the fully connected network compared to the RGG. This finding can be a result of the network architecture also being a significant factor for the convergence speed (conclusion Chapter 7). The RGG and fully connected network differ much regarding how dense they are connected. Hence, this is most likely causing the difference in magnitude.

Another finding is that the confidence intervals in the RGG are significantly larger than the confidence intervals of the fully connected graph. One of the differences between the architecture of these graphs, is that the fully connected graph is highly symmetric, while the RGG is not. In the fully connected graph each node has a degree of 19, while there is a lot of variation in degrees in the RGG, as can be seen in Fig. 8.1. A possible explanation for the found difference in confidence intervals might be that certain initial value sets fit better to the architecture of this RGG than others. This is reflected in the variation of the measured convergence times.

### 8.3.2 Limitations

The experiments in this chapter are limited in a number of ways. Firstly, the number of evaluated standard deviations is limited. The used standard deviations are sufficient to observe a trend between the variance of the initial value set and the convergence time, but this experiment does not show a precise relationship. Secondly, the presented results of the RGG are limited to the specific RGG used in the experiment. While it gives representative results of an RGG with $r = 0.31$ and $N = 20$, other RGGs with the same radius and network size did not result in exactly the same average convergence times over the range of standard deviations. However, three other assessed RGGs, which are not presented in this chapter, showed similar positive relationships. This supports the discussed findings and implications

of this experiment. Thirdly, the used standard deviation represents the parameter that defines the normal distribution for the generation of random values, and not the exact standard deviation of the generated initial value set. When a normal distribution with $\mu$ and $\sigma$ is used in a random experiment, it is known that when the number of trials approaches infinity, the set of outcomes approaches a normal distribution with mean $\mu$ and standard deviation $\sigma$. Yet, the number of generated values in the experiments of this chapter is small with $N = 20$. Therefore, the actual distribution of the generated initial values approximates a normal distribution with $\mu$ and $\sigma$, but deviates from this to some extent. This could have resulted in small artifacts in the presentation of the results but this is not assessed explicitly. A better way to present the relationship between the initial value set variance and convergence time is with a scatter plot. In such a plot, each individual observation can be marked, which makes the spread in the standard deviations of the generated initial value sets clear.

## 8.4 Conclusion

The purpose of the experiment of this chapter was to evaluate how the variance of the initial value set relates to the convergence time of a network. The two assessed networks, a sparsely-connected RGG and a fully connected network, showed a (non-linear) relation between the convergence time and the standard deviation. This confirms the hypothesis that regardless of the node degree distribution, the variance of the initial values is related to the convergence time of a network. Therefore, the main conclusion of the experiments in this chapter is that the variance of the initial value set is a significant factor that affects to convergence speed of a network. Hence, in applications that require bounds on the convergence time of an algorithm, the variance of the initial value is one of the factors that should be considered when one wants to make a proper estimation of the convergence times inside a network.

# 9 Discussion, Conclusion, and Future Work

This chapter first discusses the limitations of this research and the implications of the findings of this thesis for the energy context in Section 9.1. Subsequently, the conclusions of this work are discussed in Section 9.2. The last section, Section 9.3, offers directions for future research.

## 9.1 Discussion

This section consists of two parts: Section 9.1.1 describes general limitations of this research and Section 9.1.2 discusses the implications of this research for the implementation of communication mechanisms in energy networks.

### 9.1.1 Limitations

Next to the limitations discussed in Chapter 6, 7, and 8, this research also has a number of general limitations. Firstly, this study only considered networks with $N = 20$ nodes. While this network size is large enough to show certain concepts and to investigate behavior of gossip protocols, this study lacks the verification in larger networks. Secondly, this research is limited by the graphs that were used. For example, in this thesis the RGG has a prominent role. This kind of graph has a good resemblance with wireless networks, but wired networks most likely have different architectures. While this research is intended to make deductions that are generally applicable, wired network architectures are not evaluated, and thus this research cannot give full guarantees for the similar results in different network types. Lastly, this research has only considered unidirectional gossip in the experiments. Bidirectional gossip protocols, such as pairwise gossip, are also popular in research and have different characteristics with respect to convergence and how the algorithms execute. Therefore, the findings of this research cannot be used to derive conclusions for bidirectional gossip protocols.

### 9.1.2 Implications

The objective of this thesis is motivated by DSM offering a solution to grid congestion. This section discusses how the obtained results presented in this thesis can be used for the implementation of DSM in energy networks.

One of the major results presented in Chapter 6 and Chapter 7 is that network density is a significant factor affecting the convergence speed in a network. Sparsely-connected networks result in significant larger convergence times compared to densely-connected networks. This is important to consider in the design of communication algorithms applied in communication networks for distributed energy networks. In a densely-connected energy network, little improvement can be achieved by customizing the used gossip protocol. Hence, in that case, established gossip protocols like Push-Sum are most likely a good choice because they are relatively simple and do not underperform compared to the customized versions of gossip protocols that were investigated in this thesis. However, if for some reason many nodes crash or communication links disappear, and as a result the communication network becomes more sparsely connected, then it could be beneficial to switch to a customized protocol. For such circumstances, the results of this thesis suggest that two aspects influence the convergence speed: the diffusion matrix shares and properties of the network. Hence, these aspects should be considered and further investigated, for the design of communication mechanisms that operate in energy networks. In addition, this research shows that degree centrality is limited and did not improve the network convergence speed. Hence, further research is needed to utilize diffusion matrix shares in the optimizing convergence speed.

The second important implication of the findings of this research for energy networks concerns the variance of input values. The demonstrated relation between the variance and the convergence speed in Chapter 8 is valuable for an energy distribution context. Peak-shaving algorithms used in DSM, like Profile Steering, are intended to optimize the global energy usage by using the flexibility on the demand side of the energy network. Such optimization algorithms include finding an optimal global power schedule by adjusting power schedules of individual nodes. If the MG consists of many similar participants, for example a neighborhood of regular houses, the variation between energy schedules is likely to be little. Compared to an MG consisting of a combination of low energy-consuming households, moderate energy-consuming corporate buildings, and large energy-consuming factories, the variation between the energy schedules is significantly larger. According to our findings, the latter scenario most likely converges slower due to the larger variation. Hence, in the decisions about what communication protocol to use in such environments,

one should consider the expected variation between schedules with time-critical applications because it is a significant factor influencing the convergence speed. Furthermore, DSM optimization algorithms aim to decrease the variance between power schedules, making the variance between the individual schedules decrease during the operation of the optimization algorithm. Hence, based on the findings of this research we expect a correlation between the convergence speed of the network and iteration of the optimization algorithm. One can therefore expect that the convergence time of the gossip protocol decreases, as the optimization algorithm proceeds.

## 9.2 Conclusions

The main objective of this research was to explore the influence of communication mechanisms and network properties on the convergence speed of a network. Several subjects are investigated, first we summarize the contributions of this thesis for each sub question of this research and conclude this section by answering the main research question.

1. *How could a network converge from different partial views to a shared global view of the network?*

This thesis presented an overview of various gossip protocols which are suitable solutions to reach network convergence in distributed networks. The most important requirements to guarantee network convergence is that a communication protocol adheres to the mass conservation and the stable average properties. Many types of gossip protocols exist in literature, all having their advantages and disadvantages, which are discussed in Chapter 3. In addition, Chapter 3 introduced a framework to characterize any gossip protocol based on network objective, casting type, directionality, and type of partner selection. Furthermore, this chapter explained how different gossip protocols work and how they differ from each other. This can be used as reference guide to select a type of gossip protocol that suits the requirements of certain applications best and is also a useful introduction to the area of gossip protocols. This thesis used unidirectional gossip, based on Weighted Gossip, to investigate factors that influence the convergence speed of a network.

2. *How can the convergence speed of a network be improved by the design of communication algorithms?*

Two protocols, DWG and PS*, are proposed in Chapter 6, aimed at utilizing knowledge about the degrees of nodes to improve the convergence speed. Both protocols turned out to perform worse than standard PS. The experiments with these protocols identified that larger shares of the diffusion matrix do not lead to faster convergence at receiving nodes. Besides, increasing the probability that high-degree nodes get selected, does not result in faster network convergence. Hence, the major finding is that standard PS performs better than the proposed protocols that use degree-based prioritization regarding convergence speed, and also results in more stable behavior between all runs, giving more certainty in predictions on the convergence speed.

3. *How does the density of a network relate to the convergence speed of the network?*

The relation between the density of a network and the network convergence speed is assessed in multiple ways in this research. This thesis assessed this relation first in Chapter 6 by means of varying the radius of RGGs. From the results we conclude that low radii close to the critical radius result in significant higher convergence times for all assessed gossip protocols, compared to larger radii result. What is more, in very densely-connected RGGs ($r > 0.83$) the differences between the gossip protocols become negligible. Therefore, sparsely-connected networks have the most potential for convergence speed optimization.

The second way this thesis investigated the relation between graph density and convergence speed is with $k$-regular graphs in Chapter 7. We observed irregular steps in convergence times at the lowest network degrees ($k \leq 10$), which is caused by the network setup. However, a general trend was visible: Sparsely-connected $k$-regular graphs result in significant larger convergence times than densely-connected graphs. In addition, results of the extreme cases of a path graph and a fully connected graph in Chapter 7 are consistent with this finding as well. Hence, the research has found a recurring trend that the density of a network is a significant factor for the convergence speed of a network.

The unexpected outcome of the significant difference between convergence times in odd values of network degree $k$ compared to adjacent even values of $k$ found in the experiment with $k$-regular graphs, does implicate the importance of network architecture to the prediction of convergence speed. While node degrees turned out to be unsuitable for this prediction, other ways to characterize network architecture might be helpful in predicting network convergence speed.

4. *How does the variation in input values influence the convergence speed of a network?*

The experiments from Chapter 8 regarding the relation between input value set variance and convergence speed identified the existence of a non-linear relationship. A sparsely-connected RGG and fully connected graph both show a similar increasing relationship. This result confirms the hypothesis that the variation of the initial value set is a significant factor influencing the convergence speed of a network, regardless of the density or degree distribution of a graph.

5. *How can the results of the researched topics of guaranteeing network convergence, algorithm design, network topology, and the input value set be used in the implementation of Demand-Side Management in energy networks?*

The related literature research and the theoretical background on gossip protocols in this thesis highlight the potential for gossip protocols being a suitable tool for communication algorithms inside distributed networks. Although distributed networks are not applied for energy management in reality yet, this research identified a couple of important factors to consider when gossip-based communication is going to be applied for Demand-Side Management in the future. Firstly, the density of the network is a significant factor determining the convergence speed. Sparsely-connected graphs have the most potential to achieve improvement with the design of the gossip protocol, as these graphs result in significantly larger convergence times. This research observed the largest differences in performance in sparsely-connected graphs. On the other hand, in densely-connected networks an established gossip protocols like Push-Sum might suffice. All gossip protocol showed equal performance in densely-connected graphs and Push-Sum has the advantage of simplicity in implementation. In addition, the composition of participants in a network and their energy consumption profiles are another important factor to consider. Larger variation of energy profiles will result in larger convergence times. Besides, a correlation between the iteration number of the optimization algorithm and the convergence of the gossip algorithm can play a role for the design of gossip algorithms used in energy networks.

Finally, we form an answer to the main research question by summarizing the conclusions and the contributions of this research:

> *How can communication mechanisms and network properties be utilized to improve convergence speed, to aid the implementation of Demand-Side Management in distributed energy networks?*

This thesis contributes in a number of ways to the body of research that studies the convergence speed of gossip-based communication. Firstly, the findings of this

research imply that the density of a network and the variance of the input values are significant factors that affect the convergence speed of a network. Hence, these factors are of interest for the design of gossip-based communication that must operate with certain timing requirements. Secondly, this thesis established a framework to characterize existing gossip protocols, which is useful to understand the different versions of gossip protocols and their related advantages and disadvantages. Thirdly, this thesis has provided a deeper insight into the working of unidirectional gossip, and how the diffusion matrix shares influence the convergence speed in a network. Fourthly, this thesis attempted to improve the convergence speed by utilizing the degrees of a network in diffusion matrix shares and neighbor selection. The results imply that degree centrality is insufficient for the purpose of improving the convergence speed. Lastly, the findings of this work identified important factors to consider when gossip-based communication is applied in distributed energy networks, namely the network density, expected variation in energy schedules and the correlation with the optimization algorithm.

## 9.3 Future work

This research has brought up several directions open for future research. The following paragraphs suggest directions for future research.

**Convergence of lower-degree nodes:** Future research that focuses on improving the individual convergence speed of lower-degree nodes could be fruitful for improving the network convergence speed of networks. The hypothesis of Chapter 6 was misdirected with the focus on improving the individual convergence speed of higher-degree nodes to improve the overall network convergence speed. The results of individual convergence speeds per node degree in Section 6.4.2 showed significant slower individual convergence of lower-degree nodes, compared to higher-degree nodes. Therefore, the topic of individual convergence speed of lower-degree nodes is interesting to investigate further. Section 6.6 already suggested an idea to improve the individual convergence of lower-degree nodes, namely by applying an algorithm with two phases. In the first phase, standard PS is used for communication in the network with the intention to let higher-degree nodes converge. In the second phase, the lower degree nodes can be prioritized by using large outward shares of the higher-degree nodes. According to the analysis of Scenario 3, presented in Chapter 4, this would increase convergence speed of the lower-degree nodes, since the higher-degree nodes have a good estimate of the global average.

**Other centrality types:** As Section 6.5.4 and Section 7.5 already suggest, other forms of centrality should be explored in future research. The usage of degree centrality, as applied in this research, did not improve network convergence and turned out to be of limited use to improve individual convergence of nodes. Other centrality types, such as closeness centrality, betweenness centrality, and eigenvector centrality, could result in a better ranking of nodes in a network which could potentially be more effective to improve convergence speed.

***k*-Regular graphs:** The experiment with $k$-regular graphs in Chapter 7 showed artifacts in the results, due to the setup of the experiment. Section 7.4.2 offered an explanation for the observations, and Section 7.5 suggested a different experiment setup which most likely does not result in similar artifacts. Further research applying such an experiment setup could be meaningful to further support the findings of this research and obtain a deeper understanding of the influence of node degrees.

**Ordering of initial values:** Chapter 8 investigated the influence of variance of the initial value set on the convergence speed. In the experiments initial value sets where generated randomly and assigned to nodes in the order in which they were generated. However, the experiments did not consider if the order in which the generated initial values were assigned to nodes also affects the convergence speed. In other words, maybe certain orderings of initial values fit better to a network topology than other orderings. The relation between the ordering of initial value sets and network topology is an interesting direction for future research.

**Simulations with realistic MGs:** This research mainly used abstract network types. The results of these networks give deeper insight in the operation of gossip protocols and network convergence, but do not all translate conveniently to real-life networks. Simulations with network architectures suitable for MG, would be meaningful for further investigation of the application of gossip-based communication. For example, [29] and [30] use small MG that resemble the architecture of electricity grids. In addition, a potential use-case with a larger MG could be a LV distribution grid in a neighborhood of households. Such a network consists of approximately 150 households [51], which could be a realistic MG in which nodes cooperate to balance the load to the grid. Investigating such use-cases would be valuable additions to the body of work about the application of gossip-based communication in distributed energy networks.

**Knowledge about individual convergence:** In real-life networks it is relevant for nodes to know whether they have converged, or how reliable their estimate of the global average is. This topic is out of the scope of this research, but is nonetheless an integral challenge to be solved. Especially for large networks where nodes lack

knowledge of the complete network architecture, it is an interesting topic to explore. Strategies for solving this challenge are required in order to apply gossip-based communication in practice.

**Correlation between optimization algorithm and communication protocol:** Section 9.1.2 explained the expected correlation between the iterations of the optimization algorithm and the convergence speed of the gossip-based communication. This correlation is an interesting topic to investigate in future research. Multiple studies already considered the combination of the optimization algorithm and convergence of gossip protocol [11, 30], but did not assess if the convergence speed changes during the optimization process. Because the optimization algorithm and the communication protocol are strongly intertwined in DSM approaches, further research on this subject could be of great help to improve the operation of DSM with timing restrictions.

# Bibliography


[1] *Capaciteitskaart elektriciteitsnet*. https://capaciteitskaart.netbeheernederland.nl/. Accessed on 13 December 2023 (cit. on p. 2).

[2] C. Schwaegerl and L. Tao. "The Microgrids Concept". In: ISBN: 9781118720677. John Wiley and Sons Ltd, 2014. Chap. 1, p. 5 (cit. on p. 3).

[3] Gerwin Hoogsteen. "A Cyber-Physical Systems Perspective on Decentralized Energy Management". PhD thesis. PO Box 217, 7500 AE Enschede, The Netherlands: University of Twente, Dec. 2017 (cit. on pp. 3, 4).

[4] IEA. *Technology Roadmap - Smart Grids*. https://www.iea.org/reports/technology-roadmap-smart-grids. Accessed on 30 November 2023 (cit. on p. 4).

[5] Yanling Zheng and Qingshan Liu. "A review of distributed optimization: Problems, models and algorithms". In: *Neurocomputing* 483 (2022), pp. 446–459 (cit. on p. 5).

[6] Alexandros G. Dimakis, Soummya Kar, José M. F. Moura, Michael G. Rabbat, and Anna Scaglione. "Gossip Algorithms for Distributed Signal Processing". In: *Proceedings of the IEEE* 98.11 (2010), pp. 1847–1864 (cit. on p. 10).

[7] Márk Jelasity. "Gossip". In: *Self-organising Software: From Natural to Artificial Adaptation*. Ed. by Giovanna Di Marzo Serugendo, Marie-Pierre Gleizes, and Anthony Karageorgos. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011. Chap. 7, pp. 139–162 (cit. on p. 10).

[8] The Apache Software Foundation. *Open Source NoSQL Database*. https://cassandra.apache.org/_/index.html. Accessed on 28 November 2023 (cit. on p. 10).

[9] DataStax. *DS201.10 Gossip | Foundations of Apache Cassandra*. https://www.youtube.com/watch?v=vEk3VDC0J7k. Accessed on 28 November 2023 (cit. on p. 10).

[10] DataStax. *Apache Cassandra 3.0: Internode communications (gossip)*. https://docs.datastax.com/en/cassandra-oss/3.0/cassandra/architecture/archGossipAbout.html. Accessed on 28 November 2023 (cit. on p. 10).

[11] Angelia Nedić, Alex Olshevsky, and Michael G. Rabbat. “Network Topology and Communication-Computation Tradeoffs in Decentralized Optimization”. In: *Proceedings of the IEEE* 106.5 (2018), pp. 953–976 (cit. on pp. 13, 20, 124).

[12] Wencong Su and Jianhui Wang. “Energy Management Systems in Microgrid Operations”. In: *The Electricity Journal* 25.8 (2012), pp. 45–60 (cit. on pp. 17–19).

[13] Jose Maurilio Raya-Armenta, Najmeh Bazmohammadi, Juan Gabriel Avina-Cervantes, et al. “Energy management system optimization in islanded microgrids: An overview and future trends”. In: *Renewable and Sustainable Energy Reviews* 149 (2021), p. 111327 (cit. on pp. 17–19).

[14] Daniel E. Olivares, Claudio A. Cañizares, and Mehrdad Kazerani. “A Centralized Energy Management System for Isolated Microgrids”. In: *IEEE Transactions on Smart Grid* 5.4 (2014), pp. 1864–1875 (cit. on p. 19).

[15] Marco E. T. Gerards, Hermen A. Toersche, Gerwin Hoogsteen, et al. “Demand side management using profile steering”. In: *2015 IEEE Eindhoven PowerTech*. 2015 (cit. on pp. 19, 20, 73).

[16] F. Tangerding, I.A.M. Varenhorst, G. Hoogsteen, M.E.T. Gerards, and J.L. Hurink. “GridShield: A Robust Fall-Back Control Mechanism for Congestion Management in Distribution Grids”. English. In: *2022 IEEE PES Innovative Smart Grid Technologies Conference Europe (ISGT-Europe)*. United States: IEEE, Nov. 2022 (cit. on p. 19).

[17] Nayara Aguiar, Anamika Dubey, and Vijay Gupta. “Network-Constrained Stackelberg Game for Pricing Demand Flexibility in Power Distribution Systems”. In: *IEEE Transactions on Smart Grid* 12.5 (2021), pp. 4049–4058 (cit. on p. 19).

[18] Xiao Kou, Fangxing Li, Jin Dong, et al. “A Scalable and Distributed Algorithm for Managing Residential Demand Response Programs Using Alternating Direction Method of Multipliers (ADMM)”. In: *IEEE Transactions on Smart Grid* 11.6 (2020), pp. 4871–4882 (cit. on p. 19).

[19] Verena Menzel, Johann L. Hurink, and Anne Remke. “Securing SCADA networks for smart grids via a distributed evaluation of local sensor data”. In: *2021 IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids (SmartGridComm)*. 2021, pp. 405–411 (cit. on p. 19).

[20] Verena Menzel, Nataly Bañol Arias, Johann L. Hurink, and Anne Remke. “Securing Smart Grids Locally using a Power Flow-based Intrusion Detection System”. In: *2023 IEEE Belgrade PowerTech*. 2023 (cit. on p. 19).

[21] Verena Menzel, Kai Oliver GroBhanten, and Anne Remke. “Evaluating a Process-Aware IDS for Smart Grids on Distributed Hardware”. In: *2023 IEEE International Conference on Cyber Security and Resilience (CSR)*. 2023, pp. 418–425 (cit. on p. 19).

[22] Daniele Croce, Fabrizio Giuliano, Ilenia Tinnirello, et al. “Overgrid: A Fully Distributed Demand Response Architecture Based on Overlay Networks”. In: *IEEE Transactions on Automation Science and Engineering* 14.2 (2017), pp. 471–481 (cit. on pp. 20, 21).

[23] Aditya Pappu, Gerwin Hoogsteen, and Johann L. Hurink. “Distributed Co-operative Demand Side Management for Energy Communities”. In: *2022 IEEE PES Innovative Smart Grid Technologies Conference Europe (ISGT-Europe)*. 2022 (cit. on p. 20).

[24] K. De Brabandere, K. Vanthournout, J. Driesen, G. Deconinck, and R. Belmans. “Control of Microgrids”. In: *2007 IEEE Power Engineering Society General Meeting*. 2007 (cit. on p. 20).

[25] A. Krkoleva, V. Borozan, A. Dimeas, and N. Hatziargyriou. “Gossip based message dissemination schemes in future power systems”. In: *2011 16th International Conference on Intelligent System Applications to Power Systems*. 2011 (cit. on p. 21).

[26] A. Krkoleva, V. Borozan, A. Dimeas, and N. Hatziargyriou. “Requirements for implementing gossip based schemes for information dissemination in future power systems”. In: *2011 2nd IEEE PES International Conference and Exhibition on Innovative Smart Grid Technologies*. 2011 (cit. on p. 21).

[27] Filipe Campos, Miguel Matos, José Pereira, and David Rua. “A peer-to-peer service architecture for the Smart Grid”. In: *14-th IEEE International Conference on Peer-to-Peer Computing*. 2014 (cit. on p. 21).

[28] D. Kempe, A. Dobra, and J. Gehrke. “Gossip-based computation of aggregate information”. In: *44th Annual IEEE Symposium on Foundations of Computer Science, 2003. Proceedings.* 2003, pp. 482–491 (cit. on pp. 21, 29, 33, 34, 36, 55, 95, 96, 108).

[29] Despina I. Koukoula and Nikos D. Hatziargyriou. “Convergence acceleration of gossip protocols applied for decentralized distribution grid management”. In: *2015 IEEE Eindhoven PowerTech*. 2015 (cit. on pp. 21, 123).

[30] Jonas Engels, Hamada Almasalma, and Geert Deconinck. “A distributed gossip-based voltage control algorithm for peer-to-peer microgrids”. In: *2016 IEEE International Conference on Smart Grid Communications (SmartGridComm)*. 2016, pp. 370–375 (cit. on pp. 21, 22, 123, 124).

[31] Maurizio Di Bisceglie, Silvia Liberata Ullo, and Alfredo Vaccaro. “The role of cooperative information spreading paradigms for Smart Grid monitoring”. In: *2012 16th IEEE Mediterranean Electrotechnical Conference*. 2012, pp. 814–817 (cit. on p. 22).

[32] Nancy A. Lynch. *Distributed Algorithms*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1996, pp. 21–22, 197 (cit. on pp. 27, 67).

[33] Florence Bénézit, Vincent Blondel, Patrick Thiran, John Tsitsiklis, and Martin Vetterli. “Weighted Gossip: Distributed Averaging using non-doubly stochastic matrices”. In: *2010 IEEE International Symposium on Information Theory*. 2010, pp. 1753–1757 (cit. on pp. 29, 32, 33, 36, 37, 39, 55, 62, 67, 81).

[34] Ji Liu, Shaoshuai Mou, A. Stephen Morse, Brian D. O. Anderson, and Changbin Yu. “Deterministic Gossiping”. In: *Proceedings of the IEEE* 99.9 (2011), pp. 1505–1524 (cit. on pp. 29, 36, 39, 42, 47, 52).

[35] Bernhard Haeupler. “Simple, Fast and Deterministic Gossip and Rumor Spreading”. In: *J. ACM* 62.6 (Dec. 2015) (cit. on pp. 29, 30, 44, 47).

[36] Benjamin Doerr, Tobias Friedrich, and Thomas Sauerwald. “Quasirandom Rumor Spreading”. In: *ACM Trans. Algorithms* 11.2 (Oct. 2014) (cit. on pp. 29, 48).

[37] Petra Berenbrink, Robert Elsässer, and Thomas Sauerwald. “Communication Complexity of Quasirandom Rumor Spreading”. In: *Algorithmica* 72.2 (June 2015), pp. 467–492 (cit. on pp. 29, 48).

[38] Tuncer C. Aysal, Mehmet E. Yildiz, Anand D. Sarwate, and Anna Scaglione. “Broadcast gossip algorithms: Design and analysis for consensus”. In: *2008 47th IEEE Conference on Decision and Control*. 2008, pp. 4843–4848 (cit. on pp. 29, 42, 43, 49, 50).

[39] Keren Censor-Hillel and Hadas Shachnai. “Fast information spreading in graphs with large weak conductance”. In: *SIAM Journal on Computing* 41.6 (2012), pp. 1451–1465 (cit. on pp. 29, 36, 39, 51).

[40] A.M. Frieze and G.R. Grimmett. “The shortest-path problem for graphs with random arc-lengths”. In: *Discrete Applied Mathematics* 10.1 (1985), pp. 57–77 (cit. on p. 30).

[41] David Picard, Jérôme Fellus, and Stéphane Garnier. “Non asymptotic bounds in asynchronous sum-weight gossip protocols”. In: *arXiv preprint arXiv:2111.10248* (2021) (cit. on pp. 32, 33).

[42] S. Boyd, A. Ghosh, B. Prabhakar, and D. Shah. “Gossip algorithms: design, analysis and applications”. In: *Proceedings IEEE 24th Annual Joint Conference of the IEEE Computer and Communications Societies.* Vol. 3. 2005, 1653–1664 vol. 3 (cit. on p. 39).

[43] Paul E. Black. *Dense graph*. https://xlinux.nist.gov/dads/HTML/densegraph.html. Accessed on 9 September 2024 (cit. on p. 52).

[44] Paul E. Black. *Sparse graph*. https://xlinux.nist.gov/dads/HTML/sparsegraph.html. Accessed on 9 September 2024 (cit. on p. 52).

[45] Gopinath Srinivasan. *Vertex connectivity*. https://xlinux.nist.gov/dads/HTML/vertexConnectivity.html. Accessed on 9 September 2024 (cit. on p. 53).

[46] Gopinath Srinivasan. *Edge connectivity*. https://xlinux.nist.gov/dads/HTML/edgeConnectivity.html. Accessed on 9 September 2024 (cit. on p. 53).

[47] Josep Díaz, Dieter Mitsche, and Xavier Pérez-Giménez. “Dynamic Random Geometric Graphs”. In: *CoRR* abs/cs/0702074 (2007). arXiv: cs/0702074 (cit. on pp. 54, 81, 82).

[48] Team SimPy. *SimPy. Discrete event simulation for Python*. https://simpy.readthedocs.io/en/latest/. Accessed on 4 September 2024 (cit. on p. 68).

[49] NetworkX developers. *NetworkX. Network Analysis in Python*. https://networkx.org/. Accessed on 4 September 2024 (cit. on p. 68).

[50] MathWorks. *SimEvents*. https://www.mathworks.com/products/simevents.html. Accessed on 4 September 2024 (cit. on p. 68).

[51] *Basisinformatie over energie-infrastructuur*. https://www.netbeheernederland.nl/nieuws/informatie-over-energienetten-handig-bij-elkaar-gebracht-1279. Accessed on 8 January 2024 (cit. on p. 123).

# Appendix

# A

## A.1 Individual convergence per node degree for DWG, PS*, and DWG*

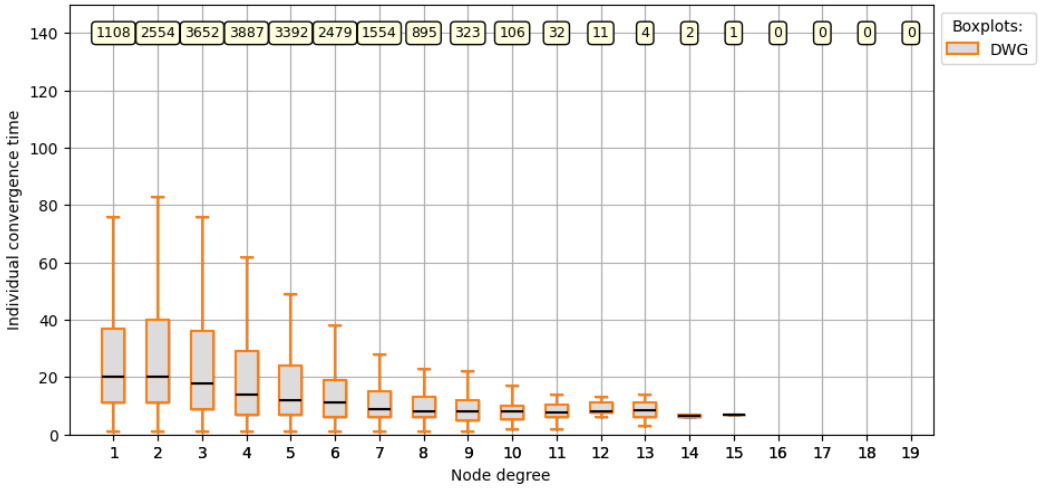



**Fig. A.1:** Box plots of individual convergence times per node degree of DWG in RGGs ($r = 0.31$). The results are obtained with $p = 1000$ simulation runs.

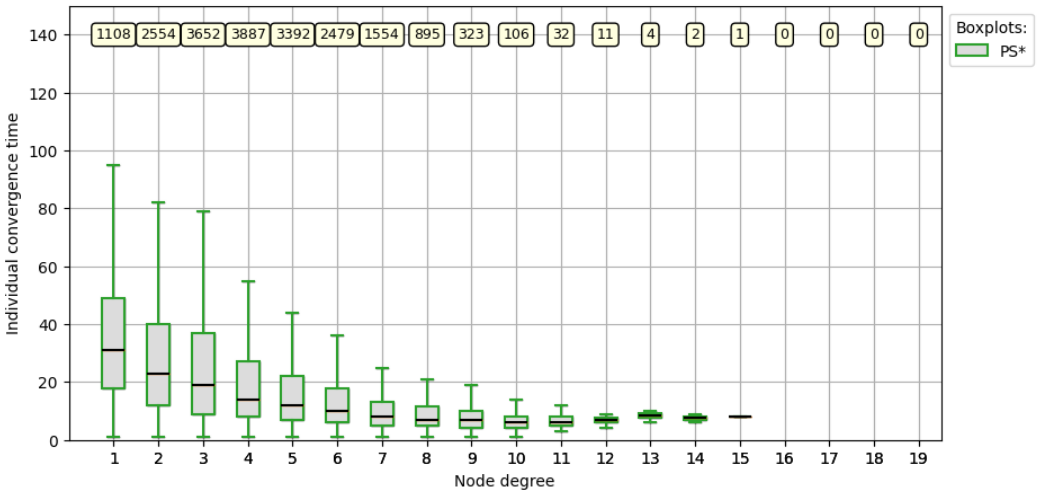



**Fig. A.2:** Box plots of individual convergence times per node degree of PS* in RGGs ($r = 0.31$). The results are obtained with $p = 1000$ simulation runs.

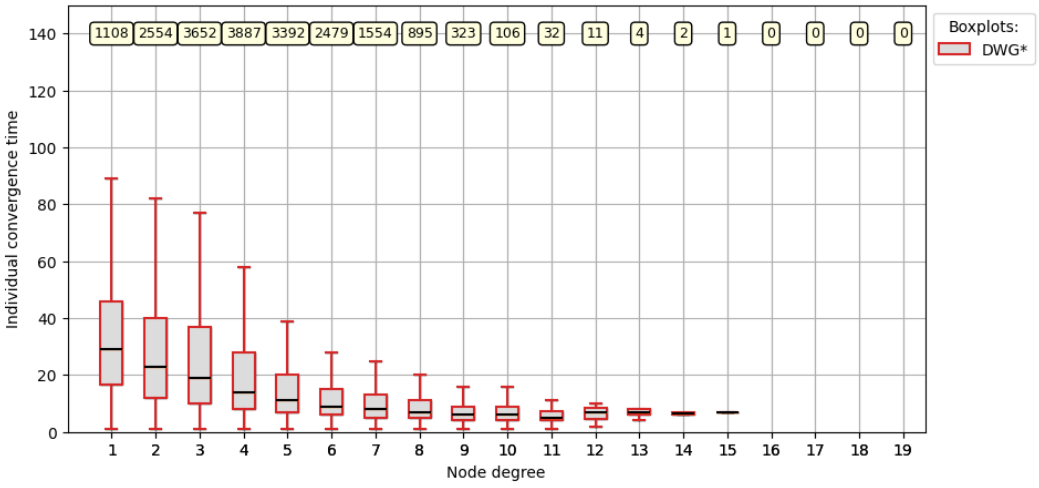



**Fig. A.3:** Box plots of individual convergence times per node degree of PS* in RGGs ($r = 0.31$). The results are obtained with $p = 1000$ simulation runs.

# Colophon

This thesis was typeset with LaTeX $2_\varepsilon$. It uses the *Clean Thesis* style developed by Ricardo Langner. The design of the *Clean Thesis* style is inspired by user guide documents from Apple Inc.

Download the *Clean Thesis* style at `http://cleanthesis.der-ric.de/`.