KTH Electrical Engineering

Optimization and Control in Dynamical Network Systems

HÅKAN TERELIUS

Doctoral Thesis
Stockholm, Sweden 2016

KTH School of Electrical Engineering
Department of Automatic Control
SE-100 44 Stockholm
SWEDEN

TRITA-EE 2016:167
ISSN 1653-5146
ISBN 978-91-7729-144-2

Akademisk avhandling som med tillstånd av Kungliga Tekniska högskolan framlägges till offentlig granskning för avläggande av teknologie doktorsexamen i elektro- och systemteknik fredagen den 11 november 2016, klockan 14.00 i sal F3, Kungliga Tekniska högskolan, Lindstedtsvägen 26, Stockholm.

Tryck: Universitetsservice US AB

Abstract

Dynamical network systems are complex interconnected systems useful to describe many real world problems. The advances in information technology has led the current trend towards connecting more and more systems, creating "intelligent" systems, where the intelligence originates in the scale and complexity of the network. With the growing scale of networked systems comes also higher demands on performance and continuous availability and this creates the need for optimization and control of network systems. This thesis makes four important contributions in this area.

In the first contribution, we consider a collaborative road freight transportation system. An efficiency measure for the road utilization in collaborative transportation scenarios is introduced, which evaluates the performance of collaboration strategies in comparison to an optimal central planner. The efficiency measure is used to study a freight transport simulation in Germany and taxi trips using real data from New York City. This is followed by a study of the optimal idling locations for trucks, and the optimal locations for distribution centers. These locations are then exploited in a simulation of a realistic collaborative freight transport system.

The second contribution studies the important problem of gathering data that are distributed among the nodes in an anonymous network, i.e., a network where the nodes are not endowed with unique identifies. Two specific tasks are considered: to estimate the size of the network, and to aggregate the distribution of local measurements generated by the nodes. We consider a framework where the nodes require anonymity and have restricted computational resources. We propose probabilistic algorithms with low resource requirements, that quickly generate arbitrarily accurate estimates. For dynamical networks, we improve the accuracy through a regularization term which captures the trade-off between the reliability of the gathered data and a-priori assumptions for the dynamics.

In the third contribution, a peer-to-peer network is utilized to improve a live-streaming media application. In particular, we study how an overlay network, constructed from simple preference functions, can be used to build efficient topologies that reduce both network latency and interruptions. We present necessary and sufficient convergence conditions, as well as convergence rate estimates, and demonstrate the improvements for a real peer-to-peer video streaming application.

The final contribution is a distributed optimization algorithm. We consider a distributed multi-agent optimization problem of minimizing the sum of convex objective functions. A decentralized optimization algorithm is introduced, based on dual decomposition, together with the subgradient method for finding the optimal solution. The convergence rate is analyzed for different step size rules, constant and time-varying communication delays, and noisy communication channels.

Sammanfattning

Dynamiska nätverkssystem är komplexa sammankopplade system med många praktiska tillämpningar. Den snabba utvecklingen inom informationsteknologin har drivit trenden att sammankoppla större och större system till nätverk av “intelligenta” system, där intelligensen kommer från komplexiteten av nätverken. Med den ökande storleken på nätverkssystemen kommer också ökade krav på dess prestanda och tillgänglighet, vilket är drivkraften bakom utvecklingen av optimering och styrning av nätverkssystem. Den här avhandlingen presenterar fyra viktiga bidrag inom detta område.

Det första bidraget handlar om kooperativ lastbilstransport. Först introduceras ett mått som mäter effektiviteten i systemet jämfört med en central planerare. Detta mått används sedan för att utvärdera vinsterna med kooperativa transporter, men används också för att utvärdera taxiförarnas vägval med verkliga data från New York City. Detta följs av en studie av de optimala vänteplatserna för lastbilar och de optimal placeringarna av distributionscentraler. Dessa positioner används sedan för att förbättra transportprestandan i ett kooperativt transportsystem.

I det andra bidraget studeras informationsaggregering i anonyma nätverkssystem, det vill säga nätverk där noderna saknar unika identiteter. Två specifika problem hanteras: att estimera storleken på nätverket, och att sammanställa fördelningen av lokala mätvärden i nätverket. Noderna i detta nätverk kräver anonymitet, men antas också ha strikt begränsad beräkningskapacitet. Vi presenterar stokastiska algoritmer med låga beräkningskrav, som dessutom har snabb konvergens och som kan justeras till att ge godtycklig precision. För dynamiska nätverk förbättras prestandan genom att en regulariseringsterm används för att väga observerad data mot förväntat beteende hos systemet.

I tredje bidraget analyseras ett peer-to-peer nätverk för direktsänd videodistribution. Speciellt studeras konvergensen av nätverkstopologin som genereras från lokala preferensfunktioner, och hur resultaten kan används för att minska fördröjningarna och avbrotten under videouppspelning. Vi ger nödvändiga och tillräckliga villkor för konvergens, samt karakteriserar gränsvärden för hur snabbt användare kan ansluta eller lämna nätverket utan att påverka prestandan.

Det sista bidraget är en distribuerad optimeringsalgoritm. Problemet består i att minimera summan av konvexa funktioner för varje nod i ett nätverk. En decentraliserad optimeringsalgoritm presenteras som baseras på det duala optimeringsproblemet tillsammans med subgradient-metoden. Konvergenshastigheten analyseras för olika val av steglängder, konstanta samt tidsberoende kommunikationsfördröjningar och brusiga kommunikationskanaler.

Acknowledgements

Life is an adventure, and during the past few years I have had the fortune to share this adventure with many great persons, to whom I am sincerely grateful.

First of all, I would like to express my deepest gratitude to my advisor Karl Henrik Johansson, for your continuous support and enthusiasm. Every time I have entered your office in frustration and despair, you have been cheerful and encouraging, and provided me with valuable guidance.

To my current co-advisor Alexandre Proutiere, thank you for our valuable discussions and your excitement. You have set an excellent example as a researcher and mentor. I would also like to thank my former co-advisor Ather Gattami, with whom I worked closely during the first year. Thank you for all the insight you provided, and our great discussions. This gratitude extends to the entire faculty at the automatic control department, for creating such an exciting research environment and a special mention to Henrik Sandberg, Mikael Johansson, Dimos Dimarogonas and Jonas Mårtensson.

I am very grateful to my co-authors. Damiano Varagnolo for inviting me to Padova, our successful collaborations, and your friendship. Guodong Shi, for our interesting collaborations and your technical advise. I would also like to thank the industrial collaborations with SICS, P1 and Scania, with whom I have had the pleasure to work.

This journey began when I visited California Institute of Technology, and I am immensely thankful to Professor Richard M. Murray and Professor Ufuk Topcu for introducing me to the world of research, and to my friend Dionysios Barmpoutis.

To all my friends and colleagues at the automatic control department, both current and former. It has been a great pleasure to work, laugh, suffer and enjoy the time together with you! A special mention to Aitor, Alireza, André, António, Arda, Bart, Farhad, Hamid, José*, Kuo-Yun, Lars, Martin, Meng, Niklas, Olle, Patricio*, Pedro*, Sadegh and Sebastian for all the moments and discussions we have shared together.

A special thanks to the administrators Hanna, Anneli, Karin, Gerd, Katharine and Kristina for always being helpful, and creating a great office atmosphere (not to mention the waffles!).

I am also very grateful to the Swedish Research Council, the Knut and Alice Wallenberg Foundation and the KTH School of Electrical Engineering, through the Program of Excellence, for their financial support of my studies.

Finally, I would like to thank my friends and family for your encouragement and patience through this adventure. To my long-lasting friendship with Martin, to my former classmates Ulf, Einar, Erik and Sebastian, to Tanja, Erik and John, and to all my triathlon friends who gave me the energy to complete this adventure and for bringing happiness to my life. In particular, I would like to express my gratitude to my parents and my brother for all their support throughout my life.

Thank you!

Håkan Terelius
Stockholm, September 2016.

Contents

Contents **ix**

Notations **xi**

Abbreviations **xiii**

1 Introduction **1**
1.1 Motivational Examples 2
1.2 Problem Formulations 7
1.3 Thesis Outline and Contributions 8

2 Background **13**
2.1 Control Theory 13
2.2 Probability Theory 15
2.3 Graph Theory 17
2.4 Optimization Theory 20
2.5 Consensus Algorithms 23
2.6 Transportation Networks 25
2.7 Anonymous Networks 27
2.8 Network Estimation 27
2.9 Peer-to-Peer Networks 29

3 Efficiency in Transportation Networks **31**
3.1 Transportation Efficiency 32
3.2 Freight Transportation Case Study 37
3.3 New York City Taxi Case Study 42
3.4 A One-dimensional Freight Transportation Model 48
3.5 Optimal Idling Location 50
3.6 Optimal Distribution Center Location 51
3.7 Time-Space Diagrams 59
3.8 Simulation Study 62
3.9 Conclusions 67

4 Estimation in Anonymous Networks 69
4.1 Size Estimation Problem . . . 70
4.2 Network Size Estimation Algorithm . . . 71
4.3 Properties under a Markovian P2P Model . . . 76
4.4 Evaluation of the Size Estimation Algorithm . . . 80
4.5 Estimation of Probability Mass Functions . . . 84
4.6 PMF Estimators Based on Consensus Protocols . . . 86
4.7 Statistical Characterization of the Max Consensus PMF Estimator 89
4.8 Transient Evaluation of PMF Estimators . . . 92
4.9 Estimating Time Dependent PMFs . . . 99
4.10 Conclusions . . . 101

5 Topology Convergence in Peer-to-Peer Networks 103
5.1 The Gradient Topology Problem . . . 104
5.2 Convergence Analysis . . . 106
5.3 Convergence Rate Estimation . . . 110
5.4 Convergence Rate with Network Churn . . . 112
5.5 Convergence Simulation . . . 115
5.6 Evaluating Live-Streaming using the Gradient Topology . . . 121
5.7 Conclusions . . . 126

6 Distributed Optimization via Dual Decomposition 127
6.1 Distributed Optimization Problem . . . 128
6.2 Dual Optimization Algorithm . . . 130
6.3 Convergence Analysis . . . 131
6.4 Communication Cost . . . 147
6.5 Numerical Results . . . 148
6.6 Conclusions . . . 157

7 Conclusions and Future Work 159
7.1 Conclusions . . . 159
7.2 Future Work . . . 160

Bibliography 165

Notations

$\doteq$	Definition
$\mathbb{N}$	Set of natural numbers, $0, 1, 2, \ldots$
$\mathbb{R}$	Set of real numbers
$\mathbb{C}$	Set of complex numbers
A^{T}	Transpose of matrix A
$\lambda(A)$	Eigenvalues of matrix A
$\mathbb{1}$	Column vector with all elements equal to one
$\mathrm{diag}(v)$	Diagonal matrix, with elements from vector v
$\mathcal{G}$	Graph consisting of nodes $\mathcal{V}$ and edges $\mathcal{E}$
$\mathcal{V}$	Set of nodes (vertices or agents) for graph $\mathcal{G}$
$\mathcal{E}$	Set of edges (links or relations) for graph $\mathcal{G}$
$\mathcal{N}_i$	Set of neighbors to node i
$\mathcal{U}[0,1]$	Uniform distribution over the interval $[0,1]$
$p(x)$	Probability density function
$\mathbb{P}[A]$	Probability of event A
$\mathbb{E}[X]$	Expectation of random variable X
$\mathrm{var}(X)$	Variance of random variable X
$\mathrm{cov}(X,Y)$	Covariance between random variables X and Y
$\widehat{x}$	Estimate of quantity x
$\Gamma(n)$	Gamma function of n
$\Gamma(k,\theta)$	Gamma distribution with shape k and scale θ
$\text{I-}\Gamma(k,\theta)$	Inverse gamma distribution with shape k and scale θ

Abbreviations

a.e.	almost everywhere
CDF	cumulative distribution function
EM	expectation-maximization
i.i.d.	independent and identically distributed
ITS	intelligent transportation systems
MC	Monte Carlo
ML	maximum likelihood
MSE	mean squared error
NYC	New York City
P2P	peer-to-peer
PDF	probability density function
PID	proportional integral derivative
PMF	probability mass function
RMSE	root-mean-square error
SICS	Swedish Institute of Computer Science

Chapter 1

Introduction

"Clear? Huh! Why a four-year-old child could understand this report! Run out and find me a four-year-old child, I can't make head or tail of it."
— Groucho Marx

Networks of interconnected systems appear everywhere in modern societies, driven by technological developments in computation and communication. Prominent applications are found in communication networks, such as the cellular network and the Internet, large-scale infrastructures, such as transportation systems and power grids, and even biological and social networks. A common feature of all these application examples is that they are composed of several smaller subsystems, and that their complexity arises from the interconnections into large networks. The size and scale of these networked systems are continuing to accelerate, and this growing imposes a fundamental challenge to our understanding of the world around us.

Feedback control has been an essential part of the industrial revolution, and continues to play a crucial role for automation, contributing to improved efficiency and productivity. The field has advanced from the centrifugal governor for controlling the rotational speed in the early steam engines, through PID controllers for cruise control in our cars, to today, where developments within artificial intelligence is used towards creating completely self-driving cars. Following this trend, it is only natural to incorporate feedback control into network systems, both for controlling the performance of the networked system, but also for utilizing sensing, computation and actuation distributed over the network.

In this thesis, we aim to improve the current understanding of network systems, and we focus on four specific areas. In Chapter 3 we study efficient collaborative transportation, in Chapter 4 we study distributed computation and estimation schemes in anonymous networks, while in Chapter 5 we study efficient network topologies for video distribution, and in Chapter 6 we develop a distributed optimization method. The reminder of this chapter is organized as follows. In Section 1.1 we introduce several motivating examples for this work. In Section 1.2, we

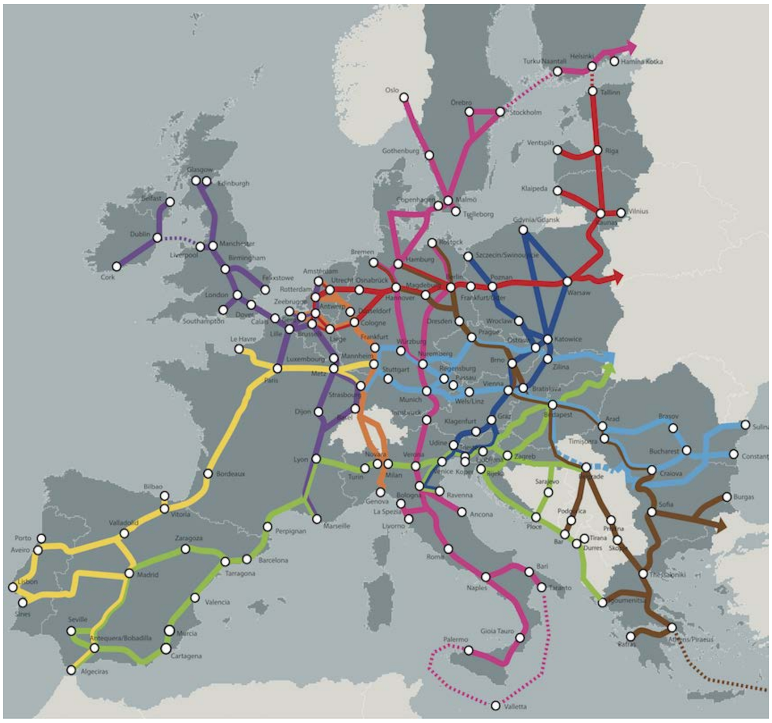


Figure 1.1: The Trans-European Transport Network (Courtesy of European Commission).

define the research problems for this thesis, and in Section 1.3 we list the contributions and the thesis outline.

1.1 Motivational Examples

In the following, three examples of important network systems are given.

Transportation Networks

The transportation network (Figure 1.1) is a prominent example of a network system that has an immense effect on the global economy and environment. This can be illustrated by the fact that the financial crisis in 2008 caused a significant decrease in transportation activity during 2009 (−11 % compared to 2007). In the European Union, road transportation accounts for roughly a quarter of the total energy consumption and a sixth of all greenhouse gas emission [European Union, 2014]. Moreover, transportation safety is a critical problem, as there were 54 439 road fatalities reported in the European Union during 2012 [European Union, 2015].

Figure 1.2: A traffic jam in Delhi (Courtesy of Wikimedia).

Road transportation is a particular problem in urban areas, where the transportation demand exceeds the capacity of the road, and traffic congestion sets in (Figure 1.2). Researchers cannot fully predict when traffic jams will occur, because minor incidents can have ripple effects in the entire traffic system. Traffic congestion is a major source of air pollution around our cities [Barth and Boriboonsomsin, 2008], and is also negatively affecting the productivity and global economy due to wasted time. Furthermore, the increased commuting times, and in particular the uncertainties in traveling time is dampening the job market growth [Hymel, 2009].

Simply building new roads might not be the best solution for congestion problems due to the enormous required investments and the complex network effects. Braess's paradox [Braess et al., 2005], originally published in 1968, describes a scenario where adding extra capacity to a network can reduce the overall performance, caused by the Nash equilibrium not equating the global optimal flow. Instead, there is a huge potential in improving the situation through better traffic management and personalized recommendations to the drivers.

Optimizing and controlling the road transportation system is an inherently difficult task due to the distributed nature of the system, where every driver has their own objective. It is also difficult due to the hybrid nature of the traffic system, considering both continuous traffic flows and discrete vehicles. A centralized signaling mechanism was implemented in Stockholm in 2006 as congestion taxes, and resulted in a significant reduction in congestion [Börjesson et al., 2012]. After an initial seven months trial and preceding criticism, the system became permanent in 2007 due to its success, with a 20 % reduction of traffic volumes and a 30 % decrease in traveling times.

The optimization challenges do not evade the public transportation system, which is often evolving sequentially, and poorly adapting to the new demands of the travelers. The results are unnecessarily long traveling times and multiple transfers, decreasing the reliability of the system [Mandl, 1980]. Taxi drivers face similar optimization challenges, when deciding where to drive to pick up the next passenger. As there is limited information regarding the current and rapidly changing customer demand, data mining previous trips can be useful to build probabilistic models for future demand [Moreira-Matias et al., 2012].

Road freight transportation in the European Union amounts to 3.5 trillion metric ton-km per year, and employs over 3 million people [European Union, 2014]. At the same time, 20 % of the trucks are estimated to travel empty [Bureau of Transportation Statistics, 2015], which means that there is a great potential for utilizing this back-hauling capacity for improved efficiency. Reducing the empty mileage through collaboration is studied in Chapter 3. Another approach for reducing the fuel consumption in road transportation through collaboration is by dynamically building platoons of trucks driving autonomously with a small gap between the vehicles. This enables fuel savings of up to 10 % [Alam et al., 2010, 2015], and at the same time is expected to reduce the traffic fatalities [Davila and Nombela, 2012].

The transition towards intelligent transportation systems is to a large extent driven by the development of connected sensors enabling vehicle-to-vehicle and vehicle-to-infrastructure communication, but this also raises concerns about the privacy for the users, as well as protecting trade secrets for the transportation providers. Finding the right balance between anonymity and transportation efficiency is a challenging problem [Glancy, 1995]. Methods for monitoring a network of anonymous agents is considered in Chapter 4.

Internet and Peer-to-Peer Networks

The Internet has penetrated our daily lives as the single most important information exchange system, and is used for sending messages, reading news and watching television. The annual global IP traffic will reach 1 zettabytes (10^{21} bytes) during 2016, and is expected to double until 2019. A majority of the Internet traffic consists of video delivery, constituting 64 % of all consumer Internet traffic in 2014 and expecting to grow to 80 % by 2019 [Cisco, 2015]. Put in perspective, by 2019 a million minutes of video content will cross the Internet every second. This huge demand for network bandwidth is creating a lot of pressure towards efficient content distribution strategies.

A peer-to-peer (P2P) distribution model (Figure 1.3) is based on the idea that the users of the system contribute a share of their bandwidth to redistribute the

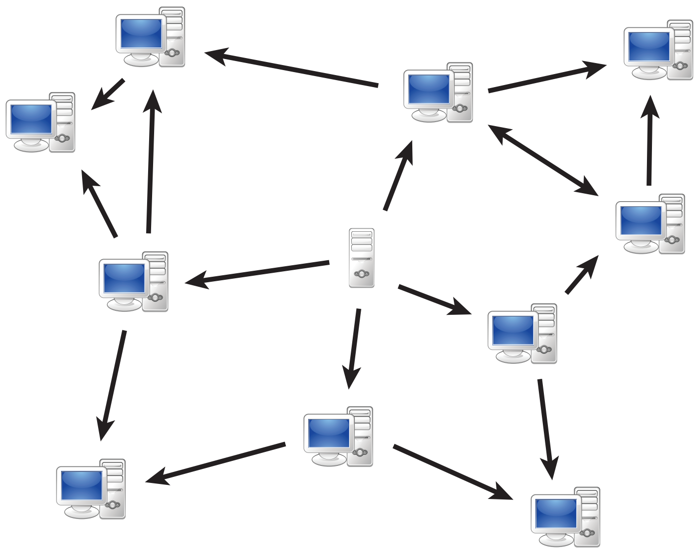


Figure 1.3: An illustration of a peer-to-peer file sharing network, where users are relaying information from a central server to each other.

content, thereby reducing the pressure on the central content provider. An advantage with this model is that the available capacity of the network scales with the number of users, and is especially useful when the user demand is unpredictable. Thus, the P2P networks are by design resilient to network failure and user churn, and the distributed nature makes them suitable in many situations where the users are concerned about their privacy.

Several commercial applications use P2P systems for video distribution [Thampi, 2013], but they are also commonplace in other applications. For example the Swedish music streaming service, Spotify, used P2P streaming [Kreitz and Niemelä, 2010], as does the video chat application Skype [Baset and Schulzrinne, 2006]. Even traditional client–server applications can use P2P technology internally when using distributed content delivery networks to increase the capacity by utilizing a large number of servers. Another application where the resilience and anonymity of P2P networks has played a central role is for the distributed crypto-currency BitCoin [Nakamoto, 2008].

However, the task of designing self-organizing P2P networks is still a challenging problem, demanding both resilience to high churn and an efficient content distribution. The construction of an efficient gradient P2P topology is considered in Chapter 5, and privacy preserving estimations of network properties is considered in Chapter 4.

Figure 1.4: A Tmote Sky low power wireless sensor module, used for wireless sensor network applications (Courtesy of Moteiv).

Wireless Sensor Networks

In the last several decades we have seen great advances in computation and communication hardware, leading to the availability of tiny wireless devices (Figure 1.4) capable of performing sensing, computation and control. These systems have found applications in a wide variety of domains, including industrial control systems and in building automation [Araújo, 2014].

In industrial process automation, such as the Iggesund's paper mill (Figure 1.5), the control systems typically consist of hundreds of control loops, and these utilize thousands of sensors and actuators for continuous monitoring and controlling of the plant. Traditionally, these sensors and actuators have been connected through wired communication channels, resulting in high setup and maintenance cost for physical wires, with costs ranging between 300–6000 USD per meter [Samad et al., 2007]. Low-powered wireless technology could provide a cost-effective alternative to installing wires for these control applications, as the cabling cost begins to stand-out compared to the sensor and actuator cost, and the wireless technology allows for many more sensors and thereby collecting much more information from the processes.

Two challenges in particular are critical to overcome for the adoption of wireless sensor networks in the process industry: the power management and the communication reliability. Thousands of wireless devices, with a battery lifetime in the order of a year, would require full-time staff for battery replacement. This imposes strict limitations on the computational and communicational capabilities of the wireless devices, and motivates the research on simple but reliable distributed algorithms. Algorithms for aggregation of measurements are considered in Chapter 4 and distributed optimization algorithms are considered in Chapter 6, for networks with severely limited resource constraints.

Figure 1.5: Iggesund's paper mill, where wireless control systems have been implemented as part of the WiComPI project [Araújo, 2014].

1.2 Problem Formulations

The prevalent appearance of challenges in networked systems has lead us to limit the scope of this thesis to the following research questions.

In Chapter 3, we investigate the transportation network, and the implications of a specific intelligent transportation solution for collaborative transport. We specifically ask the research questions:

Q1: How can the transportation system efficiency be measured for a collaborative transportation scenario?

Q2: Where should trucks and distribution centers be positioned to optimize the transportation efficiency?

Question **Q1** is essential for evaluating collaboration strategies in transportation systems, and question **Q2** gives some insight into optimal collaboration policies.

In Chapter 4, we consider an anonymous computation framework, where the nodes do not have unique identifiers, motivated by both the privacy concerns in transportation and P2P networks as well as the limited computational capacities in wireless sensor networks. We ask the research questions:

Q3: In an anonymous network, how can the number of nodes be estimated?

Q4: In an anonymous network, how can measurement distributions be aggregated?

Questions **Q3** and **Q4** are essential for network maintenance, where sudden changes could require restorative control action.

In Chapter 5 we turn to a scalable live video distribution system using P2P networks, and we ask the questions:

Q5: How should P2P video distribution network topologies be organized for efficient delivery?

Q6: How is the topology structure affected by network churn, i.e., nodes joining and leaving the network?

The questions **Q5** and **Q6** facilitate the use of P2P networks for building scalable distribution networks at low cost.

Finally, in Chapter 6 we consider the distributed optimization problem of minimizing a sum of local objective functions.

Q7: Can existing distributed optimization methods be extended to the dual problem formulation?

Q8: Is the distributed dual optimization method robust to time-varying delays and communication noise?

Answering questions **Q7** and **Q8** can establish key technologies for many control problems in distributed network systems.

1.3 Thesis Outline and Contributions

This thesis is a compilation of results presented in peer-reviewed scientific venues. The remainder of this thesis is organized as follows.

Chapter 2: Background

In this chapter, we provide mathematical preliminaries and a review of the existing research literature for the topics covered by this thesis.

Chapter 3: Efficiency in Transportation Networks

In this chapter, we investigate collaboration strategies for transportation networks, motivated by the development within intelligent transportation. First, we study an efficiency measure for the collaborative transportation scenarios, where we are able to determine how efficient the road utilization is in comparison to a centralized planner. The efficiency measure is also evaluated on real data from the NYC Taxi trips during 2013. In the second part, we study the optimal idling locations for trucks, and the optimal locations for distribution centers along a single highway going through multiple cities. These locations are then exploited in a simulation of a collaborative freight transport system.

This chapter is based on the publications:

- Håkan Terelius and Karl Henrik Johansson. An efficiency measure for road transportation networks with application to two case studies. In *54th IEEE Conference on Decision and Control*, pages 5149–5155, December 2015
- Kuo-Yun Liang, Sebastian van de Hoef, Håkan Terelius, Valerio Turri, Bart Besselink, Jonas Mårtensson, and Karl Henrik Johansson. Networked control challenges in collaborative road freight transport. *European Journal of Control*, 30:2–14, May 2016
- Håkan Terelius and Karl Henrik Johansson. On the optimal location of distribution centers for a one-dimensional transportation system. In *55th IEEE Conference on Decision and Control*, Las Vegas, USA, December 2016c. (to appear)
- Håkan Terelius and Karl Henrik Johansson. Efficiency modeling and optimization for road transportation. *Transportation Research Part B: Methodological*, 2016b. (submitted)

Chapter 4: Estimation in Anonymous Networks

In this chapter, we consider the problem of estimating the state of a sensor network, motivated by network maintenance. In particular, we study two problems, first estimating the size of the network, and secondly estimating the entire empirical distribution of sensor measurements over the network. The proposed algorithms are based on max consensus information exchange protocols, since they lead to fast convergence speeds as well as small communication burdens. What makes our scheme special is that we assume the agents to be anonymous, thus severely limiting the communication information.

This chapter is based on the publications:

- Håkan Terelius, Damiano Varagnolo, and Karl Henrik Johansson. Distributed size estimation of dynamic anonymous networks. In *51st IEEE Conference on Decision and Control*, pages 5221–5227, Maui, HI, USA, December 2012
- Håkan Terelius, Damiano Varagnolo, Carlos Baquero, and Karl Henrik Johansson. Fast distributed estimation of empirical mass functions over anonymous networks. In *52nd IEEE Conference on Decision and Control*, pages 6771–6777, Florence, Italy, December 2013b

Chapter 5: Topology Convergence in Peer-to-Peer Networks

In this chapter, we investigate the topology convergence in a P2P network system. The goal of the system is to maximize live-streaming performance through establishing a gradient overlay topology. The gradient overlay network is characterized by a directed graph, where each node has one set of neighbors with the same utility value and one set of neighbors containing higher utility values, such that paths of increasing utilities emerge in the network topology. The gradient overlay network is built using gossiping and a preference function that samples nodes from a uniform random peer sampling service. Evaluation of the gradient overlay topology in the live-streaming application GLive was performed by SICS.

This chapter is based on the publications:

- Håkan Terelius, Guodong Shi, Jim Dowling, Amir H. Payberah, Ather Gattami, and Karl Henrik Johansson. Converging an overlay network to a gradient topology. In *50th IEEE Conference on Decision and Control and European Control Conference*, pages 7230–7235, Orlando, FL, USA, December 2011a

- Håkan Terelius and Karl Henrik Johansson. Peer-to-peer gradient topologies in networks with churn. *IEEE Transactions on Control of Network Systems*, 2016a. (submitted)

Chapter 6: Distributed Optimization via Dual Decomposition

In this chapter, we study a distributed multi-agent optimization problem of minimizing the sum of convex objective functions. A decentralized optimization algorithm is introduced, based on dual decomposition together with the subgradient method, for finding the optimal solution. The convergence rate is analyzed both for different step size rules, constant and time-varying communication delays, and noisy communication channels.

This chapter is based on the publication:

- Håkan Terelius, Ufuk Topcu, and Richard M. Murray. Decentralized multi-agent optimization via dual decomposition. In *18th IFAC World Congress*, pages 11245–11251, Milan, Italy, August 2011b

Chapter 7: Conclusions and Future Work

In the final chapter, a summary of this thesis is provided and some possible future research directions are discussed.

Other Contributions

The following publication is not covered in this thesis, but is relevant for networked systems.

- Håkan Terelius, Guodong Shi, and Karl Henrik Johansson. Consensus control for multi-agent systems with a faulty node. In *4th IFAC Workshop on Distributed Estimation and Control in Networked Systems*, pages 425–432, Koblenz, Germany, September 2013a

Author's Contributions

The order of the paper authors reflects their contributions. The thesis author has formulated and solved the problems, as well as written the papers. The co-authors have participated in discussions and conventional supervision.

For the paper Liang et al. [2016], the thesis author developed one section and contributed to the rest of the paper. For the paper Terelius et al. [2011a], the evaluation of the P2P live-streaming application using a gradient topology (Section 5.6) was provided by Dowling and Payberah at SICS.

Chapter 2

Background

"The future is like a corridor into which we can see only by the light coming from behind."
— Edward Weyer Jr.

In this chapter, we provide mathematical preliminaries and a review of the existing research literature related to optimization and control in dynamical network systems.

2.1 Control Theory

A feedback system is typically characterized by the feedback loop (Figure 2.1), where the output $y(t)$ of a system G is routed back to a regulator F, which feeds the control signal $u(t)$ into the system G.

Feedback control is omnipresent in natural and biological systems, but the earliest mathematical analysis was laid out by Maxwell [1868], who studied the centrifugal governor for controlling the rotational speed in the early steam engines, patented by James Watt in 1788 (Figure 2.2). In 1868, there were more than 75 000 governors installed in England and, at this time, proportional, integral and derivative (PID) actions were understood and implemented by mechanical or hydraulic devices. Other early work within the control field includes the stability analysis of Nyquist [1932] and Bode [1940].

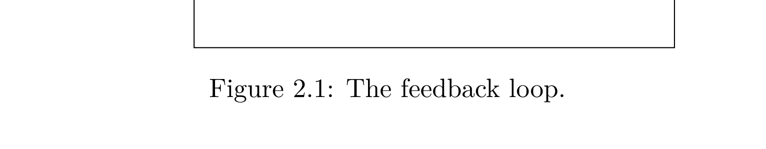


Figure 2.1: The feedback loop.

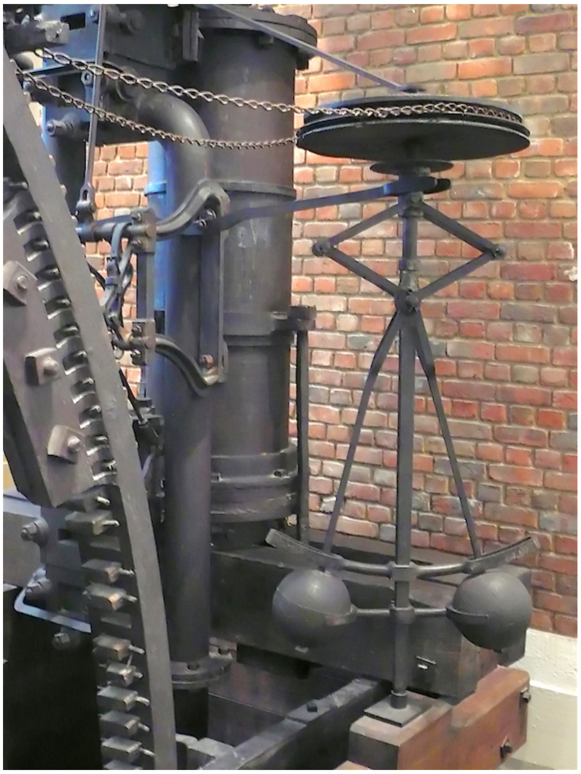


Figure 2.2: A centrifugal governor on a steam engine.
(Courtesy of M. Junge, Wikipedia).

Digital computers was quickly adopted in the control community, and intensified the development of optimal control (Bellman [1952] and Pontryagin et al. [1962]). Computer control also led to the development of model predictive control (MPC), where the current control action is obtained by solving a finite horizon optimal control problem [Mayne et al., 2000].

Feedback control played an essential role in the industrial revolution, and continues to play a crucial role today for automation, contributing to improved efficiency and productivity. A beautiful summary of the control theory development, including an extensive reference list, is provided by Åström and Kumar [2014], but the theory of feedback control is far from complete, as noted by Blondel and Megretski [2004].

2.2 Probability Theory

Probability theory is the branch of mathematics concerned about random events, and constitutes the foundation of statistics [Grinstead and Snell, 1997]. We will now review some of the basic concepts used throughout this thesis.

Definitions

A random variable X can be interpreted as the outcome of an uncertain event, for example a toss of a coin. The sample space Ω is the set of all possible outcomes. When Ω is countable, then X is said to be discrete, and the probability mass function (PMF) $m : \Omega \to [0, 1]$ for X is a function satisfying

$$\sum_{x \in \Omega} m(x) = 1.$$

The probability of an event $E \subseteq \Omega$ is defined as

$$\mathbb{P}[E] \doteq \sum_{x \in E} m(x).$$

The frequentist interpretation of the probability for an event E is that if n_e is the number of occurrences of an event E in n trials, then

$$\lim_{n \to \infty} \frac{n_e}{n} = \mathbb{P}[E].$$

Similarly, if the sample space Ω of X is continuous real-valued, then the cumulative distribution function (CDF) $F_X : \Omega \to [0, 1]$ for X is a monotonically non-decreasing continuous function satisfying

$$\lim_{x \to -\infty} F_X(x) = 0$$
$$\lim_{x \to \infty} F_X(x) = 1.$$

The corresponding probability density function (PDF) $p : \Omega \to [0, 1]$ is defined as

$$p(x) \doteq \frac{\mathrm{d}F_X(x)}{\mathrm{d}x}.$$

The probability of an event $E \subseteq \Omega$ is defined as

$$\mathbb{P}[E] \doteq \int_{x \in E} p(x)\ \mathrm{d}x.$$

An event E is said to happen almost surely if $\mathbb{P}[E] = 1$.

An important concept in probability theory is the expected value, that is the probability-weighted average of all possible values. For a discrete random variable X, the expected value $\mathbb{E}\left[X\right]$ is defined as

$$\mathbb{E}\left[X\right] \doteq \sum_{x \in \Omega} x\, m\left(x\right),$$

and for a continuous random variable, it is defined as

$$\mathbb{E}\left[X\right] \doteq \int_{x \in \Omega} x\, p\left(x\right)\ \mathrm{d}x.$$

The expected value is a characterization of the location of a probability distribution, and the variance characterizes the dispersion of the probability distribution around the expected value. Let $\mu \doteq \mathbb{E}\left[X\right]$ denote the expected value, then the variance of X is defined as

$$\operatorname{var}\left(X\right) \doteq \mathbb{E}\left[(X - \mu)^2\right].$$

Given two random variables X and Y, the covariance measures the linear dependency between X and Y, defined as

$$\operatorname{cov}\left(X, Y\right) \doteq \mathbb{E}\left[(X - \mathbb{E}\left[X\right])(Y - \mathbb{E}\left[Y\right])\right].$$

Markov Chains

A stochastic process is a time sequence for the evolution of a system which depends on random events. In the case of discrete time, the stochastic process can be thought of as a sequence of random variables. A Markov chain is a memoryless stochastic process [Levin et al., 2009]. Thus, a sequence of random variables X_1, X_2, …, is a Markov chain if

$$\mathbb{P}\left[X_t = x_t \mid X_{t-1} = x_{t-1},\ \ldots, X_1 = x_1\right] = \mathbb{P}\left[X_t = x_t \mid X_{t-1} = x_{t-1}\right]$$

holds for all $t > 1$, and this is referred to as the Markov property. Thus, the probability of the current state only depends on the previous state, and not on the entire history. Especially for finite discrete Markov chains, the conditional probabilities are referred to as transition probabilities, and defines a transition matrix P by its elements p_{ij} as

$$p_{ij} \doteq \mathbb{P}\left[X_t = j \mid X_{t-1} = i\right].$$

Let $\boldsymbol{\pi}^{(t)}$ denote the row vector of state probabilities at time t, thus defined by its elements

$$\boldsymbol{\pi}_i^{(t)} \doteq \mathbb{P}\left[X_t = i\right],$$

then the state evolution can be written compactly as

$$\boldsymbol{\pi}^{(t+1)} = \boldsymbol{\pi}^{(t)} P = \boldsymbol{\pi}^{(1)} P^t,$$

thus the analysis of the Markov chain considers the matrices P^t and their properties.

Monte Carlo Methods

Monte Carlo methods belong to a class of computational methods that rely on repeated sampling from a stochastic process to solve a mathematical problem [Metropolis and Ulam, 1949]. A typical Monte Carlo method breaks down a complex mathematical problem into simple deterministic computations by considering a hypothetical initial condition. Then, a large set of initial conditions is generated from a stochastic process, and the results from the simpler calculations are aggregated to an approximate solution for the original problem.

For example, the optimization problem of minimizing $f(x)$ where $x \in [0, 1]$ can be solved by generating a sequence of random values $x_1,\ x_2,\ \ldots \in [0, 1]$, and then evaluating $f(x_1),\ f(x_2),\ \ldots$. The optimal value computed by the Monte Carlo method is then the minimal value of $f(x_1),\ f(x_2),\ \ldots$.

Monte Carlo methods have been used in many applications, from mathematical optimization and integration to particle physics and statistical mechanics, as well as for biological cell populations and operations research [Halton, 1970].

2.3 Graph Theory

This thesis focuses on the topic of *networks*, which are mathematically modeled as *graphs*, a concept of pairwise relations between objects. The foundation was laid by Euler [1741] with the famous work on the seven bridges of Königsberg. The scientific field studying complex network systems, appearing in for example social, biological, and telecommunication systems, is often called network science [Barabási, 2016].

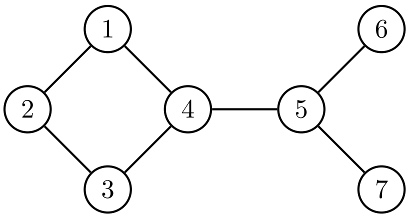
In this section, we introduce some notations and basic concepts from graph theory that will be useful for describing network systems. A more complete description of this topic can be found in the book by Diestel [2005].

A graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$ consists of a set of objects, called nodes, vertices, or agents, denoted by $\mathcal{V}$, and a set of pairwise relations, called edges or links, denoted by $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$. We usually denote the number of nodes in a graph $\mathcal{G}$ by $N = |\mathcal{V}|$.

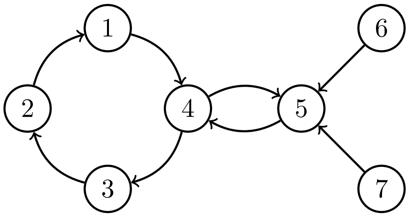
Undirected graphs are the graphs where the edges are unordered pairs of vertices, hence (i, j) and (j, i) are considered to be the same edge, and $(i, j) \in \mathcal{E}$ if and only if $(j, i) \in \mathcal{E}$. All other graphs are directed graphs, where the edge (i, j) is directed from node i to node j.

The usual way of picturing an undirected graph is by drawing a circle for each node, and joining two of the circles by a line if the corresponding nodes form an edge (Figure 2.3a). For a directed graph, circles are joined by an arrow pointing from i to j if (i, j) forms an edge of the graph (Figure 2.3b).

The neighbors of a node i are those nodes that have a common edge with i, and are denoted by $\mathcal{N}_i = \{j \mid (i, j) \in \mathcal{E}\}$. For a directed graph, the neighbors can be divided into the in-neighbors and out-neighbors of a node, depending on the direction of the edge. The degree of a node is the size of its neighborhood $d_i = \deg(i) = |\mathcal{N}_i|$.



(a) An undirected graph with 7 nodes and 7 edges.



(b) A directed graph with 7 nodes and 8 edges.

Figure 2.3: Graphs are illustrated with circles representing nodes, and lines representing edges.

A path in $\mathcal{G}$ is a list of distinct vertices $v_0, v_1, \ldots, v_n$ such that $(v_i, v_{i+1}) \in \mathcal{E}$, $i = 0, 1, \ldots, n-1$. The number of edges in the path is the length of the path. A directed graph is strongly connected if there exists a path from every vertex to every other vertex in the graph, while for an undirected graph, it is simply called connected. The distance $\operatorname{dist}(v_0, v_n)$ between two nodes v_0 and v_n, is the length of the shortest path between them, or ∞ if there is no such path. Further, the diameter of a graph is the greatest distance between any two nodes in the graph.

Spectral Graph Theory

Spectral graph theory studies graphs by representing them with matrices and analyzing the corresponding matrix properties, such as eigenvalues [Godsil and Royle, 2001]. The adjacency matrix A of a graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$ is an $N \times N$ matrix, whose entries a_{ij} are given by the edges as:

$$a_{ij} = \begin{cases} 1 & \text{if } i \neq j \text{ and } (i,j) \in \mathcal{E}, \\ 0 & \text{otherwise.} \end{cases}$$

The next important matrix is the Laplacian matrix L of a graph, defined as $L = D - A$, where D is the diagonal degree matrix, and A is the adjacency matrix. The Laplacian matrix has the elements:

$$l_{ij} = \begin{cases} \deg(i) & \text{if } i = j, \\ -1 & \text{if } i \neq j \text{ and } (i,j) \in \mathcal{E}, \\ 0 & \text{otherwise.} \end{cases}$$

The eigenvalues of the Laplacian matrix are important for many network applications, including robustness and convergence time for dynamic networks, as described by Fax and Murray [2002] and Rahmani et al. [2009]. Note for example that the multiplicity of the zero eigenvalue of the Laplacian is equal to the number of connected components in the graph.

For example, the adjacency matrix and Laplacian matrix of the undirected graph in Figure 2.3a are

$$A = \begin{bmatrix} 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix}, \quad L = \begin{bmatrix} 2 & -1 & 0 & -1 & 0 & 0 & 0 \\ -1 & 2 & -1 & 0 & 0 & 0 & 0 \\ 0 & -1 & 2 & -1 & 0 & 0 & 0 \\ -1 & 0 & -1 & 3 & -1 & 0 & 0 \\ 0 & 0 & 0 & -1 & 3 & -1 & -1 \\ 0 & 0 & 0 & 0 & -1 & 1 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 & 1 \end{bmatrix}.$$

These definitions can be directly extended to weighted graphs, where each edge $(i, j) \in \mathcal{E}$ has an associated weight $w_{ij} \in \mathbb{R}$, and the corresponding matrix is the weighted adjacency matrix.

Random Graph Theory

Random graph theory was coined by Gilbert [1959] and Erdős and Rényi [1959], and describe graphs generated by random processes. Different random graph models exist, characterized by their probability distribution. The Gilbert model $G(n, p)$ generates graphs with n nodes, where every possible edge occurs with probability p. The Erdős–Rényi model $G(n, M)$, on the other hand, considers graphs with n nodes, where every graph containing M edges occur with equal probability.

A third model is the Barabási–Albert model [Albert and Barabási, 2002], which builds graphs by sequentially adding nodes, connecting them using preferential attachment. The process starts with a graph consisting of m_0 nodes, and then each new node is connected to $m \leq m_0$ of the existing nodes with a probability proportional to the degrees of the existing nodes. This class of random graphs has a degree distribution that follows a power law $\mathbb{P}[\deg(i) = k] \sim k^{-3}$, and is commonly observed in natural networks.

Random graphs are used to generate typical network behaviors, and is a fundamental tool in the network sciences.

2.4 Optimization Theory

In mathematics, an *optimization problem* is the problem of finding an optimal solution among all feasible solutions. The standard form for an optimization problem can be written as [Boyd and Vandenberghe, 2009]:

$$\begin{aligned} \underset{x\in\mathbb{R}^n}{\text{minimize}} \quad & f(x), \\ \text{subject to} \quad & c_i(x) \leq b_i, \quad i = 1, \dots, m. \end{aligned} \tag{2.1}$$

Thus, the standard problem is to choose the optimization variable $x = (x_1, \dots, x_n) \in \mathbb{R}^n$ such that the scalar-valued objective function $f : \mathbb{R}^n \to \mathbb{R}$ attains its minimal value over the feasible domain. The feasible domain is determined by the constraint functions $c_i(x) : \mathbb{R}^n \to \mathbb{R}$, and can be defined as

$$\mathcal{D} = \{x \mid c_i(x) \leq b_i \quad i = 1, \dots, m\}.$$

The points $x \in \mathcal{D}$ are said to be feasible points, and the optimal value f^* to the problem (2.1) is the maximum lower bound to f in the feasible domain,

$$f^* = \inf \{f(x) \mid x \in \mathcal{D}\}.$$

A vector $x' \in \mathcal{D}$ is a local optimal solution if it minimizes f for a neighborhood around x', i.e., that there exits an $r > 0$ such that x' solves the optimization problem:

$$\begin{aligned} \underset{x\in\mathbb{R}^n}{\text{minimize}} \quad & f(x), \\ \text{subject to} \quad & c_i(x) \leq b_i, \quad i = 1, \dots, m, \\ & ||x - x'||_2 \leq r. \end{aligned}$$

Convex Optimization

A very important class of optimization problems are those where both the objective and the constraint functions are convex.

Definition 2.1. A function $f : \mathbb{R}^n \to \mathbb{R}$ is said to be *convex* if it satisfies the inequality

$$f(\alpha x + (1 - \alpha)y) \leq \alpha f(x) + (1 - \alpha)f(y), \tag{2.2}$$

for all $x, y \in \mathbb{R}^n$ and all $\alpha \in \mathbb{R}$, $0 \leq \alpha \leq 1$.

It is worth to notice that in particular all linear functions are convex, since they satisfy the condition with equality. A quadratic function $f(x) = x^T Qx + c^T x + b$ is convex if and only if $Q \succeq 0$ is positive semidefinite.

Definition 2.2. A set $C \subseteq \mathbb{R}^n$ is said to be *convex* if the line segment between any two points in C also lies in C. Thus, for any points $x_1, x_2 \in C$, and for $0 \leq \alpha \leq 1$,

$$\alpha x_1 + (1 - \alpha)x_2 \in C.$$

Notice that if the constraint function $c_i(x)$ is a convex function, then the set of points satisfying the condition $\{x \,|\, c_i(x) \leq b_i\}$ is a convex set. Further, the intersection of convex sets is also convex, hence, if all constraint functions c_i are convex, then so is the feasible domain $\mathcal{D}$.

A fundamental property of convex optimization problems is that any local optimal solution is also a global optimal solution. The implication of this is that, for convex optimization problems, it is enough to find a local optimal solution, which, in general, is a much easier problem than finding a global optimal solution. Many efficient algorithms exist with the purpose of finding a local optimal solution [Boyd and Vandenberghe, 2009].

Prominent work in convex optimization include the development of the simplex method [Dantzig et al., 1955], the interior point methods [Nesterov and Nemirovskii, 1994], and the semidefinite programming [Vandenberghe and Boyd, 1996].

Subgradient Methods

The concept of subgradients is a generalization of the gradient to non-differentiable convex continuous functions [Boyd and Vandenberghe, 2009, Hiriart-Urruty and Lemaréchal, 2001], developed for optimization by Shor [1983]. The generalization to subgradient methods especially allows for piecewise differential functions to be considered.

Definition 2.3. A vector $g \in \mathbb{R}^n$ is a *subgradient* to $f : X \subseteq \mathbb{R}^n \to \mathbb{R}$ at the point $x \in X$ if, for all other points $z \in X$, the following holds

$$f(z) \geq f(x) + g^T(z - x).$$

Further, the *subdifferential* of the function f, at $x \in X$, is the set of subgradients to f at x, and it is denoted by $\partial f(x)$. Thus,

$$\partial f(x) = \left\{ g : f(z) \geq f(x) + g^T(z - x) \quad \forall z \in X \right\}.$$

Remark. If f is a convex function, and differentiable at x, then there is exactly one subgradient of f at x, and it coincides with the gradient.

The *subgradient method* is a simple first-order algorithm for minimizing a non-differentiable convex function. It is based on the well-known gradient descent method, originally proposed by Cauchy in 1847 [Boyd and Vandenberghe, 2009, Snyman, 2005], but extended to non-differentiable functions by replacing the gradient with a subgradient to the function.

Consider the unconstrained optimization problem

$$\text{minimize}\, f(x),$$

where $f : \mathbb{R}^n \to \mathbb{R}$ is a convex function. The subgradient method solves this optimization problem by the iterative algorithm

$$x(t+1) = x(t) - \alpha_t g(t).$$

Here, $x(t)$ denotes the estimate of the solution at step t, $g(t) \in \partial f(x(t))$ is a subgradient to f at $x(t)$, and α_t is the step size.

Let the norm of the subgradients be uniformly bounded by $G > 0$, $||g(t)||_2 \leq G$ $\forall t$, and the distance from the initial point to the optimal set be bounded by $R > 0$. Let $f_{\min}$ be the minimal solution found until step t, then the convergence results for the subgradient method can be stated as

$$f_{\min}(t) \leq f^* + \frac{R^2 + G^2 \sum_{i=0}^{t} \alpha_i^2}{2 \sum_{i=0}^{t} \alpha_i}. \tag{2.3}$$

For a constant step size $\alpha_t = \alpha$, the convergence results becomes

$$f_{\min}(t-1) \leq f^* + \frac{R^2 + G^2 \alpha^2 t}{2\alpha t}. \tag{2.4}$$

The subgradient method is a first-order method, and can have a slow convergence rate compared to more advanced methods, such as the interior-point methods. However, it is still attractive in many situations, due to its ability to handle non-smooth optimization problems using a very low complexity in each iteration, and without prior knowledge of the problem structure, compared to the work by Nesterov [2005].

Distributed Optimization

The ability to compute an optimal distributed decision has gained a lot of attention during the last decades, e.g., Tsitsiklis [1984], Lynch [1996], Johansson [2008], and is also the topic of this thesis. Distributed optimization problems appear in a broad range of practical applications, such as when a set of network users are competing for a shared resource.

Decomposition methods for large-scale optimization have been the focus of a lot of research for some time, see for instance the work by Cohen [1980]. This area has seen renewed interest through the development of new networked applications, such as multi-agent systems, sensor networks and distributed computing.

Nedić and Bertsekas [2001] introduced the idea of applying incremental subgradient updates, and Rabbat and Nowak [2004] used this idea to propose a distributed subgradient optimization method, where the nodes sequentially update the optimization variable, while Nedić and Ozdaglar [2007] extended this idea using the average consensus algorithm to perform parallel updates. These methods continue to get a lot of research attention: Nedić et al. [2010], Srivastava and Nedić [2011] considered constrained optimization variations, while Ram et al. [2010] considered stochastic subgradient errors. Tsianos and Rabbat [2012] extended the method to an online algorithm, and Hale and Egerstedt [2015] constructed a resembling algorithm which preserves the agents' privacy. Relaxed communication assumptions, which do not require bidirectional communication, were considered by Nedić and Olshevsky [2015] and Mai and Abed [2016]. This line of research also includes our work on the distributed dual optimization problem, discussed in Chapter 6.

Different approaches to distributed optimization include the dual averaging technique by Duchi et al. [2012] based on the primal-dual subgradient by Nesterov [2009] (not to be confused with the dual decomposition techniques). Zargham et al. [2011] considers accelerated distributed dual subgradient methods specifically for network flow problems.

Another approach to distributed optimization is based on the alternating direction method of multipliers (ADMM), using sequential partial updates to the dual variables of an augmented Lagrangian. It has found use in large-scale statistical learning [Boyd et al., 2011, Gabay and Mercier, 1976]. An asynchronous extension, albeit centralized, was proposed by Zhang and Kwok [2014], while Wei and Ozdaglar [2013] proposed a decentralized and asynchronous version, but restricted to only activating subsets of the agents based on the optimization constraints. Teixeira et al. [2016] proposed optimal parameters for quadratic problems in a decentralized but synchronous context.

2.5 Consensus Algorithms

The consensus problem, or agreement problem, is a distributed computing problem with the goal of reaching agreement on a final value, typically in a setting with failing nodes and limited computational and communication resources.

An application for consensus algorithms is distributed cooperative control of multi-agent systems, for example in formation control of unmanned vehicles, where a group of unmanned aircrafts or underwater vehicles can drastically increase the efficiency in surveillance and search-and-rescue operations. The models can be divided into continuous-time models, capturing vehicle dynamics, and discrete-time models, capturing packet-based communication protocols. Consensus algorithms for multi-agent systems have been studied by Fax and Murray [2004], who focused on the information flow, while Olfati-Saber and Murray [2004] studied time-varying topologies, which they later developed further [Olfati-Saber et al., 2007]. Ji and Egerstedt [2007] considered the problem of keeping the multi-agent system connected, when the topology is state-dependent, while Blondel et al. [2009] considered a similar model for opinion dynamics, where the agents are separated into clusters. Seyboth et al. [2013] studied event-based strategies, both for single and double integrator vehicle dynamics models, while Guo and Dimarogonas [2013] considered quantization effects.

Cao et al. [2005] and Blondel et al. [2005] focus instead on the discrete model for coordination of multi-agent systems, while Xiao et al. [2005] and Speranzon et al. [2006] considers a similar model for sensor fusion in wireless sensor networks. Time-varying topologies and delays has been further studied by Sun et al. [2008]. Fagnani and Zampieri [2008] considered probabilistic average consensus problems, while Silvestre et al. [2013] considered consensus algorithms where an attacker is injecting perturbations, and Como and Fagnani [2016] studied average consensus on weakly connected networks.

Next, we consider two specific consensus algorithms: to reach the average and the maximal initial value, respectively. Both of these algorithms assumes a strongly connected network $\mathcal{G}$, where each node $i \in \mathcal{V}$ is given an initial real value $x_i(0) \in \mathbb{R}$. Each node then updates its own value based only on the values of its neighbors $\mathcal{N}_i$, without any influence from the rest of the network.

Average Consensus

The average consensus algorithm is based on linear iterations over the neighbors [Xiao and Boyd, 2003, Xiao et al., 2006]. The goal is to let every node's value converge to the average of the initial values,

$$x_i(t) \xrightarrow{t\to\infty} \frac{1}{N}\sum_{j\in\mathcal{V}} x_j(0), \quad \forall i \in \mathcal{V}.$$

Each node follows the update rule

$$x_i(t+1) = w_{ii}x_i(t) + \sum_{j\in\mathcal{N}_i} w_{ij}x_j(t),$$

for some weights w_{ii} and w_{ij}. If W is the weight matrix with entries w_{ij} (or 0 if w_{ij} is not present in the updates above), then, with $x = \left[x_1, x_2, \cdots, x_N\right]^{\mathrm{T}}$, the entire network update can be condensed into

$$x(t+1) = Wx(t).$$

This system will converge to the average values, for an arbitrary initial condition, if and only if the weight matrix satisfies

$$\lim_{t\to\infty} W^t = \frac{1}{N}\mathbb{1}\mathbb{1}^{\mathrm{T}}$$

An equivalent condition is that W is a double stochastic matrix, and that

$$\rho(W - (1/N)\mathbb{1}\mathbb{1}^{\mathrm{T}}) < 1.$$

Notice that the convergence is asymptotic in the number of iterations.

Max Consensus

The max consensus algorithm has the goal to let every node's value converge to the maximum of the initial values,

$$x_i(t) \xrightarrow{t\to\infty} \max_{j\in\mathcal{V}} x_j(0), \quad \forall i \in \mathcal{V}. \tag{2.5}$$

The local update consists of taking the maximum of the neighbors' values,

$$x_i(t+1) = \max_{j\in\mathcal{N}_i} x_j(t). \tag{2.6}$$

This protocol can be implemented with either gossip or broadcast communications. In the latter case, agents sequentially broadcast their local values, and whoever receives this information updates its local value with the maximum. Under weak assumptions on the communication process, the max consensus protocols are proven to converge to the true maximum in a finite time, bounded by the diameter of the network [Iutzeler et al., 2012].

2.6 Transportation Networks

The transportation network is an important example of a network system, where intersections can be modeled as nodes, and roads as the edges. Transportation of both goods and people is essential for the function of our society. The transportation system is facing great challenges, as the demand is steadily increasing, while the cost and environmental impacts need to minimized.

Transportation research is a vast and active field, but in this thesis we are mainly concerned with the subfield of intelligent transportation systems (ITS), see Figure 2.4. An intelligent transportation system is defined by the use of information and communication for transportation, including infrastructure, vehicles and users. A long-term goal is to provide a completely autonomous transportation system, thereby increasing the safety and efficiency. Even the subfield of ITS is vast, with surveys carried out by An et al. [2011], Figueiredo et al. [2001], Gusikhin et al. [2008], Li et al. [2014], Toral et al. [2010], Zhang et al. [2011], Zheng et al. [2014]. Next follows a description of some of the relevant work.

The transportation flow and congestion problem in the transportation system has been studied for over 80 years [Greenshields, 1935]. The distribution problem between a set of origins and a set of destinations was formulated by Hitchcock [1941], and spawned a wide research in optimal flow allocation [Edmonds and Karp, 1972, Ford and Fulkerson, 1957, Koopmans, 1949]. Research continues today with combinatorial pick-up and delivery optimization [Treleaven et al., 2013], but many of the combinatorial optimization problems are NP-complete, and therefore intractable to solve, including the original traveling salesman problem [Garey and Johnson, 1979].

A challenge for freight carriers is the need to move empty vehicles to avoid an accumulation of empty vehicles in a region, known as dead mileage. The U.S. Bureau of Transportation Statistics [2015] reports that 20 % of the truck mileage in 2002 consisted of empty vehicles, and other studies have estimated that up to 40 % of both the mileage and cost in different transportation scenarios are due to empty vehicles [Dejax and Crainic, 1987, Muyldermans et al., 2002]. Thus, there is an enormous potential for optimizing the freight transportation systems [Crainic and Laporte, 1997]. The transition towards just-in-time supply chains, employed to minimize the waste in the merchandise industry, is affecting the entire logistics chain [Lai and Cheng, 2009] and transforming the requirements for the transportation industry further. A consequence is the need for real-time transportation planning and adaption of transportation assignments [Mes et al., 2007, Yang et al., 2004].

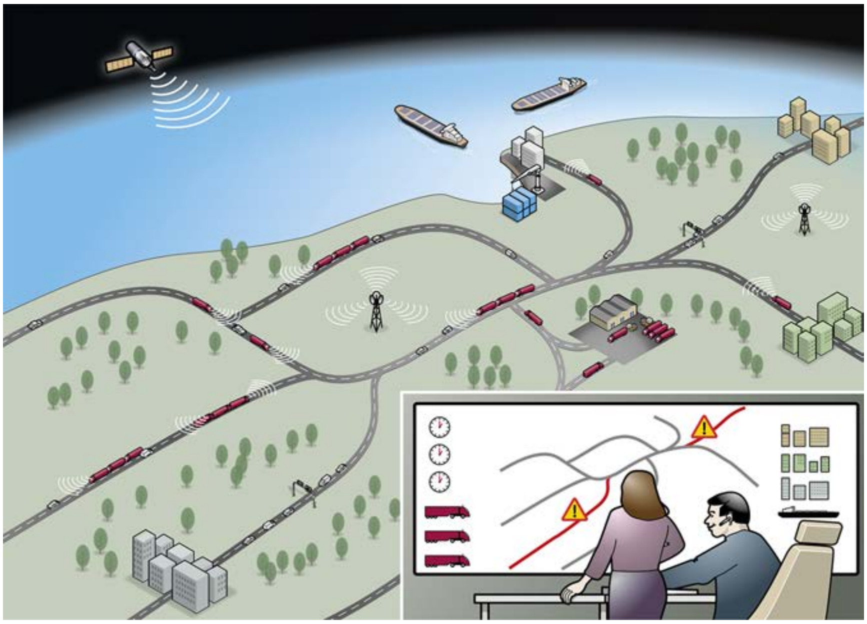


Figure 2.4: An intelligent transportation system, using cooperative control. Vehicles are communicating vehicle-to-vehicle to form platoons, and vehicle-to-infrastructure to optimize a just-in-time logistics operation.

Large-scale transportation service providers are also interested in optimizing the entire fleet management system [Crainic and Laporte, 1998], which involves the design of distribution center locations [Chen, 2001, Perl and Daskin, 1985].

The intelligent infrastructure development includes building automated highway systems [Horowitz and Varaiya, 2000, Varaiya, 1993], ride-sharing lanes [Kwon and Varaiya, 2008] and reversible lanes [Bede et al., 2010], but also traffic light control [Zhao and Chen, 2003]. On the vehicle side, current research is making cooperative vehicle platooning a reality [Alam et al., 2015, Mårtensson et al., 2012], which has been shown to reduce aerodynamic drag and thereby create fuel savings of up to 10 %. Besselink et al. [2016] considers the fleet management process for platooning.

Real-time data gathering has been proven to increase the efficiency and flexibility in the planning of transport assignments [Zheng et al., 2014], and over the last decades, technology development has enabled a widespread adoption of GPS receivers for determining the position of vehicles, while smart-phones has made it easy to collect and share this position data. This means that today we have access to huge datasets of trajectories from past transportation assignments, and this has opened up new opportunities for understanding transportation patterns [Gidófalvi and Pedersen, 2007, Jenelius and Koutsopoulos, 2013, Matsubara et al., 2013]. City authorities have been collecting data from the transportation system for a long time, but recently we have seen an increase in the availability of extensive datasets. For example, Lee et al. [2008], Moreira-Matias et al. [2012], Wang et al. [2012] and Yuan et al. [2013] studied strategies for cruising taxis and dispatch systems, while [Zhan et al., 2014] estimated the efficiency of a taxi system using a model of perfect prior knowledge of the demand. A related work includes the strategies for repositioning bikes in bike-sharing systems [Raviv et al., 2013]

2.7 Anonymous Networks

Glancy [1995] highlighted the importance of privacy in the development of intelligent transportation systems. Here we focus on a general computational model for networks requiring anonymity. Angluin [1980] introduced an anonymous computation framework by asking which functions could be computed by a local algorithm. The anonymity is assumed to be the lack of unique identifiers, instead each node only has a port numbering of its neighbors, also called a local edge labeling. This is the only information regarding the network that is known to each node.

The work by Angluin [1980] spawned an active field of characterizing computational functions in anonymous networks, notably the works by Boldi and Vigna [2001], Yamashita and Kameda [1988, 1996a,b]. Many results are of the negative form in the deterministic case, i.e., functions cannot be computed in this framework. The fundamental computational barrier is due to possible symmetries in the network, making it impossible to distinguish two messages from identical nodes at symmetric positions. Especially, there does not exist a deterministic algorithm guaranteed to compute the exact network size in an anonymous network [Cidon and Shavitt, 1995, Itai and Rodeh, 1990].

Extending the anonymous model to randomized algorithms, where each node is able to generate random bits, enables a larger class of functions to be computed [Codenotti et al., 1997, Itai and Rodeh, 1990]. For example, each node could then generate, with high probability, a unique identifier. After this step, the nodes could repeatedly interchange their local views, thereby recognizing the entire network topology. However, the methods based on exchanging local views uses growing message sizes and memory consumption.

2.8 Network Estimation

The importance of distributed estimation is reflected by the variety of applications where agents interact and cooperate to reach a common goal. Examples of these systems include environmental monitoring [Lynch et al., 2008], management of the electrical grid [Bolognani and Zampieri, 2013] and the public transportation system [Herring et al., 2009].

A common approach to network size estimation is to use *random walks* [Gkantsidis et al., 2006, Massoulie et al., 2006, Ribeiro and Towsley, 2010], relying on a token being passed around the network to collect information each time it visits an agent. Another strategy is to use randomly generated numbers [Kostoulas et al., 2007], and then exploit classical results on order statistics to infer the number of participants [Baquero et al., 2012, Cardoso et al., 2009, Chassaing and Gerin, 2006, Giroire, 2009, Lumbroso, 2010, Varagnolo et al., 2013]. These probabilistic techniques have been statistically analyzed [Cichon et al., 2012b, Clifford and Cosma, 2012], and are extensions of methodologies for estimating sums over networks [Cohen, 1997, Mosk-Aoyama and Shah, 2008]. Other network size estimation schemes use the *capture-recapture* concept [Peng et al., 2009, Petrovic and Brown, 2009], where seed numbers are randomly disseminated through the network, and then, by counting how many seeds are in a given subset, inferring the size of the network. Some studies [Cichon et al., 2011, 2012a] exploit probabilistic counting algorithms [Flajolet et al., 2007] usually implemented in non-distributed contexts. Other techniques take advantage of their specific framework and are not implementable in general settings [Ali et al., 2009, Dolev et al., 2006, Howlader et al., 2008, Leshem and Tong, 2005, Naini et al., 2015].

The previous studies mainly dealt with static networks, but there are also some extensions to dynamic settings. Fusy and Giroire [2007] used order statistics, Psaltoulis et al. [2004] considered random walks, Chabchoub and Hébrail [2010] exploited probabilistic counting algorithms and Alouf et al. [2002, 2004] considered multicast applications, while Shafaat et al. [2008] estimated the size in ring-based overlay networks.

When it comes to estimating probability mass functions over networks, the literature can be divided into parametric and non-parametric approaches. Parametric approaches assume the estimand to have a certain structure before obtaining observations, e.g., to be a sum of Gaussian distributions. Distributed implementations of the expectation-maximization (EM) algorithm [Forero et al., 2008, Jiang and Jin, 2006, Nowak, 2003, Vlassis et al., 2005] are examples of a parametric approach. Non-parametric approaches, on the other hand, do not assume a fixed structure a-priori, but rather select it from the observations. Kernel density estimation [Hu et al., 2007], classification [Klusch et al., 2003] and clustering approaches [Nguyen et al., 2005] are all examples of non-parametric approaches.

The literature can also be characterized by how information is propagated and aggregated over the network. There are strategies based on pre-established hierarchic tree routing structures, where the nodes compute the distributions in their sub-trees and propagate the information towards the root [Greenwald and Khanna, 2004, Madden et al., 2002, Motegi et al., 2006, Shrivastava et al., 2004]. Borges et al. [2012], Haridasan and van Renesse [2008], Sacha et al. [2010] all used gossip communications, and exploit averaging techniques to explicitly compute the cumulative distribution functions, while Cheng et al. [2010], Massoulie et al. [2006] estimated how many agents are in a specific state.

Recent work by Lucchese et al. [2015] have considered a maximum likelihood estimation using random number generation and a bit-wise max consensus protocol.

2.9 Peer-to-Peer Networks

Peer-to-peer (P2P) networking is a computer network architecture, where the nodes or peers both supply and consume resources equally. Thus, compared to a classical client–server architecture, in a peer-to-peer network, peers are both clients and servers at the same time. Androutsellis-Theotokis and Spinellis [2004] provided a survey of P2P content distribution technologies, Risson and Moors [2006] surveyed P2P search methods, while Meshkova et al. [2008] discussed P2P resource discovery solutions, and Liu et al. [2008], Thampi [2013] surveyed P2P video streaming systems.

An important utility for designing P2P architectures is the *peer sampling services*, which provides uniformly random samples of peers from the network. Gossip-based peer sampling systems have been developed by Jelasity et al. [2004, 2007], extended to handle NAT traversal by Payberah et al. [2011], and was corrected for bias in networks with churn by Baldoni et al. [2010].

Randomized gossiping algorithms have also been used as tools for building distributed systems, in particular in the areas of overlay networks, sensor networks and cloud computing storage services [Boyd et al., 2006, Kermarrec and van Steen, 2007]. Convergence properties of gossip-based aggregation algorithms have been studied for fixed topologies [Olshevsky and Tsitsiklis, 2006] and accelerated methods for regular graphs, where each node has the same number of neighbors [Liu et al., 2009].

Research in gossiping has also focused on using the preferential connectivity model [Mihail et al., 2003] to construct overlay network topologies, where nodes initially connected in a random graph use a preferential connection function to break the symmetry of the random graph, and build a topology that contains useful global information. Barabási [2002] first described how a preferential attachment function in a growing network can build a scale-free network topology from a random graph. Barabási's preferential attachment functions are based on the global state, but in overlay networks, nodes only have a relatively small partial view of the system. Thus, the preference functions can only be based on the local state and the state of the node's neighbors. Examples of overlay networks that construct their topologies using gossiping and preference functions include Spotify, that preferentially connects nodes with similar music play-lists [Kreitz and Niemelä, 2010], Sepidar, that preferentially connects P2P live-streaming nodes with similar upload bandwidth capacity [Payberah et al., 2010b], and T-Man, a framework that provides a generic preference function for building such overlays [Jelasity et al., 2009].

A fundamental property of P2P networks is user churn, i.e., that peers can join and leave the network at any time. Stutzbach and Rejaie [2006] worked on characterizing churn models, while Raftopoulou and Petrakis [2010], Baldoni et al. [2006] and Kuhn et al. [2005] consider resilience against churn, and Wang et al. [2008] chose to identify stable peers.

Chapter 3

Efficiency in Transportation Networks

"People are so bad at driving cars that computers don't have to be that good to be much better. Any time you stand in line at the DMV and look around, you're like, "Oh, my God, I wish all these people were replaced by computer drivers"."

— Marc Andreessen

Enabling efficient transportation is a major challenge for large cities, as the transportation need is increasing, while the environmental impact has to be minimized. In this chapter we consider collaborative transportation as a means to improve the transportation efficiency. For example, a transportation provider might have unused capacity on certain routes, which could be utilized by other actors.

In the first part of this chapter, we define an efficiency measure that shows how much of the current transportation mileage that is really necessary to meet all the transportation assignments. It is used for analyzing large datasets of trip trajectories, and determines where the mileage could be reduced in an ideal setting when the actors collaborate. We show that the efficiency measure can be computed efficiently as a minimum-cost flow, and we apply it on two case studies. The first case demonstrates the efficiency measure on a freight transportation system in Germany, and the second case computes the measure for a large real-world data set from the New York City taxis. Both of these examples show a large potential reduction in transportation mileage by collaboration.

In the second part of this chapter, we study a transportation system, modeled as a single transportation route with multiple cities. In this model, we consider the optimal idling locations and the optimal locations for distribution centers for a freight transport scenario. These locations are then exploited to improve the performance in a transportation simulation.

The remainder of this chapter is organized as follows: In Section 3.1, a new efficiency measurement for transportation systems is introduced, and some of its properties are presented. In Section 3.2 the efficiency measure is demonstrated on a freight transportation system, and in Section 3.3 the efficiency measure is computed on a huge dataset from the New York City taxi system. Then, in Section 3.4, a one-dimensional road transportation model is introduced. In Section 3.5, the optimal waiting locations for idling vehicles are considered, and an explicit solution for uniformly distributed random assignments is derived. Section 3.6 continues with the optimal locations for distribution centers in order to minimize the traveling time, and an explicit solution for uniformly distributed random assignments is derived, as well as an efficient algorithm for computing the locations for discrete random distributions. Section 3.7 introduces time-space diagrams, which provides a natural way of extending the optimal idling locations to dynamical scenarios. In Section 3.8, the strategies are evaluated with numerical simulations, before the conclusions of this chapter in Section 3.9.

3.1 Transportation Efficiency

In this section, we define a transportation efficiency measure, and show some of its properties. The aim is to measure the efficiency of a transportation system a-posteriori, from vehicle trajectories. The actors are assumed to be homogeneous, thus allowing transportation assignments to be exchanged, and we also ignore challenges with driver scheduling [Goel and Kok, 2012]. The efficiency measure evaluates the a-posteriori trajectories compared to an idealized case, where all actors would collaborate to satisfy the demands. To this end, it is assumed that for every trajectory, the vehicle is labeled as either being occupied or vacant.

Network Model

Here, a network flow model is introduced for the transportation problem, considering the network in a static or time-slotted scenario. A directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is given, where $\mathcal{V}$ is the set of nodes, and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ is the set of directed edges. For each edge $(u, v) \in \mathcal{E}$, there are two associated transportation flows, $f_1 : \mathcal{E} \to \mathbb{R}_+$ and $f_0 : \mathcal{E} \to \mathbb{R}_+$. The first flow, f_1, represents the desired *transportation assignments*, while the second flow, f_0, represents the vacant trips taken in order to move the vehicles to their next transportation assignments, which we refer to as *vacant flow*. Thus, all occupied trips passing an edge in the network are aggregated into the flow over the edge, and correspondingly, the vacant trips are aggregated into the vacant flow. Each edge $(u, v) \in \mathcal{E}$ also has an associated weight $w : \mathcal{E} \to \mathbb{R}_+$, which is the cost of transporting one unit of flow across the edge.

Transportation Efficiency Measure

Next, let us consider a measure for the efficiency of the transportation system. First, define the network flow cost C as the total cost for all trips (both the transportation assignments and the vacant flow), thus

$$C \doteq \sum_{(u,v)\in\mathcal{E}} w(u,v)\left(f_0(u,v)+f_1(u,v)\right).$$

Consider an optimal efficiency scenario, where the transportation assignments, given by f_1, have to be completed, but where the vacant vehicle flow, given by f_0, should be minimized while still preserving the availability of vehicles at each node. We formulate the following optimization problem:

$$\begin{aligned} f_0^\star \doteq \arg\min_{f} & \sum_{(u,v)\in\mathcal{E}} w(u,v)\left(f(u,v)+f_1(u,v)\right) \\ \text{subject to} & \\ & 0 \le f(u,v) \le f_0(u,v), \quad \forall\,(u,v)\in\mathcal{E}, \\ \text{and} & \\ & \sum_{\substack{v\in\mathcal{V}\\(u,v)\in\mathcal{E}}} f(u,v) - \sum_{\substack{v\in\mathcal{V}\\(v,u)\in\mathcal{E}}} f(v,u) \\ & = \sum_{\substack{v\in\mathcal{V}\\(u,v)\in\mathcal{E}}} f_0(u,v) - \sum_{\substack{v\in\mathcal{V}\\(v,u)\in\mathcal{E}}} f_0(v,u), \quad \forall\,u\in\mathcal{V}. \end{aligned} \tag{3.1}$$

The first constraint implies that the optimal vacant flow $f_0^\star$ is a subset to the vacant flow f_0, i.e., it does not increase the flow over any edge. The second constraint implies that the excess flow at every node is preserved, i.e., that the optimal flow preserves the same number of vacant vehicles at every node, ready for their next transportation assignments.

We can now define the optimal network flow cost as

$$C_{\text{opt}} \doteq \sum_{(u,v)\in\mathcal{E}} w(u,v)\left(f_0^\star(u,v)+f_1(u,v)\right),$$

and we are now ready for the transportation efficiency definition.

Definition 3.1. The *transportation efficiency measure* η is defined as

$$\eta \doteq \frac{C_{\text{opt}}}{C}.$$

Remark. We have $\eta \in [0,1]$, and $\eta = 1$ if the transportation system is optimal. A value $\eta < 1$ shows how inefficient the system is, as it measures the percentage of the trips that are necessary to fulfill all the transportation assignments.

Remark. Since the optimal flow $f_0^\star$ only reduces the initial vacant flow f_0, we neglect any changes in congestion that would appear from an increased traffic flow. Thus, computing this efficiency measure η can be done directly from historical GPS trajectories, as seen in Section 3.3.

Remark. We do not assume that the actual transportation assignments are known, but only the historical traces of the vehicle trajectories. Hence, it is natural to assume that all transportation assignments f_1 have to be fulfilled, and that they also represent the complete transportation demand.

Remark. Time constraints on the transportation assignments are not directly captured by this flow model. However, a possible solution is to only consider the trips that occur during a limited time period, and then recompute the efficiency measure for each time period.

Remark. Even though the entire transportation system is not homogeneous, we can consider a subsystem with homogeneous actors, e.g., a set of long haul trucking companies, or a set of taxi drivers, which could benefit from collaboration.

Example

As a simple example, consider a scenario with three companies (Blue, Red and Green) moving cargo between three cities (A, B and C), as in Figure 3.1. Company Blue moves cargo from city B to city A, and returns empty to pick-up the next cargo. Company Red similarly moves cargo from C to B, and returns empty, while company Green moves cargo from A to C, and returns empty, as illustrated in Figure 3.1a.

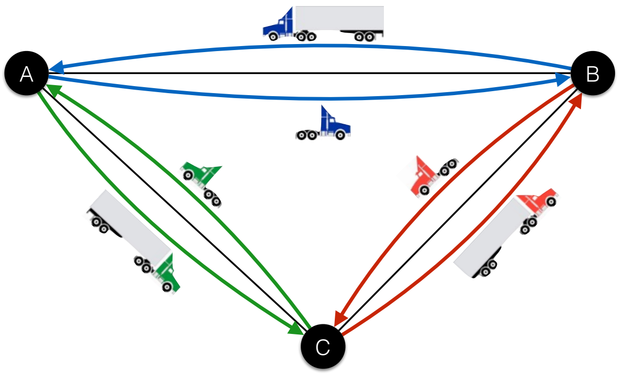
With a unit cost $w(u, v) = 1$ for all edges $(u, v) \in \mathcal{E}$, the total cost is $C = 6$. Notice that in this example, there is a cycle of empty trucks going around from A to B to C, and back to A, and that the transportation assignments could be served by a single truck going around from A to C to B and back to A, as shown in Figure 3.1b.

The cost for this optimized network is $C_{\text{opt}} = 3$, thus the efficiency of the transportation system is only $\eta = C_{\text{opt}}/C = 3/6 = 50\,\%$.

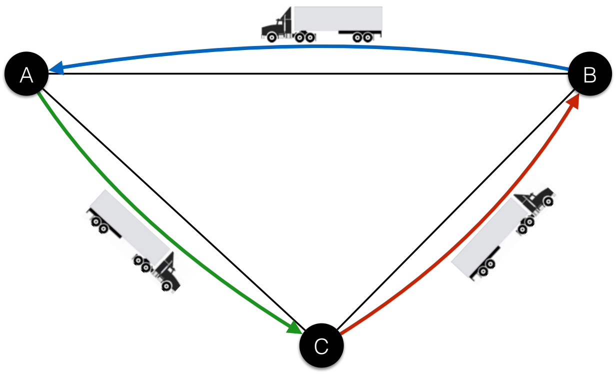
Computational Complexity

In this section, we show that the efficiency measure can be computed efficiently. In many practical scenarios, the road network consists of thousands of nodes, and there can be billions of collected transportation trajectories, as will be shown in Section 3.3. Therefore it is essential that the measure η can be computed efficiently.

The main computational step is to solve the optimal network flow problem in equation (3.1), which is equivalent to the *minimum-cost flow* problem. Recall the minimum-cost flow problem formulation [Ahuja et al., 1993], which can be stated



(a) The transportation network for three individual companies in an uncooperative scenario.



(b) The optimized transportation network for three companies in a cooperative scenario. All assignments can be handled by a single truck.

Figure 3.1: Illustration of three individual transportation companies, with and without cooperation. Without cooperation, six trips are necessary to complete the assignments, while with cooperation only three trips are necessary.

as

$$\min_f \sum_{(u,v)\in\mathcal{E}} w(u,v)f(u,v)$$

subject to

$$0 \le f(u,v) \le c(u,v), \quad \forall\,(u,v)\in\mathcal{E}, \tag{3.2}$$

and

$$\sum_{\substack{v\in\mathcal{V}\\(u,v)\in\mathcal{E}}} f(u,v) - \sum_{\substack{v\in\mathcal{V}\\(v,u)\in\mathcal{E}}} f(v,u) = b_u, \quad \forall\, u\in\mathcal{V},$$

where $c(u,v)$ is the edge capacity, and b_u is the node supply/demand. It is furthermore assumed by the *feasibility assumption* that $\sum_{u\in\mathcal{V}} b_u = 0$, and that there exists a feasible solution to equation (3.2).

Clearly, with the edge capacity given by the original vacant flow, $c(u,v) = f_0(u,v)$ for all edges $(u,v)\in\mathcal{E}$, and the node supply/demand given by the excess vacant flow, i.e.,

$$b_u = \sum_{\substack{v\in\mathcal{V}\\(u,v)\in\mathcal{E}}} f_0(u,v) - \sum_{\substack{v\in\mathcal{V}\\(v,u)\in\mathcal{E}}} f_0(v,u)$$

for all nodes $u\in\mathcal{V}$, our optimization problem in equation (3.1) is of the same form as equation (3.2). Further, by rearranging the sums, we see that

$$\begin{aligned}\sum_{u\in\mathcal{V}} b_u &= \sum_{u\in\mathcal{V}} \sum_{\substack{v\in\mathcal{V}\\(u,v)\in\mathcal{E}}} f_0(u,v) - \sum_{u\in\mathcal{V}} \sum_{\substack{v\in\mathcal{V}\\(v,u)\in\mathcal{E}}} f_0(v,u)\\ &= \sum_{u\in\mathcal{V}} \sum_{\substack{v\in\mathcal{V}\\(u,v)\in\mathcal{E}}} f_0(u,v) - \sum_{u\in\mathcal{V}} \sum_{\substack{v\in\mathcal{V}\\(u,v)\in\mathcal{E}}} f_0(u,v) = 0.\end{aligned}$$

Finally, it is straightforward to verify that $f = f_0$ is a feasible solution, thus we can conclude that we have a feasible minimum-cost flow problem.

A complexity survey of minimum-cost flow algorithms was presented by Kovács [2015]. He showed that the *generalized cost-scaling algorithm with dynamic trees* is one of the asymptotically fastest algorithms for minimum-cost flow problems, with time complexity $\mathcal{O}\left(nm\log(n^2/m)\min\{\log(nW), m\log n\}\right)$, where $n = |\mathcal{V}|$ is the number of nodes, $m = |\mathcal{E}|$ is the number of edges, and $W = \max_{(u,v)\in\mathcal{E}} w(u,v)$ is the largest edge weight.

We summarize this result in the following proposition.

Proposition 3.1. *The transportation efficiency measure η can be computed, using a minimum-cost flow algorithm, in polynomial time, asymptotically bounded by $\mathcal{O}\left(nm\log(n^2/m)\min\{\log(nW), m\log n\}\right)$.*

Remark. There are many algorithms for solving the minimum-cost flow problem. We used the *successive shortest path algorithm* [Edmonds and Karp, 1972] successfully for the following case studies, with a theoretical worst case performance $\mathcal{O}(D(m + n \log n))$, where D is the maximum flow value, using integer capacities.

Remark. As shown by Goldberg and Tarjan [1988], solving the minimum-cost flow problem is equivalent to removing all negative weight cycles from a feasible solution. This can be interpreted in our application as removing all cycles traveled by the empty vehicles, as illustrated in Figure 3.1.

3.2 Freight Transportation Case Study

In this section, we make a minor modification to the transportation efficiency measure η to evaluate cooperation policies between freight transportation companies on a simulated road system. The simulation consists of competitive transportation companies that receive transportation assignments, and independently optimize their own vehicle fleet once per hour. The network flow cost C is computed from the minimum-cost flow using the assignment as node constraints, and it is compared against the optimal flow cost C_{opt} given by full cooperation between the companies.

Let us consider three collaboration scenarios for a long haul freight transportation system, where a set of vehicles fulfills transportation assignments over the German road network, seen in Figure 3.2. The transportation network is given by the 14 largest cities in Germany, where the edge weights w are given by the travel distance between the cities in kilometers. Transportation assignments consist of a pick-up time and location, and a drop-off location. The pick-up and drop-off locations are randomly and independently selected with probability proportional to the population size of the cities. The pick-up times are randomly generated from the distribution of actively moving trucks, collected during 24 hours from a fleet management system, see Figure 3.3 and the work by Liang and Johansson [2014] for details. The average length of the generated assignments is shown in Figure 3.4.

The following is an example of three generated transportation assignments:

```
2015-01-01,03:45,Leipzig,51.3938,12.2523,Berlin,52.5018,13.2123
2015-01-01,04:11,Dortmund,51.4374,7.6016,Dresden,51.0503,13.6646
2015-01-01,04:17,Bremen,53.1032,8.8689,Munich,48.0983,11.6457
```

Each line specifies one assignment with the pick-up date and time, the pick-up location by city name and latitude–longitude coordinate, and the drop-off location by city name and coordinate.

As a base scenario, we consider perfect collaboration among all vehicle operators, where a central planner is minimizing the total empty transportation flow for completing all assignments, producing the optimal flow cost C_{opt}. The second scenario considers four competing, non-collaborating, vehicle operators. Each operator receives a fourth of all assignments, and is only concerned about minimizing their own fleet's empty transportation flow. The final scenario considers four non-collaborating regional operators, each operator located in one of the four regions

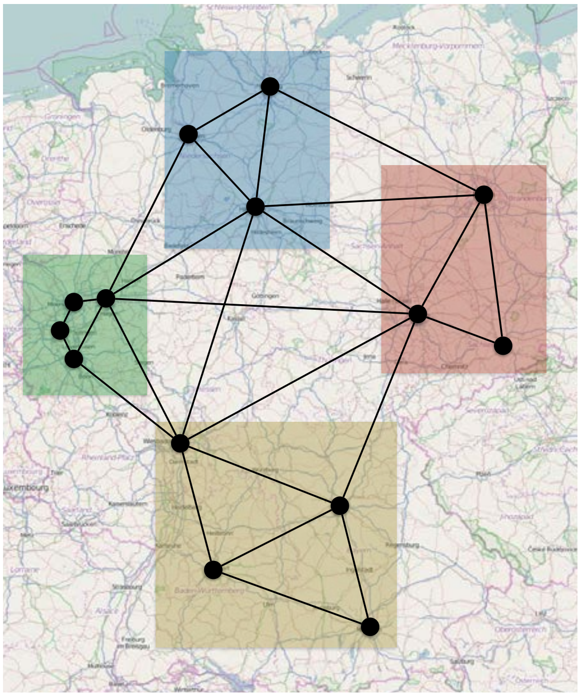


Figure 3.2: The road network between Germany's 14 largest cities represented as a graph. The cities are also grouped into four geographical regions. (Map courtesy of OpenStreetMap.)

shown in Figure 3.2. The company operating in a region receives all transportation assignments with the pick-up location in that region, but may need to deliver the cargo outside its region.

Given a set of 5 000 assignments per day, we optimize the transportation flows once per hour, considering all assignments created during that hour. The vehicles that have completed their assignments are available at their destination for a new assignment, while the vehicles still moving are unavailable. If there are not enough vehicles to serve all assignments, then the companies can add new vehicles, but at a randomly selected node with probability proportional to the population of the city. For the scenario with regionally restricted companies, the vehicles appear only within the company's region. Similarly, if there are more vehicles available than assignments, then the company needs to return the vehicles to some nodes, selected by the same random distribution. The transportation policy generated during each hour thus consists of the vacant flow created by driving the trucks to their next assignment, followed by the transportation flow determined by the assignment. We repeat these Monte Carlo simulations for 1 000 days in order to produce a daily

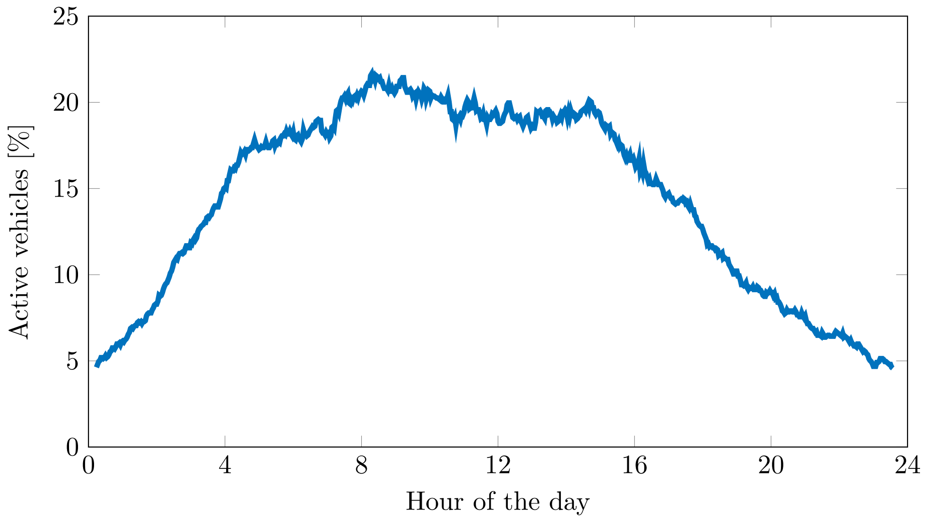


Figure 3.3: The proportion of actively moving vehicles during 24 hours. Data from Scania's fleet management system [Liang and Johansson, 2014].

average traveled distance.

The total traveled distance, e.g., the network flow cost C, per hour for the three scenarios is shown in Figure 3.5. With full collaboration, the total traveled distance of the empty vehicles is only 10 % of the total traveled distance (comprising both the assignments and the empty vehicles), while with four identical non-collaborating companies, the empty traveled distance increases to 17 % of the total. Notice that the empty traveled distance thus increased by 70 %. However, with geographically divided regional companies that do not take advantage of collaboration, and thus have no backhauling when returning to their region, the empty traveled distance increases to 47 % of the total traveled distance. This is also expressed by the transportation efficiency μ, shown in Figure 3.6, where the geographically restricted companies only achieve $\mu \approx 59\,\%$, while the non-collaborating companies achieve $\mu \approx 92\,\%$ efficiency compared to full collaboration. This clearly illustrates the importance of collaboration in the freight transportation system.

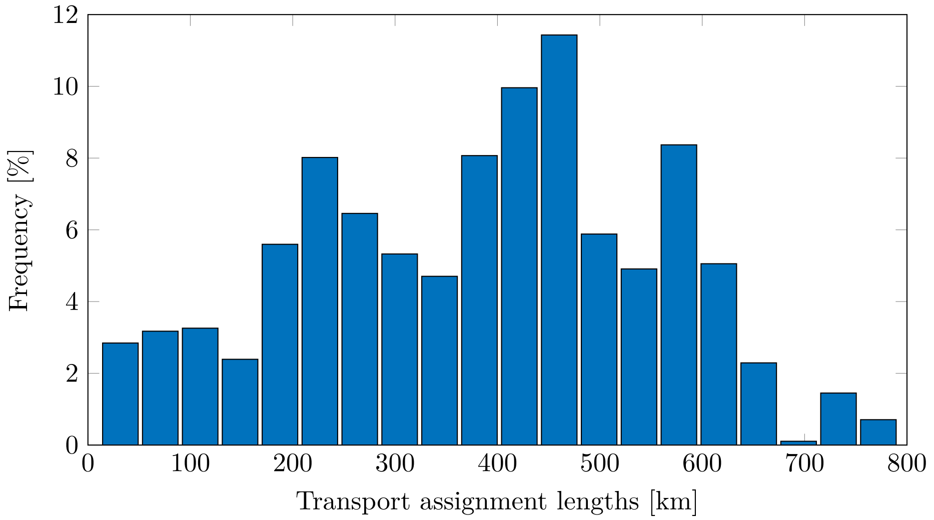


Figure 3.4: The length distribution of the transportation assignments in the collaborative freight simulation.

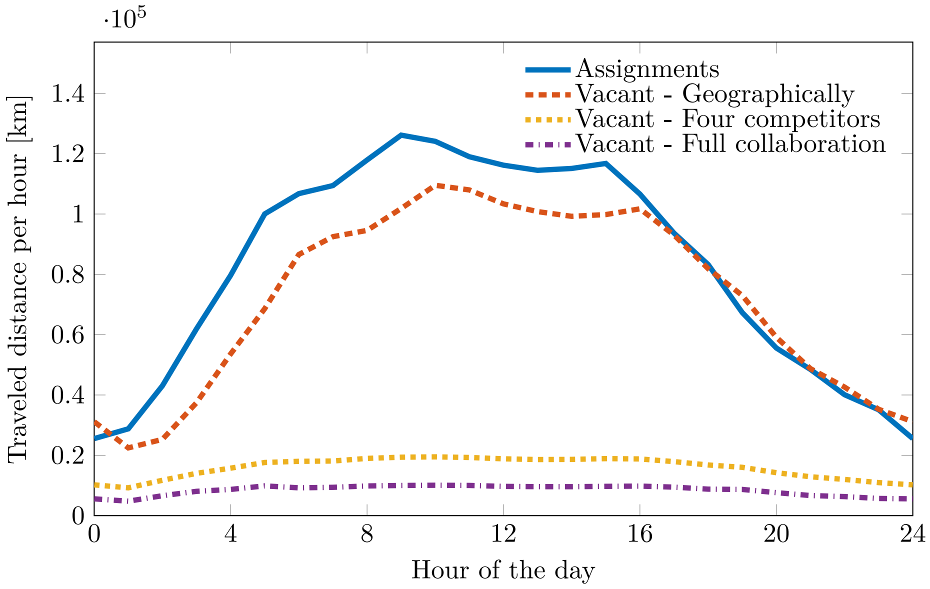


Figure 3.5: Evaluation of the transportation efficiency simulation for the three collaborative scenarios. The blue line shows the traveled distance for the assignments, while the other lines show the traveled distance for the empty vehicles in three scenarios. The first case is with four geographically restricted companies, the second case with four competing companies, and the final case is with full collaboration.

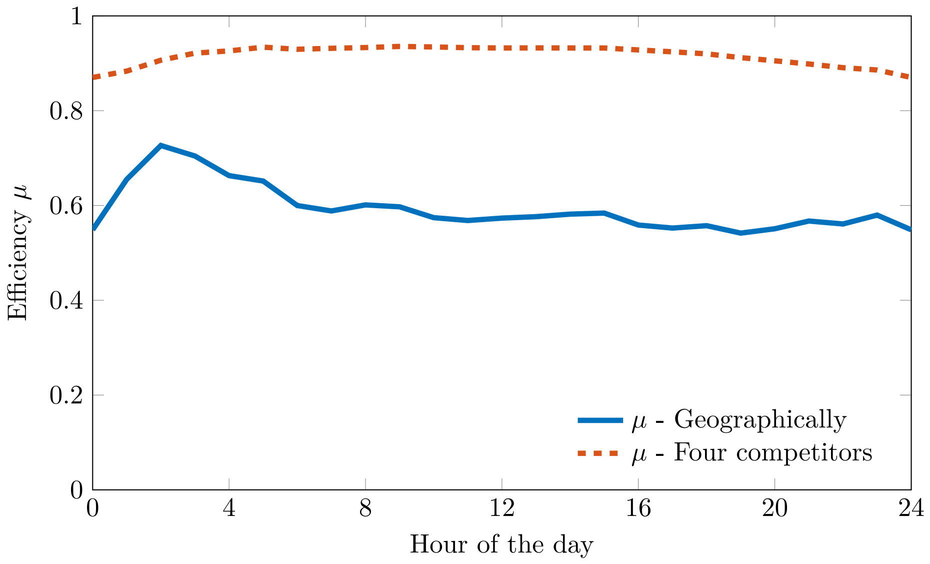


Figure 3.6: Efficiency μ in the freight transport simulation. The efficiency compares the four non-collaborating companies ($\mu \approx 92\,\%$) and four geographically restricted companies ($\mu \approx 59\,\%$) against the full collaboration scenario.

3.3 New York City Taxi Case Study

We now turn to a second illustration of the efficiency measure η, computed on a real dataset from the New York City taxis. We will show how the efficiency of the taxi system varies depending on the time of the day.

In New York City, the taxi system consists of more than 13 000 yellow medallion taxis, which completed more than 174 million trips during 2013. Records from all these trips have been made publicly available by the NYC Taxi & Limousine Commission. Each trip in the dataset is specified with the following data fields:

- Car ID
- Driver ID
- Pick-up time
- Drop-off time
- Passenger count
- Trip distance
- Pick-up location (GPS position)
- Drop-off location (GPS position)

The dataset is illustrated in Figure 3.7, showing a heat map for the active taxis, where a majority of the trips are centered on Manhattan Island.

To compute and analyze the efficiency measure η we first need to construct the flow network from the taxi data. This procedure is described in the next four subsections.

Reconstructing the Vacant Flow

The dataset contains the transportation assignments, i.e., when the taxis are driving with passengers, but does not include the cruising trips when the driver is looking for new passengers. However, since the transportation assignments are specified with both a car identity, a driver identity, as well as the time of the trip, we can determine the next trip for each car and driver. If a car–driver pair drove another trip within one hour, then we add the empty trip from the previous trip's drop-off location to the next trip's pick-up location.

For 91.3% of all transportation assignments, this condition could find a following vacant flow trip. The average transportation assignments and the vacant flow variation over a day is shown in Figure 3.8.

Remark. There is actually a dip in the number of available taxis in the middle of the afternoon rush, because the drivers traditionally change shifts at this time [Grynbaum, 2011].

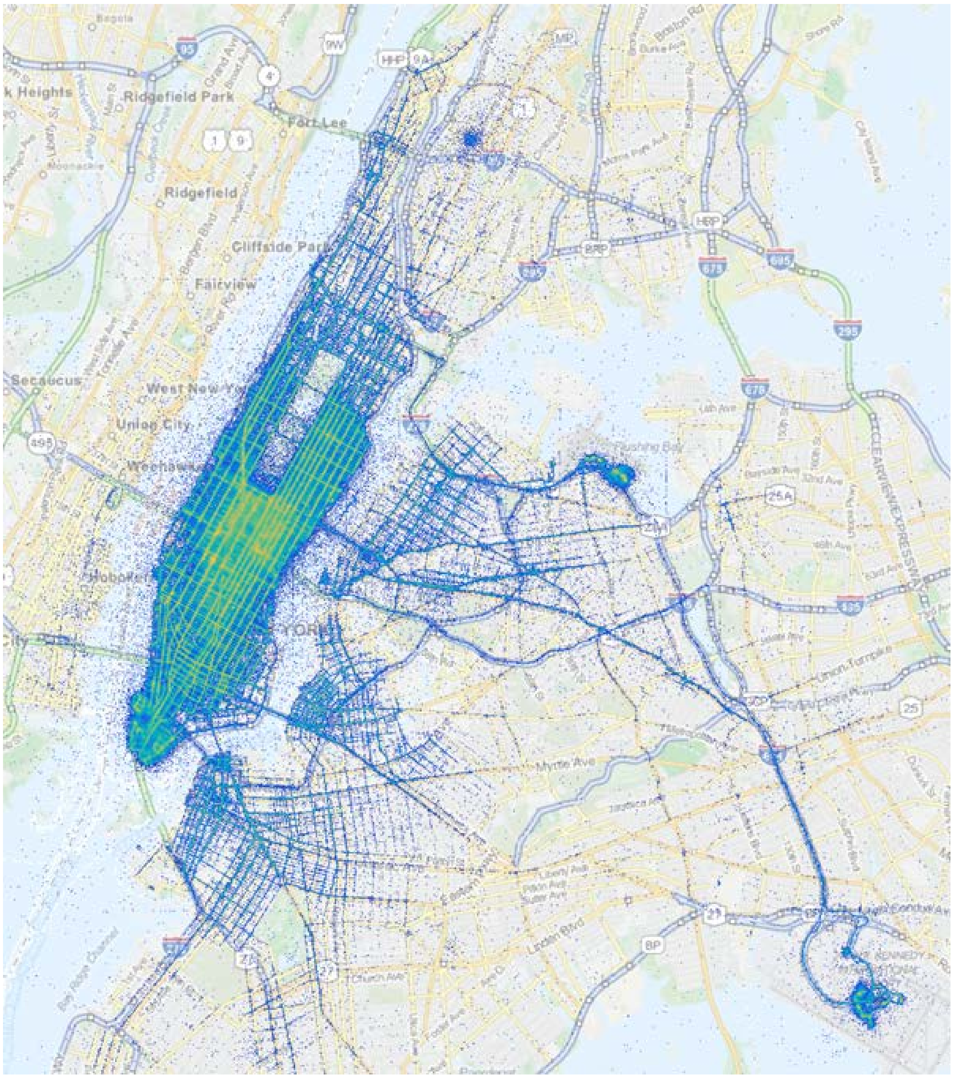


Figure 3.7: Heat map of the New York City taxis' pick-up locations. (Map courtesy of OpenStreetMap.)

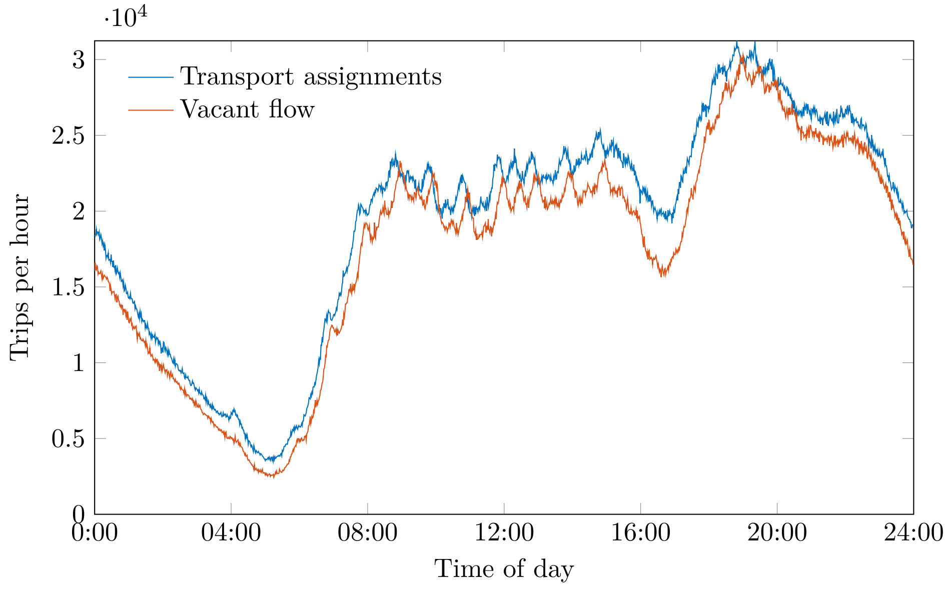


Figure 3.8: The average number of taxi trips in NYC, as a function of the time of day. Transport assignments correspond to trips in the dataset, and vacant flow corresponds to the identified relocations.

Remark. The vacant flow is created by assuming that the drivers drive the shortest path to their next assignment, thus underestimating the actual vacant mileage. The lack of passenger information often leads to taxis cruising along the streets, randomly looking for new passengers [Yuan et al., 2013].

OpenStreetMap Network

The trip data only contain the pick-up and drop-off locations, therefore we construct a network based on the road data from OpenStreetMap [2015]. The extracted map region around New York City contains 1 460 536 nodes and 2 967 562 edges, and the pick-up and drop-off locations for each trip are mapped to the closest nodes in the OpenStreetMap data.

The next step is to find the path through the road network for every trip. To this end, we compute the shortest path for each trip, taking into account the road type, speed limits and one-way directions. The result is a prediction of how the taxis were moving, similar to the suggestions given by GPS navigators.

Grid Regions

The OpenStreetMap data contain a very detailed road network, including many local phenomena, e.g., complex intersections, parallel lanes, antiparallel one-way streets, etc. But we are only interested in the general trajectories of the taxis, which would be hidden by excessive details of the map.

To address this problem, we divide the map into a square grid, where cell sizes from 100 m × 100 m up to 5 000 m × 5 000 m are tested. Each grid cell becomes a node in our final flow network, and a taxi trip is represented as a sequence of adjacent grid cells. Because the grid cells have the same size, we use a unit weight $w(u, v) = 1$ for all $(u, v) \in \mathcal{E}$, and the flow is equal to the number of taxis passing between two grid cells.

Remark. This has the additional benefit of reducing the number of nodes in the flow network, which makes the computations faster.

Computing the Efficiency Measure

We have now constructed the flow network from the pick-up and drop-off locations of the taxis, and from this description we are able to compute the efficiency η.

Given a year's worth of data, we introduce time slots, where the length is varied from 1 minute up to 60 minutes. For each time slot, we compute the efficiency measure η, and in Figure 3.9 the taxi transportation efficiency is shown as a function of the time of the day. In Table 3.1, the efficiency is shown for different grid sizes and time slots.

Results

The average efficiency measure over January 2013 is shown in Table 3.1. The efficiency of the NYC taxi system varies from 90.7% using 1 minute time slots and a grid of 100 m × 100 m, down to 83.8% for 60 minutes time slots and 5 000 m × 5 000 m grid size.

Notice that the efficiency drops when the time slots and grid size increases. This supports our intuition, because by increasing the time slots, we consider more vehicles at each step, and are therefore more likely to find vehicles that can be removed. Similarly for the grid size, a larger grid means that more roads will be part of the same flow edge, e.g., when two cars are traveling on parallel one-way streets, in opposite directions.

Remark. The time slot and grid size can be interpreted as how close two taxis need to be, in time and space, in order to be redundant.

In Figure 3.9, the efficiency is shown as a function of the time of day. Again, we see that a larger time slot yields a consistently lower efficiency. Notice that the efficiency has a peak during the afternoon rush at 5 p.m. Comparing with Figure 3.10, we see that this peak corresponds to a very low vacant flow, i.e., the high demand for taxis makes it easy to pick up new passengers. In contrast, the

Table 3.1: New York City Taxi efficiency results, for grid sizes from $(100\,\mathrm{m})^2$ to $(5000\,\mathrm{m})^2$, and time slots ranging from 1 min to 60 min, compared to an entire month's data.

			Time slot			
			1 min	5 min	60 min	1 month
Grid size	$(100\,\mathrm{m})^2$	Average flow	25409	127047	1524573	1132757899
		Vacant flow	28.1%	28.1%	28.1%	27.3%
		Efficiency η	90.7%	86.6%	84.7%	80.3%
	$(500\,\mathrm{m})^2$	Average flow	4724	23620	283442	210597439
		Vacant flow	27.6%	27.5%	27.5%	26.8%
		Efficiency η	88.2%	85.5%	84.4%	80.6%
	$(1000\,\mathrm{m})^2$	Average flow	2222	11111	133340	99072081
		Vacant flow	27.6%	27.6%	27.6%	26.9%
		Efficiency η	87.4%	85.1%	84.2%	80.6%
	$(5000\,\mathrm{m})^2$	Average flow	346	1730	20763	15427468
		Vacant flow	28.4%	28.4%	28.4%	27.8%
		Efficiency η	86.1%	84.5%	83.8%	81.2%

morning peak at 4 a.m. corresponds to a high percentage of vacant flow. Comparing also with Figure 3.8 explains that this is due to much fewer taxis being available in the morning, since there is a lower demand for taxis.

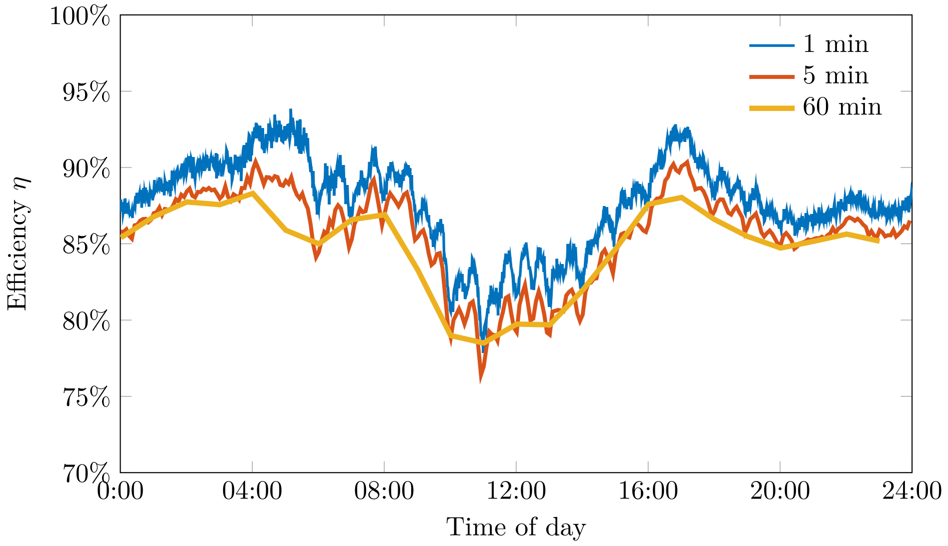


Figure 3.9: The average efficiency measure by the time of day, using a grid size of $1\,000\,\mathrm{m} \times 1\,000\,\mathrm{m}$.

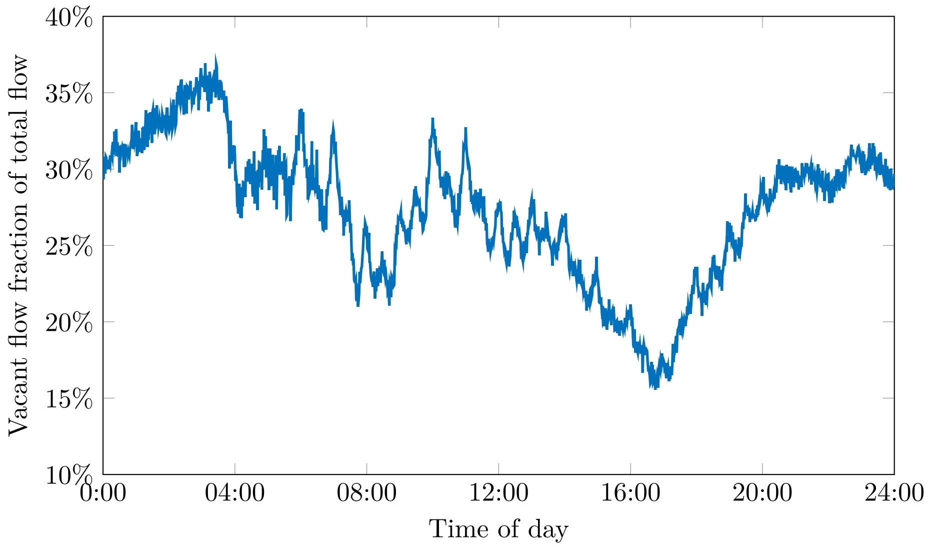


Figure 3.10: The fraction $f_0/(f_0 + f_1)$ of vacant flow compared to the total flow.

3.4 A One-dimensional Freight Transportation Model

In this section, we shift our focus from the efficiency measure to a real-time transportation service provider, operating along a single highway. Transportation service providers are under severe pressure to enable real-time logistics planning from a constantly changing demand. We consider a scenario where transportation assignments arrive following a Poisson process, and the transportation service provider is operating on this road system with a fleet of vehicles, trying to minimize the expected delivery time. In the next sections, the optimal locations for idle vehicles, and the optimal locations for distribution centers are considered. The strategies are evaluated with numerical simulations along a Swedish highway in Section 3.8.

Consider a road freight transportation system between two cities, for example the main highway connecting the two largest cities in Sweden, as depicted Figure 3.11. As shown, there are several major cities located along this road, being potential destinations for the transports. A position along this route can be represented with its relative position in the interval $[0,1]$. Thus, using this model, any position of a vehicle, or destination, is given by a real number in the interval $[0,1]$. Furthermore, positions in the interval $[0,1]$ can be scaled such that the difference $|x_1 - x_2|$ between two positions x_1 and x_2 is proportional to the transportation cost between these locations on the map, in terms of either travel distance, travel time or fuel consumption.

A real-time transportation provider is operating on this road system with a fleet of vehicles. Transportation assignments arrive randomly following a Poisson process with rate λ, and the pick-up location l_1 and drop-off location l_2 are sampled from a joint probability density function $\rho(l_1, l_2) : [0,1] \times [0,1] \to \mathbb{R}_+$, where we assume the transportation providers have prior knowledge about ρ. Each of the transportation provider's vehicles cycle through the states in Figure 3.12, where it starts in an idle state waiting for an assignment. After being selected for an assignment, it drives to the pick-up location to collect the goods. The assignment is then brought to a distribution center, before being delivered to the drop-off location, after which the vehicle is returned to an idle state.

The transportation provider evaluates its performance as the time it takes from receiving a transportation assignment until the delivery at the drop-off location. This time can be divided into three parts, the time it takes for a vehicle to arrive at the pick-up location, the time it takes to drive to the drop-off location, and the extra time spent visiting the distribution center. Here, the time taken to drive from the pick-up location to the drop-off location is given by the assignment and road conditions, and is outside the control of the transportation provider, but the time to pick up an assignment depends on the location strategy for the idle vehicles, and the extra time spent going to a distribution center depends on where the distribution centers are located. We consider both of these optimization problems in the next sections.

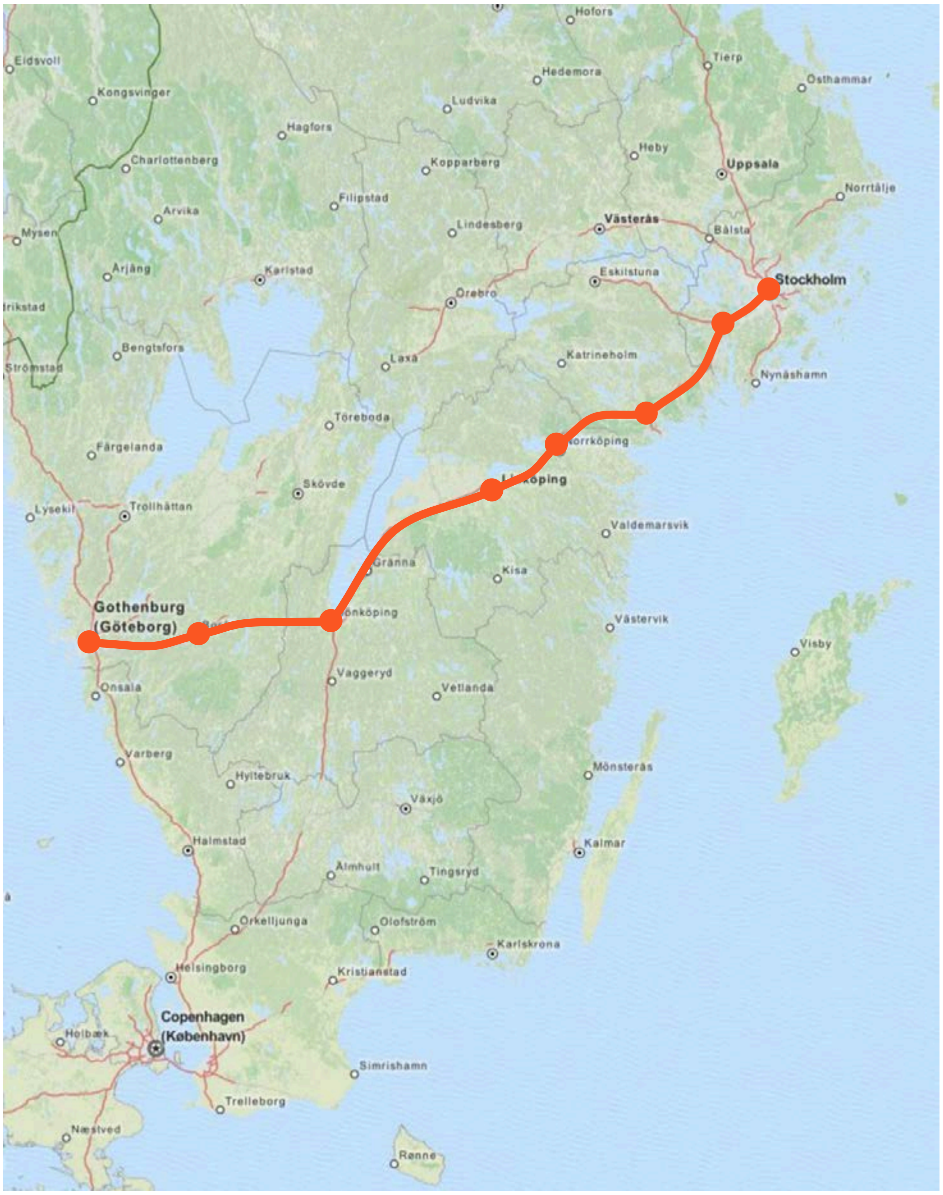


Figure 3.11: Map of southern Sweden, highlighting the main road connecting the largest cities Stockholm and Göteborg, together with the major cities along the road. (Map courtesy of OpenStreetMap.)

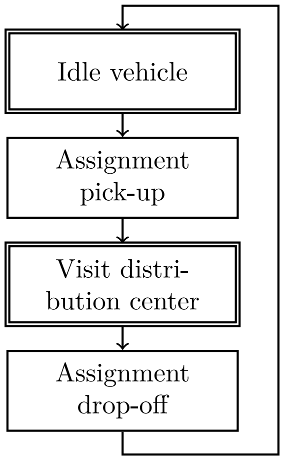


Figure 3.12: Flow chart of the states for each vehicle of a transportation provider. In this section, we focus on optimizing the idle vehicle locations and the distribution center locations.

3.5 Optimal Idling Location

In this section, the static optimization problem of deciding where idle vehicles should wait for their next assignment is considered. A transportation provider serving the system with N vehicles would like to distribute the vehicles to minimize the expected time to pick up the next assignment, where the pick-up location l_1 is randomly chosen from the probability density function $\rho(l_1) : [0,1] \to \mathbb{R}_+$. Let $x_1, \ldots, x_N$ denote the locations of the transportation provider's N vehicles, and $\mathbb{E}[\cdot]$ the expected value. The problem can then be formulated as

$$\min_{x_1,\ldots,x_N} \mathbb{E}_{l_1}\left[\min_{i=1,\ldots,N} |x_i - l_1|\right] = \min_{x_1,\ldots,x_N} \int_0^1 \left[\rho(l_1) \min_{i=1,\ldots,N} |x_i - l_1|\right] \mathrm{d}l_1. \quad (3.3)$$

Remark. In this formulation, the vehicles may stop at any location along the road, i.e., $x_i \in [0,1]$.

Uniform Distributions

We will now derive an explicit solution for the locations of the vehicles, when the transport assignments have a uniform probability distribution.

Proposition 3.2. *Assume that new transportation assignments arrive at locations following a uniform distribution $\mathcal{U}[0,1]$ over the road system, i.e., $\rho(l_1) = 1$ for all $l_1 \in [0,1]$. The optimal locations of the N vehicles is then equidistantly distributed over the line, with $x_i = \frac{2i-1}{2N}$, $i = 1, \ldots, N$.*

Proof. Without loss of generality, assume that $x_1 \leq x_2 \leq \cdots \leq x_N$. The integral in equation (3.3) can then be split into parts as

$$\begin{aligned}
&\int_0^1 \left[\min_{i=1,\dots,N} |x_i - l_1| \right] \mathrm{d}l_1 \\
&\qquad = \int_0^{x_1} (x_1 - l_1)\,\mathrm{d}l_1 \\
&\qquad\quad + \sum_{i=1}^{N-1} \left(\int_{x_i}^{(x_i+x_{i+1})/2} (l_1 - x_i)\,\mathrm{d}l_1 + \int_{(x_i+x_{i+1})/2}^{x_{i+1}} (x_i - l_1)\,\mathrm{d}l_1 \right) \\
&\qquad\quad + \int_{x_N}^1 (l_1 - x_N)\,\mathrm{d}l_1 \\
&\qquad = \underbrace{\frac{1}{2}x_1^2 + \sum_{i=1}^{N-1} \frac{1}{4}(x_{i+1} - x_i)^2 + \frac{1}{2}(1 - x_N)^2}_{G}.
\end{aligned}$$

Thus, the optimal vehicle locations $x_1, \dots, x_N$ should be chosen such that G is minimized, which happens when the gradient is zero:

$$\frac{\partial G}{\partial x_1} = \frac{3}{2}x_1 - \frac{1}{2}x_2 = 0 \quad \Rightarrow \quad 3x_1 = x_2$$

$$\frac{\partial G}{\partial x_i} = \frac{1}{2}(2x_i - x_{i-1} - x_{i+1}) = 0 \quad \Rightarrow \quad x_i = \frac{x_{i-1} + x_{i+1}}{2} \qquad \forall i \in 2, \dots, N-1$$

$$\frac{\partial G}{\partial x_N} = \frac{3}{2}x_N - \frac{1}{2}x_{N-1} - 1 = 0 \quad \Rightarrow \quad 3x_N = x_{N-1} + 2$$

Solving this equation system yields the solution where $x_i = \frac{2i-1}{2N}$ for all $i = 1, \dots, N$. □

3.6 Optimal Distribution Center Location

In this section, the static optimization problem of deciding where to build distribution centers is considered. Distribution centers are used to store and sort goods, and to coordinate transportation assignments efficiently. We assume that every piece of goods needs to visit a distribution center before being delivered to its final destination.

A transportation assignment consists of a pick-up location $l_1 \in [0, 1]$ and a drop-off location $l_2 \in [0, 1]$, and the goods is transported from the pick-up location to any distribution center before being delivered to the drop-off location. The goal is to decide where to build M distribution centers such that the expected total transportation cost is minimized. Let $\rho(l_1, l_2) : [0, 1] \times [0, 1] \to \mathbb{R}_+$ be the joint probability density function for an assignment to have the pick-up location l_1 and drop-off location l_2, and let $d_1, \ldots, d_M$ be the locations for the distribution centers. The optimization problem can be formulated as

$$\begin{aligned}\min_{d_1,\ldots,d_M} & \mathbb{E}_{l_1,l_2}\left[\min_{i=1,\ldots,M}\left(|d_i - l_1| + |d_i - l_2|\right)\right] \\ &= \min_{d_1,\ldots,d_M}\int_0^1\int_0^1\left[\rho(l_1,l_2)\min_{i=1,\ldots,M}\left(|d_i - l_1| + |d_i - l_2|\right)\right]\mathrm{d}l_2\,\mathrm{d}l_1. \qquad (3.4)\end{aligned}$$

Uniform Distributions

Assume that the pick-up and drop-off locations are i.i.d. random variables with uniform probability distribution $l_1, l_2 \sim \mathcal{U}[0,1]$, i.e., $\rho(l_1,l_2) = 1$. Let us first consider the case with only one distribution center.

Proposition 3.3. *The optimal location, d, for a single distribution center, when the assignment locations have uniform probability density $\rho(l_1,l_2) = 1$, is at $d = 1/2$.*

This is intuitively clear from a symmetry argument, but we will none the less prove it here.

Proof. The distribution center location d is determined by the following optimization problem.

$$\begin{aligned}\min_d & \mathbb{E}_{l_1,l_2}\left[|d - l_1| + |d - l_2|\right] \\ &= \min_d \int_0^1\int_0^1 \left(|d - l_1| + |d - l_2|\right)\mathrm{d}l_2\,\mathrm{d}l_1 \\ &= 2\min_d \int_0^1 |d - l|\,\mathrm{d}l \\ &= 2\min_d \left(\int_0^d (d - l)\,\mathrm{d}l + \int_d^1 (l - d)\,\mathrm{d}l\right) \\ &= 2\min_d \left(\frac{d^2}{2} + \frac{(d-1)^2}{2}\right) \\ &= \min_d \left(2d^2 - 2d + 1\right),\end{aligned}$$

which has the solution $d = 1/2$. □

We now proceed to the general case, with $M > 1$ distribution centers.

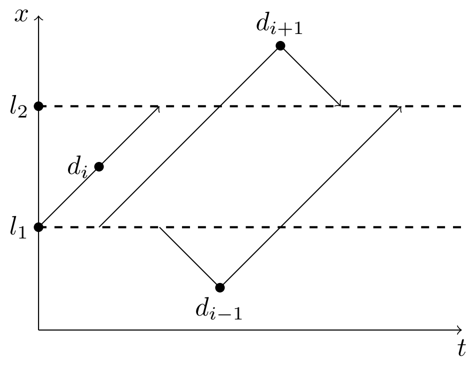


Figure 3.13: A schematic representation for the trip from l_1 to l_2, using three different possible distribution centers d_{i-1}, d_i or d_{i+1}. If there exists a distribution center d_i between the locations l_1 and l_2, then the direct path between them is optimal, otherwise a detour is needed to visit a distribution center d_{i-1} or d_{i+1}.

Theorem 3.4. *The optimal locations $d_1, \ldots, d_M$ for $M > 1$ distribution centers, when the assignment locations have uniform probability density $\rho(l_1, l_2) = 1$, are equidistantly spaced at $d_1, d_1 + (\frac{1-2d_1}{M-1}), d_1 + 2(\frac{1-2d_1}{M-1}), \ldots, d_1 + (M-1)(\frac{1-2d_1}{M-1}) = 1 - d_1$, with the boundary distance $d_1 = \frac{2-\sqrt{2}}{6-4\sqrt{2}+2M(\sqrt{2}-1)}$.*

Proof. The locations are determined by the following optimization problem:

$$\min_{d_1,\ldots,d_M} \mathbb{E}_{l_1,l_2}\left[\min_{i=1,\ldots,M} (|d_i - l_1| + |d_i - l_2|)\right]$$
$$= \min_{d_1,\ldots,d_M} \int_0^1 \int_0^1 \min_{i=1,\ldots,M} (|d_i - l_1| + |d_i - l_2|)\, \mathrm{d}l_2\, \mathrm{d}l_1.$$

Without loss of generality, we can assume that $d_1 \leq d_2 \leq \cdots \leq d_M$. When considering using a distribution center d_i for the assignment between l_1 and l_2, there are three possibilities, as illustrated in Figure 3.13. If d_i is between l_1 and l_2, then it is an optimal distribution center, since it is on the direct path between the two locations. Otherwise, if both locations l_1, l_2 belong to an interval $[d_i, d_{i+1}]$ for some i, then we need to consider both d_i and d_{i+1} as possible candidates, and the additional travel distance is $2 \cdot \min(\min(l_1, l_2) - d_i,\ d_{i+1} - \max(l_1, l_2))$ for visiting a distribution center. Using this property, we rewrite the double integral as

$$\begin{aligned}
&\int_0^1 \int_0^1 \min_{i=1,\ldots,M} \left(|d_i - l_1| + |d_i - l_2|\right) \mathrm{d}l_2 \, \mathrm{d}l_1 \\
&\qquad = \int_0^1 \int_0^1 |l_1 - l_2| \, \mathrm{d}l_2 \, \mathrm{d}l_1 \\
&\qquad + 2 \int_0^{d_1} \int_0^{d_1} \left(d_1 - \max(l_1, l_2)\right) \mathrm{d}l_2 \, \mathrm{d}l_1 \\
&\qquad + 2 \sum_{i=1}^{M-1} \int_{d_i}^{d_{i+1}} \int_{d_i}^{d_{i+1}} \min(\min(l_1, l_2) - d_i, \, d_{i+1} - \max(l_1, l_2)) \, \mathrm{d}l_2 \, \mathrm{d}l_1 \\
&\qquad + 2 \int_{d_M}^1 \int_{d_M}^1 \left(\min(l_1, l_2) - d_M\right) \mathrm{d}l_2 \, \mathrm{d}l_1 .
\end{aligned}$$

Notice that the first double integral is the transportation cost for driving between l_1 and l_2, which is independent of the distribution center locations $d_1, \ldots, d_M$, thus its value will not affect the minimization problem. Let us now compute the remaining three double integrals, which represent the extra traveling cost pertaining to the distribution centers. First,

$$\begin{aligned}
&\int_0^{d_1} \int_0^{d_1} \left(d_1 - \max(l_1, l_2)\right) \mathrm{d}l_2 \, \mathrm{d}l_1 \\
&\qquad = \int_0^{d_1} \left(\int_0^{l_1} (d_1 - l_1) \, \mathrm{d}l_2 + \int_{l_1}^{d_1} (d_1 - l_2) \, \mathrm{d}l_2 \right) \mathrm{d}l_1 \\
&\qquad = \frac{1}{3} d_1^3 .
\end{aligned}$$

Similarly,

$$\begin{aligned}
&\int_{d_M}^1 \int_{d_M}^1 \left(\min(l_1, l_2) - d_M\right) \mathrm{d}l_2 \, \mathrm{d}l_1 \\
&\qquad = \int_{d_M}^1 \left(\int_{d_M}^{l_1} (l_2 - d_M) \, \mathrm{d}l_2 + \int_{l_1}^1 (l_1 - d_M) \, \mathrm{d}l_2 \right) \mathrm{d}l_1 \\
&\qquad = \frac{1}{3} (1 - d_M)^3 .
\end{aligned}$$

Finally,

$$\begin{aligned}
&\int_{d_i}^{d_{i+1}} \int_{d_i}^{d_{i+1}} \min(\min(l_1, l_2) - d_i,\ d_{i+1} - \max(l_1, l_2))\, \mathrm{d}l_2\, \mathrm{d}l_1 \\
&\quad = \int_{d_i}^{\frac{d_i + d_{i+1}}{2}} \Bigg(\int_{d_i}^{l_1} (l_2 - d_i)\, \mathrm{d}l_2 + \int_{l_1}^{d_{i+1}+d_i-l_1} (l_1 - d_i)\, \mathrm{d}l_2 \\
&\qquad\qquad + \int_{d_{i+1}+d_i-l_1}^{d_{i+1}} (d_{i+1} - l_2)\, \mathrm{d}l_2 \Bigg) \mathrm{d}l_1 \\
&\quad\ + \int_{\frac{d_i + d_{i+1}}{2}}^{d_{i+1}} \Bigg(\int_{d_i}^{d_{i+1}+d_i-l_1} (l_2 - d_i)\, \mathrm{d}l_2 + \int_{d_{i+1}+d_i-l_1}^{l_1} (d_{i+1} - l_1)\, \mathrm{d}l_2 \\
&\qquad\qquad + \int_{l_1}^{d_{i+1}} (d_{i+1} - l_2)\, \mathrm{d}l_2 \Bigg) \mathrm{d}l_1 \\
&\quad = \frac{1}{6}(d_{i+1} - d_i)^3.
\end{aligned}$$

Hence, the optimization problem for the optimal locations $d_1, \ldots, d_M$ can be written as

$$\min_{d_1,\ldots,d_M} \underbrace{\left(\frac{2}{3} d_1^3 + \sum_{i=1}^{M-1} \frac{1}{3}(d_{i+1} - d_i)^3 + \frac{2}{3}(1 - d_M)^3 \right)}_{G}.$$

Thus, the optimal locations $d_1, \ldots, d_M$ for the distribution centers should be chosen such that G is minimized, which happens when the gradient is zero:

$$\frac{\partial G}{\partial d_1} = d_1^2 + 2d_1 d_2 - d_2^2 = 0 \quad \Rightarrow \quad d_1 = (\sqrt{2} - 1) d_2$$

$$\frac{\partial G}{\partial d_i} = (d_{i+1} - d_{i-1})(2d_i - d_{i+1} - d_{i-1}) = 0 \quad \Rightarrow \quad d_i = \frac{d_{i+1} + d_{i-1}}{2} \qquad \forall i \in 2, \ldots, N-1$$

$$\frac{\partial G}{\partial d_M} = d_{M-1}^2 + 2d_M(2 - d_{M-1}) - d_M^2 - 2 = 0 \quad \Rightarrow \quad d_M = (\sqrt{2} - 1)(\sqrt{2} + d_{M-1})$$

Notice that the middle equation implies that all distribution centers are located equidistant. Furthermore, solving this equation system for d_1 yields

$$d_1 = \frac{2 - \sqrt{2}}{6 - 4\sqrt{2} + 2M(\sqrt{2} - 1)}.$$

□

Remark. The locations of the distribution centers for $M = 1, \dots, 5$ are

M	Distribution center locations $d_1, \dots, d_M$				
1	0.5				
2	0.2929	0.7071			
3	0.2071	0.5	0.7929		
4	0.1602	0.3867	0.6133	0.8398	
5	0.1306	0.3153	0.5	0.6847	0.8694

Remark. With uniform probability distribution, both the idling vehicle locations and the distribution center locations will be equidistantly spaced, but note that they have different boundary conditions.

Discrete Distributions

In the road network depicted in Figure 3.11 there are a discrete number C of cities $\mathcal{C} = \{c_1, \dots, c_C\}$ located along the road. We now assume that both the pick-up and drop-off locations are limited to this set of cities, thus the probability density function can be written as

$$\rho(l_1, l_2) = \sum_{u \in \mathcal{C}} \sum_{v \in \mathcal{C}} p_{u,v}\, \delta(u - l_1)\delta(v - l_2),$$

where $\delta(\cdot)$ is Dirac's delta function, u and v are city positions, and $p_{u,v}$ is the probability mass function for an assignment to be from city u to city v. The optimal positioning of the distribution centers can be written as

$$\begin{aligned}\min_{d_1,\dots,d_M} \mathbb{E}_{l_1,l_2} &\left[\min_{i=1,\dots,M} \left(|d_i - l_1| + |d_i - l_2|\right)\right] \\ &= \min_{d_1,\dots,d_M} \sum_{u \in \mathcal{C}} \sum_{v \in \mathcal{C}} \left[p_{u,v} \min_{i=1,\dots,M} \left(|d_i - u| + |d_i - v|\right)\right].\end{aligned}$$

Proposition 3.5. *The distribution centers can optimally be built at a subset of the cities, i.e., only locations $d_1, \dots, d_M \in \mathcal{C}$ need to be considered.*

Proof. Assume without loss of generality that $c_1 \le c_2 \le \cdots \le c_C$, and further assume that $\tilde{d}_1, \dots, \tilde{d}_M$ is an optimal solution with $\tilde{d}_i \in (c_k, c_{k+1})$ located between two cities, for some i and k. Let $\mathcal{D} \subseteq \mathcal{C} \times \mathcal{C}$ denote the set of assignments using the distribution center $\tilde{d}_i$, i.e., $(u, v) \in \mathcal{D}$ if $i = \arg\min_{j=1,\dots,M}(|\tilde{d}_j - u| + |\tilde{d}_j - v|)$.

Consider now if the set of assignments $\mathcal{D}$ instead was handled by a distribution center located at c_k. Since $\tilde{d}_i$ is optimal, we know that

$$\sum_{(u,v) \in \mathcal{D}} p_{u,v}(|\tilde{d}_i - u| + |\tilde{d}_i - v|) \le \sum_{(u,v) \in \mathcal{D}} p_{u,v}(|c_k - u| + |c_k - v|).$$

Notice that if $u \leq \tilde{d}_i$ then also $u \leq c_k$, and if $u \geq \tilde{d}_i$ then $u \geq c_k$, since $u \in \mathcal{C}$, and similarly for v. Thus

$$\sum_{\substack{(u,v)\in\mathcal{D}\\ u\leq\tilde{d}_i\leq v\\ \text{or}\\ v\leq\tilde{d}_i\leq u}} p_{u,v}(|\tilde{d}_i - u| + |\tilde{d}_i - v|) = \sum_{\substack{(u,v)\in\mathcal{D}\\ u\leq\tilde{d}_i\leq v\\ \text{or}\\ v\leq\tilde{d}_i\leq u}} p_{u,v}(|c_k - u| + |c_k - v|),$$

so the inequality only needs to consider when $u, v \leq \tilde{d}_i$ or $u, v \geq \tilde{d}_i$. Expanding the left hand side yields

$$\begin{aligned}
&\sum_{\substack{(u,v)\in\mathcal{D}\\ u,v\leq\tilde{d}_i}} p_{u,v}(|\tilde{d}_i - u| + |\tilde{d}_i - v|) + \sum_{\substack{(u,v)\in\mathcal{D}\\ u,v\geq\tilde{d}_i}} p_{u,v}(|\tilde{d}_i - u| + |\tilde{d}_i - v|)\\
&\qquad = \sum_{\substack{(u,v)\in\mathcal{D}\\ u,v\leq\tilde{d}_i}} p_{u,v}(|c_k - u| + |c_k - \tilde{d}_i| + |c_k - v| + |c_k - \tilde{d}_i|)\\
&\qquad\quad + \sum_{\substack{(u,v)\in\mathcal{D}\\ u,v\geq\tilde{d}_i}} p_{u,v}(|c_k - u| - |c_k - \tilde{d}_i| + |c_k - v| - |c_k - \tilde{d}_i|)\\
&\qquad \leq \sum_{\substack{(u,v)\in\mathcal{D}\\ u,v\leq\tilde{d}_i}} p_{u,v}(|c_k - u| + |c_k - v|) + \sum_{\substack{(u,v)\in\mathcal{D}\\ u,v\geq\tilde{d}_i}} p_{u,v}(|c_k - u| + |c_k - v|).
\end{aligned}$$

Simplifying this inequality, we have

$$\sum_{\substack{(u,v)\in\mathcal{D}\\ u,v\leq\tilde{d}_i}} p_{u,v} \leq \sum_{\substack{(u,v)\in\mathcal{D}\\ u,v\geq\tilde{d}_i}} p_{u,v}.$$

Repeating this argument with c_{k+1} instead of c_k yields

$$\sum_{\substack{(u,v)\in\mathcal{D}\\ u,v\leq\tilde{d}_i}} p_{u,v} \geq \sum_{\substack{(u,v)\in\mathcal{D}\\ u,v\geq\tilde{d}_i}} p_{u,v}.$$

Together, this means that the original inequality is satisfied with equality, and hence that the location $\tilde{d}_i$ can be moved to either c_k or c_{k+1} without changing the value of the optimization problem. □

The locations of the distribution centers are thus given by the optimization problem

$$\min_{d_1,\ldots,d_M\in\mathcal{C}} \sum_{u\in\mathcal{C}} \sum_{v\in\mathcal{C}} \left[p_{u,v} \min_{i=1,\ldots,M} (|d_i - u| + |d_i - v|) \right].$$

Remark. It is clear that having $M > C$ distribution centers will not reduce the transportation cost, since when $M = C$, a distribution center could be built at every city.

Solving this optimization problem by brute force would consider all $\binom{C}{M}$ subsets of the cities, which grows exponentially. Instead, we propose a dynamic programming algorithm for solving this optimization problem in $\mathcal{O}\left(C^4\right)$ complexity. For notational simplicity, $c \in \mathcal{C}$ can denote either the position of city c or its index, as should be clear from the context. The key idea is to let cost$[m][k]$ denote the expected cost of transporting all assignments with $l_1 \le k$ or $l_2 \le k$, using at most m distribution centers, where the last distribution center is located at city k, i.e.,

$$\mathrm{cost}[m][k] = \min_{\substack{d_1,\ldots,d_m \in \mathcal{C} \\ d_1 \le \cdots \le d_m = k}} \sum_{\substack{u,v \in \mathcal{C} \\ u \le k \\ \text{or} \\ v \le k}} \left[p_{u,v} \min_{i=1,\ldots,m} \left(|d_i - u| + |d_i - v| \right) \right].$$

Algorithm 3.1 Optimal Cities for Distribution Centers

```
 1: for $i \in \mathcal{C}$ do                                    ▷ Pre-computations
 2:     for $j \in \mathcal{C}$, $j \ge i$ do
 3:         $a[i][j] \leftarrow \sum_{\substack{u,v \in \mathcal{C} \\ i<u,v \\ u \text{ or } v \le j}} p_{u,v} \min(|i-u| + |i-v|,\ |j-u| + |j-v|)$
 4:     end for
 5:     $b[i] \leftarrow \sum_{\substack{u,v \in \mathcal{C} \\ i<u,v}} p_{u,v}(|u-i| + |v-i|)$
 6: end for
 7: for all $k > 0$ do                                          ▷ Initialize
 8:     cost$[0][k] \leftarrow \infty$
 9: end for
10: for all $m \ge 0$ do                                        ▷ Initialize
11:     cost$[m][0] \leftarrow 0$
12: end for
13: for $m = 1$ to $M$ do
14:     for $k = m$ to $C$ do
15:         cost$[m][k] \leftarrow \min_{i=0,\ldots,k} (\text{cost}[m-1][i] + a[i][k])$
16:     end for
17: end for
18: cost$^\star \leftarrow \min_{k=M,\ldots,C} (\text{cost}[M][k] + b[k])$
```

Algorithm 3.1 produces the optimal cost* of the solution. The optimal locations can be extracted by also memorizing which location minimizes the expression in the inner loop.

3.7 Time-Space Diagrams

In this section, we introduce the time-space diagrams, a useful tool for understanding dynamical transportation models. The time-space diagram shows the (space)-location of vehicles and assignments as they evolve over time, see Figure 3.14.

Let us now consider a dynamic transportation model, where transportation assignments arrive following a stochastic process. The vehicles move with a constant speed along the road, and the objective is to minimize the average time until pick-up of the next transport assignment. However, the vehicles do not have a-priori knowledge about the next assignment, thus they would have to choose idling strategies to minimize the expected pick-up time.

Note that it is easy to illustrate the reachable region for a vehicle in the time-space diagrams, as shown in Figure 3.15. This yields a simple interpretation of the optimal idling locations from equation (3.3), where the optimal idling locations are minimizing the maximal distance to any unreachable point, see Figure 3.16.

Based on this, we propose a dynamic strategy for the idling vehicles. Notice that once an assignment has been picked up, that vehicle cannot be used for any other assignment, hence its reachable region is moved to the assignment's destination, as shown in Figure 3.17. Thus, the remaining vehicles determine their idling locations as to minimize the maximal distance to any unreachable point, using the prior knowledge about the occupied vehicles' reachable regions. Once a new assignment arrives, we assume that the non-occupied vehicle with the shortest pick-up time will be selected for the next assignment, and becomes unavailable until that assignment is completed.

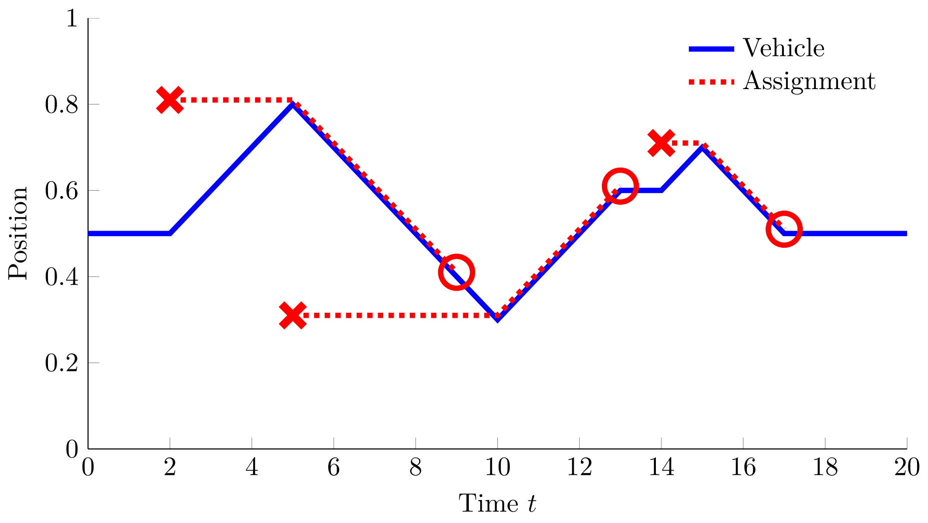


Figure 3.14: A time-space diagram, showing a single vehicle handling three transportation assignments. The red cross denotes the arrival of an assignment, and the red circle the delivery.

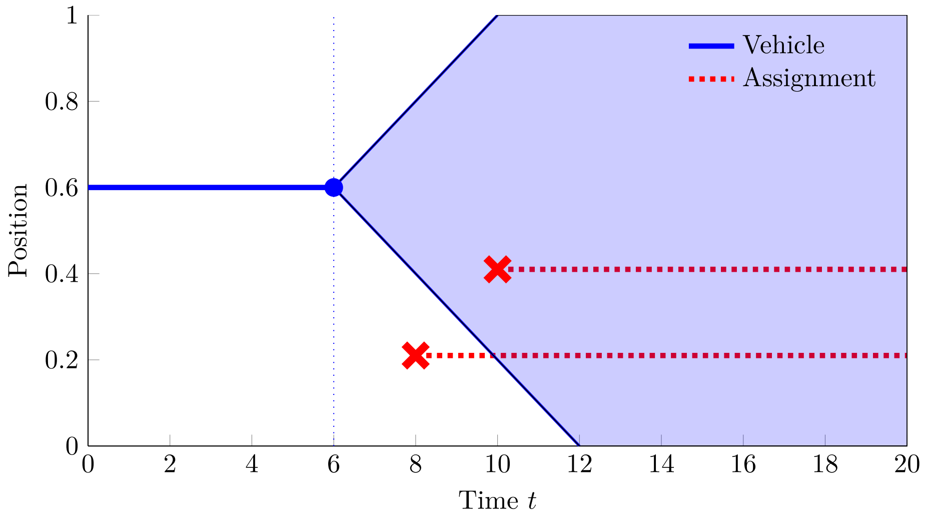


Figure 3.15: A time-space diagram, showing a single vehicle and two assignments. At the current time $t = 6$, the shaded area corresponds to the vehicle's reachable time-space. Thus, it is impossible for the vehicle to pick up the first assignment on time, but the second assignment is still reachable.

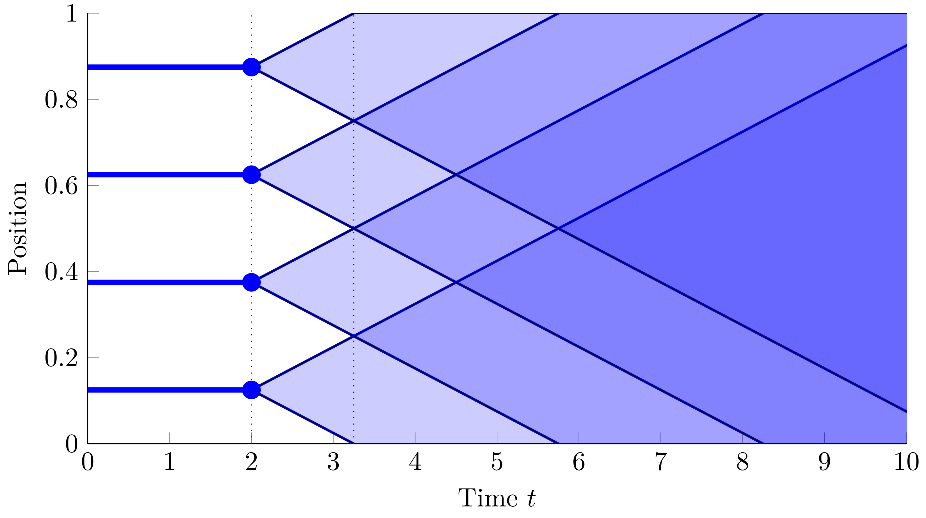


Figure 3.16: A time-space diagram, showing the reachable time-space for four vehicles at time $t = 2$. Note that every position in time-space is currently reachable after $t > 3.25$.

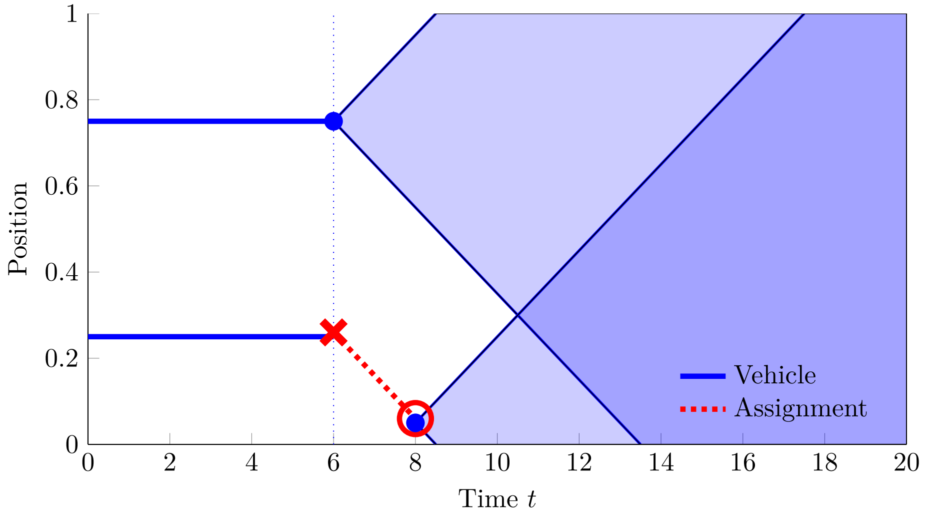


Figure 3.17: A time-space diagram, showing two vehicles and one assignment. Note that once a vehicle is selected for an assignment, its reachable time-space is moved to the assignment's destination.

3.8 Simulation Study

In this section, we exploit the previous optimal solutions in a dynamical transportation model with numerical simulations. Recall that the vehicles operate according to the flow chart in Figure 3.12, and that the total time to handle an assignment consists of the time it takes to pick up the goods and the time it takes to deliver the goods to the destination, including visiting a distribution center. We thus simulate these steps independently in the following subsections.

Idling Vehicles with Uniform Distribution

First, we consider transportation assignments arriving following a Poisson process with rate λ, i.e., the mean time between assignments is $1/\lambda$, and where the pick-up and drop-off locations are chosen uniformly over the interval $[0, 1]$. The transportation assignments are served by $N = 5$ vehicles moving with a unit speed along the road, and the objective is to minimize the average time it takes to pick up each new transport assignment. Notice that we focus on the waiting time, and ignore the fuel cost of transporting the empty vehicles in this section.

Each time a new assignment arrives, all non-occupied vehicles will be considered and the vehicle with the shortest pick-up time will be selected for the assignment. The vehicle then becomes unavailable until it has completed the transport assignment. In Section 3.5 we computed the optimal locations for the idling vehicles to be equidistantly spread out over the road system, and in Section 3.7 we extended this to incorporate information about the vehicles that are about to deliver their assignment. We now exploit this solution as a control law for the unassigned vehicles, where they immediately start to redistribute themselves according to the optimal locations. For example, when one out of five vehicles is selected for a long transport assignment, the remaining four available vehicles will drive towards the locations 0.125, 0.375, 0.625 and 0.875.

We compare this strategy to the base scenario, where the vehicles simply stay where they are after completing an assignment, waiting for a new assignment. The two methods are evaluated using Monte Carlo simulations for different arrival rates λ, and for each arrival rate, the average waiting time is computed for 200 000 random assignments. The results are shown in Figures 3.18 and 3.19. Notice that by exploiting the optimal vehicle location strategy, we are able to reduce the average waiting time by almost 50 % at up to moderate arrival rates λ. As seen in Figure 3.18, when the arrival rate λ approaches 7 assignments per time unit, 5 vehicles will not be sufficient to handle all assignments, which means that the waiting time starts to diverge.

Distribution Center Location with Discrete Distribution

Consider now the transportation stage between the pick-up location and the drop-off location, which is affected by the locations of distribution centers. We use the

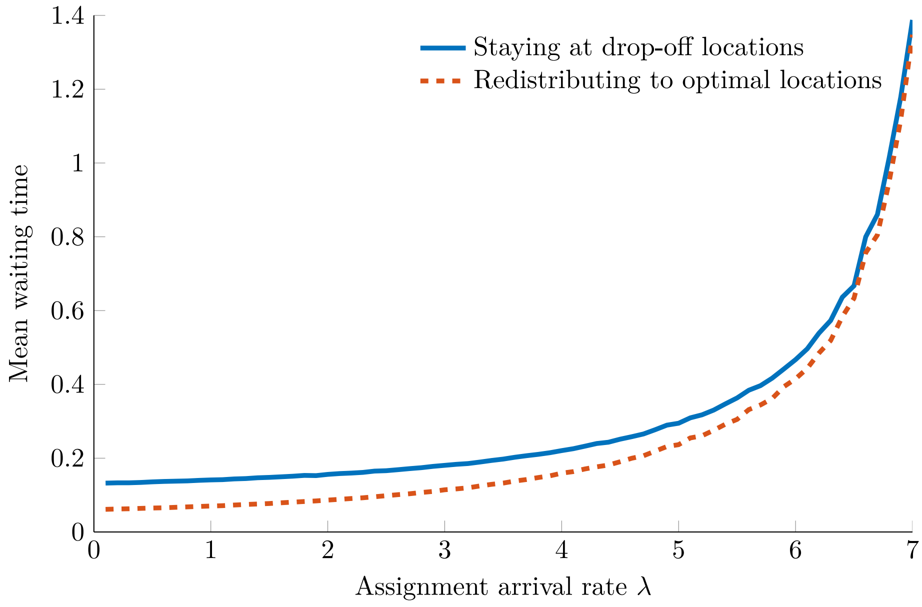


Figure 3.18: The average waiting time to pick up transportation assignments arriving following a Poisson process with rate λ using 5 vehicles. Two different strategies are compared, either the vehicles stay at their drop-off location until the next assignment, or they redistribute according to the optimal locations.

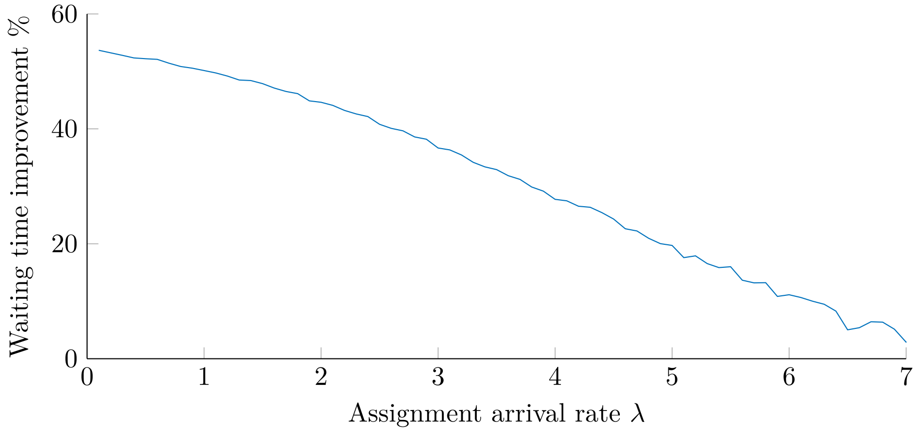


Figure 3.19: The improvement in the average waiting time by redistributing the vehicles towards their optimal locations, compared to staying at the drop-off location, as shown in Figure 3.18.

City name	Population	Distance	Relative position
Stockholm	923 516	0 km	0.00
Södertälje	93 202	34 km	0.07
Nyköping	54 262	101 km	0.21
Norrköping	137 035	160 km	0.34
Linköping	152 966	198 km	0.42
Jönköping	133 310	322 km	0.68
Borås	108 488	406 km	0.86
Göteborg	548 190	470 km	1.00

Table 3.2: Major cities along the Swedish highway in Figure 3.11. Population data provided by SCB [2015]. Distance given as the road distance measured from Stockholm.

cities for the Swedish main highway, shown in Figure 3.11, as a discrete distribution for the assignment locations. Along this road there are 8 major cities, see Table 3.2, and the transport assignment location probabilities $p_{u,v}$ are selected proportional to the population of the cities. The population mass function is shown in Figure 3.20.

The optimal distribution center locations are computed for each $M = 1, \ldots, 8$ number of distribution centers, and the resulting cities are indicated in Figure 3.21. The range of mean traveling times is shown in Figure 3.22, where the lower bound corresponds to the optimal and selected distribution centers in Figure 3.21. As shown, the locations of the distribution centers can significantly affect the assignment transportation time.

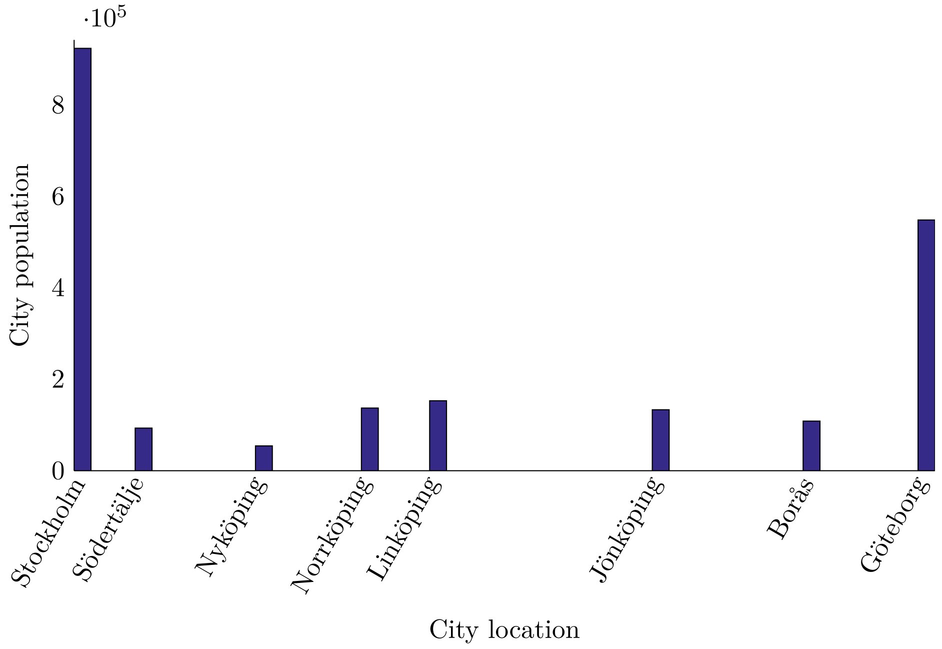


Figure 3.20: The city population with their relative position on the Swedish highway in Figure 3.11.

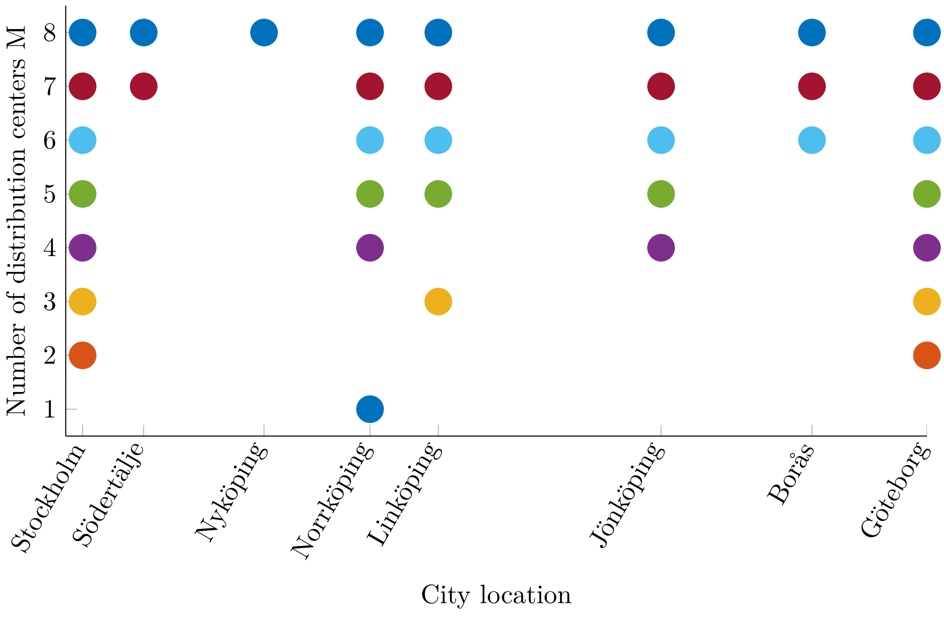


Figure 3.21: The optimal cities for distribution center locations, depending on the number of distribution centers. The markers denote where the distribution centers should be located.

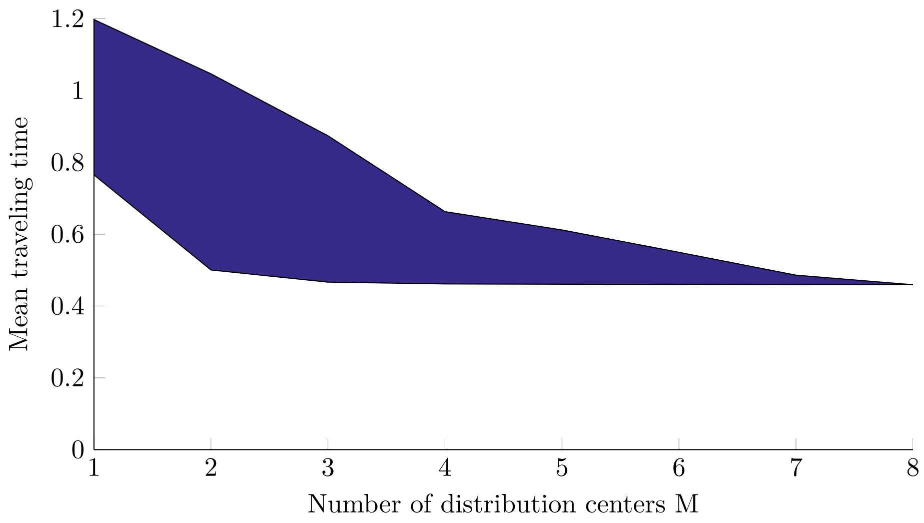


Figure 3.22: The range of mean traveling times for all possible choices of distribution centers. The lower bound corresponds to the optimal choice of distribution centers, shown in Figure 3.21.

3.9 Conclusions

In this chapter, we developed a new efficiency measure η for transportation systems, where the trips are divided into actual transportation assignments and vacant trips. The efficiency measure is especially useful when evaluating a transportation system based on collected GPS trajectories, and we showed that the efficiency measure can easily be computed, even for huge datasets.

We used the efficiency measure to evaluate a road freight transportation scenario, with multiple competitive transportation companies. We showed that even if the companies had optimized their own transportation routes, they only achieved 59 % efficiency when assignments where allocated geographically, and 92 % when randomizing assignments, compared to the fully cooperative scenario. We also demonstrated the efficiency measure on a real data set from New York City's taxis, where we showed that the total mileage could be reduced by between 9 % and 20 %, depending on the time of day and flexibility in the system.

We then considered a transportation system along a major transportation route. The goal of a real-time transport service provider is to minimize the time from the reception of a transport assignment until the delivery. This resulted in two separate problems: a strategy for distributing idling vehicles, and a strategy for locating distribution centers.

We formulated these problems as stochastic optimization problems, and provided explicit solutions for uniform distributions, as well as an efficient algorithm for discrete probability distributions. The methods were evaluated with numerical simulations from a Swedish highway.

This work shows that there is a tremendous potential for improving the efficiency of our current transportation system by improved planning and coordination among different actors.

Chapter 4

Estimation in Anonymous Networks

"Arguing that you don't care about the right to privacy because you have nothing to hide is no different than saying you don't care about free speech because you have nothing to say."

— Edward Snowden

In this chapter we consider distributed estimation of data obtained by the nodes in a network $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. Two specific problems are analyzed: first to estimate the size of the network $N = |\mathcal{V}|$, and later to estimate the empirical distribution of local measurements z_i generated by each node $i \in \mathcal{V}$. We explicitly target dynamical networks $\mathcal{G}(t)$ by utilizing a regularization term which captures a-priori assumptions on the dynamic network evolution.

Our aim is to obtain distributed algorithms, where all nodes execute the same algorithm in parallel, and where neither leaders nor an overlay structure is present. We assume that the nodes have no knowledge of the network topology, and that they have narrowly bounded computational, memory and bandwidth resources, where especially the size of the exchanged information packets stays constant over time. Finally, the goal is that all agents should quickly reach consensus, in the sense that they should share the same estimates of the global properties for the network as fast as possible.

We restrict our methods to anonymous networks, where the uniqueness of the node identifiers is not guaranteed [Yamashita and Kameda, 1988], thus avoiding the possibility of tracing or characterizing a single agent. The anonymity is motivated for maintaining users' privacy (e.g., in P2P networks where users may not want to disclose information about their identity), but is also beneficial in applications when the estimation strategies must be simple with limited resource requirements.

The outline of this chapter is as follows: In Section 4.1 we introduce the network size estimation problem for dynamical networks, which we analyze in Section 4.2. We continue by specifically considering quadratic regularization terms in Section 4.3, and evaluate the estimator in Section 4.4 with numerical experiments.

Next, we turn to the PMF estimation problem for static networks in Section 4.5, and introduce two different estimators for this problem in Section 4.6. These estimators are analyzed in Section 4.7, and evaluated with numerical simulations in Section 4.8.

Finally, we combine the dynamical network size estimator from Section 4.1 with the PMF estimator for static networks from Section 4.6 to estimate PMFs in dynamical networks in Section 4.9. In Section 4.10 we conclude this chapter.

4.1 Size Estimation Problem

The first problem we consider is to estimate the network size $N(t) = |\mathcal{V}(t)|$ of a time-dependent network $\mathcal{G}(t) = (\mathcal{V}(t), \mathcal{E}(t))$. In many distributed network applications, the network manager will need to redirect resources or take other restorative actions if the network topology changes. Thus, being able to estimate the size and, in particular, changes in the network size is indispensable for automatic network reconfiguration and fault detection.

The considered network model $\mathcal{G}(t)$ of interconnected agents $\mathcal{V}(t) = \{1, \ldots, N(t)\}$ is based on agents which can join or leave at any time. The goal is to distributively track the network size, i.e., each agent should create an estimate of the number of agents as the network size is evolving.

We restrict this problem to the class of anonymous networks, i.e., where the agents do not have unique identifiers, following the work by Angluin [1980]. In this framework, it has been proven that there cannot exist deterministic algorithms which are guaranteed to compute the correct network size, see Cidon and Shavitt [1995], but by exploiting probabilistic methods, the entire network topology can be reconstructed, see Codenotti et al. [1997]. However, these methods are based on exchanging the local views (each agent has local enumeration of its neighbors), and fail when there are communication and memory limitations.

In for example wireless sensor networks, the communication, computation and memory capacity are strictly limited resources, thus we further restrict the algorithms to broadcast communication with a fixed message size, and all agents running identical algorithms. We formalize this by assuming that each agent $i \in \mathcal{V}$ has a local variable $x_i(t)$ that can be modified at time $t+1$ by accessing the states $x_j(t)$ of the neighboring nodes $j \in \mathcal{N}_i(t)$, and performing the aggregation operation

$$x_i(t+1) = f\left(x_i(t),\, x_{j_1}(t),\, x_{j_2}(t),\, \ldots\right), \quad j_1,\, j_2,\, \ldots \in \mathcal{N}_i(t)$$

that preserves the dimension of $x_i(t)$, for some function f. Each agent then computes a local estimate of the network size from the local variable $x_i(t)$,

$$\widehat{N}(t) = J\left(x_i(t)\right)$$

for an appropriate estimation function J. Thus, we do not even assume that a local view of the network exist, and in particular allow for time-varying topologies.

Remark. In the following section, we assume $x_i(t) \in \mathbb{R}^M$ for the analysis of the estimators statistical properties, but in Section 4.4 we perform numerical evaluations of the estimator using b bits when representing each real value, thus limiting the communication message size.

Let us introduce the following notation, where $N(t)$ represents the true number of agents in the network at time $t \in \mathbb{N}$, while $\widehat{N}(t)$ denotes the estimated value of $N(t)$. In the maximum likelihood (ML) estimator, we denote a generic hypothesis by $\overline{N}(t)$ for the estimated value of $N(t)$. The estimator will simultaneously estimate the network size for a time window of length $\tau + 1$, and we also utilize previous estimates up until time $t - \eta$, where $\eta \geq \tau$, in the regularization term. We therefore introduce the following vectorized versions of the previous quantities, where the bold italics indicate vectors:

$$\boldsymbol{N}(t) \doteq [N(t), \ldots, N(t-\tau)]^T \tag{4.1}$$

$$\overline{\boldsymbol{N}}(t) \doteq \left[\overline{N}(t), \ldots, \overline{N}(t-\tau)\right]^T \tag{4.2}$$

$$\widehat{\boldsymbol{N}}(t) \doteq [\widehat{N}(t), \ldots, \widehat{N}(t-\tau)]^T \tag{4.3}$$

$$\widehat{\boldsymbol{N}}_\tau^\eta(t) \doteq [\widehat{N}(t-\tau-1), \ldots, \widehat{N}(t-\eta)]^T. \tag{4.4}$$

Thus, $\boldsymbol{N}(t)$ refers to the true values over a time window of length $\tau + 1$, $\overline{\boldsymbol{N}}(t)$ refers to a generic hypothesis on the true value of $\boldsymbol{N}(t)$, and $\widehat{\boldsymbol{N}}(t)$ refers to the estimate of the true values. $\widehat{\boldsymbol{N}}_\tau^\eta(t)$ contains an additional memory of previous estimates that is used to improve the regularization process of the estimate. Notice that $\tau, \eta \in \mathbb{N}$ are fixed design parameters of the algorithm.

4.2 Network Size Estimation Algorithm

We will here propose a distributed size estimation algorithm for anonymous networks, using probabilistic initialization of the state vector. The basic idea behind the network size estimation scheme, introduced by Varagnolo et al. [2010], is that each agent $i \in \mathcal{V}(t)$ generates a uniform, random sample $x_i \sim \mathcal{U}[0,1]$, and then the max consensus protocol, described in Section 2.5, is used to compute the maximum $f = \max_{i \in \mathcal{V}}(x_i)$ of these samples. This yields a sample of a random variable, whose distribution is the maximum of $N(t)$ independent and identically distributed (i.i.d.) random variables, which depends upon $N(t)$. Hence, this computed sample can be used for an ML estimate of $N(t)$.

Compared to the previous literature, we derive a distributed estimator that extends techniques based on order statistics with a regularization approach [Vapnik, 1998, Wahba, 1990]. We introduce a regularization term that allows the designer to combine the empirical evidence from the data with a-priori beliefs on the expected behavior of the network size to be estimated, and then provide an analysis of quadratic regularization functions.

The specific extension of this idea to dynamic network size estimation includes a repetitive generation of new random samples and computation of the maximal value. The samples from the max distribution is kept for a time window of length $\tau+1$, to simultaneously estimate $\boldsymbol{N}(t)$ with the a-priori assumptions on the evolution. Our algorithm also includes an additional memory of the previous estimates $\widehat{\boldsymbol{N}}_\tau^\eta(t)$ for an extended time window of length $\eta-\tau$, which are considered as fixed parameters in the estimation scheme. All these data are then used to compute a penalized ML estimate $\widehat{\boldsymbol{N}}(t)$, as described in Algorithm 4.1 and equation (4.6).

Algorithm 4.1 Dynamic Network Size Estimation Algorithm

```
1: for every $t = 1, 2, \dots$ do
2:     (Generation step) Each agent $i = 1, \dots, N(t)$ generates $M$ i.i.d. random values
           $x_{i,m}(t) \sim \mathcal{U}[0,1], \quad m = 1, \dots, M.$
3:     (Communication step) Agents compute, through max consensus strategies, the $M$-dimensional max vector
           $\boldsymbol{f}(t) \doteq [f_1(t), \dots, f_M(t)]^T,$
       where
           $f_m(t) = \max_{i=1,\dots,N(t)} x_{i,m}(t).$
4:     (Computation step) Each agent estimates the total number of agents in the network through the penalized ML scheme as
           $\widehat{\boldsymbol{N}}(t) = \arg\min_{\overline{\boldsymbol{N}} \in \mathbb{R}^{\tau+1}} J\Big(\overline{\boldsymbol{N}}\ ;\ \boldsymbol{f}(t), \dots, \boldsymbol{f}(t-\tau), \widehat{\boldsymbol{N}}_\tau^\eta(t)\Big).$   (4.5)
5: end for
```

Remark. The time index t does not need to denote physical quantities (such as seconds), but rather *epochs*, under which each iteration of Algorithm 4.1 is completed, and the network size is estimated. Hence, we have assumed a network synchronization to the epochs, even if the max-consensus communication protocol can be asynchronous during each epoch. We implicitly also assume that the agents always reach the max consensus on the locally generated samples within each epoch. For the max consensus to succeed, the network needs to be strongly connected during each epoch, otherwise the size estimation algorithm extends to each agent estimating the number of agents who are able to influence it during the epoch.

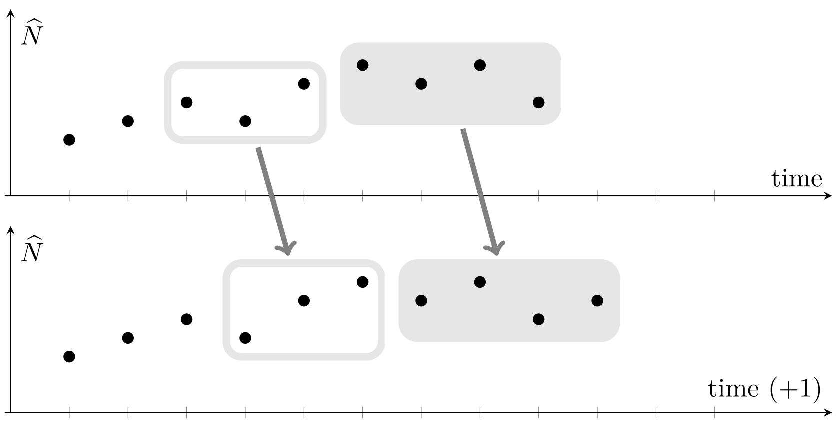


Figure 4.1: Example of the time behavior of the estimation scheme of equation (4.6). The white rectangle indicates the extra parameters $\widehat{\boldsymbol{N}}_{\tau}^{\eta}(t)$, while the gray rectangle indicates the time-window where the optimization problem of equation (4.5) acts to obtain new estimates.

The penalized log-likelihood function J in equation (4.5) is defined as:

$$J\Big(\overline{\boldsymbol{N}}\ ;\ \boldsymbol{f}(t),\ \ldots,\ \boldsymbol{f}(t-\tau), \widehat{\boldsymbol{N}}_{\tau}^{\eta}(t)\Big) \doteq \\ -\log\left(p\left(\boldsymbol{f}(t),\ \ldots,\ \boldsymbol{f}(t-\tau)\ ;\ \overline{\boldsymbol{N}}\right)\right) + \gamma\mathcal{R}\left(\overline{\boldsymbol{N}}, \widehat{\boldsymbol{N}}_{\tau}^{\eta}(t)\right). \quad (4.6)$$

This allows us to estimate the network size $\boldsymbol{N}(t)$ while penalizing hypotheses $\overline{\boldsymbol{N}}$ that deviate from expected behaviors by means of the regularization term $\mathcal{R}$: $\mathbb{R}^{\tau+1} \times \mathbb{R}^{\eta-\tau} \to \mathbb{R}_{+}$. Thus, given a hypothesis $\overline{\boldsymbol{N}}$, equation (4.6) evaluates both its plausibility in the regularization term and its empirical evidence in the log-likelihood function [Schölkopf and Smola, 2002, Chap. 4]. The parameter γ in equation (4.6) is called the *regularization parameter*, and can be tuned to capture the trade-off between the empirical evidence of $\overline{\boldsymbol{N}}$ and its plausibility.

Notice that the hypothesis $\overline{\boldsymbol{N}}$ corresponds to a time-window of length $\tau + 1$, while the regularization term $\mathcal{R}$ explicitly depends on the memory of the past estimates $\widehat{\boldsymbol{N}}_{\tau}^{\eta}(t)$ up to time $t - \eta$ ($\eta \geq \tau$), defined in equation (4.4). The past estimates $\widehat{\boldsymbol{N}}_{\tau}^{\eta}(t)$ are not changed by the estimator, and are used as fixed extra parameters. An illustrative description of how these time windows shift in time is given in Figure 4.1.

Remark. If the regularization term is removed, $\mathcal{R} = 0$, then Algorithm 4.1 is reduced to sequentially computing the estimates as

$$\begin{aligned}\widehat{N}(t) &= \arg\min_{\overline{N}\in\mathbb{R}} \Big(-\log\left(p\left(\boldsymbol{f}(t)\ ;\ \overline{N}\right)\right) \Big) \\ &= -\left(\frac{1}{M}\sum_{i=1}^{M}\log\left(f_i\right)\right)^{-1}. \end{aligned} \tag{4.7}$$

In this case, the various $\widehat{N}(t)$'s are estimated independently at each time t, and this corresponds to the ML approach used for static anonymous networks [Varagnolo et al., 2010]. The accuracy is clearly improved by increasing the number of random samples M, as can be seen from the statistical properties of $\widehat{N}$, since in this case $\widehat{N}$ is inverse-gamma distributed, i.e., $\widehat{N} \sim \text{I-}\Gamma\left(M, NM\right)$ (assuming $M > 2$):

$$\mathbb{E}\left[\frac{\widehat{N}(t)}{N(t)}\ ;\ M\right] = \frac{M}{M-1},$$

$$\mathbb{E}\left[\left(\frac{N(t)-\widehat{N}(t)}{N(t)}\right)^2\ ;\ M\right] = \frac{M+2}{(M-1)(M-2)}.$$

Parameter Design Constraints

For the dynamical network size estimation, the estimation accuracy is intuitively non-decreasing in M, τ and η. However, the number of samples M is bounded by transmission costs (in the max consensus step), τ is bounded by computational constraints (in the size of the optimization problem, equation (4.5)), while η is bounded by memory limitations and modeling accuracy of the regularization term.

To minimize the memory impact, the following proposition shows that the vectors $\boldsymbol{f}(t)$, $\boldsymbol{f}(t-1)$, … can be compressed into scalars without loss of information.

Proposition 4.1. *Let $s(\tau) \doteq -\sum_{m=1}^{M}\log\left(f_m(\tau)\right)$. Then $s(\tau)$ is a complete and minimal sufficient statistic for $N(\tau)$.*

Proof. Since the samples $x_{i,m}(\tau)$ are i.i.d., it follows that $p\left(\boldsymbol{f}(t),\ldots,\boldsymbol{f}(1)\ ;\ \overline{\boldsymbol{N}}\right) = \prod_{\tau=1}^{t} p\left(\boldsymbol{f}(\tau)\ ;\ \overline{N}(\tau)\right)$. To prove the proposition it is then sufficient to show that $s(\tau)$ is a complete and minimal sufficient statistic for $N(\tau)$.

Let us start by showing that $s(\tau)$ is a sufficient statistic. Consider the probability density of $\boldsymbol{f}(\tau)$ given $\overline{N}(\tau)$,

$$p\left(\boldsymbol{f}(\tau)\ ;\ \overline{N}(\tau)\right) = \prod_{m=1}^{M} \overline{N}(\tau)\cdot f_m(\tau)^{\overline{N}(\tau)-1} = \overline{N}(\tau)^M e^{-\left(\overline{N}(\tau)-1\right)s(\tau)},$$

for all τ, thus, $s(\tau)$ is a sufficient statistic for $N(\tau)$ because of the Fisher-Neyman Factorization Theorem [Zacks, 1971]. It is also clearly minimal since it is a scalar.

To show the completeness of $s(\tau)$, we must show that if $g(s(\tau))$ is a generic measurable function such that $\mathbb{E}\left[g(s(\tau)) \mid N\right] = 0$ independently of N, then it must hold that $g(\cdot) = 0$ almost everywhere (a.e.). Consider now that $-\log\left(f_i(\tau)\right)$ is an exponential random variable with rate N. Thus, $s(\tau)$ is the sum of i.i.d. exponential random variables, i.e., $s(\tau) \sim \Gamma\left(M, \frac{1}{N}\right)$. $\mathbb{E}\left[g(s(\tau)) \mid N\right] = 0$ can then be rewritten as

$$\Gamma\left(M\right)^{-1} N^M \int_0^{+\infty} g(s) s^{M-1} \exp\left(-sN\right) \, \mathrm{d}s \equiv 0 \, .$$

This is equivalent to the fact that the Laplace transform of $g(s)s^{M-1}$ has to be zero a.e., and this happens if and only if $g(s)$ is zero a.e. □

This compression of variables actually results in a memory saving of τM scalars, and only a single vector of M scalars is needed during the max consensus step for computing the current $\boldsymbol{f}(t)$.

By introducing $\boldsymbol{s}(t) \doteq \left[s(t), \ldots, s(t-\tau)\right]^T$, the penalized likelihood in equation (4.6) can be rewritten as

$$J\Big(\overline{\boldsymbol{N}} \, ; \, \boldsymbol{s}(t), \, \widehat{\boldsymbol{N}}_\tau^\eta(t)\Big) = -\log\left(p\left(\boldsymbol{s}(t) \, ; \, \overline{\boldsymbol{N}}\right)\right) + \gamma \mathcal{R}\left(\overline{\boldsymbol{N}}, \, \widehat{\boldsymbol{N}}_\tau^\eta(t)\right).$$

Quadratic Regularization

Adding a regularization term $\mathcal{R}$ in empirical risk minimization problems, as we did in equation (4.6), generally improves their conditioning properties [Schölkopf and Smola, 2002, Chap. 4]. The usage of these terms can also be motivated by Bayesian perspectives, where the penalty $\mathcal{R}$ reflects a-priori beliefs on a typical behavior.

Here we explicitly consider quadratic regularization terms, i.e.,

$$\mathcal{R}\left(\overline{\boldsymbol{N}}, \widehat{\boldsymbol{N}}_\tau^\eta\right) = \begin{bmatrix} \overline{\boldsymbol{N}} - \boldsymbol{\mu}_1 \\ \widehat{\boldsymbol{N}}_\tau^\eta - \boldsymbol{\mu}_2 \end{bmatrix}^T \underbrace{\begin{bmatrix} \mathcal{Q}_{11} & \mathcal{Q}_{12} \\ \mathcal{Q}_{12}^T & \mathcal{Q}_{22} \end{bmatrix}}_{Q^{-1}} \begin{bmatrix} \overline{\boldsymbol{N}} - \boldsymbol{\mu}_1 \\ \widehat{\boldsymbol{N}}_\tau^\eta - \boldsymbol{\mu}_2 \end{bmatrix}, \tag{4.8}$$

where $\boldsymbol{\mu} = \mathbb{E}\left[\boldsymbol{N}\right]$ is a nominal behavior of $\boldsymbol{N}$, and Q^{-1} is a symmetric positive definite matrix.

Proposition 4.2. *Given a quadratic regularization term as in equation* (4.8)*, the optimal estimator* $\widehat{\boldsymbol{N}}(t)$ *for equation* (4.5) *satisfies the quadratic equation system*

$$\operatorname{diag}\left(\widehat{\boldsymbol{N}}(t)\right) \cdot \left(\boldsymbol{s}(t) + 2\gamma \mathcal{Q}_{11}\left(\widehat{\boldsymbol{N}}(t) - \boldsymbol{\mu}_1\right) + 2\gamma \mathcal{Q}_{12}\left(\widehat{\boldsymbol{N}}_\tau^\eta(t) - \boldsymbol{\mu}_2\right)\right) - M \cdot \mathbb{1} = \boldsymbol{0}. \tag{4.9}$$

Proof. Since $s(t), \ldots, s(t-\tau)$ are independent, and their probability distribution is

$$p\left(s(\tau)\ ;\ \overline{N}(\tau)\right) = \overline{N}(\tau)^M e^{-(\overline{N}(\tau)-1)s(\tau)},$$

it follows that

$$-\log\left(p\left(s(t), \ldots, s(t-\tau)\ ;\ \overline{\boldsymbol{N}}\right)\right) = \sum_{i=t-\tau}^{t} \left((\overline{N}(i)-1)s(i) - M\log\left(\overline{N}(i)\right)\right).$$

We can thus rewrite the estimator in equation (4.5) as

$$\begin{aligned}\arg\min_{\overline{\boldsymbol{N}}} \sum_{i=t-\tau}^{t} & \left((\overline{N}(i)-1)s(i) - M\log\left(\overline{N}(i)\right)\right) \\ & +\gamma\left(\overline{\boldsymbol{N}}-\boldsymbol{\mu}_1\right)^T \mathcal{Q}_{11}\left(\overline{\boldsymbol{N}}-\boldsymbol{\mu}_1\right) \\ & +2\gamma\left(\overline{\boldsymbol{N}}-\boldsymbol{\mu}_1\right)^T \mathcal{Q}_{12}\left(\widehat{\boldsymbol{N}}_\tau^\eta-\boldsymbol{\mu}_2\right) \\ & +\gamma\left(\widehat{\boldsymbol{N}}_\tau^\eta-\boldsymbol{\mu}_2\right)^T \mathcal{Q}_{22}\left(\widehat{\boldsymbol{N}}_\tau^\eta-\boldsymbol{\mu}_2\right).\end{aligned}$$

Setting the gradient with respect to $\overline{\boldsymbol{N}}$ equal to zero yields, for each $i = t-\tau, \ldots, t$,

$$s(i) - \frac{M}{\overline{N}(i)} + 2\gamma\left(\mathcal{Q}_{11}^{(i)}\left(\overline{\boldsymbol{N}}-\boldsymbol{\mu}_1\right) + \mathcal{Q}_{12}^{(i)}\left(\widehat{\boldsymbol{N}}_\tau^\eta-\boldsymbol{\mu}_2\right)\right) = 0,$$

where $\mathcal{Q}_{11}^{(i)}$ is the i-th row of $\mathcal{Q}_{11}$ (similarly for $\mathcal{Q}_{12}^{(i)}$). Multiplying by $\overline{N}(i)$ and vectorizing the previous equation into a matrix equality leads to equation (4.9). □

Quadratic regularization terms, as in equation (4.8), especially capture the design strategies where $\mathcal{R}$ penalizes just the changes between consecutive estimates $N(t), \ldots, N(t-\eta)$. In fact, by defining $\Omega_{ij} \doteq (\boldsymbol{e}_i - \boldsymbol{e}_j)(\boldsymbol{e}_i - \boldsymbol{e}_j)^T$, where $\{\boldsymbol{e}_i\}$ is the standard basis of $\mathbb{R}^n$, $\boldsymbol{x} = [x_1, \ldots, x_n]^T$, and letting $Q^{-1} = \sum_{i,j} q_{ij}\Omega_{ij}$ with $q_{ij} > 0$ then $\|\boldsymbol{x} - \mu\mathbb{1}\|_Q^2 = \sum_{i,j} q_{ij}(x_i - x_j)^2$. In this case, choices for η larger than $\eta = \tau + 1$ are meaningless, since a larger value would just add a constant to the regularization term.

4.3 Properties under a Markovian P2P Model

We now derive the quadratic regularization term as an approximation of the probabilistic model for a simple but practical network example. This example also illustrates an important extension of the algorithm; that it can be used to count the number of nodes that satisfies any property, as long as the nodes can determine this property themselves.

Consider an anonymous P2P file sharing network, where a certain file is only available at a subset of the peers, and the goal is to estimate how many peers have this file. At any time, a user who does not have the file can choose to download it, and a user who does have the file can choose to delete it. All peers in the network will participate in the estimation algorithm, but only the peers who have the file perform the first step of generating new random values. Those peers who do not have the file would instead initialize their state with zeros in order to not affect the max consensus protocol. We further assume that:

- there exists a boundary on the total number of peers [1], say $N_{\max}$;
- downloading and deleting files happen independently among the peers;
- the stochastic process that peer i downloads or deletes the file is a Markov process with (known) probabilities:

$$\begin{aligned} p &\doteq \mathbb{P}\left[x_i(t) = 1 \mid x_i(t-1) = 0\right] \\ q &\doteq \mathbb{P}\left[x_i(t) = 0 \mid x_i(t-1) = 1\right] \end{aligned} \tag{4.10}$$

  where $x_i(t) = 1$ corresponds to peer i having the file at time t, while $x_i(t) = 0$ corresponds to that peer i does not have the file at time t.

Given these assumptions, we derive an estimator for the current time step ($\tau = 0$), but with two steps of regularization memory ($\eta = 1$).

Derivation of the Regularization Term

Let us consider the Bayesian interpretation of the quadratic regularization term as a log-Gaussian prior on $[N(t), N(t-1)]$. Given the independence assumptions stated above, we need to compute the nominal behavior $\mu \doteq \mathbb{E}\left[N(t)\right]$ and variance

$$Q \doteq \mathbb{E}\left[\begin{bmatrix} N(t) - \mu \\ N(t-1) - \mu \end{bmatrix} \begin{bmatrix} N(t) - \mu \\ N(t-1) - \mu \end{bmatrix}^T\right].$$

Lemma 4.3. *Let $\alpha \doteq {}^{p}/_{q}$ be the radio between the two transition probabilities in the Markov chain. Then,*

[1] As stated later in this section, this assumption is not strictly required and could be removed.

$$\mu \doteq \mathbb{E}\left[N(t)\right] = \frac{\alpha}{1+\alpha} N_{\max}, \tag{4.11}$$

$$\operatorname{var}\left(N(t)\right) = \frac{\alpha}{(1+\alpha)^2} N_{\max}, \quad \textit{and} \tag{4.12}$$

$$\operatorname{cov}\left(N(t), N(t-1)\right) = (1-p-q)\frac{\alpha}{(1+\alpha)^2} N_{\max}. \tag{4.13}$$

Proof. Notice that $N(t) = \sum_{a=1}^{N_{\max}} x_a(t)$, where the processes x_a are i.i.d. Thus, let us first compute the expected value, variance and covariance for a single agent.

The Markov process in equation (4.10) is described by the transition matrix P given by

$$P = \begin{bmatrix} 1-p & p \\ q & 1-q \end{bmatrix}.$$

The equilibrium distribution, $\boldsymbol{\pi} = \boldsymbol{\pi} P$, for the Markov process is $\boldsymbol{\pi} = \frac{1}{1+\alpha}\begin{bmatrix} 1 & \alpha \end{bmatrix}$, thus the expected value is

$$\mathbb{E}\left[x_a(t)\right] = \frac{\alpha}{1+\alpha}.$$

Further, the variance is

$$\begin{aligned} \operatorname{var}\left(x_a(t)\right) &= \mathbb{E}\left[x_a(t)^2\right] - \mathbb{E}\left[x_a(t)\right]^2 \\ &= \frac{\alpha}{1+\alpha} - \left(\frac{\alpha}{1+\alpha}\right)^2 \\ &= \frac{\alpha}{(1+\alpha)^2}. \end{aligned}$$

Finally, for a single agent we have the covariance

$$\begin{aligned} \operatorname{cov}\left(x_a(t), x_a(t-1)\right) &= \mathbb{E}\left[x_a(t)x_a(t-1)\right] - \mathbb{E}\left[x_a(t)\right]\mathbb{E}\left[x_a(t-1)\right] \\ &= \frac{\alpha}{1+\alpha}(1-q) - \left(\frac{\alpha}{1+\alpha}\right)^2 \\ &= (1-p-q)\frac{\alpha}{(1+\alpha)^2}. \end{aligned}$$

For the entire system $N(t) = \sum_{a=1}^{N_{\max}} x_a(t)$ we can simply multiply the results for a single agent by $N_{\max}$, because the different agents are i.i.d., and because of the linearity of the expected value, variance and covariance. □

Thus we have the quadratic regularization term given by

$$Q = N_{\max}\frac{\alpha}{(1+\alpha)^2}\begin{bmatrix} 1 & 1-p-q \\ 1-p-q & 1 \end{bmatrix}. \tag{4.14}$$

Derivation of the Estimator

As we supposed that $\tau = 0$ and $\eta = 1$, our variables correspond to $\widehat{\boldsymbol{N}}(t) = \widehat{N}(t)$, $\widehat{\boldsymbol{N}}_\tau^\eta(t) = \widehat{N}(t-1)$,

$$\begin{aligned}
\boldsymbol{\mu}_1 = \boldsymbol{\mu}_2 &= \mu = \frac{\alpha}{1+\alpha} N_{\max}, \\
\mathcal{Q}_{11} = \mathcal{Q}_{22} &= \frac{1}{\mu q\big(2 - q(1+\alpha)\big)}, \\
\mathcal{Q}_{12} = \mathcal{Q}_{21} &= \frac{q(1+\alpha) - 1}{\mu q\big(2 - q(1+\alpha)\big)}.
\end{aligned}$$

In this case, the condition on the optimal estimator in equation (4.9) simplifies into the quadratic equation

$$a\widehat{N}^2(t) + \left(b\widehat{N}(t-1) + c\right)\widehat{N}(t) - M = 0, \tag{4.15}$$

where

$$\begin{aligned}
a &\doteq 2\gamma \mathcal{Q}_{11}, \\
b &\doteq 2\gamma \mathcal{Q}_{12}, \\
c &\doteq s(t) - 2\gamma\left(\mathcal{Q}_{11} + \mathcal{Q}_{12}\right)\mu.
\end{aligned}$$

The unique admissible solution for $\widehat{N}(t)$ is given by

$$\widehat{N}(t) = \sqrt{\left(\frac{b\widehat{N}(t-1) + c}{2a}\right)^2 + \frac{M}{a}} - \left(\frac{b\widehat{N}(t-1) + c}{2a}\right). \tag{4.16}$$

Remarkably, our penalized ML approach leads to a recursive estimator that is nonlinear but still easy to implement in devices with small computational capabilities. The reason that the obtained smoother is nonlinear is because even though we derived the regularization term using Gaussian assumptions on $[N(t), N(t-1)]$, the likelihood term in equation (4.6) is non-Gaussian. If the likelihood had been Gaussian, then the estimator would have been a linear smoother, leading to a Kalman filter.

Remark. Note that the derivation of Q using Gaussian assumptions implies that $N(t)$, $N(t-1)$ could take negative values. Despite the error of this approximation, the effects vanish as $N_{\max}$ increases since $N(t) = \sum_{a=1}^{N_{\max}} x_a(t)$ is approximatively Gaussian due to the central limit theorem. A formally correct probabilistic interpretation would require the regularization term $\mathcal{R}$ to be derived from the actual prior distribution, but this would lead to a non-quadratic $\mathcal{R}$, and not-closed-form solutions of equation (4.5).

The Role of the Regularization Parameter γ

In equation (4.6), the log-likelihood function $-\log\left(p\left(\boldsymbol{s}(t)\ ;\ \overline{\boldsymbol{N}}\right)\right)$ takes into account the experimental evidence, while the regularization term $\mathcal{R}$ reflects the a-priori information about the regularity of the solution. The regularization parameter γ then captures the trade-off between these two components, and represents how much one trusts the regularity assumptions. Notice that the γ maximizing the predictive capabilities of the filter strongly depends on the number of samples M, i.e., on the accuracy of the experimental evidence.

If $N_{\max}$ is not known a-priori, or if its value is uncertain, then γ can also be tuned on-line, e.g., with cross-validation methods [Hastie et al., 2009, Section 7.10]. In this case, tuning γ, assuming that the probabilities q and p are known, corresponds to estimating $N_{\max}$ given q, p and M.

4.4 Evaluation of the Size Estimation Algorithm

Let us evaluate the regularization-based dynamic network size estimator on the Markovian network model introduced in Section 4.3. We generate networks with $N_{\max} = 1000$ peers, and let the transition probabilities be $p = q = 0.01$. Each active peer is generating $M = 200$ uniformly random samples at each time step, and the regularization parameter is chosen as $\gamma = 0.001$.

We start by noticing the beneficial effects of our regularization approach from equation (4.16) in Figure 4.2, where we compare the outcomes of our estimator with a point-wise estimation (corresponding to $\gamma = 0$).

We examine the effect of the regularization parameter γ when the network size changes rapidly in Figure 4.3, e.g., by a flash crowd joining the network, followed by a catastrophic network failure. The estimator derived for the previous Figure 4.2 is used, but γ varies between 0.001, 0.010 and 0.100. Here, the true network size is piecewise constant, at 200 peers until $t = 20$, then increasing instantaneously to 800 peers, before finally dropping down to 500 peers at $t = 60$. It is evident that a larger regularization parameter γ yields a smoother tracking of the network size, but will also be slower at detecting rapidly changing topologies.

Next we examine the effects of the parameters p, q, γ, $N_{\max}$ and M on the estimation performance by considering 4 different scenarios: $p = q = 0.1$ or 0.01; $N_{\max} = 1000$ or 2000. For each of these scenarios we evaluate the root-mean-square error (RMSE) by generating 1000 independent network models $N_j(t)$, $t = 1, \ldots, 100$, $j = 1, \ldots, 1000$ from the Markovian model in Section 4.3. For each trajectory $N_j(t)$ we compute the estimator of equation (4.16) using different sample sizes M in $[10, 200]$ and different regularization parameters γ in $[10^{-6}, 10^{-2}]$. In Figure 4.4 each of the 4 subplots then illustrates the dependency on M and γ of

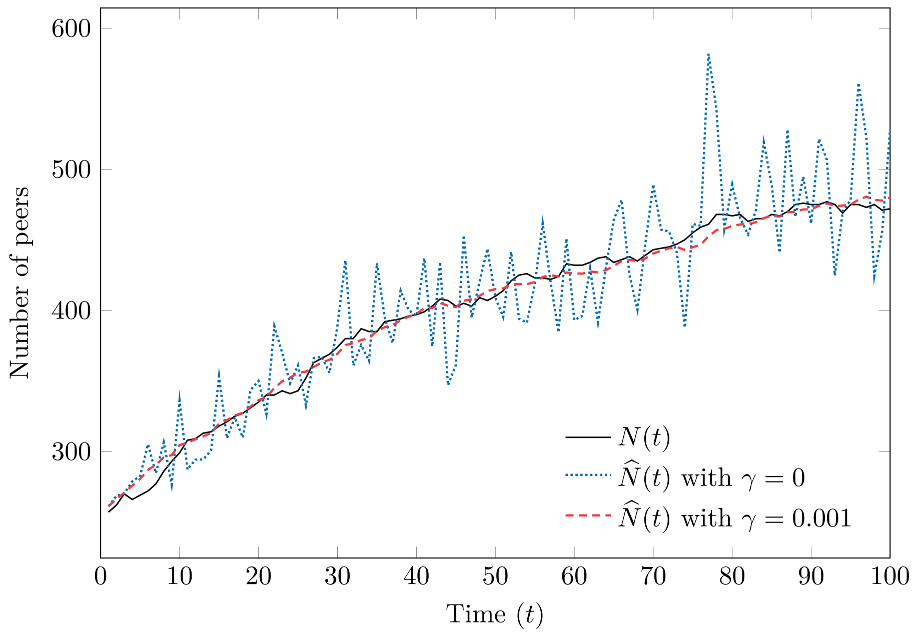


Figure 4.2: Comparison of the results for a regularization based estimator and a point-wise estimator, for the same set of $s(\tau)$. The network consists of $N_{\max} = 1000$ nodes, with transition probabilities $p = q = 0.01$.

the RMSE, defined as:

$$\mathrm{RMSE}(M, \gamma) \doteq \sqrt{\frac{1}{10^5} \sum_{j=1}^{1000} \sum_{t=1}^{100} \left(N_j(t) - \widehat{N}_j(t\,;\ M, \gamma) \right)^2}, \tag{4.17}$$

Assuming that p, q, M and $N_{\max}$ are fixed, then there exists an optimal regularization parameter γ^* minimizing the RMSE. The behaviors of the four surfaces supports the following intuitive rules-of-thumb for selecting the estimator's parameters:

- if p, q and $N_{\max}$ are fixed, then increasing the sample size M leads to a smaller optimal regularization parameter γ;
- if M and $N_{\max}$ are fixed, then increasing the rates p and q leads to a smaller optimal regularization parameter γ;
- if p, q and M are fixed, then increasing the maximal network size $N_{\max}$ leads to a smaller optimal regularization parameter γ.

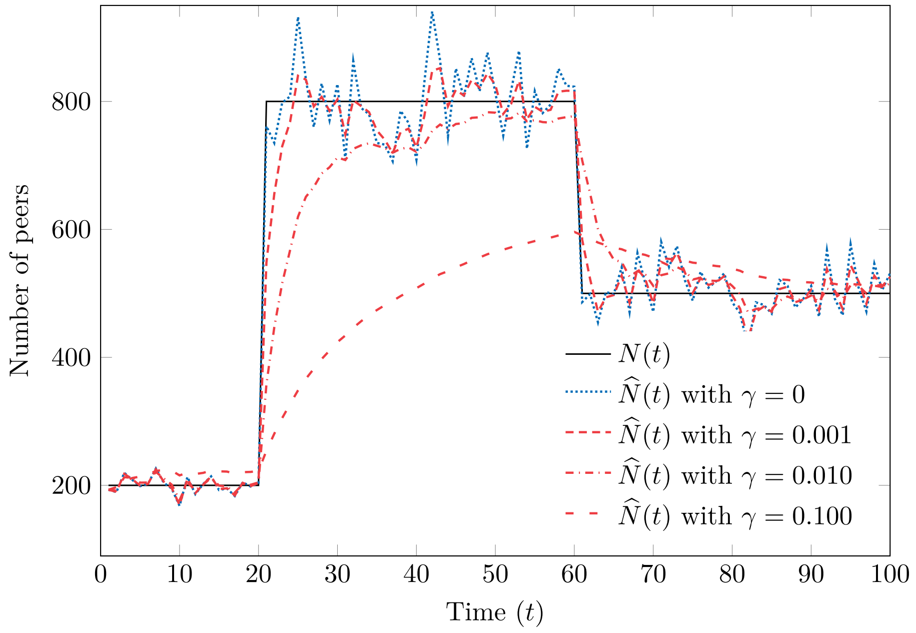


Figure 4.3: Comparison of different regularization parameters γ for a network with sudden step changes. The network starts with 200 peers, which increases to 800 peers at time $t = 20$, and then drops down to 500 peers at time $t = 60$. A larger regularization parameter γ leads to a smoother tracking, but a slower response to sudden changes in size.

We finally evaluate how finite representations using only b bits of the random samples $x_{i,m}(t)$ in Algorithm 4.1 can affect the estimation performances, i.e.,

$$x_{i,m}(t) \in \{0,\, \alpha,\, 2\alpha,\, \ldots,\, 1\} \quad \text{with} \quad \alpha = \frac{1}{2^b - 1}. \tag{4.18}$$

Considering again 1000 trajectories $N_j(t)$, $t = 1, \ldots, 100$, $j = 1, \ldots, 1000$ from the network model in Section 4.3, with $N_{\max} = 1000$, $p = q = 0.01$, $M = 200$ and $\gamma = 0.001$, as in Figure 4.2. During the communication step the peers only use b-bits precision, but later in the local computation of the estimate in equation (4.16), they use full 64-bits precision. The average RMSE performance index shows, in Figure 4.5, that for small networks it is sufficient to represent the random samples $x_{i,m}(t)$ with 12 bits.

Remark. The experiments in Figures 4.2 to 4.4 have been computed with the discretization scheme in Figure 4.5 using 12 bits. With $M = 200$ random samples, and ignoring the communication protocol overheads, this leads to data packets consisting of 300 bytes.

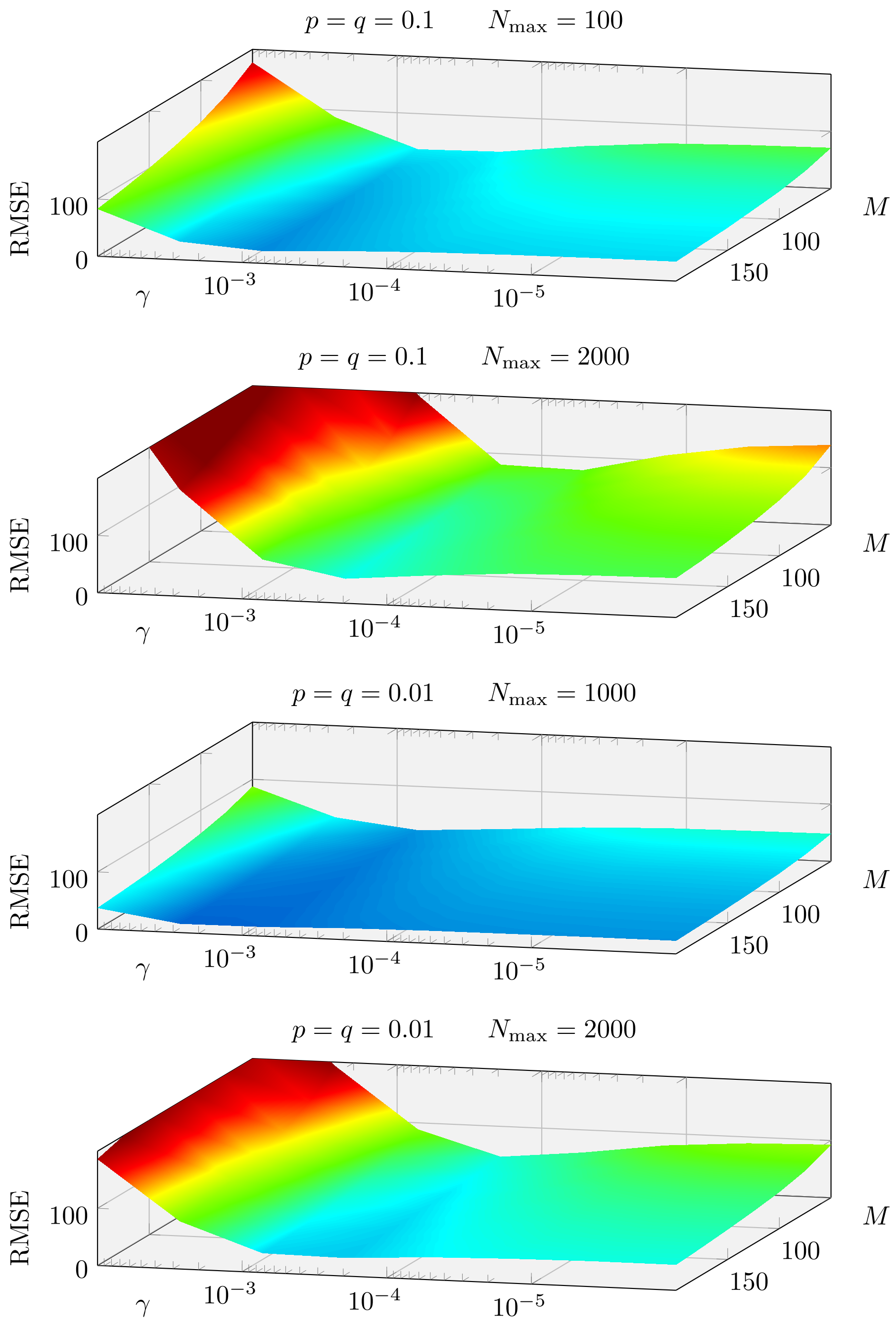


Figure 4.4: Dependency of the average root-mean-square error on the parameters M and γ for various values of p, q and $N_{\max}$.

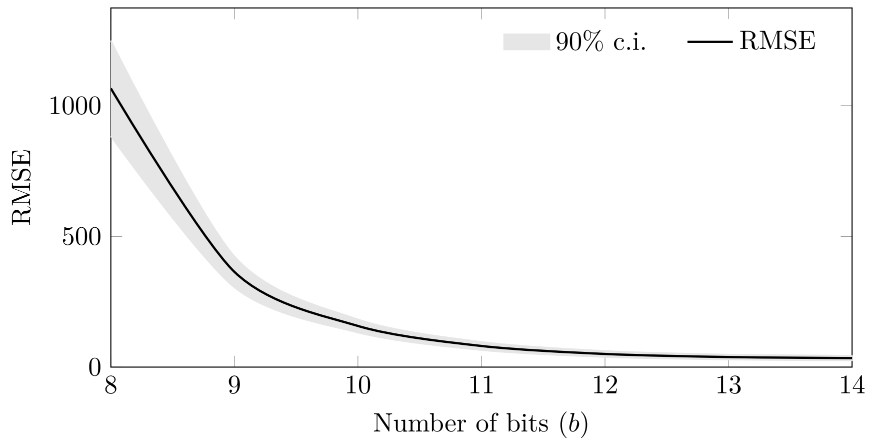


Figure 4.5: Dependency of the RMSE (4.17) on the number of bits used to represent the samples $x_{i,m}(t)$, assuming equations (4.16) and (4.17) to be computed using 64-bits precisions. The expected RMSE and the 90 % confidence interval are shown. $N_{\max} = 1000$, $p = q = 0.01$, $M = 200$ and $\gamma = 0.001$.

4.5 Estimation of Probability Mass Functions

As we have seen so far, aggregating and estimating data over networks is essential for many distributed systems. However, simple aggregations, such as computing averages, maxima, or sums of a distributed data set, lose a lot of the information contained in the original data. We will now continue and propose algorithms that estimate the entire empirical PMF over anonymous networks. Specifically, we consider protocols that aim to estimate the empirical distributions in the shortest possible time.

Our proposed strategy is based on the max consensus estimator in Section 4.2. From an algorithmic point of view our strategy departs from Borges et al. [2012], Sacha et al. [2010] by substituting the average consensus schemes with max consensus. This apparently minor modification actually makes the two estimators completely different, and opens up for a variety of novel problems. In fact, while the average consensus scheme requires exchanging very few scalars per iteration, and where the agents compute the *exact* PMF asymptotically in time, the max consensus scheme converges much faster than the average consensus scheme, but not to the exact value. Again, the statistical performance depends on how many scalars are exchanged per iteration. We specifically compare the temporal behavior of these two strategies.

The PMF Estimation Problem

Consider a strongly connected network $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ of $N = |\mathcal{V}|$ agents communicating through the links $\mathcal{E}$. In this section we assume the network to be static, and return to the problem of estimating PMFs in time dependent networks in Section 4.9. Let $\mathcal{N}_i$ denote the set of neighbors of agent i, and $\mathcal{N}_i^{(t)}$ the set of its t-steps neighbors, i.e., all agents that can be reached by paths of length at most t steps from agent i. Recall that $\mathcal{N}_i^{(t)}$ can be defined for $t = 0$ as $\mathcal{N}_i^{(0)} = \{i\}$ and, for $t \geq 1$, through the recursion

$$\mathcal{N}_i^{(t)} \doteq \bigcup_{j \,:\, (i,j) \in \mathcal{E}} \mathcal{N}_j^{(t-1)} . \tag{4.19}$$

Let every agent $i \in \mathcal{V}$ belong to a discrete state $z_i \in \mathbb{N}_B \doteq \{0, \dots, B-1\}$, e.g., given by sensor measurements, where $\mathbb{N}_B$ is the set of plausible states. We are then interested in distributively estimating the relative frequencies of the local states $z_1, \dots, z_N$, i.e., if $n_b \doteq \left|\{i \;:\; z_i = b\}\right|$ is the number of agents in state b, then we aim to estimate the PMF

$$p_b \doteq \frac{n_b}{N}, \qquad b \in \mathbb{N}_B, \tag{4.20}$$

given that the network size N is unknown, while the plausible states $\mathbb{N}_B$ are known.

We focus on distributed algorithms where each agent $i \in \mathcal{V}$ has a local variable $x_i(t)$ that can be modified at time $t+1$ by accessing the states $x_j(t)$'s of the neighboring nodes, and performing the aggregation operation

$$x_i(t+1) = f\left(x_i(t), x_{j_1}(t), x_{j_2}(t), \dots\right), \quad j_1, j_2, \dots \in \mathcal{N}_i$$

that preserves the dimension of $x_i(t)$. At every time t, each agent also computes a local estimate of the PMF from the local variable $x_i(t)$,

$$\widehat{p}_b^{(i)}(t) = g\left(x_i(t)\right)$$

for an appropriate estimation function $g(\cdot)$.

The estimation strategy is thus defined by the initial variables $x_i(0)$, the update function f and the estimation function g. In order to compare different estimation strategies we consider the mean squared error (MSE) as a performance index, i.e.,

$$\mathrm{MSE}(\widehat{p}_1, \dots, \widehat{p}_B) \doteq \mathbb{E}\left[\frac{1}{N \cdot B} \sum_{b \in \mathbb{N}_B, i \in \mathcal{V}} \left(p_b - \widehat{p}_b^{(i)}\right)^2\right], \tag{4.21}$$

where the expectation is taken over all initial conditions.

4.6 PMF Estimators Based on Consensus Protocols

We consider two particular estimators, one based on the average consensus strategy, and one based on the max consensus size estimation strategy from Section 4.2.

In the following, we abstract away the message transmission and consider a distributed system where agents communicate by synchronous rounds. At each round, and over each edge, only a constant size message is transmitted.

Remark. For notational simplicity we consider synchronous communication steps. Nonetheless this could be relaxed for both estimators, since they can be adapted to operate with asynchronous gossip communication.

Estimator Based on Average Consensus

In the average consensus based estimator, each node starts with a PMF estimate given by only its own state, and then, by averaging all these estimates, the network's PMF is computed. Thus, the local variable is a B-dimensional real vector $x_i(t) \in \mathbb{R}^B$ containing the current estimate of the PMF. At initialization, each node sets its local variable based on its own state as

$$x_i^{(b)}(0) = \begin{cases} 1 & \text{if } z_i = b, \\ 0 & \text{otherwise.} \end{cases}$$

Let $\boldsymbol{x}^{(b)}$ denote the vector of all agents' states $x_i^{(b)}$. It is known that if at each time the local variables are updated with an average consensus update

$$\boldsymbol{x}^{(b)}(t+1) = W\boldsymbol{x}^{(b)}(t), \quad b \in \mathbb{N}_B, \tag{4.22}$$

where W is a doubly-stochastic weight matrix (for example chosen as the Metropolis weights), then, assuming perfect computations [2], every $x_i^{(b)}(t)$ converges to the average of the initial values [Fagnani and Zampieri, 2008]. Thus

$$x_i^{(b)}(t) \xrightarrow{t\to\infty} \frac{1}{N}\sum_{j\in\mathcal{V}} x_i^{(b)}(0) = \frac{n_b}{N} = p_b.$$

The PMF estimate is simply the local state,

$$\widehat{p}_b^{(i)}(t) = x_i^{(b)}(t). \tag{4.23}$$

To describe the convergence properties of equation (4.22), recall that the estimation error can be bounded by an exponential function [Xiao and Boyd, 2003], i.e., by

$$\left|\left|p_b - \widehat{p}_b(t)\right|\right|_2 \leq ce^{-\alpha t}$$

where c and α depend on the initial condition, the network topology and the choice of the weights.

[2]For simplicity we do not consider the quantization effects [Carli et al., 2010].

Remark. We do not consider more advanced protocols, such as accelerated average consensus [Aysal et al., 2009], or finite-time average consensus [Yuan et al., 2011]. The reason is that we want to characterize the simplest averaging algorithm, with the smallest demands from both communication and computational points of view.

Estimator Based on Max Consensus

The max consensus based estimator uses the the previous size estimation technique to count the number of nodes in each state, from which the PMF is computed. Thus, the local variable is instead a $B \times M$-dimensional real matrix $x_i(t) \in \mathbb{R}^{B\times M}$, for which the elements are initially generated from a uniform random distribution based on the local state as

$$x_i^{(b,m)}(0) \sim \begin{cases} \mathcal{U}\left[0,1\right] & \text{if } z_i = b, \\ 0 & \text{otherwise,} \end{cases} \tag{4.24}$$

where $\mathcal{U}\left[0,1\right]$ is the uniform distribution between 0 and 1. Then at each time t, the local variables are updated with the max consensus update

$$x_i^{(b,m)}(t) = \max_{j\in\mathcal{N}_i} \left\{ x_j^{(b,m)}(t-1) \right\}, \quad b \in \mathbb{N}_B,\ m = 1, \ldots, M. \tag{4.25}$$

Notice that the definition of a t-steps neighborhood $\mathcal{N}_i^{(t)}$ precisely captures the agents that contributed to the generation of $x_i^{(b,m)}(t)$, i.e.,

$$x_i^{(b,m)}(t) = \max_{j\in\mathcal{N}_i^{(t)}} \left\{ x_j^{(b,m)}(0) \right\}.$$

Let $N_i^{(t)} \doteq \left|\mathcal{N}_i^{(t)}\right|$ be the number of t-step neighbors, then

$$p_b^{(i)}(t) \doteq \frac{\left|\{i \in \mathcal{N}_i^{(t)} \ : \ z_i = b\}\right|}{N_i^{(t)}},$$

and $n_b^{(i)}(t) \doteq p_b^{(i)}(t) N_i^{(t)}$. As was shown in equation (4.7), the ML estimator for $n_b^{(i)}(t)$ given the $x_i^{(b,m)}(t)$'s is

$$\widehat{n}_b^{(i)} = -\left(\frac{1}{M} \sum_{m=1}^{M} \log\left(x_i^{(b,m)} \right) \right)^{-1}. \tag{4.26}$$

Note that the PMF is given by

$$p_b^{(i)}(t) = \frac{p_b^{(i)}(t)}{\sum_{\beta\in\mathbb{N}_B} p_\beta^{(i)}(t)} = \frac{n_b^{(i)}(t)}{\sum_{\beta\in\mathbb{N}_B} n_\beta^{(i)}(t)}.$$

Now, because of the functional invariance property of ML estimators [Casella and Berger, 2002, Thm. 7.2.10, p. 320], the ML estimate of $p_b^{(i)}(t)$ given the $x_i^{(b,m)}(t)$'s is

$$\widehat{p}_b^{(i)}(t) = \frac{\widehat{n}_b^{(i)}(t)}{\sum_{\beta \in \mathbb{N}_B} \widehat{n}_\beta^{(i)}(t)}. \tag{4.27}$$

For $t \geq d$ (d being the network diameter) the max consensus strategy converges globally, and $n_b^{(i)}(t) = n_b$, thus the PMF estimated $p_1^{(i)}(t), \ldots, p_B^{(i)}(t)$ converges to an estimate of the global PMF $p_1, \ldots, p_B$.

Additionally, this estimator provides estimates of the distributions of the states in every t-steps neighborhood. Considering a certain agent i, the set of $p_b^{(i)}(0)$, $p_b^{(i)}(1), \ldots$ corresponds to local views of the neighborhood's empirical distribution that can be used by i to rapidly infer if close neighbors tend to have similar states.

Notice that the statistical properties of the estimator in equation (4.27) are essentially different from the previous network size estimation scheme, since the vectors $\left[\widehat{p}_1^{(i)}(t), \ldots, \widehat{p}_B^{(i)}(t)\right]$ have correlated components. We also notice that appropriate termination rules can be based on estimates of the diameter d of the network, again obtained by exploiting max consensus approaches as done by Cardoso et al. [2009] and Garin et al. [2012].

Finally, notice that under continuity assumptions, the choice of the stochastic generation mechanism proposed in equation (4.24) is general, since as soon as we neglect quantization effects, substituting $\mathcal{U}[0,1]$ with another continuous probability distribution leads to estimators with identical statistical performance [Varagnolo et al., 2010].

Summary of the Differences Between the Two Estimators

The max consensus scheme in equation (4.27) converges in a finite d steps to an estimate of the true PMF. Given a fixed sample size M, its MSE (4.21) will vary up to time $t = d$ and then remain constant. By increasing the sample size M, the MSE curves are also expected to get closer to zero, due to the consistency property of ML estimators. The average consensus scheme in equation (4.23) requires nodes to exchange less information, and generally converges asymptotically for $t \to +\infty$. These comments are illustrated in Figure 4.6.

The aim is to find conditions for the sample size M and the network for which it is possible to state which algorithm is preferred when $t \leq d$, i.e., when time is a concern. To this end, we first need to describe the statistical properties of the max consensus estimator.

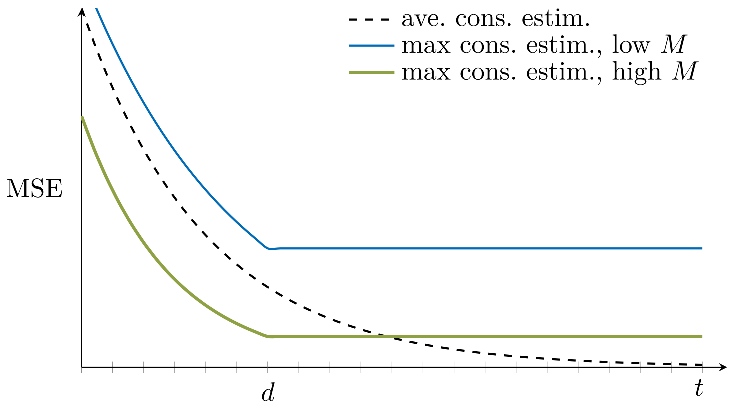


Figure 4.6: MSE for average and max consensus PMF estimators. By increasing the sample size M it is possible to make the max consensus estimator in equation (4.27) perform better than the average consensus scheme in equation (4.23) for $t \leq d$, where d is the network diameter.

4.7 Statistical Characterization of the Max Consensus PMF Estimator

Interrupting the max consensus protocol before it has reached consensus is equivalent to estimating the PMF for a t-step neighborhood. Thus, for notational simplicity we consider the stationary state where the max consensus has already been reached, i.e., where $x_i^{(b,m)}(t) = x^{(b,m)} \doteq \max_{i \in \mathcal{V}} \left\{ x_i^{(b,m)}(0) \right\}$. With this assumption the joint PMF $p\left(\widehat{n}_b \; ; \; n_1, \ldots, n_B, M\right)$ is equal to each node i's local PMF $p\left(\widehat{n}_b^{(i)}(t) \; ; \; n_1^{(i)}(t), \ldots, n_B^{(i)}(t), M\right)$. To derive this distribution, consider that $x^{(b,m)}$ is statistically independent of the parameter n_β if $b \neq \beta$. Thus, from simple order-statistic arguments [David and Nagaraja, 2004],

$$p\left(x^{(b,m)} \; ; \; n_1, \ldots, n_B\right) = p\left(x^{(b,m)} \; ; \; n_b\right) = n_b \left(x^{(b,m)}\right)^{n_b - 1}$$

for all m (omitting the dependency on the parameter M for notational simplicity). Since the states $x^{(b,m)}$'s are i.i.d., we have

$$p\left(x^{(b,1)}, \ldots, x^{(b,M)} \; ; \; n_b\right) = \prod_{m=1}^{M} p\left(x^{(b,m)} \; ; \; n_b\right) = n_b^M \prod_{m=1}^{M} \left(x^{(b,m)}\right)^{n_b - 1}. \quad (4.28)$$

To derive the probability density $p\left(\widehat{n}_b \; ; \; n_b\right)$, consider that $y \doteq -\log\left(\left(x^{(b,m)}\right)\right)$ is an exponential random variable with rate n_b, i.e.,

$$p\left(y \; ; \; n_b\right) = \begin{cases} n_b e^{-n_b y} & \text{if } y \geq 0, \\ 0 & \text{otherwise.} \end{cases} \quad (4.29)$$

From equation (4.26), $M\widehat{n}_b^{-1}$ is the sum of M i.i.d. exponential random variables with rate n_b, i.e., $M\widehat{n}_b^{-1}$ is a Γ variate with shape M and scale n_b^{-1}. Thus $M^{-1}\widehat{n}_b \sim$ I-$\Gamma\left(M, n_b\right)$, where I-$\Gamma\left(\cdot,\cdot\right)$ is the inverse Gamma distribution with shape M and scale n_b, i.e.,

$$\begin{aligned} p\left(\widehat{n}_b \; ; \; n_b, M\right) &= \text{I-}\Gamma\left(M, M n_b\right) \\ &= \Gamma\left(M\right)^{-1} \frac{1}{\widehat{n}_b} \left(\frac{M n_b}{\widehat{n}_b}\right)^M \exp\left(-\frac{M n_b}{\widehat{n}_b}\right). \end{aligned}$$

For the estimate in equation (4.27), $\widehat{p}_b$ is the ratio of correlated sums of inverse-Gamma variates, each with its own scale.

To the best of our knowledge, there exists no literature describing a distribution of this kind. The closest results characterize ratios of the form $\frac{x}{x+y}$, where x and y are independent inverse Gamma variates [Ali et al., 2007]. Moreover both the Gamma and inverse Gamma distributions are not closed, i.e., linear combinations of independent copies of these kinds of variates do not have the same original distribution up to location and scale parameters [Witkovský, 2001]. This means that there is, in general, no possibility to reduce equation (4.27) to the case described by Ali et al. [2007]. Our characterization of the statistical properties of $\widehat{p}_b$ rely instead on Monte Carlo (MC) integration methods.

Case $\mathbb{N}_B = \{0, 1\}$

Consider the restricted case when the distribution only consists of two states, $\mathbb{N}_B = \{0, 1\}$. Then $\widehat{p}_b^{(i)}(t)$ becomes a special ratio that is described by Ali et al. [2007], with probability density

$$p_{\widehat{p_0}}\left(x \; ; \; n_0, \, n_1, \, M\right) = \frac{\Big(x(1-x)\Big)^{M-1}}{\left(\dfrac{n_0}{n_1}\right)^M B\left(M, M\right)} \left(1 + \frac{n_1 - n_0}{n_0} x\right)^{-2M}, \tag{4.30}$$

where $B\left(\cdot,\cdot\right)$ is the Beta function, and $x \in [0, 1]$. Its cumulative distribution is given by

$$\begin{aligned} F_{\widehat{p_0}}\left(x \; ; \; n_0, \, n_1, \, M\right) &= \frac{\left(1 + \frac{n_1}{n_0}\frac{1-x}{x}\right)^{-M}}{M \cdot B\left(M, M\right)} \\ &\qquad \times \; {}_2F_1\left(M, 1-M; M+1; \left(1 + \frac{n_1}{n_0}\frac{1-x}{x}\right)^{-1}\right), \end{aligned}$$

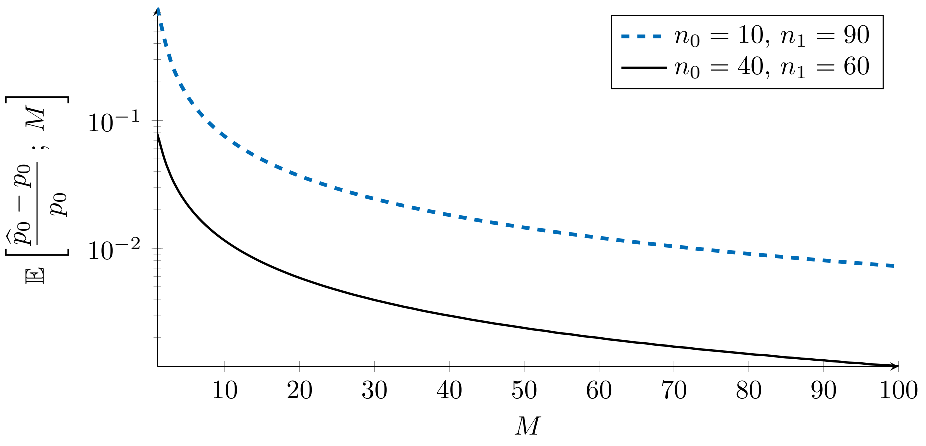


Figure 4.7: The relative bias, $\mathbb{E}\left[\frac{\widehat{p}_0 - p_0}{p_0};\ M\right]$, dependency on M for different values of n_0 and n_1. The estimator is unbiased if $n_0 = n_1$.

where

$$_2F_1\left(a,\, b;\, c;\, x\right) \doteq \sum_{i=0}^{+\infty} \frac{(a)_i\,(b)_i}{(c)_i \cdot i!} x^i \tag{4.31}$$

is the Gauss hypergeometric function and

$$(x)_i \doteq x(x+1)\cdots(x+i-1) \tag{4.32}$$

is the Pochhammer symbol (with the convention that $(x)_0 = 1$). From this, it is possible to compute the moments of $\widehat{p}_0$ using the relation

$$\mathbb{E}\left[(\widehat{p}_0)^k\right] = \begin{cases} \dfrac{B\,(M+k, M)}{B\,(M, M)} \mathcal{F}(k,\, M,\, n_0,\, n_1) & \text{if } n_0 > n_1, \\ \left(\dfrac{n_0}{n_1}\right)^k \dfrac{B\,(M+k, M)}{B\,(M, M)} \mathcal{F}(k,\, M,\, n_1,\, n_0) & \text{otherwise.} \end{cases} \tag{4.33}$$

where

$$\mathcal{F}(k,\, M,\, a,\, b) \doteq {}_2F_1\left(k,\, M;\, 2M+k;\, \frac{a-b}{a}\right)$$

Remark. Note that n_0 and n_1 appear in inverted positions in the two cases in equation (4.33).

When $n_0 = n_1$, the estimators are unbiased for every M, otherwise, as expected, they are only asymptotically unbiased (for $M \to +\infty$).

In Figure 4.7 we evaluate the relative bias, and in Figure 4.8 the relative MSE of the estimator, based on the design parameter M and on the distribution of the states. Note that the MSE performance follows the typical $1/M$ behavior for this kind of estimators.

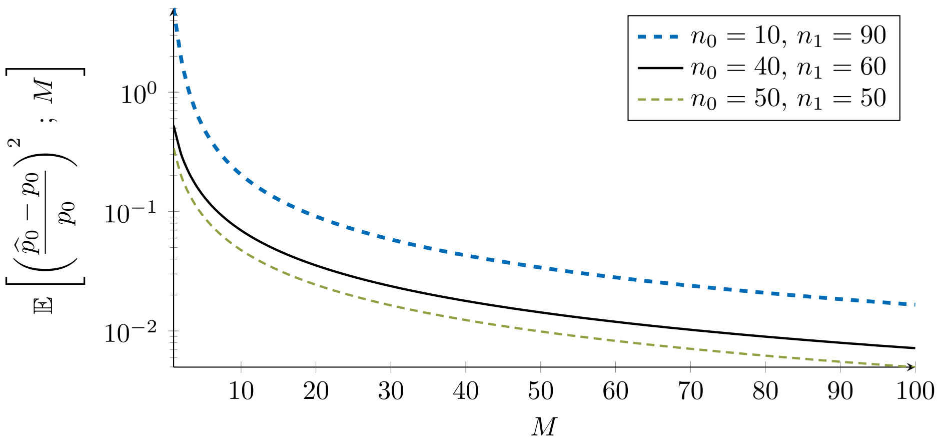


Figure 4.8: The relative MSE, $\mathbb{E}\left[\left(\frac{\widehat{p}_0 - p_0}{p_0}\right)^2 ;\ M\right]$, dependency on M for different values of n_0 and n_1.

Remark. The performance indicators summarized in Figures 4.7 and 4.8 are valid for general $\widehat{p}_b^{(i)}(t)$'s when associated with the local $n_b^{(i)}(t)$'s. The derivations of this section also characterize the behavior of the estimators during the transient phase.

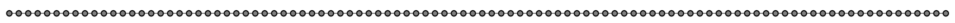
4.8 Transient Evaluation of PMF Estimators

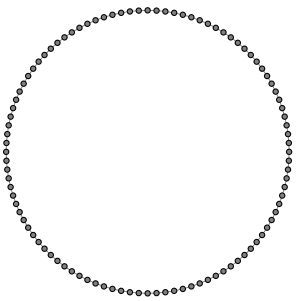
Here we evaluate the performance of the average consensus based estimator of equation (4.23) and the max consensus based estimator of equation (4.27) during their transient phases. The primary goal is to determine when to use each algorithm, and how to tune the sample size M for the max consensus estimator.

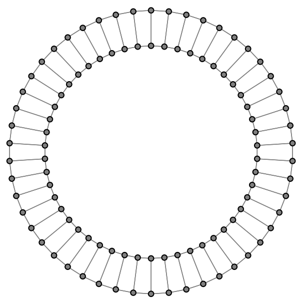
We consider four different network topologies, the line topology (Figure 4.9a), the cyclic topology (Figure 4.9b), the cyclic grid topology (Figure 4.9c) and a geometric random topology (Figure 4.9d), where each network consists of 100 agents.

We evaluate the algorithms through MC simulations, using the MSE from equation (4.21) as the performance index, where the mean is taken over all agents and all MC simulations. On each network the communication protocol proceeds in synchronous steps, where nodes cyclically repeat the algorithms described in equation (4.22) and equation (4.25).

In the first experiment, Figure 4.10, the initial state is selected randomly for each MC simulation, where each agent is placed in either state $z_i = 0$ or $z_i = 1$ with equal probability. The figure shows the 95 % confidence intervals for both the average consensus based estimator as well as for the max consensus based estimator with $M = 10$, $M = 100$ and $M = 1000$.


(a) Line network


(b) Cyclic network


(c) Cyclic grid network with (2×50) nodes

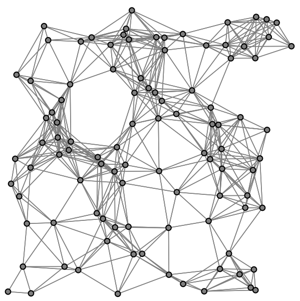

(d) Geometric random network

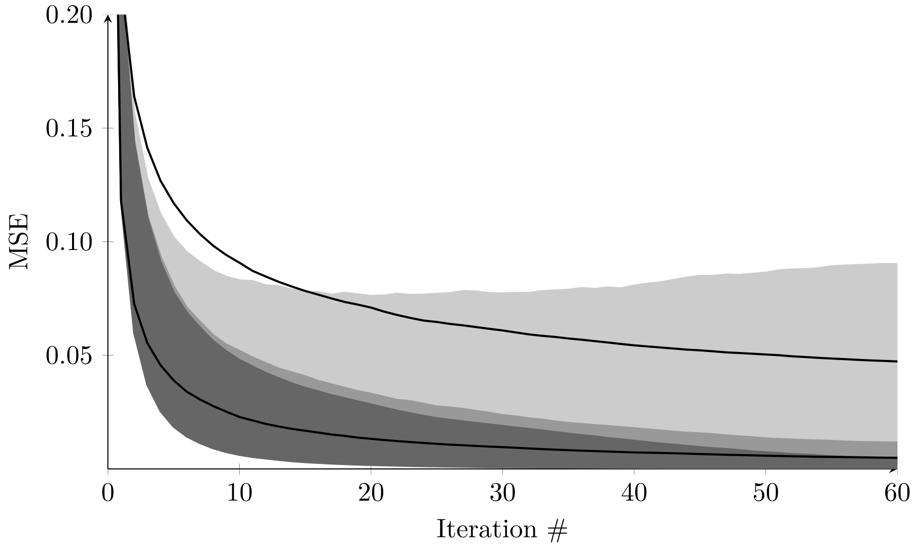
Figure 4.9: Network topologies, with 100 nodes.

As expected, the average consensus based estimator converges *asymptotically* to the true value, while the max consensus based estimator converges *in finite time* (after d steps, where d is the diameter of the network). However, the max consensus based estimate does not converge to the true value, but instead its MSE decreases with the sample size M. In this scenario the choices of $M = 100$ and $M = 1000$ yield similar precisions that outperform the average consensus in most reasonable time scales.

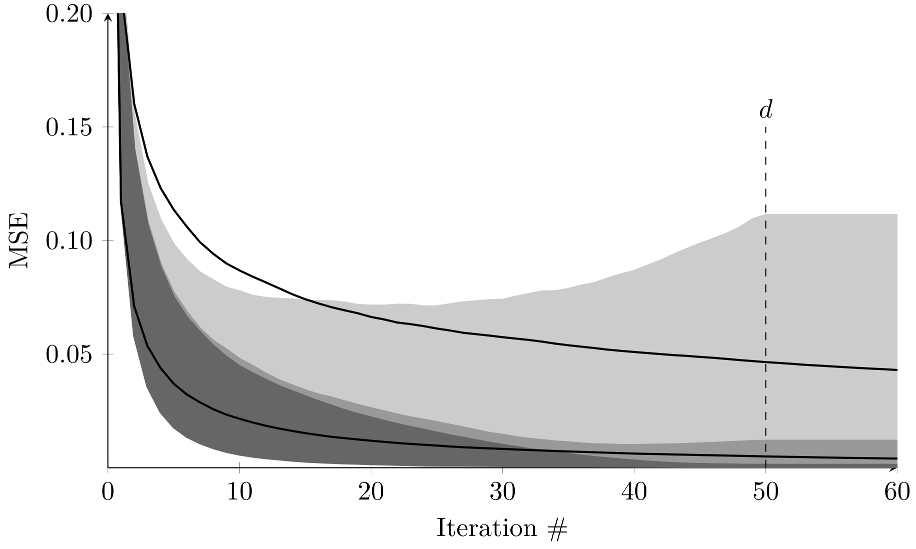
We observe that for the max consensus-based scheme a remarkable phenomenon may appear, especially when the sample size M is small (e.g., $M = 10$ in Figure 4.10), the MSE actually sometimes increases with the number of iterations. This phenomenon appears because the MSE index sums the agents' local MSEs, and small M's induce estimates with high statistical variance, i.e., increase the chances that at least one agent will have some $\widehat{p}_b^{(i)}(t)$ noticeably overestimated. At time $t = 1$ this overestimation has not yet influenced a majority of the agents and the overall MSE, since it only affected the erroneous agent, but as time passes, the max consensus protocol spreads this overestimation through the network of agents, which is seen in the MSE.

In the second experiment, Figure 4.11, we switch the initial condition to a single worst-case distribution of the states z_i, where the leftmost half of the agents in Figure 4.9 are in state 0 and the rightmost half are in state 1. Notice that this corresponds exactly to a spatial correlation between the agents' positions and their measurements, which is actually a reasonable assumption for estimation applications in wireless sensor networks.

Since there is only one fixed initial state, the average consensus based estimator is now deterministic and unique. The figure thus compares the confidence intervals of the max consensus estimators (depending upon the realization of the initial ran-

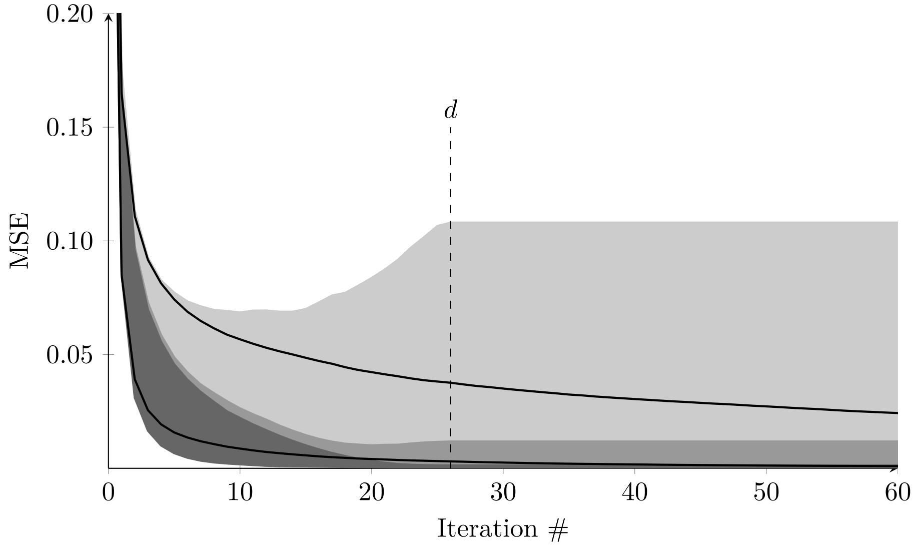


(a) Line network (Diameter $d = 99$)

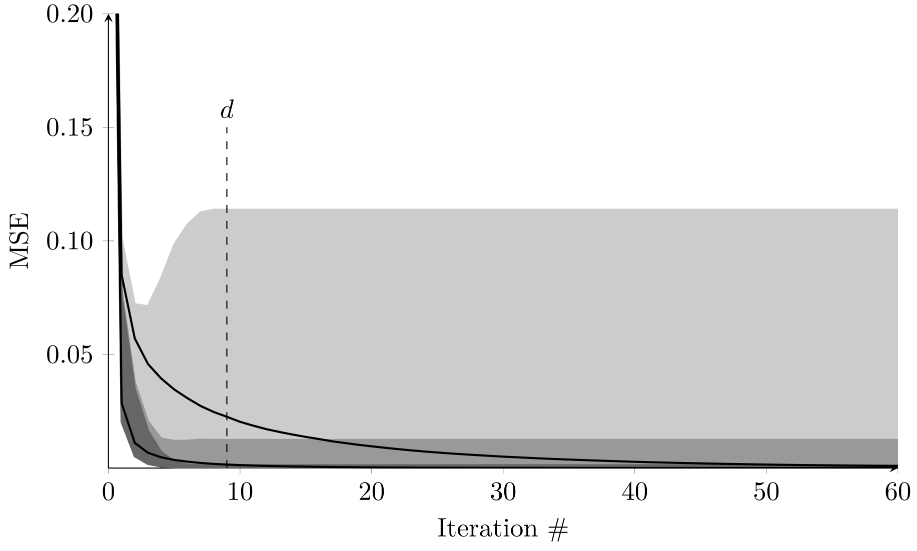


(b) Cyclic network

▨ $M = 10$ ▨ $M = 100$ ▨ $M = 1000$ — Average



(c) Cyclic grid network (2×50 nodes)



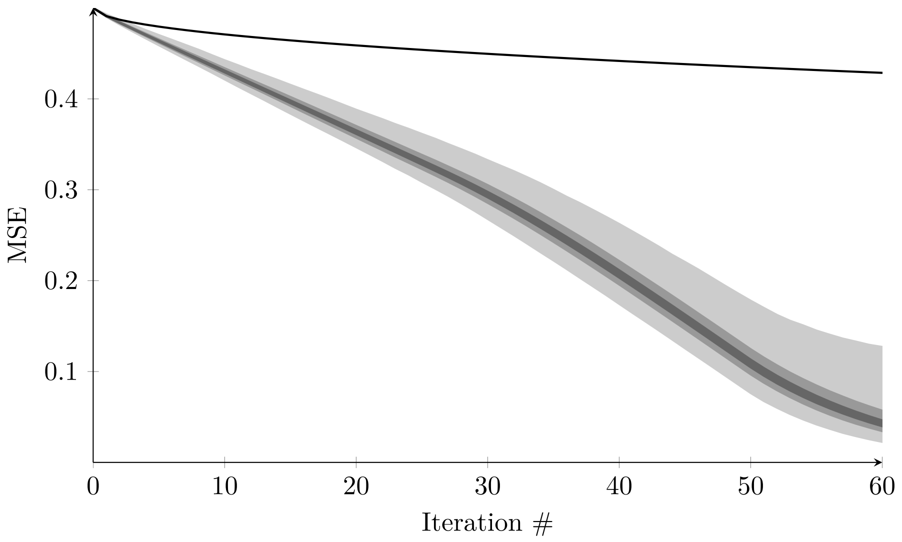
(d) Geometric network

▨ $M = 10$ ▨ $M = 100$ ▨ $M = 1000$ —— Average

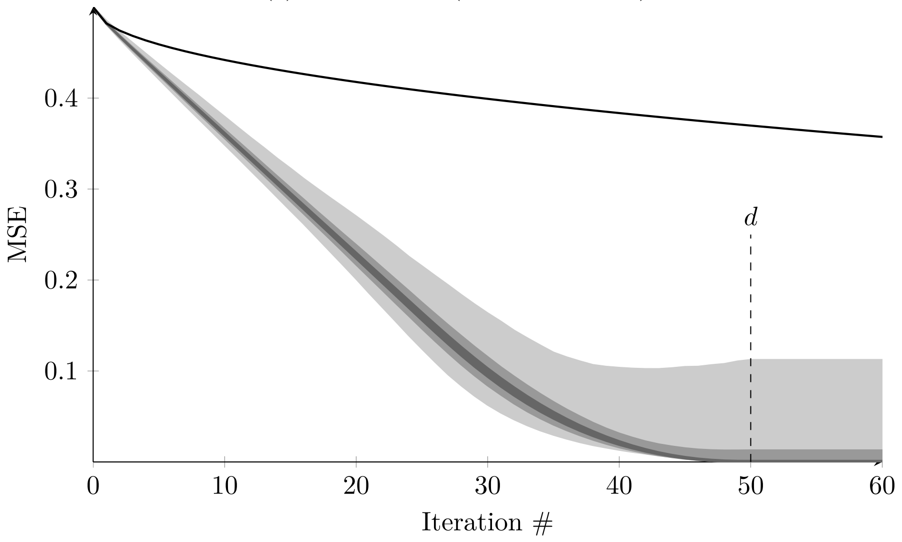
Figure 4.10: Comparison of max consensus based estimator against the average consensus based estimator. Each network consists of 100 nodes, and the network diameter d is marked in the figures. The shaded regions mark the 95 % confidence interval for the max consensus estimator, while the two solid lines mark the upper and lower bound of the 95 % confidence interval for the average consensus estimator.

dom sample $x_i^{(b,m)}$) against the performance of the deterministic average consensus estimate.

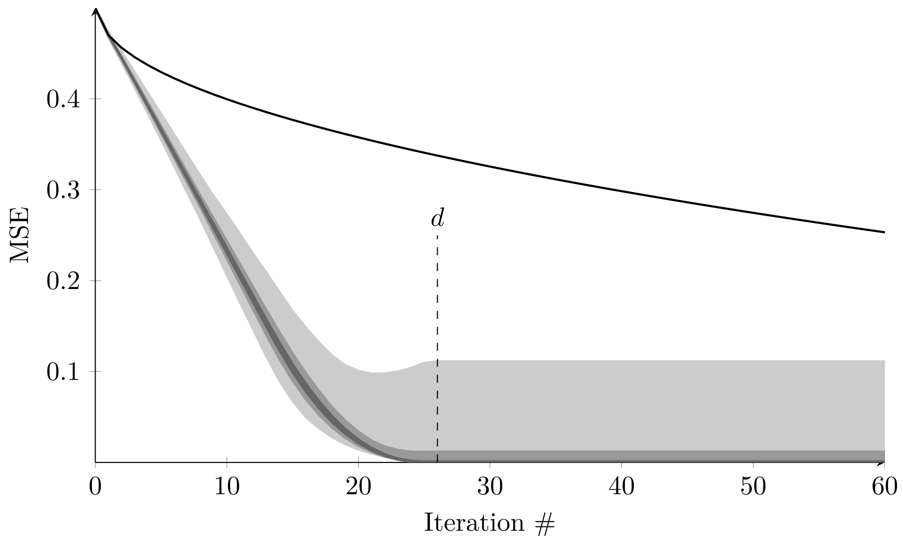
The first thing we notice is that the convergence speed is slower than for the randomized initial distribution, because in that case each t-step neighborhood was a reasonable sample of the complete distribution, while in this case a t-step neighborhood ($t \ll d$) is not a good representation of the complete distribution. Still, this example shows even more clearly that a max consensus based estimator can be much faster and accurate than an average consensus based counterpart, even for very small M's (even though a larger M improves the accuracy). The motivation is that the max consensus protocol has a much faster mixing time, and if the distribution of states in the network topology is not homogeneous then the max consensus is much more efficient at propagating information about certain states through the network.



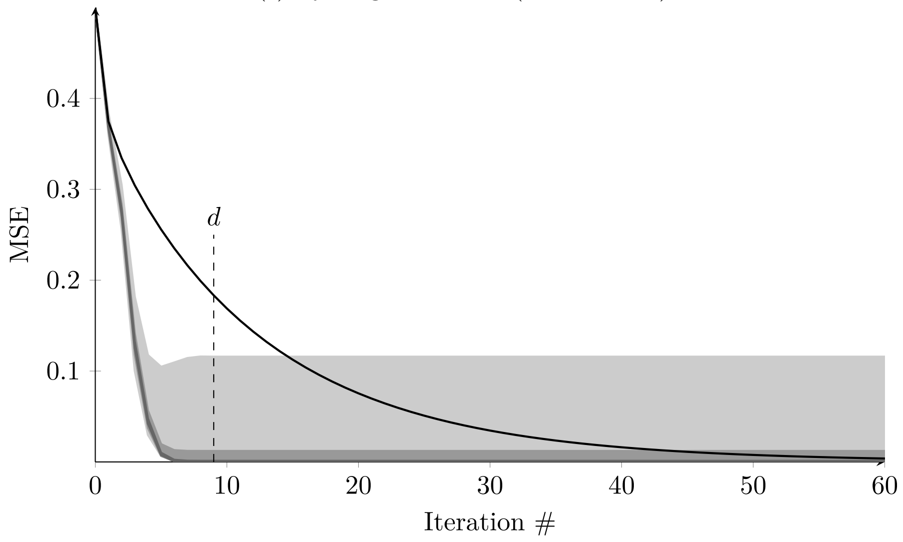
(a) Line network (Diameter $d = 99$)



(b) Cyclic network



(c) Cyclic grid network (2×50 nodes)



(d) Geometric network

Figure 4.11: Comparison of max consensus based estimator against the average consensus based estimator for a single worst-case initial condition. Each network consists of 100 nodes, and the initial state is determined by the agents spatial configuration. The shaded regions mark the 95 % confidence interval for the max consensus estimator, while the solid line marks the deterministic estimation for the average consensus estimator.

4.9 Estimating Time Dependent PMFs

The PMF estimator we proposed in Section 4.6 is using the max consensus based network size estimation scheme that we developed in Section 4.2 to handle time dependent networks. It is therefore straightforward to handle time dependent PMF estimation by substituting the regularization based estimator in equation (4.6) into the PMF estimator in equation (4.26).

This can be demonstrated by considering a Markov chain driven state, similar to Section 4.3. We create a network with $N = 1000$ nodes, each node initially belonging to one out of four states $z_i \in \mathbb{N}_4 = \{0, 1, 2, 3\}$. The nodes change their state according to a Markov process, i.e., there exists a transition matrix $P \in \mathbb{R}^{4\times 4}$ with entries p_{ij} giving the probabilities

$$p_{ij} \doteq \mathbb{P}\left[z_i(t+1) = j \mid z_i(t) = i\right].$$

Thus, after updating their state, they generate $B \times M$ random values (all except M equal to zero), according to equation (4.24). After the max consensus step, each node can estimate $\widehat{n}_b$, $b \in \mathbb{N}_4$, where we replace equation (4.26) with the regularization based estimator of equation (4.6), and finally compute the dynamic PMF estimate through equation (4.27).

In Figure 4.12 we evaluate the regularization based dynamic PMF estimator ($\gamma = 0.005$) against the point-wise PMF estimator ($\gamma = 0$), both using the same realization of random samples, and a small $M = 10$. In the regularization based estimator we only use one step memory, $\tau = 0$, $\eta = 1$.

Since the number of samples are relatively small (only $M = 10$, compared to Figures 4.10 and 4.11), the variance in the estimates are quite significant, but the regularization based estimator does successfully reduce the variance and yields an improved estimate of the true distribution.

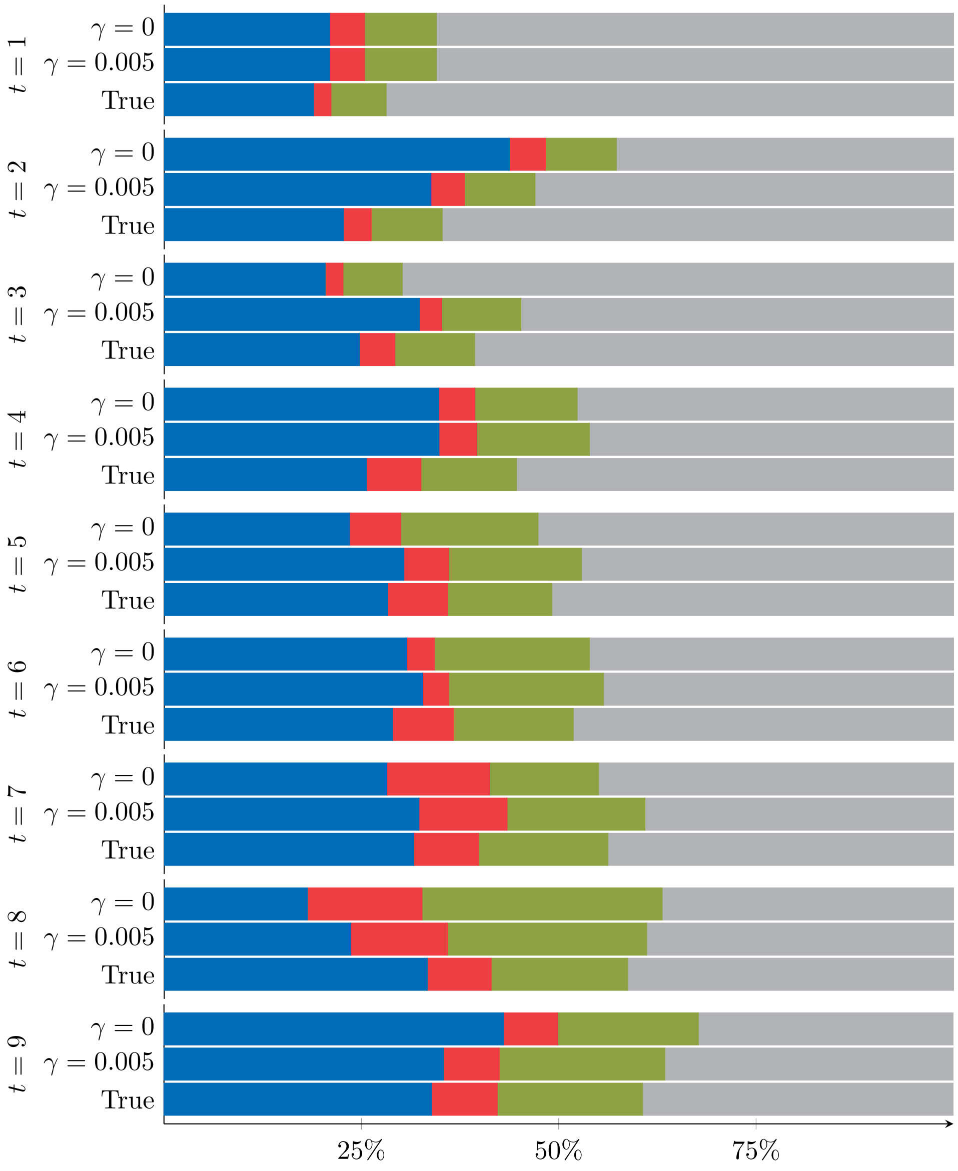


Figure 4.12: Comparing the regularization based ($\gamma = 0.005$) and the point-wise ($\gamma = 0$) PMF estimator against the true dynamic distribution for four states, represented with different colors. In the first step (top figure), both estimators yield the same result since the regularization based estimator has not yet initialized its memory, but in the following steps it reduces the variance compared to the point-wise estimator.

4.10 Conclusions

In this chapter, the max consensus protocol was used to derive two specific estimators, one which estimates the network size in time dependent networks, and one which estimates PMFs. Finally, we showed how these estimators could be combined to estimate time dependent PMFs.

One of the main advantages with the max consensus protocol compared to other strategies is that it has the fastest possible convergence time, and it is also very easy to implement, even in networks with unreliable communication. It does neither require any global information nor global identities among the nodes, so it is a suitable choice for anonymous networks. The main disadvantage of the max consensus strategy is that it does not converge to the exact value, but is instead based on probabilistic estimation schemes. However, its accuracy can be improved through increasing the packet sizes.

The network size estimation of anonymous dynamical networks extended the static network estimation strategy through a regularization term, which penalizes hypotheses conflicting with a-priori assumptions on the network's behavior. We explicitly considered and characterized the class of quadratic regularization terms, which resulted in a closed-form estimator that corresponds to a nonlinear smoother.

Two distributed estimators of empirical PMFs were considered: one based on max consensus and one based on average consensus. The main difference is that the average consensus based estimator converges asymptotically in time, while the max consensus based estimator converges in finite time, but not to the exact value. The accuracy of the max consensus based estimator can be improved through increasing the packet size, which means that it will typically use larger packets than the average consensus based estimator. However, our experiments showed that if estimation speed is important, and in particular if there is a spatial correlation among the nodes' initial values, then the max consensus based estimator outperforms the average consensus based estimator.

The algorithms were derived using some simplifying assumptions, in particular, reliable communications and infinite numerical precision. We showed however, with numerical experiments, that quantization effects seem to play a minor role, especially, representing the numbers with just a few bits is enough even for networks of hundreds of agents. Furthermore, we remark that if the max consensus protocol does not reach consensus, it would imply that the network is, at least temporarily, not strongly connected. The estimation process would still succeed in the estimation of the reachable subset of nodes, which is an interesting extension.

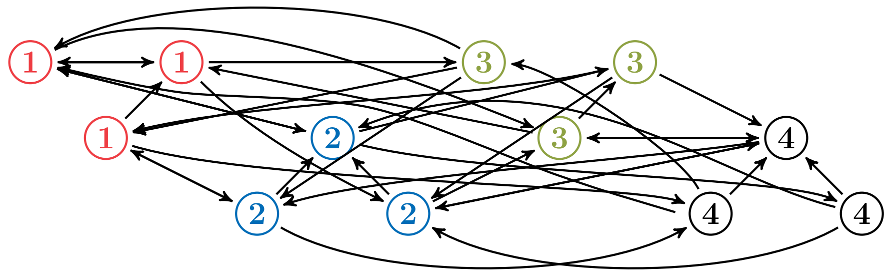
Chapter 5

Topology Convergence in Peer-to-Peer Networks

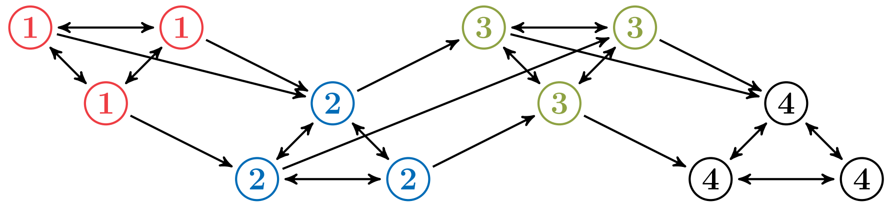
"We've heard that a million monkeys at a million keyboards could produce the complete works of Shakespeare; now, thanks to the Internet, we know that is not true."
— Robert Wilensky

In this chapter, we investigate a peer-to-peer (P2P) network for efficient live-streaming television, inspired by gradienTv and Sepidar [Payberah et al., 2010a,b]. The goal of this application is to distribute a data stream from a small set of seed nodes to every other node in the network, and the problem is to design distributed algorithms for creating an efficient overlay network topology. In particular, this chapter deals with the convergence problem, in which the network graph converges to a complete *gradient overlay network*. The gradient topologies are fundamental in self-organizing systems, and generalize the rooted trees topologies. The contribution of this chapter is the convergence analysis of the given algorithm, including convergence rate estimates, and the derivation of a threshold on the churn rate for a gradient topology to emerge.

The outline of this chapter is as follows: In Section 5.1 we introduce the network model and topology convergence problem, and in Section 5.2 we give necessary and sufficient conditions for convergence. In Section 5.3, we study the convergence rate for the system, and in Section 5.4 we study convergence properties when the network is subject to churn. In Section 5.5 we simulate the construction of a gradient topology using the model in Section 5.1, and in Section 5.6 we evaluate the live-streaming performance in a real P2P application using the gradient overlay topology. Finally, Section 5.7 concludes this chapter.



(a) The initial random overlay network



(b) The network after converging to a complete gradient topology

Figure 5.1: The gradient network is described as a directed graph. The nodes are labeled with their respective utility value, and the edges from the similar neighbor sets are shown. In the gradient topology, paths of increasing utilities emerge.

5.1 The Gradient Topology Problem

An *overlay network* is a virtual network built on top of another network. Here, it denotes the P2P network topology built for television streaming over the Internet. The *gradient topology* belongs to the class of gossip-generated overlay networks that are built from a random overlay network through symmetry breaking using a preference function. Thus, we are given a node set $\mathcal{V} = \{1, \dots, N\}$, and need to select directed edges $\mathcal{E}$ to construct our network $\mathcal{G}(\mathcal{V}, \mathcal{E})$.

In the live streaming application, the idea is to utilize the nodes in the P2P network to aid in the content distribution, but since the peers are heterogeneous, not all peers will be equally useful. Thus, we classify each node $i \in \mathcal{V}$ with its *utility value* $u_i \in \mathbb{R}$, which captures, for example, the node's upload capacity, latency and reliability for the P2P network.

A gradient topology is an overlay network satisfying that, for any two nodes v_1 and v_2 with utility values u_{v_1} and u_{v_2}, if $u_{v_1} \geq u_{v_2}$ then $\operatorname{dist}(v_1, v_\star) \leq \operatorname{dist}(v_2, v_\star)$, where $v_\star$ is a node with highest utility in the system and $\operatorname{dist}(\cdot, \cdot)$ is the length of the shortest path between the nodes in the network [Sacha, 2009]. In other words, nodes with a higher utility value should be closer to the seed nodes compared to nodes with a lower utility value, so that gradient paths of increasing utilities emerge in the system, see Figure 5.1.

In constructing the gradient overlay, the nodes $i \in \mathcal{V}$ build two sets of neighbors: a *similar view* $\mathcal{N}_i^s$ and a *random view* $\mathcal{N}_i^r$. For the similar view, nodes prefer neighbors with close but slightly higher utility values, while the random view is used to sample new nodes with uniform probability for possible inclusion in the similar view. Thus, node i's neighbors are $\mathcal{N}_i = \mathcal{N}_i^s \cup \mathcal{N}_i^r$.

Each node i defines a *preference function* $>_i$ over its neighbors, where node i is said to prefer node a over node b (denoted by $a >_i b$) if

$$\begin{array}{ll} u_a \geq u_i > u_b & \text{or if} \\ |u_a - u_i| < |u_b - u_i| & \text{when } u_a, u_b > u_i \text{ or } u_a, u_b < u_i. \end{array}$$

Further, let $\min \mathcal{N}_i^s$ denote node i's least preferred neighbor in its similar neighbor set.

For any given initial overlay network, the topology is evolving through each node i at each time t updating its own neighbor set $\mathcal{N}_i(t)$ independently of the other nodes according to Algorithm 5.1. The algorithm can be summarized as sampling random nodes from the network, and if the random node is preferred over the least preferred node in the similar set, then those two neighbors are exchanged.

Algorithm 5.1 Topology Dynamics

```
1: for each node $i \in \mathcal{V}$ do
2:     for every $t = 1, 2, 3, \ldots$ do
3:         Let $\mathcal{N}_i^r(t) = \{j\}$, where $j \in \mathcal{V}$ is a randomly selected node with uniform
   probability $p_t$, $0 < Np_t < 1$. Otherwise $\mathcal{N}_i^r(t) = \emptyset$.
4:         if $\mathcal{N}_i^r(t) \neq \emptyset$ then
5:             if $j \notin \mathcal{N}_i^s(t-1)$ and $j >_i \min \mathcal{N}_i^s(t-1)$ then
6:                 $\mathcal{N}_i^s(t) = \mathcal{N}_i^s(t-1) \cup \{j\} \setminus \{ \min \mathcal{N}_i^s(t-1) \}$
7:             else
8:                 $\mathcal{N}_i^s(t) = \mathcal{N}_i^s(t-1)$
9:             end if
10:        end if
11:    end for
12: end for
```

It is assumed that the node out-degree $d_i(t) \doteq |\mathcal{N}_i^s(t)| = d_i$ stays constant throughout the algorithm. Note that the sampling probabilities p_t are time dependent and govern whether the random neighbor set $\mathcal{N}_i^r(t)$ is empty (with probability $1 - Np_t$). The reason for this is that a node can adapt its sampling frequency to minimize the network overhead for building a gradient topology, and typically samples more frequently just after joining the network to improve its neighbor sets, and then lowering its sampling rate when the neighbors have stabilized. Notice also that Algorithm 5.1 can be run asynchronously on the nodes.

Remark. Note that no constraint is enforced on the in-degree of the nodes. However, in Section 5.6, the gradient topology is used for sampling nodes for a second auction algorithm, which limits the in-degree for the information dissemination network.

The preference function $>_i$ induced a partial order on the nodes $\mathcal{V}$. In order to study the system topology convergence to a gradient structure with the proposed algorithm, we let Λ_i denote the set of optimal similar neighbor sets for node i, i.e., $\forall \widehat{N} \in \Lambda_i$ there are no nodes $j \in \widehat{N}$ and $k \in \mathcal{V} \backslash \widehat{N}$ such that $k >_i j$. Notice that there could be multiple optimal neighbor sets.

For every node $i \in \mathcal{V}$, we define $X_i(t)$ as a counter for the number of non-optimal neighbors in i's similar neighbor set,

$$X_i(t) \doteq d_i - \max_{\widehat{N} \in \Lambda_i} \left| \mathcal{N}_i^s(t) \cap \widehat{N} \right| .$$

Notice that $X_i(t)$ is monotonically decreasing under Algorithm 5.1 since an optimal neighbor will never be removed from the similar neighbor set $\mathcal{N}_i^s(t)$.

Let $\mathcal{G}(t)$ be the graphs generated by Algorithm 5.1. Then we give the definition of gradient topology convergence as follows (see Figure 5.1).

Definition 5.1. $\mathcal{G}(t)$ is said to converge to a gradient topology if

$$\lim_{t \to \infty} X_i(t) = 0$$

for all nodes $i \in \mathcal{V}$.

5.2 Convergence Analysis

Since each node updates its neighbor set independently, the analysis can be carried out separately for each $X_i(t)$. We therefore simplify the notations in the following discussion, and let $X(t)$ represents $X_i(t)$ for an arbitrary node $i \in \mathcal{V}$.

Denote the maximum degree $D = \max_i\{d_i\}$, then $X(0) = D$ would be the worst possible initial condition. Furthermore, $X(t)$ decreases precisely when the randomly sampled node is a new optimal neighbor, and the probability of this event occurring is minimal when the optimal solution is unique, and then the probability is equal to

$$\mathbb{P}\left[X(t+1) = k-1 \mid X(t) = k\right] = kp_t, \quad k = 1, \dots, D, \tag{5.1}$$

where k is the number of non-optimal neighbors.

In the following theorem we propose a necessary and sufficient condition for the probabilities p_t for almost sure convergence of Algorithm 5.1.

Theorem 5.1. *The graph generated by Algorithm 5.1 converges to a gradient topology* $(X(t) = 0)$ *with probability 1 if and only if*

$$\lim_{T \to \infty} \prod_{t=0}^{T} (1 - p_t) = 0.$$

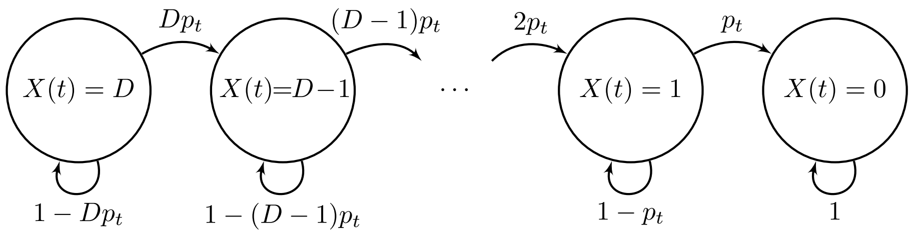


Figure 5.2: Markov chain for the state evolution of a single node.

Before proving Theorem 5.1, let us take a closer look at Algorithm 5.1, and notice especially that the stochastic process in equation (5.1) for $X(t)$ has the Markov property, hence we can describe it as a simple Markov chain, see Figure 5.2.

Let $\boldsymbol{\pi}(t)$ denote the row vector of probabilities for the states $X(t)$, i.e.,

$$\pi_i(t) = \mathbb{P}\left[X(t) = D - i\right].$$

Remark. In this chapter, we are using a zero-based indexing for $\boldsymbol{\pi}$, i.e., $\boldsymbol{\pi} = [\pi_0, \dots, \pi_D]$ for notational simplicity.

The evolution of $\boldsymbol{\pi}(t)$ can be written in matrix form as

$$\boldsymbol{\pi}(t+1) = \boldsymbol{\pi}(t)P_t, \tag{5.2}$$

where P_t is the transition matrix at time t,

$$P_t = \begin{bmatrix} 1-Dp_t & Dp_t & 0 & \cdots & 0 & 0 \\ 0 & 1-(D-1)p_t & (D-1)p_t & \cdots & 0 & 0 \\ 0 & 0 & 1-(D-2)p_t & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 1-p_t & p_t \\ 0 & 0 & 0 & \cdots & 0 & 1 \end{bmatrix}.$$

Since P_t is a triangular matrix, the eigenvalues are given by the diagonal elements, i.e., the eigenvalues of P_t are $\lambda_i(t) = 1-(D-i)p_t$, $i = 0, \dots, D$. Notice that there is a single unit eigenvalue $\lambda_D(t) = 1$, and all other eigenvalues are strictly less than one. Furthermore, all eigenvalues are distinct, hence the eigenvectors form a basis for $\mathbb{R}^{D+1}$. In the following lemma, we characterize the eigenvectors.

Lemma 5.2. *The eigenvector $\xi^i(t)$ corresponding to eigenvalue $\lambda_i(t)$ is independent of $p_t \neq 0$, for $i = 0, \dots, D$.*

Proof. The (left-)eigenvectors of P_t satisfy $\lambda_i(t)\xi^i(t) = \xi^i(t)P_t$. Let $\xi^i_j(t)$ denote the j:th component of $\xi^i(t)$, then, by inspection of the matrix P_t, we have the system of equations

$$\begin{aligned}
(1-(D-i)p_t)\,\xi_0^i(t) &= (1-Dp_t)\,\xi_0^i(t) \\
(1-(D-i)p_t)\,\xi_j^i(t) &= (1-(D-j)p_t)\,\xi_j^i(t) \\
&\quad + (D-j+1)p_t\xi_{j-1}^i(t) \qquad\qquad j=1,\dots,D,
\end{aligned}$$

which is equivalent to

$$\begin{aligned}
i\xi_0^i(t) &= 0 \\
(i-j)\xi_j^i(t) &= (D-j+1)\xi_{j-1}^i(t) \qquad\qquad j=1,\dots,D,
\end{aligned}$$

or further

$$\begin{aligned}
\xi_j^i(t) &= 0 && \text{if } j<i \\
\xi_j^i(t)\frac{i-j}{D-j+1} &= \xi_{j-1}^i(t) && \text{if } j>i
\end{aligned} \tag{5.3}$$

where $\xi_i^i(t)$ can be chosen as an arbitrary non-zero value, as a scaling factor for the eigenvector. From equation (5.3) it is evident that the eigenvectors are independent of p_t. □

An important consequence of Lemma 5.2 is that all P_t, independent of t, have the same eigenvectors, and are thus simultaneously diagonalizable. Hence we can simplify the notation by dropping the parameter t from the eigenvectors ξ^i.

Let us now return to the initial probability distribution $\boldsymbol{\pi}(0)$, and decompose it into the eigenvector basis as

$$\boldsymbol{\pi}(0) = \sum_{i=0}^{D} \alpha_i \xi^i, \tag{5.4}$$

for some real numbers α_i.

Lemma 5.3. *In the decomposition of the initial probability distribution $\boldsymbol{\pi}(0)$ into the eigenvector basis, we have $\alpha_D\xi^D = \mathbf{e}_D$, where $\mathbf{e}_i$ is the standard basis $\mathbf{e}_i = [0,\dots,0,\underbrace{1}_{i:th},0,\dots,0]^T$.*

Proof. Let us consider $\xi^i\mathbb{1}$ for $i=0,\dots,D-1$. By equation (5.3),

$$\xi^i\mathbb{1} = \sum_{j=0}^{D}\xi_j^i = \sum_{j=i}^{D}\xi_j^i = \sum_{j=0}^{D-i}\xi_{i+j}^i.$$

We will show by induction that

$$\sum_{j=0}^{k} \xi^i_{i+j} = \frac{D-i-k}{D-i} \xi^i_{i+k}. \tag{5.5}$$

The case when $k = 0$ is clearly true. Thus, assume that equation (5.5) holds for k and consider the case $k + 1$,

$$\begin{aligned} \sum_{j=0}^{k+1} \xi^i_{i+j} &= \sum_{j=0}^{k} \xi^i_{i+j} + \xi^i_{i+k+1} \\ &= \frac{D-i-k}{D-i} \xi^i_{i+k} + \xi^i_{i+k+1} \\ &= \frac{D-i-k}{D-i} \frac{-(k+1)}{D-i-k} \xi^i_{i+k+1} + \xi^i_{i+k+1} \\ &= \frac{D-i-(k+1)}{D-i} \xi^i_{i+k+1}. \end{aligned}$$

Using equation (5.5) implies that $\xi^i \mathbb{1} = 0$, $i = 0, \ldots, D-1$, and thus, $\boldsymbol{\pi}(0)\mathbb{1} = \alpha_D \xi^D \mathbb{1}$. Now, since $\boldsymbol{\pi}(0)$ is a probability distribution, we know that $\boldsymbol{\pi}(0)\mathbb{1} = 1$, but equation (5.3) tells us that only the last component of ξ^D is non-zero, hence the lemma follows. □

We are now ready to prove the main theorem.

Proof of Theorem 5.1. The almost sure convergence to a gradient topology, by Definition 5.1, can be expressed as

$$\lim_{T\to\infty} \mathbb{P}\left[X(T) = 0\right] = 1,$$

or, equivalently for the probability vector,

$$\lim_{T\to\infty} \boldsymbol{\pi}(T) = \mathbf{e}_D.$$

Equations (5.2) and (5.4) give us

$$\begin{aligned} \boldsymbol{\pi}(T) = \boldsymbol{\pi}(0) \prod_{t=0}^{T-1} P_t \\ &= \sum_{i=0}^{D} \alpha_i \xi^i \prod_{t=0}^{T-1} P_t \\ &= \sum_{i=0}^{D} \alpha_i \xi^i \prod_{t=0}^{T-1} \lambda_i(t) \\ &= \sum_{i=0}^{D-1} \alpha_i \xi^i \prod_{t=0}^{T-1} \lambda_i(t) + \mathbf{e}_D. \end{aligned} \tag{5.6}$$

Consider the limit

$$\lim_{T\to\infty} |\boldsymbol{\pi}(T) - \mathbf{e}_D| = \lim_{T\to\infty} \left| \sum_{i=0}^{D-1} \alpha_i \xi^i \prod_{t=0}^{T-1} \lambda_i(t) \right|$$
$$\leq \sum_{i=0}^{D-1} \left|\alpha_i \xi^i\right| \cdot \lim_{T\to\infty} \prod_{t=0}^{T-1} (1 - p_t).$$

Clearly, the right-hand side vanishes if $\lim_{T\to\infty} \prod_{t=0}^{T} (1 - p_t) = 0$. This proves the sufficiency part of the theorem.

Furthermore, for the necessity part, note that the set of initial probability distributions spawns $\mathbb{R}^{D+1}$. Thus, an initial probability distribution $\boldsymbol{\pi}(0)$ exists such that $\alpha_{D-1} \neq 0$. Assume that the limit $\lim_{T\to\infty} \prod_{t=0}^{T} (1 - p_t) = c > 0$ is strictly positive (the limit exists since it is a monotone bounded sequence), then

$$\lim_{T\to\infty} |\boldsymbol{\pi}(T) - \mathbf{e}_D| = \left| \sum_{i=0}^{D-2} \alpha_i \xi^i \left(\lim_{T\to\infty} \prod_{t=0}^{T-1} \lambda_i(t) \right) + c\alpha_{D-1} \xi^{D-1} \right| > 0, \tag{5.7}$$

since the eigenvectors are linearly independent. Thus, we have proven the theorem. □

Corollary 5.1. *The graph generated by Algorithm 5.1 converges to a gradient topology with probability 1 if and only if*

$$\lim_{T\to\infty} \sum_{t=0}^{T} p_t = \infty.$$

Proof. This follows directly from Theorem 5.1, and the relation

$$\lim_{T\to\infty} \prod_{t=0}^{T} (1 - p_t) = 0 \quad \Leftrightarrow \quad \lim_{T\to\infty} \sum_{t=0}^{T} p_t = \infty$$

for $0 < p_t < 1$. □

Remark. Corollary 5.1 can be interpreted such that the network converges to a gradient topology if and only if each node continues searching for its optimal neighbors for an expected infinite number of times.

5.3 Convergence Rate Estimation

We will now investigate the convergence rate of $X(t)$, with a constant sampling probability $p_t = p$. Define the stochastic variable T_i as the time when $X(t)$ reaches 0, when starting at $X(0) = i$,

$$T_i = \inf_t [X(t) = 0 \mid X(0) = i].$$

Further, let $M_i = \mathbb{E}[T_i]$ denote the expected convergence time when starting at $X(0) = i$. Clearly $M_0 = 0$, and for $i = 1, \dots, D$ we have the recursion

$$\begin{aligned} M_i &= 1 + \mathbb{P}[X(t+1) = i-1 \mid X(t) = i] \cdot M_{i-1} \\ &\quad + \mathbb{P}[X(t+1) = i \mid X(t) = i] \cdot M_i \\ &= 1 + ipM_{i-1} + (1-ip)M_i, \end{aligned}$$

which can be further simplified to

$$M_i = \frac{1 + ipM_{i-1}}{ip} = \frac{1}{ip} + M_{i-1}.$$

By continuing with induction, we can sum up the expected convergence time as

$$M_i = \frac{1}{p} \sum_{d=1}^{i} \frac{1}{d}.$$

The worst initial case is when $X(0) = D$, where the expected convergence time is

$$M_D = \frac{1}{p} \sum_{d=1}^{D} \frac{1}{d} \leq \frac{1 + \ln(D)}{p}. \tag{5.8}$$

Remark. M_D is the expected time for an individual node's neighbor set to converge, not the expected time for *all* nodes to converge to a gradient topology. As such, it provides a lower bound on the convergence time. In the next section, we will consider the global network convergence problem.

Global Convergence Rate

In this section, we will analyze the asymptotic convergence rate for the entire network to a gradient topology, in contrast to the analysis of a single node in the previous section. We continue assuming a constant sampling probability ($p_t = p$, $P_t = P$), thus the probability distribution for a single node in equation (5.6) is simplified to

$$\boldsymbol{\pi}(t) = \boldsymbol{\pi}(0)P^t = \sum_{i=0}^{D-1} \alpha_i \xi^i \lambda_i^t + \mathbf{e}_D,$$

where $\lambda_i = 1 - (D-i)p$, $i = 0, \dots, D$. The probability distribution for a single node approaches the gradient topology state $\mathbf{e}_D$ asymptotically as $1 - \mathcal{O}\left(\lambda_{D-1}^t\right)$, where $\lambda_{D-1} = 1 - p$ is the second largest eigenvalue of P.

Here, we will study how the entire network convergence is affected by the network size N, and to this end we consider a continuous-time approximation with a system for which the probability of being in the target state is $1 - \lambda^t$ at time t (where $\lambda = 1 - p$).

Theorem 5.4. *The expected global convergence time to a gradient topology for N nodes is*

$$-\frac{1}{\log(\lambda)}\sum_{n=1}^{N}\frac{1}{n},$$

where the nodes are modeled by i.i.d. processes, whose individual probability distribution is given by $1-\lambda^t$.

Proof. The probability for the entire system of N i.i.d. processes to be in the target state at time t is $\phi \doteq (1-\lambda^t)^N$. Notice that the probability for the system to reach the target state at time t is given by the derivative $\frac{\mathrm{d}\phi}{\mathrm{d}t} = -N(1-\lambda^t)^{N-1}\lambda^t\log(\lambda)$. The expected convergence time, i.e., the time to reach the gradient topology, is computed by

$$\int_0^\infty t\cdot\frac{\mathrm{d}\phi}{\mathrm{d}t}\,\mathrm{d}t = -N\log(\lambda)\int_0^\infty t(1-\lambda^t)^{N-1}\lambda^t\,\mathrm{d}t.$$

Using a variable substitution $x=\lambda^t$, this integral can be rewritten as

$$\int_0^\infty t(1-\lambda^t)^{N-1}\lambda^t\,\mathrm{d}t = -\frac{1}{\log(\lambda)^2}\int_0^1 \log(x)\,(1-x)^{N-1}\,\mathrm{d}x.$$

Recall that this integral is equal to [Devoto and Duke, 1984]

$$\int_0^1 \log(x)\,(1-x)^{N-1}\,\mathrm{d}x = -\frac{1}{N}\sum_{n=1}^{N}\frac{1}{n},$$

hence, the expected convergence time is

$$\int_0^\infty t\cdot\frac{\mathrm{d}\phi}{\mathrm{d}t}\,\mathrm{d}t = -\frac{1}{\log(\lambda)}\sum_{n=1}^{N}\frac{1}{n}. \tag{5.9}$$

□

Remark. Notice that $-\frac{1}{\log(\lambda)} = -\frac{1}{\log(1-p)} \approx \frac{1}{p}$ for small p, thus this is in agreement with the upper bound in equation (5.8).

Remark. The convergence time scales asymptotically as $\mathcal{O}(\log(N))$ for large network sizes N, since $\sum_{n=1}^{N}\frac{1}{n} < 1+\log(N)$.

5.4 Convergence Rate with Network Churn

In this section we consider the topology convergence to a gradient topology when the system is subject to churn, i.e., the nodes are changing over time. We model the churn as a probability $\epsilon > 0$ that a node will be replaced with a new node, that

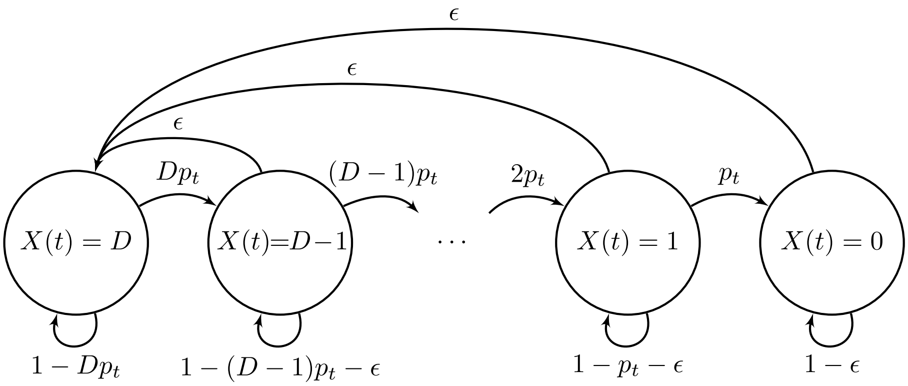


Figure 5.3: Markov chain for the state evolution of a single node with churn.

is starting from state $X = D$. The corresponding Markov chain for a single node is illustrated in Figure 5.3.
The corresponding transition matrix P for the Markov chain $\boldsymbol{\pi}(t+1) = \boldsymbol{\pi}(t)P$ is

$$P = \begin{bmatrix} 1-Dp & Dp & 0 & \cdots & 0 & 0 \\ \epsilon & 1-(D-1)p-\epsilon & (D-1)p & \cdots & 0 & 0 \\ \epsilon & 0 & 1-(D-2)p-\epsilon & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ \epsilon & 0 & 0 & \cdots & 1-p-\epsilon & p \\ \epsilon & 0 & 0 & \cdots & 0 & 1-\epsilon \end{bmatrix}.$$

Assuming that $0 < \epsilon$, and also $0 < Dp < 1$ and $(D-1)p + \epsilon < 1$, we have the following theorem characterizing the stationary distribution.

Theorem 5.5. *The Markov chain in Figure 5.3, describing the stochastic node process with churn, has a unique stationary distribution $\boldsymbol{\pi}$, which satisfies*

$$\pi_0 = \frac{\epsilon}{Dp + \epsilon}$$

and

$$\pi_i = \frac{(D-i+1)p}{(D-i)p+\epsilon}\pi_{i-1} \quad i = 1, \ldots, D.$$

Proof. Notice that the Markov chain is finite ($D+1$ states), irreducible (every state can be reached from any other state) and aperiodic (because of the self-loops), thus it has a unique stationary distribution corresponding to the eigenvalue 1.

Consider now the stationary distribution $\boldsymbol{\pi}$ satisfying $\boldsymbol{\pi} = \boldsymbol{\pi}P$, and especially for column $i = 1, \ldots, D$ we have

$$\pi_i = (1 - (D-i)p - \epsilon)\pi_i + (D-i+1)p\pi_{i-1},$$

that is,

$$\pi_i = \frac{(D-i+1)p}{(D-i)p+\epsilon}\pi_{i-1}.$$

Next, let us show the following property for the partial sum $\pi_d + \cdots \pi_D$ through induction:

$$\sum_{i=d}^{D} \pi_i = (D-d+1)\frac{p}{\epsilon}\pi_{d-1} \quad d = 1, \ldots D.$$

The case $d = D$ follows directly from the previous recursion. Let us continue with the induction step:

$$\begin{aligned}\sum_{i=d}^{D} \pi_i = \pi_d + \sum_{i=d+1}^{D} \pi_i = \pi_d + (D-d)\frac{p}{\epsilon}\pi_d = \frac{(D-d)p+\epsilon}{\epsilon}\pi_d \\ = \frac{(D-d)p+\epsilon}{\epsilon}\frac{(D-d+1)p}{(D-d)p+\epsilon}\pi_{d-1} = (D-d+1)\frac{p}{\epsilon}\pi_{d-1}.\end{aligned}$$

First, we use this to validate that the eigenvector satisfies $\boldsymbol{\pi} = \boldsymbol{\pi}P$ for the first column,

$$\pi_0 = (1-Dp)\pi_0 + \epsilon\sum_{d=1}^{D} = (1-Dp)\pi_0 + \epsilon D\frac{p}{\epsilon}\pi_0 = \pi_0.$$

Second, the stationary probability distribution should be normalized such that

$$\sum_{i=0}^{D} \pi_i = 1,$$

thus

$$\sum_{i=0}^{D} \pi_i = \pi_0 \sum_{i=1}^{D} \pi_i = \pi_0 + D\frac{p}{\epsilon}\pi_0 = \frac{Dp+\epsilon}{\epsilon}\pi_0$$

or

$$\pi_0 = \frac{\epsilon}{Dp+\epsilon},$$

which proves the theorem. □

Remark. Theorem 5.5 shows that if $p = \epsilon$, then the stationary distribution is uniform with $\pi_i = \frac{1}{D+1}$, $i = 0, \dots, D$, thus a node is equally likely to be in any state. When $p > \epsilon$, the nodes are more likely to be in the later states, i.e., closer to a gradient topology, and when $p < \epsilon$ the nodes are more likely to be in the earlier states, i.e., having a random neighbor set. Thus, we conclude that for a gradient topology to appear, it is necessary for the sampling probability p to be greater than the churn rate ϵ.

Next, the convergence speed will be considered through analyzing the second largest eigenvalue of the transition matrix P.

Lemma 5.6. *The asymptotic convergence time for the entire network with churn is*

$$\log(N)\,\frac{1}{p+\epsilon}.$$

Proof. It is straightforward to verify that the remaining eigenvalues of P (less than one) are

$$\lambda_i = 1 - (D-i)p - \epsilon,$$

for $i = 0, \dots, D-1$, with the corresponding eigenvectors

$$\xi_i = \left[\underbrace{0, \dots, 0}_{i}, (-1)^0 \binom{D-i}{0}, (-1)^1 \binom{D-i}{1}, \dots, (-1)^{D-i} \binom{D-i}{D-i}\right].$$

Hence, the second largest eigenvalue of P is $\lambda_{D-1} = 1 - p - \epsilon$.

Using Theorem 5.4, with $\lambda = 1 - p - \epsilon$, and the approximations $\sum_{n=1}^{N} \frac{1}{n} \approx \log(N)$ and $-\frac{1}{\log(\lambda)} = -\frac{1}{\log(1-p-\epsilon)} \approx \frac{1}{p+\epsilon}$ proves this lemma. □

Remark. A larger ϵ will yield a faster convergence rate, but to a steady state solution further from the complete gradient topology.

5.5 Convergence Simulation

Here we examine the convergence rate of Algorithm 5.1 using numerical simulations, and compare the outcome with our theoretical results. We start with a network consisting of $N = 100$ nodes, where the degree of each node is $D = d_i = 10$. The similar view $\mathcal{N}_i^s(0)$ is initialized with D nodes uniformly chosen among all nodes in the network, and the sampling probability p_t is held at a constant value of $\frac{1}{2N}$. Hence, for each node and at each iteration of the algorithm, the random view is empty with 50 % probability. The state trajectories for all nodes are shown in Figure 5.4, and the convergence times ranges from 193 to 1116 iterations, with an average convergence time of 554 iterations. The convergence time can be compared to the expected convergence time given by equation (5.8) for a single node, i.e., 585 iterations, and the global convergence rate given by equation (5.9), i.e., 1035 iterations. A corresponding heat map of the states are shown in Figure 5.5. The system converges to a gradient topology, as guaranteed by Theorem 5.1.

In the second simulation we change the sampling probability into a decaying probability $p_t = \frac{1}{N}\frac{1}{(1+t/100)^2}$. Notice that $\sum_{t=0}^{\infty} Np_t < 101$, hence, by Corollary 5.1, there is a positive probability that the algorithm does not converge to a gradient topology. This is also confirmed by the simulation trajectories in Figure 5.6 and the corresponding state heat map in Figure 5.7.

In the third simulation, we return to the constant sampling probability $\frac{1}{2N}$, but consider a network with $N = 500$ nodes and a node degree of $D = 50$. The expected state heat map is shown in Figure 5.8, and the convergence time can be compared to the expected convergence time of 4499 iterations for a single node and 6789 iterations for the entire network.

Finally, we simulate the influence of churn on the network. Consider a network consisting of $N = 100$ nodes, with node degree $D = 10$ and a constant sampling probability $p_t = \frac{1}{2N}$. In Figure 5.9, the churn rate is $\epsilon = \frac{1}{2}p_t$, and we see that nodes tend to favor states closer to a gradient topology as predicted by Theorem 5.5. In fact, 27% of the nodes are in their optimal state $X(t) = 0$, with another 14% are in state $X(t) = 1$. In Figure 5.10, the churn rate is increased to $\epsilon = p_t$, and all states are equally likely in the steady state solution, while in Figure 5.11 the churn rate is further increased to $\epsilon = 2p_t$ and nodes tend to have a more random neighborhood, with 17% of the nodes having a completely random neighborhood $X(t) = D$.

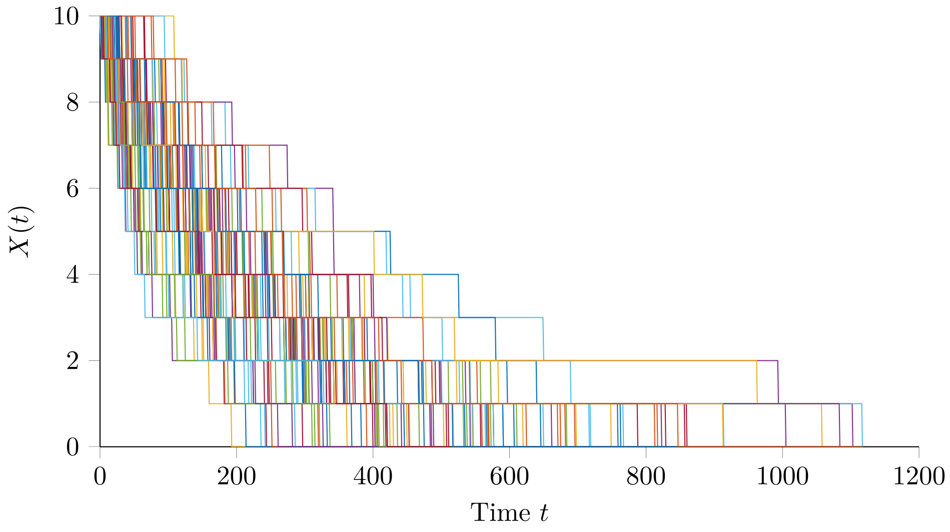


Figure 5.4: State trajectories for a network with 100 nodes and degree $D = 10$. Each line represents a single node, and a constant sampling probability $Np_t = 1/2$ is used. The network converges to a gradient topology.

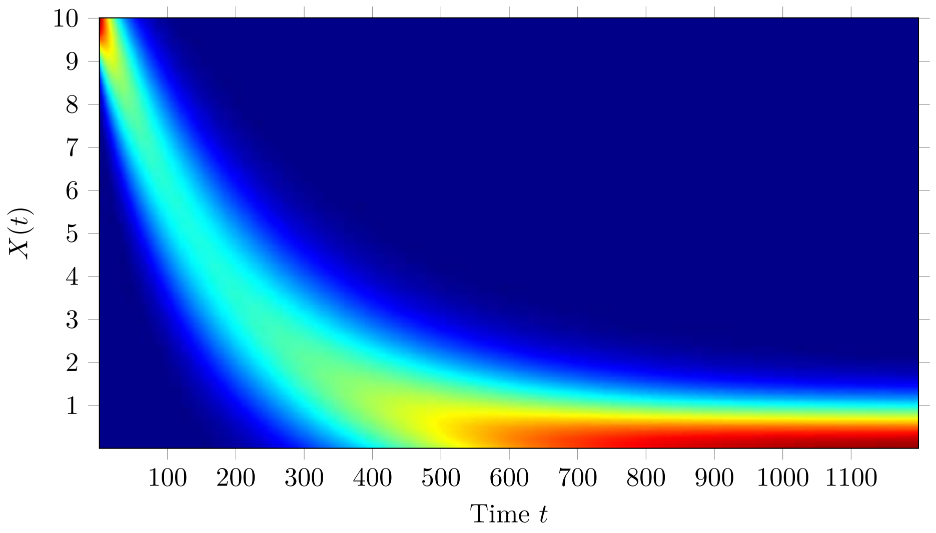


Figure 5.5: State heat map for a network with 100 nodes, degree $D = 10$, and constant sampling probability $Np_t = 1/2$. Brighter colors indicate more likely states.

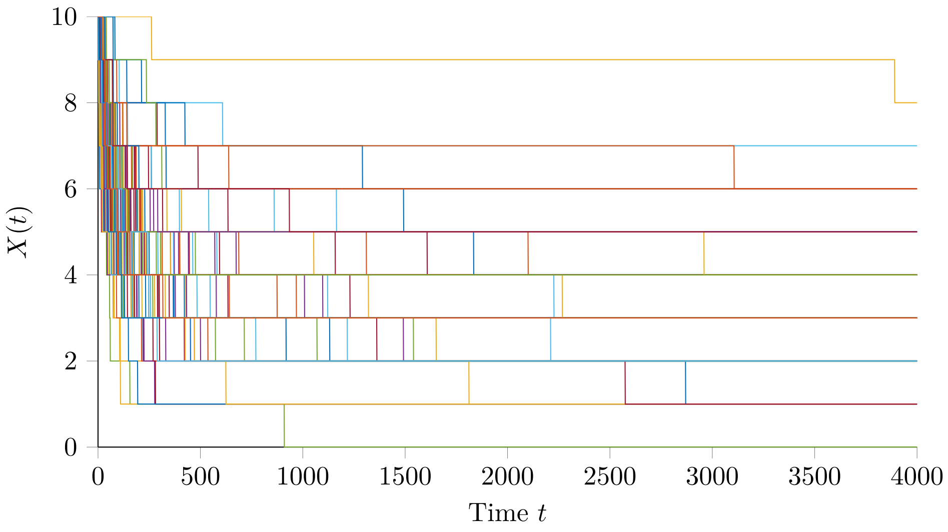


Figure 5.6: State trajectories for a network with 100 nodes and degree $D = 10$. Each line represents a single node, and a decaying sampling probability $Np_t = \frac{1}{(1+t/100)^2}$ is used. The network does not converge to a gradient topology.

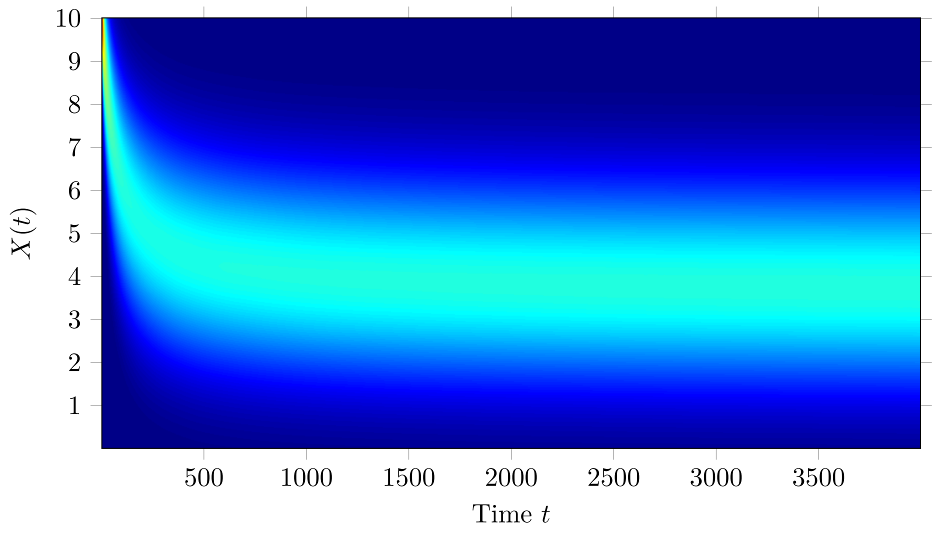


Figure 5.7: State heat map for a network with 100 nodes, degree $D = 10$, and a decaying sampling probability $Np_t = \frac{1}{(1+t/100)^2}$. Brighter colors indicate more likely states. The network does not converge to a gradient topology.

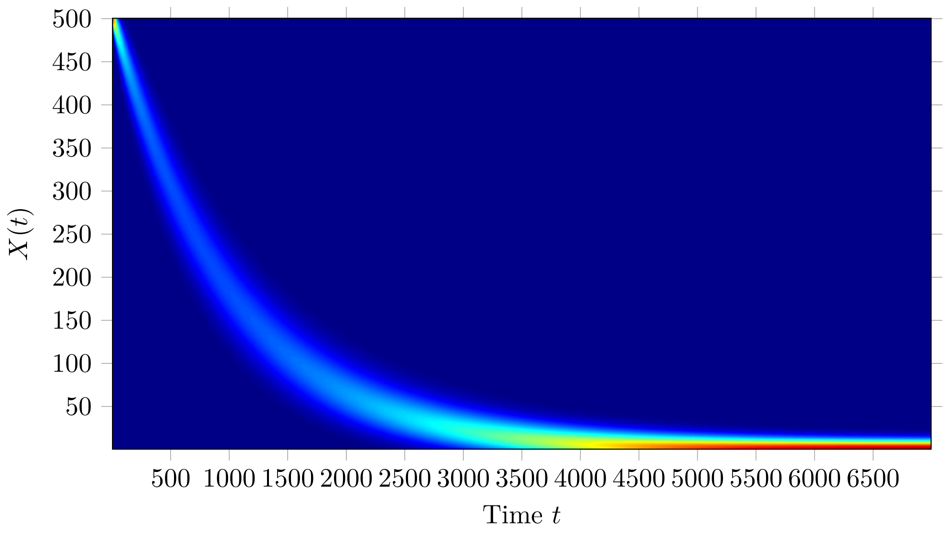


Figure 5.8: State heat map for a network with 500 nodes, degree $D = 50$, and constant sampling probability $Np_t = 1/2$. Brighter colors indicate more likely states.

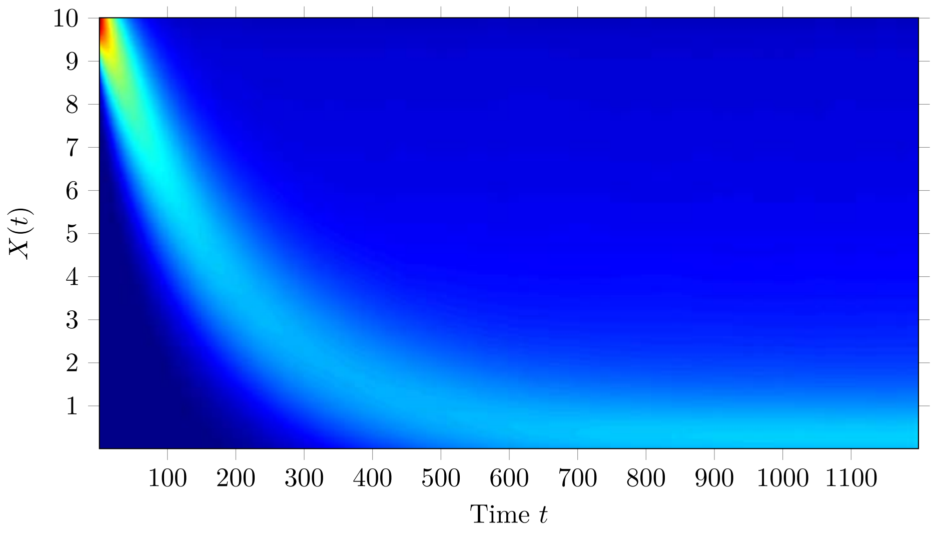


Figure 5.9: State heat map for a network with churn, consisting of 100 nodes, degree $D = 10$, and a constant sampling probability $Np_t = 1/2$. Brighter colors indicate more likely states. The churn probability is $\epsilon = \frac{1}{2}p_t$, thus the network converges to a steady state close to a gradient topology.

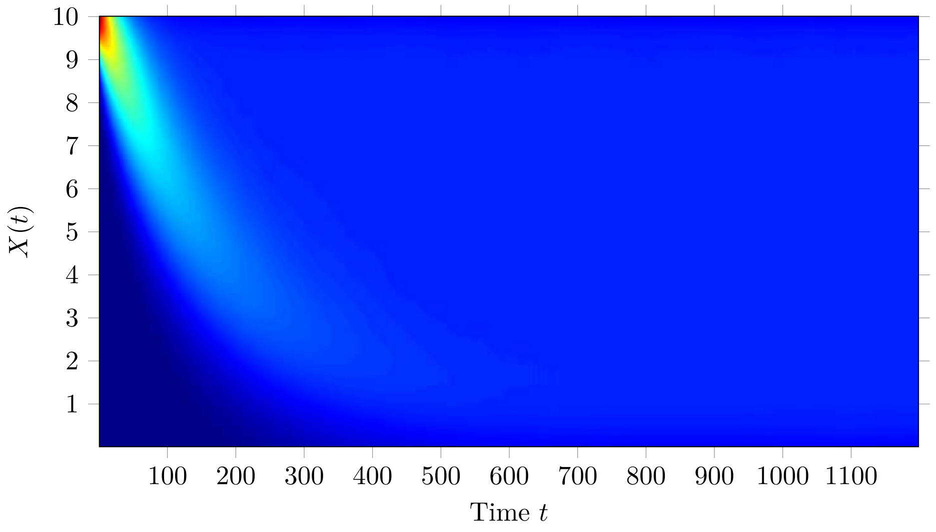


Figure 5.10: State heat map for a network with churn, consisting of 100 nodes, degree $D = 10$, and a constant sampling probability $Np_t = 1/2$. Brighter colors indicate more likely states. The churn probability is $\epsilon = p_t$, thus the network converges to a steady state where every state is equally likely.

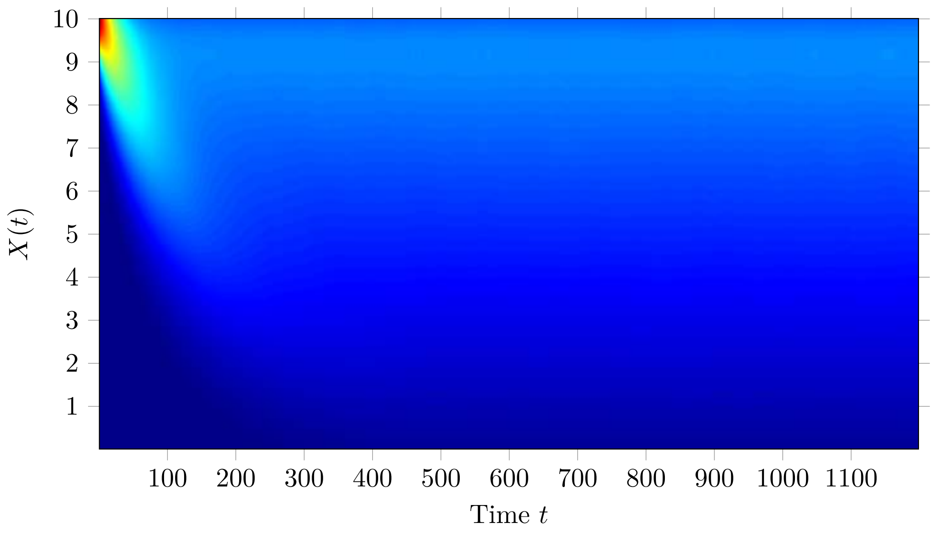


Figure 5.11: State heat map for a network with churn, consisting of 100 nodes, degree $D = 10$, and a constant sampling probability $Np_t = 1/2$. Brighter colors indicate more likely states. The churn probability is $\epsilon = 2p_t$, thus the network converges to a steady state where the initial random neighborhood is more likely, and the gradient topology is missing.

5.6 Evaluating Live-Streaming using the Gradient Topology

Now we turn to an evaluation of the effect of sampling nodes from the gradient overlay network compared to a random overlay network when building a P2P live-streaming application, called GLive. GLive is based on nodes cooperating to share a media stream supplied by a source node, and uses an approximate auction algorithm to match nodes that are willing and able to share the stream with one another. GLive extends the tree-based live-streaming, gradienTv [Payberah et al., 2010a] and Sepidar [Payberah et al., 2010b], to mesh-based live-streaming.

Nodes want to establish connections to other nodes that are as close as possible to the source. They bid for connections to the best neighbors using their own upload bandwidth, and nodes share their bounded number of connections with nodes who bid the highest (contribute the most upload bandwidth). Auctions are continuous and restarted on failures or free-riding. The desired effect of the auction algorithm is that the source will upload to nodes who contribute the most upload bandwidth, who will, in turn, upload to nodes who contribute the next highest amount of bandwidth, and so on until the topology is fully constructed.

One of the main problems with the lack of global information about nodes' upload bandwidths is that it affects the rate of convergence of the auction algorithm. Nodes would ideally like to bid for connections to other nodes who they can afford to connect to, rather than win a connection to a better node and later be removed because a better bid was received. The traditional way to discover nodes to bid on is using a uniform random peer-sampling service [Jelasity et al., 2007]. Instead, we use the gradient overlay to sample nodes, where a node's utility value is the upload bandwidth it contributes to the system. As such, the gradient should provide other nodes with references to nodes who have well-matched upload bandwidths. Payberah et al. [2010b] showed that using the gradient overlay network reduced the rate of parent switching for tree-based live-streaming by 20 % compared to random peer sampling. Here, we show for GLive the effect of sampling neighbors using random peer sampling (GLive/Random) versus sampling from the gradient overlay (GLive/Gradient).

GLive is implemented using Kompics' discrete-event simulator [Arad et al., 2009] that provides several bandwidth, latency and churn models. In our experimental setup, we set the streaming rate to 512 kbit/s, which is divided into blocks of 16 kB. Nodes start playing the media after buffering it for 5 seconds. The size of the similar view in GLive is 15 nodes, and in the auction algorithm, nodes have 8 download connections. To model upload bandwidth, we assume that each upload connection has an available bandwidth of 64 kbit/s and that the number of upload connections for the nodes is set to $2i$, where i is picked randomly from the range 1 to 10. This means that nodes have an upload bandwidth capacity between 128 kbit/s and 1.25 Mbit/s. As the average upload bandwidth of 704 kbit/s is not much higher than the streaming rate of 512 kbit/s, nodes need to find good parents to achieve the streaming performance. The media source is a single node with 40 upload connections, providing five times the upload bandwidth of the stream rate. We assume 11 utility levels, such that nodes contributing the same amount of upload bandwidth are located at the same utility level. Latencies between nodes are modeled using a latency map based on the King data-set [Gummadi et al., 2002]. We assume that the size of the sliding window for downloading is 32 blocks, such that the first 16 blocks are considered as the in-order set and the next 16 blocks are the blocks in the rare set. A block is chosen for download from the in-order set with 90 % probability, and from the rare set with 10 % probability. In the experiments, we measure the following metrics:

1. *Playback continuity*: the percentage of blocks that a node received before their playback time. We consider the case where nodes have a playback continuity greater than 99 %;

2. *Playback latency*: the difference in seconds between the playback point of a node and the playback point at the media source.

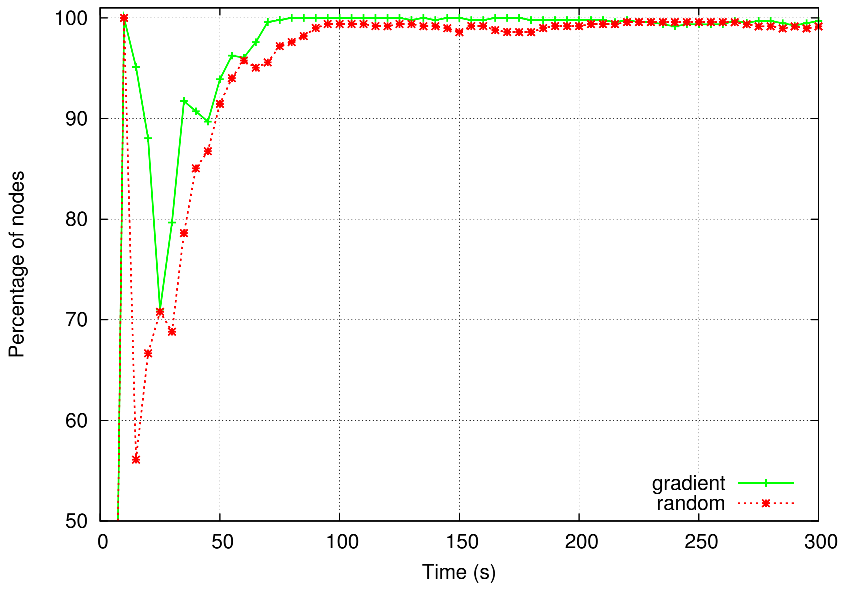
We compare the playback continuity and playback latency of GLive/Gradient and GLive/Random in the following three scenarios:

1. *Churn*: 500 nodes join the system following a Poisson distribution, with an average inter-arrival time of 100 milliseconds. Then, until the end of the simulation, nodes join and fail continuously following the same distribution with an average inter-arrival time of 1000 milliseconds;

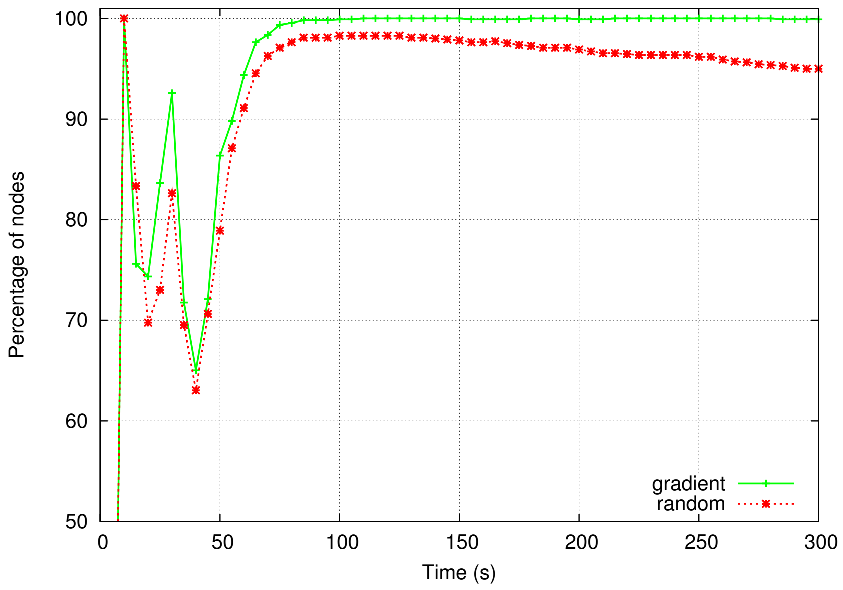
2. *Flash crowd*: 100 nodes join the system following a Poisson distribution with an average inter-arrival time of 100 milliseconds. After 150 seconds, 1000 nodes join following the same distribution with a shortened average inter-arrival time of 10 milliseconds;

3. *Catastrophic failure*: 1000 nodes join the system following a Poisson distribution with an average inter-arrival time of 100 milliseconds. After 150 seconds, 500 existing nodes fail following a Poisson distribution with an average inter-failure time of 10 milliseconds.

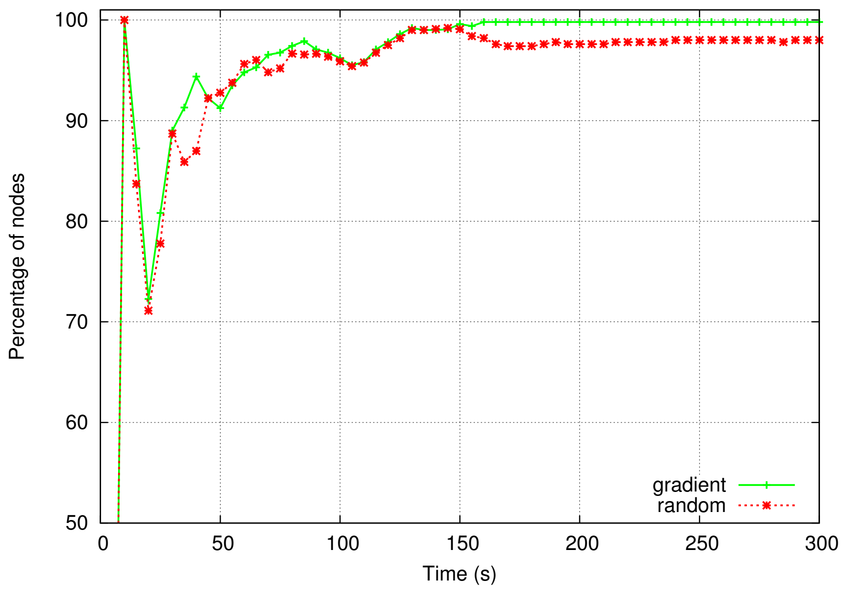
Figure 5.12 shows the percentage of the nodes that have a playback continuity of at least 99 %. We see that all the nodes in GLive/Gradient receive at least 99 % of all the blocks very quickly in all scenarios, while it takes slightly more time for GLive/Random. This is because nodes in GLive/Gradient find a good set of matches faster than nodes in GLive/Random by running the auction algorithm against nodes with similar upload bandwidth. One point to note is that the 5 seconds of buffering cause the spike in playback continuity at the start, which then drops off as nodes are joining the system. To summarize, using the gradient overlay instead of random sampling produces better performance when the system



(a) Churn



(b) Flash Crowd

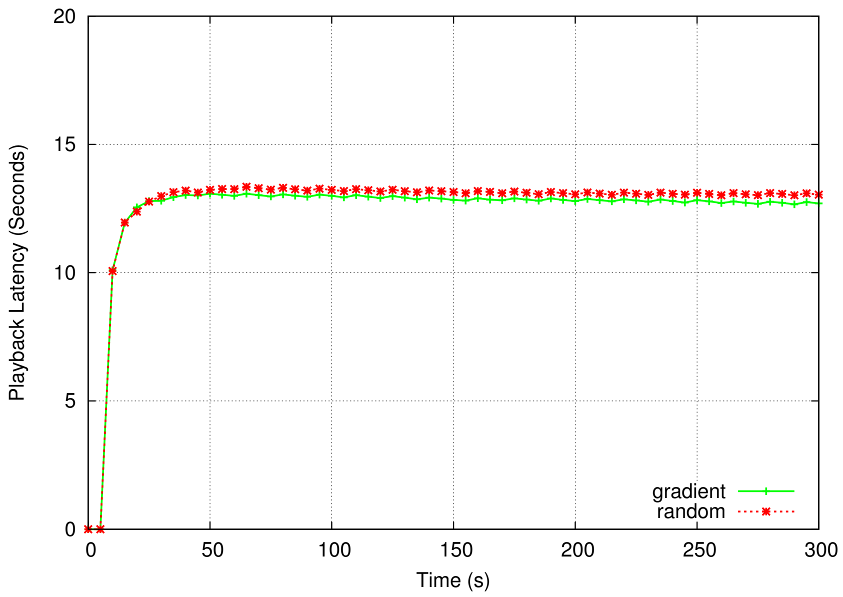


(c) Catastrophic failure

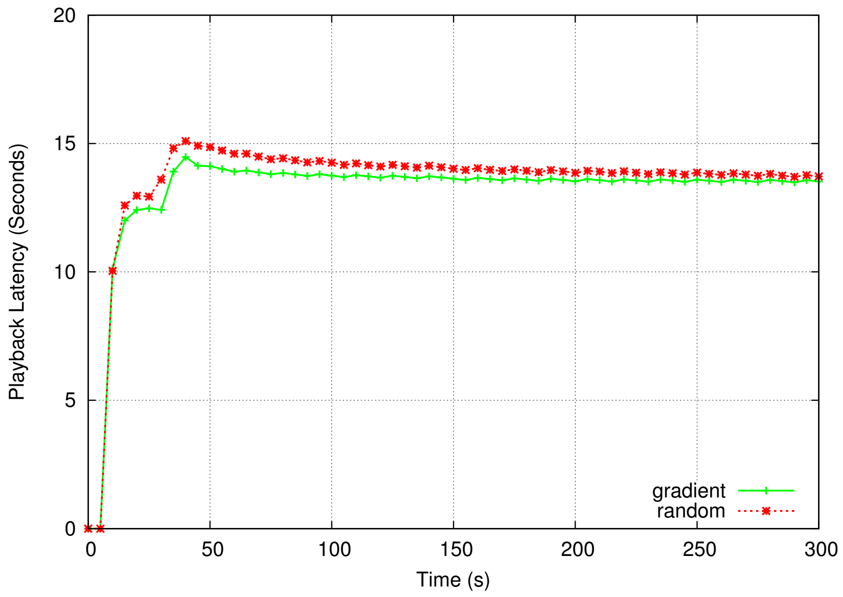
Figure 5.12: Playback continuity of the GLive/Gradient and GLive/Random systems in the churn, flash crowd and catastrophic failure scenarios.

is undergoing large changes - such as large numbers of nodes joining or failing over a short period of time.

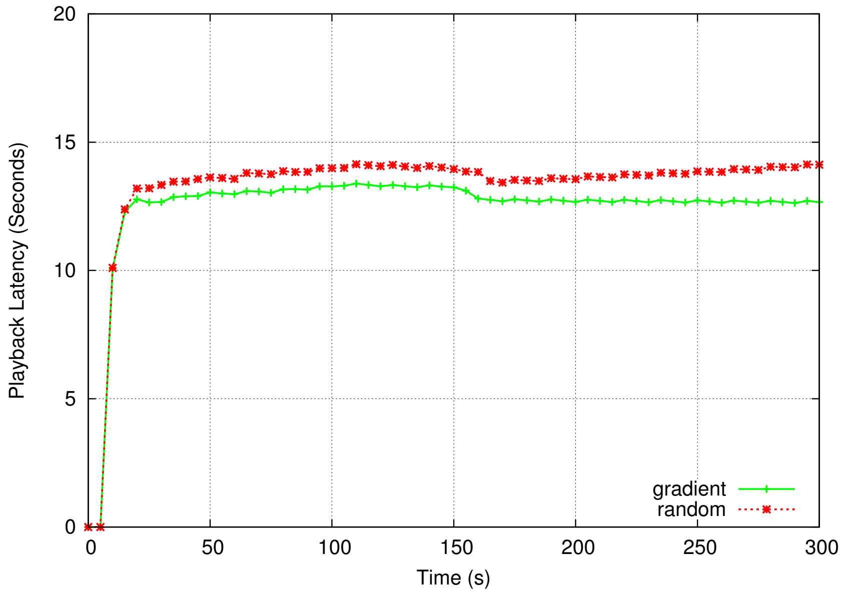
Figure 5.13 shows the playback latency of the systems in the different scenarios. As we can see, although there is only a small difference between the systems, GLive/Gradient consistently maintains relatively shorter playback latency than GLive/Random for all experiments. The playback latency includes both the 5 seconds buffering time and the time required to pull the blocks over the live-streaming overlay constructed using the auction algorithm.



(a) Churn



(b) Flash Crowd



(c) Catastrophic failure

Figure 5.13: Playback latency of the GLive/Gradient and GLive/Random systems in the churn, flash crowd and catastrophic failure scenarios.

5.7 Conclusions

In this chapter, we studied the network topology convergence problem for the gossip-generated gradient overlay network. A necessary and sufficient condition for convergence to a complete gradient topology was shown in terms of the neighbor sampling probabilities. Further, the expected convergence time was characterized for a single node, and extended to an asymptotic convergence rate estimate for the entire network. Finally, networks with churn were considered, and a threshold on the churn rate was derived for a gradient topology to emerge.

Live-streaming experiments showed the potential advantages of network topologies built using preference functions. We showed how nodes can use implicit information, captured in the gradient topology, to efficiently find suitable neighbors compared to using random sampling. As such, our work on proving convergence properties of the gradient topology could have significance for other future information-carrying topologies.

Chapter 6

Distributed Optimization via Dual Decomposition

"You don't get any points in life for doing things the hard way."
— Tim Fargo

In this chapter, we study a distributed optimization problem, essential for a broad range of network applications. Distributed optimization problems appear when a set of network users are competing for a shared resource, with different individual objectives. These problems are commonly described as a utility maximization problem, where each user has its own utility function. The goal of the network is to assign the resources such that the total utility for all users is maximized.

Decomposition methods are essential for large-scale optimization [Cohen, 1980], and they can be categorized by two approaches: those based on *primal* and *dual* decomposition, respectively. In this work, we propose and analyze a decentralized optimization algorithm based on dual decomposition, inspired by the distributed optimization method with primal decomposition used by Nedić and Ozdaglar [2007]. Both of these optimization algorithms use the subgradient method, a first-order optimization method that leads to simple decentralized algorithms. Another advantage of the subgradient method is that it can solve non-smooth optimization problems without prior knowledge of the problem structure, in comparison to the work by Nesterov [2005].

In Section 6.1, we introduce the distributed optimization problem and define the corresponding dual optimization problem. In Section 6.2, we develop a decentralized algorithm based on the dual optimization problem, and in Section 6.3, we analyze the convergence rate of the algorithm. In Section 6.4, we study the communication requirements for the algorithm, and finally, in Section 6.5, we evaluate the algorithm with numerical simulations.

6.1 Distributed Optimization Problem

We consider an optimization problem, where a network $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ of $N = |\mathcal{V}|$ interconnected agents are minimizing the sum of their individual convex (potentially non-smooth) objective functions $f^i : \mathbb{R}^n \to \mathbb{R}$, where $i \in \mathcal{V}$. The global objective function is thus

$$f(x) \doteq \sum_{i=1}^{N} f^i(x), \tag{6.1}$$

and the optimization problem is to compute

$$f^* \doteq \min_{x \in X} \; f(x) = \min_{x \in X} \; \sum_{i=1}^{N} f^i(x), \tag{6.2}$$

where $X \subseteq \mathbb{R}^n$ is the feasible domain, assumed to be non-empty and convex.

Each individual objective function f^i is assumed to be known only by agent i. Thus, the agents $\mathcal{V} = \{1, \ldots, N\}$ need to coordinate their decisions through a limited set of communication links $\mathcal{E} \subset \mathcal{V} \times \mathcal{V}$. The goal is to solve the minimization problem in a decentralized fashion, where the agents cooperatively find the optimal solution without a central coordinator. In Section 6.2 we propose an optimization procedure based on dual decomposition, and as a prelude to the proposed procedure, let us discuss the dual decomposition technique.

The dual decomposition is based on moving the coupling between different agents from the objective function to a set of constraints. To this end, let us introduce a local estimate $x^i \in X$ of the common decision variable $x \in X$ for each agent $i \in \mathcal{V}$ in the network. The optimization problem corresponding to equation (6.2) can then be written as

$$\begin{array}{ll} \underset{x^1, \ldots, x^N \in X}{\text{minimize}} & \sum_{i=1}^{N} f^i(x^i), \\ \text{subject to} & x^1 = x^2 = \cdots = x^N. \end{array}$$

Note that the coupling between the objective functions is moved from the decision variable to the consistency constraints $x^1 = x^2 = \cdots = x^N$. These constraints contain redundancies, since $x^1 = x^2$ and $x^2 = x^3$ implies that $x^1 = x^3$, thus only a subset of these are actually necessary. A subset of these constraints is selected by first partitioning the decision variable $x \in X$ into N parts, such that each part is associated with a unique agent. Let

$$x = \begin{bmatrix} x_1^T, & x_2^T, & \ldots, & x_N^T \end{bmatrix}^T$$

be a partitioning such that $x_i \in \mathbb{R}^{n_i}$ for $i \in \mathcal{V}$ with $n_i \geq 0$ and $\sum_{i=1}^{N} n_i = n$. This partitioning can be arbitrary, however, in many applications there is a natural partitioning, where x_i represents the internal state of agent i. For example, in formation control $x_i \in \mathbb{R}^{n_i}$ would represent the position of agent i, and $x_i^j \in \mathbb{R}^{n_i}$ denotes agent j's estimate of agent i's position.

We will now proceed to define the dual problem [Boyd and Vandenberghe, 2009] to (6.2), and to this end we introduce the dual variables $\lambda_{ij} \in \mathbb{R}^{n_i}$ for $i, j \in \mathcal{V}$, where λ_{ij} is associated with the constraint $x_i^i = x_i^j$. For notational simplicity, let us also introduce λ as

$$\lambda \doteq \left[\lambda_{11}^T, \ldots, \lambda_{1N}^T, \ \lambda_{21}^T, \ldots, \lambda_{2N}^T, \ldots, \lambda_{N1}^T, \ldots, \lambda_{NN}^T\right]^T.$$

We continue by forming the Lagrangian L as

$$L(x^1, x^2, \ldots, x^N, \lambda) \doteq \sum_{i=1}^N f^i(x_1^i, x_2^i, \ldots, x_N^i) + \sum_{i=1}^N \sum_{j=1}^N \lambda_{ij}^T \left(x_i^i - x_i^j\right),$$

and the corresponding Lagrange dual function q is the infimum with respect to the primal variables $x^1, \ldots, x^N$,

$$q(\lambda) \doteq \inf_{x^1, x^2, \ldots, x^N \in X} L(x^1, x^2, \ldots, x^N, \lambda).$$

Remark. The Lagrangian L is independent of the dual variables λ_{ii}, since λ_{ii} accounts for the difference $x_i^i - x_i^i = 0$. For notational simplicity we keep λ_{ii}, but define $\lambda_{ii} \doteq 0$.

The Lagrange dual function can be rewritten by introducing the subproblems $\phi^i(\lambda)$ as

$$\phi^i(\lambda) \doteq \inf_{x^i \in X} f^i(x_1^i, x_2^i, \ldots, x_N^i) + \sum_{j=1}^N \lambda_{ij}^T x_i^i - \sum_{j=1}^N \lambda_{ji}^T x_j^i. \tag{6.3}$$

Note that the subproblem ϕ^i only depends on the dual variable λ, and is the infimum over the local variable x^i. Thus, agent i is able to compute the subproblem $\phi^i(\lambda)$ locally, independent of all other agents. The Lagrange dual function is the sum of all the subproblems,

$$q(\lambda) = \sum_{i=1}^N \phi^i(\lambda).$$

Finally, the Lagrange dual optimization problem consists of the maximization of the Lagrange dual function, thus we define the optimal dual value q^* as

$$q^* \doteq \max_\lambda q(\lambda) = \max_\lambda \sum_{i=1}^N \phi^i(\lambda). \tag{6.4}$$

Slater's condition [Boyd and Vandenberghe, 2009] guarantees strong duality, i.e., $q^* = f^*$, assuming the original problem is convex and the feasible region X has an interior point. Thus, the value of the original problem is computed by the dual problem, and each agent computes a solution for the primal variables x^i as an intermediate step when solving (6.3). Moreover, if the problem is strictly convex then all agents' primal solutions will coincide.

6.2 Dual Optimization Algorithm

In this section, we propose a decentralized subgradient method for solving the Lagrange dual problem (6.4). The subgradient method is a generalization of the gradient descent method to non-differentiable functions, using the iterations

$$\lambda(t+1) = \lambda(t) + \alpha_t g(t), \tag{6.5}$$

where α_t is the step size used at time t, and $g(t)$ is a subgradient to the Lagrange dual function $q(\lambda)$ at $\lambda(t)$, i.e., $g(t)$ satisfies

$$q(\bar{\lambda}) \leq q(\lambda(t)) + g(t)^T(\bar{\lambda} - \lambda(t)), \tag{6.6}$$

for all $\bar{\lambda}$. For differentiable functions, the subgradient is unique and equal to the gradient. The subgradient to $q(\lambda)$ with respect to a single component λ_{ij} of the dual variables is $x_i^i - x_i^j$, hence the subgradient update in equation (6.5) can be written as

$$\lambda_{ij}(t+1) = \lambda_{ij}(t) + \alpha_t \left(x_i^i(t) - x_i^j(t) \right), \tag{6.7}$$

where $x^i(t)$ is given by the solution to subproblem ϕ^i, evaluated at $\lambda(t)$.

The subproblems in equation (6.3) are computationally similar to the primal subproblems of minimizing f^i, but can be solved independently by each agent. The remaining part of the algorithm handles the update of the dual variables, λ, in a decentralized manner. Assume that all agents perform their computations and communications synchronously, at the discrete times $t = 0, 1, \ldots$. Further, assume that only neighbors in the network topology $\mathcal{G}$ can communicate with each other during one time step.

For the subgradient update in equation (6.7) both agent i's estimate x_i^i and agent j's estimate x_i^j are necessary to update the dual variable λ_{ij}. Thus, we let agent i send its state estimate x_i^i to agent j, who then is able to update the dual variable using its own estimate x_i^j. Because the two agents might not be neighbors, we introduce the *time-delay* δ_{ij} that measures the multi-hop delay from agent i to agent j. Further, let $d_{ij} \doteq \delta_{ij} + \delta_{ji}$ denote the *round-trip time* between agent i and agent j.

The update rule in equation (6.7) is now modified to account for the communication delays. Let $\lambda_{ij}(t)$ denote the *commonly known dual variable* that is known to both agent i and agent j at time t, which they use to solve their respective subproblem ϕ^i and ϕ^j at time t. The solutions to the subproblems, given $\lambda(t)$, is denoted by $x^i(t)$ and $x^j(t)$. Agent i transmits the primal variable $x_i^i(t)$ to agent j, who receives it at time $t + \delta_{ij}$. Agent j then updates the dual variable λ_{ij}, which it transmits back to agent i. The total time it takes to update the commonly known dual variable is equal to the round trip time d_{ij} between agent i and agent j. The update can be formally expressed as

$$\lambda_{ij}(t + d_{ij} + 1) = \lambda_{ij}(t + d_{ij}) + \alpha_t \left(x_i^i(t) - x_i^j(t) \right). \tag{6.8}$$

Algorithm 6.1 Dual Optimization Algorithm

```
1: for every $t = 0, 1, 2, \ldots$ do
2:     for each node $i \in \mathcal{V}$ do                    ▷ Every agent $i$ runs this algorithm.
3:         Compute $\phi^i(\lambda(t)) \to x^i(t)$                   ▷ Compute primal variables.
4:         Send $x^i_i(t)$ to neighbors
5:         for each received $x^j_j(t - \delta_{ji})$ do                ▷ Update dual variables
6:             Let $\lambda_{ji}(t + \delta_{ij}) = \lambda_{ji}(t + \delta_{ij} - 1) + \alpha_{t-\delta_{ji}} \left( x^j_j(t - \delta_{ji}) - x^i_j(t - \delta_{ji}) \right)$
7:             Send $\lambda_{ji}(t + \delta_{ij})$ to agent $j$
8:             Relay received $x^j_j$ to neighbors
9:         end for
10:        Receive and relay dual variables, updating $\lambda(t + 1)$
11:    end for
12: end for
```

The procedure executed on each agent is summarized in Algorithm 6.1

Remark. The commonly known dual variable does not necessarily satisfy the subgradient condition in equation (6.6), and this is the main contribution that is analyzed in the next section.

Remark. Different parts of the dual variable λ can be updated with different round-trip delays.

6.3 Convergence Analysis

Before we move into the analysis of the algorithm, we state four assumptions about the problem instance. Assumptions 6.1 to 6.3 are commonly used for the subgradient methods and Assumption 6.4 is for the time-delays we introduced.

Assumption 6.1 (Existence of Maximizer)**.** There exists at least one finite maximizer of q, denoted by λ^*. Let Λ_{opt} denote the set of maximizers to q.

Assumption 6.2 (Bounded Subgradients)**.** Every subgradient $g(t)$ to the Lagrange dual function $q(\lambda(t))$ is uniformly bounded by G,

$$||g(t)||_2 \leq G \quad \forall t.$$

Remark. Assumption 6.2 especially holds if the feasibility set X is compact.

Assumption 6.3 (Bounded Initial Distance)**.** The distance from the initial dual variable, $\lambda(0)$, to the optimal set is bounded by R,

$$\operatorname{dist}(\lambda(0), \Lambda_{\text{opt}}) \leq R.$$

In order for the communication protocol to work, it is necessary to assume that the network is strongly connected. We impose this by assuming that the time-delays between every pair of agents are bounded, which also implicitly implies that no transmissions are lost in the network.

Assumption 6.4 (Bounded Time-Delays)**.** There exists an upper bound D on the round-trip time,

$$d_{ij} = \delta_{ij} + \delta_{ji} \leq D \quad \forall i, j \in \mathcal{V}.$$

Convergence with Constant Delays

The main result of this convergence analysis is to create a bound between the computed values of the Lagrange dual function $q(\lambda(t))$ and the optimal value q^*, which is given in Theorem 6.5. This result is built through a sequence of lemmas, but as a prelude to this, we will introduce some additional notation. Recall that $g(t)$ is a subgradient to the Lagrange dual function, q, evaluated at $\lambda(t)$. Let us define the time-shifted vector of dual variables $\bar{\lambda}(t)$ by

$$\bar{\lambda}_{ij}(t) \doteq \lambda_{ij}(t + d_{ij}),$$

thus, the entire vector $\bar{\lambda}(t)$ is

$$\bar{\lambda}(t) \doteq \begin{bmatrix} \lambda_{11}(t + d_{11}) \\ \vdots \\ \lambda_{ij}(t + d_{ij}) \\ \vdots \\ \lambda_{NN}(t + d_{NN}) \end{bmatrix}.$$

Using this notation the update rule in equation (6.8) can be written compactly as

$$\bar{\lambda}(t+1) = \bar{\lambda}(t) + \alpha_t g(t). \tag{6.9}$$

Remark. Although being a similar expression to the ordinary subgradient iteration in equation (6.5), remember that $g(t)$ is a subgradient at $\lambda(t)$ and not at $\bar{\lambda}(t)$.

The entire dual vector $\lambda(t)$ can be expressed in terms of $\bar{\lambda}(t)$ using the following vector projection P_d. Let $P_d(x)$ be defined by

$$[P_d(x)]_{ij} \doteq \begin{cases} x_{ij} & \text{if } d_{ij} = d; \\ 0 & \text{otherwise.} \end{cases}$$

In other words, $P_d(x)$ selects those components of x for which the round-trip time is equal to d. Under Assumption 6.4, P_d has two important properties; first, for any vector x, we have

$$\sum_{d=0}^{D} P_d(x) = x.$$

Second, we can express the dual variables $\lambda(t)$ from the time-shifted variables $\bar{\lambda}(t)$ as

$$\sum_{d=0}^{D} P_d(\bar{\lambda}(t-d)) = \lambda(t).$$

Lemma 6.1. *Under Assumption 6.3, the initial time-shifted dual variables are bounded to the optimal set as*

$$\operatorname{dist}\left(\bar{\lambda}(0), \Lambda_{opt}\right) \leq R.$$

Proof. By the definition of $\bar{\lambda}$, we have

$$\bar{\lambda}_{ij}(0) = \lambda_{ij}(d_{ij}).$$

Note that it takes d_{ij} time steps until agent i receives its first update on the dual variable λ_{ij}, hence $\lambda_{ij}(0) = \cdots = \lambda_{ij}(d_{ij})$ and $\bar{\lambda}(0) = \lambda(0)$. The lemma now follows from Assumption 6.3. □

Lemma 6.2. *Under Assumption 6.2, the step change in $\bar{\lambda}$ is bounded by*

$$\left|\left|\bar{\lambda}(t+1) - \bar{\lambda}(t)\right|\right|_2 \leq \alpha_t G \quad \forall t.$$

Proof. From the update rule in equation (6.9), we have

$$\bar{\lambda}(t+1) - \bar{\lambda}(t) = \alpha_t g(t).$$

Thus, by Assumption 6.2,

$$\left|\left|\bar{\lambda}(t+1) - \bar{\lambda}(t)\right|\right|_2 = \alpha_t ||g(t)||_2 \leq \alpha_t G.$$

□

In the next lemma we give an upper bound on the difference between $\bar{\lambda}(t)$ and $\lambda(t)$.

Lemma 6.3. *Under Assumptions 6.2 and 6.4, we have*

$$\left|\left|\bar{\lambda}(t) - \lambda(t)\right|\right|_2 \leq G \sum_{d=t-D}^{t-1} \alpha_d \quad \forall t.$$

Proof. Recall that $\bar{\lambda}(t) - \lambda(t)$ can be written as

$$\begin{aligned}\bar{\lambda}(t) - \lambda(t) &= \bar{\lambda}(t) - \sum_{d=0}^{D} P_d\left(\bar{\lambda}(t-d)\right)\\ &= \sum_{d=1}^{D} P_d\left(\bar{\lambda}(t) - \bar{\lambda}(t-d)\right).\end{aligned}$$

By replacing the term $\bar{\lambda}(t) - \bar{\lambda}(t-d)$ with a telescoping sum, we have

$$\begin{aligned}\bar{\lambda}(t) - \lambda(t) &= \sum_{d=1}^{D} P_d \left(\sum_{i=0}^{d-1} \left(\bar{\lambda}(t-i) - \bar{\lambda}(t-i-1) \right) \right) \\ &= \sum_{d=1}^{D} \sum_{i=0}^{d-1} P_d \left(\bar{\lambda}(t-i) - \bar{\lambda}(t-i-1) \right),\end{aligned}$$

since P_d is a linear function. Now, by changing the order of summation in this double sum,

$$\bar{\lambda}(t) - \lambda(t) = \sum_{i=0}^{D-1} \sum_{d=i+1}^{D} P_d \left(\bar{\lambda}(t-i) - \bar{\lambda}(t-i-1) \right).$$

By the triangle inequality,

$$\left|\left|\bar{\lambda}(t) - \lambda(t)\right|\right|_2 \leq \sum_{i=0}^{D-1} \left|\left| \sum_{d=i+1}^{D} P_d \left(\bar{\lambda}(t-i) - \bar{\lambda}(t-i-1) \right) \right|\right|_2.$$

Further, since the projections P_d are disjoint for different d,

$$\begin{aligned}&\left|\left| \sum_{d=i+1}^{D} P_d \left(\bar{\lambda}(t-i) - \bar{\lambda}(t-i-1) \right) \right|\right|_2 \\ &\leq \left|\left| \sum_{d=0}^{D} P_d \left(\bar{\lambda}(t-i) - \bar{\lambda}(t-i-1) \right) \right|\right|_2 \\ &= \left|\left| \bar{\lambda}(t-i) - \bar{\lambda}(t-i-1) \right|\right|_2.\end{aligned}$$

Using Lemma 6.2 yields

$$\left|\left| \bar{\lambda}(t-i) - \bar{\lambda}(t-i-1) \right|\right|_2 \leq G\alpha_{t-i-1}.$$

Assembling everything together gives us

$$\left|\left|\bar{\lambda}(t) - \lambda(t)\right|\right|_2 \leq \sum_{i=0}^{D-1} G\alpha_{t-i-1} = G \sum_{d=t-D}^{t-1} \alpha_d.$$

□

Remark. For the time steps $t < D$ before the first complete round-trip communication, we have $\lambda(t) = \sum_{d=0}^{t} P_d(\bar{\lambda}(t-d))$ and the bound in Lemma 6.3 becomes

$$||\bar{\lambda}(t) - \lambda(t)||_2 \leq G \sum_{d=0}^{t-1} \alpha_d.$$

For notational simplicity, we define $\alpha_t = 0$ for $t < 0$.

Lemma 6.4. *Under Assumptions 6.2 and 6.4, we have*

$$g(t)^T \left(\bar{\lambda}(t) - \lambda^*\right) \leq G^2 \sum_{d=t-D}^{t-1} \alpha_d + q(\lambda(t)) - q^* \quad \forall t. \tag{6.10}$$

Proof. Adding $g(t)^T (\lambda(t) - \lambda(t)) = 0$ to the left-hand side of equation (6.10) yields

$$g(t)^T \left(\bar{\lambda}(t) - \lambda^*\right) = g(t)^T \left(\bar{\lambda}(t) - \lambda(t)\right) + g(t)^T (\lambda(t) - \lambda^*).$$

By Cauchy-Schwarz inequality,

$$g(t)^T \left(\bar{\lambda}(t) - \lambda(t)\right) \leq ||g(t)||_2 \cdot ||\bar{\lambda}(t) - \lambda(t)||_2.$$

Since $g(t)$ is a subgradient at $\lambda(t)$, the subgradient definition (6.6) implies that

$$g(t)^T (\lambda(t) - \lambda^*) \leq q(\lambda(t)) - q^*.$$

Thus,

$$g(t)^T \left(\bar{\lambda}(t) - \lambda^*\right) \leq ||g(t)||_2 \cdot ||\bar{\lambda}(t) - \lambda(t)||_2 + q(\lambda(t)) - q^*.$$

Finally, using Lemma 6.3 and Assumption 6.2, we get

$$g(t)^T \left(\bar{\lambda}(t) - \lambda^*\right) \leq G^2 \sum_{d=t-D}^{t-1} \alpha_d + q(\lambda(t)) - q^*.$$

□

The function value $q(\lambda(t))$ is not guaranteed to be monotonically increasing. Therefore, the algorithm is evaluated as the maximum value achieved over T iterations. Let us now state and prove the main convergence theorem for the dual optimization algorithm.

Theorem 6.5. *Let Assumptions 6.1 to 6.4 hold. Then, the maximum value of $q(\lambda(t))$ satisfies the following bound*

$$\max_{t=0,\dots,T} q(\lambda(t)) \;\geq\; q^* - \frac{R^2 + G^2 \sum_{t=0}^{T} \left(\alpha_t^2 + 2\alpha_t \sum_{d=t-D}^{t-1} \alpha_d\right)}{2\sum_{t=0}^{T} \alpha_t}. \tag{6.11}$$

Proof. Let λ^* be any optimal point to q, and let $\bar{\lambda}(t)$ be given by the iterations in the update rule (6.9). Consider the following relation

$$\begin{aligned}0 \leq \left|\left|\bar{\lambda}(T+1) - \lambda^*\right|\right|_2^2 &= \left|\left|\bar{\lambda}(T) + \alpha_T g(T) - \lambda^*\right|\right|_2^2 \\ &= \left|\left|\bar{\lambda}(T) - \lambda^*\right|\right|_2^2 + 2\alpha_T g(T)^T\left(\bar{\lambda}(T) - \lambda^*\right) + \alpha_T^2 ||g(T)||_2^2.\end{aligned}$$

This is a recursive equation in $\left|\left|\bar{\lambda}(t) - \lambda^*\right|\right|_2^2$, and can be expanded until $t = 0$, yielding

$$0 \leq \left|\left|\bar{\lambda}(0) - \lambda^*\right|\right|_2^2 + \sum_{t=0}^{T} 2\alpha_t g(t)^T\left(\bar{\lambda}(t) - \lambda^*\right) + \sum_{t=0}^{T} \alpha_t^2 ||g(t)||_2^2.$$

Using equation (6.10) from Lemma 6.4 gives us

$$\begin{aligned}0 &\leq \left|\left|\bar{\lambda}(0) - \lambda^*\right|\right|_2^2 + \sum_{t=0}^{T} 2\alpha_t\left(G^2 \sum_{d=t-D}^{t-1} \alpha_d + q(\lambda(t)) - q^*\right) \\ &\quad + \sum_{t=0}^{T} \alpha_t^2 ||g(t)||_2^2 \\ &\leq \left|\left|\bar{\lambda}(0) - \lambda^*\right|\right|_2^2 + \sum_{t=0}^{T} 2\alpha_t G^2 \sum_{d=t-D}^{t-1} \alpha_d \\ &\quad + \left(\max_{t=0,\ldots,T} q(\lambda(t)) - q^*\right) \sum_{t=0}^{T} 2\alpha_t + \sum_{t=0}^{T} \alpha_t^2 ||g(t)||_2^2.\end{aligned}$$

Thus,

$$\begin{aligned}\max_{t=0,\ldots,T} q(\lambda(t)) \geq q^* &- \frac{\left|\left|\bar{\lambda}(0) - \lambda^*\right|\right|_2^2}{\sum_{t=0}^{T} 2\alpha_t} \\ &- \frac{\sum_{t=0}^{T} 2\alpha_t G^2 \sum_{d=t-D}^{t-1} \alpha_d + \sum_{t=0}^{T} \alpha_t^2 ||g(t)||_2^2}{\sum_{t=0}^{T} 2\alpha_t}.\end{aligned}$$

Note that λ^* is an arbitrary optimal point, thus $\left|\left|\bar{\lambda}(0) - \lambda^*\right|\right|_2$ can be replaced with $\operatorname{dist}\left(\bar{\lambda}(0), \Lambda_{\text{opt}}\right)$. Finally, using Lemma 6.1 and Assumption 6.2, we get

$$\begin{aligned}\max_{t=0,\ldots,T} q(\lambda(t)) &\geq q^* - \frac{R^2 + \sum_{t=0}^{T} 2\alpha_t G^2 \sum_{d=t-D}^{t-1} \alpha_d + \sum_{t=0}^{T} \alpha_t^2 G^2}{\sum_{t=0}^{T} 2\alpha_t} \\ &= q^* - \frac{R^2 + G^2 \sum_{t=0}^{T}\left(\alpha_t^2 + 2\alpha_t \sum_{d=t-D}^{t-1} \alpha_d\right)}{2\sum_{t=0}^{T} \alpha_t}.\end{aligned}$$

□

Corollary 6.1. *Let Assumptions 6.1 to 6.4 hold, then, for a constant step size $\alpha_t = \alpha$, we have*

$$\max_{t=0,\ldots,T-1} q(\lambda(t)) \geq q^* - \frac{R^2 + G^2\alpha^2 T\,(1+2D)}{2\alpha T}. \tag{6.12}$$

Remark. If $D = 0$ in Corollary 6.1, then the convergence result simplifies to the ordinary convergence result for the subgradient method in equation (2.4), without any delays.

Remark. Furthermore, the convergence result consists of two parts, a decaying part $\frac{R^2}{2\alpha T}$ equal to the ordinary subgradient method, and a constant term $\frac{G^2\alpha(1+2D)}{2}$, a factor $(1+2D)$ larger than that for the ordinary subgradient method. The conclusion is that the introduction of communication delays does not affect the convergence speed, but instead causes the algorithm to converge to a larger neighborhood around the optimal solution. Intuitively, we find that while the ordinary subgradient method can overshoot the optimal point with half the step length, using delayed information can cause the algorithm to continue past the optimal point for an additional D steps.

Corollary 6.2. *Consider a square summable, but not summable step size $\alpha_t \geq \alpha_{t+1} \geq 0$, for $t = 0, 1, \ldots$, such that $\sum_{t=0}^{\infty} \alpha_t^2 < \infty$ and $\sum_{t=0}^{\infty} \alpha_t = \infty$. Let Assumptions 6.1 to 6.4 hold, then*

$$\max_{t=0,\ldots,T} q(\lambda(t)) \geq q^* - \frac{R^2 + G^2(1+2D)\sum_{i=0}^{T} \alpha_{i-D}^2}{2\sum_{i=0}^{T} \alpha_i} \rightarrow q^*$$

as $T \to \infty$. Thus, this choice of step size guarantees asymptotic convergence to the optimal solution.

Convergence with Time-Varying Delays

In this section, the results from Theorem 6.5 are extended to the case when the time delays can vary in time, i.e., the transmission from agent i to agent j at time t will arrive at time $t + \delta_{ij}(t)$. In particular, this enables the possibility for the transmissions to arrive out-of-order. To accommodate this, the dual variable update in equation (6.8) is modified as

$$\lambda_{ij}(t+1) = \lambda_{ij}(t) + \sum_{\hat{t}\,:\,\hat{t}+d_{ij}(\hat{t})=t} \alpha\left(x_i^i(\hat{t}) - x_i^j(\hat{t})\right), \tag{6.13}$$

where we for simplicity have assumed a constant step size α.

The proof proceeds similarly to the constant delay case. Define the time-dependent vector-projection $P_{t,d}(x)$, for a vector x, as

$$[P_{t,d}(x)]_{ij} \doteq \begin{cases} x_{ij} & \text{if } d_{ij}(t) = d; \\ 0 & \text{otherwise.} \end{cases}$$

Note that for a fixed t, $P_{t,d}$ has the same properties as P_d in the preceding section. Thus, under Assumption 6.4 (extended to time-varying delays), for any vector x, and any fixed time t, we have

$$\sum_{d=0}^{D} P_{t,d}(x) = x.$$

With the vector projection $P_{t,d}$, it is possible to express the update rule (6.13), for the entire state vector λ, as

$$\lambda(t+1) = \lambda(t) + \alpha \sum_{d=0}^{D} P_{t-d,d}\left(g(t-d)\right). \tag{6.14}$$

Lemma 6.6. *The following is true under Assumptions 6.2 and 6.4,*

$$||\lambda(t+1) - \lambda(t)||_2 \leq \alpha(D+1)G.$$

Proof. From the update rule (6.14), we have

$$\lambda(t+1) - \lambda(t) = \alpha \sum_{d=0}^{D} P_{t-d,d}\left(g(t-d)\right).$$

Thus,

$$||\lambda(t+1) - \lambda(t)||_2 = \alpha \left|\left| \sum_{d=0}^{D} P_{t-d,d}\left(g(t-d)\right) \right|\right|_2.$$

By the triangle inequality, we further have

$$\left|\left| \sum_{d=0}^{D} P_{t-d,d}\left(g(t-d)\right) \right|\right|_2 \leq \sum_{d=0}^{D} ||P_{t-d,d}\left(g(t-d)\right)||_2 \leq \sum_{d=0}^{D} ||g(t-d)||_2.$$

Finally, by using Assumption 6.2, we have

$$||\lambda(t+1) - \lambda(t)||_2 \leq \alpha \sum_{d=0}^{D} ||g(t-d)||_2 \leq \alpha(D+1)G.$$

□

Lemma 6.7. *The following is true under Assumptions 6.2 and 6.4,*

$$g(t)^T \sum_{d=0}^{D} P_{t,d} \left(\lambda(t+d) - \lambda(t)\right) \leq \alpha G^2 D(D+1)$$

for all $t \geq 0$.

Proof. Using Cauchy-Schwarz inequality, we have

$$g(t)^T \sum_{d=0}^{D} P_{t,d} \left(\lambda(t+d) - \lambda(t)\right) \leq ||g(t)||_2 \cdot \left|\left|\sum_{d=0}^{D} P_{t,d} \left(\lambda(t+d) - \lambda(t)\right)\right|\right|_2.$$

Replacing $\lambda(t+d) - \lambda(t)$ with the telescoping sum $\sum_{i=1}^{d} \lambda(t+i) - \lambda(t+i-1)$ yields

$$\begin{aligned}\left|\left|\sum_{d=0}^{D} P_{t,d} \left(\lambda(t+d) - \lambda(t)\right)\right|\right|_2 &= \left|\left|\sum_{d=1}^{D} P_{t,d} \left(\sum_{i=1}^{d} \lambda(t+i) - \lambda(t+i-1)\right)\right|\right|_2 \\ &= \left|\left|\sum_{d=1}^{D} \sum_{i=1}^{d} P_{t,d} \left(\lambda(t+i) - \lambda(t+i-1)\right)\right|\right|_2.\end{aligned}$$

Changing the order of summation, and using the triangle inequality gives us

$$\begin{aligned}\left|\left|\sum_{d=1}^{D} \sum_{i=1}^{d} P_{t,d} \left(\lambda(t+i) - \lambda(t+i-1)\right)\right|\right|_2 &= \left|\left|\sum_{i=1}^{D} \sum_{d=i}^{D} P_{t,d} \left(\lambda(t+i) - \lambda(t+i-1)\right)\right|\right|_2 \\ &\leq \sum_{i=1}^{D} \left|\left|\sum_{d=i}^{D} P_{t,d} \left(\lambda(t+i) - \lambda(t+i-1)\right)\right|\right|_2.\end{aligned}$$

Further,

$$\begin{aligned}\sum_{i=1}^{D} \left|\left|\sum_{d=i}^{D} P_{t,d} \left(\lambda(t+i) - \lambda(t+i-1)\right)\right|\right|_2 &\leq \sum_{i=1}^{D} \left|\left|\sum_{d=0}^{D} P_{t,d} \left(\lambda(t+i) - \lambda(t+i-1)\right)\right|\right|_2 \\ &= \sum_{i=1}^{D} ||\lambda(t+i) - \lambda(t+i-1)||_2.\end{aligned}$$

By using Lemma 6.6, we have

$$\sum_{i=1}^{D} \left|\left|\sum_{d=i}^{D} P_{t,d} \left(\lambda(t+i) - \lambda(t+i-1)\right)\right|\right|_2 \leq D\alpha(D+1)G,$$

and with Assumption 6.2,

$$g(t)^T \sum_{d=0}^{D} P_{t,d} \left(\lambda(t+d) - \lambda(t)\right) \leq \alpha G^2 D(D+1).$$

□

Since the subgradient updates can arrive out-of-order, we evaluate the algorithm in the *stop model*, where the subgradients $g(t)$ are not updated after T iterations, i.e.,

$$g(t) = \begin{cases} \text{a subgradient to } q \text{ at } \lambda(t) \text{ if } t < T; \\ 0 \text{ otherwise.} \end{cases}$$

Thus, after $T + D$ iterations, all subgradient updates have completed their round-trip, and the performance of the algorithm is evaluated based on the stop time T. Note that both Lemmas 6.6 and 6.7 hold for the stop model as well. Let us now state and prove the main convergence theorem with time-varying delays:

Theorem 6.8. *Let Assumptions 6.1 to 6.4 hold. Then, the maximum value in the stop model satisfies the following bound*

$$\max_{t=0,\ldots,T-1} q(\lambda(t)) \geq q^* - \frac{R^2 + \alpha^2 D(D+1)^2 G^2 + 3\alpha^2 T(D+1)^2 G^2}{2\alpha T}. \qquad (6.15)$$

Proof. Let λ^* be an arbitrary optimal point to q, and let $\lambda(t)$ be given by the iterations in equation (6.14). Consider the following relation

$$\begin{aligned} 0 &\leq ||\lambda(T+D+1) - \lambda^*||_2^2 \\ &= \left|\left|\lambda(T+D) + \alpha \sum_{d=0}^{D} P_{T+D-d,d} \left(g(T+D-d)\right) - \lambda^*\right|\right|_2^2 \\ &= ||\lambda(T+D) - \lambda^*||_2^2 + 2\alpha \sum_{d=0}^{D} P_{T+D-d,d} \left(g(T+D-d)\right)^T \left(\lambda(T+D) - \lambda^*\right) \\ &\quad + \alpha^2 \left|\left|\sum_{d=0}^{D} P_{T+D-d,d} \left(g(T+D-d)\right)\right|\right|_2^2. \end{aligned}$$

This is a recursive equation in $||\lambda(t) - \lambda^*||_2^2$, and can be expanded until $t = 0$, thus yielding

$$\begin{aligned} 0 \leq ||\lambda(0) - \lambda^*||_2^2 &+ 2\alpha \sum_{t=0}^{T+D-1} \sum_{d=0}^{D} P_{t-d,d} \left(g(t-d)\right)^T \left(\lambda(t) - \lambda^*\right) \\ &+ \alpha^2 \sum_{t=0}^{T+D-1} \left|\left| \sum_{d=0}^{D} P_{t-d,d} \left(g(t-d)\right) \right|\right|_2^2 . \end{aligned} \tag{6.16}$$

In the *stop model*, using the fact that $g(t) = 0$ for $t \geq T$, the middle sum can be reindexed as

$$\sum_{t=0}^{T+D-1} \sum_{d=0}^{D} P_{t-d,d} \left(g(t-d)\right)^T \left(\lambda(t) - \lambda^*\right) = \sum_{t=0}^{T-1} \sum_{d=0}^{D} P_{t,d} \left(g(t)\right)^T \left(\lambda(t+d) - \lambda^*\right).$$

Consider now the effect the projection $P_{t,d}$ has on the scalar product, especially that it is symmetric in the operands and can thus be moved to the second operand, i.e.,

$$P_{t,d} \left(g(t)\right)^T \left(\lambda(t+d) - \lambda^*\right) = g(t)^T P_{t,d} \left(\lambda(t+d) - \lambda^*\right).$$

The subgradient $g(t)$ can be moved outside the summation, and by both adding and subtracting $\lambda(t)$ from the projection, we have

$$\begin{aligned} &\sum_{t=0}^{T+D-1} \sum_{d=0}^{D} P_{t-d,d} \left(g(t-d)\right)^T \left(\lambda(t) - \lambda^*\right) \\ &\qquad = \sum_{t=0}^{T-1} g(t)^T \sum_{d=0}^{D} \Big(P_{t,d}\big(\lambda(t+d) - \lambda(t)\big) + P_{t,d}\big(\lambda(t) - \lambda^*\big) \Big). \end{aligned}$$

Lemma 6.7 can be applied to the first part of this sum, and for the second part we have,

$$g(t)^T \sum_{d=0}^{D} P_{t,d} \left(\lambda(t) - \lambda^*\right) = g(t)^T \left(\lambda(t) - \lambda^*\right) \leq q(\lambda(t)) - q^*,$$

since $g(t)$ is a subgradient at $\lambda(t)$ for $t < T$. Thus,

$$\sum_{t=0}^{T+D-1} \sum_{d=0}^{D} P_{t-d,d} \left(g(t-d)\right)^T \left(\lambda(t) - \lambda^*\right) \leq \sum_{t=0}^{T-1} \Big(\alpha G^2 D(D+1) + q(\lambda(t)) - q^* \Big).$$

Return to equation (6.16), and consider the last term. Using Lemma 6.6, recalling that $\alpha \sum_{d=0}^{D} P_{t-d,d}\left(g(t-d)\right) = \lambda(t+1) - \lambda(t)$, we get

$$\alpha^2 \sum_{t=0}^{T+D-1} \left|\left| \sum_{d=0}^{D} P_{t-d,d}\left(g(t-d)\right) \right|\right|_2^2 \leq \alpha^2 (T+D)(D+1)^2 G^2.$$

Assembling the results attained so far yields

$$0 \leq ||\lambda(0) - \lambda^*||_2^2 + 2\alpha \sum_{t=0}^{T-1} \left(\alpha G^2 D(D+1) + q(\lambda(t)) - q^*\right) + \alpha^2 (T+D)(D+1)^2 G^2.$$

Note that λ^* is an arbitrary optimal point, thus using Assumption 6.3, we have that $||\lambda(0) - \lambda^*||_2$ is bounded by R. Finally, evaluating the maximal value $\max_{t=0,\ldots,T-1} q(\lambda(t))$ yields

$$\begin{aligned}\max_{t=0,\ldots,T-1} q(\lambda(t)) &\geq q^* - \frac{R^2 + 2\alpha^2 T G^2 D(D+1) + \alpha^2 (T+D)(D+1)^2 G^2}{2\alpha T} \\ &\geq q^* - \frac{R^2 + \alpha^2 D(D+1)^2 G^2 + 3\alpha^2 T (D+1)^2 G^2}{2\alpha T}.\end{aligned}$$

□

Remark. The bound in Theorem 6.8 is conservative. For example, simplifying the theorem when there is no delay, $D = 0$, yields a convergence rate three times slower than in Corollary 6.1.

Noisy Communication Channels

In the final analysis, a noisy communication and computation model is considered. The model assumes that the subgradients $g(t)$ are affected by an additive noise vector $\varepsilon(t)$. Thus, the update rule in equation (6.9) is modified such that

$$\bar{\lambda}(t+1) = \bar{\lambda}(t) + \alpha_t \left(g(t) + \varepsilon(t)\right). \tag{6.17}$$

Assumption 6.5 (Bounded Noise)**.** There exists an upper bound E on the norm of the noise vectors,

$$||\varepsilon(t)||_2 \leq E \quad \forall t. \tag{6.18}$$

The analysis for the noisy communication channels follows the ideas from the previous section.

Lemma 6.9. *Under Assumptions 6.2 and 6.5, we have*

$$\left|\left|\bar{\lambda}(t+1) - \bar{\lambda}(t)\right|\right|_2 \leq \alpha_t \left(G + E\right).$$

Proof. By the definition of the update rule in equation (6.17), we have

$$\left|\left|\bar{\lambda}(t+1) - \bar{\lambda}(t)\right|\right|_2 = \alpha_t ||g(t) + \varepsilon(t)||_2 \leq \alpha_t \left(||g(t)||_2 + ||\varepsilon(t)||_2\right).$$

Thus, by Assumptions 6.2 and 6.5,

$$\left|\left|\bar{\lambda}(t+1) - \bar{\lambda}(t)\right|\right|_2 \leq \alpha_t \left(G + E\right).$$

□

Lemma 6.10. *Under Assumptions 6.2, 6.4 and 6.5, we have*

$$\left|\left|\bar{\lambda}(t) - \lambda(t)\right|\right|_2 \leq (G+E) \sum_{d=t-D}^{t-1} \alpha_d.$$

Proof. The proof follows Lemma 6.3. We have

$$\left|\left|\bar{\lambda}(t) - \lambda(t)\right|\right|_2 \leq \sum_{i=0}^{D-1} \left|\left|\bar{\lambda}(t-i) - \bar{\lambda}(t-i-1)\right|\right|_2.$$

Using Lemma 6.9 yields

$$\left|\left|\bar{\lambda}(t) - \lambda(t)\right|\right|_2 \leq (G+E) \sum_{d=t-D}^{t-1} \alpha_d.$$

□

Lemma 6.11. *The following is true under Assumptions 6.2, 6.4 and 6.5,*

$$\left(g(t) + \varepsilon(t)\right)^T \left(\bar{\lambda}(t) - \lambda^*\right) \leq G(G+E) \sum_{d=t-D}^{t-1} \alpha_d + q(\lambda(t)) - q^* + E\left|\left|\bar{\lambda}(t) - \lambda^*\right|\right|_2.$$

Proof. By the Cauchy–Schwarz inequality,

$$\begin{aligned}\left(g(t) + \varepsilon(t)\right)^T \left(\bar{\lambda}(t) - \lambda^*\right) \leq\ & ||g(t)||_2 \cdot \left|\left|\bar{\lambda}(t) - \lambda(t)\right|\right|_2 \\ & + g(t)^T \left(\lambda(t) - \lambda^*\right) + ||\varepsilon(t)||_2 \cdot \left|\left|\bar{\lambda}(t) - \lambda^*\right|\right|_2.\end{aligned}$$

Recall that $g(t)$ is a subgradient to q at $\lambda(t)$, thus

$$g(t)^T \left(\lambda(t) - \lambda^*\right) \leq q(\lambda(t)) - q^*.$$

Using Assumptions 6.2 and 6.5 and Lemma 6.10 yields

$$(g(t)+\varepsilon(t))^T\left(\bar{\lambda}(t)-\lambda^*\right) \leq G(G+E)\sum_{d=t-D}^{t-1}\alpha_d + q(\lambda(t)) - q^* + E\left|\left|\bar{\lambda}(t)-\lambda^*\right|\right|_2.$$

□

We provide two different extensions of Theorem 6.5 for the noisy subgradients model in equation (6.17). In the first extension, we assume that the dual variables are bounded.

Theorem 6.12. *Let Assumptions 6.1 to 6.5 hold. Furthermore, assume that the dual variables are bounded by L such that*

$$\operatorname{dist}\left(\bar{\lambda}(t), \Lambda_{opt}\right) \leq L \quad \forall t.$$

Then, the maximum value satisfies the following bound,

$$\max_{t=0,\dots,T} q(\lambda(t)) \geq q^* - \frac{R^2 + \sum_{t=0}^{T}\left(\alpha_t^2(G+E)^2 + 2\alpha_t G(G+E)\sum_{d=t-D}^{t-1}\alpha_d + 2\alpha_t EL\right)}{2\sum_{t=0}^{T}\alpha_t}. \tag{6.19}$$

Proof. Let λ^* be any optimal point to q, and let $\bar{\lambda}(t)$ be given by the noisy iterations in equation (6.17). Consider the following relation

$$\begin{aligned} 0 &\leq \left|\left|\bar{\lambda}(T+1)-\lambda^*\right|\right|_2^2 = \left|\left|\bar{\lambda}(T)+\alpha_T(g(T)+\varepsilon(T))-\lambda^*\right|\right|_2^2 \\ &= \left|\left|\bar{\lambda}(T)-\lambda^*\right|\right|_2^2 + 2\alpha_T(g(T)+\varepsilon(T))^T\left(\bar{\lambda}(T)-\lambda^*\right) + \alpha_T^2||g(T)+\varepsilon(T)||_2^2. \end{aligned}$$

By Lemma 6.11, we have

$$\begin{aligned} \left|\left|\bar{\lambda}(T+1)-\lambda^*\right|\right|_2^2 &\leq \left|\left|\bar{\lambda}(T)-\lambda^*\right|\right|_2^2 + \alpha_T^2||g(T)+\varepsilon(T)||_2^2 \\ &\quad + 2\alpha_T\left(G(G+E)\sum_{d=T-D}^{T-1}\alpha_d + q(\lambda(T)) - q^* + E\left|\left|\bar{\lambda}(T)-\lambda^*\right|\right|_2\right). \end{aligned}$$

Since λ^* is an arbitrary optimal point, this especially holds for

$$\lambda^* = \operatorname*{arg\,min}_{\lambda\in\Lambda_{\text{opt}}}\left|\left|\bar{\lambda}(T)-\lambda\right|\right|_2,$$

hence we have the inequality

$$\begin{aligned}\operatorname{dist}\left(\bar{\lambda}(T+1),\, \Lambda_{\text{opt}}\right)^2 &\le \left|\left|\bar{\lambda}(T+1) - \lambda^*\right|\right|_2^2 \\ &\le \operatorname{dist}\left(\bar{\lambda}(T),\, \Lambda_{\text{opt}}\right)^2 + \alpha_T^2 ||g(T) + \varepsilon(T)||_2^2 \\ + 2\alpha_T &\left(G(G+E) \sum_{d=T-D}^{T-1} \alpha_d + q(\lambda(T)) - q^* + E \operatorname{dist}\left(\bar{\lambda}(T),\, \Lambda_{\text{opt}}\right) \right). \end{aligned} \tag{6.20}$$

This is a recursive relation in terms of the distance to the optimal set Λ_{opt}. Expanding this until $T = 0$ gives us

$$\begin{aligned} 0 \le \operatorname{dist}\left(\bar{\lambda}(0),\, \Lambda_{\text{opt}}\right)^2 &+ \sum_{t=0}^{T} \alpha_t^2 ||g(t) + \varepsilon(t)||_2^2 \\ &+ 2\sum_{t=0}^{T} \alpha_t \left(G(G+E) \sum_{d=t-D}^{t-1} \alpha_d + q(\lambda(t)) - q^* + E \operatorname{dist}\left(\bar{\lambda}(t),\, \Lambda_{\text{opt}}\right) \right). \end{aligned}$$

Using Lemma 6.1 and Assumptions 6.2 and 6.5, and the assumption on the bound L for the dual variables gives us

$$0 \le R^2 + \sum_{t=0}^{T} \alpha_t^2 \left(G+E\right)^2 + 2\sum_{t=0}^{T} \alpha_t \left(G(G+E) \sum_{d=t-D}^{t-1} \alpha_d + q(\lambda(t)) - q^* + EL \right).$$

Thus,

$$\max_{t=0,\ldots,T} q(\lambda(t)) \ge q^* - \frac{R^2 + \sum_{t=0}^{T} \left(\alpha_t^2 (G+E)^2 + 2\alpha_t G(G+E) \sum_{d=t-D}^{t-1} \alpha_d + 2\alpha_t EL \right)}{2\sum_{t=0}^{T} \alpha_t}.$$

□

Corollary 6.3. *For a constant step size $\alpha_t = \alpha$, the result in Theorem 6.12 becomes*

$$\max_{t=0,\ldots,T-1} q(\lambda(t)) \ge q^* - \frac{R^2 + \alpha^2 T (G+E)\,(G+E+2GD)}{2\alpha T} + EL.$$

As a final extension of Theorem 6.5 on the noisy subgradient model, we consider Lagrange dual functions q with a *sharp maxima*, as an alternative to bounding the dual variables in Theorem 6.12.

Theorem 6.13. *Let Assumptions 6.1 to 6.5 hold. Furthermore, assume that $q(\lambda)$ decreases at least linearly as λ moves away from the optimal set Λ_{opt}, i.e., for some constant $\mu > 0$*

$$q(\lambda) \leq q^* - \mu \operatorname{dist}(\lambda, \Lambda_{opt}).$$

If $\mu > E$, then, the maximum value satisfies the following bound,

$$\max_{t=0,\dots,T} q(\lambda(t)) \geq q^* - \frac{R^2 + (G+E)^2 \sum_{t=0}^{T} \left(\alpha_t^2 + 2\alpha_t \sum_{d=t-D}^{t-1} \alpha_d\right)}{\left(1 - \frac{E}{\mu}\right) \sum_{t=0}^{T} 2\alpha_t}. \tag{6.21}$$

Proof. Let λ^* be any optimal point to q, and let $\bar{\lambda}(t)$ be given by the noisy iterations in equation (6.17). The proof proceeds from equation (6.20) in the previous proof, using the triangle inequality

$$\operatorname{dist}\left(\bar{\lambda}(T), \Lambda_{\text{opt}}\right) \leq \left|\left|\lambda(T) - \bar{\lambda}(T)\right|\right|_2 + \operatorname{dist}\left(\lambda(T), \Lambda_{\text{opt}}\right).$$

Further, by the sharp maxima assumption, we have for some constant $\mu > 0$

$$\operatorname{dist}\left(\bar{\lambda}(T), \Lambda_{\text{opt}}\right) \leq \left|\left|\lambda(T) - \bar{\lambda}(T)\right|\right|_2 - \frac{1}{\mu}\left(q(\lambda(T)) - q^*\right).$$

Thus, by also using Lemma 6.10, we have

$$\begin{aligned}\operatorname{dist}\left(\bar{\lambda}(T+1), \Lambda_{\text{opt}}\right)^2 \leq{}& \operatorname{dist}\left(\bar{\lambda}(T), \Lambda_{\text{opt}}\right)^2 + \alpha_T^2 ||g(T) + \varepsilon(T)||_2^2 \\ &+ 2\alpha_T \left(1 - \frac{E}{\mu}\right)\left(q(\lambda(T)) - q^*\right) + 2\alpha_T (G+E)^2 \sum_{d=T-D}^{T-1} \alpha_d.\end{aligned}$$

With $\mu > E$ then $\left(1 - \frac{E}{\mu}\right) > 0$. Expanding the recursive relation until $T = 0$ yields

$$\begin{aligned}0 \leq \operatorname{dist}\left(\bar{\lambda}(0), \Lambda_{\text{opt}}\right)^2 + \sum_{t=0}^{T} \alpha_t^2 ||g(t) + \varepsilon(t)||_2^2 + \sum_{t=0}^{T} 2\alpha_t \left(1 - \frac{E}{\mu}\right)\left(q(\lambda(t)) - q^*\right) \\ + \sum_{t=0}^{T} 2\alpha_t (G+E)^2 \sum_{d=t-D}^{t-1} \alpha_d.\end{aligned}$$

Using Lemma 6.1 and Assumptions 6.2 and 6.5, the lower bound on the maximal value can then be written as

$$\max_{t=0,\dots,T} q(\lambda(t)) \geq q^* - \frac{R^2 + (G+E)^2 \sum_{t=0}^{T} \left(\alpha_t^2 + 2\alpha_t \sum_{d=t-D}^{t-1} \alpha_d\right)}{\left(1 - \frac{E}{\mu}\right) \sum_{t=0}^{T} 2\alpha_t}.$$

□

Corollary 6.4. *For a constant step size $\alpha_t = \alpha$, the result in Theorem 6.13 becomes*

$$\max_{t=0,\dots,T-1} q(\lambda(t)) \geq q^* - \frac{R^2 + \alpha^2 T (G+E)^2 (1+2D)}{2\alpha T \left(1 - \frac{E}{\mu}\right)}.$$

6.4 Communication Cost

In distributed systems, such as wireless sensor networks, the limiting resource is often the communication capacity [Akyildiz et al., 2002]. Therefore it is important to analyze the convergence rate not only based on time, but also in relation to the communication cost. Assuming a fixed numeric precision, we define the communication cost as the number of real valued scalars transmitted over each edge in the network. There are two contributions to the communication cost: the distribution of primal variables $x_i^i(t)$, and the accumulation of dual variables λ_{ji}. These contributions can be analyzed separately, and per iteration of the algorithm.

First, consider the distribution of primal variables. Recall that agent i only needs to distribute the part $x_i^i(t) \in \mathbb{R}^{n_i}$ of its state estimate $x^i(t)$ to the other agents in the network in order for the dual variable update to take place. We assume that a network routing procedure is in place, such that an agent receives the estimate $x_i^i(t)$ once and only once. Since there are N agents in the network, each part $x_i^i(t) \in \mathbb{R}^{n_i}$ is thus transferred over exactly $N-1$ edges. The total communication cost for all primal variables becomes

$$\sum_{i=1}^{N} (N-1) n_i = (N-1) n.$$

Let us now consider the communication cost for aggregating the dual variables $\lambda_{ij} \in \mathbb{R}^{n_i}$. By similarly assuming that each dual variable is only received at most once for each agent, this would yield the following communication cost

$$\sum_{i=1}^{N} \sum_{j=1}^{N} (N-1) n_i = (N-1) N n.$$

This is significantly larger than the communication cost for the primal variables, since there are many more dual variables. However, recall that agent i's subproblem $\phi^i(\lambda)$ only depends on the dual variables λ_{ji}, $j \in \mathcal{V}$, and the sum of dual variables $\sum_{j=1}^{N} \lambda_{ij}$. Since the dual variables λ_{ji} are computed locally at agent i, each agent only needs to receive the sum of the dual variables $\sum_{j=1}^{N} \lambda_{ij}$, and we can aggregate this in the network.

Thus, when an agent k receives a set of dual variables λ_{ij}, $j \in \mathcal{V}$ directed towards agent i, it can compute the sum of all these values and its own λ_{ik}, and only transmit this sum instead of the individual values. Thus, all dual variables λ_{i*} directed towards agent i are then only transferred over $N-1$ edges, and by aggregating the dual variables, the total communication cost for all dual variables is

$$\sum_{i=1}^{N}(N-1)n_i = (N-1)n.$$

Proposition 6.14. *The total communication cost, in terms of transmitted real scalar values per iteration, for the dual optimization algorithm is*

$$2(N-1)n. \tag{6.22}$$

Remark. By aggregating the dual variable updates, and neglecting the communication cost for the routing protocol overhead, the communication cost becomes independent of the network topology.

Remark. Inspection of the problem structure may reduce the communication cost further. Recall that each local objective function f^i is assumed to depend on all of the decision variables $x_1, x_2, \ldots, x_N$. If the optimization problem instead has a sparse dependency structure, e.g., agents only depending on their neighbors' states, then only local communication would be needed. For example, if f^i does not depend on x_j then agent i will not need a local estimate x^i_j, and the dual variable λ_{ji} accounting for the difference between x^j_j and x^i_j could also be removed.

6.5 Numerical Results

In this section we will study the convergence rate of the distributed optimization algorithm on different network topologies. The dual distributed optimization algorithm presented above was inspired by a distributed optimization scheme by Nedić and Ozdaglar [2007], which we refer to as the *primal algorithm.* Therefore, the convergence rate and communication cost for the dual distributed optimization algorithm is compared to the primal algorithm, which is introduced next.

A Primal Distributed Optimization Algorithm

The primal algorithm is also an iterative optimization algorithm, where each agent has a local estimate $x^i(t)$ of the decision variable. However, the iterations for the primal variables consist of two steps, an average consensus update and a subgradient step for the local objective function, i.e., agent i updates its estimate $x^i(t)$ by

$$x^i(t+1) = \underbrace{\sum_{j=1}^{N} W_{j,i}(t)x^j(t)}_{\text{Average consensus}} \underbrace{- \alpha^i_t g^i(t)}_{\text{Subgradient}}, \tag{6.23}$$

where W is an average consensus matrix, and $g^i(t)$ is a subgradient to f^i at $x^i(t)$. The average consensus matrix $W_{j,i}(t)$ is zero if there is no directed edge from agent j to agent i at time t.

This algorithm has attracted a lot of research interest, in part due to its remarkable simplicity, while still having robust convergence properties. Especially, it does not require a routing network protocol for communication, but can be implemented with asynchronous gossip or broadcast communication [Srivastava and Nedić, 2011].

The communication for the primal optimization algorithm consists of transmissions of the entire state x^i from each agent i to all of its neighbors, hence the communication cost for the primal algorithm is $n|\mathcal{E}|$.

Remark. In certain communication protocols, using broadcast transmissions could improve the communication cost for the primal algorithm, since the agents are transmitting the same information to all of their neighbors.

Evaluating the Convergence Rate

In both the primal and dual optimization algorithms, each agent $i \in \mathcal{V}$ maintains its own estimate of the entire primal decision variable $x^i(t)$, which prevents the evaluation of $f(t)$ as defined in equation (6.1). In order to compare the convergence rate, we proceed to define the current function value evaluated over the average agent state. To this end, define the average state as

$$x_{\mathrm{avg}}(t) \doteq \frac{1}{N}\sum_{i=1}^{N} x^i(t).$$

The function value is then evaluated as

$$f(t) \doteq \sum_{i=1}^{N} f^i(x_{\mathrm{avg}}(t)).$$

Finally, since the function does not need to be monotonically decreasing, we evaluate each algorithm as the minimum value attained until time T,

$$f_{\min}(T) \doteq \min_{t=0,\dots,T} f(t).$$

For the dual optimization algorithm, we can evaluate the function value based on each agent's own estimate of the primal state, as well as evaluate the Lagrange dual function over the commonly known dual variable $\lambda(t)$. Thus, we define the current Lagrange dual function value as

$$q(t) \doteq \sum_{i=1}^{N} \phi^i(\lambda(t)).$$

Since the dual function does not need to be monotonically increasing, we evaluate the dual value as the maximum attained until time T, e.g.,

$$q_{\max}(T) \doteq \max_{t=0,\ldots,T} q(t).$$

Simulation Model

In the following examples, we will study different network topologies, each consisting of $N = 15$ agents. The network topologies are time-invariant, and the Metropolis weights [Xiao et al., 2005] are used for the average consensus step for the primal algorithm. For the dual optimization algorithm, the communication delay δ_{ij} between agent i and agent j is equal to $\operatorname{dist}(i, j) - 1$ in the communication graph. This means that neighbors can communicate directly with each other, and it takes one extra step for each intermediate agent on the shortest path between two agents. In particular, the upper bound on the round-trip times, D, is less than or equal to $2(N-2)$.

Each agent is associated with a single scalar $x_i \in \mathbb{R}$, thus the optimization variable $x \in \mathbb{R}^{15}$. The initial values are set to zero, $x(0) = 0$ and $\lambda(0) = 0$. The local objective functions used for the simulations are selected as quadratic functions. The reason for using this class of objective functions is primarily because they are easy to work with, and efficient to compute, while still being non-trivial. Thus, the local objective function $f^i(x)$ can be written as

$$f^i(x) = x^T A_i x + b_i^T x + c_i.$$

This function is convex if and only if the matrix A_i is positive semidefinite, i.e., all eigenvalues are non-negative. The parameters A_i, b_i and c_i are chosen randomly, b_i and c_i are drawn from a normal distribution, and A_i is generated as a random symmetric matrix, with eigenvalues drawn from a uniform distribution in the interval $[0, 10]$.

In order to have a uniform upper bound on the subgradients, the optimization variables are restricted to a compact convex set X, where X is determined by

$$x \in X \;\Leftrightarrow\; -100 \leq x_i \leq 100 \quad \forall i.$$

Furthermore, both algorithms use a step size parameter, but they are on different scales. Therefore, we limit both algorithms to use a constant step size $\alpha_t = \alpha$, and then for each algorithm select the optimal constant step size. Thus, we select a step size α for each algorithm such that $f_{\min}(T)$ is minimal at time step $T = 1000$, while we extend the simulation horizon for 2000 steps.

Connected Random Graph

As a first example, we consider a random, strongly connected network, as shown in Figure 6.1. The maximal delay between any two agents is 3, and hence, the

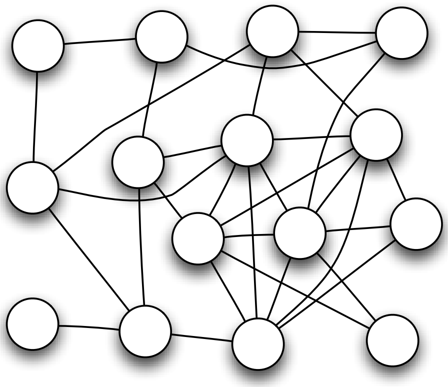


Figure 6.1: A network $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ consisting of 15 agents, drawn as circles, connected by 31 undirected communication lines. The diameter of the network is 4.

upper bound on the round-trip time is $D = 6$. The step sizes are $\alpha = 0.00054$ for the primal algorithm, and $\alpha = 0.080$ for the dual algorithm, as determined in Figure 6.2.

Optimal value	f^*	-7649.960
Primal alg.	$f_{\min}(1000)$	-7649.409
Primal alg.	$f_{\min}(2000)$	-7649.425
Dual alg.	$f_{\min}(1000)$	-7649.956
Dual alg.	$f_{\min}(2000)$	-7649.960

Table 6.1: The minimal function values for the primal and dual optimization algorithms, evaluated after 1000 and 2000 iterations, using the optimal step sizes from Figure 6.2.

The convergence rate is evaluated on the time interval $t = 0, \ldots, 2000$, shown in Figure 6.3. Note that the primal algorithm reaches a neighborhood around the optimal value at time step 1000, and does not significantly improve during the next 1000 steps. This is a trade-off in the step size choice; a smaller step size would yield a slower convergence, but to a value closer to the optimal, thus giving a worse solution at time step 1000 but a better solution at time step 2000. The dual algorithm, on the other hand, is closer to the optimal value before 1000 steps, and continues to improve during the next 1000 steps.

Finally, let us compare the communication cost for each algorithm. The network in Figure 6.1 has 62 edges, counted with direction, and hence, the communication cost is $62n$ for the primal algorithm, and $28n$ for the dual algorithm.

Thus, for this problem instance, the dual algorithm converges faster, to a smaller neighborhood around the optimal value, and with less communication than the

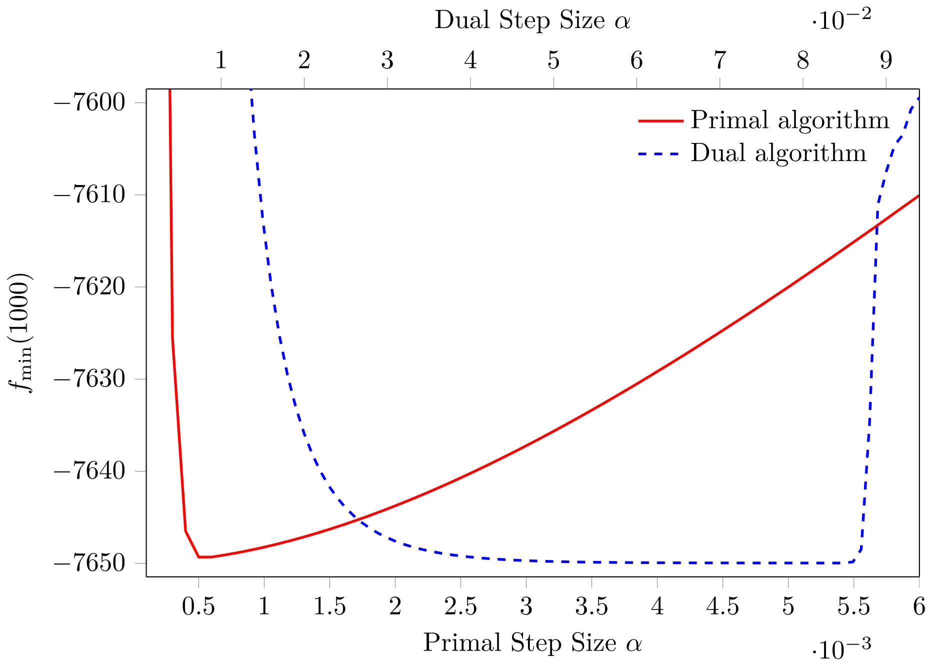


Figure 6.2: The step sizes are tuned for the primal and dual optimization algorithm separately. The minimal functional value $f_{\min}$ is evaluated after 1000 iterations for different step sizes. Note the different scales for the primal and dual step sizes.

primal algorithm.

Line Graph

Next, consider a network with a line topology, as shown in Figure 6.4. A line graph with N agents is a connected graph with diameter $N-1$, and thus, has the highest round-trip time among all connected graphs of N agents. The round-trip time is bounded by $D = 26$. The convergence rate is evaluated on the time interval $t = 0, \ldots, 2000$ in Figure 6.5. Note that the optimal step sizes also changed to $\alpha = 0.00044$ for the primal algorithm, and $\alpha = 0.014$ for the dual algorithm, thus both algorithms use a smaller step size when the communication delay is large. This coupling between delay and step size makes it hard to directly compare the convergence rates with the theoretical results in equation (6.12).

The results here are in favor of the primal algorithm, which has reached a lower value than the dual algorithm after 1000 iterations. The dual algorithm continues to improve slowly, and after 1500 iterations it closes in on the primal algorithm. Note the characteristic staircase appearance in the figure, caused by oscillations in

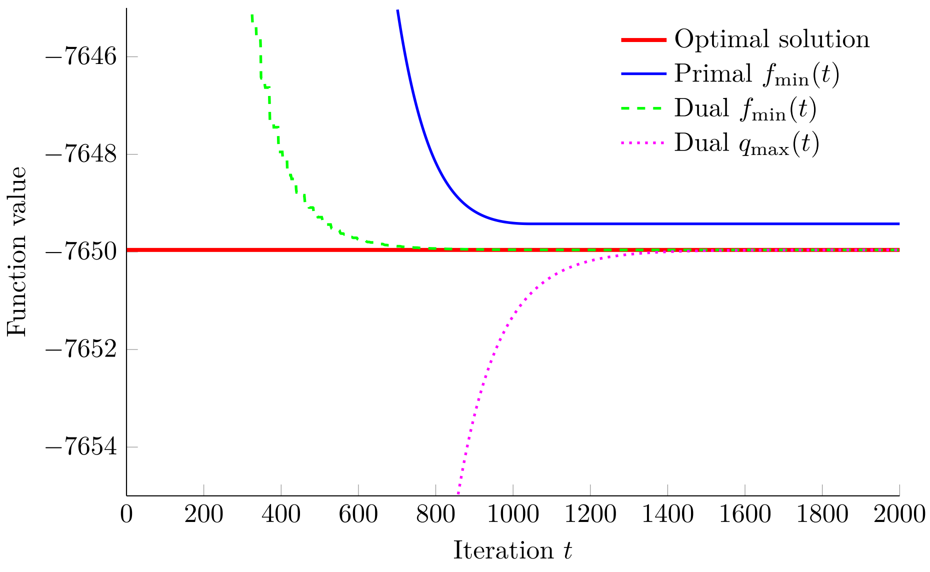


Figure 6.3: Convergence simulation for agents communicating through a connected network (Figure 6.1). The primal algorithm reaches a neighborhood around the optimal value, while the dual algorithm converges to the optimal value.

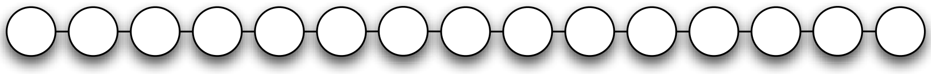


Figure 6.4: The line graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ consisting of 15 agents. The line topology has the largest round-trip time among all network topologies, with a diameter of 14.

the underlying function values $f(t)$, which are due to the communication delays in the network. The number of directed edges in the network is 28, thus the communication cost for both the primal and dual algorithm are equal to $28n$.

For this network topology, the primal algorithm converges faster, and with the same communication cost as the dual algorithm.

Circular Graph

Consider a network with circular topology, as shown in Figure 6.6. Notice that this topology can be obtained from the line graph by adding a single edge, however, this causes the diameter of the graph to shrink from 14 to 7, and the upper bound on the round-trip time is $D = 12$ instead of 26. The convergence rate is evaluated on the time interval $t = 0, \ldots, 2000$ in Figure 6.7. The optimal step sizes also changed

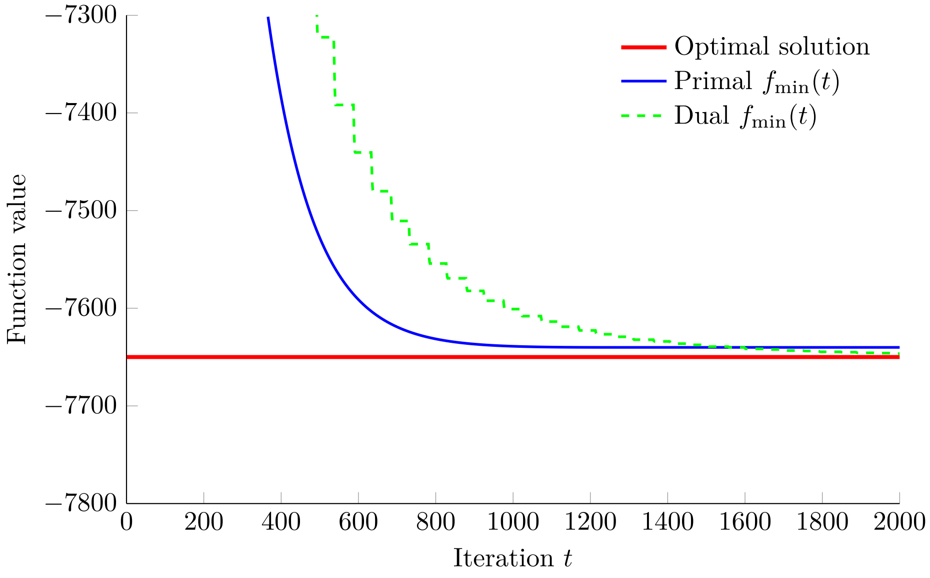


Figure 6.5: Convergence simulation for agents communicating through a line topology (Figure 6.4). The primal algorithm attains a lower value than the dual algorithm after 1000 iterations, but the dual algorithm continues to improve. The dual function value $q_{\max}(t)$ is outside the scope of this figure.

to $\alpha = 0.00050$ for the primal algorithm, and $\alpha = 0.041$ for the dual algorithm, following the change in the round-trip time.

The results are similar to those of the first example, where the primal algorithm reaches a neighborhood around the optimal value before step 1000, while the dual algorithm converges to the optimal value. Also, the communication cost for the primal algorithm is $30n$ compared to the $28n$ for the dual algorithm.

Noisy Communication Channels

As a final example, we return to the network shown in Figure 6.1, but this time we study the convergence when the communication channels are subject to the noisy update in equation (6.17). The stochastic noise signal $\varepsilon(t)$ is generated such that $|\varepsilon(t)| < 0.05$, and the convergence rate is evaluated in Figure 6.8.

It seems as if the algorithm is converging at the same rate with and without noise, but without noise it reaches a neighborhood close to the optimal value.

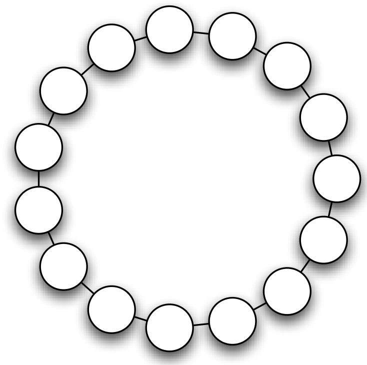


Figure 6.6: The circular topology $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ consisting of 15 agents. The circular topology has one additional edge compared to the line topology, but reduces the round-trip time to half.

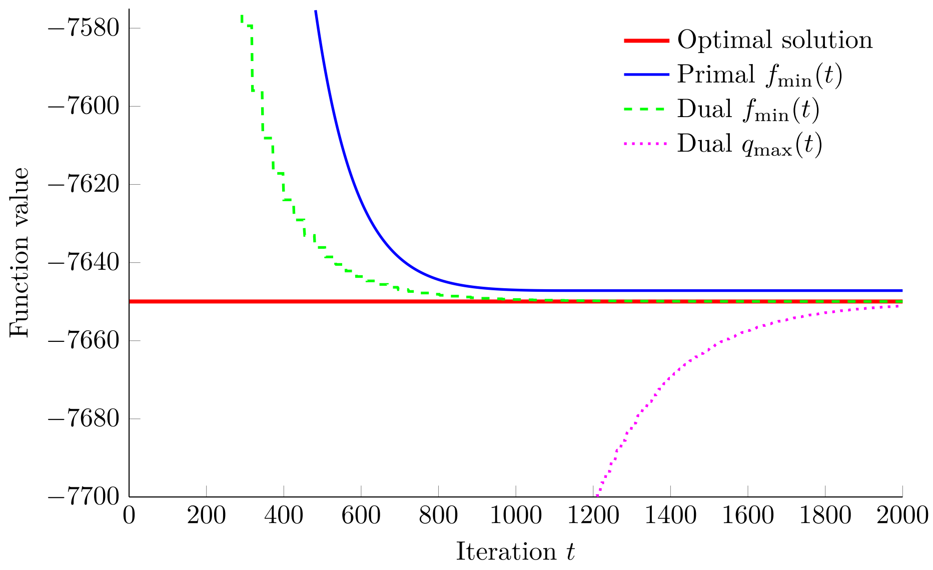


Figure 6.7: Convergence simulation for agents communicating through a circular topology (Figure 6.6). The primal algorithm has reached a neighborhood around the optimal value, while the dual algorithm converges to the optimal value.

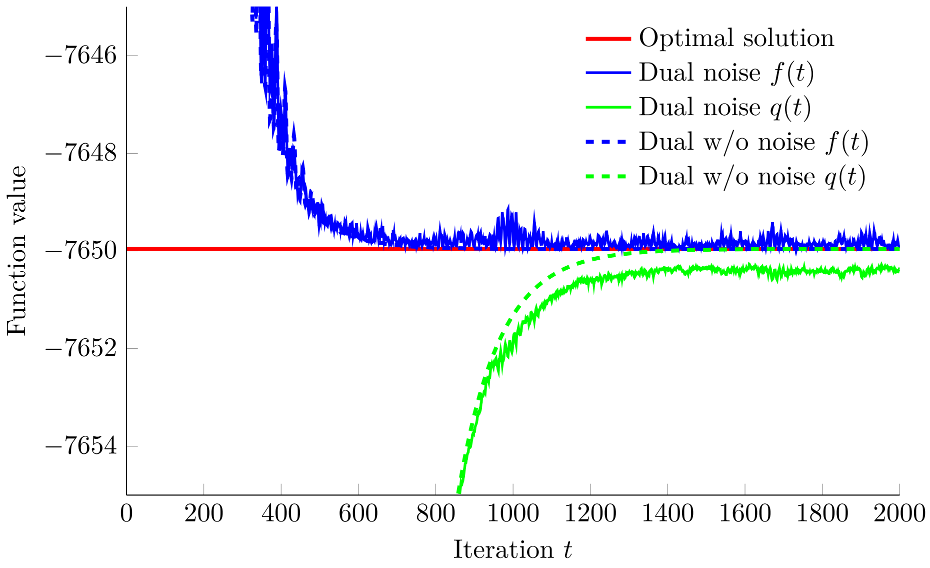


Figure 6.8: Convergence simulation for agents communicating over noisy channels in the network from Figure 6.1. The convergence rate is similar with and without noise. Notice that in this figure, the value $f(t)$ at each iteration is shown compared to the minimal value $f_{\min}(t)$ in the previous figures.

6.6 Conclusions

Distributed optimization problems appear in a broad range of practical applications. In this chapter, we have developed a novel decentralized optimization algorithm, based on the dual decomposition technique and the subgradient method. The optimization algorithm is capable of handling communication delays, and it was proven that the algorithm converges to the optimal solution when the communication delay is bounded. The convergence results were also extended to time-varying communication delays, and noisy communication channels. A communication cost analysis shows that this algorithm is always at least as communication efficient as the primal counterpart.

The numerical simulations show that the proposed dual optimization algorithm is capable of achieving both a faster convergence speed and a decrease in the communication cost compared to the primal optimization algorithm. The dual optimization algorithm was particular efficient when the network had a small diameter, and hence small communication delays. A particular class of applications that could be suitable for this optimization method is within formation control, where each agent is optimizing its trajectory based on the trajectories of its closest neighbors, which are also within communication range. Hence, only communication with the direct neighbors would be necessary, thereby minimizing the communication delay.

Chapter 7

Conclusions and Future Work

"Even though the future seems far away, it is actually beginning right now."

— Mattie Stepanek

This final chapter concludes the thesis with a brief summary of the main contributions, and some interesting directions for future research.

7.1 Conclusions

In this thesis we have explored some important and interesting dynamical network systems. We considered applications within the transportation field and peer-to-peer (P2P) content distribution, but also developed methods for optimization and estimation over networks. The main contributions from each chapter are summarized below.

Efficiency in Transportation Networks

We investigated collaboration strategies for transportation networks, motivated by the development within intelligent transportation, mainly for road freight transport. An efficiency measure for the collaborative transportation scenarios was introduced, that determines how efficient the road utilization is in comparison to a centralized planner. The efficiency measure was evaluated on real data from the New York City taxis.

We also studied the optimal idling locations for trucks, and the optimal locations for distribution centers along a single highway going through multiple cities. These locations were then exploited in a simulation of a collaborative freight transport system in Sweden.

These studies showed that there is great potential for collaborative transport solutions to improve the efficiency in the transportation system.

Estimation in Anonymous Networks

We derived distributed estimators to estimate both the network size and probability mass functions when agents are anonymous. These estimators were based on probabilistic methods and the max consensus protocol. One of the main advantages with the max consensus protocol compared to other strategies is that it has the fastest possible convergence time, and is very easy to implement, even in networks with unreliable communication, through pairwise gossiping communication. Even though the accuracy of these methods is not perfect, it can be made sufficiently good by increasing the packet sizes.

We designed these methods to specifically handle time dependent networks through a regularization term, which penalizes hypotheses conflicting with a-priori assumptions on the network's behavior. We explicitly considered and characterized the class of quadratic regularization terms, which resulted in closed-form estimators.

Topology Convergence in Peer-to-Peer Networks

We studied a P2P network for live-streaming video applications, and considered the advantages of using a gradient topology for finding suitable neighbors. The advantage of this method is that the peers can find a good set of neighbors more quickly, and thus avoid rapid switching, thereby decreasing the latency and probability of interruptions. We derived both necessary and sufficient conditions for convergence to a complete gradient topology, and we also characterized the expected convergence time. We extended this analysis to networks with churn, and examined how high churn rates could be tolerated while still maintaining a reasonable gradient overlay structure, as a condition for the sampling rate.

Distributed Optimization via Dual Decomposition

A decentralized optimization algorithm was introduced, based on dual decomposition together with the subgradient method, for finding the optimal solution. The main contribution of this chapter was to prove the convergence of the decentralized optimization algorithm over time-varying networks, with time-varying communication delays, and noisy communication channels. Further, the convergence rate was explicitly expressed in terms of the network parameters.

The convergence rate of the proposed algorithm was compared to previously known algorithms with extensive numerical simulations. Besides the convergence rate, the communication cost was a main concern, and it was shown that the communication cost could be kept to a minimum.

7.2 Future Work

There are many natural extensions of the work presented in this thesis. The future work can be divided into general formulations in the field of dynamical network systems, and specific problem formulations related to the problems studied in this thesis. We summarize below some ideas for future research.

General Research Directions

The research field of dynamical network systems is extremely rich, with many interesting open problems. The general research directions can be divided into two tracks: the experimental and the theoretical.

From a theoretical perspective, one fundamental task is to develop general tools for analyzing dynamical networks, that could be used across disciplines. In the field of automatic control, feedback loops have long been of great interest. What role does feedback loops play in the formation of network structures? How can feedback loops that govern the evolution of networks be found? Can concepts such as controllability and observability aid in the design of high performance dynamical network systems?

With an abundance of practical applications, it would be interesting to work with and analyze real-world data further, to validate the mathematical models. For example studying community formation in social networks, and exploring the interaction between the network structure and the information propagation through the network.

Efficiency in Transportation Networks

A fundamental assumption in our work was that the actors want to collaborate, which is not always the case in the highly competitive transportation market. We showed that through collaboration, the global transportation system could improve its efficiency, but it remains to design pricing mechanism such that collaboration actually becomes profitable for every actor.

In the collaboration scenarios we considered, all vehicles were homogeneous, and any vehicle could handle any transportation assignment. It could also happen that the optimal solution involved a single vehicle operating continuously, where we did not consider the strict laws for driver rest, maintenance stops for vehicles, nor creating a fair distribution of assignments.

The efficiency measure was developed to be evaluated a-posteriori on historical GPS traces, when the transportation routes were known. In practice, the transportation demands are rapidly changing, thus it would be interesting to extend and adapt this efficiency measure to handle incomplete information and stochastic demand models.

The road transportation model considered can capture some realistic scenarios, when there is only a single major transportation route. This model could be extended to general topologies, starting with tree-shaped topologies, to capture a wider range of transportation scenarios. Another extension is to allow vehicles to handle multiple concurrent assignments, and coordinating multiple vehicles at the distribution centers.

Estimation in Anonymous Networks

We estimated two specific properties in the anonymous network framework. A natural continuation is to ask what else can be computed or estimated in this framework. We saw in the literature review that this problem has attracted some attention, both with deterministic computing and using random number generation, but our assumption on limited communication and computation further restricts the problem. Specifically, what can be learned about the network topology besides the size?

One of the reasons for estimating the network size was to detect network faults that needs restorative actions. Further research is needed to study how to build intelligent networks that automatically reconfigure themselves to be robust against failures, while preserving the anonymity of the nodes.

It was shown that the accuracy could be improved by increasing the packet size. An important question is to consider the quantization effects due to limited number precision, and carry out a formal analysis of this effect. The next problem would be to design an optimal alphabet which results in the highest precision per bit transmitted. Some work in this direction has been carried out by Lucchese et al. [2015].

A natural continuation would also consist of developing on-line algorithms for tuning the estimation parameters, especially the packet size based on the current performance requirements in the application.

We assumed that the communication was synchronized to epochs, but it should be possible to relax this assumption. Either by introducing distributed clock synchronization schemes, or by changing our algorithms to an event-triggered version. This could be accomplished by assuming that any agent could initiate a new epoch using a randomized time-out event. Could additional properties be estimated by using the transient information while the consensus algorithm is converging to the maximum?

Another research direction could be to study the robustness of this scheme against different types of attacks. Either modeled as randomized noise or node failures, or with malicious agents who are actively trying to influence the estimate in a particular direction. In the current scheme, a single node which initializes its vector with ones would destroy the estimation.

Topology Convergence in Peer-to-Peer Networks

We considered general time-dependent sampling rates, but an important question is how to tune these probabilities. Could they be tuned on-line as topology changes are detected?

Furthermore, we used uniform sampling among all peers, but some of the random peer sampling services give biased samples, most notably those based on random walks. What would the effect be of these biased samples, in particular if the samples were generated by random walks within the same gradient overlay topology? What if the sampling strategy is further extended, so that nodes can share their neighbor sets with their neighbors, how could that improve the convergence performance?

Another property that we ignored, was the underlying physical network, where the latency can be specified as a pairwise map. Thus, the utility value of a node might not be a truly universal property, and the utility value could instead be defined as a pairwise function describing the utility a node provides to a specific other node.

Distributed Optimization via Dual Decomposition

We focused on optimization problems where the coupling between the subproblems were in the objective functions. Another possibility is that the coupling is in constraint functions, for example, when a set of independent multi-agents are using a common, limited resource. The dual decomposition method already handles constraint coupling by introducing Lagrange dual variables, so it should be a straightforward procedure to add another set of dual variables for these constraints. A future research topic is to study the convergence rate for this class of optimization problems.

It was assumed that every package sent on the network will eventually arrive to its destination. A common practice in network control is to drop packages to avoid network congestion. Thus, a possible extension is to analyze the behavior of the algorithms when packages can be lost in the network.

The optimization models assumed that all agents are computing and communicating at synchronous time steps. An interesting extension is to develop, and analyze, an asynchronous version of these distributed optimization algorithms. Also, it was assumed that the dual variable λ_{ij} was applied at the same time by agent i and agent j, but this assumption should be possible to relax.

One of the most interesting applications for this optimization scheme is when the coupling between the state variables coincides with the communication topology. For example in multi-agent formation control, where each agent's position depends on the closest neighbors' positions, which are within wireless communication distance. Thus, in this case no multi-hop communication would be necessary, and the communication delays would be minimal. The problem includes determining an optimal state partitioning when associating states with agents.

As outlined here, many open research problems remain, and optimization and control in dynamical network systems continues to be an important research field.

Bibliography

"A dwarf standing on the shoulders of a giant may see farther than a giant himself."

— Robert Burton

Ravindra K. Ahuja, Thomas L. Magnanti, and James B. Orlin. *Network flows: theory, algorithms, and applications.* Prentice Hall, 1993.

Ian F. Akyildiz, Weilian Su, Yogesh Sankarasubramaniam, and Erdal Cayirci. Wireless sensor networks: a survey. *Computer networks*, 38(4):393–422, March 2002.

Assad Alam, Ather Gattami, and Karl Henrik Johansson. An experimental study on the fuel reduction potential of heavy duty vehicle platooning. In *13th IEEE Conference on Intelligent Transportation Systems*, pages 306–311, Madeira Island, Portugal, September 2010.

Assad Alam, Bart Besselink, Valerio Turri, Jonas Mårtensson, and Karl Henrik Johansson. Heavy-duty vehicle platooning for sustainable freight transportation: A cooperative method to enhance safety and efficiency. *IEEE Control Systems Magazine*, 35(6):34–56, 2015.

Réka Albert and Albert-László Barabási. Statistical mechanics of complex networks. *Reviews of Modern Physics*, 74(1):47–97, January 2002.

Masoom M. Ali, Manisha Pal, and Jungsoo Woo. On the ratio of inverted gamma variates. *Austrian Journal of Statistics*, 36(2):153–159, 2007.

Redouane Ali, Suksant Sae Lor, and Miguel Rio. Two algorithms for network size estimation for master/slave ad hoc networks. In *3rd IEEE International Symposium on Advanced Networks and Telecommunication Systems*, pages 1–3, December 2009.

Sara Alouf, Eitan Altman, and Philippe Nain. Optimal on-line estimation of the size of a dynamic multicast group. In *21st Annual Joint Conference of the IEEE Computer and Communications Societies*, pages 1109–1118, 2002.

Sara Alouf, Eitan Altman, Chadi Barakat, and Philippe Nain. On the dynamic estimation of multicast group sizes. In *16th International Symposium on Mathematical Theory of Networks and Systems*, Leuven, Belgium, 2004.

Sheng-hai An, Byung-Hyug Lee, and Dong-Ryeol Shin. A survey of intelligent transportation systems. In *3rd IEEE International Conference on Computational Intelligence, Communication Systems and Networks*, pages 332–337, July 2011.

Stephanos Androutsellis-Theotokis and Diomidis Spinellis. A survey of peer-to-peer content distribution technologies. *ACM Computing Surveys*, 36(4):335–371, December 2004.

Dana Angluin. Local and global properties in networks of processors. In *12th ACM Symposium on Theory of Computing*, pages 82–93, 1980.

Cosmin Arad, Jim Dowling, and Seif Haridi. Developing, simulating, and deploying peer-to-peer systems using the kompics component model. In *4th ACM International Conference on Communication System Software and Middleware*, Dublin, Ireland, 2009.

José Araújo. *Design, implementation and validation of resource-aware and resilient wireless networked control systems.* PhD thesis, KTH, 2014.

Karl J. Åström and Panganamala Ramana Kumar. Control: A perspective. *Automatica*, 50(1):3–43, January 2014.

Tuncer Can Aysal, Boris N. Oreshkin, and Mark J. Coates. Accelerated distributed average consensus via localized node state prediction. *IEEE Transactions on Signal Processing*, 57(4):1563–1576, April 2009.

Roberto Baldoni, Silvia Bonomi, A. Rippa, L. Querzoni, Sara Tucci Piergiovanni, and Antonino Virgillito. Evaluation of unstructured overlay maintenance protocols under churn. In *26th IEEE International Conference on Distributed Computing Systems Workshops*, pages 13–13, 2006.

Roberto Baldoni, Marco Platania, Leonardo Querzoni, and Sirio Scipioni. Practical uniform peer sampling under churn. In *9th International Symposium on Parallel and Distributed Computing*, pages 93–100. IEEE, 2010.

Carlos Baquero, Paulo Sérgio Almeida, Raquel Menezes, and Paulo Jesus. Extrema propagation: Fast distributed estimation of sums and network sizes. *IEEE Transactions on Parallel and Distributed Systems*, 23(4):668–675, 2012.

Albert-László Barabási. *Linked: The New Science of Networks.* Perseus Books Group, May 2002.

Albert-László Barabási. *Network Science.* Cambridge University Press, June 2016.

Matthew Barth and Kanok Boriboonsomsin. Real-world carbon dioxide impacts of traffic congestion. *Transportation Research Record: Journal of the Transportation Research Board*, 2058:163–171, December 2008.

Salman A. Baset and Henning G. Schulzrinne. An analysis of the Skype peer-to-peer internet telephony protocol. In *25th IEEE International Conference on Computer Communications*, pages 1–11, April 2006.

Zsuzsanna Bede, Géza Szabó, and Tamás Péter. Optimalization of road traffic with the applied of reversible direction lanes. *Periodica Polytechnica Transportation Engineering*, 38(1):3–8, 2010.

Richard Bellman. On the theory of dynamic programming. *Proceedings of the National Academy of Sciences*, 38:716–719, August 1952.

Bart Besselink, Valerio Turri, Sebastian van de Hoef, Kuo-Yun Liang, Assad Alam, Jonas Mårtensson, and Karl Henrik Johansson. Cyber–physical control of road freight transport. *Proceedings of the IEEE*, 104(5):1128–1141, 2016.

Vincent D. Blondel and Alexandre Megretski. *Unsolved Problems in Mathematical Systems and Control Theory.* Princeton University Press, July 2004.

Vincent D. Blondel, Julien M. Hendrickx, Alex Olshevsky, and John N. Tsitsiklis. Convergence in multiagent coordination, consensus, and flocking. In *44th IEEE Conference on Decision and Control*, pages 2996–3000, 2005.

Vincent D. Blondel, Julien M. Hendrickx, and John N. Tsitsiklis. On Krause's multi-agent consensus model with state-dependent connectivity. *IEEE Transactions on Automatic Control*, 54(11):2586–2597, 2009.

Hendrik Wade Bode. Relations between attenuation and phase in feedback amplifier design. *Bell System Technical Journal*, 19(3):421–454, July 1940.

Paolo Boldi and Sebastiano Vigna. An effective characterization of computability in anonymous networks. In *International Symposium on Distributed Computing*, pages 33–47. Springer, Berlin, Heidelberg, September 2001.

Saverio Bolognani and Sandro Zampieri. A distributed control strategy for reactive power compensation in smart microgrids. *IEEE Transactions on Automatic Control*, 58(11):2818–2833, November 2013.

Miguel Borges, Paulo Jesus, Carlos Baquero, and Paulo Sérgio Almeida. Spectra: Robust estimation of distribution functions in networks. *Distributed Applications and Interoperable Systems*, 7272:96–103, April 2012.

Maria Börjesson, Jonas Eliasson, Muriel B. Hugosson, and Karin Brundell-Freij. The Stockholm congestion charges—5 years on. Effects, acceptability and lessons learnt. *Transport Policy*, 20:1–12, March 2012.

Stephen P. Boyd and Lieven Vandenberghe. *Convex Optimization.* Cambridge University Press, 2009.

Stephen P. Boyd, Arpita Ghosh, Balaji Prabhakar, and Devavrat Shah. Randomized gossip algorithms. *IEEE Transactions on Information Theory*, 52(6): 2508–2530, June 2006.

Stephen P. Boyd, Neal Parikh, Eric Chu, Borja Peleato, and Jonathan Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, 3(1):1–122, January 2011.

Dietrich Braess, Anna Nagurney, and Tina Wakolbinger. On a paradox of traffic planning. *Transportation Science*, 39:446–450, November 2005.

Bureau of Transportation Statistics. *Freight Facts and Figures.* U.S. Department of Transportation, 2015.

Ming Cao, A. Stephen Morse, and Brian D. O. Anderson. Coordination of an asynchronous multi-agent system via averaging. *IFAC Proceedings Volumes*, 38 (1):17–22, 2005.

Jorge C. S. Cardoso, Carlos Baquero, and Paulo Sérgio Almeida. Probabilistic estimation of network size and diameter. In *4th Latin-American Symposium on Dependable Computing*, pages 33–40, João Pessoa, Brazil, September 2009. IEEE Computer Society.

Ruggero Carli, Fabio Fagnani, Paolo Frasca, and Sandro Zampieri. Gossip consensus algorithms via quantized communication. *Automatica*, 46(1):70–80, January 2010.

George Casella and Roger L. Berger. *Statistical inference.* Thomson Learning, 2002.

Yousra Chabchoub and Georges Hébrail. Sliding HyperLogLog: estimating cardinality in a data stream over a sliding window. In *10th IEEE International Conference on Data Mining Workshops*, pages 1297–1303, Sydney, Australia, December 2010.

Philippe Chassaing and Lucas Gerin. Efficient estimation of the cardinality of large data sets. In *4th Colloquium on Mathematics and Computer Science Algorithms*, pages 419–422, France, 2006. IECN.

Chen-Tung Chen. A fuzzy approach to select the location of the distribution center. *Fuzzy Sets and Systems*, 118(1):65–73, February 2001.

Siyao Cheng, Jianzhong Li, Qianqian Ren, and Lei Yu. Bernoulli sampling based (ε, δ)-approximate aggregation in large-scale sensor networks. In *29th IEEE International Conference on Computer Communications*, pages 1181–1189, San Diego, CA, March 2010.

Jacek Cichon, Jakub Lemiesz, and Marcin Zawada. On cardinality estimation protocols for wireless sensor networks. In *10th International Conference on Ad-hoc, Mobile, and Wireless Networks*, pages 322–331, Paderbom, Germany, July 2011. Springer Berlin Heidelberg.

Jacek Cichon, Jakub Lemiesz, Wojciech Szpankowski, and Marcin Zawada. Two-phase cardinality estimation protocols for sensor networks with provable precision. In *IEEE Wireless Communications and Networking Conference*, pages 2009–2013, 2012a.

Jacek Cichon, Jakub Lemiesz, and Marcin Zawada. On message complexity of extrema propagation techniques. In *11th International Conference on Ad-hoc, Mobile, and Wireless Networks*, pages 1–13, Belgrade, Serbia, July 2012b. Springer-Verlag.

Israel Cidon and Yuval Shavitt. Message terminating algorithms for anonymous rings of unknown size. *Information Processing Letters*, 54(2):111–119, April 1995.

Cisco. Cisco visual networking index: Forecast and methodology, 2014-2019. Technical report, May 2015.

Peter Clifford and Ioana A. Cosma. A statistical analysis of probabilistic counting algorithms. *Scandinavian Journal of Statistics*, 39:1–14, 2012.

Bruno Codenotti, P. Gemmell, P. Pudlak, and J. Simon. On the amount of randomness needed in distributed computations. Technical report, September 1997.

Edith Cohen. Size-estimation framework with applications to transitive closure and reachability. *Journal of Computer and System Sciences*, 55(3):441–453, December 1997.

Guy Cohen. Auxiliary problem principle and decomposition of optimization problems. *Journal of Optimization Theory and Applications*, 32(3):277–305, November 1980.

Giacomo Como and Fabio Fagnani. From local averaging to emergent global behaviors: The fundamental role of network interconnections. *Systems & Control Letters*, 95:70–76, September 2016.

Teodor Gabriel Crainic and Gilbert Laporte. Planning models for freight transportation. *European Journal of Operational Research*, 97(3):409–438, March 1997.

Teodor Gabriel Crainic and Gilbert Laporte, editors. *Fleet Management and Logistics*. Springer, 1998.

George B. Dantzig, Alex Orden, and Philip Wolfe. The generalized simplex method for minimizing a linear form under linear inequality restraints. *Pacific Journal of Mathematics*, 5(2):183–195, 1955.

Herbert A. David and H. N. Nagaraja. *Order Statistics*. John Wiley & Sons, April 2004.

Arturo Davila and Mario Nombela. Platooning – safe and eco-friendly mobility. In *SAE 2012 World Congress & Exhibition*, pages 2012–01–0488, Warrendale, PA, USA, April 2012.

Pierre J. Dejax and Teodor Gabriel Crainic. A review of empty flows and fleet management models in freight transportation. *Transportation Science*, 21(4): 227–248, November 1987.

Alberto Devoto and Dennis W. Duke. Table of integrals and formulae for Feynman diagram calculations. *La Rivista Del Nuovo Cimento*, 7(6):1–39, June 1984.

Reinhard Diestel. *Graph theory*. Springer Verlag, 2005.

Danny Dolev, Osnat Mokryn, and Yuval Shavitt. On multicast trees: Structure and size estimation. *IEEE/ACM Transactions on Networking*, 14(3):557–567, June 2006.

John C Duchi, Alakh Agarwal, and Martin J. Wainwright. Dual averaging for distributed optimization: Convergence analysis and network scaling. *IEEE Transactions on Automatic Control*, 57(3):592–606, March 2012.

Jack Edmonds and Richard M. Karp. Theoretical improvements in algorithmic efficiency for network flow problems. *Journal of the ACM*, 19(2):248–264, April 1972.

Paul Erdős and Alfréd Rényi. On random graphs I. *Publicationes Mathematicae (Debrecen)*, 6:290–297, 1959.

Leonhard Euler. Solutio problematis ad geometriam situs pertinentis. *Commentarii academiae scientiarum Petropolitanae*, 8:128–140, 1741.

European Union. *EU Transport in Figures*. European Union, 2014.

European Union. *Energy, transport and environment indicators*. European Union, 2015.

Fabio Fagnani and Sandro Zampieri. Randomized consensus algorithms over large scale networks. *IEEE Journal on Selected Areas in Communications*, 26(4):634–649, May 2008.

Alexander J. Fax and Richard M. Murray. Graph laplacians and stabilization of vehicle formations. In *IFAC Proceedings Volumes*, pages 55–60, 2002.

Alexander J. Fax and Richard M. Murray. Information flow and cooperative control of vehicle formations. *IEEE Transactions on Automatic Control*, 49(9):1465–1476, 2004.

Lino Figueiredo, Isabel Jesus, J. A. Tenreiro Machado, Jose Rui Ferreira, and J. L. Martins de Carvalho. Towards the development of intelligent transportation systems. In *4th Intelligent Transportation Systems*, pages 1206–1211. IEEE, 2001.

Philippe Flajolet, Éric Fusy, Olivier Gandouet, and Frédéric Meunier. HyperLogLog : the analysis of a near-optimal cardinality estimation algorithm. In *Conference on Analysis of Algorithms*, pages 127–146, Juan des Pins, France, 2007.

Lester Randolph Ford and Delbert Ray Fulkerson. A simple algorithm for finding maximal network flows and an application to the Hitchcock problem. *Canadian Journal of Mathematics*, pages 210–218, 1957.

Pedro A. Forero, Alfonso Cano, and Georgios B. Giannakis. Consensus-based distributed expectation-maximization algorithm for density estimation and classification using wireless sensor networks. In *IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages 1989–1992, Las Vegas, NV, USA, 2008.

Éric Fusy and Frédéric Giroire. Estimating the number of active flows in a data stream over a sliding window. In *Workshop on Analytic Algorithmics and Combinatorics*, 2007.

Daniel Gabay and Bertrand Mercier. A dual algorithm for the solution of nonlinear variational problems via finite element approximation. *Computers & Mathematics with Applications*, 2(1):17–40, 1976.

Michael R. Garey and David S. Johnson. *Computers and Intractability.* A Guide to the Theory of NP-Completeness. W. H. Freeman & Company, January 1979.

Federica Garin, Damiano Varagnolo, and Karl Henrik Johansson. Distributed estimation of diameter, radius and eccentricities in anonymous networks. In *3rd IFAC Workshop on Distributed Estimation and Control in Networked Systems*, 2012.

Győző Gidófalvi and Torben Bach Pedersen. Mining long, sharable patterns in trajectories of moving objects. *GeoInformatica*, 13(1):27–55, December 2007.

Edgar Nelson Gilbert. Random graphs. *The Annals of Mathematical Statistics*, 30 (4):1141–1144, December 1959.

Frédéric Giroire. Order statistics and estimating cardinalities of massive data sets. *Discrete Applied Mathematics*, 157(2):406–427, January 2009.

Christos Gkantsidis, Milena Mihail, and Amin Saberi. Random walks in peer-to-peer networks: Algorithms and evaluation. *Performance Evaluation*, 63(3): 241–263, March 2006.

Dorothy J. Glancy. Privacy and intelligent transportation technology. *Santa Clara Computer & High Technology Law Journal*, 11(1):151–203, January 1995.

Chris Godsil and Gordon F. Royle. *Algebraic Graph Theory*. New York : Springer, 2001.

Asvin Goel and Leendert Kok. Efficient scheduling of team truck drivers in the European Union. *Flexible Services and Manufacturing Journal*, 24(1):81–96, March 2012.

Andrew Goldberg and Robert Tarjan. Finding minimum-cost circulations by canceling negative cycles. In *20th ACM Symposium on Theory of Computing*, January 1988.

Bruce D. Greenshields. A study of traffic capacity. *Highway research board proceedings*, 14:448–477, 1935.

Michael B. Greenwald and Sanjeev Khanna. Power-conserving computation of order-statistics over sensor networks. In *23rd ACM Symposium on Principles of Database Systems*, pages 275–285, Paris, France, June 2004.

Charles M. Grinstead and James Laurie Snell. *Introduction to probability*. American Mathematical Society, 1997.

Michael M. Grynbaum. Where do all the cabs go in the late afternoon? *The New York Times*, January 2011.

Krishna P. Gummadi, Stefan Saroiu, and Steven D. Gribble. King: Estimating latency between arbitrary internet end hosts. In *2nd ACM SIGCOMM Workshop on Internet Measurement*, pages 5–18, 2002.

Meng Guo and Dimos V. Dimarogonas. Consensus with quantized relative state measurements. *Automatica*, 49(8):2531–2537, August 2013.

Oleg Gusikhin, Dimitar Filev, and Nestor Rychtyckyj. Intelligent vehicle systems: Applications and new trends. In *Informatics in Control Automation and Robotics*, pages 3–14. Springer, Berlin, Heidelberg, 2008.

Matthew Hale and Magnus Egerstedt. Cloud-enabled differentially private multi-agent optimization with constraints. *arXiv.org*, pages –, July 2015.

John H. Halton. A retrospective and prospective survey of the Monte Carlo method. *SIAM review*, 12, January 1970.

Maya Haridasan and Robbert van Renesse. Gossip-based distribution estimation in peer-to-peer networks. In *7th International Workshop on Peer-to-Peer Systems*, 2008.

Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning.* Data Mining, Inference, and Prediction. Springer, 2 edition, August 2009.

Ryan Herring, Aude Hofleitner, Saurabh Amin, Tania Abou Nasr, Amin Abdel Khalek, Pieter Abbeel, and Alexandre Bayen. Using mobile phones to forecast arterial traffic through statistical learning. In *89th Annual Meeting of the Transportation Research Board*, August 2009.

Jean-Baptiste Hiriart-Urruty and Claude Lemaréchal. *Fundamentals of Convex Analysis.* Springer, Berlin, Heidelberg, 2001.

Frank L. Hitchcock. The distribution of a product from several sources to numerous localities. *Journal of Mathematical Physics*, 20(1):224–230, April 1941.

Roberto Horowitz and Pravin Varaiya. Control design of an automated highway system. *Proceedings of the IEEE*, 88(7):913–925, 2000.

Shafiul Azam Howlader, Michael R. Frater, and Michael J. Ryan. Estimating the number and distribution of the neighbors in an underwater communication network. In *Second International Conference on Sensor Technologies and Applications*, pages 693–698, August 2008.

Yusuo Hu, Jian-Guang Lou, Hua Chen, and Jiang Li. Distributed density estimation using non-parametric statistics. In *27th IEEE International Conference on Distributed Computing Systems*, Toronto, ON, 2007.

Kent Hymel. Does traffic congestion reduce employment growth? *Journal of Urban Economics*, 65(2):127–135, March 2009.

Alon Itai and Michael Rodeh. Symmetry breaking in distributed networks. *Information and Computation*, 1990.

Franck Iutzeler, Philippe Ciblat, and Jérémie Jakubowicz. Analysis of max-consensus algorithms in wireless channels. *IEEE Transactions on Signal Processing*, 60(11):6103–6107, August 2012.

Márk Jelasity, Rachid Guerraoui, Anne-Marie Kermarrec, and Maarten van Steen. *The peer sampling service: experimental evaluation of unstructured gossip-based implementations.* Springer-Verlag New York, October 2004.

Márk Jelasity, Spyros Voulgaris, Rachid Guerraoui, Anne-Marie Kermarrec, and Maarten van Steen. Gossip-based peer sampling. *ACM Transactions on Computer Systems*, 25(3), August 2007.

Márk Jelasity, Alberto Montresor, and Ozalp Babaoglu. T-Man: Gossip-based fast overlay topology construction. *Computer networks*, 53(13):2321–2339, 2009.

Erik Jenelius and Haris N. Koutsopoulos. Travel time estimation for urban road networks using low frequency probe vehicle data. *Transportation Research Part B: Methodological*, 53:64–81, July 2013.

Meng Ji and Magnus Egerstedt. Distributed coordination control of multiagent systems while preserving connectedness. *IEEE Transactions on Robotics*, 23(4): 693–703, 2007.

Hongbo Jiang and Shudong Jin. Scalable and robust aggregation techniques for extracting statistical information in sensor networks. In *26th International Conference on Distributed Computing Systems*, Lisbon, Portugal, July 2006.

Björn Johansson. *On Distributed Optimization in Networked Systems*. PhD thesis, KTH, 2008.

Anne-Marie Kermarrec and Maarten van Steen. Gossiping in distributed systems. *ACM SIGOPS Operating Systems Review*, 41(5):2–7, October 2007.

Matthias Klusch, Stefano Lodi, and Gianluca Moro. Distributed clustering based on sampling local density estimates. In *18th International Joint Conference on Artificial Intelligence*, pages 485–490, 2003.

Tjalling C. Koopmans. Optimum utilization of the transportation system. *Econometrica*, 17:136–146, 1949.

Dionysios Kostoulas, Dimitrios Psaltoulis, Indranil Gupta, Kenneth P. Birman, and Alan J. Demers. Active and passive techniques for group size estimation in large-scale and dynamic distributed systems. *Journal of Systems and Software*, 80(10): 1639–1658, October 2007.

Péter Kovács. Minimum-cost flow algorithms: an experimental evaluation. *Optimization Methods and Software*, 30(1):94–127, 2015.

Gunnar Kreitz and Fredrik Niemelä. Spotify - large scale, low latency, P2P music-on-demand streaming. In *10th IEEE International Conference on Peer-to-Peer Computing*, Delft, Netherlands, 2010.

Fabian Kuhn, Stefan Schmid, and Roger Wattenhofer. A self-repairing peer-to-peer system resilient to dynamic adversarial churn. In *4th International Workshop on Peer-to-Peer Systems*, pages 13–23, Berlin, Heidelberg, 2005. Springer.

Jaimyoung Kwon and Pravin Varaiya. Effectiveness of California's high occupancy vehicle (HOV) system. *Transportation Research Part C: Emerging Technologies*, 16(1):98–115, February 2008.

Kee-hung Lai and T.C.E. Cheng. *Just-in-time Logistics*. Gower, 2009.

Junghoon Lee, Inhye Shin, and Gyung-Leen Park. Analysis of the passenger pick-up pattern for taxi location recommendation. In *4th International Conference on Networked Computing and Advanced Information Management*, pages 199–204. IEEE, September 2008.

Amir Leshem and Lang Tong. Estimating sensor population via probabilistic sequential polling. *IEEE Signal Processing Letters*, 12(5):395–398, May 2005.

David A. Levin, Yuval Peres, and Elizabeth L. Wilmer. *Markov Chains and Mixing Times.* American Mathematical Society, 2009.

Li Li, Ding Wen, and Danya Yao. A survey of traffic control with vehicular communications. *IEEE Transactions on Intelligent Transportation Systems*, 15(1): 425–432, 2014.

Kuo-Yun Liang and Karl Henrik Johansson. Fuel-saving potentials of platooning evaluated through sparse heavy-duty vehicle position data. In *IEEE Intelligent Vehicles Symposium*, pages 1061–1068, 2014.

Kuo-Yun Liang, Sebastian van de Hoef, Håkan Terelius, Valerio Turri, Bart Besselink, Jonas Mårtensson, and Karl Henrik Johansson. Networked control challenges in collaborative road freight transport. *European Journal of Control*, 30:2–14, May 2016.

Ji Liu, Brian D. O. Anderson, Ming Cao, and A. Stephen Morse. Analysis of accelerated gossip algorithms. In *48th IEEE Conference on Decision and Control and 28th Chinese Control Conference*, pages 871–876, Shanghai, 2009.

Yong Liu, Yang Guo, and Chao Liang. A survey on peer-to-peer video streaming systems. *Peer-to-Peer Networking and Applications*, 1(1):18–28, January 2008.

Riccardo Lucchese, Damiano Varagnolo, Jean-Charles Delvenne, and Julien M. Hendrickx. Network cardinality estimation using max consensus: The case of Bernoulli trials. In *54th IEEE Conference on Decision and Control*, pages 895–901, 2015.

Jérémie Lumbroso. An optimal cardinality estimation algorithm based on order statistics and its full analysis. In *21st International Meeting on Probabilistic, Combinatorial, and Asymptotic Methods in the Analysis of Algorithms*, pages 489–504, 2010.

Kevin M. Lynch, Ira B. Schwartz, Peng Yang, and Randy A. Freeman. Decentralized environmental modeling by mobile sensor networks. *IEEE Transactions on Robotics*, 24(3):710–724, 2008.

Nancy A. Lynch. *Distributed Algorithms.* Morgan Kaufmann, April 1996.

Samuel Madden, Michael J. Franklin, Joseph M. Hellerstein, and Wei Hong. TAG: a Tiny AGgregation service for ad-hoc sensor networks. In *5th Symposium on Operating Systems Design and Implementation*, pages 131–146, 2002.

Van Sy Mai and Eyad H. Abed. Distributed optimization over weighted directed graphs using row stochastic matrix. In *2016 American Control Conference*, pages 7165–7170. IEEE, 2016.

Christoph E. Mandl. Evaluation and optimization of urban public transportation networks. *European Journal of Operational Research*, 5(6):396–404, December 1980.

Jonas Mårtensson, Assad Alam, Sagar Behere, Muhammad Altamash Ahmed Khan, Joakim Kjellberg, Kuo-Yun Liang, Henrik Pettersson, and Dennis Sundman. The development of a cooperative heavy-duty vehicle for the GCDC 2011: Team scoop. *IEEE Transactions on Intelligent Transportation Systems*, 13(3): 1033–1049, 2012.

Laurent Massoulie, Erwan Le Merrer, Anne-Marie Kermarrec, and Ayalvadi Ganesh. Peer counting and sampling in overlay networks: Random walk methods. In *25th ACM Symposium on Principles of Distributed Computing*, pages 123–132, July 2006.

Yasuko Matsubara, Lei Li, Evangelos Papalexakis, David Lo, Yasushi Sakurai, and Christos Faloutsos. F-trail: Finding patterns in taxi trajectories. In *17th Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 86–98, April 2013.

James Clerk Maxwell. On governors. *Proceedings of the Royal Society of London*, 16:270–283, February 1868.

David Quinn Mayne, James Blake Rawlings, Christopher V. Rao, and Pierre O. M. Scokaert. Constrained model predictive control: Stability and optimality. *Automatica*, 36(6):789–814, June 2000.

Martijn Mes, Matthieu Van Der Heijden, and Aart Van Harten. Comparison of agent-based scheduling to look-ahead heuristics for real-time transportation problems. *European Journal of Operational Research*, 181(1):59–75, August 2007.

Elena Meshkova, Janne Riihijärvi, Marina Petrova, and Petri Mähönen. A survey on resource discovery mechanisms, peer-to-peer and service discovery frameworks. *Computer Networks*, 52(11):2097–2128, August 2008.

Nicholas Metropolis and Stanislaw Ulam. The Monte Carlo method. *Journal of the American Statistical Association*, 44(247):335–341, 1949.

Milena Mihail, Christos Papadimitriou, and Amin Saberi. On certain connectivity properties of the internet topology. In *44th Annual IEEE Symposium on Foundations of Computer Science*, pages 28–35, Washington, DC, USA, October 2003.

Luis Moreira-Matias, Ricardo Fernandes, Joao Gama, Michel Ferreira, Joao Mendes-Moreira, and Luis Damas. An online recommendation system for the taxi stand choice problem (Poster). In *Vehicular Networking Conference*, pages 173–180. IEEE, 2012.

Damon Mosk-Aoyama and Devavrat Shah. Fast distributed algorithms for computing separable functions. *IEEE Transactions on Information Theory*, 54(7): 2997–3007, 2008.

Shinji Motegi, Kiyohito Yoshihara, and Hiroki Horiuchi. DAG based in-network aggregation for sensor network monitoring. In *International Symposium on Applications and the Internet*, pages 292–299, Phoenix, AZ, USA, 2006.

Luc Muyldermans, Dirk Cattrysse, Dirk Van Oudheusden, and Tsippy Lotan. Districting for salt spreading operations. *European Journal of Operational Research*, 139(3):521–532, June 2002.

Farid Movahedi Naini, Olivier Dousse, Patrick Thiran, and Martin Vetterli. Opportunistic sampling for joint population size and density estimation. *IEEE Transactions on Mobile Computing*, 14(12):2530–2543, 2015.

Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system. *Cryptovest*, October 2008.

Angelia Nedić and Dimitri P. Bertsekas. Incremental subgradient methods for nondifferentiable optimization. *SIAM Journal on Optimization*, 2001.

Angelia Nedić and Alex Olshevsky. Distributed optimization over time-varying directed graphs. *IEEE Transactions on Automatic Control*, 60(3):601–615, 2015.

Angelia Nedić and Asuman Ozdaglar. On the rate of convergence of distributed subgradient methods for multi-agent optimization. In *46th IEEE Conference on Decision and Control*, pages 4711–4716, 2007.

Angelia Nedić, Asuman Ozdaglar, and Pablo A. Parrilo. Constrained consensus and optimization in multi-agent networks. *IEEE Transactions on Automatic Control*, 55(4):922–938, April 2010.

Yurii Nesterov. Smooth minimization of non-smooth functions. *Mathematical Programming*, 103(1):127–152, 2005.

Yurii Nesterov. Primal-dual subgradient methods for convex problems. *Mathematical Programming*, 120(1):221–259, 2009.

Yurii Nesterov and Arkadii Nemirovskii. *Interior-Point Polynomial Algorithms in Convex Programming.* SIAM, 1994.

XuanLong Nguyen, Martin J. Wainwright, and Michael I. Jordan. Nonparametric decentralized detection using kernel methods. *IEEE Transactions on Signal Processing*, 53(11):4053–4066, November 2005.

Robert Nowak. Distributed EM algorithms for density estimation and clustering in sensor networks. *IEEE Transactions on Signal Processing*, 51(8):2245–2253, August 2003.

Harry Nyquist. Regeneration theory. *Bell System Technical Journal*, 11(1):126–147, January 1932.

Reza Olfati-Saber and Richard M. Murray. Consensus problems in networks of agents with switching topology and time-delays. *IEEE Transactions on Automatic Control*, 49(9):1520–1533, 2004.

Reza Olfati-Saber, Alexander J. Fax, and Richard M. Murray. Consensus and cooperation in networked multi-agent systems. *Proceedings of the IEEE*, 95(1): 215–233, January 2007.

Alex Olshevsky and John N. Tsitsiklis. Convergence rates in distributed consensus and averaging. In *45th IEEE Conference on Decision and Control*, pages 3387–3392, 2006.

OpenStreetMap. OpenStreetMap – The Free Wiki World Map. `http://www.openstreetmap.org/copyright`, 2015. © OpenStreetMap contributors.

Amir H. Payberah, Jim Dowling, Fatemeh Rahimian, and Seif Haridi. gradienTv: Market-based P2P live media streaming on the gradient overlay. *Distributed Applications and Interoperable Systems*, 6115:212–225, 2010a.

Amir H. Payberah, Fatemeh Rahimian, Seif Haridi, and Jim Dowling. Sepidar: Incentivized market-based P2P live-streaming on the gradient overlay network. In *12th IEEE International Symposium on Multimedia*, pages 1–8, 2010b.

Amir H. Payberah, Jim Dowling, and Seif Haridi. Gozar: NAT-friendly peer sampling with one-hop distributed NAT traversal. In *6th International Federated Conferences on Distributed Computing Techniques*, pages 1–14, Berlin, Heidelberg, 2011. Springer.

Shao-Liang Peng, Shan-Shan Li, Xiang-Ke Liao, Yu-Xing Peng, and Nong Xiao. Estimation of a population size in large-scale wireless sensor networks. *Journal of Computer Science and Technology*, 24(5):987–997, September 2009.

Jossef Perl and Mark S. Daskin. A warehouse location-routing problem. *Transportation Research Part B: Methodological*, 19(5):381–396, October 1985.

Sanja Petrovic and Patrick Brown. A new statistical approach to estimate global file populations in the eDonkey P2P file sharing system. In *21st International Teletraffic Congress*, pages 1–8, Paris, France, 2009.

Lev Semenovich Pontryagin, Vladimir Grigorevich Boltyansky, Revaz Valerianovic Gamkrelidze, and E. F. Mishchenko. *The mathematical theory of optimal processes.* Wiley, New York, NY, 1962.

Dimitrios Psaltoulis, Dionysios Kostoulas, Indranil Gupta, Kenneth P. Birman, and Alan J. Demers. Practical algorithms for size estimation in large and dynamic groups. Technical report, University of Illinois, Urbana-Champaign, 2004.

Michael Rabbat and Robert Nowak. Distributed optimization in sensor networks. In *Information Processing in Sensor Networks*, pages 20–27, New York, USA, April 2004. ACM.

Paraskevi Raftopoulou and Euripides G. M. Petrakis. Peer rewiring in semantic overlay networks under churn. In *On the Move to Meaningful Internet Systems: OTM 2010*, pages 573–581. Springer Berlin Heidelberg, Berlin, Heidelberg, 2010.

Amirreza Rahmani, Meng Ji, Mehran Mesbahi, and Magnus B. Egerstedt. Controllability of multi-agent systems from a graph-theoretic perspective. *SIAM Journal on Control and Optimization*, 48:162–186, 2009.

Srinivasan Sundhar Ram, Angelia Nedić, and Venugopal V. Veeravalli. Distributed stochastic subgradient projection algorithms for convex optimization. *Journal of Optimization Theory and Applications*, 147(3):516–545, July 2010.

Tal Raviv, Michal Tzur, and Iris A. Forma. Static repositioning in a bike-sharing system: models and solution approaches. *EURO Journal on Transportation and Logistics*, 2(3):187–229, January 2013.

Bruno Ribeiro and Don Towsley. Estimating and sampling graphs with multidimensional random walks. In *10th ACM SIGCOMM Conference on Internet Measurement*, pages 390–403, Melbourne, Australia, November 2010.

John Risson and Tim Moors. Survey of research towards robust peer-to-peer networks: Search methods. *Computer networks*, 50(17):3485–3521, December 2006.

Jan Sacha. *Exploiting Heterogeneity in Peer-to-Peer Systems using Gradient Topologies.* PhD thesis, Trinity College, Dublin, 2009.

Jan Sacha, Jeff Napper, Corina Stratan, and Guillaume Pierre. Adam2: Reliable distribution estimation in decentralised environments. In *30th International Conference on Distributed Computing Systems*, pages 697–707, June 2010.

Tariq Samad, Paul McLaughlin, and Joseph Lu. System architecture for process automation: Review and trends. *Journal of Process Control*, 17(3):191–201, 2007.

SCB. *Folkmängd i riket, län och kommuner.* Statistiska centralbyrån (SCB), 2015.

Bernhard Schölkopf and Alexander J. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond.* MIT Press, 2002.

Georg S. Seyboth, Dimos V. Dimarogonas, and Karl Henrik Johansson. Event-based broadcasting for multi-agent average consensus. *Automatica*, 49(1):245–252, January 2013.

Tallat M. Shafaat, Ali Ghodsi, and Seif Haridi. A practical approach to network size estimation for structured overlays. In *Self-organizing Systems*, pages 71–83. Springer Berlin Heidelberg, Vienna, Austria, 2008.

Naum Zuselevich Shor. Generalized gradient methods of nondifferentiable optimization employing space dilatation operations. In *Mathematical Programming: The State of the Art*, pages 501–529. Springer, Berlin, Heidelberg, 1983.

Nisheeth Shrivastava, Chiranjeeb Buragohain, Divyakant Agrawal, and Subhash Suri. Medians and beyond: New aggregation techniques for sensor networks. In *2nd International Conference on Embedded Networked Sensor Systems*, pages 239–249, 2004.

Daniel Silvestre, Paulo Rosa, Rita Cunha, Joao P. Hespanha, and Carlos Silvestre. Gossip average consensus in a byzantine environment using stochastic set-valued observers. In *52nd IEEE Conference on Decision and Control*, pages 4373–4378, 2013.

Jan A. Snyman. *Practical Mathematical Optimization*, volume 97 of *An Introduction to Basic Optimization Theory and Classical and New Gradient-Based Algorithms.* Springer-Verlag, New York, 2005.

Alberto Speranzon, Carlo Fischione, and Karl Henrik Johansson. Distributed and collaborative estimation over wireless sensor networks. In *45th IEEE Conference on Decision and Control*, pages 1025–1030, 2006.

Kunal Srivastava and Angelia Nedić. Distributed asynchronous constrained stochastic optimization. *IEEE Journal of Selected Topics in Signal Processing*, 5(4): 772–790, 2011.

Daniel Stutzbach and Reza Rejaie. Understanding churn in peer-to-peer networks. In *6th ACM Conference on Internet Measurement*, pages 189–202, New York, USA, 2006.

Yuan Gong Sun, Long Wang, and Guangming Xie. Average consensus in networks of dynamic agents with switching topologies and multiple time-varying delays. *Systems & Control Letters*, 57(2):175–183, February 2008.

André Teixeira, Euhanna Ghadimi, Iman Shames, Henrik Sandberg, and Mikael Johansson. The ADMM algorithm for distributed quadratic problems: Parameter selection and constraint preconditioning. *IEEE Transactions on Signal Processing*, 64(2):290–305, 2016.

Håkan Terelius and Karl Henrik Johansson. An efficiency measure for road transportation networks with application to two case studies. In *54th IEEE Conference on Decision and Control*, pages 5149–5155, December 2015.

Håkan Terelius and Karl Henrik Johansson. Peer-to-peer gradient topologies in networks with churn. *IEEE Transactions on Control of Network Systems*, 2016a. (submitted).

Håkan Terelius and Karl Henrik Johansson. Efficiency modeling and optimization for road transportation. *Transportation Research Part B: Methodological*, 2016b. (submitted).

Håkan Terelius and Karl Henrik Johansson. On the optimal location of distribution centers for a one-dimensional transportation system. In *55th IEEE Conference on Decision and Control*, Las Vegas, USA, December 2016c. (to appear).

Håkan Terelius, Guodong Shi, Jim Dowling, Amir H. Payberah, Ather Gattami, and Karl Henrik Johansson. Converging an overlay network to a gradient topology. In *50th IEEE Conference on Decision and Control and European Control Conference*, pages 7230–7235, Orlando, FL, USA, December 2011a.

Håkan Terelius, Ufuk Topcu, and Richard M. Murray. Decentralized multi-agent optimization via dual decomposition. In *18th IFAC World Congress*, pages 11245–11251, Milan, Italy, August 2011b.

Håkan Terelius, Damiano Varagnolo, and Karl Henrik Johansson. Distributed size estimation of dynamic anonymous networks. In *51st IEEE Conference on Decision and Control*, pages 5221–5227, Maui, HI, USA, December 2012.

Håkan Terelius, Guodong Shi, and Karl Henrik Johansson. Consensus control for multi-agent systems with a faulty node. In *4th IFAC Workshop on Distributed Estimation and Control in Networked Systems*, pages 425–432, Koblenz, Germany, September 2013a.

Håkan Terelius, Damiano Varagnolo, Carlos Baquero, and Karl Henrik Johansson. Fast distributed estimation of empirical mass functions over anonymous networks. In *52nd IEEE Conference on Decision and Control*, pages 6771–6777, Florence, Italy, December 2013b.

Sabu M. Thampi. A review on P2P video streaming. *arXiv.org*, April 2013.

Sergio L. Toral, Rocio Martinez-Torres, Federico Barrero, and Manuel R. Arahal. Current paradigms in intelligent transportation systems. *IET Intelligent Transport Systems*, 4(3):201–211, 2010.

Kyle Treleaven, Marco Pavone, and Emilio Frazzoli. Asymptotically optimal algorithms for one-to-one pickup and delivery problems with applications to transportation systems. *IEEE Transactions on Automatic Control*, 58(9):2261–2276, 2013.

Konstantinos I. Tsianos and Michael G. Rabbat. Distributed strongly convex optimization. In *50th Annual Allerton Conference on Communication, Control, and Computing*, pages 593–600. IEEE, October 2012.

John N. Tsitsiklis. *Problems in Decentralized Decision making and Computation.* PhD thesis, MIT, 1984.

Lieven Vandenberghe and Stephen P. Boyd. Semidefinite programming. *SIAM review*, 38:49–95, March 1996.

Vladimir Naumovich Vapnik. *Statistical learning theory.* Wiley-Interscience, New York, 1998.

Damiano Varagnolo, Gianluigi Pillonetto, and Luca Schenato. Distributed statistical estimation of the number of nodes in sensor networks. In *49th IEEE Conference on Decision and Control*, pages 1498–1503, December 2010.

Damiano Varagnolo, Gianluigi Pillonetto, and Luca Schenato. Distributed cardinality estimation in anonymous networks. *IEEE Transactions on Automatic Control*, October 2013.

Pravin Varaiya. Smart cars on smart roads: problems of control. *IEEE Transactions on Automatic Control*, 38(2):195–207, 1993.

Nikos Vlassis, Yiannis Sfakianakis, and Wojtek Kowalczyk. Gossip-based greedy gaussian mixture learning. In *Advances in Informatics*, pages 349–359. Springer, Berlin, Heidelberg, 2005.

Grace Wahba. *Spline Models for Observational Data.* SIAM, September 1990.

Chenggang Wang, Wee Keong Ng, and Huaixin Chen. From data to knowledge to action: A taxi business intelligence system. In *15th International Conference on Information Fusion*, pages 1623–1628, 2012.

Feng Wang, Jiangchuan Liu, and Yongqiang Xiong. Stable peers: Existence, importance, and application in peer-to-peer live video streaming. In *27th IEEE International Conference on Computer Communications*, pages 1364–1372, 2008.

Ermin Wei and Asuman Ozdaglar. On the O(1=k) convergence of asynchronous distributed alternating direction method of multipliers. In *IEEE Global Conference on Signal and Information Processing*, pages 551–554, 2013.

Viktor Witkovský. Computing the distribution of a linear combination of inverted gamma variables. *Kybernetika*, 37(1):79–90, 2001.

Lin Xiao and Stephen P. Boyd. Fast linear iterations for distributed averaging. In *42nd IEEE Conference on Decision and Control*, pages 4997–5002, 2003.

Lin Xiao, Stephen P. Boyd, and Sanjay Lall. A scheme for robust distributed sensor fusion based on average consensus. In *4th International Symposium on Information Processing in Sensor Networks*, pages 63–70. IEEE, 2005.

Lin Xiao, Stephen P. Boyd, and Sanjay Lall. Distributed average consensus with time-varying metropolis weights. *Automatica*, 2006.

Masafumi Yamashita and Tsunehiko Kameda. Computing on anonymous networks. *7th Annual ACM Symposium on Principles of Distributed Computing*, pages 117–130, 1988.

Masafumi Yamashita and Tsunehiko Kameda. Computing on anonymous networks: Part II - Decision and membership problems. *IEEE Transactions on Parallel and Distributed Systems*, 7(1):90–96, 1996a.

Masafumi Yamashita and Tsunehiko Kameda. Computing on anonymous networks: Part I - Characterizing the solvable cases. *IEEE Transactions on Parallel and Distributed Systems*, 7(1):69–89, 1996b.

Jian Yang, Patrick Jaillet, and Hani Mahmassani. Real-time multivehicle truckload pickup and delivery problems. *Transportation Science*, pages 135–148, May 2004.

Nicholas Jing Yuan, Yu Zheng, Liuhang Zhang, and Xing Xie. T-finder: A recommender system for finding passengers and vacant taxis. *IEEE Transactions on Knowledge and Data Engineering*, 25(10):2390–2403, October 2013.

Ye Yuan, Guy-Bart Stan, Mauricio Barahona, Ling Shi, and Jorge Gonçalves. Decentralised minimal-time consensus. In *50th IEEE Conference on Decision and Control and European Control Conference*, pages 4282–4289, Orlando, FL, December 2011.

Shelemyahu Zacks. *The theory of statistical inference.* Wiley, 1971.

Michael Zargham, Alejandro Ribeiro, Asuman Ozdaglar, and Ali Jadbabaie. Accelerated dual descent for network optimization. In *American Control Conference*, 2011.

Xianyuan Zhan, Xinwu Qian, and Satish V. Ukkusuri. Measuring the efficiency of urban taxi service system. In *3rd International Workshop on Urban Computing*, New York City, USA, August 2014.

Junping Zhang, Fei-Yue Wang, Kunfeng Wang, Wei-Hua Lin, Xin Xu, and Cheng Chen. Data-driven intelligent transportation systems: A survey. *IEEE Transactions on Intelligent Transportation Systems*, 12(4):1624–1639, 2011.

Ruiliang Zhang and James T. Kwok. Asynchronous distributed ADMM for consensus optimization. *ICML*, 2014.

Xiaohua Zhao and Yangzhou Chen. Traffic light control method for a single intersection based on hybrid systems. In *IEEE International Conference on Intelligent Transportation Systems*, pages 1105–1109, 2003.

Yu Zheng, Licia Capra, Ouri Wolfson, and Hai Yang. Urban computing: Concepts, methodologies, and applications. *Transactions on Intelligent Systems and Technology*, 5(3), September 2014.