Ph.D. in Computer Science – XXXII Cycle
Doctoral Thesis

Simple Randomized Distributed Algorithms for Graph Clustering

Emilio Cruciani

Supervisors
Luca Becchetti
Emanuele Natale

Internal Advisor
Gianlorenzo D'Angelo

A thesis submitted in fulfillment of the requirements for the degree of Doctor of Philosophy at the

Gran Sasso Science Institute
Viale Francesco Crispi 7, 67100, L'Aquila, Italy

Declaration

I, Emilio Cruciani, declare that most of the contents of this thesis titled "Simple Randomized Distributed Algorithms for Graph Clustering" are based on co-authored papers published in Proceedings of Italian and International Conferences (ICTCS, AAMAS, AAAI, ISAAC) and in the Bulletin of the EATCS:

I confirm that:

- The content of Chapter 5 is based on works co-authored by Emanuele Natale, André Nusser, and Giacomo Scornavacca which have been published in the Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems (*AAMAS 2018*) [CNNS18b], in the *Bulletin of the EATCS 125* [CNNS18c], and in the Proceedings of the 19th Italian Conference on Theoretical Computer Science (*ICTCS 2018*) [CNNS18a].
- The content of Chapter 6 is based on works co-authored by Emanuele Natale and Giacomo Scornavacca which have been published in the Proceedings of the 33rd AAAI Conference on Artificial Intelligence (*AAAI 2019*) [CNS19], in the *Bulletin of the EATCS 125* [CNS18], and in the Proceedings of the 19th Italian Conference on Theoretical Computer Science (*ICTCS 2018*) [CNNS18a].
- The content of Chapter 7 is based on a work co-authored by Luca Becchetti, Francesco Pasquale, and Sara Rizzo which has been published in the Proceedings of the 30th International Symposium on Algorithms and Computation (*ISAAC 2019*) [BCPR19].

Date: 08/12/2019 Signature: 

A Emanuela

Acknowledgments

È attribuita a Galileo l'affermazione "le cose sono unite da legami invisibili, non puoi cogliere un fiore senza turbare una stella." Mi piace partire da questa riflessione per ringraziare tutte le persone che, in modo più o meno significativo e importante, mi hanno aiutato a compiere questo lavoro. Tutti sono stati essenziali e tutti ringrazio di cuore, uno ad uno.

Ringrazio i miei relatori Luca Becchetti ed Emanuele Natale per l'entusiasmo, la grinta e lo spirito audace e attento con cui mi hanno seguito in questo percorso.

Ringrazio Gianlorenzo D'Angelo, Luca Aceto, Antonia Bertolino e Francesco Pasquale per la disponibilità, la piena collaborazione e i preziosi consigli.

Ringrazio Andrea Clementi e Robert Elsässer per i commenti e gli stimoli costruttivi con cui hanno contribuito a perfezionare la mia tesi.

Ringrazio tutti i miei coautori: André Nusser, Antonia Bertolino, Breno Miranda, Emanuele Natale, Federico Corò, Francesco Pasquale, Giacomo Scornavacca, Gianlorenzo D'Angelo, Luca Becchetti, Mohammad Abouei Mehrizi, Roberto Verdecchia, Sara Rizzo, Stefano Ponziani. Senza uno solo di loro la mia avventura non sarebbe stata la stessa.

Ringrazio i compagni di dottorato all'Aquila per la loro presenza quotidiana e per i dubbi e i successi condivisi.

Ringrazio i miei amici vicini e lontani perché posso contare sempre su di loro e perché sanno quale birra preferisco in ogni circostanza.

Ringrazio la mia famiglia che mi ha sempre sostenuto e incoraggiato, senza mai smettere di credere in me.

Ringrazio Emanuela per l'amore paziente e coraggioso con cui rende più ricca e vera la mia vita e per il modo in cui mi regala ogni giorno la gioia immensa di poterla sentire accanto.

Abstract

Label Propagation Algorithms are a class of heuristics for the problem of *graph clustering*, i.e., the problem of detecting groups of nodes whose connections are dense within each group and sparse between the groups. At the onset, a label is assigned to each node of the graph; then, each node iteratively updates its label according to a function of the labels of its neighbors. Empirical studies show that, after only a few rounds, nodes in the same cluster share the same label while nodes in different clusters have different labels. Although they are widely used in practice given their simplicity, efficiency, and effectiveness, there is no theoretical foundation to explain why such simple algorithms are able to perform such a hard task. The absence of theoretical progress in the analysis of Label Propagation Algorithms is due to the lack of mathematical techniques for handling the interplay between the non-linearity of their update rule and the topology of the underlying graph.

In this thesis we contextualize Label Propagation Algorithms in the framework of *computational dynamics*, simple dynamical processes on networks whose behavior has been formally characterized on some classes of graphs. The analyses of computational dynamics were mainly focused on graphs with good connectivity properties, such as cliques or expanders, and on the problem of *consensus*, showing that they naturally converge to a configuration in which all the nodes agree on some value. We move a step forward in this direction by rigorously analyzing two simple dynamics, the *2-Choices dynamics* and the *Averaging dynamics*, reaching a more fine-grained comprehension of their consensus behavior in classes of graphs that exhibit a clustered structure. In particular we formally prove that, with non-negligible probability, the two dynamics quickly bring the graph in a configuration where each cluster reaches an *internal consensus* on a value that is different among the clusters, and then enters a long *metastable phase* in which the internal consensus are maintained.

We show how to exploit such metastable behavior to design *simple randomized distributed algorithms* for graph clustering.

I Label Propagation Algorithms *sono una classe di euristiche per il problema del* clustering di grafi*, cioè il problema di suddividere i nodi di un grafo in gruppi, o* cluster*, in modo tale che le connessioni tra nodi all'interno dello stesso gruppo sono dense mentre quelle tra nodi in gruppi diversi sono sparse. All'inizio viene assegnato un colore a ogni nodo del grafo; successivamente, in maniera iterativa, ogni nodo aggiorna il proprio colore in funzione dei colori dei propri vicini. Alcuni studi mostrano empiricamente che, dopo poche iterazioni, i nodi appartenenti allo stesso cluster condividono uno stesso colore, mentre nodi appartenenti a cluster diversi hanno un colore diverso. Sebbene siano largamente usati in pratica per la loro semplicità, efficienza ed efficacia, non ci sono fondamenta teoriche per spiegare perché questi semplici algoritmi sono in grado di risolvere un problema così difficile. L'assenza di progresso teorico nell'analisi dei Label Propagation Algorithms è dovuta alla mancanza di tecniche matematiche che trattano l'interazione tra la nonlinearità della funzione con cui i nodi aggiornano il proprio colore e le caratteristiche strutturali della rete di comunicazione che connette i nodi.*

In questa tesi i Label Propagation Algorithms vengono contestualizzati nel quadro teorico delle dinamiche computazionali*, cioè semplici processi dinamici su reti il cui comportamento è stato caratterizzato formalmente su alcune classi di grafi. Le analisi di queste dinamiche si focalizzano su grafi con ottime proprietà di connettività e sul problema del* consenso*, ovvero provando la loro naturale convergenza a configurazioni cromatiche in cui tutti i nodi concordano su uno stesso colore. In questa tesi viene fatto un passo in avanti in questa direzione: le dinamiche* 2-Choices *e* Averaging *sono analizzate rigorosamente, comprendendo meglio come il consenso viene raggiunto in classi di grafi con una struttura che presenta dei cluster. Più nel dettaglio viene dimostrato che, con una probabilità non trascurabile, le due dinamiche portano i nodi in una configurazione cromatica dove ogni cluster raggiunge velocemente un* consenso interno*, su un colore diverso per ogni cluster, prima di iniziare una lunga fase* metastabile *in cui i consensi interni sono mantenuti.*

Nella tesi viene mostrato come questo comportamento metastabile che caratterizza alcune dinamiche possa essere sfruttato per progettare semplici algoritmi probabilistici e distribuiti per il problema del clustering di grafi.

Contents

Preface

This thesis presents one of the three lines of researched I carried out, together with my collaborators, during my years as a Ph.D. student. In this Preface I would like to give a brief global overview of what I studied during the last three years and of the publications the research led to.

Simple Randomized Distributed Algorithms for Graph Clustering

This is the line of research investigated in the thesis. We analyze simple dynamical, possibly stochastic, processes on networks known in literature as computational dynamics [Nat17]. The theoretical computer science community, that recently got interested in them, proved that such dynamics can be used as efficient distributed algorithm for consensus. In this line of research we show that, when the underlying network has a community structure, some dynamics exhibit a metastable phase that highlights the community structure before converging to consensus and can thus be used, despite their simplicity, as efficient distributed algorithms for graph clustering.

This line of research resulted in the following papers:

- E. Cruciani, E. Natale, A. Nusser, G. Scornavacca. "Phase Transition of the 2-Choices Dynamics on Core-Periphery Networks." *Full paper* published in Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems (*AAMAS 2018*) [CNNS18b].

- E. Cruciani, E. Natale, A. Nusser, G. Scornavacca. "Phase Transition of the 2-Choices Dynamics on Core-Periphery Networks." *Extended abstract* published in *Bulletin of the EATCS 125* (2018) [CNNS18c].

- E. Cruciani, E. Natale, G. Scornavacca. "On the Metastability of Quadratic Majority Dynamics and its Biological Implications." *Extended abstract* presented at the 6th workshop on Biological Distributed Algorithms (*BDA 2018*) and published in *Bulletin of the EATCS 125* (2018) [CNS18].

- E. Cruciani, E. Natale, A. Nusser, G. Scornavacca. "On the Emergent Behavior of the 2-Choices Dynamics." *Short communication* published in Proceedings of the 19th Italian Conference on Theoretical Computer Science (*ICTCS 2018*) [CNNS18a].

- E. Cruciani, E. Natale, G. Scornavacca. "Distributed Community Detection via Metastability of the 2-Choices Dynamics." *Full paper* published in Proceedings of the 33rd AAAI Conference on Artificial Intelligence (*AAAI 2019*) [CNS19].

- L. Becchetti, E. Cruciani, F. Pasquale, S. Rizzo. "Step-by-Step Community Detection in Volume-Regular Graphs." *Full paper* published in Proceedings of the 30th International Symposium on Algorithms and Computation (*ISAAC 2019*) [BCPR19].

Models and Algorithms for Election Control through Influence Maximization

We investigate the problem of controlling elections through social influence [WV18], i.e., exploiting social influence in a network of voters in order to change their opinion about some target candidate to change the outcome of the election in her favor. We introduce two models based on the *Linear Threshold Model* [KKT15] and provide positive and negative approximation results to the problem of election control.

This line of research resulted in the following papers:

- F. Corò, E. Cruciani, G. D'Angelo, S. Ponziani. "Vote for Me! Election Control via Social Influence in Arbitrary Scoring Rule Voting Systems." *Extended abstract* published in Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems (*AAMAS 2019*) [CCDP19b].

- F. Corò, E. Cruciani, G. D'Angelo, S. Ponziani. "Exploiting Social Influence to Control Elections Based on Scoring Rules." *Full paper* published in Proceedings of the 28th International Joint Conference on Artificial Intelligence (*IJCAI 2019*) [CCDP19a].

- M. Abouei Mehrizi, F. Corò, E. Cruciani, G. D'Angelo, S. Ponziani. "Models and Algorithms for Election Control through Influence Maximization." *Short communication* published in Proceedings of the 20th Italian Conference on Theoretical Computer Science (*ICTCS 2019*) [MCC+19].

- M. Abouei Mehrizi, F. Corò, E. Cruciani, G. D'Angelo. "Dealing with Uncertainty in Election Control through Social Influence." *Full paper* currently under review.

FAST Approaches for Software Testing

We propose scalable solutions for some important issues in software regression testing, using tools from Information Retrieval and Data Mining. In particular we map these problems to ranking and clustering ones and we use efficient tools from the Big

Data domain. The approaches we developed are thought for large scale scenarios where state-of-the-art techniques cannot deal with the test suites due to their size.

This line of research resulted in the following papers:

- B. Miranda, E. Cruciani, R. Verdecchia, A. Bertolino. "FAST Approaches to Scalable Similarity-based Test Case Prioritization." *Full paper* published in Proceedings of the 40th International Conference on Software Engineering (*ICSE 2018*) [MCVB18].

- E. Cruciani, B. Miranda, R. Verdecchia, A. Bertolino. "Scalable Approaches for Test Suite Reduction." *Full paper* published in Proceedings of the 41st International Conference on Software Engineering (*ICSE 2019*) [CMVB19]. The paper won the *ACM SIGSOFT Distinguished Paper Award*.

- F. Corò, R. Verdecchia, E. Cruciani, B. Miranda, A. Bertolino. "JTeC: A Large Collection of Java Test Classes for Test Code Analysis and Processing." *Tool/Dataset paper* currently under review.

- A. Bertolino, E. Cruciani, B. Miranda, R. Verdecchia. "Know Your Neighbor: Fast Static Prediction of Test Flakiness." *Full paper* currently under review.

1 Introduction

Network science is the area that studies *complex networks*, systems whose behavior is hard to model due to the dependencies, relationships, and interactions between their parts and with the environment they are set in. They appear in a wide variety of fields such as biology, chemistry, physics, sociology, economy, and computer science. Examples are telecommunication networks, computer networks, biological networks, cognitive and semantic networks, social networks, and many others. A common way to represent complex systems is through the use of *networks*, i.e., graphs where nodes represent the components of the systems and links represent their interactions.

Complex networks have non-trivial topological features, with patterns that are neither regular nor purely random. Many real-world networks exhibit a *community structure*, i.e., several groups of nodes that are densely interconnected among them, and more weakly connected towards the rest of the system. Among others, some example could be: social networks, that form communities groups based on common interests or locations; citation networks, that form communities based on research topics; protein interaction networks, that form communities based on similar functionalities inside a biological cell [For10].

Detecting communities is fundamental for many applications, e.g., in order to be able to zoom-out and look the network at a different scale where each community is a meta-node that simplifies the network structure or to provide insights into how network topology and function affect each other [For10].

One of the most successful approaches is that of *Label Propagation Algorithms*, a widely used class of protocols inspired by epidemic processes on networks that is appealing for its simplicity, efficiency, and effectiveness [RAK07]. A label is initially assigned to each node of the network; then, the label of each node is updated to the most frequent label in its neighborhood; after few updates, nodes belonging to the same cluster end up having the same label, while nodes in different clusters have different labels. Although widely used in practice, there is no theoretical foundation to explain why such simple algorithms are able to perform graph clustering. In fact, most of the work in the area is empirical except very few exceptions ([TK08, KPS13, CDIG+15]).

The absence of substantial theoretical progress in the analysis of LPAs is largely due to the lack of techniques for handling the interplay between the non-linearity of the local update rules and the topology of the underlying graph [UB13]. In order to move a step forward in this direction there is a need to contextualize LPAs in a more rigorously defined mathematical framework such as that of *computational dynamics* [Nat17], where some *simple dynamics* on networks are analyzed with the goal of exploiting their behavior to design lightweight distributed algorithms.

Computational dynamics are simple synchronous dynamical processes on networks, possibly of stochastic nature, where each node has an initial state and updates it over time according to a symmetric function of its state and of the states of its neighbors [Nat17]. This kind of processes has been classically studied in theoretical physics and statistical mechanics [Lig06, Lig09]; a recent renewed interest in the analysis of dynamics from theoretical computer scientists proved that these processes can be utilized as simple and lightweight distributed algorithms. Hassin and Peleg [HP01] proved how the arguably simplest dynamics, i.e., the *Voter dynamics* in which each node updates its state by coping that of a random neighbor, can be exploited as an extremely simple distributed algorithm to solve the *consensus* problem, which can be informally defined as follows: Given a network of agents each possessing some value, how can the agents agree on a single value among those initially possessed? Consensus is one of the most fundamentals problem in distributed computing with possible applications including the agreement on the identity of a leader, clock synchronization, usage in blockchains, and many others.

Other dynamics, besides the Voter, have also been analyzed in the framework of computational dynamics and thus as simple and lightweight distributed algorithms. Examples are: the 2-Choices [CER14, CER+15, CRRS17], where each node samples two neighbors and, if they have the same state, copies it; the Undecided-State [BCN+15, CGG+18], where there is an extra "undecided" state and each node becomes undecided if samples a neighbor with a state different from its own or copies the state of the sampled neighbor if it is undecided; the 2-Median [DGM+11], where each node updates its state to the median color among its color and that of two random neighbors (assuming an ordering of the colors); with all these dynamics the focus has been again on the consensus problem. Dynamics have also been successfully employed in order to design efficient solutions for other problems such as *noisy rumor spreading* [FN16], *exact majority* [MNRS17], and *clock synchronization* [BKN17].

The analyses of such dynamics were mainly focused on graphs which exhibit good connectivity properties, e.g., complete graphs or expanders, since these topologies facilitate the convergence of the dynamics to consensus. However, as previously introduced, many real-world networks do not have such desired properties and, instead, exhibit a community structure. The behavior of dynamics that naturally reach consensus could in fact change when the topology of the underlying network

changes: While the long term behavior of the dynamics remains the same, the time needed to reach it could drastically increase due to the effect of the topology of the communication network in the information spreading.

A first step in this direction was done by Becchetti et al. [BCN+17b, BCM+18], that analyzed the *Averaging dynamics* on graphs sampled from the regular stochastic block model, i.e., graphs exhibiting a clustered structure with two communities connected by sparse cuts, proving that a simple dynamics can be used to perform community detection. The averaging dynamics is a deterministic protocol where each node updates its state by averaging the states of its neighbors. Becchetti et al. show that the dynamics does converge to the global average (as originally proved in [BGPS06, Sha09]), i.e., to a consensus, but that, with some randomness in the initialization, the convergence of a node to the global average depends on the community the node belongs to: After a first short phase of settling, the state of a node slowly and monotonically increase or decrease towards the global average according to the community of the node.

In this thesis, we follow this line of research by deepening the theoretical understanding of computational dynamics and reaching a more fine-grained comprehension of the consensus behavior of the analyzed dynamics when the underlying network topology does not have good connectivity properties, but instead exhibits a community structure. In particular, we focus on two different dynamics, the *2-Choices dynamics* and the *Averaging dynamics*, rigorously analyzing their evolution and showing that their behavior can be exploited to design simple randomized distributed algorithms for graph clustering.

Recent developments in spectral graph theory (e.g., generalized Cheeger's inequality and related tools [LGT14, PSZ17]) and new techniques for bounding the convergence time of simple randomized protocols [BCN+15, BCN+17a, BCN+17b] are combined to rigorously analyze the behavior of these dynamics. This combination requires, in turn, to revisit known techniques which take into account the topology of the network to a greater extent than what has been done so far, and to deepen the understanding of the relation between the eigenspaces associated to the adjacency matrix of the network and its cluster structure.

We analyze the behavior of such dynamics on different classes of graphs having an evident community structure; in this setting we show that the processes quickly (e.g., in $\mathcal{O}(\log n)$ rounds, where n is the number of nodes in the networks) bring the system to a configuration where the nodes of each community agree on the same state, i.e., an *internal consensus*, while the consensus of different clusters are on different states; moreover, once in this configuration that highlights the community structure of the network, we show that the process enters a long *metastable phase* where most of the nodes retain their state before reaching (in the very long run, e.g., in $n^{\omega(1)}$ number of rounds) a stable consensus configuration.

1.1 Research contributions

Herein we show how simple randomized distributed algorithms, known in literature as *dynamics* and mainly studied to solve the consensus problem, can be also efficiently used to *detect communities* in networks, deepening the theoretical understanding on the class of graph clustering algorithms known as *label propagation algorithms*.

The central idea and recurrent theme of the thesis is that of considering dynamics that naturally bring the network to a consensus and to formally prove that, when the network exhibits a community structure, each community rapidly reaches an internal consensus (possibly different between communities) and maintains it for a very long time interval. We call this behavior of dynamics *metastable*, since in the long run the dynamics will still converge to a consensus configuration. Such metastable behavior is of extreme interest for computational purposes since if the time interval is long enough the dynamics naturally highlights the community structure of the network that can be thus identified by the agents using only local information and with extremely low individual computational resources.

2-Choices dynamics

In Part II we analyze the *2-Choices dynamics* on two different classes of networks: Core-Periphery networks [ALP+14] and clustered graphs sampled from the regular Stochastic Block Model [HLL83].

With the analysis on Core-Periphery networks [CNNS18b, CNNS18c, CNNS18a], i.e., networks that exhibit a bipartition of the nodes into a *core* (small, i.e., $\mathcal{O}(\sqrt{n})$ nodes, and densely connected) and a *periphery* (large, i.e., $\mathcal{O}(n)$ nodes, and sparsely connected), we move a step forward considering a class of networks different than those with strong expansion properties; moreover we investigate the role that the initial configuration of the states of the nodes plays in the dynamics' behavior. Starting from a natural configuration where nodes in the core and in the periphery initially have different states, we prove that a phase transition phenomenon occurs depending on the strength of the cut that separates the two partitions: Either the core's initial state is rapidly spread among the entire network, or the periphery resists retaining its initial state and starting a long metastable phase in which both states coexist in the network, *with high probability* (Definition E.1.5).

The theoretical analysis initially considers the settings in which the nodes in the core are *stubborn*, i.e., they do not change state, later substituting this assumption with milder hypothesis on the core's structure. We give bounds on the expected change of the number of agents supporting a given state, obtaining a concentration of probability around the expected evolution with the use of Chernoff bounds. The evolution of the dynamics when the core is stubborn can be regarded as a SIS-like

epidemic model [Het00, EJN02], where the infection probability together with a certain probability of spontaneous infection are given by the 2-Choices dynamics.

This first result of the thesis shows that, when the graph is partitioned into two groups of nodes, the 2-Choices dynamics could have a metastable behavior that highlights the partition; moreover the results suggests that the 2-Choices dynamics could be used to design a distributed label propagation algorithm.

We also validate our theoretical predictions by performing extensive experiments on real-world networks (Section 5.3). In order to do that, we design an algorithm to extract the core of a network by iteratively computing densest-subgraph approximation and run simulations of our dynamics. The results of the simulations show a strong correlation with the theoretical predictions made by our model.

With the analysis of the 2-Choices dynamics on clustered regular graphs [CNS19, CNS18, CNNS18a] we show a more fine-grained understanding of the consensus behavior of the dynamics. Our new analysis combines symmetry-breaking techniques [BCN+16, CGG+18] and concentration of probability arguments with a linear algebraic approach [CER+15, CRRS17] in order to obtain the *first symmetry-breaking analysis of dynamics on non-complete topologies*. In particular, we show that if the nodes of the network initially choose a random binary state then there is a constant, non-negligible probability that the process converges to an almost-clustered configuration, i.e., a configuration where almost all nodes within each community share the same state but the predominant states in the communities are different. The process rapidly reaches such configuration and maintains it for an extremely long time and thus the states of the nodes constitute a labeling which reveals the clustered structure of the network.

The aforementioned constant probability for the labeling to reveal the community structure of the network can be amplified using a *Community-Sensitive Labeling* [BCM+18], hence transforming the 2-Choices dynamics into a *distributed label propagation algorithm* with quasi-linear message complexity.

Our result is the *first analytical result* on the behavior of a *sparsified* label propagation algorithm. The 2-Choices dynamics performs an implicit sparsification of the graph, an interesting property for the design of sparse clustering algorithms [SZ17], in particular for opportunistic network settings [BCM+18], since the nodes sample in each round the neighbors instead of looking at all of them. The only other analysis in literature is that of Max-LPA [KPS13], a deterministic majority dynamics. Compared to the analysis of Max-LPA, our result holds for much sparser communities at the price of a stricter condition on the cut. Moreover, given the distributed nature of the two algorithms, the message complexity of the 2-Choices dynamics is lower and does not depend on the density of the graph.

Our results on the metastability of the 2-Choices dynamics on clustered graphs has some *biological implications* (Section 6.3).

In the context of *evolutionary biology*, dynamics such as the Moran process, which has been proved to be equivalent to the Voter dynamics when the underlying graph is regular [Gia16], have been used to model the spread of mutations in genetic populations [LHN05]. However, no simple dynamics has been proposed so far in the context of evolutionary graph theory for explaining one of evolution's fundamental phenomena, namely *speciation* [CO04]. We propose the 2-Choices dynamics as an evolutionary dynamics for *sympatric/parapatric speciation*, i.e., divergence of two species in the scenario in which there is no complete geographical isolation, and provide an analytical evolutionary graph-theoretic proof of concept of how speciation can emerge from the simple nonlinear underlying dynamics of the evolutionary process at the population level.

In the context of *neuroscience*, evolutionary graph dynamics have been used to model the innervation of synapses on muscular junctions during development. Our result, corroborated by numerical simulations on a wider class of dynamics, provide evidence that, in order for the model to comply with experimental evidence on the outcome of the innervation process, either the innervation sites do not exhibit spatial bottlenecks or the dynamics cannot be based on majority-like mechanism.

Averaging dynamics

In Part III we analyze the *Averaging dynamics* on a class of graphs that we call *k-volume-regular*, extending the result of Becchetti et al. [BCN+17b] to the case of k unbalanced communities, also relaxing the regularity constraint with the use of the weaker notion of k-volume-regularity that we introduce. The concept of volume-regularity is deeply related with that of *ordinary lumpability* of Markov chains [KS60]. Moreover, volume-regular graphs are the largest class of undirected, possibly weighted graphs that admit k "stepwise" eigenvectors (i.e., with constant values over the k "steps" that identify the k partitions of the graph), necessary to follow the arguments used by Becchetti et al. to prove the behavior of the dynamics.

We show that, if the k stepwise vectors of a volume-regular graph are those associated with the k largest eigenvalues and the gap between the k-th and the $(k+1)$-th eigenvalues is sufficiently large, the averaging dynamics allows to *reconstruct the original partition* of the graph after a short time and for a long time interval, with high probability. In order to prove this, we provide a *family of orthogonal vectors* which have the same span of the (unknown) stepwise vectors of any volume-regular graph and that generalize the Fiedler vector [Fie89], used in spectral partitioning algorithms.

We also introduce the concept of *community sensitive algorithm*, an algorithm that if executed in parallel for a suitable number of times produces a *community-sensitive labeling* [BCM+18], i.e., assigns a binary signature to each node of the graph with the property that the Hamming distance between signatures of nodes in

the same community is small, while that of nodes belonging to different communities is large.

Moreover, we show that variants of the averaging dynamics can address different problems such as detecting if a graph is *bipartite*.

1.2 Organization of the thesis

After the Introduction and an overview of the results (Chapter 1), we divide the thesis into three parts.

In Part I we start by presenting some essential background material. We first introduce the problem of *graph clustering*, also known as community detection, giving an overview of the most popular methods used to cope with it (Chapter 2) and with a particular focus on *Label Propagation Algorithms* (Chapter 3). Then we introduce the framework of *computational dynamics* (Chapter 4), in which we contextualize our analytical results, presenting previous works that are fundamental for this thesis.

In Part II we present the results obtained by analyzing the *2-Choices dynamics* on two different classes of graphs: *Core-Periphery networks* (Chapter 5), in which we show a phase-transition on the behavior of the dynamics, and *Clustered networks* (Chapter 6), in which we show that the dynamics reaches and maintains a metastable phase that highlights the clustered structure of the network.

In Part III we present the results relative to the analysis of the *Averaging dynamics* on a class of graphs that we call volume-regular, showing that the dynamics can be used to reconstruct the hidden k-partition of the underlying graph during a long time interval (Chapter 7).

In Chapter 8 we conclude the thesis and present open problems and possible future research directions.

In the Appendix we start by introducing the asymptotic notation used for the analysis of algorithms (Appendix A); then we present definitions, basic results, and more advanced tools in the areas of Graph Theory (Appendix B), Linear Algebra (Appendix C), Spectral Graph Theory (Appendix D), Probability Theory and Stochastic Processes (Appendix E).

I

Background

2
Graph Clustering

2.1 Networks and communities

Network theory is a relatively novel discipline that spans the natural, social, and computer sciences as well as engineering [FH16]. In fact several complex systems of different nature, e.g. biological or cyber-physical ones, can be represented as networks, which describe the interaction among the agents populating the systems, and many research areas study and analyze phenomena happening on them. Many complex systems usually seen as networks are part of our everyday lives: the Internet, where web pages are interconnected by hyperlinks; social networks, such as Facebook or Twitter, where people are connected by relationships; chemical reaction networks, where bio-chemical molecules are connected if they are part of a reaction; collaboration, communication, and transport networks are just some other examples.

Nowadays, in the era of Big Data, the amount of network data and the number and size of network datasets increases at an incredibly fast pace, as well as the interest in more efficient and suitable computational techniques that can help to understand the properties of networks.

Networks, given their essence, are well suited to be represented as graphs: The agents in the systems form the set of nodes of the graphs, while the interaction among them the set of edges connecting the nodes. In this thesis we consider undirected (possibly weighted) graphs, since they are simpler to reason about and still powerful enough to model many real-world scenarios.

Obviously, the structural properties of networks as well as the dynamics of the interactions among the agents, can be studied under a graph-theoretical point of view, exploiting analytical frameworks and results already known in computer science and other areas.

Most networks of interest exhibit a *community structure*, i.e., the nodes in the graph can be grouped together according to some criteria, as can be seen in Figure 2.1.[1] In general, communities could represent groups of people sharing some interests or involved in similar activities in social networks, such as members of the

[1] Source: https://griffsgraphs.wordpress.com/2012/07/02/a-facebook-network/

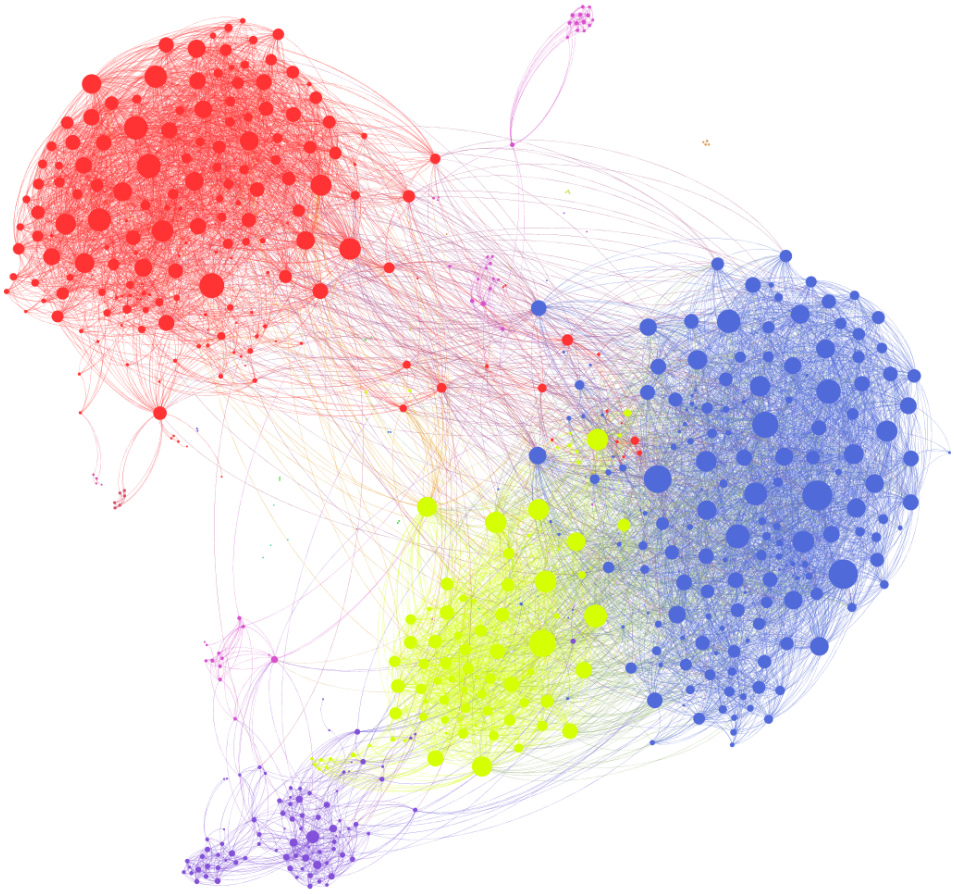


Figure 2.1: The network of friendships of a single Facebook user. Each color represent a community: high-school friends (red), girlfriend's friends (yellow), university colleagues (blue), other smaller communities (remaining colors).

same fan club or classmates in high-school, or web-pages coming from the same domain in the Internet. The task of identifying such communities in networks is referred to as *community detection*, or equivalently *network* or *graph clustering*. Grouping nodes into communities may offer insights on the network organization, allowing us to focus on specific areas of the network. Moreover it helps to classify the nodes based on their role with respect to the communities they belong to, e.g. nodes at the boundary of the cluster could play an important role in processes such as information spreading dynamics and it is of interest to be able to identify them.

In many real-world scenarios communities can overlap among them, sharing some nodes. For instance, in social networks a user can be part of different groups at the same time (family, friends, work, etc.), while in bio-informatics genes can be co-regulated or co-expressed and thus part of different clusters at the same time. In this case we talk about *soft clustering*, opposed to the *hard clustering* (or just clustering), in which we look for non overlapping communities, and which is applied in many disciplines such as medicine, biology, and economics [BDSY99].

2.2 Community quality metrics

Although the problem of graph clustering is fundamental in network analysis and it has been thoroughly studied for several years from scientists with different backgrounds, the problem itself is not well defined since communities are just informally defined as "groups of nodes highly interconnected within each group and loosely connected between them." In fact, there exist no formal definition of community and no unique metric to measure their quality. On the one hand, it is not clear how to assess the performance of graph clustering algorithms and how to compare them in terms of effectiveness; on the other hand, different metrics are able to approach the problem from different perspectives that can vary depending on the specific research question.

2.2.1 Internal criteria

Modularity

An important metric that captures the community structure of a graph is the *modularity*, which is defined as follows. Consider an Erdös-Rényi random graph [ER60]; the probability that two nodes are connected is the same for every pair of nodes and consequently the graph has no well-defined community structure. This concept is known as the *null model* in the community detection area, and has been introduced to define the modularity [New04]. The idea is to compare the partitions of a graph with the null model to assess the degree of its community structure. In particular, the modularity measure is computed as the total fraction of edges within the communities minus the expected fraction if they were distributed at random, e.g., as in an Erdös-Rényi graph. Formally, given an undirected graph $G = (V, E)$, with $|V| = n$ and $|E| = m$, and a partition of the nodes $C = \{C_1, \ldots, C_k\}$, such that $\bigcup_{i=1}^{k} C_i = V$ and $C_i \cap C_j = \emptyset$ for every pair C_i, C_j, the modularity of the partition C is defined as

$$Q = \sum_{C_i \in C} \left[\frac{|E(C_i)|}{m} - \left(\frac{\sum_{v \in C_i} deg(v)}{2m} \right)^2 \right], \tag{2.1}$$

where $E(C_i)$ is the set of edges having both endpoints in C_i, and $deg(v)$ is the degree of a node v, i.e., $deg(v) = |\{u : (u, v) \in E\}|$. A more general and more compact matrix definition, where the null model is not specified, is the following

$$Q = \frac{1}{2m} \sum_{(i,j) \in V \times V} (A_{ij} - P_{ij})\ \delta(C_i, C_j), \tag{2.2}$$

where A is the adjacency matrix of G, P is the expected adjacency matrix of the null model, and $\delta(C_i,\ C_j)$ is the Kronecker delta[2] of the communities to which nodes i and j belong to.

[2]The Kronecker delta function $\delta(x, y)$ is defined as $\delta(x, y) = 1$ if $x = y$ and $\delta(x, y) = 0$ otherwise.

The above definition of modularity (Eq. (2.1)), even if widely used as objective function by many graph clustering algorithms, has some limitations. In particular, modularity optimization may fail to identify small clusters whenever the number of edges of the network is big enough with respect to the number of edges of the interconnected communities, even in cases where the clusters are extremely well defined [FB07]. This phenomenon is called *resolution limit* of the modularity and it is a consequence of the modularity definition, which uses a global null model, i.e., which takes into account the structure of the whole graph. Some variants of the modularity consider different null models [BBV08], i.e., different random graph models other than the Erdös-Rényi one, or a local null model that consider the structure of each single cluster.

Conductance and k-way expansion

Another metric which is widely used for community detection and is worth mentioning is the *edge expansion* or *conductance* [Bol98]. Given a graph $G = (V, E)$ and a set of its vertices $S \subseteq V$, the conductance of the set S is defined as

$$\phi(S) = \frac{|E(S,\ V \setminus S)|}{vol(S)}, \tag{2.3}$$

where $vol(S) = \sum_{v \in S} deg(v)$. It measures the average fraction of neighbors going outside of S for a random node $v \in S$, and it compares the actual number of edges crossing the cut $(S, V \setminus S)$ with the trivial upper bound $vol(S)$. If the conductance of a graph is low then there exist a sparse cut and the two subsets of nodes divided by such cut can be considered as communities. The conductance of the whole graph, instead, is defined as

$$\phi(G) = \min_{S} \phi(S). \tag{2.4}$$

The notion of edge expansion can be generalized to the case of multiple partitions through the notion of *k-way expansion* [LGT14], which is defined as

$$\rho(G) = \min_{S_1,\ldots,S_k} \max\{\phi(S_i) : i = 1, \ldots, k\}. \tag{2.5}$$

2.2.2 External criteria

Internal criteria are used to assess graph clustering quality as discussed before, but are not the only existing metrics. More specifically, there exist metrics used to compare results obtained by different graph clustering algorithms or with a ground truth, i.e., a correct classification of the nodes into clusters.

Normalized mutual information

One of the most used external criteria is the *Normalized Mutual Information* (NMI). Given two partitions $\mathcal{C}_X = \{C_1, \ldots, C_m\}$ and $\mathcal{C}_Y = \{C_1, \ldots, C_n\}$, e.g., the ground

truth and the partition found by the algorithm, we define two random variables X, Y that represent the random assignment of a node to a community, respectively in $\mathcal{C}_X$ and $\mathcal{C}_Y$. Formally, $X \in \{1, \dots, m\}$ and $\Pr(X = x) = \frac{|C_x|}{\sum_{i=1}^{m} |C_i|}$ and similarly for Y. Hence, the metric is defined as

$$\mathrm{NMI}(X, Y) = 2 \left[1 - \frac{H(X, Y)}{H(X) + H(Y)} \right], \tag{2.6}$$

where $H(X)$ is the *entropy* of X and $H(X, Y)$ the *joint entropy* of X and Y, namely

$$H(X) = -\sum_{x} \Pr(X = x) \log \Pr(X = x), \tag{2.7}$$

$$H(X, Y) = -\sum_{x} \sum_{y} \Pr(X = x, Y = y) \log \Pr(X = x, Y = y). \tag{2.8}$$

Rand index

The *Rand Index* (RI) is another metric, for two communities but generalizable to many, which is equal to the fraction of node pairs correctly classified in both partitions over the total number of pairs. Given a graph $G = (V, E)$ and two partitions of the vertex set $X = \{X_1, \dots, X_r\}$, $Y = \{Y_1, \dots, Y_s\}$, let us define:

- a as the number of pairs of nodes that are in the same subset in X and in the same subset in Y;
- b as the number of pairs of nodes that are in the different subset in X and in the different subset in Y;
- c as the number of pairs of nodes that are in the same subset in X and in the different subset in Y;
- d as the number of pairs of nodes that are in the different subset in X and in the same subset in Y.

Hence, the metric is defined as

$$\mathrm{RI}(X, Y) = \frac{a + b}{a + b + c + d}, \tag{2.9}$$

i.e., the number of agreements in the assignments of the nodes in the two partitions over the total number of pairs.

2.3 Networks with community structure

2.3.1 Synthetic network models

The *Stochastic Block Model* (SBM) [HLL83] is a generative model for random graphs which tends to produce graphs presenting community structure. In particular, SBM takes as parameters the number of nodes n of the graph to generate, a partition of

the nodes into k communities $C_1, \ldots, C_k$, and a symmetric $k \times k$ matrix P; the edges are samples according to P, i.e. the nodes $u \in C_i$ and $v \in C_j$ are connected with probability P_{ij}. Some special cases of SBM are the Erdös-Rényi model, in which $P_{ij} = p$ for each i, j, or the *planted partition model*, in which $P_{ii} = p$ for each i and $P_{ij} = q$ for each pair i, j such that $i \neq j$. The planted partition model is important in the area of community detection since the graph sampled from it present a strong community structure whenever $p \gg q$. Given a graph sampled from the planted partition model, a common task for community detection algorithms is to exactly or partially reconstruct the latent partition without knowing the model parameters the input graph comes from.

There exist other models for the creation of graph with community structure. For example, the *caveman model* starts with k cliques of size l and rewires one single edge per clique to another node in an adjacent clique, or its relaxed version, the *relaxed caveman model*, in which each edge is rewired with probability p toward another clique.

A modified version of the planted partition model that keeps into account for the heterogeneity of degrees and community sizes is called *Gaussian random partition* [BGW03]. The cluster sizes have a Gaussian distribution, so they do not have the same, although they do not differ much from each other. Such heterogeneity in the cluster sizes also introduces a diversity in the degree distribution since the expected degree of a node depends on the size of the cluster it belongs to. The variability of degree and cluster size, though, is not substantial.

Lancichinetti et al. tried to improve on this direction by introducing a class of benchmark graphs known as LFR (from the names of the authors), where the distributions of degree and community size are power laws, thus similarly to many real-world networks [LFR08].

2.3.2 Real-world networks

Synthetic networks are good as benchmarks for community detection algorithms, but do not always have the same properties and cluster structure of real-world networks. In fact, many real world networks are used to test graph clustering algorithms and compare their effectiveness in identifying the communities. The most used networks are usually taken from existing social networks, such as Facebook or Twitter; collaboration network, using author information of scientific articles; biological networks, where gene co-occurrences in DNA sequences are considered; and many others.

There exist many repository of network datasets used in scientific research, presenting real-world networks with different properties. Some of the most popular datasets used as benchmarks can be found on the following repositories:

- `http://networkrepository.com`

- https://snap.stanford.edu/data
- http://konect.uni-koblenz.de
- http://www-personal.umich.edu/~mejn/netdata
- http://vlado.fmf.uni-lj.si/pub/networks/data
- http://socialcomputing.asu.edu/pages/datasets
- https://sites.google.com/site/ucinetsoftware/datasets

2.4 Algorithms for graph clustering

The problem of graph clustering is approached in many different ways. Some algorithms require some knowledge about the number of clusters to identify, while others do not. This is because graph clustering is similar to the *graph partitioning* problem, a classical problem in computer science known to be in *NP*-hard. They both have the same goal, i.e., partition the graph into clusters satisfying specific properties, but graph partitioning knows the number k of partitions beforehand. Algorithms for graph partitioning, though, are not good for community detection, since require information that is, in principle, unknown and that is expected as output of the algorithm itself.

Many approaches partition the graph into two clusters, and then repeat recursively the procedure into each cluster. From a methodological perspective this is not reliable, since it could be imprecise in case the number of clusters is odd.

2.4.1 Traditional methods

An intuitive way to partition a graph into two community is to find the minimum cut of the graph, i.e., the partition that minimizes the number of edges across the cut. The minimum cut can be found in polynomial time with a maximum flow algorithm, such as *Ford-Fulkerson* or *Edmonds-Karp*, based on the following theorem.

Theorem 2.4.1 (Max-flow min-cut theorem)**.** *Given a graph $G = (V, E)$ and such that $s, t \in V$ are the source and target nodes, the maximum value of an $s - t$ flow is equal to the minimum capacity over all $s - t$ cuts.*

This is not enough to solve graph partitioning though, since cutting out single nodes does not help much in identifying clusters and does not reveal any higher-level structural properties [Sch07]. An example of graph clustering algorithm using minimum cuts is the one proposed by Flake et al. in [FTT04]. The algorithm inserts an artificial sink node in the graph and it computes the maximum flow towards the artificial sink node; a *Gomory-Hu minimum-cut tree* [GH61] is then used to assign each vertex to its cluster. A *Gomory-Hu tree* of a weighted graph, of which we can

see an example in Figure 2.2,[3] is a weighted tree that represents the minimum $s-t$ cuts for all the $s-t$ pairs in the graph and can be constructed in $|V|-1$ maximum flow computations.

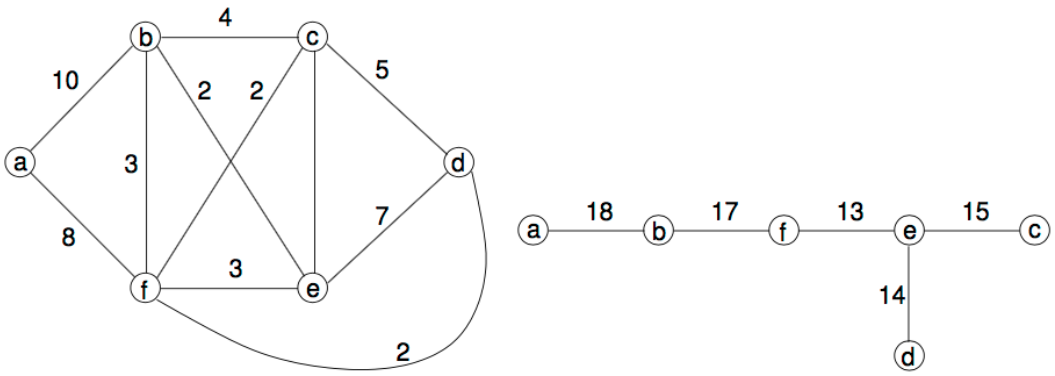


Figure 2.2: A graph G with its corresponding Gomory-Hu tree.

The Kernighan-Lin heuristic [KL70] is one of the earliest methods proposed to solve the problem and is still frequently used, especially combined with other techniques [For10]. It is a local-search algorithm, i.e., iteratively apply only a small change to the current candidate solution, moving towards a better one. Starting from a random bisection of the graph, the algorithm iteratively improve the quality of the partition by swapping every pair of nodes that decreases the size of the cut. The original algorithm separates the graph into two clusters of predefined size, but it has been extended to divide it into an arbitrarily number of clusters with an increase in the run-time and storage costs that grows with the number of clusters [SK88].

The basic version works as follows: let $G = (V, E)$ a graph and (S, T), with $T = (V \setminus S)$, a partition of G such that $|S| = n_1$ and $|T| = n_2$; the algorithm tries to minimize $|(S, T)|$ and has three phases which are repeated for r rounds.

Each round of the algorithm goes as follows:

1. Pick a random partition (S, T) of G, such that $|S| = n_1$ and $|T| = n_2$; all the pairs of nodes are not marked for now.

2. For every unmarked pair $(u, v) \in S \times T$ such that swapping them decreases the size of the cut (S, T), swap them and then mark the pair, namely

$$S := S \cup \{u\} \setminus \{v\}, \tag{2.10}$$

$$T := T \cup \{v\} \setminus \{u\}. \tag{2.11}$$

3. Return (S, T) as solution.

Note that at the end no pair (u, v) improves the solution swapping the nodes. The algorithm then returns r candidate solutions, among which the best one, i.e.

[3] Source: https://courses.engr.illinois.edu/cs598csc/sp2010/Lectures/Lecture6.pdf

the one with the minimum size partition, is chosen. It runs in $\mathcal{O}(rmn^2)$, but has been proved to scale to $\mathcal{O}(n^2 \log n)$ performing only a constant number of swaps at each iteration. Though, increasing the number of clusters found by the solution, space and time costs rapidly increase too [For10].

2.4.2 Partitional methods

Partitional clustering is a collection of methods that separate a set of points belonging to a metric space into an a priori fixed number of partitions k. Such techniques are best known as a solution to the data clustering problem, but can be adapted to work on graphs under certain constraints [SLBK03, HW04, RMJ07].

The most famous technique is *k-means clustering*, which objective is to subdivide the point into k clusters minimizing the *sum of squared errors* (SSE), i.e., the sum of the distances of each point to its closest centroid, defined as

$$\text{SSE} = \sum_{j=1}^{k} \sum_{x \in C_j} \|x - \mu_j\|_2^2, \tag{2.12}$$

where x is a data point belonging to cluster C_j and μ_j is the *centroid* of the cluster, i.e., the center of mass of the points in C_j. The problem has several variants, such as *k-medoids* or *k-medians* which are slightly different. In the former the centroids are datapoints, while in the latter the cost function is the ℓ_1 norm, i.e., the centroid becomes the median point.

The most used technique to solve the k-means problem is Lloyd's algorithm [Llo82]. The algorithm starts by choosing k random centroids among the data points. In the Expectation phase it assign each data point to its closest centroid. In the Maximization phase it recompute centroids. The algorithm alternates expectation-maximization phases until the centroids remain the same for two consecutive rounds. Lloyd's algorithm is just an heuristic, i.e., even if it is proved to converge it only guarantees a local optimum.

It works as follows:

1. Initialization: choose k random centroids among the data points.
2. Centroid assignment: assign each data point to the closest centroid.[4]
3. Centroid recomputation: recompute each centroid as a function of the data points assigned to it, e.g., the center of mass or median point.
4. Convergence check: if the centroids remains the same for two consecutive rounds then the algorithm converged; if the algorithm has not converged, go back to Step 2.

[4]The definition of closest depends on the distance function we are using in the metric space. A distance function in $\mathbb{R}^n$ could be, for example, the Euclidean distance $d(x, y) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$.

The algorithms is proved to converge, although it does not always bring to the global optimal solution but only to a local one. What is usually done, in fact, is to run the algorithm multiple times with different initializations and pick the best solution, i.e. the ones with minimum cost function.

As in graph partitioning the number of clusters k that the algorithms has to produce has to be declared before running the algorithm. To find the "right" k there are several approaches, but the most used are the *rule of thumb*, setting $k = \sqrt{\frac{n}{2}}$, or the *elbow method* (Figure 2.4), which computes the Sum of Squared Errors (SSE) for various values of k and picks the value for which the SSE starts to decrease more smoothly; the SSE is defined as

$$\text{SSE} := \sum_{j=1}^{k} \sum_{x \in C_j} d(x, \mu_j)^2,$$

with $d(x, y)$ being the Euclidean distance between x and y. Both method are heuristics and do not guarantee optimality.

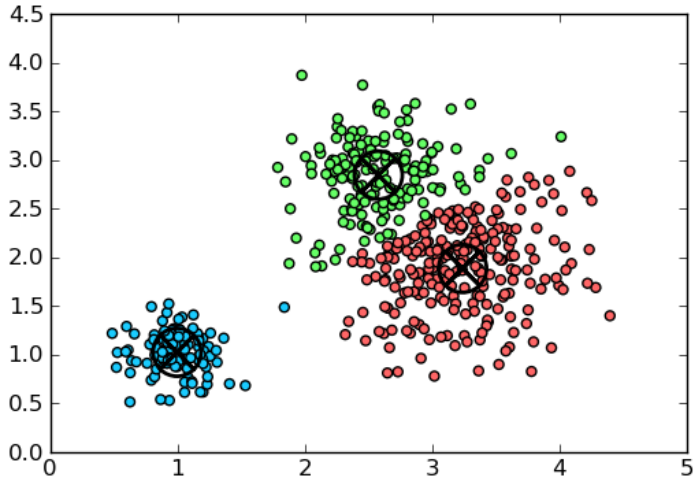


Figure 2.3: Example of *k-means* algorithm clustering 2-dimensional data points.

2.4.3 Hierarchical methods

Hierarchical clustering is a method used for clustering both data points and networks. The main advantage with respect to traditional methods is that the number of clusters it is not needed a priori. It produces a *dendrogram*, as that of Figure 2.5, which is a tree diagram that shows the multilevel clustered structure of the graph. There are two approaches: divisive and agglomerative algorithms.

Divisive algorithms. Divisive algorithms are top-down: They start considering the whole graph as a single cluster and iteratively split it into two or more partitions

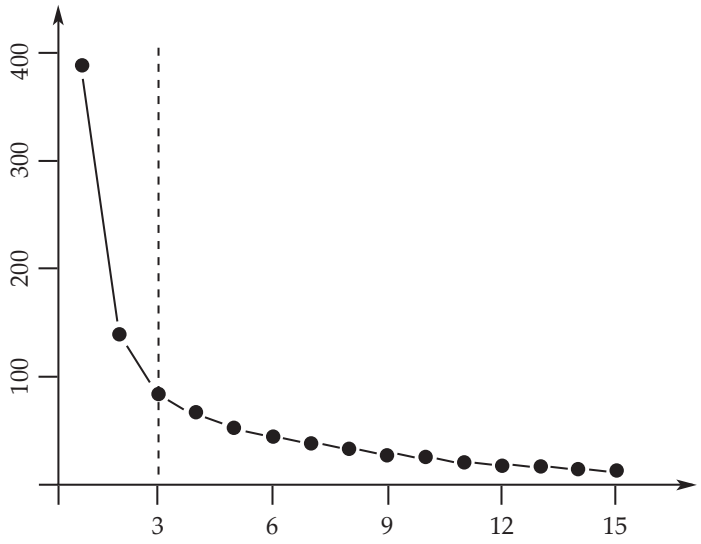


Figure 2.4: An example plot of the *elbow method* for choosing the right value of k in clustering applications. On the x axis the number k of cluster; on the y axis the SSE. The name comes from the usual shape of the plot, as we can see, that is similar to an elbow.

until each node is isolated at the end of the process. Girvan and Newman is the most important example of divisive algorithm for graph clustering [NG04]. It works by iteratively removing the most central edge from the graph, according to its *betweenness centrality* (see Appendix B.4), and update the dendrogram if the graph becomes disconnected after the removal. The algorithm finally selects from the dendrogram the partition with highest modularity.

In detail:

1. Initialize all nodes to the same color, meaning that initially they are all in the same cluster.

2. Compute the modularity of the network. If the value is higher than the best one computed so far, save the value and the partition.

3. Find the edge or, if more than one, the edges with highest betweenness centrality and remove it/them from the graph.

4. If the graph splits, resulting in more than one connected components, then update the partitions labeling the nodes of each component with a different color.

5. If there are still edges in the graph go back to Step 2.

6. Return the partition with highest modularity, saved after Step 2.

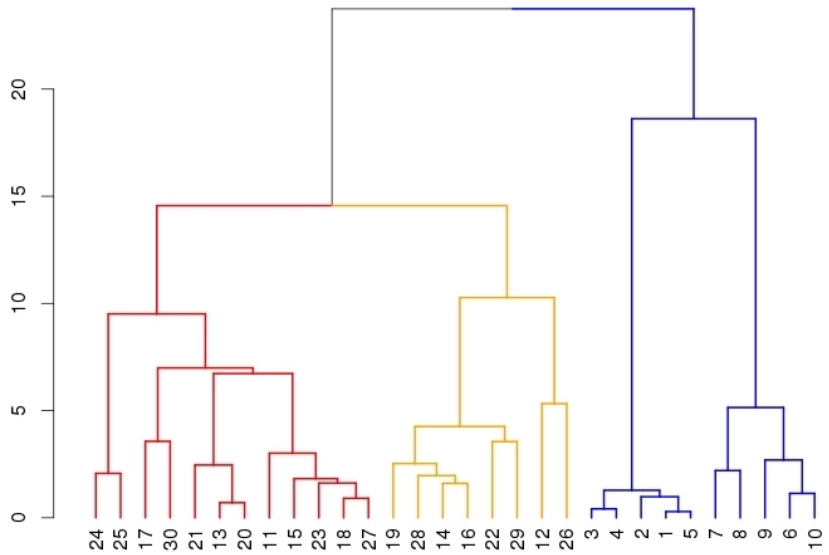


Figure 2.5: A dendrogram showing hierarchical clusters. The three colors indicate a possible partition into three clusters. The height of the dendrogram corresponds to the clusters obtained in different iterations of the algorithm.

The algorithm turns out to be slow, since at each iteration it has to recompute the edge betweenness, which cost $\mathcal{O}(n(n+m))$.

Agglomerative algorithms. Agglomerative algorithms, contrarily, are bottom-up: Each node initially forms its own cluster and then the clusters are iteratively merged together until all the nodes are in the same cluster. Newman also proposed an agglomerative algorithm [New04], which has a better time complexity, since it does not compute the edge betweenness at each iteration, but that produces worse results. It is a greedy modularity maximization algorithms: At each iteration the algorithm merges the clusters that produce a higher increase (or lower decrease) of the modularity of the network until all the nodes belong to the same cluster. As for the divisive approach, the algorithm selects from the dendrogram the partition with highest modularity.

It works as follows:

1. Initialize each node with a different color, meaning that each node is initially in a different cluster.
2. Compute the modularity of the network. If the value is higher than the best one computed so far, save the value and the partition.
3. Aggregate the pairs of communities resulting in a higher increase or smaller decrease of the modularity of the network.
4. If there is more than one community go back to Step 2.

5. Return the partition with highest modularity, saved after Step 2.

A popular faster alternative, also based on a greedy approach that maximizes the modularity, is the heuristic proposed by Clauset et al. in [CNM04].

An extremely fast agglomerative approach is the *Louvain method* [BGLL08]. It is again a greedy modularity maximization algorithm and works in two phases that are repeated iteratively: In the first phase, each node chooses the community of its neighbor that brings the greatest increase in modularity; in the second phase, communities are replaced by supernodes and two supernodes are connected if there is at least an edge between nodes of the corresponding communities. The computational complexity of the algorithm is $\mathcal{O}(n \log n)$.

2.4.4 Spectral methods

Spectral clustering algorithms use properties of the spectrum of the matrix representation of graphs to perform clustering. The first contribution to spectral clustering algorithms, was probably the one of Fiedler [Fie73], where he realized that it is possible to get a bipartition of the graph with very low cut size using the eigenvector of the second smallest eigenvalue of the *Laplacian matrix* associated to the graph (see Appendix B.2). In fact, given a graph G with k connected components, the Laplacian of G is such that the eigenvalues $\lambda_1, \ldots, \lambda_k = 0$ and the eigenvectors associated to them identify the k connected components. If we consider a graph with k communities, i.e., we add a few links among the k disconnected components described before, the Laplacian spectrum will not be much different having $\lambda_1 = 0$ and the other $k-1$ eigenvalues close to λ_1. The consequence of this property is that the k eigenvectors associated with the k smallest eigenvalues are still representative of the k communities and can be used to partition the graph.

The motivations of such intuition rely on the *modes of vibrations* or *harmonics* of a string, which is the frequencies assumed by a tight string, as one of a guitar, when plucked. As we can see in Figure 2.6,[5] in the case of λ_2 the harmonics perfectly cuts in two equally sized parts. We can write down the physical equations related to such phenomenon and model it as a graph in which nodes are masses and edges are springs: solving the frequencies and shapes of harmonics we exactly get the eigenpairs of the Laplacian of the graph we used, and furthermore the second eigenvector cuts the graph in half.

Spectral partitioning

A simple algorithm to bisect a graph using the *Fiedler vector* [Fie73], i.e., the eigenvector associated to the second smallest eigenvalue of the Laplacian, works as follows: First compute the Fiedler vector $\mathbf{f}$, then sort it, and finally consider the partition formed by the $\frac{n}{2}$ nodes that have the biggest values in $\mathbf{f}$. More in detail:

[5] Source: https://people.eecs.berkeley.edu/~demmel/cs267/lecture20/lecture20.html

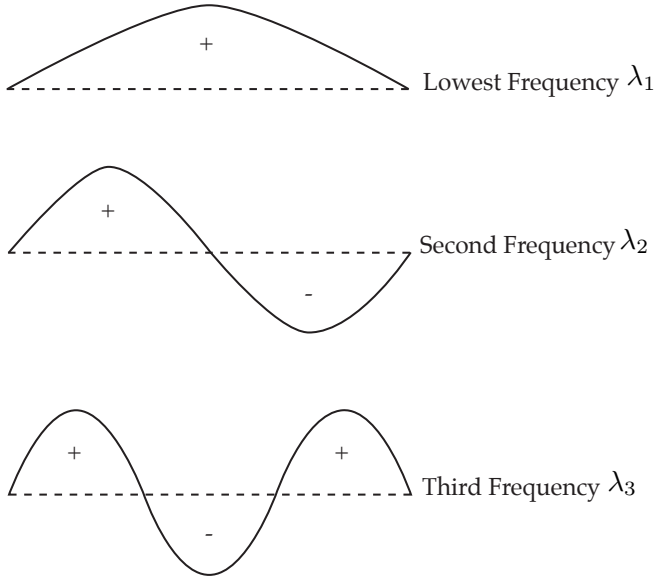


Figure 2.6: The frequencies of a vibrating string, intuition at the base of spectral partitioning. The symbols + and − indicates whether the string is above or below its rest position.

1. Compute the second eigenvalue λ_2 and its eigenvector $\mathbf{v}_2$ of L.
2. Sort the components of $\mathbf{v}_2$.
3. Create two candidate partitions as follows:
   (a) Put the n_1 nodes with largest values in S and the remaining n_2 in T as a first candidate partition.
   (b) Put the n_1 nodes with smallest values in S and the remaining n_2 in T as a second candidate partition.
4. Return the partition with lower cut size $e(S, T)$.

A simple variation where the sizes n_1 and n_2 are not defined works at follows:

1. Compute the second eigenvalue λ_2 and its eigenvector $\mathbf{v}_2$ of L.
2. Sort the components of $\mathbf{v}_2$ from the largest to the smallest.
3. Return the minimum *conductance* cut among the $n - 1$ cuts given by the ordering done in Step 2.

Such simple technique can be generalized to the case of unbalanced communities, e.g. of size n_1 and n_2, and applied recursively in each cluster to find more than two communities.

Analysis. Here we provide an overview of the analysis, skipping some calculations. Let $G = (V, E)$ be a graph such that $V = S \cup T$, A its adjacency matrix, and L its Laplacian matrix. We want to find a partition (S, T) such that $|S| = n_1$, $|T| = n_2$, and such that the cut of the partition, $e(S, T)$, is minimum. We can write the cut size as

$$e(S,T) = \frac{1}{2} \sum_{\substack{i,j \\ \mathcal{S}(i) \neq \mathcal{S}(j)}} A_{ij}. \tag{2.13}$$

Now we want to rewrite it in another notation which uses the graph Laplacian to exploit its spectral properties. Let's first define a mapping $\mathcal{S} : V \to \{S, T\}$ and the vector $\mathbf{s} = (s_1, \ldots, s_n)$, such that

$$s_i = \begin{cases} 1 & \text{if } \mathcal{S}(i) = S, \\ -1 & \text{if } \mathcal{S}(i) = T. \end{cases} \tag{2.14}$$

We can note that, using the components of $\mathbf{s}$

$$\frac{1}{2}(1 - s_i s_j) = \begin{cases} 1 & \text{if } \mathcal{S}(i) \neq \mathcal{S}(j), \\ 0 & \text{if } \mathcal{S}(i) = \mathcal{S}(j). \end{cases} \tag{2.15}$$

Moreover the adjacency matrix can be written in terms of the components of $\mathbf{s}$ too:

$$\sum_{i,j \in V} A_{ij} = \sum_{i \in V} d_i \tag{2.16}$$

$$= \sum_{i \in V} d_i s_i^2 \tag{2.17}$$

$$= \sum_{i,j \in V} d_i \delta_{ij} s_i s_j, \tag{2.18}$$

where d_i is the degree of the node i and δ_{ij} is the delta of Kronecker, i.e., $\delta_{ij} = 1$ if $i = j$ and $\delta_{ij} = 0$ otherwise.

Thus, combining the previous equations, we can rewrite the cut as

$$e(S,T) = \frac{1}{4} \sum_{i,j \in V} A_{i,j}(1 - s_i s_j) \tag{2.19}$$

$$= \frac{1}{4} \sum_{i,j \in V} (d_i \delta_{ij - A_{ij}}) s_i s_j \tag{2.20}$$

$$= \frac{1}{4} \sum_{i,j \in V} L_{ij} s_i s_j, \tag{2.21}$$

since the (i, j)-th element of the Laplacian matrix is defined exactly as $L_{ij} = d_v \delta_{ij} - A_i j$. Such formulation can be written in matrix form too, getting

$$e(S,T) = \frac{1}{4} \mathbf{s}^\intercal L \mathbf{s}. \tag{2.22}$$

Thus, we need to minimize $\mathbf{s}^\intercal L \mathbf{s}$ subject to $\sum_{i=1}^{n} s_i = n_1 - n_2$ and such that $s_i \in \{-1, 1\}$ for $i = 1, \ldots, n$. Such problem is hard and we will relax it letting

$\mathbf{s} \in \mathbb{R}^n$, but with the additional constraint that $\sum_{i=1}^{n} s_i^2 = n$. Mathematically, we can reformulate our original problem in its relaxed version as follows

$$\begin{aligned} \text{minimize} \quad & \mathbf{s}^\intercal L \mathbf{s} \\ \text{s.t.} \quad & \mathbf{1}^\intercal \mathbf{s} = n_1 - n_2 \\ & \|\mathbf{s}\|_2^2 = n. \end{aligned} \tag{2.23}$$

Using the method of Lagrangian multipliers we can solve the relaxed version of our optimization problem (we will not present the calculations), and get that the objective function (2.22) satisfies

$$e(S,T) = \frac{1}{4}\mathbf{s}^\intercal L \mathbf{s} \tag{2.24}$$

$$= \frac{1}{4}\mathbf{v}^\intercal L \mathbf{v} \tag{2.25}$$

$$= \frac{1}{4}\lambda \mathbf{v}^\intercal \mathbf{v} \tag{2.26}$$

$$= \frac{n_1 n_2}{n}\lambda, \tag{2.27}$$

where $\lambda, \mathbf{v}$ is any eigenpair of L, and because $\mathbf{v}^\intercal \mathbf{v} = 4\lambda\frac{n_1 n_2}{n}$ (from the skipped calculations). In addition we also get that $\mathbf{v}$ has to be orthogonal to $\mathbf{1}$, meaning that $\lambda \neq 0$ since $(0, \mathbf{1})$ is always an eigenpair of any Laplacian matrix. Which is then the best eigenpair to use? We want to choose the smallest λ, since n_1, n_2, and n are fixed: we know that $0 = \lambda_1 \leqslant \lambda_2 \leqslant \ldots \leqslant \lambda_n$ and, since we already excluded λ_1, the best choice is $(\lambda_2, \mathbf{v}_2)$. It follows (from the skipped calculations) that the components of $\mathbf{s}$ will be of the form

$$s_i = v_i + \frac{n_1 - n_2}{n}. \tag{2.28}$$

The original version of the problem, then, is solved by making $\mathbf{s}^\intercal \mathbf{s}$ as big as possible such that each component $s_i \in \{-1, 1\}$, which means that $s_i = 1$ for the n_1 largest components of $\mathbf{v}$ and $s_i = -1$ for the remaining ones. Obviously the symmetric solution is also valid and that is why we compute both and pick the one with smallest cut size, as in the algorithm.

Spectral modularity maximization

One of the ways to look for a partition is that of maximizing the modularity of the graph. Here we will see how such kind of maximization can be done with a spectral approach too, assuming we are just looking for two partitions.

We can note that $\delta(c_i, c_j) = \frac{(s_i s_j + 1)}{2}$, with

$$s_i = \begin{cases} -1 & \text{if } i \in c_1, \\ 1 & \text{if } i \in c_2, \end{cases} \tag{2.29}$$

where c_1 and c_2 being the two communities of the graph. Therefore we can rewrite the modularity as

$$Q = \frac{1}{2m}\sum_{i,j \in V}\left(A_{ij} - \frac{d_i d_j}{2m}\right) \tag{2.30}$$

$$= \frac{1}{4m} \sum_{i,j \in V} \mathbf{B}_{ij}(s_i s_j + 1) \tag{2.31}$$

$$= \frac{1}{4m} \sum_{i,j \in V} \mathbf{B}_{ij} s_i s_j \tag{2.32}$$

$$= \frac{1}{4m} \mathbf{s}^\intercal \mathbf{B} \mathbf{s}, \tag{2.33}$$

with $\mathbf{B} = A - \frac{\mathbf{d}\mathbf{d}^\intercal}{2m}$ being the modularity matrix, and $\mathbf{d}$ the degree vector.

As for the Fiedler vector, we want to maximize a function subject to the fact that the vector $\mathbf{s} \in \{-1, 1\}$, and again we relax the problem (adding the extra constraint $\mathbf{s}^\intercal \mathbf{s} = n$) and use the method of Lagrange multipliers to find the optimal solution. We find out that it holds that $\mathbf{B}\mathbf{s} = \beta \mathbf{s}$, being $\beta, \mathbf{s}$ an eigenpair of $\mathbf{B}$ and that the modularity is equal to

$$Q = \frac{1}{4m} \beta \mathbf{s}^\intercal \mathbf{s} = \frac{n}{4m} \beta \tag{2.34}$$

In this case, then, we want β to be as large as possible and we pick the largest eigenvalue of the matrix $\mathbf{B}$. In general we cannot choose $\mathbf{s} = \mathbf{v}_\beta$ because of the extra constraint, but we can choose $s_i = 1$ if $v_i > 0$ and $s_i = -1$ if $v_i \leqslant 0$.

So the algorithm follows:

1. Compute the biggest eigenvalue β and its eigenvector $\mathbf{v}_\beta$ of the modularity matrix of the graph.

2. Assign vertices to communities according to the sign of the corresponding component of $\mathbf{v}_\beta$.

Differently from the Laplacian, $\mathbf{B}$ is not a sparse matrix and then computing the eigenvalues could be computationally expensive. Luckily, though, $\mathbf{B}$ is similar to the sparse matrix A, as defined above, and it is still possible to compute $\mathbf{B}\mathbf{x}$ in $\mathcal{O}(n + m)$.

Other spectral methods

Another noticeable clustering algorithm exploiting the spectral properties of the graph is the one proposed by Shi and Malik [SM98, SM00], which makes use of the *Normalized Laplacian matrix* (see Appendix B.2). They proposed a global criterion for partitioning the graph, the *normalized cut*, which is related to the conductance. Such metric takes into account both the total dissimilarity between the different clusters and the total similarity within the clusters. A very nice and complete tutorial on spectral clustering, with insights from different perspectives on why it works and on what is happening, is the one of Von Luxburg [VL07].

All the spectral clustering techniques require to compute the eigenvectors associated with the k smallest eigenvalues of the Laplacian matrix of the input

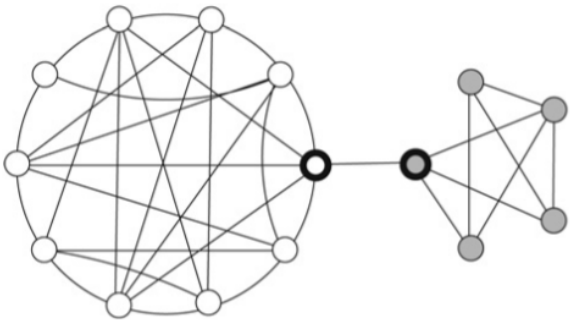


Figure 2.7: A graph with two communities (white nodes and grey nodes). A random walker starting in the white (grey) cluster will stay there for a long time until going to the bold node and then moving to the grey (white) cluster with probability $\frac{1}{6}$.

graph. If the graph is large the exact computation of the eigenvectors unfeasible, since it requires a time in $\mathcal{O}(n^3)$. Although, some techniques such as the *power method* [MPG29] can be used to approximate the first k eigenvector, with a running time that depends on the gap between λ_k and λ_{k+1}, i.e., the larger $|\lambda_{k+1} - \lambda_k|$ the faster is the convergence.

2.4.5 Dynamical methods

Another approach to graph clustering is the one of running dynamical processes on networks in which each node has a state that evolves as time passes according to some interaction with its neighbors [BBV08]. Dynamical approaches are closely related to spectral ones. In fact the adjacency matrix is strictly related to the *transition matrix* (see Appendix B.2) of a Random Walk on the graph (see Appendix E.2). In particular the components of the second largest eigenvalue of the transition matrix P measures how long it takes for the random walk to reach every node [Sch07]. Intuitively, in fact, two nodes in the same cluster should be easy to reach from each other, while if the random walk is currently in a cluster it should take more time to move to the other one, as in Figure 2.7.[6]

The concept that a random walker gets "stuck" in a community has been also used in *Infomap* [RAB09], an extremely effective information-theoretic methods to cluster graphs that could yield, for some network structures, dramatically dissimilar results from those obtained by modularity maximization algorithms given its different nature. The *map equation* [RAB09], based on the information flow, is the objective function of Infomap and is used to find a compressed representation of a set of random walks on the graph. Such compact representation can be expressed through clusters rather than single nodes. In practice, the more compact the representation the better the community detection result.

[6] Source: [Sch07]

Van Dongen discussed the use of Markov Chains for graph clustering in his PhD thesis [VD01]. He showed how applying a sequence of algebraic matrix operations on the transition matrix of a graph can isolate the clusters, removing inter-clusters edges. Meila and Shi, instead, showed the connection between the normalized cut and the stationary distribution of a random walk in the graph and thus linked the mathematics of random walks with the one of cut-based clustering [MS01a, MS01b].

Wu and Huberman [WH04] proposed a linear time algorithm to bisect a graph that uses notions of voltage drops across networks. Suppose that one can find two nodes u and v belonging to two different communities; initialize them with values 1 and 0 respectively, and then initialize all the other nodes value 0. At each iteration of the algorithm, all nodes except u and v update their values as the average of their neighbors values until convergence. Then values are sorted between 0 and 1: sweeping through the sorted values there will be a sudden drop at the border of two communities, which is used to identify them.

Label Propagation Algorithms

Another important class of dynamical clustering algorithms is the one of *Label Propagation Algorithms* (LPAs), introduced by Raghavan et al. [RAK07]. LPAs remind of epidemic processes on networks: Nodes interact with their neighbors and change their states based on some function of the neighbors states. Being most central for this thesis, they will be discussed in detail in Chapter 3.

3

Label Propagation Algorithms

3.1 General overview

Label Propagation Algorithms (LPAs) are dynamical clustering algorithms that resemble epidemic spread processes on networks. Each node of the network is initially assigned a label, which is updated to the most frequent in each node's neighborhood; one most prominent label will take over each cluster and then stop its diffusion give the sparsity of the cuts between the clusters. LPAs have proved to be fast and effective algorithms for community detection and, since their first proposal, have been modified and improved over time to handle huge graphs and evolving ones. A recent comprehensive survey on LPAs has been made by Garza & Schaeffer [GS19].

The generic pattern of such algorithms, that can be visualized in Figure 3.1, can be described as follows:

1. *Initialization*: A label taken from a finite set is assigned to each node according to some rule, e.g, they choose a unique value or sample a value uniformly at random from a finite set.

2. *Activation rule*: Nodes are activated following some *synchronous/asynchronous* rule, e.g., according to some permutation of their IDs or in rounds.

3. *Update rule*: Active nodes interact with their neighbors and update their labels according to some *local*, *majority-based* function.

After the first algorithm, known in literature as LPA, has been proposed and its effectiveness empirically assessed [RAK07], a new line of research started with the goal of improving the quality of the detected communities and the efficiency of the algorithm [LHLC09, LM10, BRSV11, ŠB11a, ŠB11b, XS13, ZRS+17], and to investigate more general settings, e.g., dynamic networks [XCS13, CDIG+15]. Many variants with small modifications on initialization rule, activation rule, and update rule have been proposed, but they have only been validated experimentally. On the other hand, there exist only few theoretical works. The absence of substantial theoretical progress in the analysis of LPAs is largely due to the lack of techniques for handling the interplay between the non-linearity of the local update rules and the topology of the graph.

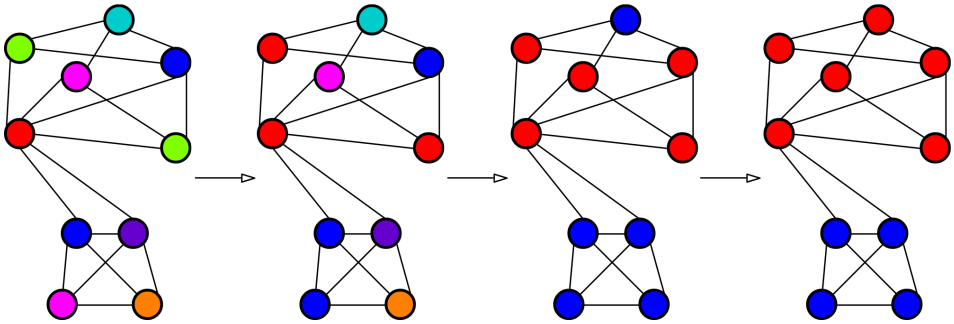


Figure 3.1: A visualization of Label Propagation Algorithm phases on a simple graph with two communities.

3.2 A brief history of LPAs

Raghavan et al. designed an extremely simple but fast and effective method based on label propagation for graph clustering, known as LPA [RAK07]. A unique label is initially assigned to each node, and, at each iteration, a random permutation of the nodes is considered. Each node, in the sequential order dictated by the permutation, update its label with the label shared by the majority of its neighbors; if there is no majority, the tie is broken with a random choice among the labels. This simple process makes the labels propagate across the graph: Many of them will soon start to disappear, while some will rapidly gain consensus. After some iterations the process ends in a stable situations in which each node has a label equal to the majority of the labels of its neighbors. The clusters are then defined as groups of nodes with the same label after the convergence of the algorithm, having more neighbors in its community than to others, according to the stopping criterion. The random tie-break rule does not guarantee a unique solution though, but empirical experiment show that the results are similar. The advantages of this technique are the fact that it does not need any information on the number and size of clusters, and that it has a low computational complexity: Each iteration runs in $\mathcal{O}(m)$ and the number of iterations needed seems to grow logarithmically with respect to it [LHLC09].

Tibély and Kertész [TK08] showed that LPA is equivalent to find the local minima of a simple zero-temperature kinetic Potts model, a generalization of the Ising model used in statistical mechanics to describe the spin interaction of electrons on a crystalline lattice.

Barber and Clark reformulated LPA as an equivalent optimization problem with an objective functions whose maxima identify communities in the graph [BC09]. They identified a practical drawback of the objective function: The global optimum is the trivial solution, i.e. the community containing all the nodes. Consequently they modified it adding some penalizers to it, making LPA stop to local maxima with higher modularity. Moreover they showed that a particular penalizer make the LPA optimize the modularity. This version of the algorithm is known as LPAm.

Gregory presented COPRA [Gre10], a generalized LPA able to detect overlapping communities through the use of *belonging coefficients*, parameters that indicate the strength of a node's membership to a community. COPRA is also able to work on bipartite graphs as showed in the experiments presented in the same work.

Another improvement to LPA is made by Liu and Murata with their LPAm+, an advanced modularity-specialized LPA [LM10]. The simple LPAm gets stuck in poor local maxima identifying communities with similar volumes. Instead, LPAm+ make use of a Multistep Greedy algorithm (MSG) [SC08] that can merge multiple communities at each time improving the modularity of the solution and offering a good trade-off between accuracy and speed. LPAm+ works in two phases: First it applies LPAm until it converges in a local maximum; then it uses MSG to merge pairs of communities that bring an increase in the modularity. This two phases are alternated until no further improvement in modularity can be reached. They also proposed a version of LPA that is able to detect communities in bipartite networks [LM09].

Cordasco and Gargano [CG10] improved the running time of the original LPA with a Semi-Synchronous LPA. The goal is to take advantage both from the synchronous model, where the propagation phase is performed in parallel by all the nodes, and from the asynchronous one, where the propagation phase is performed sequentially to guarantee convergence. The Semi-Synchronous LPA works in two phases. The coloring phase assign a color to the nodes such that adjacent nodes have different colors and is easy parallelizable. The propagation phase, inspired by [LM09], is divided into as many stages as many colors: In each stage only one color is active, and nodes with that color synchronously propagate their labels. Their solution has been tested on benchmark graphs and proved to be more effective, efficient, and stable (different runs produce similar results, regardless of the random initialization). Furthermore, they formally proved that their algorithm guarantees finite convergence time on any static graph.

Layered Label Propagation, presented by Boldi et al. [BRSV11], is an algorithm that builds on LPAm for clustering to reorder very large graphs favoring their compression. The interesting part, though, is that LPAm is modified in order to handle huge graphs: Modularity is not a good measure in big networks due to its resolution limit. In particular, a *discount term* is added to the objective function to increase the density of the sparsest community. This modification produces both a high number of clusters and clusters with a high number of nodes, following a power law distribution.

Xie and Szymanski proposed a generalization of LPA [XS11]. In particular, a new update rule and label propagation criterion are presented in order to improve both the efficiency and the effectiveness of LPA. This is obtained by reducing the number of iterations of the algorithm, by avoiding unnecessary updates: The procedure keeps a list of *active* nodes, i.e. the ones that are not at the boundary and will

update according to the label propagation phase, and skip the update phase for the nodes not present in the list. The generalized update rule, instead, gives a weight to the labels which is related to the *local clustering coefficient* of the node.[1] This idea follows the intuitions that when a node joins a group it may take into account not only the number of members of that group but also if they are connected to his neighbors.

Xie and Szymanski also proposed LabelRank [XS13], a modified version of LPA that makes the algorithm deterministic and extendible to other applications, e.g. community detection in dynamic graphs. LabelRank relies on four operators applied to the labels: *propagation*, *inflation*, *cutoff*, *conditional update*. The propagation is not only of a single label as in LPA, but each node maintains and propagates an entire distribution of labels. The inflation is similar to an operator used in MCL [VD01] but is applied on the labels, decoupling from the network structure; it increases the gap among labels with high and low probabilities. The cutoff remove labels with a probability below a certain threshold, to mitigate the memory overhead introduced by the algorithm. The conditional update, finally, is the real novelty of LabelRank: it preserve the algorithm from moving away from the highest quality communities via updating only when a node is significantly different from its neighbors.

Ugander and Backstrom [UB13] developed a balanced LPA with the specific goal of dealing with massive graphs while greedily maximizing the number of edges that are assigned to the same shard of a partition. They formulate the problem as a linear problem with constraints in the form of upper and lower bounds to the partition sizes. An iteration of the algorithm is similar to the one of LPA. In fact, compared to it, the only difference is that rather than updating the labels of all the nodes, balanced LPA first solves the constrained linear optimization problem and then updates as many nodes as possible without "breaking the balance". The algorithm works well with random initialization, but is able to converge within a single update step if labels are initialized using external geographic data, e.g. IP addresses in a computer network, domains in the Internet, geographical information provided by the users in social networks.

Kothapalli et al. [KPS13] are the first to provide a formal analysis of a simple algorithm based on LPA that solves community detection. In particular, they analyze Max-LPA both in terms of convergence time and quality of solution on clustered Erdös and Rényi graphs, i.e. graphs sampled from the planted partition model. Max-LPA works exactly as LPA, but whenever there is a tie in the update rule this is broken in favor of the largest label. They show that with general restrictions on the cuts sparsity Max-LPA is able to detect the clusters in just two rounds, conjecturing, with empirical evidences, that this would be possible in a polylogarithmic

[1] The local clustering coefficient measures how close the neighbors of a node are to a clique. It is given by the fraction of edges between the nodes within the neighborhood of a node over the number of possible between them.

number of rounds even when the clusters are much sparser.

Clementi et al. [CDIG+15] investigated on the problem of distributed community detection on evolving random graphs. They propose a simple LPA-based distributed protocol and proved that in some classes of random graphs their protocol is able to reconstruct the planted partition with high probability, even when the graph becomes disconnected over time. Moreover, they are the first to formally prove a logarithmic bound on the convergence time of a LPA-based protocol.

The majority in the previously discussed works improve the original LPA with respect to running time, effectiveness, and stability of the solution making it a usable algorithm for clustering huge real-world networks. Others introduce modifications of LPA in order to solve slightly different problems, e.g. soft clustering or community detection evolving graphs. Although, the trend in the line of research has been almost exclusively empirical: Most of the works present a variation of LPA and report experiments performed on benchmark graphs without providing analytical results about the algorithm. Formal analysis of LPAs, in fact, have started only more recently [KPS13, CDIG+15] opening many theoretical questions regarding the capabilities of LPAs.

4

Computational Dynamics

A very interesting and fascinating issue is the apparent difficulty to provide non-trivial mathematical characterizations of the *complexity from simplicity* phenomenon, i.e., the hidden interplay between simple local interactions occurring at microscopic level and the global evolution of the system at macroscopic level. Distributed computing naturally addresses such questions, focusing on the design and analysis of systems of computational agents that collectively achieve some global task. Several research efforts in distributed computing are moving towards formulating a general theory that characterizes the behavior of (simple) computing entities, in a way similar to that of statistical mechanics for interacting particle systems. Examples of such efforts are with respect to programmable matter [DDG+14, CDRR16] , chemical reaction networks [CSWB09, Dot14, CKW18], sensor networks [AAD+06, AFJ06].

One possible, algorithmic abstraction of the interplay between microscopic and macroscopic scales of complex phenomena is the notion of *dynamics*. Dynamics are rules to update an agent's state according to a function which is invariant with respect to time, network topology, and identity of an agent's neighbors, and whose arguments are only the agent's current state and those of its neighbors [MT17, Nat17].

Most of the scientific results in the sub-area of distributed computing that studies dynamics focus on their *computational power*, i.e., their ability to globally solve some specific problem such as consensus or synchronization, on their *convergence time*, i.e., the number of rounds needed to the dynamics to reach (and eventually maintain) a configuration that solves the problem, and their *fault tolerance* and *self-stabilization*, i.e., their ability to go to a configuration with certain properties of interest even in the presence of faulty communication links or malicious nodes that deviate their behavior from that of the dynamics' local rule.

Examples of other dynamics are: the Undecided-State [CGG+18], where there is an extra "undecided" state, nodes pick a random neighbor and becomes undecided if the pick has a different state or support that color if they were undecided; the h-Majority (e.g., studied for $h = 3$ in [BCN+17a, BCN+16]), where each node samples h neighbors and update its color to the most supported state among those that have been sampled; the 2-Median [DGM+11], where each node

samples two neighbors and update its state to the median state (assuming an ordering of the states) among the to sampled states and its state. Examples of problems for which dynamics have been successfully employed in order to design an efficient solution are *Noisy Rumor Spreading* [FN16], *Exact Majority* [MNRS17], and *Clock Synchronization* [BKN17].

In the following we focus on the *voter dynamics*, arguably the most simple nontrivial dynamics, and on the dynamics that are object of analysis in this thesis: the *2-Choices dynamics* and the *Averaging dynamics*.

4.1 Communication models

Consider a network of agents exchanging messages over a connected graph G according to some fixed communication model $\mathcal{M}$. The communication model $\mathcal{M}$ describes the rule that agents follow in the communication, e.g., the communication is synchronous or asynchronous, the size of the exchanged messages is limited, etc.

In this thesis we consider the *communication model* known as *synchronous uniform pull*. We suppose agents have a common notion of time that is marked into discrete *rounds*, making the communication *synchronous*. In each round, every agent chooses a message (e.g., their state); every agent then *pulls* the messages of a fixed-size subset of its neighbors. The subset of neighbors is chosen *uniformly at random* and independently from the choices of the other agents, making the communication *uniform*. Sometimes, the size of the exchanged messages can be limited to some fixed number of bits (either constant or dependent on the size of the network). The communication is noiseless, i.e., every transmitted message is received with no error. Moreover, we always assume the network is *anonymous*, i.e., agents have no distinct identities (e.g., they do not possess an ID they can share with their neighbors to mark their messages).

4.2 Consensus

Consensus, also known as *agreement*, is a simplified model of the way inconsistencies and disagreements are resolved in social networks, biological systems, and peer-to-peer systems [MNT14]. The problem of consensus is unsurprisingly a primary problem in the theory of distributed computing, with applications in many problems and as a primitive building block for more complex procedures. In distributed models that restrict the way in which nodes communicate, upper and lower bounds for consensus protocols give insights on how to break symmetry in networks in the case of balanced initial color configurations. Its original version [Dij74] can be informally defined as follows: A set of nodes, each having a state (an element taken from a fixed set Σ), interact via an underlying communication network with the

goal of agreeing on one of the elements of Σ initially held by at least one of the nodes.

For the consensus task in the presence of adversaries (*Byzantine Agreement*) [Rab83], the goal is to design a protocol that brings with the following properties:

- *Agreement:* all non-corrupted nodes eventually have the same state σ.
- *Validity:* the state σ must be a valid one, i.e., a state which was initially supported by at least one non-corrupted node.
- *Termination:* every non-corrupted node can correctly decide to stop running the protocol in some round.

The problem of *plurality consensus* requires that the nodes eventually obtain a consensus on the state that was initially supported by the plurality of the nodes, i.e., to the initially most supported state.

The problem of *proportionate consensus*, instead, requires that the system eventually reaches a consensus configuration such that the final state $\sigma \in \Sigma$ is supported with a probability proportional to the volume of the set of nodes that initially supported state σ, for each $\sigma \in \Sigma$.

4.3 Dynamics

As previously introduced, *dynamics* are *simple* and *lightweight* and typically have a behavior that rely on randomness. We report below a definition that tries to formalize such concept that appeared in [Nat17].

Definition 4.3.1 (Dynamics)**.** *A* dynamics *is a distributed algorithm characterized by a very simple structure, whereby the state of a node at round t depends only on its state and on a symmetric function of the multi-set of states of its neighbors at round $t-1$, while the update rule is the same for every graph and for every node and it does not change over time.*

Within the constraints of the previous definition, it may still be possible to come up with computational rules that appear cumbersome and unnatural. The goal of Definition 4.3.1 is just to provide a reference, and not to replace reliance of the scientific community on the real world phenomena the concept is intended to capture.

4.3.1 Voter dynamics

The voter dynamics is arguably the simplest nontrivial dynamics: at each round, each node looks at a random neighbor and copies its state. A more formal definition can be found in Algorithm 1.

Algorithm 1 Voter dynamics

Initialization: At round $t = 0$, every node $v \in V$ has a value $\boldsymbol{x}^{(t)}(v) \in \{0, 1\}$.

Update: At each subsequent round $t \geqslant 1$, every node $v \in V$ samples a neighbor $u \in N_v$ uniformly at random and sets $\boldsymbol{x}^{(t)}(v) = \boldsymbol{x}^{(t-1)}(u)$.

The Voter dynamics, in its asynchronous version where only one node chosen uniformly at random updates its state, has been studied in statistical mechanics [Lig06, Lig09]. The study of the synchronous version of the dynamics, the one previously described, has been first considered in distributed computing as a *proportionate agreement protocol* [HP01]. The voter dynamics reaches proportionate consensus in a polynomial number of rounds, but it turns out to be slow, i.e., $\Omega(n)$ rounds, on many networks with small diameter such as complete graphs or expander graphs.

4.3.2 2-Choices dynamics

The 2-Choices dynamics (more formally described in Algorithm 2 and illustrated in Figure 4.1) is a local synchronous protocol that works as follows: In each round, each node u chooses two neighbors v, w uniformly at random with replacement; if v and w support the same state, then u updates its own state to their state, otherwise u keeps its previously supported state.

Algorithm 2 2-Choices dynamics

Initialization: At round $t = 0$, every node $v \in V$ follows one the following rules to choose its initial value $\boldsymbol{x}^{(t)}(v)$:

- *Deterministic initialization*: sets $\boldsymbol{x}^{(t)}(v) = 0$ if $v \in S$ and $\boldsymbol{x}^{(t)}(v) = 1$ if $v \in V \setminus S$, for some $S \in V$ (used in Chapter 5).
- *Random initialization*: sets $\boldsymbol{x}^{(t)}(v) \in \{0, 1\}$ uniformly at random and independently from other nodes (used in Chapter 6).

Update: At each subsequent round $t \geqslant 1$, every node $v \in V$:

1. *Sampling*: samples two neighbors $u, w \in N_v$ uniformly at random with replacement.
2. *Majority*: sets $\boldsymbol{x}^{(t)}(v) = 0$ if $\boldsymbol{x}^{(t-1)}(u) + \boldsymbol{x}^{(t-1)}(v) + \boldsymbol{x}^{(t-1)}(w) < \frac{3}{2}$ and $\boldsymbol{x}^{(t)}(v) = 1$ otherwise.

It can be seen as an implicit *sparsification* of a *deterministic majority dynamics*, where each node updates its state the the most frequent in its neighborhood, given the sampling used in the update rule. Moreover, it can be seen also as a 2-*majority*

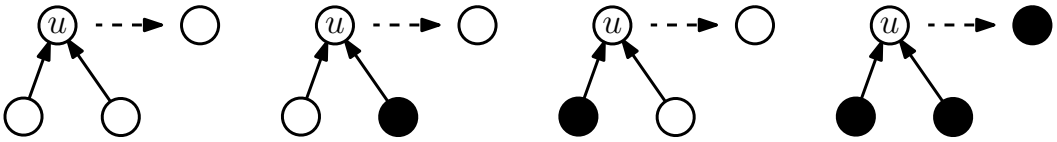


Figure 4.1: The update rule of the 2-Choices dynamics: The node updates its current state only if the two randomly chosen neighbors share the same state; in all other cases, the node keeps its current state.

dynamics where ties are not broken at random, but towards the previously supported state. It can arguably be considered the simplest type of dynamics after the Voter dynamics and, until now, it constitutes one of the few processes whose behavior has been characterized on non-complete topologies [CER14, CER+15, CRRS17].

In the binary case, i.e., where each node can support one of two possible states, the 2-Choices dynamics turns out to be equivalent to the 2-Median dynamics. The results obtained in [DGM+11] thus follow as immediate consequence, i.e., the dynamics essentially converges to an *almost*[1] *plurality consensus* in $\mathcal{O}(\log n)$ number of rounds if the initial plurality is supported enough.

A deep analysis of this dynamics for a number of colors $k = \mathcal{O}(n^{\varepsilon})$ has been recently presented by Elsässer et al. [EFK+16]. They consider an initial configurations having some initial *bias* s, i.e., the plurality is well represented, and show that this dynamics is an efficient, fault-tolerant protocol for *plurality consensus*, converging in $\mathcal{O}(k \log n)$ rounds.

Ghaffari et al. [GL17] provide a tight bound on the convergence time starting from any configuration with a polynomially-bounded number of colors, i.e., $k = \mathcal{O}\left(\frac{n}{\log n}\right)$. Their analysis uses and improves an approach previously introduced by Becchetti et al. [BCN+16] to analyze 3-Majority dynamics in a similar setting.

Cooper et al. [CER14] analyze, for the first time, the 2-Choices dynamics on d-regular graphs. They gave different bounds on the convergence time depending on several parameters such as the degree d, the initial bias s, and on the *expansion* of the graph (the spectral gap between the largest and second largest eigenvalues of the transition matrix of a simple random walk on the graph).

Further analysis of the 2-Choices dynamics on non-complete graph topologies consider the binary case over non-regular expander graphs [CER+15], where essentially it is proved that the dynamics converges to a plurality consensus with high probability after a polylogarithmic number of rounds whenever the initial *bias* is greater than a given function of the network's *expansion* [CER14] (with milder assumptions with respect to those of [CER14]). Such result was later generalized to the multi-color case on regular expander graphs [CRRS17].

[1] A certain number of nodes could still support the other state.

4.3.3 Averaging dynamics

The *averaging dynamics* works as follows. Initially, each node of the network has a (possibly random) value; then, at each synchronous time step every node updates its state to the (possibly weighted, if the underlying graph is weighted) average of the values held by its neighbors. In this section we consider a simple Rademacher initialization, i.e., each node independently chooses a value uniformly at random in $\{-1, 1\}$. The dynamics is more formally described in Algorithm 3, and a visualization of its update rule can be seen in Figure 4.2.[2]

Algorithm 3 Averaging dynamics

Initialization: At round $t = 0$, every node $v \in V$ independently samples its value $\boldsymbol{x}^{(0)}(v)$ from $\{-1, +1\}$ uniformly at random.

Update: At each subsequent round $t \geqslant 1$, every node $v \in V$:

1. *Averaging*: updates its value $\boldsymbol{x}^{(t)}(v)$ to the weighted average of the values of its neighbors at the end of the previous round.
2. *Labeling*: if $\boldsymbol{x}^{(t)}(v) \geqslant \boldsymbol{x}^{(t-1)}(v)$ then v sets $\texttt{label}^{(t)}(v) = 1$; otherwise v sets $\texttt{label}^{(t)}(v) = 0$.

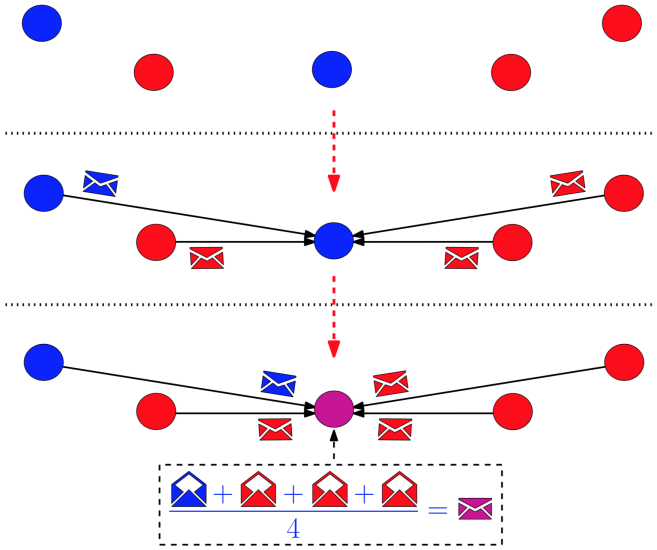


Figure 4.2: A visualization of the update rule of the averaging dynamics.

The averaging dynamics is one of the simplest, yet most interesting examples of linear dynamics. It always converges to a consensus when the underlying network is connected and not bipartite; in more detail, the consensus is given by the weighted global average of the initial states of the nodes. Note that the consensus is not valid,

[2] Source: [Nat17]

since the global average could not be one of the initial states. The convergence time of the dynamics strictly depends on the second largest eigenvalue λ_2 of the transition matrix of a random walk on the underlying graph.

The averaging dynamics has been analyzed by Becchetti et al. [BCN+17b] on graphs with two equal-sized communities sampled from the regular Stochastic Block Model [HLL83]. They show that, with a simple labeling scheme that assigns to each node a "blue" state if its value has increased with respect to the previous iteration and "red" otherwise, the algorithm reconstruct the *hidden partition* of the graph by labeling all the nodes in one community of one colors and the nodes in the other community of the other color after $\mathcal{O}(\log n)$ rounds. They extend the analysis also to the case of k equal-sized communities and bound the error in the reconstruction when the graph is *almost* regular as a function of such irregularity.

In a follow-up work [BCM+18] Becchetti et al. also extend the analysis considering another communication model, in which the nodes update their states asynchronously, i.e., in each round a node is sampled uniformly at random, and with a sparsified update rule, i.e., the sampled node averages its value with that of a random neighbor. The results they obtained in this different setting are comparable to those of the original work: the dynamics converges in $\mathcal{O}(n \log n)$ asynchronous iterations and reconstructs the hidden partition.

II

2-Choices Dynamics

5

Phase Transition on Core-Periphery Networks

In this chapter, we aim at contributing to the general understanding of the evolution of simple dynamics in richer classes of network topologies. We study the behavior of the 2-Choices dynamics (Algorithm 2) both theoretically and empirically on *core-periphery networks*, graphs that exhibit a bipartition of the nodes into a small, dense *core* and a large, sparse *periphery*. In particular we consider a *natural* initial configuration in which the core and the periphery have different opinions. Our experiments on real-world networks show that the execution of the 2-Choices dynamics tends to fall mainly within two opposite kinds of possible behavior:

1. *Consensus:* The opinion of the core spreads in the periphery and takes over the network in a short time.

2. *Metastability:* The periphery *resists* and, although the opinion of the core may continuously "infect" agents in the periphery, most of them retain the initial opinion.

In Theorem 5.2.1 we initially consider the setting in which *agents in the core are stubborn*, i.e., they don't change their initial opinion. We later show, in Corollary 5.2.1, how to substitute this assumption with milder hypotheses on the core's structure. In Section 5.2 we provide a careful bound on the expected change of the number of agents supporting a given opinion. Together with the use of Chernoff bounds, we obtain a concentration of probability around the expected evolution, and identifiy a *phase transition* phenomenon:

> **Informal description of Corollaries 5.2.1 and 5.2.2.** There exists a universal constant $c^\star = \frac{\sqrt{2}-1}{2}$ such that, on a core-periphery network of n agents, if the dominance parameter c_d is greater than $c^\star$, an *almost-consensus* is reached within $\mathcal{O}(\log n)$ rounds, with high probability; otherwise, if c_d is less than $c^\star$, a *metastable* phase in which the periphery retains its opinion takes place, namely, although the opinion of the core may continuously "infect" agents in the periphery, most of them will retain the initial opinion for $n^{\omega(1)}$ rounds, with high probability.

We remark that the evolution of the 2-Choices dynamics, together with the latter assumption on the stubbornness of the core, can be regarded as a SIS-like epidemic model [Het00, EJN02]. In such a model, the network is the subgraph induced by the periphery and the infection probability is given by the 2-Choices dynamics, which also determines a certain probability of *spontaneous infection* (that in the original process corresponds to the fact that agents in the periphery interact with agents in the core, or vice versa). This interpretation of our results may be of independent interest.

We validate our theoretical predictions by extensive experiments on real-world networks chosen from a variety of domains including social networks, communication networks, road networks, and the web and thoroughly discuss them in Section 5.3. The experiments showed some weaknesses of the core extraction heuristic used in [ALP+14] and, to avoid those drawbacks, we designed a new core extraction heuristic which repeatedly calculates densest-subgraph approximations. Our experimental results on real-world networks show a strong correlation with the theoretical predictions made by our model. They further suggests an empirical threshold larger than $c^\star$ for which the aforementioned correlation is even stronger. We discuss which aspects of the current theoretical model may be responsible for such discrepancy, and thus identify possibilities for a model which is even more accurate.

We remark that our investigation represents an original contribution with respect to the line of research on *consensus*, as it shows a drastic change in behavior for the 2-Choices dynamics on an arguably *typical* broad family of social networks which is not directly characterized by expansion properties. In particular, the convergence to the core's opinion in our theoretical and experimental results is a highly nontrivial fact when compared to previous analytical works (see Section 5.3).

Overall, our theoretical and experimental results highlight new potential relations between the typical core-periphery structure in social and economic networks and the behavior of simple opinion dynamics – both, in terms of getting insights into the driving forces that may determine certain structures to appear frequently in real-world networks, as well as in terms of the possibility to provide analytical predictions on the outcome of simplistic models of interaction in networks of agents.

5.1 Preliminaries

Core-periphery networks (Figure 5.1) are typical economic and social networks which exhibit a core-periphery structure, a well-known concept in the analysis of social networks in sociology and economics [BE00, DJ10], which defines a bipartition of the agents into *core* and *periphery*, such that certain key features are identified.

We consider an axiomatic framework that has been introduced in previous work in computer science [ALP+14], which is based on two parameters only, *dominance* and *robustness*. The ranges for these parameters in the theorems we obtain include

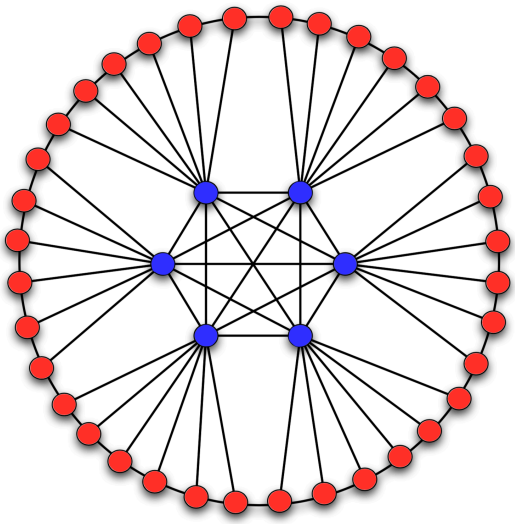


Figure 5.1: A visualization of a Core-Periphery network. The core is pictured in blue, while the periphery in red.

the values used in Section 5.3, in which our results are experimentally validated on important datasets of real-world networks.

Intuitively, the core is a set of agents that *dominates* the rest of the network. In order to do so, it maintains a large amount of external influence on the periphery, higher than or at least comparable to the internal influence that the periphery has on itself. Similarly, to maintain its *robustness*, namely to hold its position and stick to its opinions, the core must be able to resist the "outside" pressure in the form of external influence. To achieve that, the core must maintain a higher (or at least not significantly lower) influence on itself than the external influence exerted on it by the periphery. Both, high dominance and high robustness, are essential for the core to be able to maintain its dominating status in the network. Moreover, it seems natural for the core size to be as small as possible. In social-network terms this is motivated by the idea that if membership in a social elite entails benefits, then keeping the elite as small as possible increases the share for each of its members.

The above requirements are formalized in the following axioms [ALP+14]. Given a network $G = (V, E)$ and two subsets of agents $A, B \subseteq V$, let $e(A, B) := \{(u, v) \mid u \in A, v \in B, (u, v) \in E\}$ be the *set of cut edges* between A and B. The *density* of a set $X \subseteq V$ is defined as $\delta(X) = |e(X, X)|/|X|$. Let c_d and c_r be two positive constants and let $V = \mathcal{C} \dot{\cup} \mathcal{P}$, where $\mathcal{C}$ is the set of agents in the core and $\mathcal{P}$ the set of agents in the periphery. Then, the axioms are as follows:

- *Dominance*: The core's influence dominates the periphery, i.e., $|e(\mathcal{C}, \mathcal{P})| \geqslant c_d \cdot |e(\mathcal{P}, \mathcal{P})|$.
- *Robustness*: The core can withstand outside influence from the periphery, i.e., $|e(\mathcal{C}, \mathcal{C})| \geqslant c_r \cdot |e(\mathcal{P}, \mathcal{C})|$.

- *Compactness*: The core is a minimal set satisfying the above dominance and robustness axioms.[1]

- *Density*: The core is denser than the whole network, i.e., $\delta(\mathcal{C}) > \delta(V)$.

Our analytical and experimental results leverage the dominance and robustness axioms only (see Definition 5.2.1), showing how assumptions on the values of c_d and c_r are sufficient to provide a good characterization of the behavior of the dynamics.

5.2 Theoretical analysis

We give a formal analysis of the 2-Choices dynamics on a specific network topology, i.e., on Core-Periphery networks. We use colors instead of opinions to facilitate intuitive understanding of the analysis. Specifically, we consider the setting in which the agents belonging to the core $\mathcal{C}$ initially support the color *blue* while the remaining agents, from the periphery $\mathcal{P}$, support the color *red*. We show that, depending on the value of some parameters describing the core-periphery structure of the network, either the opinion of the core rapidly spreads among almost the whole network (*almost-consensus*, i.e., almost all the agents support the *blue* color after a few rounds) or most of the periphery resists for a long time (*metastability*, i.e., most agents in $\mathcal{P}$ remain *red*).

First, we formally describe our characterization of Core-Periphery networks and of the 2-Choices dynamics. Then, we prove two technical results which will be exploited in order to provide a rigorous analysis of the 2-Choices dynamics on Core-Periphery networks.

Definition 5.2.1. *Let $n \in \mathbb{N}$ and $\varepsilon, c_r, c_d \in \mathbb{R}^+$, with $\varepsilon \in \left[\frac{1}{2}, 1\right]$. We define an $(n, \varepsilon, c_r, c_d)$-Core-Periphery network $G = (V, E)$ as a network where each agent either belongs to the core $\mathcal{C}$ or to the periphery $\mathcal{P}$, i.e., $V = \mathcal{C} \dot{\cup} \mathcal{P}$, with $|\mathcal{C}| = n^{\varepsilon}$ and $|\mathcal{P}| = n$, and such that:*

- *for each agent $u \in \mathcal{C}$, it holds that*

$$|N(u) \cap \mathcal{C}| = c_r \cdot |N(u) \cap \mathcal{P}|, \tag{5.1}$$

- *for each agent $v \in \mathcal{P}$, it holds that*

$$|N(v) \cap \mathcal{C}| = c_d \cdot |N(v) \cap \mathcal{P}|, \tag{5.2}$$

where $N(v)$ is the set of neighbors of agent v.

The definition we just provided matches the requirements of the core-periphery structure as axiomatized by Avin et al. [ALP+14]. However, observe that it is more

[1] The core is a minimal set and not necessarily the minimum one.

restrictive: The values c_r and c_d define properties that hold for each agent of the network and not only globally, i.e., for the partition induced by the core.

In order to analyze the 2-Choices dynamics on Core-Periphery networks, we present a more general technical result that will be exploited later. In fact, both in the analysis of the *almost-consensus* and of the *metastability*, we can focus on the worst-case scenario for the core and for the periphery: Each time an agent in one of the two sets picks a neighbor in the other set, that neighbor has the initial color of the set it belongs to. Alternatively, such a scenario can be seen as the following variant of the 2-Choices dynamics.

Definition 5.2.2 (Biased-2-Choices(p, σ) dynamics)**.** *Let $p \in [0,1]$ be a constant and let $\sigma \in \{red, blue\}$ be a color. We define the Biased-2-Choices(p, σ) dynamics as a variation of the 2-Choices dynamics: Every time an agent picks a neighbor, with probability p that neighbor supports color σ regardless of its actual color.*

The technical result we present considers a network of agents running the Biased-2-Choices(p, σ) dynamics, all having the same initial color different from σ. Informally, it shows that there exists a value $p^\star$ such that if the agents are running the Biased-2-Choices(p, σ) dynamics with $p > p^\star$, then the color σ rapidly spreads among almost the whole network, while if $p < p^\star$, then most of the network does not support the color σ for a superpolynomial number of rounds. Given a set of agents A, we define the *volume* of A as $\mathrm{vol}(A) := \sum_{v \in A} d_v$, where d_v is the degree of v.

Theorem 5.2.1. *Let $G = (V, E)$ be a network of n agents such that each agent v has a color σ_v and $d_v = \omega(\log n)$ neighbors. Let $p \in [0,1]$ be a constant. Then, starting from a configuration where all agents initially support the* red *color and letting the agents run Biased-2-Choices(p, blue), it holds that:*

- Almost-consensus: *If $p > 3 - 2\sqrt{2}$, then the agents reach a configuration such that the* volume *of agents supporting the* blue *color is $(1 - o(1))\mathrm{vol}(V)$ within $\mathcal{O}(\log n)$ rounds, w.h.p.*
- Metastability: *If $p < 3 - 2\sqrt{2}$, then the* volume *of the* blue *agents never exceeds $\frac{1-3p}{4(1-p)}\mathrm{vol}(V)$ for any* poly(n) *number of rounds, w.h.p.*

Proof. Let $B^{(t)}$ be the set of *blue* agents and $R^{(t)} = V \setminus B^{(t)}$ be the set of *red* agents at time t. For any agent v, let $N_R(v) = N(v) \cap R^{(t)}$ be the set of *red* neighbors and $N_B(v) = N(v) \cap B^{(t)}$ the set of *blue* neighbors of v. Furthermore, let $r_v^{(t)}$ be the number of *red* neighbors of v at time t, i.e., $r_v^{(t)} = |N_R(v)|$. Let $\phi_v^{(t)} = \frac{r_v^{(t)}}{d_v}$ be the fraction of *red* agents in the neighborhood of v; let $\phi_{\min}^{(t)} = \min_{v \in V} \phi_v^{(t)}$ and $\phi_{\max}^{(t)} = \max_{v \in V} \phi_v^{(t)}$ be, respectively, the minimum and maximum fractions of *red* neighbors among all agents in V. Let $\mathbf{c}^{(t)} \in \{red, blue\}^n$ be the configuration of the colors of the agents at time t. In the following, for the sake of readability,

whenever we omit the time index, we refer to the value at time t, e.g., ϕ_v stands for $\phi_v^{(t)}$. Similarly, we concisely denote with $\mathbf{P}_R(v) = \mathbf{P}(v \in R^{(t+1)} \mid \mathbf{c}^{(t)})$ the probability that agent v will be supporting the *red* color in the next round of the Biased-2-Choices(p, *blue*), i.e.,

$$\mathbf{P}_R(v) = \begin{cases} 1 - \big(p + (1-p)(1-\phi_v)\big)^2 & \text{if } v \in R, \\ (1-p)^2\phi_v^2 & \text{if } v \in B. \end{cases} \tag{5.3}$$

Furthermore, note that:

$$\min_{w \in R} \mathbf{P}_R(w) = 1 - \big(p + (1-p)(1-\phi_{\min})\big)^2, \tag{5.4}$$

$$\min_{w \in B} \mathbf{P}_R(w) = (1-p)^2\phi_{\min}^2. \tag{5.5}$$

Given a configuration $\mathbf{c}^{(t)}$, we can give a lower bound for the expected fraction of *red* neighbors of any agent v as follows:

$$\mathbf{E}\left[\phi_v^{(t+1)} \;\middle|\; \mathbf{c}^{(t)}\right] = \frac{1}{d_v}\left(\sum_{w \in N_R(v)} \mathbf{P}_R(w) + \sum_{w \in N_B(v)} \mathbf{P}_R(w)\right) \tag{5.6}$$

$$\geqslant \frac{1}{d_v}\left(|N_R(v)| \min_{w \in R} \mathbf{P}_R(w) + |N_B(v)| \min_{w \in B} \mathbf{P}_R(w)\right) \tag{5.7}$$

$$= \frac{r_v}{d_v} \min_{w \in R} \mathbf{P}_R(w) + \left(1 - \frac{r_v}{d_v}\right) \min_{w \in B} \mathbf{P}_R(w) \tag{5.8}$$

$$= \frac{r_v}{d_v}\left(1 - \big(p + (1-p)(1-\phi_{\min})\big)^2\right) + \left(1 - \frac{r_v}{d_v}\right)(1-p)^2\phi_{\min}^2 \tag{5.9}$$

$$= \frac{r_v}{d_v}\left(1 - \big(p + (1-p)(1-\phi_{\min})\big)^2 - (1-p)^2\phi_{\min}^2\right) + (1-p)^2\phi_{\min}^2 \tag{5.10}$$

$$\geqslant \phi_{\min}\left(1 - \big(p + (1-p)(1-\phi_{\min})\big)^2 - (1-p)^2\phi_{\min}^2\right) + (1-p)^2\phi_{\min}^2 \tag{5.11}$$

$$= \phi_{\min}\left(1 - \big(p + (1-p)(1-\phi_{\min})\big)^2 + (1-p)^2(1-\phi_{\min})\phi_{\min}\right) \tag{5.12}$$

$$= \phi_{\min}\left(1 - 2(1-p)^2\phi_{\min}^2 + (1-p)(3-p)\phi_{\min} - 1\right). \tag{5.13}$$

Note that we could cancel out 1 and -1, however, leaving them facilitates the analysis. Note that in Equation (5.11), we can lower bound $\frac{r_v}{d_v}$ with $\phi_{\min}$ because its coefficient is non-negative, i.e., $\big(1 - \big(p + (1-p)(1-\phi_{\min})\big)^2 - (1-p)^2\phi_{\min}^2\big) \geqslant 0$ for every $p, \phi_{\min} \in [0,1]$.
Conversely, we can upper bound the expectation using $\phi_{\max}$, i.e.,

$$\mathbf{E}\left[\phi_v^{(t+1)} \;\middle|\; \mathbf{c}^{(t)}\right] \leqslant \phi_{\max}\left(1 - 2(1-p)^2\phi_{\max}^2 + (1-p)(3-p)\phi_{\max} - 1\right). \tag{5.14}$$

Note that the lower and the upper bound for the expectation have the same form. In fact, defining the functions

$$f_p(\phi) := 2(1-p)^2\phi^2 - (1-p)(3-p)\phi + 1, \tag{5.15}$$

$$g_p(\phi) := \phi(1 - f_p(\phi)), \tag{5.16}$$

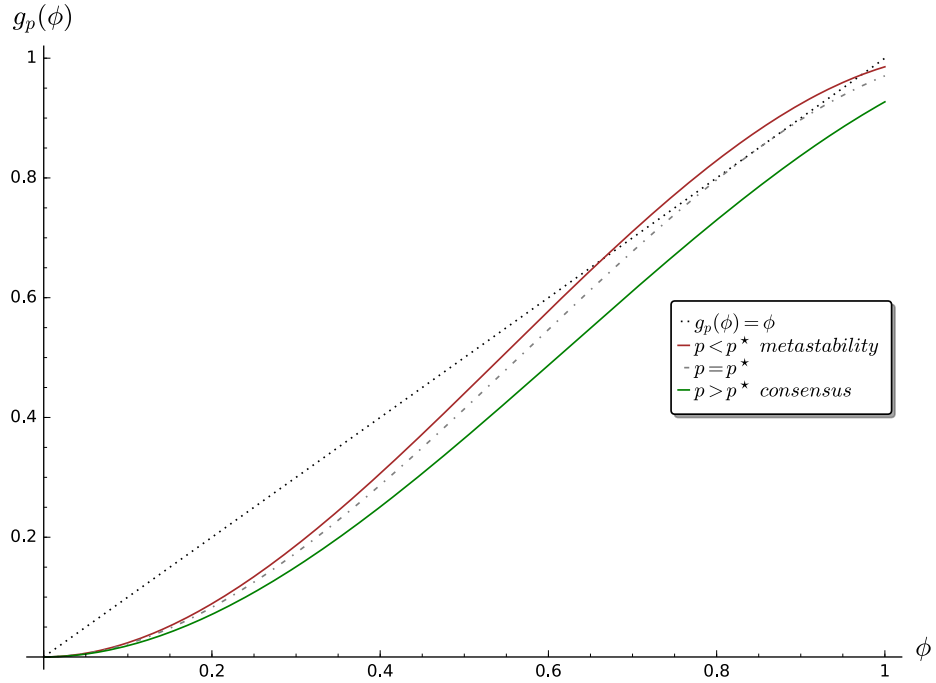


Figure 5.2: A plot of $g_p(\phi)$, where we call the critical value $p^\star = 3 - 2\sqrt{2}$. The red line is for a value $p < p^\star$, while the green line is for a value $p > p^\star$; the dotted gray line, instead, is for $p = p^\star$ and is tangent to the bisector line. When the line is above the bisector the expected fraction of red neighbors in the next round increases; when the line is below the bisector the expected fraction of red neighbors in the next round decreases.

the lower and the upper bound for the expectation can respectively be written as $g_p(\phi_{\min})$ and $g_p(\phi_{\max})$. Thus, analyzing $f_p(\phi)$, we can see for which values of p the function $g_p(\phi)$ is increasing or decreasing.

Before proving the *almost-consensus* and the *metastability* configurations that can be reached by the agents running the Biased-2-Choices(p, *blue*), we study $f_p(\phi)$ in order to characterize the bounds for the expectation. The roots of $f_p(\phi)$ are in $\frac{3-p\pm\sqrt{p^2-6p+1}}{4(1-p)}$ while its derivative is $f'_p(\phi) = 4(1-p)^2\phi - (1-p)(3-p)$. It follows that $f_p(\phi)$ has a minimum point in $\bar{\phi} = \frac{3-p}{4(1-p)}$. Moreover, the sign of $f_p(\bar{\phi})$ exclusively depends on p. In fact

$$f_p(\bar{\phi}) > 0 \qquad \text{if } p > 3 - 2\sqrt{2}, \tag{5.17}$$

$$f_p(\bar{\phi}) < 0 \qquad \text{if } p < 3 - 2\sqrt{2}, \tag{5.18}$$

since $f_p(\phi)$ is convex and in (5.17) the discriminant of $f_p(\phi)$ is negative, while in (5.18) the discriminant is positive.

A plot of g_p that shows the consensus and metastable behaviors of the process while p varies is available in Figure 5.2.

Almost-consensus. Let $p > 3 - 2\sqrt{2}$. Let $f_p(\bar{\phi}) = \varepsilon$ be the local minimum of f. Note that ε is positive because of (5.17) and it is a constant since it only depends on p and $\bar{\phi}$, which are both constants. Due to the convexity of $f_p(\phi)$, it holds that $f_p(\phi) \geqslant \varepsilon$ for every $\phi \in [0,1]$. Thus, for every $v \in V$, we have that

$$\mathbf{E}\left[\phi_v^{(t+1)} \;\middle|\; \mathbf{c}^{(t)}\right] \leqslant g_p(\phi_{\max}) = \phi_{\max}(1 - f_p(\phi_{\max})) \leqslant \phi_{\max}(1-\varepsilon). \tag{5.19}$$

Thus, we can apply a multiplicative form of the Chernoff bounds directly to the upper bound of the expected fraction of *red* neighbors at the next round, as shown in [DP09, Exercise 1.1], and get that

$$\begin{aligned}
&\mathbf{P}\left(\phi_v^{(t+1)} > (1-\varepsilon^2)\phi_{\max} \;\middle|\; \mathbf{c}^{(t)}\right) && (5.20)\\
&= \mathbf{P}\left(\phi_v^{(t+1)} > (1+\varepsilon)(1-\varepsilon)\phi_{\max} \;\middle|\; \mathbf{c}^{(t)}\right) && (5.21)\\
&= \mathbf{P}\left(r_v^{(t+1)} > (1+\varepsilon)(1-\varepsilon)\phi_{\max} d_v \;\middle|\; \mathbf{c}^{(t)}\right) && (5.22)\\
&\leqslant e^{-\frac{\varepsilon^2}{3}(1-\varepsilon)\phi_{\max} d_v} \leqslant e^{-2\log n} = n^{-2}, && (5.23)
\end{aligned}$$

where in the last inequality we can assume that the configuration $\mathbf{c}^{(t)}$ is such that $\phi_{\max} \geqslant \frac{6\log n}{\varepsilon^2(1-\varepsilon)d_v}$, which is $\phi_{\max} = o(1)$, since $d_v = \omega(\log n)$ by definition. Thus, using the union bound over all the agents, we get that $\phi_{\max}^{(t+1)} \leqslant (1-\varepsilon^2)\phi_{\max}$, w.h.p. Formally,

$$\begin{aligned}
&\mathbf{P}\left(\exists v \in V : \phi_v^{(t+1)} > (1-\varepsilon^2)\phi_{\max} \;\middle|\; \mathbf{c}^{(t)}\right) && (5.24)\\
&\leqslant \sum_{v\in V} \mathbf{P}\left(\phi_v^{(t+1)} > (1-\varepsilon^2)\phi_{\max} \;\middle|\; \mathbf{c}^{(t)}\right) && (5.25)\\
&\leqslant \sum_{v\in V} n^{-2} = n^{-1}. && (5.26)
\end{aligned}$$

Such a multiplicative decrease rate of the expected maximum fraction of *red* neighbors implies that $\phi_{\max}$ is in the order of $o(1)$ within $\mathcal{O}(\log n)$ rounds of the Biased-2-Choices(p, *blue*). Again, applying the union bound over all the agents, we get that this still happens w.h.p.

Metastability. Let $p < 3 - 2\sqrt{2}$. Define $f_p(\bar{\phi}) = -\varepsilon$ to be the local minimum of f. Recall that ε is positive because of (5.18) and it is a constant since it only depends on the constants p and $\bar{\phi}$.

Then, using the fact that $g_p(\phi)$ is monotonically nondecreasing for $\phi \in [0,1]$, for every $\phi \geqslant \bar{\phi}$ we have that

$$g_p(\phi) \geqslant g_p(\bar{\phi}) = \bar{\phi}(1 - f_p(\bar{\phi})) = \bar{\phi}(1+\varepsilon). \tag{5.27}$$

We can now use a multiplicative form of the Chernoff bounds in order to show that if the fraction of *red* neighbors of an agent v is at least $\bar{\phi}$, then the probability that the number of *red* neighbors of v in the next round is lower than $\bar{\phi}$ is negligible.

Formally, let $\mathbf{c}^{(t)}$ be an arbitrary configuration such that $\phi_{\min} \geqslant \bar{\phi}$, e.g., the initial configuration $\mathbf{c}^{(0)}$ has this property. First, note that due to $g_p(\phi) \geqslant g_p(\bar{\phi})$, we have that

$$\mathbf{E}\left[\phi_v^{(t+1)} \;\middle|\; \mathbf{c}^{(t)}\right] \geqslant g_p(\phi_{\min}) \geqslant g_p(\bar{\phi}) = \bar{\phi}(1+\varepsilon). \tag{5.28}$$

Then, it follows that

$$\begin{aligned}
&\mathbf{P}\left(\phi_v^{(t+1)} < \bar{\phi} \;\middle|\; \mathbf{c}^{(t)}\right) && (5.29)\\
&= \mathbf{P}\left(\phi_v^{(t+1)} < \frac{1}{1+\varepsilon}\bar{\phi}(1+\varepsilon) \;\middle|\; \mathbf{c}^{(t)}\right) && (5.30)\\
&= \mathbf{P}\left(\phi_v^{(t+1)} < \left(1-\frac{\varepsilon}{1+\varepsilon}\right)\bar{\phi}(1+\varepsilon) \;\middle|\; \mathbf{c}^{(t)}\right) && (5.31)\\
&\leqslant \mathbf{P}\left(\phi_v^{(t+1)} < \left(1-\frac{\varepsilon}{1+\varepsilon}\right)\mathbf{E}\left[\phi_v^{(t+1)} \;\middle|\; \mathbf{c}^{(t)}\right] \;\middle|\; \mathbf{c}^{(t)}\right) && (5.32)\\
&= \mathbf{P}\left(r_v^{(t+1)} < \left(1-\frac{\varepsilon}{1+\varepsilon}\right)d_v\mathbf{E}\left[\phi_v^{(t+1)} \;\middle|\; \mathbf{c}^{(t)}\right] \;\middle|\; \mathbf{c}^{(t)}\right) && (5.33)\\
&\leqslant \exp\left(-\frac{\varepsilon^2}{3(1+\varepsilon)^2}d_v\mathbf{E}\left[\phi_v^{(t+1)} \;\middle|\; \mathbf{c}^{(t)}\right] \;\middle|\; \mathbf{c}^{(t)}\right) && (5.34)\\
&\leqslant \exp\left(-\frac{\varepsilon^2}{3(1+\varepsilon)^2}d_v\bar{\phi}(1+\varepsilon)\right) && (5.35)\\
&= e^{-\Omega(d_v)} = e^{-\omega(\log n)} = n^{-\omega(1)}. && (5.36)
\end{aligned}$$

Applying the union bound over all the agents, we get

$$\begin{aligned}
&\mathbf{P}\left(\substack{\exists t\in\mathrm{poly}(n)\\ \exists v\in V} : \phi_v^{(t+1)} < \bar{\phi} \;\middle|\; \mathbf{c}^{(t)}\right) && (5.37)\\
&\leqslant \sum_{\substack{t\in\mathrm{poly}(n)\\ v\in V}} \mathbf{P}\left(\phi_v^{(t+1)} < \bar{\phi} \;\middle|\; \mathbf{c}^{(t)}\right) && (5.38)\\
&= \sum_{\substack{t\in\mathrm{poly}(n)\\ v\in V}} n^{-\omega(1)} = n^{-\omega(1)}. && (5.39)
\end{aligned}$$

Thus, with high probability we have that $\phi_{min} \geqslant \bar{\phi}$ for every polynomial number of rounds. Before we can use this to finish the proof, note that $\sum_{v\in B} d_v = \sum_{v\in V}(d_v - r_v)$, by simply counting the number of *blue* endpoints of an edge in two different ways. Using that $\phi_v \geqslant \phi_{\min} \geqslant \bar{\phi}$ for each v, we have

$$\begin{aligned}
\mathrm{vol}(B^{(t)}) &= \sum_{v\in B} d_v && (5.40)\\
&= \sum_{v\in V}(d_v - r_v) && (5.41)\\
&= \sum_{v\in V} d_v\left(1-\frac{r_v}{d_v}\right) && (5.42)\\
&\leqslant (1-\bar{\phi})\sum_{v\in V} d_v && (5.43)\\
&= \frac{1-3p}{4(1-p)}\mathrm{vol}(V). && (5.44)
\end{aligned}$$

This means, the volume of the *blue* agents never exceeds a fraction of $\frac{1-3p}{4(1-p)}$ of the total volume of the graph, w.h.p. ■

5.2.1 Phase transition on core-periphery networks

Theorem 5.2.1 has several interesting implications on the behavior of the 2-Choices dynamics on Core-Periphery networks which we describe in the remainder of this section.

In the following, we always assume an initial configuration in which all agents in the core $\mathcal{C}$ are *blue* and all agents in the periphery $\mathcal{P}$ are *red*. Now, let $G = (V, E)$ be an $(n, \varepsilon, c_r, c_d)$-Core-Periphery network. Furthermore, let $q_{\mathcal{C}} = \frac{|N(u) \cap \mathcal{P}|}{d_u}$ be the probability that an agent $u \in \mathcal{C}$ picks a neighbor in the periphery, and let $q_{\mathcal{P}} = \frac{|N(v) \cap \mathcal{C}|}{d_v}$ be the probability that an agent $v \in \mathcal{P}$ picks a neighbor in the core. The relations below follow from Definition 5.2.1:

$$c_r = |N(u) \cap \mathcal{C}| \,/\, |N(u) \cap \mathcal{P}| = (1 - q_{\mathcal{C}}) \,/\, q_{\mathcal{C}} \qquad \forall u \in \mathcal{C}, \tag{5.45}$$

$$c_d = |N(v) \cap \mathcal{C}| \,/\, |N(v) \cap \mathcal{P}| = q_{\mathcal{P}} \,/\, (1 - q_{\mathcal{P}}) \qquad \forall v \in \mathcal{P}. \tag{5.46}$$

Let $c^\star = \frac{\sqrt{2}-1}{2}$ be the constant which later defines a threshold between *almost-consensus* and *metastability* behavior. We get

$$c_r = 1 \,/\, (c^\star + \delta_r) \quad \Longrightarrow \quad q_{\mathcal{C}} = 3 - 2\sqrt{2} + \delta'_r \tag{5.47}$$

$$c_d = c^\star + \delta_d \quad \Longrightarrow \quad q_{\mathcal{P}} = 3 - 2\sqrt{2} + \delta'_d \tag{5.48}$$

for δ_r and δ'_r (δ_d and δ'_d) which are either both positive or both negative.

For the almost-consensus result, we require a high robustness of the core such that it remains monochromatic for a logarithmic number of rounds. The following lemma is needed to link the robustness with this property.

Lemma 5.2.1. *Let ε and δ be two positive constants. Let $G = (V, E)$ be a graph of n^ε agents, and let $0 \leqslant p \leqslant n^{-(\varepsilon+\delta)/2}$. Starting from a configuration such that each agent initially supports the* blue *color, within $\mathcal{O}(\log(n))$ rounds of the Biased-2-Choices(p, red) no agent becomes* red*, w.h.p.*

Proof. The probability that an agent v changes its color to *red* at time t, given that all the other agents are still *blue*, is

$$\mathbf{P}\left(v \in R^{(t+1)} \;\middle|\; V = B^{(t)}\right) = p^2 \leqslant \left(n^{-(\varepsilon+\delta)/2}\right)^2 = n^{-(\varepsilon+\delta)}. \tag{5.49}$$

Applying the union bound over all the agents and over $\tau = \mathcal{O}(\log n)$ rounds, we get

$$\mathbf{P}\left(\substack{\exists t \leqslant \tau \\ \exists v \in V} : v \in R^{(t+1)} \;\middle|\; V = B^{(t)}\right) \leqslant \frac{n^\varepsilon \cdot \tau}{n^{\varepsilon+\delta}} = \mathcal{O}(n^{-\delta}). \tag{5.50}$$

Thus, all agents in the graph remain *blue* for any logarithmic number of rounds, w.h.p. ■

If $c_r \geqslant n^{(\varepsilon+\delta)/2}$, by Equation (5.45) it follows that

$$q_{\mathcal{C}} = 1 \,/\, (c_r + 1) < 1 \,/\, c_r \leqslant n^{-(\varepsilon+\delta)/2}. \tag{5.51}$$

We are now ready to prove the *almost-consensus* behavior of the 2-Choices dynamics on Core-Periphery networks.

Corollary 5.2.1. *Let $c^\star = \frac{\sqrt{2}-1}{2}$ and let ε and δ be two positive constants. Let $G = (V, E)$ be an $(n, \varepsilon, c_r, c_d)$-Core-Periphery network such that each agent in the network has $\omega(\log n)$ neighbors. If $c_r > n^{(\varepsilon+\delta)/2}$ and $c_d > c^\star$ by a constant, then the agents reach a configuration such that the* volume *of* blue *agents is $(1-o(1))\mathrm{vol}(V)$ within $\mathcal{O}(\log n)$ rounds of the 2-Choices dynamics, w.h.p.*

Proof sketch. Since $c_r > n^{(\varepsilon+\delta)/2}$, we can apply Lemma 5.2.1 and thus the agents in the core never change color for $\mathcal{O}(\log n)$ rounds, w.h.p. Therefore, for any $\mathcal{O}(\log n)$ number of rounds, the process is equivalent to a Biased-2-Choices($q_{\mathcal{P}}$, *blue*) run by the periphery. Since $c_d > c^\star$ and thus $q_{\mathcal{P}} > 3 - 2\sqrt{2}$ by Equation (5.48), we can apply Theorem 5.2.1 and get almost-consensus on the *blue* color in a logarithmic number of rounds, w.h.p. ■

Finally we can also prove a *metastability* phenomenon of the 2-Choices dynamics on Core-Periphery networks.

Corollary 5.2.2. *Let $c^\star = \frac{\sqrt{2}-1}{2}$ be a universal constant. Let $G = (V, E)$ be an $(n, \varepsilon, c_r, c_d)$-Core-Periphery network such that each agent in the network has $\omega(\log n)$ neighbors. Then, with high probability, we have that for each round t and for any* poly(n) *number of rounds of the 2-Choices dynamics the following two statements hold:*

- *if $c_r > \frac{1}{c^\star}$ by a constant, then* $\mathrm{vol}(B^{(t)}) \geqslant \frac{3}{4}\mathrm{vol}(\mathcal{C})$,
- *if $c_d < c^\star$ by a constant, then* $\mathrm{vol}(R^{(t)}) \geqslant \frac{3}{4}\mathrm{vol}(\mathcal{P})$,

where $B^{(t)}$ are the blue *agents and $R^{(t)}$ the* red *agents at time t.*

Proof sketch. Consider the following worst case scenario: Every time an agent in the core (periphery) chooses a random neighbor belonging to the periphery (core), then that neighbor is *red* (*blue*). In this scenario, the 2-Choices dynamics can be thought of as two independent Biased-2-Choices(p, σ) in which for the core $p = q_{\mathcal{C}}$ and σ = *red*, and for the periphery $p = q_{\mathcal{P}}$ and σ = *blue*. From $c_r > \frac{1}{c^\star}$ and $c_d < c^\star$ and Equations (5.47) and (5.48), it follows that $q_{\mathcal{C}}$ and $q_{\mathcal{P}}$ are less than $3 - 2\sqrt{2}$. By applying the metastability result of Theorem 5.2.1, we get that the volume of the adversary never exceeds $\frac{1-3p}{4(1-p)}$ of the network's volume. Since both $q_{\mathcal{C}}$ and $q_{\mathcal{P}}$ are smaller than $3 - 2\sqrt{2}$, we have that $\frac{1-3p}{4(1-p)} \leqslant \frac{1}{4}$ (as the inequality is tight for $p = 0$ and its value is decreasing). Thus, the volumes of *red* (*blue*) agents in the core (periphery) are at most a fraction of $\frac{1}{4}$. Therefore, the volumes of *blue* and *red* agents in the whole network are at least $\frac{3}{4}$ of the volumes of $\mathcal{C}$ and $\mathcal{P}$, respectively. ■

5.3 Simulations on real-world networks

In Section 5.2 we formally studied the 2-Choices dynamics on Core-Periphery networks, observing a phase transition phenomenon that appears on a *dominance* threshold $c^\star = \frac{\sqrt{2}-1}{2}$. Here, we report the results of the empirical data obtained by simulating the 2-Choices dynamics on real-world networks. Furthermore, we discuss our results and compare them with our theoretical analysis. The source code of the experiments is freely available.[2]

We simulated the 2-Choices dynamics on 70 real-world networks, 25 of them taken from KONECT [Kun13] and 45 of them taken from SNAP [LK14]. Detailed information regarding the networks and the results of the experiments are reported in Table 5.1. The networks chosen for the experiments are drawn from a variety of domains including social networks, communication networks, road networks, and web graphs; moreover, they range in size from thousands of nodes and thousands of edges up to roughly one million of nodes and tens of millions of edges. Before simulating the 2-Choices dynamics, we pre-process the networks in order to match the theoretical setting. In particular, for all the networks, we remove the orientation of the edges and all loops, and we work on the largest (w.r.t. the number of nodes) connected component.

The first issue we faced simulating the 2-Choices dynamics was the extraction of the set of agents representing the core. In fact, there is no exact definition of what a *good* core is with respect to *dominance* and *robustness* values. We started by using a simple heuristic to extract the core, namely the *k-rich-club* method [ZM04]: This method establishes the core $\mathcal{C}$ as the set of k agents with highest degree and the periphery $\mathcal{P}$ as the remaining agents. Avin et al. [ALP+14] empirically show that when k is at the *symmetry point*, i.e., k is chosen such that $vol(\mathcal{C}) \approx vol(\mathcal{P})$, the core found by this method is sublinear in size with respect to the number of agents of the network. We remark that if $vol(\mathcal{C}) = vol(\mathcal{P})$ then, from the definitions of *robustness* and *dominance*, it follows that $c_r = \frac{1}{c_d}$.

We initially used the *k-rich-club* method to extract the core but noted that this simple heuristic produces a core with very low *robustness* values, contrary to what common sense would suggest to be a *good core*. In particular, low robustness values imply that the *dominance* values never go below our theoretical threshold $c^\star$ (see columns $\bar{c}_r$ and $\bar{c}_d$ in Table 5.1), which hinders the comparison between theoretical and experimental results. Indeed, in our theoretical analysis we assume that the core never changes color, i.e., that the *robustness* is high; however, in the experiments the core was very unstable when using the *k-rich-club* method. The main issue of this method is that it does not take into account the topological structure of the network, e.g., if we consider a regular graph with a well defined core-periphery

[2] https://github.com/ioemilio/opinion_dynamics

Table 5.1: Experimental results of the 2-Choices dynamics on networks with core-periphery structure. *Source* reports the source of the dataset, i.e. SNAP (S) or KONECT (K). The values c_r and c_d are the *robustness* and *dominance* obtained using the *densest-core* method; the values $\bar{c}_r$ and $\bar{c}_d$ are the *robustness* and *dominance* obtained using the *k-rich-club* method. $\mathcal{C}$ and $\mathcal{P}$ are the fraction of experiments in which the core's and the periphery's color respectively spread to reach an *almost-consensus*, while $\mathcal{M}$ is the fraction of experiments in which there is *metastability*, all having the core extracted with the *densest-core* method. K stands for thousand, M for million.

Dataset (Source)	$\lvert V\rvert$	$\lvert E\rvert$	c_r ($\bar{c}_r$)	c_d ($\bar{c}_d$)	$\mathcal{C}$	$\mathcal{P}$	$\mathcal{M}$
Chicago (K)	0.8K	1.6K	6.55 (0.10)	0.15 (9.72)	0.00	0.00	1.00
email-Eu-core (S)	0.9K	32.1K	0.75 (0.53)	1.32 (1.88)	0.92	0.08	0.00
Euroroad (K)	1.0K	2.6K	5.53 (0.62)	0.18 (1.61)	0.00	0.00	1.00
Blogs (K)	1.2K	33.4K	0.62 (0.38)	1.57 (2.60)	0.00	0.00	1.00
Traffic Control (K)	1.2K	4.8K	1.25 (0.51)	0.78 (1.96)	0.00	0.00	1.00
Protein (K)	1.4K	3.8K	0.90 (0.33)	1.10 (2.95)	1.00	0.00	0.00
US Airport (K)	1.5K	34.4K	0.54 (0.48)	1.82 (2.10)	0.00	0.00	1.00
Stelzl (K)	1.6K	6.2K	1.03 (0.36)	0.96 (2.73)	1.00	0.00	0.00
Bible (K)	1.7K	18.1K	0.74 (0.54)	1.33 (1.84)	0.98	0.02	0.00
Hamster full (K)	2.0K	32.1K	0.96 (0.66)	1.02 (1.51)	1.00	0.00	0.00
Opsahl OF (K)	2.9K	31.2K	0.76 (0.55)	1.30 (1.81)	1.00	0.00	0.00
OpenFlights (K)	3.3K	38.4K	0.73 (0.50)	1.35 (1.98)	0.80	0.00	0.20
bitcoin-alpha (S)	3.7K	28.2K	0.53 (0.39)	1.87 (2.52)	1.00	0.00	0.00
ego-Facebook (S)	4.0K	176.4K	4.83 (1.53)	0.20 (0.65)	0.00	0.00	1.00
ca-GrQc (S)	4.1K	26.8K	3.33 (1.29)	0.29 (0.77)	0.00	0.00	1.00
US power grid (K)	4.9K	13.1K	3.17 (0.53)	0.31 (1.86)	0.00	0.00	1.00
bitcoin-otc (S)	5.8K	42.9K	0.52 (0.38)	1.88 (2.59)	1.00	0.00	0.00
p2p-Gnutella08 (S)	6.2K	41.5K	1.20 (0.53)	0.82 (1.86)	0.00	1.00	0.00
Route Views (K)	6.4K	25.1K	0.30 (0.16)	3.26 (6.13)	0.96	0.04	0.00
wiki-Vote (S)	7.0K	201.4K	0.60 (0.44)	1.64 (2.24)	1.00	0.00	0.00
p2p-Gnutella09 (S)	8.1K	52.0K	1.08 (0.53)	0.91 (1.86)	0.00	1.00	0.00
ca-HepPh (S)	8.6K	49.6K	1.40 (0.69)	0.71 (1.44)	1.00	0.00	0.00
p2p-Gnutella06 (S)	8.7K	63.0K	0.91 (0.53)	1.09 (1.87)	1.00	0.00	0.00
p2p-Gnutella05 (S)	8.8K	63.6K	0.93 (0.54)	1.06 (1.83)	0.86	0.14	0.00
PGP (K)	10.6K	48.6K	2.54 (1.18)	0.39 (0.84)	1.00	0.00	0.00
p2p-Gnutella04 (S)	10.8K	79.9K	0.91 (0.52)	1.08 (1.90)	1.00	0.00	0.00
ca-HepTh (S)	11.2K	235.2K	3.49 (2.39)	0.28 (0.41)	0.00	0.00	1.00
ca-AstroPh (S)	17.9K	393.9K	1.54 (0.84)	0.64 (1.18)	1.00	0.00	0.00
ca-CondMat (S)	21.3K	182.5K	1.70 (0.68)	0.58 (1.46)	1.00	0.00	0.00
p2p-Gnutella25 (S)	22.6K	109.3K	0.72 (0.41)	1.37 (2.43)	1.00	0.00	0.00
E.A.T. (K)	23.1K	594.1K	0.60 (0.48)	1.64 (2.07)	0.96	0.04	0.00
Cora citation (K)	23.1K	178.3K	1.37 (0.54)	0.72 (1.83)	1.00	0.00	0.00
CAIDA (K)	26.4K	106.7K	0.31 (0.16)	3.13 (6.03)	1.00	0.00	0.00
p2p-Gnutella24 (S)	26.4K	130.7K	0.71 (0.42)	1.39 (2.34)	1.00	0.00	0.00
cit-HepTh (S)	27.4K	704.0K	1.33 (0.74)	0.74 (1.34)	1.00	0.00	0.00
Digg (K)	29.6K	169.5K	0.59 (0.49)	1.67 (2.01)	1.00	0.00	0.00
Linux (K)	30.8K	426.4K	0.47 (0.24)	2.10 (4.14)	0.90	0.10	0.00
email-Enron (S)	33.6K	361.6K	0.71 (0.54)	1.39 (1.84)	1.00	0.00	0.00

Dataset (Source)	$\|V\|$	$\|E\|$	c_r ($\bar{c}_r$)	c_d ($\bar{c}_d$)	$\mathcal{C}$	$\mathcal{P}$	$\mathcal{M}$
cit-HepPh (S)	34.4K	841.5K	1.34 (0.61)	0.74 (1.61)	1.00	0.00	0.00
Internet topology (K)	34.7K	215.4K	0.61 (0.32)	1.62 (3.08)	0.88	0.00	0.12
p2p-Gnutella30 (S)	36.6K	176.6K	0.82 (0.44)	1.21 (2.23)	1.00	0.00	0.00
loc-Brightkite (S)	56.7K	425.8K	0.99 (0.71)	1.00 (1.40)	1.00	0.00	0.00
p2p-Gnutella31 (S)	62.5K	295.7K	0.78 (0.44)	1.27 (2.26)	1.00	0.00	0.00
soc-Epinions1 (S)	75.8K	811.4K	0.72 (0.60)	1.37 (1.65)	1.00	0.00	0.00
Slashdot081106 (S)	77.3K	937.1K	0.51 (0.46)	1.93 (2.13)	0.98	0.02	0.00
soc-Slashdot0811 (S)	77.3K	938.3K	0.51 (0.46)	1.93 (2.13)	1.00	0.00	0.00
ego-Twitter (S)	81.3K	2.6M	1.12 (0.57)	0.89 (1.75)	0.00	0.00	1.00
Slashdot090216 (S)	81.8K	995.3K	0.53 (0.48)	1.87 (2.08)	1.00	0.00	0.00
Slashdot090221 (S)	82.1K	1.0M	0.53 (0.48)	1.87 (2.08)	1.00	0.00	0.00
soc-Slashdot0922 (S)	82.1K	1.0M	0.53 (0.47)	1.87 (2.08)	1.00	0.00	0.00
Prosper loans (K)	89.1K	6.6M	0.82 (0.47)	1.21 (2.10)	0.00	0.00	1.00
Livemocha (K)	104.1K	4.3M	0.47 (0.38)	2.10 (2.56)	0.94	0.06	0.00
Flickr (K)	105.7K	4.6M	2.27 (1.07)	0.43 (0.92)	0.00	0.00	1.00
ego-Gplus (S)	107.6K	24.4M	0.95 (0.54)	1.04 (1.82)	0.00	0.00	1.00
epinions (S)	119.1K	1.4M	0.64 (0.52)	1.53 (1.89)	1.00	0.00	0.00
Github (K)	120.8K	879.7K	0.88 (0.70)	1.12 (1.41)	1.00	0.00	0.00
Bookcrossing (K)	185.9K	867.2K	0.52 (0.34)	1.90 (2.87)	1.00	0.00	0.00
loc-Gowalla (S)	196.5K	1.9M	1.14 (0.80)	0.87 (1.24)	0.02	0.00	0.98
email-EuAll (S)	224.8K	679.8K	0.16 (0.06)	6.19 (14.4)	0.00	0.00	1.00
web-Stanford (S)	255.2K	3.8M	2.52 (0.37)	0.39 (2.68)	0.00	0.00	1.00
amazon0302 (S)	262.1K	1.7M	2.61 (0.44)	0.38 (2.23)	1.00	0.00	0.00
com-DBLP (S)	317.0K	2.0M	1.43 (0.70)	0.69 (1.42)	1.00	0.00	0.00
web-NotreDame (S)	325.7K	2.1M	2.63 (0.60)	0.37 (1.65)	0.00	0.00	1.00
com-amazon (S)	334.8K	1.8M	1.72 (0.32)	0.57 (3.03)	0.98	0.00	0.02
amazon0312 (S)	400.7K	4.6M	2.18 (0.41)	0.45 (2.41)	0.00	0.00	1.00
amazon0601 (S)	403.3K	4.8M	2.08 (0.40)	0.47 (2.44)	0.00	0.00	1.00
amazon0505 (S)	410.2K	4.8M	2.01 (0.41)	0.49 (2.40)	0.00	0.00	1.00
web-BerkStan (S)	654.7K	13.1M	1.60 (0.43)	0.62 (2.30)	0.00	0.00	1.00
web-Google (S)	855.8K	8.5M	1.77 (0.42)	0.56 (2.33)	0.00	0.00	1.00
roadNet-PA (S)	1.0M	3.0M	7.35 (1.01)	0.13 (0.98)	0.00	0.00	1.00

structure (which satisfies Definition 5.2.1) the *k-rich-club* method would identify the core to be a random subset of nodes.

Therefore, we introduce a novel heuristic for extracting the core which takes the network topology into account by prioritizing the *robustness* of the core over its *dominance*. The procedure, which we refer to as *densest-core* method, is described in Algorithm 4. Informally, it iteratively extracts the densest subgraph from the network and adds it to the core unless the core's volume becomes too large. In order to compute this constrained densest subgraph, it uses a variation of the 2-approximation algorithm [Cha00], which chooses every time the densest subgraph that will not make the core's volume larger than the periphery's volume.

We apply the *densest-core* method to the networks and, as expected, we obtain higher *robustness* and lower *dominance* values compared to the *k-rich-club* method. The data reported in Table 5.1 shows that the *robustness* of the core extracted

Algorithm 4 Densest-Core Extraction

```
1:  procedure DensestCore(G)
2:      C* ← ∅
3:      do
4:          C ← ∅; D ← G
5:          while D ≠ ∅ do
6:              v ← LowestDegreeNode(D)
7:              D ← D \ {v}
8:              if Density(D) > Density(C) and
                   FractionOfVolume(C* ∪ D) ⩽ 1/2 then
9:                  C ← D
10:             end if
11:         end while
12:         C* ← C* ∪ C
13:         G ← G \ C
14:     while C ≠ ∅
15:     return C*
```

by our method is higher in all the considered datasets but one. Indeed, we finally obtain *dominance* values below the theoretical threshold $c^\star$.

We proceed as follows: We initialize all the agents in $\mathcal{C}$ with *blue* and all the agents in $\mathcal{P}$ with *red*. Then, we simulate the 2-Choices dynamics on each network, keeping track of the volumes of *blue* and *red* agents in each iteration. We declare an *almost-consensus* on the majority's color if within $|V|$ iterations either the *red* or the *blue* agents reach a volume greater than 95% of the network's volume. Otherwise we consider the simulation *metastable* – waiting for a superpolynomial number of rounds would be infeasible. The experiments were repeated 50 times for each network.

As can be observed in Figure 5.3, there exists an *empirical threshold* $\sigma = \frac{1}{2}$ which is different from the theoretical one. In fact, in 81% of the datasets with a *dominance* above the threshold the 2-Choices dynamics converges to an *almost-consensus* while in 86% of the datasets with a *dominance* below the threshold, the 2-Choices dynamics ends up in a *metastable* phase. The empirical threshold is greater than the theoretical threshold because of several factors: (i) in the experiments the core actually changes color to a small extent (while in the theoretical part we ignored such small *perturbations*), and it consequently lowers the probability for an agent in the periphery to pick the core's color; (ii) the real-world network we used in the experiments do not have the regularity assumptions of the networks that we consider in the analysis; (iii) in the experiments we declare metastability only after $|V|$ iterations and this increases the likelihood of metastable runs. The gap between the theoretical and the empirical threshold should be closed in future

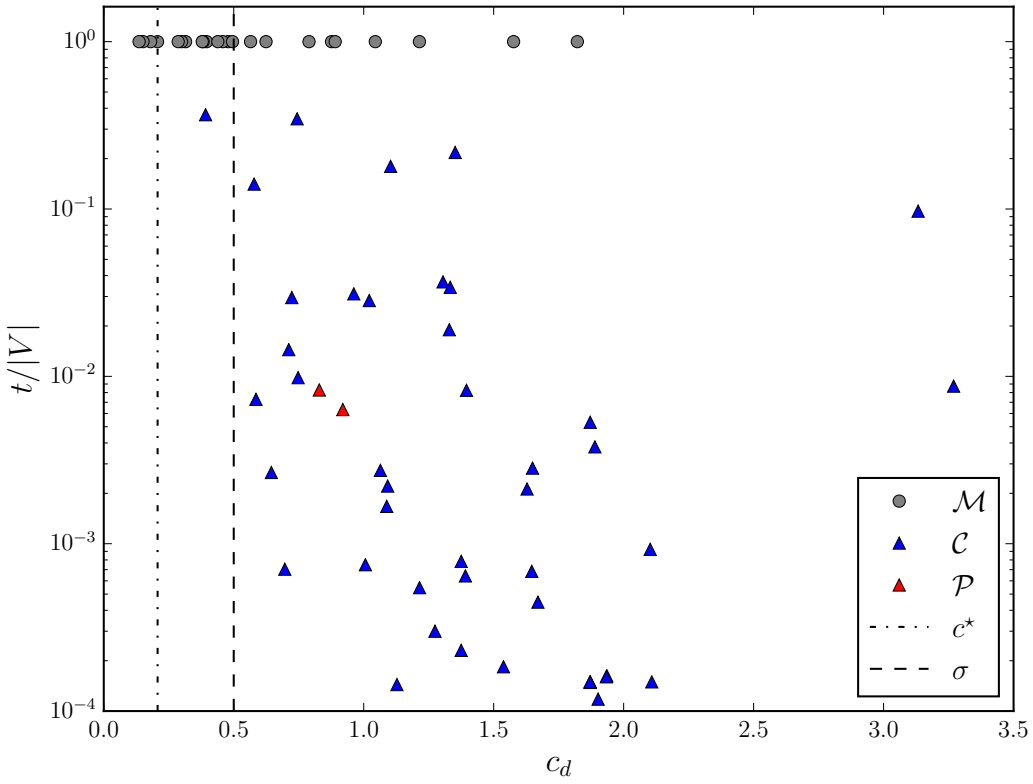


Figure 5.3: Almost-consensus and metastability of the experiments compared to the theoretical and empirical thresholds $c^\star$ and σ. In 81% of the runs there is an almost-consensus if $c_d > \sigma$; the 86% of them are metastable when $c_d < \sigma$. The value t is the arithmetic mean of the number of rounds until almost-consensus/metastability was declared.

work by providing a more fine-grained theoretical analysis which does not assume the adversary's color to be monochromatic and considers more general networks.

We want to highlight that the protocol's convergence to the core's color (as shown in Table 5.1) is remarkable in light of the fact that the *densest-core* method ensures that the sum of the agents' degrees of the core and of the periphery are equal. More precisely, note that equal volumes of core and periphery, starting from an initial configuration where two sets support different colors, is sufficient in the Voter Model to say that the two initial colors have the same probability to be the one eventually supported by all agents [HP01], regardless of the topological structure. Previous works on the 2-Choices dynamics [CER14] provided convergence results which are parametrized only in the difference of the volumes of the two sets, suggesting a similar behavior. Our experimental results highlight the insufficiency of the initial volume distribution as an accurate predictive parameter, showing that the topological structure of the core plays a decisive role.

6

Metastability on Clustered Graphs

In this chapter we consider again the 2-Choices dynamics (Definition 2) and analyze it on graphs exhibiting a clustered structure. It is known in literature that the process rapidly converges to *consensus* [CER+15, CRRS17] (see Chapter 4). Their proofs leverage an interesting property of the 2-Choices dynamics, i.e., that the expected number of agents supporting one state can be expressed as a quadratic form of the transition matrix of a simple random walk on the underlying graph. This fact allows to relate the behavior of the process to the eigenspaces of the graph.

Motivated by questions arising in *graph clustering* and *evolutionary biology*, we exploit the aforementioned relation to show a more fine-grained understanding of the *consensus* behavior of the 2-Choices dynamics. Our new analysis combines symmetry-breaking techniques [BCN+16, CGG+18] and concentration of probability arguments with a linear algebraic approach [CER+15, CRRS17] to obtain the first symmetry-breaking analysis for dynamics on non-complete topologies.

> **Informal description of Theorem 6.2.1.** Let the agents of a network initially pick a random binary state and then run the 2-Choices dynamics. If the network has a *community structure* there is a significant probability that it will rapidly converge to an *almost-clustered* configuration, where almost all nodes within each community share the same state, but the predominant states in the communities are different. In other words, with constant probability, after a short time the states of the nodes constitute a labeling which reveals the clustered structure of the network.

The aforementioned probability for the labeling to reveal the community structure can be amplified via *Community-Sensitive Labeling* [BCN+17a], transforming the 2-Choices dynamics into a *distributed label propagation algorithm* with quasi-linear message complexity.

We remark that, because of the stochastic and time-independent behavior of the 2-Choices dynamics, the process eventually leaves almost-clustered configurations and reaches a *monochromatic* configuration in which all agents have the same state. However, before that happens, we prove that the process remains in almost-clustered configurations for a time equal to a large-degree polynomial in n. Hence, the event that the process leaves the almost-clustered configuration is negligible for most practical applications. This key transitory property of some stochastic processes, called *metastability* [AFPP12, FV15], has recently attracted a lot of attention in the Theoretical Computer Science community.

Comparison with other analytical results

Let a and b respectively be the number of neighbors of each agent in its own community and in the other community; let $d := a + b$. The analysis of Max-LPA essentially requires $a \geqslant n^{3/4-\varepsilon}$ and $b \leqslant ca^2/n$, for some arbitrary constants ε and c. Our analysis requires[1] $\lambda \leqslant n^{-1/4}$, which implies $a \geqslant n^{1/2}$ because of the extremality of Ramanujan graphs, and $b/d \leqslant n^{-1/2}$. Compared to the analysis of Max-LPA, Theorem 6.2.1 holds for much sparser communities at the price of a stricter condition on the cut. Moreover, given the distributed nature of the two algorithms, Max-LPA has a message complexity of $\Omega(m)$, with m the number of edges in the graph that is at least $n^{7/4}$; instead, the message complexity of the 2-Choices dynamics is $\mathcal{O}(n \log n)$ regardless of the actual density of the edges on the graph, since the local update rule only looks at 2 labels. Our algorithm performs an implicit *sparsification* of the graph, an interesting property for the design of sparse clustering algorithms [SZ17], in particular for opportunistic network settings [BCM+18].

6.1 Preliminaries

Let $G = (V, E)$ be a $(2n, d, b)$-*clustered regular graph* (Definition 6.1.1) and let us define $a := d - b$. Note that G is composed by two a-regular communities connected by a b-regular cut (Figure 6.1) and that when $a > b$ the graph G exhibits a well-clustered structure, i.e., each node has more neighbors in its community than in the other one.

Definition 6.1.1 (Clustered regular graph [BCN+17b])**.** *A $(2n, d, b)$-clustered regular graph is a graph $G = (V, E)$ such that:*

- *$V = V_1 \cup V_2$, $V_1 \cap V_2 = \emptyset$, and $|V_1| = |V_2| = n$;*
- *every node has degree d;*
- *every node in V_1 has exactly b neighbors in V_2 and every node in V_2 has exactly b neighbors in V_1.*

[1] λ is the maximum eigenvalue, in absolute value and different from 1, of the transition matrices of the subgraphs induced by the communities.

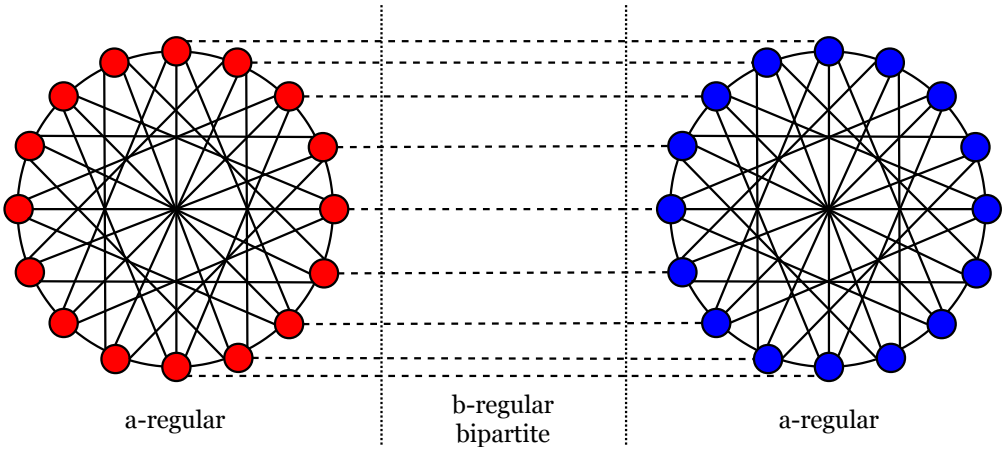


Figure 6.1: Representation of a $(2n, d, b)$-clustered regular graph where $a := d - b$. Each community induces an a-regular graph while the cut between the two communities induces a b-regular bipartite graph.

Each node of G maintains a binary *state* that we represent as a color: either *red* or *blue*. We denote the vector of states of all nodes in G at time t as the *configuration* vector $\boldsymbol{c}^{(t)}$ and we refer to the state of a node $u \in V$ at time t as $\boldsymbol{c}_u^{(t)} \in \{red, blue\}$. We call $B^{(t)}$ the set of nodes colored *blue* at time t and $R^{(t)}$ the set of nodes colored *red* at time t. For each community $i \in \{1, 2\}$ we define $B_i^{(t)} := V_i \cap B^{(t)}$ and $R_i^{(t)} := V_i \cap R^{(t)}$. We call $s_i^{(t)} = |R_i^{(t)}| - |B_i^{(t)}|$ the *bias* in community i toward color *red*. Given some initial configuration $\boldsymbol{c}^{(0)}$, we let the nodes of G run the 2-Choices dynamics.

Note that the random sequence of configurations $\{\boldsymbol{c}^{(t)}\}_{t\in\mathbb{N}}$ generated by multiple iterations of the 2-Choices dynamics on G is a Markov Chain with two absorbing states, namely the configurations where all the nodes support the same color, either *red* or *blue*.

Let us now introduce the notion of *almost-clustered configuration*.

Definition 6.1.2 (Almost-clustered configuration)**.** *A configuration* $\boldsymbol{c}^{(t)}$ *is* almost-clustered *if*

$$|s_i| \geqslant n - \mathcal{O}\left(\frac{\log n}{\log\log n}\right) \tag{6.1}$$

for each $i \in \{1, 2\}$ *and the sign of the biases is different, i.e.,* $s_1 s_2 < 0$.

Intuitively, *almost-clustered* configurations are such that the vast majority of the nodes in one community is supporting one of the two colors, and the vast majority of nodes in the other community is supporting the other color.

In the rest of the section we introduce the notation used to describe the spectral properties of the transition matrix of the underlying graph G: The analysis in expectation of the process (Lemma 6.2.2) exploits such spectral properties and our main result (Theorem 6.2.1) makes assumptions on the spectrum of the transition matrix of G.

Let $P = \frac{1}{d}A$ be the transition matrix of a simple random walk on G, where we denote with d the degree of the nodes and with A the adjacency matrix of G. Note that the transition matrix P can be decomposed as follows:

$$P = \begin{pmatrix} P_{1,1} & P_{1,2} \\ P_{2,1} & P_{2,2} \end{pmatrix} = A + B = \begin{pmatrix} P_{1,1} & 0 \\ 0 & P_{2,2} \end{pmatrix} + \begin{pmatrix} 0 & P_{1,2} \\ P_{2,1} & 0 \end{pmatrix}, \tag{6.2}$$

where A is the transition matrix of the communities if we disconnect them, while B is the transition matrix of the bipartite graph connecting the two communities. Note that since the cut is regular B is symmetric and $P_{1,2}^{\intercal} = P_{2,1}$.

We denote with $\lambda_1 \geqslant \ldots \geqslant \lambda_n$ the eigenvalues of the transition matrix of the subgraph induced by the first community $\bar{P}_{1,1} := \frac{d}{a}P_{1,1}$ and with $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n$ their corresponding eigenvectors; we denote with $\mu_1 \geqslant \ldots \geqslant \mu_n$ the eigenvalues of the transition matrix of the subgraph induced by the second community $\bar{P}_{2,2} := \frac{d}{a}P_{2,2}$ and with $\boldsymbol{w}_1, \ldots, \boldsymbol{w}_n$ their corresponding eigenvectors. Since both $\bar{P}_{1,1}$ and $\bar{P}_{2,2}$ are stochastic matrices we have that $\lambda_1 = \mu_1 = 1$ and that $\boldsymbol{v}_1 = \boldsymbol{w}_1 = \frac{1}{\sqrt{n}}\mathbf{1}$, where $\mathbf{1}$ is the vector of all ones. We consider the case in which both the subgraphs induced by the communities are connected and not bipartite; thus it holds that $\lambda_2 < 1$, $\mu_2 < 1$ and that $\lambda_n > -1$, $\mu_n > -1$.

We define $\lambda := \max(|\lambda_2|, |\lambda_n|, |\mu_2|, |\mu_n|)$. The value of λ is a representative of the second largest eigenvalues for both the subgraphs induced by the communities and is closely related to the third largest eigenvalue of P.

6.2 Theoretical analysis

Let G be a clustered regular graph (Definition 6.1.1). Let each node in G initially pick a color $\boldsymbol{c}_u^{(0)} \in \{red, blue\}$ uniformly at random and independently from the other nodes. Then let the nodes of G run the 2-Choices dynamics (Definition 2).

The variance in the initialization suggests that with some constant probability the distribution of the two colors will be slightly asymmetric w.r.t. the two communities, i.e., the first community will have a bias toward a color, while the second community will have a bias toward the other color. Without loss of generality, we consider the case in which s_1 is positive and s_2 is negative, i.e., the first community is unbalanced toward color *red* while the second community is unbalanced toward color *blue*.

Roughly speaking, we show that when the initialization is "lucky", i.e., the biases in the two communities are toward different colors, there is a significant probability that the process will rapidly make the distribution more and more asymmetric until converging to an *almost-clustered* configuration (Definition 6.1.2), i.e., a configuration in which, apart from a small number of outliers, the nodes in the two communities support different colors. This behavior of the 2-Choices dynamics is formalized in the following theorem.

Theorem 6.2.1 (Constant probability of clustering)**.** *Let $G = (V, E)$ be a connected $(2n, d, b)$-clustered regular graph such that $\frac{b}{d} = \mathcal{O}(n^{-1/2})$ and $\lambda = \mathcal{O}(n^{-1/4})$. Let $c \in \mathbb{N}$ be any constant; let us define the two following events about the 2-Choices dynamics on G:*

- *$\mathcal{C}$: "Starting from a random initialization the process reaches an* almost-clustered *configuration within $\mathcal{O}(\log n)$ rounds."*
- *$\mathcal{M}_c$: "Starting from an* almost-clustered *configuration the process stays in* almost-clustered *configurations for n^c rounds."*

For two suitable positive constants γ_1 and γ_2 it holds that

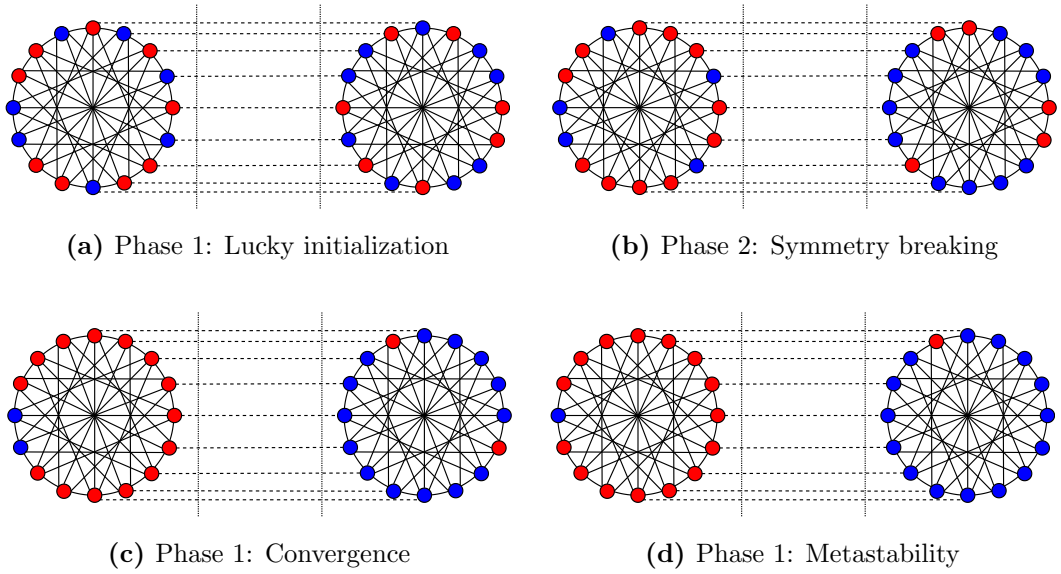
$$\mathbf{P}(\mathcal{C}) \geqslant \gamma_1 \quad \text{and} \quad \mathbf{P}(\mathcal{M}_c) \geqslant 1 - n^{-\gamma_2}. \tag{6.3}$$

Proof. The proof is divided in the following steps (a visual representation is available in Figure 6.2):

1. The bias in each community is initially $|s_i| = \Theta(\sqrt{n})$, for each $i \in \{1, 2\}$, and the sign of the biases is different, with constant probability (Lemma 6.2.1);
2. The bias in each community becomes $|s_i| = \Theta(\sqrt{n}\log n)$, for each $i \in \{1, 2\}$, in $\mathcal{O}(\log\log n)$ rounds and the sign of the biases is preserved, with constant probability (Lemma 6.2.4);
3. The bias in each community becomes $|s_i| \geqslant n - \mathcal{O}(\log n)$, for each $i \in \{1, 2\}$, in $\mathcal{O}(\log n)$ rounds and the sign of the biases is preserved, with high probability (Lemma 6.2.5);
4. The process enters an *almost-clustered* configuration in one single round and lies in the set of *almost-clustered* configurations for the next n^c rounds, with high probability (Lemma 6.2.6).

Before starting with the proof, let us introduce some extra notation. Let $\frac{b}{d} \leqslant c_1 \cdot n^{-1/2}$ for some positive constant c_1, i.e., let every node in each community have at most c_1 neighbors in the opposite community for every $\sqrt{n}$ neighbors in their own. Let $\lambda \leqslant c_2 \cdot n^{-1/4}$, for some positive constant c_2; note that the hypothesis on λ implies that the subgraph induced by each community is a good expander. Let us define the constant $h := 4(2\sqrt{2}c_1 + c_2^2)$.

We start the analysis of the process by looking at the initialization phase. In particular, in Lemma 6.2.1 we show that there is a probability at least constant that the initialization is "lucky", i.e., that the biases in the two communities are $\Theta(\sqrt{n})$ toward different colors. This is true because the Binomial distribution, i.e., the initial distribution of the colors in the graph, is well approximated by a Gaussian distribution, and the latter has a constant probability to deviate from the mean by the standard deviation. The Central Limit Theorem establishes the approximation of the distribution and we are able to quantify it using Berry-Esseen Theorem (Theorem E.3.1).



(a) Phase 1: Lucky initialization

(b) Phase 2: Symmetry breaking

(c) Phase 1: Convergence

(d) Phase 1: Metastability

Figure 6.2: A visualization of the metastable behavior of the 2-Choices dynamics on clustered regular graphs.

Lemma 6.2.1 (Lucky initialization)**.** *Let $G = (V, E)$ be a $(2n, d, b)$-clustered regular graph and let each node $u \in V$ choose a color $\boldsymbol{c}_u^{(0)} \in \{red, blue\}$ uniformly at random and independently from the others. Let c_1 and c_2 be two positive constants. Then, there exists a constant γ_1 such that*

$$\mathbf{P}\left(s_1^{(0)} \geqslant h\sqrt{n} \wedge -s_2^{(0)} \geqslant h\sqrt{n}\right) \geqslant \gamma_1. \tag{6.4}$$

Then, considering a configuration $\boldsymbol{c}^{(t)}$ at a generic time t, we look at the expected evolution of the process observing the behavior of one single community, but also taking into account the influence of the other. Informally, Lemma 6.2.2 gives a bound to the number of nodes that will support the minority color in each community at the next round as a function of all the parameters involved in the process: the number of nodes supporting the minority color in each community at the current round; the number of nodes supporting the same color in the other community at the current round; the expansion of the communities $\lambda \leqslant c_2 \cdot n^{-1/4}$; the cut density $\frac{b}{d} \leqslant c_1 \cdot n^{-1/2}$.

The proof of Lemma 6.2.2 leverages the fact that the expected number of nodes supporting a given color can be expressed as a quadratic form of the transition matrix of a simple random walk on the graph, allowing to relate the behavior of the process to the expansion of the communities, as exploited in [CER+15, CRRS17].

Lemma 6.2.2 (Expected decrease of the minority color)**.** *Let G be a $(2n, d, b)$-clustered regular graph. For any configuration $\boldsymbol{c}^{(t)}$ we have that*

$$\mathbf{E}\left[|B_1^{(t+1)}| \;\middle|\; \boldsymbol{c}^{(t)}\right] <$$

$$|B_1^{(t)}| \left[1 - \frac{s_1}{2n} + \frac{c_2^2}{\sqrt{n}} + \frac{2c_1}{\sqrt{n}} \sqrt{\frac{|B_2^{(t)}|}{|B_1^{(t)}|} \left(\frac{1}{2} - \frac{s_1}{2n} + \frac{c_2^2}{\sqrt{n}} + \frac{c_1^2 |B_2^{(t)}|}{n|B_1^{(t)}|} \right)} \right] \tag{6.5}$$

and

$$\mathbf{E}\left[|R_2^{(t+1)}| \;\middle|\; \boldsymbol{c}^{(t)} \right] < |R_2^{(t)}| \left[1 + \frac{s_2}{2n} + \frac{c_2^2}{\sqrt{n}} + \frac{2c_1}{\sqrt{n}} \sqrt{\frac{|R_1^{(t)}|}{|R_2^{(t)}|} \left(\frac{1}{2} + \frac{s_2}{2n} + \frac{c_2^2}{\sqrt{n}} + \frac{c_1^2 |R_1^{(t)}|}{n|R_2^{(t)}|} \right)} \right]. \tag{6.6}$$

It follows from Lemma 6.2.2 that the asymmetry in the coloring of the nodes in the two communities continues to grow in expectation. In fact, when in a certain range of values, the bias in the first community increases in expectation at each round while the bias in the second community decreases in expectation at each round, since the minority color in each community decreases. With Lemma 6.2.3 we prove that the increase of the bias in the first community and the decrease of the bias in the second community we have shown in expectation in Lemma 6.2.2 is multiplicative w.h.p. whenever s_1 satisfies $s_1 \in [h\sqrt{n}, \frac{n}{2}]$ and s_2 satisfies $s_2 \in [-\frac{n}{2}, -h\sqrt{n}]$. With the use of concentration of probability arguments, namely a multiplicative form of the Chernoff bounds [DP09, Lemma 1.1], we show that the number of nodes with the minority color in each community decreases and we use this fact to prove Lemma 6.2.3.

Lemma 6.2.3 (Probability of multiplicative growth of the bias)**.** *Let $\boldsymbol{c}^{(t)}$ be a configuration such that $h\sqrt{n} \leqslant s_1 \leqslant \frac{n}{2}$ and $h\sqrt{n} \leqslant -s_2 \leqslant \frac{n}{2}$. Then, it holds that*

$$\mathbf{P}\left(s_1^{(t+1)} \geqslant (1 + 1/16)\, s_1 \;\middle|\; \boldsymbol{c}^{(t)} \right) \geqslant 1 - e^{-2s_1^2/32^2 n} \tag{6.7}$$

and

$$\mathbf{P}\left(s_2^{(t+1)} \leqslant (1 + 1/16)\, s_2 \;\middle|\; \boldsymbol{c}^{(t)} \right) \geqslant 1 - e^{-2s_2^2/32^2 n}. \tag{6.8}$$

Now we know that there is a constant probability that the initialization of the process starts is "lucky" (Lemma 6.2.1); we also know that the bias in the first community will increase in expectation and the bias in the second community will decrease in expectation (Lemma 6.2.2); moreover, when in a given range, we know that the biases will follow their expected behavior with high probability (Lemma 6.2.3).

Then we need to show that the asymmetry in the coloring of the two communities will rapidly increase up to a configuration such that $|s_i| = \Theta(\sqrt{n} \log n)$, for each $i \in \{1, 2\}$, while the sign of the biases is preserved. More formally, with Lemma 6.2.4 we prove the internal symmetry breaking of each community. This is possible by applying Lemma 6.2.1, and by iterating the application of Lemma 6.2.3 for $\mathcal{O}(\log \log n)$ rounds, i.e., until the bias is large enough; finally we handle the stochastic dependency between the two biases during their respective increases in opposite directions.

Lemma 6.2.4 (Clustering – Symmetry Breaking)**.** *Starting from an initial configuration where each node $u \in V$ chooses a color $\boldsymbol{c}_u^{(0)} \in \{red, blue\}$ uniformly at random and independently from the others, it holds that, with constant probability, within $\mathcal{O}(\log\log n)$ rounds the process reaches a configuration $\boldsymbol{c}^{(t)}$ such that*

$$s_1^{(t)} \geqslant \sqrt{n}\log n \quad \textit{and} \quad -s_2^{(t)} \geqslant \sqrt{n}\log n. \tag{6.9}$$

Once the internal symmetry of each community is broken, we show that, with high probability, both biases keep increasing while preserving their sign until they rapidly reach a configuration in which the minority color in each community has at most logarithmic size. This behavior is formally proved in Lemma 6.2.5, again through the application of Lemma 6.2.2 and Lemma 6.2.3.

Lemma 6.2.5 (Convergence)**.** *Starting from a configuration $\boldsymbol{c}^{(t)}$ such that $|s_i| \geqslant \sqrt{n}\log n$, for each $i \in \{1,2\}$, there exist two rounds $\tau_1, \tau_2 = \mathcal{O}(\log n)$ such that*

$$|s_1^{(\tau_1)}| \geqslant n - \log n \quad \textit{and} \quad |s_2^{(\tau_2)}| \geqslant n - \log n \tag{6.10}$$

and the sign of the biases is preserved, with high probability.

Finally, with Lemma 6.2.6 we show that the number of wrongly colored nodes in each community drops to $\mathcal{O}(\log n/\log\log n)$ in one single round (by approximating it with a Poisson random variable through an application of Le Cam's Theorem) and then, with high probability, the process enters a *metastable* phase in which the only possible configurations are *almost-clustered*; this will last for any polynomial number of rounds. In other words, even if a few nodes in each community will continue to change color, almost all the nodes in one community will support one color while almost all the nodes in the other community will support the other color. Note that this quantity is *tight*: It is possible to prove that, within any polynomial number of rounds, there will be a round in which at least $\Omega(\log n/\log\log n)$ nodes in each community will have the wrong color.

Lemma 6.2.6 (Metastability)**.** *Let $c \in \mathbb{N}$ be any constant. Starting from a configuration $\boldsymbol{c}^{(t)}$ such that $|s_i| \geqslant n - \log n$ for each $i \in \{1,2\}$, for the next n^c rounds the process lies in the set of configurations such that*

$$|s_i| \geqslant n - \mathcal{O}\left(\frac{\log n}{\log\log n}\right) \tag{6.11}$$

and the sign of the bias is preserved, with high probability.

More formally, through Lemma 6.2.5 and Lemma 6.2.6 we can finally prove that $\mathbf{P}(\mathcal{C}) \geqslant \gamma_1$ and $\mathbf{P}(\mathcal{M}_c) \geqslant 1 - n^{-\gamma_2}$ for any constant c, concluding the proof of Theorem 6.2.1. ■

Technical proofs

Proof of Lemma 6.2.1

Proof. The initial bias of the first cluster $s_1 = |R_1| - |B_1|$ can be thought as a sum of Rademacher random variables, i.e. $s_1 = \sum_{i \in V_1} X_i$ where $X_i = 1$ if node i chooses color *red* and $X_i = -1$ if node i chooses color *blue*. Rademacher random variables have mean equal to 0, variance equal to 1, and third moment equal to 1; thus, we can apply the Berry-Esseen theorem (Theorem E.3.1) which in our case states that

$$\left| \mathbf{P}\left(\frac{\sum_{i \in V_1} X_i}{\sqrt{n}} \leqslant h \right) - \Phi(h) \right| \leqslant \frac{C}{\sqrt{n}}, \tag{6.12}$$

where Φ is the cumulative distribution function of the standard normal distribution and C is a universal positive constant. Hence,

$$\mathbf{P}\left(\sum_{i \in V_1} X_i < 4(2\sqrt{2}c_1 + c_2^2)\sqrt{n} \right) \leqslant \Phi(4(2\sqrt{2}c_1 + c_2^2)) + \frac{C}{\sqrt{n}} \tag{6.13}$$

$$\overset{(a)}{\leqslant} \Phi(4(2\sqrt{2}c_1 + c_2^2)) + \varepsilon \tag{6.14}$$

$$\overset{(b)}{\leqslant} \alpha, \tag{6.15}$$

where in (a) ε is a suitably small positive constant and (b) holds for a positive constant α strictly smaller than one, because for every h constant also $\Phi(h)$ is a constant strictly smaller than one. The same inequality also holds for s_2. Note that, since the random variables s_1 and s_2 are independent, we have:

$$\mathbf{P}\left(s_1 \geqslant h\sqrt{n}\right) \cdot \mathbf{P}\left(s_2 \geqslant h\sqrt{n}\right) \geqslant (1-\alpha)^2 = \gamma_1. \tag{6.16}$$

■

Proof of Lemma 6.2.2

Proof. W.l.o.g. we analyze the case of the *blue* minority color in community 1. The proof is completely symmetric for the *red* minority color in community 2.

For every set $Z \in \{B, B_1, B_2, R, R_1, R_2\}$ and for every node $v \in V$, we define $Z(v) = N(v) \cap Z$, where $N(v)$ is the set of neighbors of v. Thus, by definition of 2-Choices dynamics (Definition 2), we can write the expected number of nodes supporting the minority color in community 1 at round $t+1$ as the sum of the probabilities for each node supporting color *red* of picking two *blue* nodes (and thus becoming *blue*) and the sum of the probabilities for each *blue* node of not picking two *red* nodes (and thus remaining *blue*).

$$\mathbf{E}\left[|B_1^{(t+1)|} \;\middle|\; \boldsymbol{c}^{(t)} \right] = \sum_{x \in R_1} \left(\frac{|B(x)|}{d} \right)^2 + \sum_{x \in B_1} \left(1 - \left(\frac{|R(x)|}{d} \right)^2 \right) \tag{6.17}$$

$$= \sum_{x\in V_1}\left(\frac{|B(x)|}{d}\right)^2 - \sum_{x\in B_1}\left(\frac{|B(x)|}{d}\right)^2 + \sum_{x\in B_1}\left(1-\left(1-\frac{|B(x)|}{d}\right)^2\right) \tag{6.18}$$

$$= \sum_{x\in V_1}\left(\frac{|B(x)|}{d}\right)^2 - \sum_{x\in B_1}\left(\frac{|B(x)|}{d}\right)^2 + $$

$$+ \sum_{x\in B_1}\left(1-1+2\frac{|B(x)|}{d}-\left(\frac{|B(x)|}{d}\right)^2\right) \tag{6.19}$$

$$= \sum_{x\in V_1}\left(\frac{|B(x)|}{d}\right)^2 + 2\sum_{x\in B_1}\left(\frac{|B(x)|}{d}-\left(\frac{|B(x)|}{d}\right)^2\right) \tag{6.20}$$

$$= \sum_{x\in V_1}\left(\frac{|B(x)|}{d}\right)^2 + 2\sum_{x\in B_1}\left(\frac{|B(x)|}{d}\left(1-\frac{|B(x)|}{d}\right)\right) \tag{6.21}$$

$$\leqslant \sum_{x\in V_1}\left(\frac{|B(x)|}{d}\right)^2 + \left(\frac{B_1}{2}\right), \tag{6.22}$$

where in the last inequality we used the fact that $\frac{|B(x)|}{d}\left(1-\frac{|B(x)|}{d}\right) \leqslant \frac{1}{4}$, since it is a concave function and its maximum is $\frac{1}{4}$.

In order to bound the quantity $\sum_{x\in V_1}\left(\frac{|B(x)|}{d}\right)^2$ we use the assumptions on the structure of G, i.e. that it is $(2n,d,b)$-clustered, that $\frac{b}{d} \leqslant c_1\cdot n^{-1/2}$, and that $\lambda \leqslant c_2\cdot n^{-1/4}$. In particular, we split the quantity into three terms as follows:

$$\sum_{x\in V_1}\left(\frac{|B(x)|}{d}\right)^2 = \sum_{x\in V_1}\left(\frac{|B_1(x)|}{d}+\frac{|B_2(x)|}{d}\right)^2 \tag{6.23}$$

$$= \sum_{x\in V_1}\left(\frac{|B_1(x)|}{d}\right)^2 + \sum_{x\in V_1}\left(\frac{|B_2(x)|}{d}\right)^2 + $$

$$+ 2\sum_{x\in V_1}\frac{|B_1(x)|}{d}\cdot\frac{|B_2(x)|}{d}. \tag{6.24}$$

We upper bound the first of the terms by using $\lambda := \max(|\lambda_2|,|\lambda_n|,|\mu_2|,|\mu_n|)$, which gives a measure of the internal expansion of the graphs induced by the clusters. Let $\bar{P}_{1,1} := \frac{d}{a}P_{1,1}$ be the transition matrix of the subgraph induced by the first cluster. Notice that, since G is $(2n,d,b)$-clustered, the subgraph induced by the first cluster is a-regular and thus $\bar{P}_{1,1}$ is symmetric. Consequently the eigenvectors of $\bar{P}_{1,1}$ form an orthonormal basis of the space. Let $\mathbf{1}_{\boldsymbol{B}}^{(t)}$ be the indicator vector of the set B, i.e. $\mathbf{1}_{\boldsymbol{B}}^{(t)}(v) = 1$ if $v \in B^{(t)}$ and $\mathbf{1}_{\boldsymbol{B}}^{(t)}(v) = 0$ otherwise. When clear from the context we will omit the time t. This allows us to write the matrix in its spectral decomposition, i.e. $\bar{P}_{1,1} = \sum_{i=1}^{n}\lambda_i \boldsymbol{v}_i\boldsymbol{v}_i^{\intercal}$, and the indicator vector of the blue nodes in the first cluster as a linear combination of the eigenvectors of $\bar{P}_{1,1}$, i.e. $\mathbf{1}_{B_1} = \sum_{i=1}^{n}\alpha_i\boldsymbol{v}_i$ with $\alpha_i = \langle \boldsymbol{v}_i, \mathbf{1}_{B_1}\rangle$. Hence, we get that

$$\sum_{x\in V_1}\left(\frac{|B_1(x)|}{d}\right)^2 = \|P_{1,1}\mathbf{1}_{\boldsymbol{B}1}\|_2^2 \tag{6.25}$$

$$= \mathbf{1}_{\boldsymbol{B}1}^{\intercal}P_{1,1}^{\intercal}\cdot P_{1,1}\mathbf{1}_{\boldsymbol{B}1} \tag{6.26}$$

$$
\begin{aligned}
&= \mathbf{1}_{\boldsymbol{B}_1}^{\intercal} P_{1,1}^2 \mathbf{1}_{\boldsymbol{B}1} && (6.27)\\
&= \frac{a^2}{d^2} \mathbf{1}_{\boldsymbol{B}_1}^{\intercal} \bar{P}_{1,1}^2 \mathbf{1}_{\boldsymbol{B}1} && (6.28)\\
&\leqslant \mathbf{1}_{\boldsymbol{B}_1}^{\intercal} \bar{P}_{1,1}^2 \mathbf{1}_{\boldsymbol{B}1} && (6.29)\\
&= \mathbf{1}_{\boldsymbol{B}_1}^{\intercal} \cdot \sum_{i=1}^{n} \lambda_i^2 \boldsymbol{v}_i \boldsymbol{v}_i^{\intercal} \cdot \sum_{i=1}^{n} \alpha_i \boldsymbol{v}_i && (6.30)\\
&= \mathbf{1}_{\boldsymbol{B}_1}^{\intercal} \cdot \sum_{i=1}^{n} \lambda_i^2 \alpha_i \boldsymbol{v}_i && (6.31)\\
&= \mathbf{1}_{\boldsymbol{B}_1}^{\intercal} \cdot \left(\lambda_1^2 \alpha_1 \boldsymbol{v}_1 + \sum_{i=2}^{n} \lambda_i^2 \alpha_i \boldsymbol{v}_i \right) && (6.32)\\
&\leqslant \mathbf{1}_{\boldsymbol{B}_1}^{\intercal} \cdot \left(\lambda_1^2 \alpha_1 \boldsymbol{v}_1 + \lambda^2 \sum_{i=2}^{n} \alpha_i \boldsymbol{v}_i \right) && (6.33)\\
&\leqslant \mathbf{1}_{\boldsymbol{B}_1}^{\intercal} \cdot \left(\lambda_1^2 \alpha_1 \boldsymbol{v}_1 + \lambda^2 \sum_{i=1}^{n} \alpha_i \boldsymbol{v}_i \right) && (6.34)\\
&= \mathbf{1}_{\boldsymbol{B}_1}^{\intercal} \cdot \left(\alpha_1 \boldsymbol{v}_1 + \lambda^2 \mathbf{1}_{\boldsymbol{B}1} \right) && (6.35)\\
&= \frac{|B_1|^2}{n} + \lambda^2 |B_1|. && (6.36)
\end{aligned}
$$

The second of the terms can be bounded using the Cauchy-Schwarz inequality and the fact that the fraction of neighbors in the other community is $\frac{b}{d}$. Formally, we get

$$
\begin{aligned}
\sum_{x \in V_1} \left(\frac{|B_2(x)|}{d} \right)^2 &\leqslant \left(\| P_{1,2} \mathbf{1}_{\boldsymbol{B}2} \|_2 \right)^2 && (6.37)\\
&\leqslant \left(\| P_{1,2} \|_2 \| \mathbf{1}_{\boldsymbol{B}2} \|_2 \right)^2 && (6.38)\\
&= \left(\| P_{1,2} \|_2 \sqrt{|B_2|} \right)^2 && (6.39)\\
&\leqslant \left(\sqrt{\| P_{1,2} \|_1 \cdot \| P_{1,2} \|_\infty} \cdot \sqrt{|B_2|} \right)^2 && (6.40)\\
&= \left(\frac{b}{d} \sqrt{|B_2|} \right)^2 && (6.41)\\
&= \frac{b^2}{d^2} |B_2|, && (6.42)
\end{aligned}
$$

where in the last inequality we combined Corollary C.1.1 with the two following observations:

- $\| P_{1,2} \|_1 := \max_{1 \leqslant j \leqslant n} \sum_{i=1}^{n} |b_{ij}| = \frac{b}{d}$, since each node in the first community has exactly b neighbors in the second community.
- $\| P_{1,2} \|_\infty := \max_{1 \leqslant i \leqslant n} \sum_{j=1}^{m} |b_{ij}| = \frac{b}{d}$, since each node in the second community has exactly b neighbors in the first community and $P_{1,2} = P_{2,1}^{\intercal}$.

For the third and last term, which equals twice the product of the first two, we use the previously derived bounds and get the following quantity:

$$2\sum_{x\in V_1}\frac{|B_1(x)|}{d}\cdot\frac{|B_2(x)|}{d} = 2\|P_{1,1}b_1\|_2\cdot\|P_{1,2}b_2\|_2 \tag{6.43}$$

$$\leqslant 2\frac{b}{d}\sqrt{|B_2|\left(\frac{|B_1|^2}{n}+\lambda^2|B_1|\right)}. \tag{6.44}$$

Before combining the three bounds, we recall that by hypothesis G is such that $\frac{b}{d}\leqslant c_1\cdot n^{-1/2}$ and $\lambda\leqslant c_2\cdot n^{-1/4}$. Hence:

$$\mathbf{E}\left[|B_1^{(t+1)|}\;\middle|\;\boldsymbol{c}^{(t)}\right]$$

$$\leqslant \frac{|B_1|^2}{n}+\lambda^2|B_1|+\frac{b^2}{d^2}|B_2|+2\frac{b}{d}\sqrt{|B_2|\left(\frac{|B_1|^2}{n}+\lambda^2|B_1|\right)}+\left(\frac{|B_1|}{2}\right) \tag{6.45}$$

$$= \frac{|B_1|^2}{n}+\lambda^2|B_1|+\frac{b^2}{d^2}|B_2|+2\frac{b}{d}\sqrt{|B_1|\cdot|B_2|\left(\frac{|B_1|}{n}+\lambda^2\right)}+\left(\frac{|B_1|}{2}\right) \tag{6.46}$$

$$\leqslant \frac{|B_1|^2}{n}+c_2^2\frac{|B_1|}{\sqrt{n}}+c_1^2\frac{|B_2|}{n}+\frac{2c_1}{\sqrt{n}}\sqrt{|B_1|\cdot|B_2|\left(\frac{|B_1|}{n}+\frac{c_2^2}{\sqrt{n}}\right)}+\left(\frac{|B_1|}{2}\right) \tag{6.47}$$

$$= |B_1|\left(\frac{|B_1|}{n}+\frac{c_2^2}{\sqrt{n}}+\frac{c_1^2|B_2|}{n|B_1|}+\frac{2c_1}{\sqrt{n}}\sqrt{\frac{|B_2|}{|B_1|}\left(\frac{|B_1|}{n}+\frac{c_2^2}{\sqrt{n}}\right)}+\frac{1}{2}\right) \tag{6.48}$$

$$< |B_1|\left(\frac{1}{2}-\frac{s_1}{2n}+\frac{c_2^2}{\sqrt{n}}+\frac{c_1^2|B_2|}{n|B_1|}+\frac{2c_1}{\sqrt{n}}\sqrt{\frac{|B_2|}{|B_1|}\left(\frac{1}{2}-\frac{s_1}{2n}+\frac{c_2^2}{\sqrt{n}}\right)}+\frac{1}{2}\right) \tag{6.49}$$

$$< |B_1|\left(\frac{1}{2}-\frac{s_1}{2n}+\frac{c_2^2}{\sqrt{n}}+\frac{2c_1}{\sqrt{n}}\sqrt{\frac{|B_2|}{|B_1|}\left(\frac{1}{2}-\frac{s_1}{2n}+\frac{c_2^2}{\sqrt{n}}+\frac{c_1^2|B_2|}{n|B_1|}\right)}+\frac{1}{2}\right) \tag{6.50}$$

$$< |B_1|\left(1-\frac{s_1}{2n}+\frac{c_2^2}{\sqrt{n}}+\frac{2c_1}{\sqrt{n}}\sqrt{\frac{|B_2|}{|B_1|}\left(\frac{1}{2}-\frac{s_1}{2n}+\frac{c_2^2}{\sqrt{n}}+\frac{c_1^2|B_2|}{n|B_1|}\right)}\right). \tag{6.51}$$

Thus, we finally get

$$\mathbf{E}\left[|B_1^{(t+1)}|\;\middle|\;\boldsymbol{c}^{(t)}\right]$$

$$< |B_1|\left[1-\frac{s_1}{2n}+\frac{c_2^2}{\sqrt{n}}+\frac{2c_1}{\sqrt{n}}\sqrt{\frac{|B_2|}{|B_1|}\left(\frac{1}{2}-\frac{s_1}{2n}+\frac{c_2^2}{\sqrt{n}}+\frac{c_1^2|B_2|}{n|B_1|}\right)}\right] \tag{6.52}$$

and, having a symmetric scenario in the other community, also that

$$\mathbf{E}\left[|R_2^{(t+1)}|\;\middle|\;\boldsymbol{c}^{(t)}\right]$$

$$< |R_2|\left[1+\frac{s_2}{2n}+\frac{c_2^2}{\sqrt{n}}+\frac{2c_1}{\sqrt{n}}\sqrt{\frac{|R_1|}{|R_2|}\left(\frac{1}{2}+\frac{s_2}{2n}+\frac{c_2^2}{\sqrt{n}}+\frac{c_1^2|R_1|}{n|R_2|}\right)}\right]. \tag{6.53}$$

■

Proof of Lemma 6.2.3

Proof. Let us start with the bias in the first community. Note that our assumption on the bias implies that $\frac{n}{4} \leqslant |B_1| \leqslant \frac{n-s_1}{2} < \frac{n}{2}$, and thus $\frac{|B_2|}{|B_1|} \leqslant 4$. Therefore, under these conditions the expectation of $|B_1^{(t+1)}|$ can be upper bounded as follows:

$$\begin{aligned}
&\mathbf{E}\left[|B_1^{(t+1)|} \;\middle|\; \boldsymbol{c}^{(t)}\right] \\
&< |B_1|\left(1 - \frac{s_1}{2n} + \frac{c_2^2}{\sqrt{n}} + \frac{2c_1}{\sqrt{n}}\sqrt{\frac{|B_2|}{|B_1|}\left(\frac{1}{2} - \frac{s_1}{2n} + \frac{c_2^2}{\sqrt{n}} + \frac{c_1^2|B_2|}{n|B_1|}\right)}\right) && (6.54)\\
&< |B_1|\left(1 - \frac{s_1}{2n} + \frac{c_2^2}{\sqrt{n}} + \frac{2c_1}{\sqrt{n}}\sqrt{4\left(\frac{1}{2} - \frac{s_1}{2n} + \frac{c_2^2}{\sqrt{n}} + \frac{4c_1^2}{n}\right)}\right) && (6.55)\\
&\overset{(a)}{<} |B_1|\left(1 - \frac{s_1}{2n} + \frac{c_2^2}{\sqrt{n}} + \frac{2c_1}{\sqrt{n}}\sqrt{4 \cdot \frac{1}{2}}\right) && (6.56)\\
&< |B_1|\left(1 - \frac{s_1}{2n} + \frac{c_2^2}{\sqrt{n}} + \frac{2c_1\sqrt{2}}{\sqrt{n}}\right) && (6.57)\\
&= |B_1|\left(1 - \frac{s_1}{2n} + \frac{2c_1\sqrt{2} + c_2^2}{\sqrt{n}}\right) && (6.58)\\
&= |B_1|\left(1 - \frac{s_1}{2n} + \frac{s_1}{4n}\right) && (6.59)\\
&= |B_1|\left(1 - \frac{s_1}{4n}\right), && (6.60)
\end{aligned}$$

where in (a) we used that $\frac{s_1}{2n} \geqslant \frac{c_2^2}{\sqrt{n}} + \frac{4c_1^2}{n}$ by hypothesis.

Using the additive form of the Chernoff Bound and that $s_1 \leqslant \frac{n}{2}$, we get that

$$\begin{aligned}
&\mathbf{P}\left(|B_1|^{(t+1)} > |B_1|\left(1 - \frac{s_1}{8n}\right) \;\middle|\; \boldsymbol{c}^{(t)}\right) \\
&= \mathbf{P}\left(|B_1|^{(t+1)} > |B_1|\left(1 - \frac{s_1}{4n}\right) + \frac{s_1|B_1|}{8n} \;\middle|\; \boldsymbol{c}^{(t)}\right) && (6.61)\\
&\leqslant \mathbf{P}\left(|B_1|^{(t+1)} > |B_1|\left(1 - \frac{s_1}{4n}\right) + \frac{s_1}{32} \;\middle|\; \boldsymbol{c}^{(t)}\right) && (6.62)\\
&\leqslant \mathbf{P}\left(|B_1|^{(t+1)} > \mathbf{E}\left[|B_1|^{(t+1)} \;\middle|\; \boldsymbol{c}^{(t)}\right] + \frac{s_1}{32} \;\middle|\; \boldsymbol{c}^{(t)}\right) && (6.63)\\
&\leqslant \exp\left(-2(s_1)^2/(32^2 n)\right). && (6.64)
\end{aligned}$$

Since it holds that $s_1 = n - 2|B_1|$, then with probability $1 - \exp\left(-2(s_1)^2/(32^2 n)\right)$ it holds that

$$\begin{aligned}
s_1^{(t+1)} &\geqslant n - 2|B_1|\left(1 - \frac{s_1}{8n}\right) && (6.65)\\
&= n - (n - s_1)\left(1 - \frac{s_1}{8n}\right) && (6.66)\\
&= n - n\left(1 - \frac{s_1}{8n}\right) + s_1\left(1 - \frac{s_1}{8n}\right) && (6.67)\\
&= \frac{s_1}{8} + s_1 - \frac{s_1^2}{8n} && (6.68)\\
&\geqslant \frac{s_1}{8} + s_1 - \frac{s_1}{16} && (6.69)
\end{aligned}$$

$$= s_1 \left(1 + \frac{1}{16}\right). \tag{6.70}$$

The same reasoning can also be applied to the symmetric case of s_2. ■

Proof of Lemma 6.2.4

Proof. Let $\mathcal{I}$ be the event "the initial configuration has the property that $s_1^{(0)} \geqslant 4(2\sqrt{2}c_1 + c_2^2)\sqrt{n}$ and $-s_2^{(0)} \geqslant 4(2\sqrt{2}c_1 + c_2^2)\sqrt{n}$." In Lemma 6.2.1 we proved that $\mathcal{I}$ happens with a probability that is at least constant. Then, starting from such configuration, we use Lemma 6.2.3 in order to show that s_1 becomes greater than $\sqrt{n}\log n$ and s_2 becomes smaller than $-\sqrt{n}\log n$ within $\mathcal{O}(\log\log n)$ rounds, with constant probability.

We define a round t to be *successful* w.r.t. community 1 if one of the two following conditions hold:

- the process has not reached yet a configuration in which the bias is is multiplicatively increasing and is large enough, namely $s_1^{(t)} \geqslant s_1^{(t-1)}\left(1 + \frac{1}{16}\right)$ and $s_1^{(t-1)} < \sqrt{n}\log n$;
- the bias was already large enough in a previous round, i.e., there exists a round $t' < t$ such that $s_1^{(t')} \geqslant \sqrt{n}\log n$.

The definition extends to community 2 in a symmetric fashion.

Let $h = 4(2\sqrt{2}c_1 + c_2^2)$, $\alpha = 2(h/32)^2$, $\beta = (1 + 1/16)$, and let us define the events

$\mathcal{S}_i^{(t)}$: "The round t is successful w.r.t. community i."

$\mathcal{K}_i$: "The first $\log_\beta \log n$ rounds are successful w.r.t. community i."

Note that, after T consecutive *successful* rounds with respect to community 1, the stochastic process reaches a configuration $\bar{c}$ such that $s_1 \geqslant h\sqrt{n}(1 + 1/16)^T$ and then the probability that also the next round is successful with respect to community 1 is at least $1 - \exp\left\{-\frac{2h^2(1+1/16)^{2T}}{32^2}\right\}$. Conditioning to event $\mathcal{I}$, we have that

$$\mathbf{P}\left(\mathcal{K}_1 \mid \mathcal{I}\right) = \mathbf{P}\left(\bigwedge_{i=1}^{\log_\beta \log n} \mathcal{S}_1^{(i)} \,\middle|\, \mathcal{I}\right) \tag{6.71}$$

$$= \prod_{i=1}^{\log_\beta \log n} \mathbf{P}\left(\mathcal{S}_1^{(i)} \,\middle|\, \bar{c}^{(i-1)} : \bigwedge_{j=1}^{i-1} \mathcal{S}_1^{(j)}, \mathcal{I}\right) \tag{6.72}$$

$$\geqslant \prod_{i=1}^{\log_\beta \log n} \left(1 - e^{-2h^2(1+1/16)^{2i}/(32)^2}\right) \tag{6.73}$$

$$= \prod_{i=1}^{\log_\beta \log n} \left(1 - e^{-\alpha\beta^{2i}}\right) \tag{6.74}$$

$$= \exp\left(\log\left(\prod_{i=1}^{\log_\beta \log n} (1 - e^{-\alpha\beta^{2i}})\right)\right) \tag{6.75}$$

$$= \exp\left(\sum_{i=1}^{\log_\beta \log n} \log(1 - e^{-\alpha\beta^{2i}})\right) \tag{6.76}$$

$$> \exp\left(\sum_{i=1}^{\infty} \log(1 - e^{-\alpha\beta^{2i}})\right) \tag{6.77}$$

$$\overset{(a)}{=} \exp\left(-\sum_{i=1}^{\infty} \left[e^{-\alpha\beta^{2i}} + \mathcal{O}(e^{-2\alpha\beta^{2i}})\right]\right) \tag{6.78}$$

$$\overset{(b)}{>} \exp\left(-\frac{1}{\alpha}\sum_{i=1}^{\infty} \left[\beta^{-2i} + \mathcal{O}(\beta^{-2i})\right]\right) \tag{6.79}$$

$$= \exp\left(-\frac{1}{\alpha}\left(\frac{1}{1-\beta^{-2}}\right)(1+C)\right) \tag{6.80}$$

$$= e^{-\beta'}, \tag{6.81}$$

where in (a) we expanded $\log(1-x) = -x - \frac{x^2}{2} - \frac{x^3}{3} - \dots$ using the Taylor series, and in (b) we used that

$$e^{-\alpha\beta^{2i}} < \alpha^{-1}\beta^{-2i} \implies -\alpha\beta^{2i} < \log(\alpha^{-1}\beta^{-2i}) \implies \alpha\beta^{2i} > \log(\alpha\beta^{2i}), \tag{6.82}$$

which is always true for our values of α, β, i since we always have $\alpha\beta^{2i} > 0$; the term C appearing in the last bound is a constant due to the smaller order terms coming from the Taylor approximation.

Note that the bias, starting from $\mathcal{O}(\sqrt{n})$, reaches a value of $\mathcal{O}(\sqrt{n}\log n)$ after $\mathcal{O}(\log\log n)$ rounds. This implies that the bias reaches a value of at least $\sqrt{n}\log n$ within $\mathcal{O}(\log\log n)$ rounds with probability at least $e^{\beta'}$.

In a completely symmetric fashion, the same holds for community 2. Now we compute the probability $\mathbf{P}(\mathcal{K}_1, \mathcal{K}_2 \mid \mathcal{I})$ using the fact that, conditioning on the previous configuration, the events that a round is *successful* w.r.t. community 1 and community 2 are independent.

$$\mathbf{P}(\mathcal{K}_1, \mathcal{K}_2 \mid \mathcal{I}) > \mathbf{P}\left(\bigwedge_{i=1}^{\log_\beta \log n} \mathcal{S}_1^{(i)} \wedge \mathcal{S}_2^{(i)} \;\middle|\; \mathcal{I}\right) \tag{6.83}$$

$$= \prod_{i=1}^{\log_\beta \log n} \mathbf{P}\left(\mathcal{S}_1^{(i)} \wedge \mathcal{S}_2(i) \;\middle|\; \bar{c}^{(i-1)} : \cap_{j=1}^{i-1} \mathcal{S}_1^{(j)} \wedge \mathcal{S}_2^{(j)}, \mathcal{I}\right) \tag{6.84}$$

$$= \prod_{i=1}^{\log_\beta \log n} \mathbf{P}\left(\mathcal{S}_1^{(i)} \;\middle|\; \bar{c}^{(i-1)} : \cap_{j=1}^{i-1} \mathcal{S}_1^{(j)} \wedge \mathcal{S}_2^{(j)}, \mathcal{I}\right)$$
$$\times \prod_{i=1}^{\log_\beta \log n} \mathbf{P}\left(\mathcal{S}_2(i) \;\middle|\; \bar{c}^{(i-1)} : \cap_{j=1}^{i-1} \mathcal{S}_1^{(j)} \wedge \mathcal{S}_2^{(j)}, \mathcal{I}\right) \tag{6.85}$$

$$= \mathbf{P}\left(\bigwedge_{i=1}^{\log_\beta \log n} \mathcal{S}_1^{(i)} \;\middle|\; \mathcal{I}\right) \cdot \mathbf{P}\left(\bigwedge_{i=1}^{\log_\beta \log n} \mathcal{S}_2^{(i)} \;\middle|\; \mathcal{I}\right) \tag{6.86}$$

$$\geqslant e^{-2\beta'} \tag{6.87}$$

$$= \gamma_3. \tag{6.88}$$

■

Proof of Lemma 6.2.5

Proof. The proof has the following structure: We focus only on one of the biases and we show, in two different phases, that it will grow until it reaches the value $n - \log n$ within $\mathcal{O}(\log n)$ rounds, w.h.p. Then we apply the Union Bound and we show that this holds for both the biases, w.h.p.

W.l.o.g. we assume that both the biases are positive. For any $i \in \{1, 2\}$ we define τ_i' as the first round such that $s_i \geqslant \frac{n}{2}$ starting from a configuration such that $s_i \geqslant \sqrt{n}\log n$. Using Lemma 6.2.3 and the hypotheses $s_i \geqslant \sqrt{n}\log n$ we get that, for each round t such that $s_i^{(t)} \leqslant \frac{n}{2}$, it holds

$$\begin{aligned}
&\mathbf{P}\left(s_i^{(t+1)} \geqslant (1 + 1/16)s_i \;\middle|\; \boldsymbol{c}^{(t)} : s_i \geqslant \sqrt{n}\log n\right) \\
&\quad\geqslant 1 - \exp\left(-2(s_i)^2/(32^2 n)\right) && (6.89)\\
&\quad\geqslant 1 - \exp^{-\mathcal{O}(\log^2 n)} && (6.90)\\
&\quad> 1 - n^{-a_1} && (6.91)
\end{aligned}$$

for any positive constant a_1. By iteratively applying the Union Bound we get that w.h.p. we have $\mathcal{O}(\log n)$ consecutive rounds of this multiplicative grow and thus it holds

$$\mathbf{P}\left(\tau_i' > b_2 \log n\right) \geqslant 1 - n^{-a_2}, \tag{6.92}$$

for two suitable positive constants a_2, b_2. Now we define, for any $i \in \{1, 2\}$, τ_i as the first round such that $s_i \geqslant n - \log n$ starting from a configuration such that $s_i \geqslant \frac{n}{2}$. Let us focus on community 1. Our assumption on the sign of the biases implies that there the minority color is *blue*. This, together with the fact that $s_1 \geqslant \frac{n}{2}$, implies $|B_1| \leqslant \frac{n}{4}$.

Consider a configuration such that $2\log n \leqslant |B_1| \leqslant \frac{n}{4}$. Then it holds

$$\mathbf{P}\left(|B_1^{(t+1)|} > |B_1^{(t)}|(1 - \frac{1}{5}) \;\middle|\; \boldsymbol{c}^{(t)}\right) < n^{-\Omega(1)}. \tag{6.93}$$

Indeed, using that $\frac{s_1}{2n} \geqslant \frac{n}{2} \geqslant \frac{c_2^2}{\sqrt{n}} + \frac{c_1^2|B_2|}{n|B_1|}$ and Lemma 6.2.2 we get

$$\begin{aligned}
&\mathbf{E}\left[|B_1^{(t+1)|} \;\middle|\; \boldsymbol{c}^{(t)}\right] \\
&\quad< |B_1|\left(1 - \frac{s_1}{2n} + \frac{c_2^2}{\sqrt{n}} + \frac{2c_1}{\sqrt{n}}\sqrt{\frac{|B_2|}{|B_1|}\left(\frac{1}{2} - \frac{s_1}{2n} + \frac{c_2^2}{\sqrt{n}} + \frac{c_1^2|B_2|}{n|B_1|}\right)}\right) && (6.94)\\
&\quad< |B_1|\left(1 - \frac{s_1}{2n} + \frac{c_2^2}{\sqrt{n}} + \frac{2c_1}{\sqrt{n}}\sqrt{\frac{|B_2|}{|B_1|}\cdot\frac{1}{2}}\right) && (6.95)
\end{aligned}$$

$$\leqslant |B_1| \left(1 - \frac{1}{4} + \frac{c_2^2}{\sqrt{n}} + \frac{2c_1}{\sqrt{n}} \sqrt{\frac{n}{2\log n} \cdot \frac{1}{2}}\right) \tag{6.96}$$

$$= |B_1| \left(1 - \frac{1}{4} + \frac{c_2^2}{\sqrt{n}} + \frac{2c_1}{\sqrt{4\log n}}\right) \tag{6.97}$$

$$= |B_1| \left(1 - \frac{1}{4} + \frac{c_2^2}{\sqrt{n}} + \frac{c_1}{\sqrt{\log n}}\right) \tag{6.98}$$

$$\leqslant |B_1| \left(1 - \frac{1}{5}\right), \tag{6.99}$$

where the last inequality holds for a sufficient large n. Then, using the multiplicative form of the Chernoff Bound, we get that:

$$\begin{aligned}
&\mathbf{P}\left(B_1^{(t+1)} \geqslant (1 - \frac{1}{25})|B_1|\right) \\
&\quad = \mathbf{P}\left(B_1^{(t+1)} \geqslant \left(1 + \frac{1}{5}\right)\left(1 - \frac{1}{5}\right)|B_1|\right) && (6.100)\\
&\quad \leqslant e^{-\left(1-\frac{1}{5}\right)|B_1|/125} && (6.101)\\
&\quad \leqslant e^{-\left(1-\frac{1}{5}\right)2\log n/125} && (6.102)\\
&\quad = n^{-\gamma_2}, && (6.103)
\end{aligned}$$

for a positive constant γ_2. Let τ_i'' be the first round such that $s_i \geqslant n - \log n$, starting from a configuration such that $s_i \geqslant \frac{n}{2}$. Equation (6.93) implies that, by an application of the Union Bound,

$$\mathbf{P}\left(\tau_i'' > b_3 \log n\right) \geqslant 1 - n^{-a_3} \tag{6.104}$$

for two suitable positive constants a_3, b_3. Thus, if we define τ_i as the first round such that $s_i \geqslant n - \log n$, starting from a configuration such that $s_i \geqslant \sqrt{n}\log n$, we get

$$\begin{aligned}
&\mathbf{P}\left(\tau_1 > (b_2 + b_3)\log n \bigcup \tau_2 > (b_2 + b_3)\log n\right) \\
&\quad < \mathbf{P}\left(\tau_1' > b_2 \log n \bigcup \tau_1'' > b_3 \log n \bigcup \tau_2' > b_2 \log n \bigcup \tau_2'' > b_3 \log n\right) && (6.105)\\
&\quad < \mathbf{P}\left(\tau_1' > b_2 \log n\right) + \mathbf{P}\left(\tau_1'' > b_3 \log n\right) + \\
&\qquad + \mathbf{P}\left(\tau_2' > b_2 \log n\right) + \mathbf{P}\left(\tau_2'' > b_3 \log n\right) && (6.106)\\
&\quad < n^{-a_4}, && (6.107)
\end{aligned}$$

for a suitable positive constant a_4. ■

Proof of Lemma 6.2.6

Proof. Let us define X_u as an indicator random variable such that $X_u = 1$ if node u will support the minority color of community i at the next round and $X_u = 0$ otherwise; let p_u be the probability of having $X_u = 1$. We approximate $\sum_{u \in V} X_u$ with a Poisson random variable using Le Cam's Theorem (Theorem E.3.2). Thanks

to Le Cam's Theorem, if $\sum_{u\in V_i} p_u^2 \leqslant \frac{1}{n^\varepsilon}$, for some positive constant ε, then any result that holds on the Poisson random variable with high probability will hold with high probability also for $\sum_{u\in V_i} X_u$.

Claim 6.2.1. *It holds that* $\sum_{u\in C_i} p_u^2 = \mathcal{O}\left(\frac{\log^3 n}{n}\right)$.

Proof. Let σ_i^- be the set of nodes supporting the minority color of community i and σ_i^+ be the set of nodes supporting the majority one. Let z_u be the number of neighbors of node u belonging to σ_i^-, and $\bar{z}_u$ be the number of neighbors of u belonging to σ_i^+. Notice that $|\sigma_i^-| = \log n$ and $|\sigma_i^+| = n - \log n$. Thus

$$\sum_{u\in C_i} p_u^2 = \sum_{\substack{u\in C_i,\\ u\in\sigma_i^+}} p_u^2 + \sum_{\substack{u\in C_i,\\ u\in\sigma_i^-}} p_u^2. \tag{6.108}$$

Let us analyze the two terms separately. As for the first term we have that

$$\sum_{\substack{u\in C_i,\\ u\in\sigma_i^+}} p_u^2 \leqslant \sum_{\substack{u\in C_i,\\ u\in\sigma_i^+}} \left(\frac{z_u+b}{a+b}\right)^4 \tag{6.109}$$

$$\overset{(a)}{\leqslant} a\left(\frac{\log n+b}{a+b}\right)^2 + (n-\log n-a)\left(\frac{b}{a+b}\right)^4 \tag{6.110}$$

$$\leqslant \frac{a\log^4 n + 4ab\log^3 n + 6ab^2\log^2 n + 4ab^3\log n + b^4 n - b^4\log n}{a^4} \tag{6.111}$$

$$= \frac{\log^4 n}{a^4} + \frac{4b\log^3 n}{a^4} + \frac{6b^2\log^2 n}{a^4} + \frac{4b^3\log n}{a^4} + \frac{b^4 n}{a^4} - \frac{b^4\log n}{a^4} \tag{6.112}$$

$$\overset{(b)}{\leqslant} \frac{\log^4 n}{n^2} + \frac{4\log^3 n}{n^2} + \frac{6\log^2 n}{n^2} + \frac{4\log^2 n}{n^2} + \frac{n}{n^2} - \frac{\log n}{n^2} \tag{6.113}$$

$$= \mathcal{O}\left(\frac{1}{n}\right) \tag{6.114}$$

where in (a) we used that at most a nodes can have all the $\log n$ nodes belonging to σ_i^- as neighbors, and in (b) that $b \geqslant 1$, $a \geqslant b\sqrt{n} \geqslant \sqrt{n}$ by hypothesis of Theorem 6.2.1, and thus $\frac{b}{a} \leqslant \frac{1}{\sqrt{n}}$.

As for the second term we have that

$$\sum_{\substack{u\in C_i,\\ u\in\sigma_i^-}} p_u^2 \leqslant \sum_{\substack{u\in C_i,\\ u\in\sigma_i^-}} \left(\frac{\bar{z}_u+b}{a+b}\right)^2 \tag{6.115}$$

$$\leqslant \sum_{\substack{u\in C_i,\\ u\in\sigma_i^-}} \left(\frac{\log n+b}{a+b}\right)^2 \tag{6.116}$$

$$= \log n\left(\frac{\log n+b}{a+b}\right)^2 \tag{6.117}$$

$$\leqslant \frac{\log^3 n}{a^2} + \frac{2b\log^2 n}{a^2} + \frac{b^2\log n}{a^2} \tag{6.118}$$

$$\overset{(a)}{\leqslant} \frac{\log^3 n}{n} + \frac{2\log n}{n} + \frac{\log n}{n} \tag{6.119}$$

$$= \mathcal{O}\left(\frac{\log^3 n}{n}\right), \tag{6.120}$$

where in (a) we used again that $a \geqslant \sqrt{n}$.

Finally, by combining the two bounds together, we get

$$\sum_{u \in C_i} p_u^2 = \mathcal{O}\left(\frac{1}{n}\right) + \mathcal{O}\left(\frac{\log^3 n}{n}\right) = \mathcal{O}\left(\frac{\log^3 n}{n}\right). \tag{6.121}$$

■

We now show that a Poisson(λ) random variable is upper bounded by $\mathcal{O}\left(\frac{\log n}{\log\log n}\right)$ w.h.p. as long as λ is constant w.r.t. n.

Claim 6.2.2. *Let $X \sim$ Poisson(λ) where λ is a positive real number, that is*

$$\mathbf{P}(X = i) = \frac{\lambda^i}{i!} e^{-\lambda}.$$

If $t = c \log n / \log\log n$ for some constant $c > 0$ and λ is constant w.r.t. n, then

$$\mathbf{P}(X > t) \leqslant n^{-c+o(1)}.$$

Proof. We have

$$\begin{aligned}
\mathbf{P}(X > t) &= \sum_{i=t+1}^{\infty} \frac{\lambda^i}{i!} e^{-\lambda} && (6.122)\\
&= \frac{\lambda^t}{t!} e^{-\lambda} \sum_{i=1}^{\infty} \frac{\lambda^i}{\prod_{j=1}^{i} (t+j)} && (6.123)\\
&\leqslant \frac{\lambda^t}{t!} e^{-\lambda} \sum_{i=1}^{\infty} \frac{\lambda^i}{t^i} && (6.124)\\
&\overset{(a)}{\leqslant} \frac{\lambda^t}{t!} e^{-\lambda} \sum_{i=1}^{\infty} 2^{-i} && (6.125)\\
&\overset{(b)}{\leqslant} \left(\frac{\lambda e}{t}\right)^t e^{-\lambda} && (6.126)\\
&= \left(\frac{\lambda e}{c\frac{\log n}{\log\log n}}\right)^{c\frac{\log n}{\log\log n}} e^{-\lambda} && (6.127)\\
&= \left(e^{\log(\lambda e) - \log c - \log\log n + \log\log\log n}\right)^{c\frac{\log n}{\log\log n}} e^{-\lambda} && (6.128)\\
&= e^{-\lambda + c\frac{\log n}{\log\log n}\left(\log\frac{\lambda e}{c} - \log\log n + \log\log\log n\right)} && (6.129)\\
&= e^{-c\log n(1-o(1))} && (6.130)\\
&= n^{-c+o(1)}, && (6.131)
\end{aligned}$$

where in (a) we used $t \geqslant 2\lambda$, and in (b) we used Stirling's formula $t! \geqslant \left(\frac{t}{e}\right)^t$. ■

As a last step, we show that $\lambda = \sum_{u \in V_i} p_u$ is bounded by a constant w.r.t. n.

Claim 6.2.3. *Let $\lambda = \sum_{u \in V_i} p_u$ and σ_i^- be the set of nodes supporting the minority color of community i and σ_i^+ be the set of nodes supporting the majority one. It holds that $\lambda = \mathcal{O}(1)$.*

Proof. Let z_u be the number of neighbors of node u belonging to σ_i^-, and $\bar{z}_u$ be the number of neighbors of u belonging to σ_i^+. Notice that $|\sigma_i^-| = \log n$ and $|\sigma_i^+| = n - \log n$. Thus

$$\lambda = \sum_{u \in C_i} p_u = \sum_{\substack{u \in C_i, \\ u \in \sigma_i^+}} p_u + \sum_{\substack{u \in C_i, \\ u \in \sigma_i^-}} p_u. \tag{6.132}$$

Let us analyze the two terms separately. As for the first term, similarly to Claim 6.2.1, we have that

$$\sum_{\substack{u \in C_i, \\ u \in \sigma_i^+}} p_u \leqslant \sum_{\substack{u \in C_i, \\ u \in \sigma_i^+}} \left(\frac{z_u + b}{a + b} \right)^2 \tag{6.133}$$

$$\overset{(a)}{\leqslant} a \left(\frac{\log n + b}{a + b} \right)^2 + (n - \log n - a) \left(\frac{b}{a + b} \right)^2 \tag{6.134}$$

$$= \frac{a \log^2 n + 2ab \log n + ab^2 + b^2 n - b^2 \log n - ab^2}{(a + b)^2} \tag{6.135}$$

$$\leqslant \frac{a \log^2 n + 2ab \log n + b^2 n - b^2 \log n}{a^2} \tag{6.136}$$

$$= \frac{\log^2 n}{a} + \frac{2b \log n}{a} + \frac{b^2 n}{a^2} - \frac{b^2 \log n}{a^2} \tag{6.137}$$

$$\overset{(b)}{\leqslant} \frac{\log^2 n}{\sqrt{n}} + \frac{2 \log n}{\sqrt{n}} + \frac{n}{n} - \frac{\log n}{n} \tag{6.138}$$

$$= 1 + \frac{\log^2 n}{\sqrt{n}} + \frac{2 \log n}{\sqrt{n}} - \frac{\log n}{n}, \tag{6.139}$$

where in (a) we used that at most a nodes can have all the $\log n$ nodes belonging to σ_i^- as neighbors, and in (b) that $b \geqslant 1$, $a \geqslant b\sqrt{n} \geqslant \sqrt{n}$ by hypothesis of Theorem 6.2.1, and thus $\frac{b}{a} \leqslant \frac{1}{\sqrt{n}}$.

As for the second term we have that

$$\sum_{\substack{u \in C_i, \\ u \in \sigma_i^-}} p_u \leqslant \sum_{\substack{u \in C_i, \\ u \in \sigma_i^-}} \left(\frac{\bar{z}_u + b}{a + b} \right) \tag{6.140}$$

$$\leqslant \sum_{\substack{u \in C_i, \\ u \in \sigma_i^-}} \left(\frac{\log n + b}{a + b} \right) \tag{6.141}$$

$$= \log n \left(\frac{\log n + b}{a + b} \right) \tag{6.142}$$

$$\leqslant \frac{\log^2 n}{a} + \frac{\log n}{a} \tag{6.143}$$

$$\overset{(a)}{\leqslant} \frac{\log^2 n}{\sqrt{n}} + \frac{\log n}{\sqrt{n}}, \tag{6.144}$$

where in (a) we used again that $a \geqslant \sqrt{n}$.

Finally, by combining the two bounds together, we get

$$\lambda = \sum_{u \in C_i} p_u = \sum_{\substack{u \in C_i, \\ u \in \sigma_i^+}} p_u + \sum_{\substack{u \in C_i, \\ u \in \sigma_i^-}} p_u \tag{6.145}$$

$$\leqslant 1 + \frac{\log^2 n}{\sqrt{n}} + \frac{2 \log n}{\sqrt{n}} - \frac{\log n}{n} + \frac{\log^2 n}{\sqrt{n}} + \frac{\log n}{\sqrt{n}} \tag{6.146}$$

$$= 1 + \frac{2 \log^2 n}{\sqrt{n}} + \frac{3 \log n}{\sqrt{n}} - \frac{\log n}{n} \tag{6.147}$$

$$= 1 + o(1) \tag{6.148}$$

$$= \mathcal{O}(1). \tag{6.149}$$

■

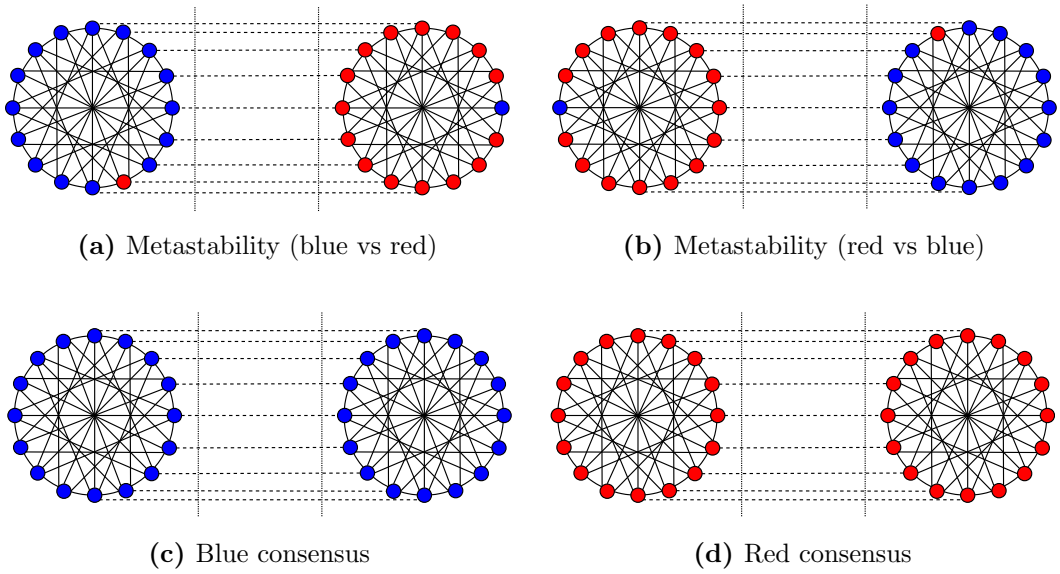
We showed that the number of wrongly colored nodes $\sum_{u \in V} X_u$ is well approximated by a Poisson random variable and such random variable, thanks to Claim 6.2.2, will be $\mathcal{O}\left(\frac{\log n}{\log \log n}\right)$ w.h.p. ■

6.2.1 A distributed graph clustering algorithm

We showed that, starting from a random initialization, the 2-Choices dynamics reaches an *almost-clustered* configuration within $\mathcal{O}(\log n)$ rounds with constant probability. This result is tight, given that there is constant probability that the two communities converge to the same color; Figure 6.3 shows the four possible behaviors of the 2-Choices dynamics on clustered regular graphs. Similarly to Lemma 6.2.1, it holds that with constant probability both the biases are unbalanced toward the same color, i.e., $s_1^{(0)} \geqslant h\sqrt{n}$ and $s_2^{(0)} \geqslant h\sqrt{n}$. It means that a suitable variant of Lemma 6.2.4 shows that there is constant probability that within $\mathcal{O}(\log \log n)$ rounds the process reaches a configuration such that $s_1^{(t)} \geqslant \sqrt{n} \log n$ and $s_2^{(t)} \geqslant \sqrt{n} \log n$. Then, Lemma 6.2.5 and Lemma 6.2.6 show that the system gets quickly stuck in a configuration where almost all nodes have the same color. This is a proof that, given the symmetric nature of the process, we need some luck in the initialization to reach an *almost-clustered* configuration.

In order to get an algorithm that works w.h.p. we sketch how to use the results of the previous sections to build a Community-Sensitive Labeling [BCM+18] within $\Theta(\log n)$ rounds. A Community-Sensitive Labeling (CSL) is made up by a labeling of the nodes and a predicate that can be applied to pairs of labels; it holds that, for all but a small number of outliers, the predicate is satisfied if the nodes belong to the same community, and it is not satisfied if the nodes belong to different communities. A visualization of the CSL can be seen in Figure 6.4.

Theorem 6.2.2 (LPA via CSL)**.** *Let $G = (V, E)$ be a connected and nonbipartite $(2n, d, b)$-clustered regular graph such that $\frac{b}{d} = \mathcal{O}(n^{-1/2})$ and $\lambda = \mathcal{O}(n^{-1/4})$. Let $\boldsymbol{c}^{(0)}$*



(a) Metastability (blue vs red)

(b) Metastability (red vs blue)

(c) Blue consensus

(d) Red consensus

Figure 6.3: A visualization of the four possible behaviors of the 2-Choices dynamics on clustered regular graphs after $\mathcal{O}(\log n)$ rounds.

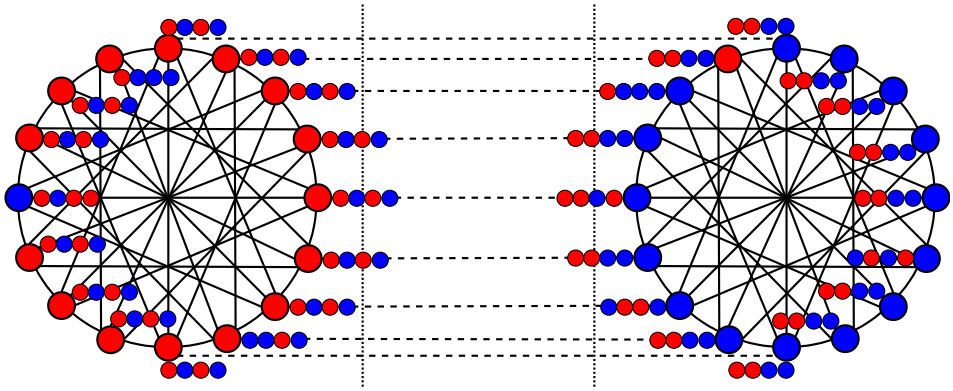


Figure 6.4: A visualization of the CSL used to design a distributed graph clustering algorithm by exploiting the metastable behavior of the 2-Choices dynamics.

be the initial configuration, where each node $u \in V$ picks a vector *of colors $\boldsymbol{c}_u^{(0)} \in \{red, blue\}^\ell$ sampled uniformly at random and independently from the other nodes, such that $\ell = c \log n$ for some positive constant c. Consider the resulting vector after $\Theta(\log n)$ rounds of independent parallel runs of the 2-Choices dynamics, each one working on a different component of the vector: For all the pairs of nodes but a polylogarithmic number, it holds that the vectors of nodes in the same community are equal while the vectors of nodes in different communities are different.*

Sketch of proof. As for the first part of the predicate, it is a simple application of Theorem 6.2.1. Indeed, at least one of the $\Theta(\log n)$ runs of the 2-Choices dynamics ends in an *almost-clustered* configuration with probability $1 - \gamma^{-\Theta(\log n)} = 1 - n^{-\Theta(1)}$. As for the second part we show that no matter if the process reaches an almost-clustering, nodes in the same community will have the same color with high

probability. This is consequence of Lemma 6.2.5 and of the following one, which we can prove by applying a general tool for Markov Chains [CGG+18, Lemma 4.5].

Lemma 6.2.7 (Consensus – Symmetry Breaking)**.** *Starting from any initial configuration $\boldsymbol{c}^{(0)}$, within $\mathcal{O}(\log n)$ rounds the system reaches a configuration $\boldsymbol{c}^{(t)}$ such that*

$$|s_1^{(t)}| \geqslant \sqrt{n}\log n \quad \textit{and} \quad |s_2^{(t)}| \geqslant \sqrt{n}\log n, \tag{6.150}$$

with high probability.

Proof. We are interested in bounding the hitting times τ_1 and τ_2 defined as, respectively, the first round such that $|s_1| \geqslant \sqrt{n}\log n$ and the first round such that $|s_2| \geqslant \sqrt{n}\log n$. In order to bound one of the two hitting times we use Lemma E.3.1, a general tool for Markov chains (see [CGG+18, Lemma 4.5]). Let Ω be the the configuration space of the process and $m = \sqrt{n}\log n$ the target value. We need to show that the following two properties hold for each $i \in \{1, 2\}$:

- For any positive constant h, there exists a positive constant $c_1 < 1$ such that for every $x \in \Omega : s_i < m$ we have

$$\mathbf{P}\left(s_i^{(t+1)} < h\sqrt{n} \;\middle|\; X_t = x\right) < c_1, \tag{6.151}$$

- There exist two positive constants ε and c_2 such that for every $x \in \Omega : h\sqrt{n} \leqslant s_i < m$ we have

$$\mathbf{P}\left(s_i^{(t+1)} < (1+\varepsilon)s_i \;\middle|\; X_t = x\right) < e^{-c_2 s_i^2/n}. \tag{6.152}$$

As for the first point, its proof is analogous to the proof of Lemma 6.2.1 since it is a consequence of the Berry-Esseen Theorem and of the variance of the process. As for the second point, we already proved it in Lemma 6.2.3, for $\varepsilon = \frac{1}{16}$ and $c_2 = \frac{2}{32^2}$. Thus we can conclude that $\mathbf{P}(\tau_1 > a\log n) < n^{-b}$ for two positive constants a, b. Using the Union Bound, it is immediate to show that both the hitting times are lower bounded by $a\log n$, w.h.p.:

$$\begin{aligned}
\mathbf{P}(\tau_1 \leqslant a\log n, \tau_2 \leqslant a\log n) &= 1 - \mathbf{P}(\tau_1 > a\log n \cup \tau_2 > a\log n) && (6.153)\\
&\geqslant 1 - [\mathbf{P}(\tau_1 > a\log n) + \mathbf{P}(\tau_2 > a\log n)] && (6.154)\\
&= 1 - 2n^b. && (6.155)
\end{aligned}$$

Note that, once a bias has reached a value of at least $\sqrt{n}\log n$, by an application of Lemma 6.2.3 and using the hypothesis $s_i \geqslant \sqrt{n}\log n$ and of the Union Bound, it follows that the bias remains above that value for $\Omega(\log n)$ rounds w.h.p. This means that the system reaches a configuration such that both the biases have value at least $\sqrt{n}\log n$ within $\mathcal{O}(\log n)$ rounds w.h.p. ■

Thus, most pairs of nodes can locally distinguish if they are in the same community with high probability by checking whether their vectors differ on any component. ■

6.3 Biological implications

6.3.1 A proof of concept for speciation

Evolutionary dynamics is the branch of genetics which studies how populations evolve genetically as a result of the interactions among the individuals [Dur11]. The study of evolutionary dynamics on graphs started with the investigation of the *fixation probability* of the *Moran process* (Figure 6.5) on different families of graphs, namely the probability that a new mutation with increased fitness eventually spreads across all individuals in the population [LHN05]. The Moran process has since then attracted the attention of the computer science community due to the algorithmic questions associated to its fixation probability [Gia16, GGG+17].

However, no simple dynamics has been proposed so far in the context of evolutionary graph theory for explaining one of evolution's fundamental phenomena, namely *speciation* [CO04]. Two fundamental classes of driving forces for speciation can be distinguished: *allopatric speciation* and *sympatric/parapatric speciation*. The former, which refers to the divergence of species resulting from geographical isolation, is nowadays considered relatively well understood [SAL+06]; on the contrary, the latter, namely divergence without complete geographical isolation, is still controversial [SAL+06, BF07]. In several evolutionary settings the spread of a mutation appears nonlinear with respect to the number of interacting individuals carrying the mutation, exhibiting a drift towards the most frequent phenotypes [CO04]. In this Section we look at the 2-Choices dynamics as a quadratic evolutionary dynamics on a clustered graph representing sympatric and parapatric scenarios. We regard the random initialization of the 2-Choices process as two inter-mixed populations of individuals with different genetic pools. The interactions for reproduction purposes between the two populations can be categorized in frequent interactions among individuals within an equal-size bipartition of the populations, i.e., the *communities*, and less frequent interactions between these two communities which, in later

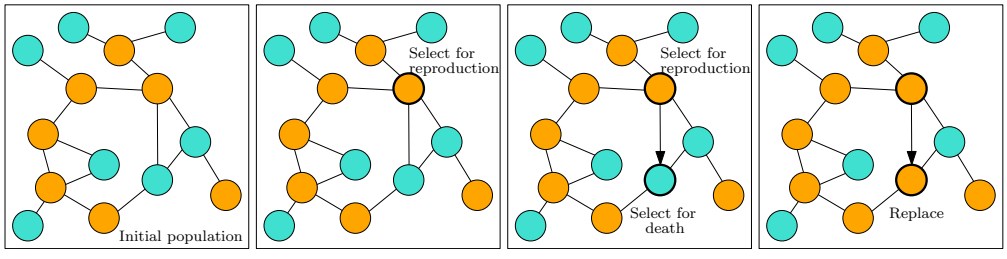


Figure 6.5: Visual representation of the *Moran process* (adapted from [LHN05]). At each time step an individual is randomly chosen for reproduction according to its *fitness*, and a second individual adjacent to it is randomly chosen for death; the offspring of the first individual then replaces the second. When the underlying network is regular, the process is equivalent to the Voter dynamics [BGKMT16].

stages of the differentiation process, may be interpreted as genetic admixture, i.e. interbreeding between two genetically-diverging populations [MDN+13].

Within the aforementioned framework our Theorem 6.2.1 provides an analytical evolutionary graph-theoretic proof of concept on how speciation can emerge from the simple nonlinear underlying dynamics of the evolutionary process at the population level.

6.3.2 On the process of innervation in muscular junctions

During mammalian development neuromuscular junctions and some other postsynaptic cells transition from having multiple neurons innervating onto them into being subject to innervation from a single neuron only [GL98, TWK+12, TL12]. This process takes place as synaptic sites are exchanged between different axons: Locations on the cell surface of the cell, on which neurons innervate, transition from being freed by the current innervating neuron to be occupied again by a new one [TL12] (see Figure 6.7).

In [TL12], it is shown that soon-to-be-eliminated axons rapidly reverse fate and grow to occupy vacant sites at a neuromuscular junction after they are artificially damaged in laboratory. Such evidence is argued to support the hypothesis that the process is driven by a form of competition at the level of neural terminations:

> *"This reversal supports the idea that axons take over sites that were previously vacated. Indeed, during normal development we observed withdrawal followed by takeover. The stimulus for axon growth is not postsynaptic cell inactivity because axons grow into unoccupied sites even when target cells are functionally innervated. These results demonstrate competition at the synaptic level and enable us to provide a conceptual framework for understanding this form of synaptic plasticity."*
> — Turney et al. [TL12].

The authors then provide a simplistic model for the aforementioned process based on evolutionary graph theory. Their model, illustrated in Figure 6.8, is equivalent to the Voter dynamics when the underlying graph is assumed to be regular.

In this section we argue that our Theorem 6.2.1 provides evidence for the fact that, in order for a model based on dynamics[2] to comply with experimental evidence on the outcome of the innervation process, either the innervation sites do not exhibit spatial bottlenecks or the dynamics cannot be based on majority-like mechanism.

Theorem 6.2.1 shows that, when the 2-Choices dynamics takes place on a clustered graph from a random initial configuration, there is a constant probability that the system will converge to a configuration in which the two clusters maintain an almost-consensus on two different values. This should be contrasted with

[2]We remark that we did not find any decisive evidence in the experimental literature that the process could not be better explained through other factors such as communication among axons via molecular clues (similarly to other developmental neural process such as [AAB+11]).

the aforementioned phenomenon of synapse elimination at developing neuromuscular junctions [TL12, TWK+12], where the outcome is a *consensus* configuration in which the whole neuromuscular junction is innervated by a single axon only. Hence, our Theorem 6.2.1 suggests that, if the competition among axons for innervating a postsynaptic cell follows a local behavior akin to the 2-Choices dynamics, the topology of the sites as formalized in [TL12] (Figure 6.8), should not exhibit a clustered structure, i.e. a partition in two communities with good expansion properties while being separated by a sparse cut as in the regular clustered graphs of Theorem 6.2.1. On the other hand, if experimental evidence would confirm that the graph associated to the innervation site can indeed exhibit a structure similar to the graphs of Theorem 6.2.1, then the dynamics implemented by axons should qualitatively differ from a super-linear majority dynamics such as the 2-Choices one.

We complement our previous argument with simulations on stochastic block models with two communities [HLL83], on a class of dynamics which generalizes the 2-Choices dynamics. We call the latter class *y-degree majority dynamics*.

Definition 6.3.1 (y-degree majority dynamics)**.** *In the y-degree majority dynamics, for a real value $y \geqslant 0$, a generic node u updates its current color with color σ with probability*

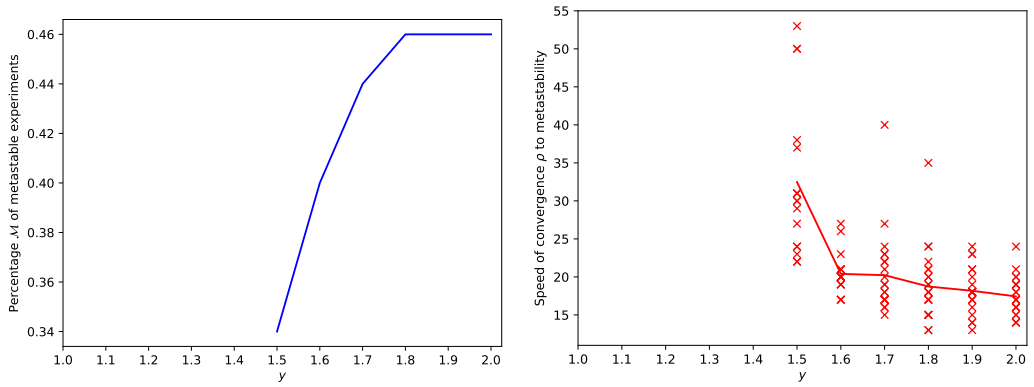
$$\left(\frac{|N(u) \cap S_\sigma|}{d_u}\right)^y, \tag{6.156}$$

where S_σ is the set of nodes supporting color σ, $N(u)$ is the set of neighbors of u, and d_u is the degree of u.

Observe that the node stays of its current color with probability $1-\sum_\sigma \left(\frac{|N(u)\cap S_\sigma|}{d_u}\right)^y$. As a special case, for $y = 2$ we get the 2-Choices dynamics and for $y = 1$ we get the Voter dynamics.

The stochastic block model (with two communities) is generated by dividing the nodes in two sets of equal size, called communities, and including each edge with probability p if its two endpoints are in the same community, and with probability q if instead the two endpoints belong to different communities. To match the parameters of Theorem 6.2.1, we set $p = n^{-\frac{1}{4}}$ and $q = n^{-\frac{3}{4}}$, with $n = 10000$, and we vary y from 1 to 2 with stepsize 0.1.

For each run (50 in total) the y-degree majority dynamics is run until the number of rounds exceeds $\sqrt{n} = 100$. The outcome of the simulations is illustrated in Table 6.1 and Figures 6.6a and 6.6b. The results show that, as y decreases from 2 to 1.5, the probability that the dynamics converges (and maintains) a non-consensus configuration in which the two clusters support different colors deteriorates from roughly $\frac{1}{2}$ to circa $\frac{1}{3}$, with a steeply decreasing derivative, while the time to converge to the almost-consensus (for the run for which that happens) starts increasing rapidly towards $y = 1.5$. Below $y = 1.5$, no convergence to an almost-clustered configuration occurs in any of the 50 runs before exceeding the time limit. This also



(a) Percentage $\mathcal{M}$ of metastable runs with respect to y-degree majority dynamics.

(b) Number of rounds ρ to converge to metastable configurations with respect to y-degree majority dynamics.

Figure 6.6: Experimental results of y-degree majority dynamics on clustered graphs. Details reported in Table 6.1.

Table 6.1: Results of the experiments on y-degree majority dynamics on clustered graphs sampled from the stochastic block model with two communities $\mathcal{G}_{n,p,q}$. Each experiment has been repeated 50 times. The parameters used for the experiments are: $n = 10000$, $p = n^{-1/4}$, $q = n^{-3/4}$. The value $\mathcal{M}$ represents the percentage of runs for which an *almost-clustered* configuration is reached in less than $\sqrt{n} = 100$ iterations; the value ρ represents the average number of rounds for the process to converge to an almost-clustered configuration (in which the two communities maintain an almost-consensus on different colors, conditioning on such state being reached in the current run).

y	1.0	1.1	1.2	1.3	1.4	1.5	1.6	1.7	1.8	1.9	2.0
$\mathcal{M}$	0.00	0.00	0.00	0.00	0.00	0.34	0.40	0.44	0.46	0.46	0.46
ρ	0.00	0.00	0.00	0.00	0.00	32.47	20.40	20.23	18.74	18.17	17.43

confirms the positive results numerically obtained in [TL12] for their dynamics, which corresponds to the Voter dynamics, where the probability and time for the dynamics to converge to a consensus have been analytically characterized [HP01].

We remark that, as a byproduct, our numerical experiments also show the low sensitivity of Theorem 6.2.1 to the regularity assumption.

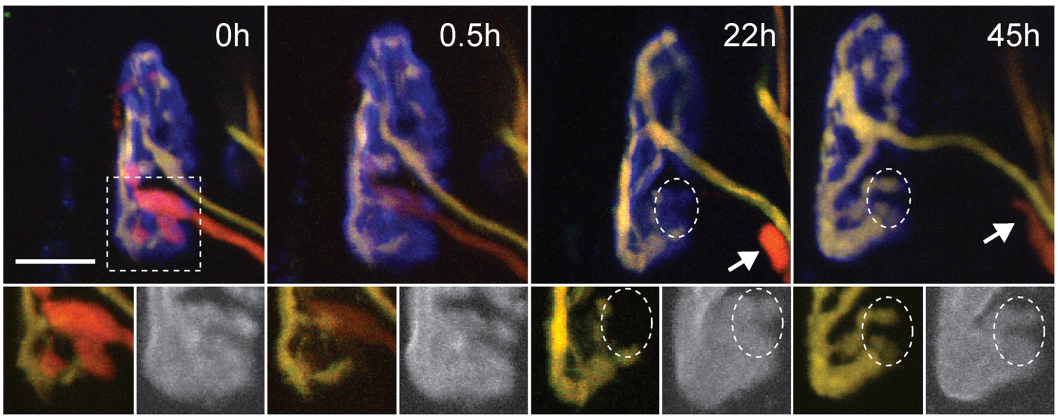


Figure 6.7: *Synaptic withdrawal precedes takeover during naturally occurring synapse elimination* (Turney & Lichtman 2012 [TL12]). An illustration from [TL12] showing the competition among different axons for innervating receptive areas during development. Shown are four views of a multiply innervated neuromuscular junction from the sternomastoid muscle of a living mouse viewed over two days (scale of 10 micrometers). At the first view (0 h) the orange-colored axon occupied the lower part of the junction and a small branch at the top. Insets below show axons (left) and receptors (right, in gray) of the bottom region of the junction. To more clearly see the extent of the territory occupied by the yellow axon, the fluorescent protein in orange-colored input was bleached at the nerve entry zone for several minutes without damaging the axon, using visible continuous wave laser light. A day later (22 h) the orange-colored input is no longer at the junction, but a remnant of the retracting axon is visible (arrow). Several sites that had previously been occupied by the orange axon are now vacant (dashed ellipse). However, by 45 h, the remaining input grew into the vacated territory. The delay between the withdrawal of the orange axon and the takeover by the yellow one is similar to time course observed following removal of an input by laser irradiation (∼1 day).

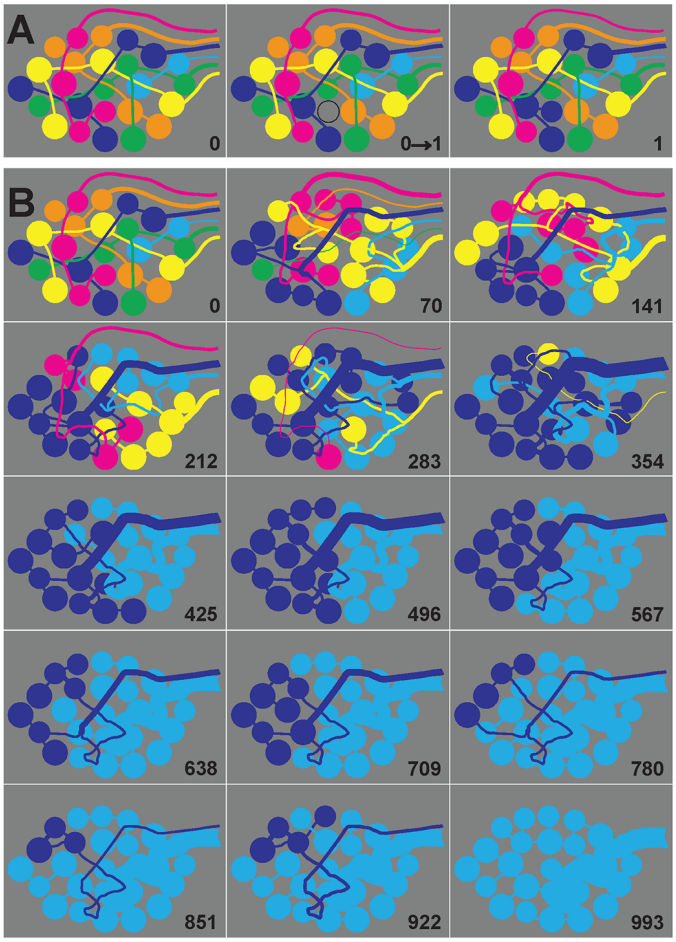


Figure 6.8: *Model of synaptic competition based on piecewise withdrawal from synaptic sites* (courtesy of Turney & Lichtman 2012 [TL12]). The simple graphical model gives rise to many of the features that have previously been experimentally observed. A junction is represented by a set of synaptic sites that are randomly distributed among six innervating axons each with a different color (leftmost panel). Based on the findings in this article showing that synaptic vacancies can induce nearby axonal branches of either the same or a different axon to grow, synaptic competition is simulated in [TL12] as an iterative process of axon withdrawal from a randomly selected site followed by takeover of the vacated site. As shown in the first two panels the process starts with the pink axon losing a synaptic contact. The vacated site is subsequently reoccupied by one of the immediately neighboring axons (in this case the dark blue one; see right panel). This stepwise process is repeated approximately 1000 times until all the sites become innervated by the same axon and the junction is in its mature singly innervated state. Over the course of this simulation, the synaptic contacts of each axon become progressively clustered, and eventually, when the junction has only two remaining inputs, the axons become completely segregated as has been seen in normal development [GL98].

III

Averaging Dynamics

7

Reconstruction of Volume-Regular Graphs

In this chapter we consider the *Averaging dynamics* (Algorithm 3) on a class of graphs that exhibits a clustered structure and that we call *volume-regular graphs* (see Section 7.1). In the classical setting, we are given a (possibly weighted) graph G and an integer k. Our goal is to partition the vertex set of G into k disjoint subsets, so that the k induced subgraphs have high inner and low outer expansion. Results such as those presented in [LGT14, PSZ17] provide further theoretical justification for spectral clustering, showing that, for a well-clustered graph with k communities, the profiles of the first k eigenvectors are correlated with the underlying community structure of G. Instead, as discussed in Chapter 4, the approach proposed in [BCN+17b] suggests that observing the temporal evolution of the power method applied to an initial random vector may, at least in some cases, provide equivalent information, without requiring explicit eigenvector computations. There, the authors show that, with a simple labeling of the nodes that depends on the local update rule, the dynamics converges in logarithmic time to a coloring that reflects the underlying cut. They further elaborated on how to extend the proposed approach to the case of multiple communities, providing an analysis for a strongly regular version of the stochastic block model with multiple communities. Their analysis exploits a property of the eigenvectors of the transition matrix of a simple random walk on the graph: If the graph is sampled by the regular stochastic block model with two equally sized communities, the eigenvector associated to the second largest eigenvalue of the transition matrix of a random walk on the graph is a linear combination of the indicator vectors of the communities.

In this chapter we push their argument to its limit, considering the most general class of graphs with "stepwise" vectors (with constant values over the components of nodes belonging to the same community) that we introduce using a connection with lumpability of Markov chains [KS60, TK06]. We prove that also in this general case the Averaging dynamics can be used to recover the underlying clustered structure of the graph.

> **Informal description of Theorem 7.2.1.** Let G be a k-volume-regular graph and let the nodes of G perform ℓ parallel and independent runs of the Averaging dynamics. If the stepwise eigenvectors are those associated to the first k eigenvalues and the gap between the k-th and the $(k+1)$-th eigenvalues is sufficiently large, after a logarithmic number of rounds, a long time interval starts where the labels of the nodes reveal the community structure of the graph, with high probability.

We remark that the graphs we consider neither require having communities of the same size nor to be regular as in [BCN+17b]; volume-regularity, in fact, is a weaker notion than regularity of a graph. The analysis of the Averaging dynamics on this class is considerably harder, but it is likely to provide insights into the challenges of analyzing the general case, without all the intricacies of the latter.

We further show that variants of the Averaging dynamics (and/or its labeling rule) can address different problems (e.g., identifying bipartiteness) and/or other graph classes (Sections 7.2.2 and 7.2.3).

7.1 Preliminaries

Notation. Consider an undirected edge-weighted graph $G = (V, E, w)$ with non-negative weights. For each node $u \in V$, we denote by $\delta(u)$ the *volume*, or *weighted degree*, of node u, namely $\delta(u) = \sum_{v:(u,v)\in E} w(u,v)$. D denotes the diagonal matrix, such that $D_{uu} = \delta(u)$ for each $u \in V$. Without loss of generality we assume $\min_u \delta(u) = 1$, since the behavior of the Averaging dynamics (and the corresponding analysis) is not affected by a normalization of the weights. We refer to the maximum volume of a node as $\Delta := \max_u \delta(u)$.

In the remainder, W denotes the *weighted adjacency matrix* of G, while $P = D^{-1}W$ is the *transition matrix* of a random walk on G, in which a transition from node u to node v occurs with probability proportional to $w(u,v)$. We call $\lambda_1, \ldots, \lambda_n$ the eigenvalues of P, in non-increasing order, and $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n$ a family of eigenvectors of P, such that $P\boldsymbol{v}_i = \lambda_i \boldsymbol{v}_i$. We let $N = D^{-\frac{1}{2}} W D^{-\frac{1}{2}} = D^{\frac{1}{2}} P D^{-\frac{1}{2}}$ denote the *normalized weighted adjacency matrix* of G. Note that N is symmetric and that its spectrum is the same as that of P. We denote by $\boldsymbol{w}_1, \ldots, \boldsymbol{w}_n$ a family of eigenvectors of N, such that $N\boldsymbol{w}_i = \lambda_i \boldsymbol{w}_i$. It is important to note that $\boldsymbol{w_i}$ is an eigenvector of N if and only if $D^{-\frac{1}{2}}\boldsymbol{w}_i$ is an eigenvector of P.

7.1.1 Averaging dynamics

The simple algorithm we consider in this chapter, named Averaging dynamics (Algorithm 3) after [BCM+18] in which the algorithm was first proposed, can be seen as an application of the power method, augmented with a Rademacher initialization and a suitable labeling scheme. In this form, it is best described as a distributed

process, executed by the nodes of an underlying edge-weighted graph. The Averaging dynamics can be used as a building-block to achieve "community detection" in some classes of "regular" and "almost regular" graphs. Herein, we extend its use and analysis to broader graph classes and, in one case, to a different problem.

Spectral decomposition of the transition matrix. Let $\boldsymbol{x}^{(t)}$ denote the *state vector* at time t, i.e., the vector whose u-th entry is the value held by node u at time t. We let $\boldsymbol{x}^{(0)} = \boldsymbol{x}$ denote the initial state vector. Globally, the *averaging* update rule of Algorithm 3 corresponds to one iteration of the power method, in this case an application of the transition matrix P to the current state vector, i.e., $\boldsymbol{x}^{(t)} = P\boldsymbol{x}^{(t-1)}$. We can write

$$\boldsymbol{x}^{(t)} = P^t\boldsymbol{x} = D^{-\frac{1}{2}}N^tD^{\frac{1}{2}}\boldsymbol{x} \overset{(a)}{=} D^{-\frac{1}{2}}\sum_{i=1}^{n}\lambda_i^t\boldsymbol{w}_i\boldsymbol{w}_i^\intercal\sum_{i=1}^{n}\beta_i\boldsymbol{w}_i = \sum_{i=1}^{n}\lambda_i^t\beta_iD^{-\frac{1}{2}}\boldsymbol{w}_i, \quad (7.1)$$

where in (a) we spectrally decomposed the matrix N^t and expressed the vector $D^{\frac{1}{2}}\boldsymbol{x}$ as a linear combination of the eigenvectors of N, i.e., $D^{\frac{1}{2}}\boldsymbol{x} = \sum_{i=1}^{n}\beta_i\boldsymbol{w}_i$, with $\beta_i = \langle D^{\frac{1}{2}}\boldsymbol{x}, \boldsymbol{w}_i\rangle$. By explicitly writing the β_is and by noting that $\boldsymbol{w}_i = \frac{D^{\frac{1}{2}}\boldsymbol{v}_i}{\|D^{\frac{1}{2}}\boldsymbol{v}_i\|}$ we conclude that

$$\boldsymbol{x}^{(t)} = \sum_{i=1}^{n}\lambda_i^t\frac{\langle D^{\frac{1}{2}}\boldsymbol{x}, D^{\frac{1}{2}}\boldsymbol{v}_i\rangle}{\|D^{\frac{1}{2}}\boldsymbol{v}_i\|}D^{-\frac{1}{2}}\frac{D^{\frac{1}{2}}\boldsymbol{v}_i}{\|D^{\frac{1}{2}}\boldsymbol{v}_i\|} = \sum_{i=1}^{n}\lambda_i^t\alpha_i\boldsymbol{v}_i, \quad (7.2)$$

where $\alpha_i := \frac{\langle D^{\frac{1}{2}}\boldsymbol{x}, D^{\frac{1}{2}}\boldsymbol{v}_i\rangle}{\|D^{\frac{1}{2}}\boldsymbol{v}_i\|^2} = \frac{\boldsymbol{x}^\intercal D\boldsymbol{v}_i}{\|D^{\frac{1}{2}}\boldsymbol{v}_i\|^2}$.

Note that $\lambda_1 = 1$ and $\boldsymbol{v}_1 = \mathbf{1}$,[1] since P is stochastic and, if G is connected and non bipartite, $\lambda_i \in (-1, 1)$ for every $i > 1$. The long term behavior of the dynamics can be written as

$$\lim_{t\to\infty}\boldsymbol{x}^{(t)} = \lim_{t\to\infty}\sum_{i=1}^{n}\lambda_i^t\alpha_i\boldsymbol{v}_i = \alpha_1\mathbf{1}, \quad (7.3)$$

with

$$\alpha_1 = \frac{\sum_{u\in V}\delta(u)\boldsymbol{x}(u)}{\sum_{u\in V}\delta(u)} = \sum_{u\in V}\frac{\delta(u)}{\mathrm{vol}(V)}\boldsymbol{x}(u), \quad (7.4)$$

i.e., each node converges to the initial global weighted average of the network.

7.1.2 Community-sensitive algorithms

We give the following definition of *community sensitive algorithm*, that closely resembles that of locality-sensitive hashing (see, e.g., [LRU14]).

Definition 7.1.1 (Community-sensitive algorithm)**.** *Let $\mathcal{A}$ be a randomized algorithm that takes in input a (possibly weighted) graph $G = (V, E)$ with a hidden partition $\mathcal{V} = \{V_1, \ldots, V_k\}$ and assigns a Boolean value $\mathcal{A}(G)[v] \in \{0, 1\}$ to each node $v \in V$. We say $\mathcal{A}$ is an (ε, δ)-Community-sensitive algorithm, for some $\varepsilon, \delta > 0$, if the following two conditions hold:*

[1] Here and in the remainder, $\mathbf{1}$ denotes the vector whose entries are 1.

1. *For each set V_i of the partition and for each pair of nodes $u, v \in V_i$ in that set, the probability that the algorithm assigns the same Boolean value to u and v is at least $1 - \varepsilon$,*

$$\forall i \in [k],\ \forall u, v \in V_i,\ \mathbf{P}\left(\mathcal{A}(G)[u] = \mathcal{A}(G)[v]\right) \geqslant 1 - \varepsilon. \tag{7.5}$$

2. *For each pair V_i, V_j of distinct sets of the partition and for each pair of nodes $u \in V_i$ and $v \in V_j$, the probability that the algorithm assigns the same value to u and v is at most δ,*

$$\forall i, j \in [k] \ \text{ with } i \neq j,\ \forall u \in V_i, \forall v \in V_j,\ \mathbf{P}\left(\mathcal{A}(G)[u] = \mathcal{A}(G)[v]\right) \leqslant \delta. \tag{7.6}$$

For example, for $(\varepsilon, \delta) = (1/n, 1/2)$, an algorithm that simply assigns the same value to all nodes would satisfy the first condition but not the second one, while an algorithm assigning 0 or 1 to each node with probability $1/2$, independently of the other nodes, would satisfy the second condition but not the first one.

Note that Algorithm 3 is a distributed algorithm that, at each round t, assigns one out of two labels to each node of a graph. In the next section (see Theorem 7.2.1) we prove that a time window $[T_1, T_2]$ exists, such that for all rounds $t \in [T_1, T_2]$, the assignment of the Averaging dynamics satisfies both conditions in Definition 7.1.1: The first condition with $\varepsilon = \varepsilon(n) = \mathcal{O}(n^{-\frac{1}{2}})$, the second with $\delta = \delta(n) = 1 - \Omega(1)$.

Community-sensitive labeling. If we execute $\ell = \Theta(\log n)$ independent runs of an (ε, δ)-Community-sensitive algorithm $\mathcal{A}$, each node is assigned a *signature* of ℓ binary values, with pairwise Hamming distances probabilistically reflecting community membership of the nodes. More precisely, let $\mathcal{A}$ be an $(\varepsilon,\ \delta)$-Community-sensitive algorithm and let $\mathcal{A}_1, \ldots, \mathcal{A}_\ell$ be $\ell = \Theta(\log n)$ independent runs of $\mathcal{A}$. For each node $u \in V$, let $\boldsymbol{s}(u) = (s_1(u), \ldots, s_\ell(u))$ denote the *signature* of node u, where $s_i(u) = \mathcal{A}_i(G)[u]$. For each pair nodes u, v, let $h(u, v) = |\{i \in [\ell] \,:\, s_i(u) \neq s_i(v)\}|$ be the Hamming distance between $\boldsymbol{s}(u)$ and $\boldsymbol{s}(v)$. The following lemma follows from a straightforward application of Chernoff bounds.

Lemma 7.1.1 (From Community-sensitive algorithm to Community-sensitive labeling)**.** *Let $\mathcal{A}$ be an $(\varepsilon,\ \delta)$-Community-sensitive algorithm with $\varepsilon = o(1)$ and $\delta = 1 - \Omega(1)$. For large enough $\ell = \Theta(\log n)$, two positive constants α, β exist, with $0 \leqslant \alpha < \beta \leqslant 1$, such that for each pair of nodes $u, v \in V$ it holds that:*

1. *If u and v belong to the same community then $h(u, v) \leqslant \alpha\ell$, w.h.p.*

2. *If u and v belong to different communities then $h(u, v) \geqslant \beta\ell$, w.h.p.*

Proof. If u and v belong to the same community, then $\mathbf{E}\left[h(u, v)\right] \leqslant \varepsilon\ell$. If they belong to different communities, then $\mathbf{E}\left[h(u, v)\right] \geqslant (1 - \delta)\ell$. The thesis follows by a standard application of Chernoff bounds, e.g., by choosing $\alpha = 2\varepsilon$ and $\beta = \frac{1-\delta}{2}$. ■

7.1.3 Volume-regular graphs

Recall that, for an undirected edge-weighted graph $G = (V, E, w)$, we denote by $\delta(u)$ the volume a node $u \in V$, i.e., $\delta(u) = \sum_{v:(u,v)\in E} w(u,v)$. Note that the transition matrix P of a random walk on G is such that $P_{uv} = w(u,v)/\delta(u)$. Given a partition $\mathcal{V} = \{V_1, \dots, V_k\}$ of the set of nodes V, for a node $u \in V$ and a partition index $i \in [k]$, $\delta_i(u)$ denotes the overall weight of edges connecting u to nodes in V_i, $\delta_i(u) = \sum_{v\in V_i\,:\,u,v\in E} w(u,v)$. Hence, $\delta(u) = \sum_{i=1}^{k} \delta_i(u)$.

Definition 7.1.2 (Volume-regular graph)**.** *Let $G = (V, E, w)$ be an undirected edge-weighted graph with $|V| = n$ nodes and let $\mathcal{V} = \{V_1, \dots, V_k\}$ be a k-partition of the nodes, for some $k \in [n]$. We say that G is* volume-regular *with respect to $\mathcal{V}$ if, for every pair of partition indexes $i, j \in [k]$ and for every pair of nodes $u, v \in V_i$, $\frac{\delta_j(u)}{\delta(u)} = \frac{\delta_j(v)}{\delta(v)}$. We say that G is k-volume-regular if there exists a k-partition $\mathcal{V}$ of the nodes such that G is volume-regular with respect to $\mathcal{V}$.*

In other words, G is volume-regular if there exists a partition of the nodes such that the fraction of a node's volume toward a set of the partition is constant across nodes of the same set. Note that all graphs with n nodes are trivially 1- and n-volume-regular.

Let $G = (V, E, w)$ be a k-volume-regular graph and let P be the transition matrix of a random walk on G. In the next lemma we prove that the span of k linearly independent eigenvectors of P equals the span of the indicator vectors of the k communities of G. The proof makes use of the correspondence between random walks on volume-regular graphs and *ordinary lumpable* Markov chains [KS60]; in particular the result follows from Lemma 7.1.3 and Lemma 7.1.4, that we prove in Section 7.1.3.

Lemma 7.1.2. *Let P be the transition matrix of a random walk on a k-volume-regular graph $G = (V, E, w)$ with k-partition $\mathcal{V} = \{V_1, \dots, V_k\}$. There exists a family $\{\boldsymbol{v}_1, \dots, \boldsymbol{v}_k\}$ of linearly independent eigenvectors of P such that $Span(\{\boldsymbol{v}_1, \dots, \boldsymbol{v}_k\}) = Span(\{\mathbf{1}_{V_1}, \dots, \mathbf{1}_{V_k}\})$, with $\mathbf{1}_{V_i}$ the indicator vector of the i-th set of the partition, for $i \in [k]$.*

In the rest of the chapter we call "stepwise" the eigenvectors of P that can be written as linear combinations of the indicator vectors of the communities. In the next definition, we formalize the fact that a k-volume-regular graph is *clustered* if the k linearly independent stepwise eigenvectors of P, whose existence is guaranteed by the above lemma, are associated to the k largest eigenvalues of P.

Definition 7.1.3 (Clustered volume-regular graph)**.** *Let $G = (V, E, w)$ be a k-volume-regular graph and let P be the transition matrix of a random walk on G. We say that G is a* clustered k-volume-regular graph *if the k stepwise eigenvectors of P are associated to the first k largest eigenvalues of P.*

Volume-regular graphs and lumpable Markov chains

The class of volume-regular graphs is deeply connected with the definition of *lumpability* [KS60] of Markov chains. We here first recall the definition of lumpable Markov chain and then show that a graph G is volume-regular if and only if the associated weighted random walk is a lumpable Markov chain.

Definition 7.1.4 (Ordinary lumpability of Markov Chains)**.** *Let $\{X_t\}_t$ be a finite Markov chain with state space V and transition matrix $P = (P_{uv})_{u,v\in V}$ and let $\mathcal{V} = \{V_1, \dots, V_k\}$ be a partition of the state space. Markov chain $\{X_t\}_t$ is* ordinary lumpable *with respect to $\mathcal{V}$ if, for every pair of partition indexes $i, j \in [k]$ and for every pair of nodes in the same set of the partition $u, v \in V_i$, it holds that*

$$\sum_{w\in V_j} P_{uw} = \sum_{w\in V_j} P_{vw}. \tag{7.7}$$

We define the lumped matrix $\widehat{P}$ *of the Markov Chain as the matrix such that $\widehat{P}_{ij} = \sum_{w\in V_i} P_{uw}$, for any $u \in V_j$.*

We first prove that random walks on Volume-regular graphs define exactly the subset of reversible and ordinary lumpable Markov chains.

Lemma 7.1.3. *A reversible Markov chain $\{X_t\}_t$ is ordinary lumpable if and only if it is a random walk on a volume-regular graph.*

Proof. We first assume that $\{X_t\}_t$ is ordinary lumpable and then assume G is k-volume-regular with respect to the partition $\mathcal{V} = \{V_1, \dots, V_k\}$.

1. Assume first assume that $\{X_t\}_t$ is ordinary lumpable and let P be the corresponding transition matrix. Consider the weighted graph $G = (V, E, w)$ obtained from P as follows: V corresponds to the set of states in P, while $w(u, v) = \pi(u)P_{uv}$, for every $u, v \in V$, with π the stationary distribution of P. Note that G is an undirected graph, i.e., $w(u, v) = \pi(u)P_{uv} \overset{(a)}{=} \pi(v)P_{vu} = w(v, u)$, where (a) holds because P is reversible. Moreover

$$\delta(u) = \sum_{z\in V} w(u, v) = \sum_{z\in V} \pi(u)P_{uv} = \pi(u) \sum_{z\in V} P_{uv} \overset{(a)}{=} \pi(u), \tag{7.8}$$

   where (a) holds because P is stochastic. Thus G meets Definition 7.1.2 because, for any $u, v \in V_i$,

$$\frac{\delta_j(u)}{\delta(u)} = \frac{1}{\pi(u)} \sum_{z\in V_j} w(u, z) = \sum_{z\in V_j} P_{uz} = \sum_{z\in V_j} P_{vz} = \frac{1}{\pi(v)} \sum_{z\in V_j} w(v, z) = \frac{\delta_j(v)}{\delta(v)}. \tag{7.9}$$

2. Next, assume G is k-volume-regular with respect to the partition $\mathcal{V} = \{V_1, \dots, V_k\}$. Let P be the transition matrix of the corresponding random walk. For every

$i, j \in [k]$ and for every $u, v \in V_i$ we have:

$$\sum_{z \in V_j} P_{uz} = \sum_{z \in V_j} \frac{w(u,z)}{\delta(u)} = \frac{\delta_j(u)}{\delta(u)} \overset{(a)}{=} \frac{\delta_j(v)}{\delta(v)} = \sum_{z \in V_j} \frac{w(v,z)}{\delta(v)} = \sum_{z \in V_j} P_{vz},$$

where (a) follows from Definition 7.1.2. Moreover note that P is reversible with respect to distribution π, where $\pi(u) = \frac{\delta(u)}{\mathrm{vol}(G)}$.

■

Note that infinitely many k-volume-regular graphs have the same k-ordinary lumpable random walk chain.

We next show that a Markov chain is k-ordinary lumpable if and only if the corresponding transition matrix P has k stepwise, linearly independent eigenvectors.

Lemma 7.1.4. *Let P be the transition matrix of a Markov chain. Then P has k stepwise linearly independent eigenvectors if and only if P is ordinary lumpable.*

Proof. We divide the proof in two parts. First, we assume that P is ordinary lumpable and show that P has k stepwise linearly independent eigenvectors. Second, we assume that P has k stepwise linearly independent eigenvectors and show that P is ordinary lumpable.

1. Let P be ordinary lumpable and $\widehat{P}$ its lumped matrix. Let $\lambda_i, \boldsymbol{v}_i$ be the eigenvalues and eigenvectors of $\widehat{P}$, for each $i \in [k]$. Let $\boldsymbol{w}_i \in \mathbb{R}^n$ be a stepwise vector defined as

$$\boldsymbol{w}_i = (\boldsymbol{v}_i(1), \ldots, \boldsymbol{v}_i(1),\ \boldsymbol{v}_i(2), \ldots, \boldsymbol{v}_i(2),\ \ldots,\ \boldsymbol{v}_i(k), \ldots, \boldsymbol{v}_i(k))^\intercal, \tag{7.10}$$

   where $\boldsymbol{v}_i(j)$ indicates the j-th component of $\boldsymbol{v}_i$, and then the n_j components relative to V_j are all equal to $\boldsymbol{v}_i(j)$.

   Since the eigenvectors $\boldsymbol{v}_i$ of $\widehat{P}$ are linearly independent, the vectors $\boldsymbol{w}_i$ are also linearly independent. Moreover, it is easy to see that $P\boldsymbol{w}_i = \lambda_i \boldsymbol{w}_i$ by just verifying the equation for every $i \in [k]$.

2. Assume P has k stepwise linearly independent eigenvectors $\boldsymbol{w}_i$, associated to k eigenvalues λ_i, for each $i \in [k]$. Let $\boldsymbol{v}_i \in \mathbb{R}^k$ the vector that has as components the k constant values in the steps of $\boldsymbol{w}_i$. Since the $\boldsymbol{w}_i$ are linearly independent, the $\boldsymbol{v}_i$ also are.

   For every eigenvector $\boldsymbol{w}_i$ and for every two states $x, y \in V_l$, for every $l \in [k]$, we have that $\lambda_i \boldsymbol{w}_i(x) = \lambda_i \boldsymbol{w}_i(y)$ since $\boldsymbol{w}_i$ is stepwise. Then, since $P\boldsymbol{w}_i = \lambda_i \boldsymbol{w}_i$, we have that

$$\sum_{j=1}^{k} \sum_{z \in V_j} P_{xz} \boldsymbol{v}_i(j) = (P\boldsymbol{w}_i)(x) = (P\boldsymbol{w}_i)(y) = \sum_{j=1}^{k} \sum_{z \in V_j} P_{yz} \boldsymbol{v}_i(j). \tag{7.11}$$

Thus $\sum_{j=1}^{k} \boldsymbol{v}_i(j) \sum_{z \in V_j} (P_{xz} - P_{yz}) = 0$ and then it follows that

$$\sum_{j=1}^{k} \boldsymbol{v}_i(j)\, \boldsymbol{u}_{xy}(j) = \langle \boldsymbol{u}_{xy}, \boldsymbol{v}_i \rangle = 0, \tag{7.12}$$

where $\boldsymbol{u}_{xy}(j) = \sum_{z \in V_j} (P_{xz} - P_{yz})$. Since the $\boldsymbol{v}_i$'s are k linearly independent vectors in a k-dimensional space, $\boldsymbol{u}_{xy}$ cannot be orthogonal to all of them and then it has to be the null vector, i.e. $\boldsymbol{u}_{xy}(j) = 0$ for all $j \in [k]$. This implies that P is ordinary lumpable, i.e. $\sum_{z \in V_j} P_{xz} = \sum_{z \in V_j} P_{yz}$. It is easy to verify that the eigenvalues and eigenvectors of $\widehat{P}$ are exactly $\lambda_i, \boldsymbol{v}_i$, with $i \in [k]$.

■

7.2 Theoretical analysis

7.2.1 Clustered graphs

For a volume-regular graph $G = (V, E, w)$ with n nodes and k-partition $\mathcal{V} = \{V_1, \ldots, V_k\}$ we name $N = \frac{\max_i |V_i|}{\min_i |V_i|}$ the ratio between the maximum and minimum sizes of the communities. In this section we prove the following result for volume-regular graphs.

Theorem 7.2.1. *Let $G = (V, E, w)$ be a connected clustered k-volume-regular graph with n nodes and k-partition $\mathcal{V} = \{V_1, \ldots, V_k\}$, with $k \leqslant \sqrt{n}$, maximum weighted degree $\Delta \leqslant poly(n)$, and $N = \mathcal{O}(\sqrt{k}/\Delta)$. If $\lambda_k > \frac{1}{2}$ and $(1-\lambda_2) \geqslant (\lambda_2 - \lambda_k)\Delta^{\frac{3}{2}} n^{1+c}$, for an arbitrarily-small positive constant c, then a time interval $[T_1, T_2]$ exists, with $T_1 = \mathcal{O}(\log n \,/\, \log(\lambda_k/\lambda_{k+1}))$ and $T_2 = \Omega(n^{c/3})$, such that for each time $t \in [T_1, T_2]$ the Averaging dynamics truncated at round t is a $(\mathcal{O}(n^{-\frac{1}{2}}), 1 - \Omega(1))$-community sensitive algorithm, w.h.p.*

In the remainder of this section, we first introduce further notation and then state the two main technical lemmas (Lemma 7.2.1 and Lemma 7.2.2), that will be used in the proof of Theorem 7.2.1, which concludes this section.

Let $G = (V, E, w)$ be a clustered k-volume-regular graph and, without loss of generality, let $V_1, \ldots, V_k$ be an arbitrary ordering of its communities. We introduce a family of stepwise vectors that generalize Fiedler vector [Fie89], namely

$$\left\{ \boldsymbol{\chi}_i = \sqrt{\frac{\hat{m}_i}{m_i}} \mathbf{1}_{V_i} - \sqrt{\frac{m_i}{\hat{m}_i}} \mathbf{1}_{\hat{V}_i} \;:\; i \in [k-1] \right\}, \tag{7.13}$$

where $\mathbf{1}_{V_i}$ is the indicator vector of the set V_i and, for convenience sake, we denoted by m_i the volume of the i-th community, $\hat{V}_i$ the set of all nodes in communities $i+1, \ldots, k$, and $\hat{m}_i$ the volume of $\hat{V}_i$, i.e., $m_i = \sum_{u \in V_i} \delta(u)$, $\hat{V}_i = \bigcup_{h=i+1}^{k} V_h$, and $\hat{m}_i = \sum_{h=i+1}^{k} m_h$. Note that vectors $\boldsymbol{\chi}_i$s are "stepwise" with respect to the communities of G (i.e., for every $i \in [k-1]$, $\boldsymbol{\chi}_i(u) = \boldsymbol{\chi}_i(v)$ whenever u and v belong to the same community).

Recall from Equation (7.2) that the initial state vector can be written as $\boldsymbol{x} = \sum_{i=1}^{n} \alpha_i \boldsymbol{v}_i$. Let $\boldsymbol{z} = \sum_{i=1}^{k} \alpha_i \boldsymbol{v}_i$ and note that $\boldsymbol{z} = \alpha_1 \mathbf{1} + \sum_{i=1}^{k-1} \gamma_i \boldsymbol{\chi}_i$ by applying Lemma 7.1.2 and because $Span\left(\{\mathbf{1}, \boldsymbol{\chi}_1, \ldots, \boldsymbol{\chi}_{k-1}\}\right) = Span\left(\{\mathbf{1}_{V_1}, \ldots, \mathbf{1}_{V_k}\}\right)$. Let us now define the vector $\boldsymbol{y} = \boldsymbol{z} - \alpha_1 \mathbf{1}$ or, equivalently,

$$\boldsymbol{y} = \sum_{i=1}^{k-1} \gamma_i \boldsymbol{\chi}_i, \text{ where } \gamma_i = \frac{\boldsymbol{x}^\intercal D \boldsymbol{\chi}_i}{\left\| D^{1/2} \boldsymbol{\chi}_i \right\|^2}. \tag{7.14}$$

Note that the coefficients γ_is are proportional to the length of the projection of the (inhomogeneously) contracted state vector on the (inhomogeneously) contracted, not anymore stepwise, $D^{\frac{1}{2}}\boldsymbol{\chi}_i$s and can be computed since the vectors in the family $\{D^{\frac{1}{2}}\mathbf{1}\} \cup \{D^{\frac{1}{2}}\boldsymbol{\chi}_i : i \in [k-1]\}$ are mutually orthogonal.[2]

The binary coloring of each node only depends on the difference of its state in two consecutive rounds (see Algorithm 3). Essentially in Lemma 7.2.1 we show that, under suitable assumptions on the transition matrix of a random walk on G, there exists a time window where the the difference of the state vector in two consecutive rounds, i.e., $\boldsymbol{x}^{(t)} - \boldsymbol{x}^{(t+1)}$, can be approximated by the previously defined vector $\boldsymbol{y}$ in a way that the sign of the two vectors is equal in any component, with high probability.

Lemma 7.2.1 (Sign of the difference)**.** *Let $G = (V, E, w)$ be a clustered k-volume-regular graph. If $\lambda_k > \frac{1}{2}$ and $(1 - \lambda_2) \geqslant (\lambda_2 - \lambda_k)\Delta^{\frac{3}{2}} n^{1+c}$, for an arbitrarily-small positive constant c, then a time interval $[T_1, T_2]$ exists, with $T_1 = \mathcal{O}\left(\frac{\log n}{\log(\lambda_k/\lambda_{k+1})}\right)$ and $T_2 = \Omega(n^{c/3})$, such that for each node $u \in V$ it holds that*

$$\operatorname{sgn}\left(\boldsymbol{x}^{(t)}(u) - \boldsymbol{x}^{(t+1)}(u)\right) = \operatorname{sgn}(\boldsymbol{y}(u)) \tag{7.15}$$

for every round $t \in [T_1, T_2]$ of the execution of the Averaging dynamics, w.h.p.

Sketch of proof. Recall from Equation (7.2) that the state vector at time t, i.e., $\boldsymbol{x}^{(t)}$, can be written as the sum of the first k stepwise vectors of P and of the remaining ones, namely

$$\boldsymbol{x}^{(t)} = \alpha_1 \mathbf{1} + \sum_{i=2}^{k} \lambda_i^t \alpha_i \boldsymbol{v}_i + \sum_{i=k+1}^{n} \lambda_i^t \alpha_i \boldsymbol{v}_i = \alpha_1 \mathbf{1} + \boldsymbol{c}^{(t)} + \boldsymbol{e}^{(t)}, \tag{7.16}$$

where we call $\boldsymbol{c}^{(t)} := \sum_{i=2}^{k} \lambda_i^t \alpha_i \boldsymbol{v}_i$ the *core contribution* and $\boldsymbol{e}^{(t)} := \sum_{i=k+1}^{n} \lambda_i^t \alpha_i \boldsymbol{v}_i$ the *error contribution*. If we look at the difference of the state vector between two consecutive rounds, for each node $u \in V$, the first term cancels out being constant over time and we get $\boldsymbol{x}^{(t)}(u) - \boldsymbol{x}^{(t+1)}(u) = \boldsymbol{c}^{(t)}(u) - \boldsymbol{c}^{(t+1)}(u) + \boldsymbol{e}^{(t)}(u) - \boldsymbol{e}^{(t+1)}(u)$. Note that the sign of the difference between two consecutive states of each node

[2] The mutual orthogonality of the vectors, including $D^{\frac{1}{2}}\mathbf{1}$, is also one of the reasons why other "simpler" families of stepwise vectors, e.g., the indicator vectors of the communities, are not used instead.

$u \in V$ is determined by the difference of the core contributions during the two consecutive rounds, i.e., $\boldsymbol{c}^{(t)}(u) - \boldsymbol{c}^{(t+1)}(u)$, whenever

$$\left|\boldsymbol{c}^{(t)}(u) - \boldsymbol{c}^{(t+1)}(u)\right| > \left|\boldsymbol{e}^{(t)}(u) - \boldsymbol{e}^{(t+1)}(u)\right|. \tag{7.17}$$

To find the conditions on t that make Eq. (7.17) hold, we give a bound to both the left and right hand side of the inequality. In detail:

1. We know from Lemma 7.2.4 that $\left|\boldsymbol{c}^{(t)}(u) - \boldsymbol{c}^{(t+1)}(u)\right| > \frac{1}{2}\lambda_k^t(1-\lambda_2)\left|\boldsymbol{y}(u)\right|$ for every $u \in V$ and for every time $t < T_2$, where $T_2 = \Omega(n^{\frac{c}{3}})$, since by hypothesis $\lambda_k > \frac{1}{2}$ and $(1-\lambda_2) \geqslant (\lambda_2 - \lambda_k)\Delta^{\frac{3}{2}}n^{1+c}$.

2. We know from Lemma 7.2.5 that $|\boldsymbol{e}^{(t)}(u)| \leqslant \lambda_{k+1}^t\sqrt{\Delta n}$, for every $u \in V$, and thus it follows that $\left|\boldsymbol{e}^{(t)}(u) - \boldsymbol{e}^{(t+1)}(u)\right| \leqslant \left|\boldsymbol{e}^{(t)}(u)\right| + \left|\boldsymbol{e}^{(t+1)}(u)\right| \leqslant 2\lambda_{k+1}^t\sqrt{\Delta n}$.

Combining Lemma 7.2.4 and Lemma 7.2.5, we get that if the following inequality holds, i.e.,

$$\frac{1}{2}\lambda_k^t(1-\lambda_2)\left|\boldsymbol{y}(u)\right| > 2\lambda_{k+1}^t\sqrt{\Delta n}, \tag{7.18}$$

then also Equation (7.17) holds. By moving the terms dependent from t on the left hand side and by taking the logarithm of both sides, we can finally find the conditions on t such that Equation (7.18) is satisfied, i.e., all times $t > T_1$ where

$$T_1 = \log\left(\frac{4\sqrt{\Delta n}}{(1-\lambda_2)\left|\boldsymbol{y}(u)\right|}\right) \cdot \frac{1}{\log\left(\frac{\lambda_k}{\lambda_{k+1}}\right)}. \tag{7.19}$$

Note that $T_1 = \mathcal{O}(\log n \,/\, \log(\frac{\lambda_k}{\lambda_{k+1}}))$ and that $T_1 = \mathcal{O}(\log n)$ when $\frac{\lambda_k}{\lambda_{k+1}} = \Omega(1)$. In fact:

1. We know by hypothesis that the maximum weighted degree of a node is at most polynomial in n, i.e., $\Delta \leqslant poly(n)$.

2. We know from the Cheeger's inequality for weighted graphs (Theorem D.2.1) the relation between the spectral gap and the Cheeger's constant of G, i.e., $1-\lambda_2 \geqslant \frac{1}{2\Delta n}$, given that $1-\lambda_2 \geqslant \frac{h_G^2}{2} \geqslant \frac{1}{2\Delta n}$.

3. We know from Lemma 7.2.3 that the length of the projection of the state vector on the stepwise vectors is not too small, i.e., $|\boldsymbol{y}(u)| \geqslant \frac{k}{\Delta n}$, w.h.p.

Since Lemma 7.2.4 holds for every time $t < T_2$, we conclude that there exists a time window $[T_1, T_2]$ such that, for every time $t \in [T_1, T_2]$ of the Averaging dynamics, it holds that $\operatorname{sgn}(\boldsymbol{x}^{(t)}(u) - \boldsymbol{x}^{(t+1)}(u)) = \operatorname{sgn}(\boldsymbol{c}^{(t)}(u) - \boldsymbol{c}^{(t+1)}(u))$, with high probability. Moreover, Lemma 7.2.4 tells us that $\operatorname{sgn}(\boldsymbol{c}^{(t)}(u) - \boldsymbol{c}^{(t+1)}(u)) = \operatorname{sgn}(\boldsymbol{y}(u))$, for every $u \in V$ and for every $t \in [T_1, T_2]$. Thus, $\operatorname{sgn}(\boldsymbol{x}^{(t)}(u) - \boldsymbol{x}^{(t+1)}(u)) = \operatorname{sgn}(\boldsymbol{y}(u))$, concluding the proof. ■

In Lemma 7.2.2, instead, we show that with some constant probability (i.e., independent from the number of nodes n) the first two "steps" of the vector $\boldsymbol{y}$ have different signs, i.e., the sign can be considered as a criterion to distinguish the first two communities.

Lemma 7.2.2 (Different communities, different signs)**.** *Let $G = (V, E, w)$ be a clustered k-volume-regular graph with maximum weighted degree $\Delta \leqslant poly(n)$ and $N = \mathcal{O}(\sqrt{k}/\Delta)$. For each pair of nodes $u \in V_i$ and $v \in V_j$, with $i \neq j$, it holds that*

$$\mathbf{P}\left(\operatorname{sgn}(\boldsymbol{y}(u)) \neq \operatorname{sgn}(\boldsymbol{y}(v))\right) = \Omega(1). \tag{7.20}$$

Proof. Since the ordering of the communities (and consequent definition of the $\boldsymbol{\chi}_i$'s) is completely arbitrary, we can without loss of generality assume $i = 1$ and $j = 2$. From Lemma 7.2.1 we have that $\operatorname{sgn}(\boldsymbol{x}^{(t)}(u) - \boldsymbol{x}^{(t+1)}(u)) = \operatorname{sgn}(\boldsymbol{y}(u))$, for every $u \in V$, during a time interval $[T_1, T_2]$, w.h.p. Let us define $X(V_i) := \sum_{w \in V_i} \delta(w)\boldsymbol{x}(w)$.

Note that $\boldsymbol{y}(u) = \gamma_1 \boldsymbol{\chi}_1(u)$ and $\boldsymbol{y}(v) = \gamma_1 \boldsymbol{\chi}_1(v) + \gamma_2 \boldsymbol{\chi}_2(v)$, since the other terms of the $\boldsymbol{\chi}_i$s are equal to 0 on the components relative to u and v. Thus, with some algebra, we get

$$\boldsymbol{y}(u) = \frac{1}{m}\left[\frac{\hat{m}_1}{m_1} X(V_1) - X(V_2) - X(\hat{V}_2)\right], \tag{7.21}$$

$$\boldsymbol{y}(v) = \frac{1}{m}\left[\frac{m_1 m_2 + m\hat{m}_2}{\hat{m}_1 m_2} X(V_2) - X(V_1) - X(\hat{V}_2)\right]. \tag{7.22}$$

Note that, by linearity of expectation, $\mathbf{E}[X(V_i)] = 0$. Moreover, since the terms $\boldsymbol{x}(w)$s are independent Rademacher random variables, we can write the standard deviation of $X(V_i)$ as

$$\sigma(X(V_i)) = \sqrt{\sum_{w \in V_i} \sigma^2(\boldsymbol{x}(w))} = \sqrt{\sum_{w \in V_i} \left(\mathbf{E}\left[\delta(w)^2 \boldsymbol{x}(w)^2\right] - \mathbf{E}\left[\delta(w)\boldsymbol{x}(w)\right]^2\right)} \tag{7.23}$$

$$= \sqrt{\sum_{w \in V_i} \delta(w)^2}. \tag{7.24}$$

Then we can upper and lower bound the standard deviation $\sigma(X(V_i))$ getting $\frac{m_i}{\sqrt{|V_i|}} \leqslant \sigma(X(V_i)) \leqslant \Delta\sqrt{|V_i|}$, where the lower bound follows from $\|\boldsymbol{d}\|_2 \geqslant \|\boldsymbol{d}\|_1 / \sqrt{|V_i|}$, where $\boldsymbol{d_i}$ is the vector of weighted degrees of nodes in community V_i, and for the upper bound we used that $\delta(w) \leqslant \Delta$, for each $w \in V$.

Let us now define the following three events:

1. E_1: $X(V_1) \geqslant \sigma(X(V_1))$. Note that E_1 implies

$$X(V_1) \geqslant \frac{m_1}{\sqrt{|V_1|}} \geqslant \frac{\min_i m_i}{\sqrt{\max_i |V_i|}}; \tag{7.25}$$

2. E_2: $X(V_2) \leqslant -\sigma(X(V_2))$. Note that E_2 implies

$$X(V_2) \leqslant -\frac{m_2}{\sqrt{|V_2|}} \leqslant -\frac{\min_i m_i}{\sqrt{\max_i |V_i|}}; \tag{7.26}$$

3. E_3: $0 \leqslant X(\hat{V}_2) \leqslant \varepsilon\sigma(X(\hat{V}_2))$. Note that E_3 implies

$$0 \leqslant X(\hat{V}_2) \leqslant \varepsilon\Delta\sqrt{\sum_{i=3}^{k}|V_i|} \leqslant \varepsilon\Delta\sqrt{k\max_i|V_i|}, \tag{7.27}$$

with ε a suitable positive constant.

Note that E_1, E_2, E_3 all hold with non-negligible, constant probability since essentially require that the outcome of a Binomial random variable minus its expected value is greater (E_1, E_2) or less (E_3) than a standard deviation. When E_1, E_2, E_3 are true it holds that $\boldsymbol{y}(v) < 0$; as for $\boldsymbol{y}(u)$ we have that

$$\frac{\hat{m}_1}{m_1}X(V_1) - X(V_2) - X(\hat{V}_2) \geqslant \frac{\hat{m}_1}{m_1}\sigma(X(V_1)) + \sigma(X(V_2)) - \varepsilon\sigma(X(\hat{V}_2)) \tag{7.28}$$

$$\geqslant \frac{k\min_i|V_i|}{\sqrt{\max_i|V_i|}} - \varepsilon\Delta\sqrt{k\max_i|V_i|}. \tag{7.29}$$

The previous inequality is greater than 0 whenever $\varepsilon < \frac{\sqrt{k}}{\Delta N}$. By hypothesis $\Delta N = \mathcal{O}(\sqrt{k})$ and thus $\frac{\sqrt{k}}{\Delta N} = \Omega(1)$, i.e., there is an $\varepsilon = \Omega(1)$ such that $\boldsymbol{y}(u) > 0$.

By approximating the random variables with Gaussian ones and using Berry-Esseen's theorem (Theorem E.3.1), it is possible to show that all three events have probability at least constant; moreover, being the events independent, also $\mathbf{P}(E_1, E_2, E_3)$ is constant. ■

Proof of Theorem 7.2.1. The proof proceeds by showing that the binary labeling of the nodes of G produced by the Averaging dynamics during the time window $[T_1, T_2]$ is such that the two conditions required by the definition of (ε, δ)-community sensitive algorithm (Definition 7.1.1) are met. The first condition follows directly from Lemma 7.2.1 and from the fact that $\boldsymbol{y}$ is a "stepwise" vector, with $\varepsilon = \mathcal{O}(n^{-\frac{1}{2}})$ (see Lemma 7.2.3 for details on the probability). The second condition follows directly from Lemma 7.2.2. ■

Full proof of Lemma 7.2.1

In this section we prove all technical lemmas used in the proof of Lemma 7.2.1. The main result in Section 7.2.1 is Lemma 7.2.3 in which we show that every component of $\boldsymbol{y}$, i.e., the projection of the (inhomogeneously) contracted initial state vector $D^{\frac{1}{2}}\boldsymbol{x}$ on the (inhomogeneously) contracted vectors $D^{\frac{1}{2}}\boldsymbol{\chi}_i$s, is not too small, w.h.p. This result is shown first since it is used in other lemmas in this section. In Section 7.2.1 we provide a lower bound on the core contribution of the state vector between two consecutive time steps. Moreover, we show that the sign of a node depends only on the sign of $\boldsymbol{y}$. Finally, in Section 7.2.1 we upper bound the error (i.e., the part of the state vector in the eigenspace of eigenvalue $\lambda_{k+1}, \ldots, \lambda_n$).

Length of the projection of the state vector

Claim 7.2.1. *Let* $\alpha(u,v) = \sum_{i=1}^{k-1} \frac{\boldsymbol{\chi}_i(u)\boldsymbol{\chi}_i(v)}{\hat{m}_{i-1}}$. *For every pair of nodes* $u, v \in V$ *it holds that*

$$\min_{u,v\in V} |\alpha(u,v)| \geqslant \frac{k}{\Delta n}. \tag{7.30}$$

Proof. Let $u \in V_l$ and $v \in V_h$, for some $l, h \in [k]$. We divide the proof in two cases. First, we assume that $l = h$, then we handle the case $l \neq h$. Without loss of generality, we assume $m_1 \leqslant \ldots \leqslant m_k$ and consequently $m = \hat{m}_0 \geqslant \hat{m}_1 \geqslant \ldots \geqslant \hat{m}_{k-1} = m_k$.

- Let us suppose $l = h$. Then

$$\begin{aligned}
\min_{u,v\in V} |\alpha(u,v)| &= \min_{u,v\in V} \left(\sum_{i<\min\{h,l\}} \frac{m_i}{\hat{m}_i \hat{m}_{i-1}} + \frac{\hat{m}_l}{m_l \hat{m}_{l-1}} \right) && (7.31)\\
&\geqslant \frac{\hat{m}_1}{m_1 \hat{m}_0} && (7.32)\\
&= \frac{m_2 + \ldots + m_k}{m_1 m} && (7.33)\\
&\geqslant \frac{k}{m} && (7.34)\\
&\geqslant \frac{k}{n \max_v \delta(v)} && (7.35)\\
&\geqslant \frac{k}{\Delta n}. && (7.36)
\end{aligned}$$

- Let us suppose $l \neq h$. Note that, in this case, $\alpha(u,v) = \sum_{i<\min\{h,l\}} \frac{m_i}{\hat{m}_i \hat{m}_{i-1}} - 1 < 0$. In fact,

$$\begin{aligned}
\sum_{i<\min\{h,l\}} \frac{m_i}{\hat{m}_i \hat{m}_{i-1}} &= \sum_{i<\min\{h,l\}} \frac{m_i}{\left(\sum_{j=i+1}^{k} m_j\right)\left(\sum_{j=i}^{k} m_j\right)} && (7.37)\\
&\overset{(a)}{\leqslant} \sum_{i<\min\{h,l\}} \frac{m_i}{(k-i)(k-i+1)m_i^2} && (7.38)\\
&= \sum_{i<\min\{h,l\}} \frac{1}{(k-i)(k-i+1)m_i} && (7.39)\\
&\leqslant \sum_{i<\min\{h,l\}} \frac{1}{(k-i)(k-i+1)} && (7.40)\\
&\leqslant \sum_{j=1}^{k} \frac{1}{j(j+1)} && (7.41)\\
&= \sum_{j=1}^{k-1} \frac{1}{j} - \sum_{j=1}^{k-1} \frac{1}{j+1} && (7.42)\\
&= 1 - \frac{1}{k} < 1, && (7.43)
\end{aligned}$$

where in (a) we use the assumption on the ordering of the volumes of the communities, i.e., $m_i \leqslant m_j$ for every $i \leqslant j$. Since $\alpha(u,v) < 0$, we have that

$$|\alpha(u,v)| = 1 - \sum_{i<\min\{h,l\}} \frac{m_i}{\hat{m}_i \hat{m}_{i-1}}. \tag{7.44}$$

Thus,

$$\min_{u,v\in V} |\alpha(u,v)| = 1 - \max_{u,v\in V} \left(\sum_{i<\min\{h,l\}} \frac{m_i}{\hat{m}_i \hat{m}_{i-1}} \right) \tag{7.45}$$

$$\geqslant 1 - \frac{(k-2)m_k}{m_k^2} \tag{7.46}$$

$$= 1 - \frac{k-2}{m_k}. \tag{7.47}$$

Note that $m_k \geqslant \frac{n}{k}$ and, given that $k \leqslant \sqrt{n}$, we get $m_k \geqslant k$. Thus

$$1 - \frac{k-2}{m_k} \geqslant \frac{2}{k} \geqslant \frac{k}{\Delta n}. \tag{7.48}$$

■

Lemma 7.2.3 (Length of the projection of the state vector)**.** *For every $u \in V$, it holds that*

$$\mathbf{P}\left(|\boldsymbol{y}(u)| \geqslant \frac{k}{\Delta n} \right) \geqslant 1 - \mathcal{O}\left(\frac{1}{\sqrt{n}} \right). \tag{7.49}$$

Proof. Let us write $\boldsymbol{y}(u) = \sum_{i=1}^{k-1} \gamma_i \boldsymbol{\chi}_i(u)$ in terms of $\boldsymbol{x}$. Recall that

$$\gamma_i = \frac{\boldsymbol{x}^\intercal D \boldsymbol{\chi}_i}{\left\| D^{1/2} \boldsymbol{\chi}_i \right\|^2}, \quad \boldsymbol{\chi}_i = \sqrt{\frac{\hat{m}_i}{m_i}} \mathbf{1}_{V_i} - \sqrt{\frac{m_i}{\hat{m}_i}} \mathbf{1}_{\hat{V}_i}. \tag{7.50}$$

Thus, we get

$$\left\| D^{1/2} \boldsymbol{\chi}_i \right\|^2 = \frac{\hat{m}_i}{m_i} \sum_{v\in V_i} \delta(v) + \frac{m_i}{\hat{m}_i} \sum_{v\in \hat{V}_i} \delta(v) = \hat{m}_i + m_i = \hat{m}_{i-1}, \tag{7.51}$$

where $\hat{m}_0 := m = \sum_{v\in V} \delta(v)$. Now, we can rewrite $\boldsymbol{y}(u)$ as

$$\boldsymbol{y}(u) = \sum_{i=1}^{k-1} \gamma_i \boldsymbol{\chi}_i(u) \tag{7.52}$$

$$= \sum_{i=1}^{k-1} \frac{\boldsymbol{x}^\intercal D \boldsymbol{\chi}_i}{\hat{m}_{i-1}} \boldsymbol{\chi}_i(u) \tag{7.53}$$

$$= \sum_{i=1}^{k-1} \left(\sum_{v\in V} \frac{\delta(v)\boldsymbol{x}(v)\boldsymbol{\chi}_i(v)}{\hat{m}_{i-1}} \right) \boldsymbol{\chi}_i(u) \tag{7.54}$$

$$= \sum_{v\in V} \left(\sum_{i=1}^{k-1} \frac{\boldsymbol{\chi}_i(u)\boldsymbol{\chi}_i(v)}{\hat{m}_{i-1}} \right) \delta(v)\boldsymbol{x}(v) \tag{7.55}$$

$$= \sum_{v \in V} \alpha(u,v)\delta(v)\boldsymbol{x}(v), \tag{7.56}$$

where

$$\alpha(u,v) := \sum_{i=1}^{k-1} \frac{\boldsymbol{\chi}_i(u)\boldsymbol{\chi}_i(v)}{\hat{m}_{i-1}}. \tag{7.57}$$

Note that, for every $u \in V_l$ and $v \in V_h$, with $l, h \in [k]$, we have

$$\boldsymbol{\chi}_i(u)\boldsymbol{\chi}_i(v) = \begin{cases} \frac{m_i}{\hat{m}_i} & \text{if } i < \min(l,h) \\ \frac{\hat{m}_i}{m_i} & \text{if } i = \min(l,h) \text{ and } l = h \\ -1 & \text{if } i = \min(l,h) \text{ and } l \neq h \\ 0 & \text{if } i > \min(l,h). \end{cases} \tag{7.58}$$

Thus, $\alpha(u,v)$ is equal to

$$\alpha(u,v) = \sum_{i=1}^{k-1} \frac{\boldsymbol{\chi}_i(u)\boldsymbol{\chi}_i(v)}{\hat{m}_{i-1}} = \begin{cases} \sum_{i<\min\{h,l\}} \frac{m_i}{\hat{m}_i \hat{m}_{i-1}} + \frac{\hat{m}_l}{m_l \hat{m}_{l-1}} & \text{if } h = l, \\ \sum_{i<\min\{h,l\}} \frac{m_i}{\hat{m}_i \hat{m}_{i-1}} - 1 & \text{if } h \neq l. \end{cases} \tag{7.59}$$

We apply Theorem E.3.3 and Claim 7.2.1 to prove that the length of the projection of the state vector $\boldsymbol{x}$ on $\{\boldsymbol{\chi}_i : i \in [k]\}$ is not too small, w.h.p.

$$\mathbf{P}\left(|\boldsymbol{y}(u)| \leqslant \frac{k}{\Delta n}\right) = \mathbf{P}\left(\left|\sum_{v \in V} \alpha(u,v)\delta(v)\boldsymbol{x}(v)\right| \leqslant \frac{k}{\Delta n}\right) \tag{7.60}$$

$$= \mathbf{P}\left(\left|\sum_{v \in V} \frac{\alpha(u,v)}{\min_{u,v} |\alpha(u,v)|}\delta(v)\boldsymbol{x}(v)\right| \leqslant \frac{k}{\Delta n \min_{u,v} |\alpha(u,v)|}\right) \tag{7.61}$$

$$\overset{(a)}{\leqslant} \mathbf{P}\left(\left|\sum_{v \in V} \frac{\alpha(u,v)}{\min_{u,v} |\alpha(u,v)|}\delta(v)\boldsymbol{x}(v)\right| \leqslant 1\right) \tag{7.62}$$

$$\overset{(b)}{\leqslant} \mathcal{O}\left(\frac{1}{\sqrt{n}}\right), \tag{7.63}$$

where in (a) we use Claim 7.2.1 to upper bound with 1 the r.h.s. term in the probability; in (b) we can apply Theorem E.3.3 given that $\min_v \delta(v) = 1$ and that $\left|\frac{\alpha(u,v)}{\min_{u,v} |\alpha(u,v)|}\right| \geqslant 1$. ■

Lower bound on the core contribution

In order to prove the main result of this section (Lemma 7.2.4) we first provide upper and lower bounds on $\boldsymbol{c}^{(t)}(u)$ in Claim 7.2.2; then, in Claim 7.2.3, we use the result of Claim 7.2.2 to bound $\boldsymbol{c}^{(t)}(u) - \boldsymbol{c}^{(t+1)}(u)$.

Claim 7.2.2. *Let $\boldsymbol{c}^{(t)} = \sum_{i=2}^{k} \lambda_i^t \alpha_i \boldsymbol{v}_i$. For every $u \in V$ it holds that*

$$\boldsymbol{c}^{(t)}(u) \geqslant \lambda_k^t \sum_{i=2}^{k} \alpha_i \boldsymbol{v}_i(u) + t\lambda_2^{t-1}(\lambda_2 - \lambda_k) \sum_{i:\alpha_i \boldsymbol{v}_i(u) < 0} \alpha_i \boldsymbol{v}_i(u), \tag{7.64}$$

$$\boldsymbol{c}^{(t)}(u) \leqslant \lambda_k^t \sum_{i=2}^{k} \alpha_i \boldsymbol{v}_i(u) + t\lambda_2^{t-1}(\lambda_2 - \lambda_k) \sum_{i:\alpha_i \boldsymbol{v}_i(u)>0} \alpha_i \boldsymbol{v}_i(u). \tag{7.65}$$

Proof. Let us start with the lower bound.

$$\begin{aligned}
\boldsymbol{c}^{(t)}(u) &= \sum_{i=2}^{k} \lambda_i^t \alpha_i \boldsymbol{v}_i(u) && (7.66)\\
&= \sum_{i:\alpha_i \boldsymbol{v}_i(u)>0} \lambda_i^t \alpha_i \boldsymbol{v}_i(u) + \sum_{i:\alpha_i \boldsymbol{v}_i(u)<0} \lambda_i^t \alpha_i \boldsymbol{v}_i(u) && (7.67)\\
&\geqslant \lambda_k \sum_{i:\alpha_i \boldsymbol{v}_i(u)>0} \lambda_i^{t-1} \alpha_i \boldsymbol{v}_i(u) + \lambda_2 \sum_{i:\alpha_i \boldsymbol{v}_i(u)<0} \lambda_i^{t-1} \alpha_i \boldsymbol{v}_i(u) && (7.68)\\
&\overset{(a)}{=} \lambda_k \sum_{i=2}^{k} \lambda_i^{t-1} \alpha_i \boldsymbol{v}_i(u) + (\lambda_2 - \lambda_k) \sum_{i:\alpha_i \boldsymbol{v}_i(u)<0} \lambda_i^{t-1} \alpha_i \boldsymbol{v}_i(u) && (7.69)\\
&\overset{(b)}{=} \lambda_k \left[\lambda_k \sum_{i=2}^{k} \lambda_i^{t-2} \alpha_i \boldsymbol{v}_i(u) + (\lambda_2 - \lambda_k) \sum_{i:\alpha_i \boldsymbol{v}_i(u)<0} \lambda_i^{t-2} \alpha_i \boldsymbol{v}_i(u) \right]\\
&\quad + (\lambda_2 - \lambda_k)\lambda_2^{t-1} \sum_{i:\alpha_i \boldsymbol{v}_i(u)<0} \alpha_i \boldsymbol{v}_i(u) && (7.70)\\
&= \lambda_k^2 \sum_{i=2}^{k} \lambda_i^{t-2} \alpha_i \boldsymbol{v}_i(u) + \lambda_k(\lambda_2 - \lambda_k) \sum_{i:\alpha_i \boldsymbol{v}_i(u)<0} \lambda_i^{t-2} \alpha_i \boldsymbol{v}_i(u)\\
&\quad + (\lambda_2 - \lambda_k)\lambda_2^{t-1} \sum_{i:\alpha_i \boldsymbol{v}_i(u)<0} \alpha_i \boldsymbol{v}_i(u) && (7.71)\\
&\geqslant \lambda_k^2 \sum_{i=2}^{k} \lambda_i^{t-2} \alpha_i \boldsymbol{v}_i(u) + \lambda_k \lambda_2^{t-2}(\lambda_2 - \lambda_k) \sum_{i:\alpha_i \boldsymbol{v}_i(u)<0} \alpha_i \boldsymbol{v}_i(u)\\
&\quad + (\lambda_2 - \lambda_k)\lambda_2^{t-1} \sum_{i:\alpha_i \boldsymbol{v}_i(u)<0} \alpha_i \boldsymbol{v}_i(u) && (7.72)\\
&= \lambda_k^2 \sum_{i=2}^{k} \lambda_i^{t-2} \alpha_i \boldsymbol{v}_i(u) + (\lambda_k \lambda_2^{t-2} + \lambda_2^{t-1})(\lambda_2 - \lambda_k) \sum_{i:\alpha_i \boldsymbol{v}_i(u)<0} \alpha_i \boldsymbol{v}_i(u) && (7.73)\\
&\geqslant \lambda_k^2 \sum_{i=2}^{k} \lambda_i^{t-2} \alpha_i \boldsymbol{v}_i(u) + 2\lambda_2^{t-1}(\lambda_2 - \lambda_k) \sum_{i:\alpha_i \boldsymbol{v}_i(u)<0} \alpha_i \boldsymbol{v}_i(u) && (7.74)\\
&\geqslant \dots && (7.75)\\
&\geqslant \lambda_k^t \sum_{i=2}^{k} \alpha_i \boldsymbol{v}_i(u) + t\lambda_2^{t-1}(\lambda_2 - \lambda_k) \sum_{i:\alpha_i \boldsymbol{v}_i(u)<0} \alpha_i \boldsymbol{v}_i(u), && (7.76)
\end{aligned}$$

where in (a) we add and subtract $\lambda_k \sum_{i:\alpha_i \boldsymbol{v}_i(u)<0} \lambda_i^{t-1} \alpha_i \boldsymbol{v}_i(u)$; in (b) we iterate the same reasoning on the first term only.

As for the upper bound, similarly to the previous case, we get

$$\boldsymbol{c}^{(t)}(u) \leqslant \lambda_k^t \sum_{i=2}^{k} \alpha_i \boldsymbol{v}_i(u) + t\lambda_2^{t-1}(\lambda_2 - \lambda_k) \sum_{i:\alpha_i \boldsymbol{v}_i(u)>0} \alpha_i \boldsymbol{v}_i(u). \tag{7.77}$$

■

Here we use the result of Claim 7.2.2 to give upper and lower bounds on the difference between the *core contribution* in two consecutive rounds.

Claim 7.2.3. *Let* $\boldsymbol{c}^{(t)} = \sum_{i=2}^{k} \lambda_i^t \alpha_i \boldsymbol{v}_i$ *and let* $\lambda_2 > \lambda_k > \frac{1}{2}$. *For every* $u \in V$, *it holds that*

$$\boldsymbol{c}^{(t)}(u) - \boldsymbol{c}^{(t+1)}(u) \geqslant \lambda_k^t(1-\lambda_2)\sum_{i=2}^{k} \alpha_i \boldsymbol{v}_i(u) + (t+1)\lambda_2^t(\lambda_2-\lambda_k) \sum_{i:\alpha_i \boldsymbol{v}_i(u)<0} \alpha_i \boldsymbol{v}_i(u),$$

$$\boldsymbol{c}^{(t)}(u) - \boldsymbol{c}^{(t+1)}(u) \leqslant \lambda_k^t(1-\lambda_2)\sum_{i=2}^{k} \alpha_i \boldsymbol{v}_i(u) + (t+1)\lambda_2^t(\lambda_2-\lambda_k) \sum_{i:\alpha_i \boldsymbol{v}_i(u)>0} \alpha_i \boldsymbol{v}_i(u).$$

Proof. Let us start with the lower bound.

$$\begin{aligned}
&\boldsymbol{c}^{(t)}(u) - \boldsymbol{c}^{(t+1)}(u) \\
&\quad= \sum_{i=2}^{k} \lambda_i^t(1-\lambda_i)\alpha_i \boldsymbol{v}_i(u) && (7.78)\\
&\quad= \sum_{i:\alpha_i \boldsymbol{v}_i(u)>0} \lambda_i^t(1-\lambda_i)\alpha_i \boldsymbol{v}_i(u) + \sum_{i:\alpha_i \boldsymbol{v}_i(u)<0} \lambda_i^t(1-\lambda_i)\alpha_i \boldsymbol{v}_i(u) && (7.79)\\
&\quad\geqslant (1-\lambda_2) \sum_{i:\alpha_i \boldsymbol{v}_i(u)>0} \lambda_i^t \alpha_i \boldsymbol{v}_i(u) + (1-\lambda_k) \sum_{i:\alpha_i \boldsymbol{v}_i(u)<0} \lambda_i^t \alpha_i \boldsymbol{v}_i(u) && (7.80)\\
&\quad= (1-\lambda_2)\sum_{i=2}^{k} \lambda_i^t \alpha_i \boldsymbol{v}_i(u) + (\lambda_2-\lambda_k) \sum_{i:\alpha_i \boldsymbol{v}_i(u)<0} \lambda_i^t \alpha_i \boldsymbol{v}_i(u) && (7.81)\\
&\quad\overset{(a)}{\geqslant} (1-\lambda_2)\left[\lambda_k^t \sum_{i=2}^{k} \alpha_i \boldsymbol{v}_i(u) + t\lambda_2^{t-1}(\lambda_2-\lambda_k) \sum_{i:\alpha_i \boldsymbol{v}_i(u)<0} \alpha_i \boldsymbol{v}_i(u)\right] \\
&\qquad + \lambda_2^t(\lambda_2-\lambda_k) \sum_{i:\alpha_i \boldsymbol{v}_i(u)<0} \alpha_i \boldsymbol{v}_i(u) && (7.82)\\
&\quad= \lambda_k^t(1-\lambda_2)\sum_{i=2}^{k} \alpha_i \boldsymbol{v}_i(u) \\
&\qquad + (\lambda_2-\lambda_k)[\lambda_2^{t-1}(1-\lambda_2)t + \lambda_2^t)] \sum_{i:\alpha_i \boldsymbol{v}_i(u)<0} \alpha_i \boldsymbol{v}_i(u) && (7.83)\\
&\quad\overset{(b)}{\geqslant} \lambda_k^t(1-\lambda_2)\sum_{i=2}^{k} \alpha_i \boldsymbol{v}_i(u) + (t+1)\lambda_2^t(\lambda_2-\lambda_k) \sum_{i:\alpha_i \boldsymbol{v}_i(u)<0} \alpha_i \boldsymbol{v}_i(u) && (7.84)
\end{aligned}$$

where in (a) we use Claim 7.2.2 and $\lambda_i^t \leqslant \lambda_2^t$; in (b) we use the hypothesis on λ_2.

As for the upper bound, similarly to the previous case, we get

$$\begin{aligned}
&\boldsymbol{c}^{(t)}(u) - \boldsymbol{c}^{(t+1)}(u) \\
&\quad\leqslant \lambda_k^t(1-\lambda_2)\sum_{i=2}^{k} \alpha_i \boldsymbol{v}_i(u) + (t+1)\lambda_2^t(\lambda_2-\lambda_k) \sum_{i:\alpha_i \boldsymbol{v}_i(u)>0} \alpha_i \boldsymbol{v}_i(u). && (7.85)
\end{aligned}$$

■

The proof of Lemma 7.2.4 requires one extra claim about the coefficients β_i, i.e., the ones such that $\alpha_i \boldsymbol{v}_i = \beta_i D^{\frac{1}{2}} \boldsymbol{w}_i$. This last bound is shown in Claim 7.2.4.

Claim 7.2.4. *Let $\boldsymbol{x} \in \{-1, 1\}^n$ be a Rademacher random vector. Let $D \in \mathbb{R}^{n \times n}$ be a positive diagonal matrix with maximum element $\Delta = \max_i D_{ii}$ and let $\boldsymbol{w} \in \mathbb{R}^n$ be a vector such that $\|\boldsymbol{w}\|_2 = 1$. Let $\beta = \langle \boldsymbol{x}, D^{\frac{1}{2}} \boldsymbol{w} \rangle$. It holds that $|\beta| \leqslant \sqrt{\Delta \log n}$, with high probability.*

Proof. Note that β is a weighted sum of Rademacher random variables with i-th coefficient equal to $(D^{\frac{1}{2}} \boldsymbol{w})(i)$ and that $\|D^{\frac{1}{2}} \boldsymbol{w}\|_2 = \sqrt{\sum_{i=1}^n \delta(i) \boldsymbol{w}(i)^2} \leqslant \sqrt{\Delta}$, since by hypothesis $\|\boldsymbol{w}\|_2 = 1$ and thus $\|D^{\frac{1}{2}} \boldsymbol{w}\|_2^2$ is a convex combination of the diagonal elements of D. Let $t = \sqrt{\log n}$; by applying Theorem E.3.4 we get

$$\mathbf{P}\left(|\beta| > \sqrt{\Delta \log n}\right) \leqslant \mathbf{P}\left(|\beta_i| > t \|D^{\frac{1}{2}} \boldsymbol{w}\|_2\right) \leqslant 2e^{-\frac{\log n}{2}} = \mathcal{O}\left(\frac{1}{n}\right). \tag{7.86}$$

Thus $|\beta| \leqslant \sqrt{\Delta \log n}$, with high probability. ∎

We now state and prove Lemma 7.2.4.

Lemma 7.2.4 (Lower bound on the core contribution)**.** *Let $\boldsymbol{c}^{(t)} = \sum_{i=2}^k \lambda_i^t \alpha_i \boldsymbol{v}_i$. Let $\lambda_k > \frac{1}{2}$ and $\frac{1-\lambda_2}{\lambda_2 - \lambda_k} \geqslant \Delta^{\frac{3}{2}} n^{1+c}$, for some positive constant c. For every $u \in V$ and for every time $t < T_2$, such that $T_2 = \Omega(n^{c/3})$, the two following conditions hold, w.h.p.:*

- $$\left|\boldsymbol{c}^{(t)}(u) - \boldsymbol{c}^{(t+1)}(u)\right| \geqslant \frac{1}{2}\lambda_k^t (1 - \lambda_2) \left|\boldsymbol{y}(u)\right|; \tag{7.87}$$
- $$\operatorname{sgn}(\boldsymbol{c}^{(t)}(u) - \boldsymbol{c}^{(t+1)}(u)) = \operatorname{sgn}(\boldsymbol{y}(u)) \tag{7.88}$$

Proof. We show the lower bound in the time window. To do that, first we suppose that $\boldsymbol{c}^{(t)}(u) - \boldsymbol{c}^{(t+1)}(u) > 0$ and show that the claim holds; then we show that the claim also holds when $\boldsymbol{c}^{(t)}(u) - \boldsymbol{c}^{(t+1)}(u) < 0$.

Let us suppose $\boldsymbol{c}^{(t)}(u) - \boldsymbol{c}^{(t+1)}(u) > 0$. If $\boldsymbol{y}(u) < 0$ the thesis follows directly; then let us suppose $\boldsymbol{y}(u) \geqslant 0$. From Claim 7.2.3 we have that

$$\begin{aligned} &\boldsymbol{c}^{(t)}(u) - \boldsymbol{c}^{(t+1)}(u) \\ &\quad \geqslant \lambda_k^t (1 - \lambda_2) \sum_{i=2}^k \alpha_i \boldsymbol{v}_i(u) + (t+1)\lambda_2^t (\lambda_2 - \lambda_k) \sum_{i : \alpha_i \boldsymbol{v}_i(u) < 0} \alpha_i \boldsymbol{v}_i(u). \end{aligned} \tag{7.89}$$

In order to prove the lemma in this first case, we need to show that

$$\frac{1}{2}\lambda_k^t (1 - \lambda_2) \sum_{i=2}^k \alpha_i \boldsymbol{v}_i(u) > -(t+1)\lambda_2^t (\lambda_2 - \lambda_k) \sum_{i : \alpha_i \boldsymbol{v}_i(u) < 0} \alpha_i \boldsymbol{v}_i(u). \tag{7.90}$$

We lower bound the left hand side and upper bound the right hand side. For the lower bound we apply Lemma 7.1.2 to get that $\sum_{i=2}^k \alpha_i \boldsymbol{v}_i(u) = \boldsymbol{y}(u)$ and

Lemma 7.2.3 to get $\boldsymbol{y}(u) \geqslant \frac{k}{\Delta n}$, with high probability. For the upper bound, instead, we rely on Claim 7.2.4 and on the fact that $\alpha_i \boldsymbol{v}_i = \beta_i D^{\frac{1}{2}} \boldsymbol{w}_i$, for every $i \in [n]$. Indeed

$$-\sum_{i:\alpha_i \boldsymbol{v}_i(u)<0} \alpha_i \boldsymbol{v}_i(u) = -\sum_{i:\ \beta_i \boldsymbol{w}_i(u)<0} \frac{\beta_i}{\sqrt{\delta(u)}} \boldsymbol{w}_i(u) \overset{(a)}{\leqslant} k\sqrt{\Delta \log n}, \tag{7.91}$$

where in (a) we can apply Claim 7.2.4 since $\|\boldsymbol{w}_i\|_2 = 1$ for every $i \in [k]$ and $\beta_i = \langle D^{\frac{1}{2}} \boldsymbol{x}, \boldsymbol{w}_i \rangle$. By combining lower and upper bounds, we get

$$\frac{1}{2}\lambda_k^t (1-\lambda_2) \sum_{i=2}^{k} \alpha_i \boldsymbol{v}_i(u) > -(t+1)\lambda_2^t (\lambda_2 - \lambda_k) \sum_{i:\alpha_i \boldsymbol{v}_i(u)<0} \alpha_i \boldsymbol{v}_i(u) \tag{7.92}$$

$$\frac{1}{2}\lambda_k^t (1-\lambda_2) \frac{k}{\Delta n} > (t+1)\lambda_2^t (\lambda_2 - \lambda_k) k\sqrt{\Delta \log n} \tag{7.93}$$

$$\left(\frac{\lambda_2}{\lambda_k}\right)^t (t+1) < \frac{1}{2} \frac{1-\lambda_2}{\lambda_2 - \lambda_k} \frac{1}{\Delta^{\frac{3}{2}} n \sqrt{\log n}}. \tag{7.94}$$

By hypothesis we have that $\frac{1-\lambda_2}{\lambda_2-\lambda_k} \geqslant \Delta^{\frac{3}{2}} n^{1+c}$ and that $\lambda_k > \frac{1}{2}$. Thus, we can derive an upper bound for $\frac{\lambda_2}{\lambda_k}$, namely

$$\frac{\lambda_2}{\lambda_k} = 1 + \frac{\lambda_2 - \lambda_k}{\lambda_k} \leqslant 1 + \frac{1-\lambda_2}{\lambda_k \Delta^{\frac{3}{2}} n^{1+c}} \leqslant 1 + \frac{1}{\Delta^{\frac{3}{2}} n^{1+c}} \leqslant 1 + \frac{1}{n^{\frac{c}{3}}}. \tag{7.95}$$

Moreover, by the hypothesis on $\frac{1-\lambda_2}{\lambda_2-\lambda_k}$, we know that

$$\frac{1}{2} \frac{1-\lambda_2}{\lambda_2 - \lambda_k} \frac{1}{\Delta^{\frac{3}{2}} n \sqrt{\log n}} \geqslant \frac{1}{2} n^{\frac{c}{2}}. \tag{7.96}$$

We apply Equations (7.95) and (7.96) to Equation (7.94) to find a time T_2 such that for every $t \leqslant T_2$ the lemma holds, and get

$$\left(1 + \frac{1}{n^{\frac{c}{3}}}\right)^t (t+1) < \frac{1}{2} n^{\frac{c}{2}}. \tag{7.97}$$

Let $T_2 = n^{\frac{c}{3}}$. Note that $\left(1 + \frac{1}{n^{\frac{c}{3}}}\right)^t \leqslant e$ for every time $t \leqslant T_2$; thus, for every time $t < T_2$, it also holds that $e\,(t+1) < \frac{1}{2} n^{\frac{c}{2}}$. We conclude that, in this first case, there exists a time $T_2 = \Omega(n^{\frac{c}{3}})$ such that, for every $t < T_2$,

$$\boldsymbol{c}^{(t)}(u) - \boldsymbol{c}^{(t+1)}(u) \geqslant \frac{1}{2}\lambda_k^t (1-\lambda_2) \sum_{i=1}^{k-1} \gamma_i \boldsymbol{\chi}_i(u). \tag{7.98}$$

Let us now suppose $\boldsymbol{c}^{(t)}(u) - \boldsymbol{c}^{(t+1)}(u) < 0$. As before, if $\boldsymbol{y}(u) > 0$ the thesis directly follows; then let us suppose $\boldsymbol{y}(u) \leqslant 0$. From Claim 7.2.3 we have that

$$\begin{aligned} &\boldsymbol{c}^{(t)}(u) - \boldsymbol{c}^{(t+1)}(u) \\ &\leqslant \lambda_k^t (1-\lambda_2) \sum_{i=2}^{k} \alpha_i \boldsymbol{v}_i(u) + (t+1)\lambda_2^t (\lambda_2 - \lambda_k) \sum_{i:\alpha_i \boldsymbol{v}_i(u)>0} \alpha_i \boldsymbol{v}_i(u). \end{aligned} \tag{7.99}$$

Similarly to the previous case, in order to prove the lemma we need to show that

$$\frac{1}{2}\lambda_k^t(1-\lambda_2)\sum_{i=2}^{k}\alpha_i\boldsymbol{v}_i(u) \leqslant -(t+1)\lambda_2^t(\lambda_2-\lambda_k)\sum_{i:\alpha_i\boldsymbol{v}_i(u)>0}\alpha_i\boldsymbol{v}_i(u). \tag{7.100}$$

Again, we upper bound the left hand side using Lemma 7.1.2 and Lemma 7.2.3 and getting $\sum_{i=2}^{k}\alpha_i\boldsymbol{v}_i(u) = \sum_{i=1}^{k-1}\gamma_i\boldsymbol{\chi}_i(u) \leqslant -\frac{k}{\Delta n}$, with high probability. As for the right hand side we use Claim 7.2.4 and get that $-\sum_{i:\alpha_i\boldsymbol{v}_i(u)>0}\alpha_i\boldsymbol{v}_i(u) \geqslant -k\sqrt{\Delta\log n}$. By combining the two bounds we get

$$-\frac{1}{2}\lambda_k^t(1-\lambda_2)\frac{k}{\Delta n} < -(t+1)\lambda_2^t(\lambda_2-\lambda_k)k\sqrt{\Delta\log n}, \tag{7.101}$$

which is exactly the same condition of the previous case. Thus, for every time $t < T_2 = \Omega(n^{\frac{3}{2}})$, we have that

$$\boldsymbol{c}^{(t)}(u) - \boldsymbol{c}^{(t+1)}(u) \leqslant \frac{1}{2}\lambda_k^t(1-\lambda_2)\sum_{i=1}^{k-1}\gamma_i\boldsymbol{\chi}_i(u). \tag{7.102}$$

By combining Eq. (7.98) and Eq. (7.102), we conclude that $\left|\boldsymbol{c}^{(t)}(u) - \boldsymbol{c}^{(t+1)}(u)\right| \geqslant \frac{1}{2}\lambda_k^t(1-\lambda_2)\left|\boldsymbol{y}(u)\right|$.

Now we show that $\operatorname{sgn}(\boldsymbol{c}^{(t)}(u) - \boldsymbol{c}^{(t+1)}(u)) = \operatorname{sgn}(\boldsymbol{y}(u))$. In particular, Equations (7.90) and (7.100) imply that $-(t+1)\lambda_2^t(\lambda_2-\lambda_k)\sum_{i:\alpha_i\boldsymbol{v}_i(u)<0}\alpha_i\boldsymbol{v}_i(u) \leqslant \frac{1}{2}\lambda_k^t(1-\lambda_2)\left|\boldsymbol{y}(u)\right|$ and that $(t+1)\lambda_2^t(\lambda_2-\lambda_k)\sum_{i:\alpha_i\boldsymbol{v}_i(u)>0}\alpha_i\boldsymbol{v}_i(u) \leqslant \frac{1}{2}\lambda_k^t(1-\lambda_2)\left|\boldsymbol{y}(u)\right|$. Thus, upper and lower bounds for $\boldsymbol{c}^{(t)}(u)-\boldsymbol{c}^{(t+1)}(u)$ in Claim 7.2.3, during for every $t < T_2$, have the same sign of $\boldsymbol{y}$ and consequently $\operatorname{sgn}(\boldsymbol{c}^{(t)}(u) - \boldsymbol{c}^{(t+1)}(u)) = \operatorname{sgn}(\boldsymbol{y}(u))$. ■

Upper bound on the error contribution

Lemma 7.2.5 (Upper bound on the error contribution)**.** *Let* $\boldsymbol{e}^{(t)} := \sum_{i=k+1}^{n}\lambda_i^t\alpha_i\boldsymbol{v}_i$. *For every* $u \in V$*, it holds that*

$$\left|\boldsymbol{e}^{(t)}(u)\right| \leqslant \lambda_{k+1}^t\sqrt{\Delta n}. \tag{7.103}$$

Proof. To bound all components of vector $\boldsymbol{e}^{(t)}$ we use its ℓ^∞ norm, defined for any vector $\boldsymbol{x}$ as $\|\boldsymbol{x}\|_\infty := \sup_i\left|\boldsymbol{x}(i)\right|$. In particular

$$\begin{aligned}
\|\boldsymbol{e}^{(t)}\|_\infty^2 &\leqslant \|\boldsymbol{e}^{(t)}\|^2 && (7.104)\\
&= \left\|\sum_{i=k+1}^{n}\lambda_i^t\alpha_i\boldsymbol{v}_i\right\|^2 && (7.105)\\
&= \left\|\sum_{i=k+1}^{n}\lambda_i^t\beta_i D^{-\frac{1}{2}}\boldsymbol{w}_i\right\|^2 && (7.106)\\
&\overset{(a)}{\leqslant} \left\|D^{-\frac{1}{2}}\right\|^2\left\|\sum_{i=k+1}^{n}\lambda_i^t\beta_i\boldsymbol{w}_i\right\|^2 && (7.107)
\end{aligned}$$

$$\overset{(b)}{=} \left\|D^{-\frac{1}{2}}\right\|^2 \sum_{i=k+1}^{n} \lambda_i^{2t}\beta_i^2 \tag{7.108}$$

$$\leqslant \left\|D^{-\frac{1}{2}}\right\|^2 \lambda_{k+1}^{2t} \sum_{i=k+1}^{n} \beta_i^2 \tag{7.109}$$

$$\leqslant \left\|D^{-\frac{1}{2}}\right\|^2 \lambda_{k+1}^{2t} \sum_{i=1}^{n} \beta_i^2 \tag{7.110}$$

$$= \left\|D^{-\frac{1}{2}}\right\|^2 \lambda_{k+1}^{2t} \left\|D^{\frac{1}{2}}\boldsymbol{x}\right\|^2 \tag{7.111}$$

$$\leqslant \left\|D^{-\frac{1}{2}}\right\|^2 \lambda_{k+1}^{2t} \left\|D^{\frac{1}{2}}\right\|^2 \|\boldsymbol{x}\|^2 \tag{7.112}$$

$$\overset{(c)}{=} \frac{\max_u \delta(u)}{\min_u \delta(u)} \lambda_{k+1}^{2t} \|\boldsymbol{x}\|^2 \tag{7.113}$$

$$\leqslant \lambda_{k+1}^{2t}\Delta n, \tag{7.114}$$

where in (a) we use Cauchy-Schwarz inequality (Theorem C.1.5) and we apply the definition of spectral norm of an operator, i.e., $\|A\| := \sup_{\boldsymbol{x}:\|\boldsymbol{x}=1\|} \|A\boldsymbol{x}\|$; in (b) we use that the $\boldsymbol{w}_i$s are orthonormal; in (c) we use that the spectral norm of a diagonal matrix is equal to its maximum value. Thus, for every $u \in V$ it holds that $|\boldsymbol{e}^{(t)}(u)| \leqslant \sqrt{\|\boldsymbol{e}^{(t)}\|_\infty^2} \leqslant \lambda_{k+1}^t\sqrt{\Delta n}$. ■

7.2.2 Bipartite graphs

Assume $G = (V, E, w)$ is a bipartite 2-volume-regular graph, i.e., $V = V_1 \cup V_2$, $E \subseteq V_1 \times V_2$ and G is volume-regular w.r.t. the bipartition (V_1, V_2). In this case, basic properties of random walks imply that the Averaging dynamics does not converge to the global (weighted) average of the values, but it periodically oscillates. In fact, in this case the transition matrix P has an eigenvector $\boldsymbol{\chi} = \mathbf{1}_{V_1} - \mathbf{1}_{V_2}$ with eigenvalue $\lambda_n = -1$ (as implied by Claim 7.2.5). Thus, the state vector is mainly affected by the eigenvectors associated to the two eigenvalues of absolute value 1 (i.e., λ_1 and λ_n). After a number of rounds of the dynamics that depends on $1/\lambda_2$, we have that, in even rounds, all nodes in V_i $(i = 1, 2)$ have a state that is close to some local average μ_i; in odd rounds, these values are swapped (as shown in Eq. (7.118)).

If one were observing the process in even rounds,[3] however, the states of nodes in V_1 would converge to μ_1 and those of nodes in V_2 would converge to μ_2. Unfortunately (and differently from clustered volume-regular graphs), convergence to the local average for nodes belonging to the same community does not eventually become monotone (i.e., increasing or decreasing). This follows since the eigenvector associated to λ_2 is no longer stepwise in general (lumpable classes are already associated to $\mathbf{1}$ and $\boldsymbol{\chi}$). However, we can easily modify the labeling scheme of the Averaging dynamics to perform *bipartiteness detection* as follows: Nodes apply the labeling rule every two time steps and they do it between the states of two consecutive rounds, i.e., each node $v \in V$ sets $\texttt{label}^{(2t)}(v) = 1$ if $\boldsymbol{x}^{(2t)}(v) \geqslant \boldsymbol{x}^{(2t-1)}(v)$

[3] Or, equivalently, in odd rounds.

and $\texttt{label}^{(2t)}(v) = 0$ otherwise. We call this new protocol Averaging Bipartite dynamics.

Let $G = (V, E, w)$ be an edge-weighted undirected bipartite volume-regular graph. We denote with $W \in \mathbb{R}^{n\times n}$ the weighted adjacency matrix of G. Since G is undirected and bipartite, the matrix W can be written as

$$W = \begin{pmatrix} 0 & W_1 \\ W_2 & 0 \end{pmatrix} = \begin{pmatrix} 0 & W_1 \\ W_1^T & 0 \end{pmatrix}. \tag{7.115}$$

Thus, the transition matrix of a simple random walk on G, i.e., $P = D^{-1}W$ where D^{-1} is a diagonal matrix and $D_{ii} = \frac{1}{\delta(i)}$, has the form

$$P = \begin{pmatrix} 0 & P_1 \\ P_1^T & 0 \end{pmatrix}. \tag{7.116}$$

Claim 7.2.5 shows that the spectrum of P is symmetric and it gives a relation between the eigenvectors of symmetric eigenvalues.

Claim 7.2.5. *Let $G = (V_1 \cup V_2, E, w)$ be an edge-weighted undirected bipartite graph with bipartition (V_1, V_2) and such that $|V_i| = n_i$. If $\boldsymbol{v} = (\boldsymbol{v}_1, \boldsymbol{v}_2)^T$, with $\boldsymbol{v}_i \in \mathbb{R}^{n_i}$, is an eigenvector of P with eigenvalue λ, then $\boldsymbol{v}' = (\boldsymbol{v}_1, -\boldsymbol{v}_2)^T$ is an eigenvector of P with eigenvalue $-\lambda$.*

Proof. If $P\boldsymbol{v} = \lambda\boldsymbol{v}$ then we have that $P_1\boldsymbol{v}_2 = \lambda\boldsymbol{v}_1$ and $P_1^T\boldsymbol{v}_2 = \lambda\boldsymbol{v}_2$. Using these two equalities we get that $P\boldsymbol{v}' = -\lambda\boldsymbol{v}'$. In fact,

$$P\boldsymbol{v}' = \begin{pmatrix} 0 & P_1 \\ P_1^T & 0 \end{pmatrix} \begin{pmatrix} \boldsymbol{v}_1 \\ -\boldsymbol{v}_2 \end{pmatrix} = \begin{pmatrix} -P_1\boldsymbol{v}_2 \\ P_1^T\boldsymbol{v}_1 \end{pmatrix} = -\lambda \begin{pmatrix} \boldsymbol{v}_1 \\ -\boldsymbol{v}_2 \end{pmatrix}. \tag{7.117}$$

■

The transition matrix P is stochastic, thus the vector $\mathbf{1}$ (i.e., the vector of all ones) is an eigenvector associated to $\lambda_1 = 1$, that is the first largest eigenvalue of P. Claim 7.2.5 implies that $\boldsymbol{\chi} = \mathbf{1}_{V_1} - \mathbf{1}_{V_2}$ is an eigenvector of P with eigenvalue $\lambda_n = -1$.

As in Section 7.1, we write the state vector at time t using the spectral decomposition of P. Let $1 = \lambda_1 > \lambda_2 \geqslant \ldots > \lambda_n = -1$ be the eigenvalues of P. We denote by $\mathbf{1} = \boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_n = \boldsymbol{\chi}$ a family of n linearly independent eigenvectors of P, where each $\boldsymbol{v}_i$ is the eigenvector associated to λ_i. Thus, we have that

$$\boldsymbol{x}^{(t)} = P^t\boldsymbol{x} = \sum_{i=1}^{n} \lambda_i^t \alpha_i \boldsymbol{v}_i = \alpha_1\mathbf{1} + (-1)^t\alpha_n\boldsymbol{\chi} + \sum_{i=2}^{n-1} \lambda_i^t\alpha_i\boldsymbol{v}_i, \tag{7.118}$$

where $\alpha_i = \frac{\langle D^{\frac{1}{2}}\boldsymbol{x}, D^{\frac{1}{2}}\boldsymbol{v}_i\rangle}{\|D^{\frac{1}{2}}\boldsymbol{v}_i\|^2}$. The last equation implies that $\boldsymbol{x}^{(t)} = P^t\boldsymbol{x}$ does not converge to some value as t tends to infinity, but oscillates. In particular, nodes in V_1 on even rounds and nodes in V_2 on odd rounds, converge to $\alpha_1 + \alpha_n$. Instead in the

symmetric case, i.e., odd rounds for nodes in V_1 and even rounds for nodes in V_2, the process converges to $\alpha_1 - \alpha_n$. These quantities are proportional to the weighted average of the initial values in the first and in the second partition, respectively.

Lemma 7.2.6, whose proof follows, shows that Averaging Bipartite dynamics performs bipartiteness detection in $\mathcal{O}(\log n \,/\, \log(1/\lambda_2))$ rounds. Note that if $\log(1/\lambda_2) = \Omega(1)$, then the Averaging Bipartite dynamics takes logarithmic time to find the bipartition.

Lemma 7.2.6. *Let $G = (V, E, w)$ be an edge-weighted bipartite volume-regular graph with bipartition V_1, V_2 and maximum weighted degree $\Delta \leqslant poly(n)$. Then for every time $t > T$, with $T = \mathcal{O}(\log n \,/\, \log(1/\lambda_2))$, the Averaging Bipartite dynamics is a $(\mathcal{O}(n^{-\frac{1}{2}}), \mathcal{O}(1))$-community sensitive algorithm, w.h.p.*

Proof. We assume that the coloring rule is applied between every even and every odd round (conversely, the signs of the nodes in the analysis are swapped). Recall the definition of the *error contribution*, namely $\boldsymbol{e}^{(t)}(u) = \sum_{i=2}^{n-1} \lambda_i^t \alpha_i \boldsymbol{v}_i(u)$. We compute the difference between the state vectors of two consecutive steps by using Eq. (7.118), namely

$$\boldsymbol{x}^{(2t)} - \boldsymbol{x}^{(2t+1)} = \alpha_1 \mathbf{1} + (-1)^{2t}\alpha_n \boldsymbol{\chi} + \boldsymbol{e}^{(2t)} - \alpha_1 \mathbf{1} - (-1)^{2t+1}\alpha_n \boldsymbol{\chi} - \boldsymbol{e}^{(2t+1)} \tag{7.119}$$

$$= 2\alpha_n \boldsymbol{\chi} + \boldsymbol{e}^{(2t)} - \boldsymbol{e}^{(2t+1)}. \tag{7.120}$$

We want to find a time T such that for every $t > T$ the sign of a node $u \in V$ depends only on $\boldsymbol{\chi}(u)$. Formally, $\operatorname{sgn}(\boldsymbol{x}^{(2t)}(u) - \boldsymbol{x}^{(2t+1)}(u)) = \operatorname{sgn}(\alpha_n \boldsymbol{\chi})$. The last equation holds whenever

$$2|\alpha_n \boldsymbol{\chi}(u)| > |\boldsymbol{e}^{(2t)}(u) - \boldsymbol{e}^{(2t+1)}(u)| \tag{7.121}$$

$$2|\alpha_n| > |\boldsymbol{e}^{(2t)}(u) - \boldsymbol{e}^{(2t+1)}(u)|. \tag{7.122}$$

We upper bound $|\boldsymbol{e}^{(2t)}(u) - \boldsymbol{e}^{(2t+1)}(u)|$ by using Lemma 7.2.5. We get that $|\boldsymbol{e}^{(2t)}(u) - \boldsymbol{e}^{(2t+1)}(u)| \leqslant 2\lambda_2^{2t}\sqrt{\Delta n}$. We get that Eq. (7.122) holds if the following holds:

$$|\alpha_n| > \lambda_2^{2t}\sqrt{\Delta n} \tag{7.123}$$

$$(1/\lambda_2)^{2t} > \frac{\sqrt{\Delta n}}{|\alpha_n|} \tag{7.124}$$

$$2t > \log\left(\frac{\sqrt{\Delta n}}{|\alpha_n|}\right)\frac{1}{\log(1/\lambda_2)}. \tag{7.125}$$

In order to find the time t which makes the last inequality hold, we provide a lower bound on $|\alpha_n|$, showing that it is not too small, with high probability. Recall that $\alpha_i = \frac{\langle D^{\frac{1}{2}}\boldsymbol{x}, D^{\frac{1}{2}}\boldsymbol{v}_i\rangle}{\|D^{\frac{1}{2}}\boldsymbol{v}_i\|^2}$ and thus

$$\alpha_n = \frac{\langle D^{\frac{1}{2}}\boldsymbol{x}, D^{\frac{1}{2}}\boldsymbol{\chi}\rangle}{\|D^{\frac{1}{2}}\boldsymbol{\chi}\|^2} = \frac{1}{\operatorname{vol}(V)}\sum_{v\in V}\delta(v)\boldsymbol{x}(v)\boldsymbol{\chi}(v), \tag{7.126}$$

where $\mathrm{vol}(V) = \sum_{v \in V} \delta(v)$. We get the lower bound, with high probability, by showing that

$$\mathbf{P}\left(|\alpha_n| \leqslant \frac{1}{\Delta n}\right) \leqslant \mathbf{P}\left(|\alpha_n| \leqslant \frac{1}{\mathrm{vol}(V)}\right) \tag{7.127}$$

$$= \mathbf{P}\left(\left|\sum_{v \in V} \delta(v)\boldsymbol{x}(v)\boldsymbol{\chi}(v)\right| \leqslant 1\right) \tag{7.128}$$

$$\overset{(a)}{=} \mathcal{O}\left(\frac{1}{\sqrt{n}}\right) \tag{7.129}$$

where in (a) we apply Theorem E.3.3. Indeed this last inequality implies that $|\alpha_n| > \frac{1}{\Delta n}$ with high probability. The thesis then follows from the above bound on $|\alpha_n|$ and from the hypothesis on $\Delta \leqslant poly(n)$. ■

7.2.3 Other non-clustered graphs

Consider k-volume-regular graphs whose k stepwise eigenvectors are associated to the k largest eigenvalues, in absolute value. These graphs include many k-partite graphs (e.g., regular ones), graphs that are "close" to being k-partite (i.e., ones that would become k-partite upon removal of a few edges). Differently from the clustered case (Theorem 7.2.1) some of the k eigenvalues can in general be negative.

Consider the following variant of the labeling scheme of the Averaging dynamics, in which nodes apply their labeling rule only on even rounds, comparing their value with the one they held at the end of the last even round, i.e., each node $v \in V$ sets $\texttt{label}^{(2t)}(v) = 1$ if $\boldsymbol{x}^{(2t)}(v) \geqslant \boldsymbol{x}^{(2t-2)}(v)$ and $\texttt{label}^{(2t)}(v) = 0$ otherwise. Since the above protocol amounts to only taking even powers of eigenvalues, the analysis of this modified protocol proceeds along the same lines as the clustered case, while the results of Theorem 7.2.1 seamlessly extend to this class of graphs.

8

Conclusion

Complex networks are ubiquitous, since are able to model many aspects in several heterogeneous areas. They have non-trivial topologies, often exhibiting a community structure. Detecting such communities is fundamental for many applications and in the last decade many techniques based on different approaches have been proposed. One of the most widely used class of such techniques is that of *label propagation algorithms* (LPAs), dynamical processes on networks inspired by epidemic spreads of viruses. Although largely used in practice for their simplicity, efficiency, and effectiveness the scientific literature lacks of theoretical results that explain the behavior of such dynamical processes on networks and why they are able to perform well in the task of graph clustering. In fact, the absence of proper mathematical tools to analyze the interplay between the non-linearity of the dynamics and the non-trivial topology of the network makes harder to obtain a substantial theoretical progress.

In this thesis we considered some simple *dynamics* that are can be seen as label propagation algorithms and rigorously analyzed them. The dynamics we considered have been rigorously studied in the past, but exclusively on graphs with extremely good connectivity properties (complete or expander graphs) and are well known for naturally bringing the nodes of the network to a consensus. Differently from previous work, we analyze such dynamics in a setting where the networks exhibit a *community structure*. The main idea is that if the communication network is more complex, presenting a community structure, some dynamics make the nodes quickly reach (and maintain for a long time window) many *local consensus* instead of a *global* one. We showed that this behavior, that we call *metastability*, can be exploited to design lightweight distributed algorithms for graph clustering, in which each node only uses local information and a very low computational power.

The metastable behavior is not a feature of every dynamics, though. The well studied *voter dynamics*, for example, would converge in polynomial time to a consensus independently of the underlying network, and hence, also in the case of networks presenting a community structure. An open question thus remain that of studying which are the properties that a dynamics should have in order to exhibit

such metastable behavior in graphs with communities, which is fundamental to utilize them as label propagation algorithms. To achieve that it should be possible to explore the behavior of other already studied dynamics for consensus, e.g., the Undecided-State dynamics [CGG+18], on non-trivial topologies and to extend the analysis of the 2-Choices dynamics to other classes of graphs with other features, e.g., graphs with power-law degree distribution.

Appendix

A

Asymptotic Notation

Here we introduce the Bachmann–Landau asymptotic notation. Let $f : \mathbb{R}^+ \to \mathbb{R}$ and $g : \mathbb{R}^+ \to \mathbb{R}^+$ be two real functions. As $n \to \infty$, we say that:

- f is asymptotically dominated by g, i.e.,

$$f(n) = o(g(n)) \iff \forall k > 0\, \exists n_0\, \forall n > n_0 \quad |f(n)| < kg(n); \tag{A.1}$$

- f is asymptotically bounded above by g, up to a constant factor, i.e.,

$$f(n) = \mathcal{O}(g(n)) \iff \exists k > 0\, \exists n_0\, \forall n > n_0 \quad |f(n)| \leqslant kg(n); \tag{A.2}$$

- f is asymptotically bounded above and below by g, up to a constant factor, i.e.,

$$f(n) = \Theta(g(n)) \iff \exists k_1 > 0\, \exists k_2 > 0\, \exists n_0\, \forall n > n_0 \\ k_1 g(n) \leqslant |f(n)| \leqslant k_2 g(n); \tag{A.3}$$

- f is asymptotically bounded below by g, up to a constant factor, i.e.,

$$f(n) = \Omega(g(n)) \iff \exists k > 0\, \exists n_0\, \forall n > n_0 \quad |f(n)| \geqslant kg(n); \tag{A.4}$$

- f asymptotically dominates g, i.e.,

$$f(n) = \omega(g(n)) \iff \forall k > 0\, \exists n_0\, \forall n > n_0 \quad |f(n)| > kg(n). \tag{A.5}$$

B

Graph Theory

Graph theory is that field of mathematics which studies graphs, structures used to model and represent relations between objects. This section covers the basic definitions, some examples of simple graphs, and some graph metrics.

B.1 Definitions

Definition B.1.1. *An* undirected graph*, or just* graph*, $G = (V, E)$ is defined as a nonempty set of vertices (or nodes) V and a set of edges (unordered pair of nodes) $E \subseteq V \times V = \{(u, v) : u, v \in V,\ u \neq v\}$.*

Given an edge $e = (u, v)$, then u and v are said *adjacent nodes* or *neighbors*. Moreover, in the whole thesis we will often use the following classic notations: $|V| = n$ and $|E| = m$.

Definition B.1.2. *A* directed graph*, or* digraph*, $G = (V, E)$ is defined as a nonempty set of vertices V and a set of edges (ordered pair of nodes) $E \subseteq V \times V$.*

Note that a digraph, differently from a graph, allows *loops* (edges going from a node v to itself) and opposite edges among the same pair of nodes, i.e. $(i, j) \neq (j, i)$.

Definition B.1.3. *The degree $deg(v)$ of a vertex v is the number of edges containing v, formally $|\{(i, j) \in E : i = v \text{ or } j = v\}|$. Sometimes we refer to $deg(v)$ as d_v.*

Definition B.1.4. *A graph G is d-regular when $deg(v_1) = \ldots = deg(v_n) = d$.*

Lemma B.1.1 (Handshaking lemma)**.** *Given a graph G, it holds that*

$$\sum_{v \in V} deg(v) = 2|E|. \tag{B.1}$$

Proof. We just need to observe that each edge is counted two times in the sum, contributing to the degree of exactly two nodes. ■

Definition B.1.5. *Given a graph $G = (V, E)$, we call a subgraph of G a graph $G' = (V', E')$ s.t. $V' \subseteq V$ and $E' \subseteq E$.*

Definition B.1.6. *A* path *on a graph $G = (V, E)$ is a sequence $v_1, e_1, \ldots, e_{k-1}, v_k$, with $v_i \in V$ and $e_i = (v_i, v_{i+1}) \in E$ for $i = 1, \ldots, k$, and such that all the v_i and e_i are distinct.*

Definition B.1.7. *A graph G is said to be connected if there exists a path connecting each pair of nodes.*

Definition B.1.8. *The* vertex connectivity *of a graph G is the minimum number of vertices that have to be removed such that G will be disconnected.*

Definition B.1.9. *The* edge connectivity *of a graph G is the minimum number of edges that have to be removed such that G will be disconnected.*

Definition B.1.10. *The* algebraic connectivity *of a graph G is the value of the second smallest eigenvalue of the Laplacian matrix of G.*

B.2 Matrix representations

Adjacency matrix. The *adjacency matrix* $A \in \mathbb{R}^{n \times n}$ of a graph $G = (V, E)$ is defined as

$$A_{ij} = \begin{cases} 1 & \text{if } (i,j) \in E, \\ 0 & \text{otherwise.} \end{cases} \tag{B.2}$$

Incidence matrix. The *incidence matrix* $B \in \mathbb{R}^{n \times m}$ of a graph $G = (V, E)$, where $e \in E$ and $v \in V$, is defined as

$$B_{ve} = \begin{cases} 1 & \text{if } e \text{ is incident to } v, \\ 0 & \text{otherwise.} \end{cases} \tag{B.3}$$

Degree matrix. The diagonal *degree matrix* $D \in \mathbb{R}^{n \times n}$ of a graph $G = (V, E)$ is defined as

$$D_{ij} = \begin{cases} deg(i) & \text{if } i = j, \\ 0 & \text{otherwise.} \end{cases} \tag{B.4}$$

Transition matrix. The *transition matrix* $P \in \mathbb{R}^{n \times n}$ of a graph $G = (V, E)$ is defined, in matrix notation, as $P = D^{-1}A$. In other words

$$P_{ij} = \frac{1}{deg(i)} A_{ij}. \tag{B.5}$$

Notice that if G is regular, then P is symmetric. The transition matrix will play a really important role in this thesis and we will extend its discussion later, especially in E.2.

Laplacian matrix. The *Laplacian matrix* $L \in \mathbb{R}^{n \times n}$ of a graph $G = (V, E)$ is defined, in matrix notation, as $L = D - A$ or, element-wise, as

$$L_{ij} = \begin{cases} deg(i) & \text{if } i = j, \\ -1 & \text{if } i \neq j \text{ and } (i,j) \in E, \\ 0 & otherwise. \end{cases} \tag{B.6}$$

Normalized Laplacian matrix. The *Normalized Laplacian matrix* $L^{RW} \in \mathbb{R}^{n \times n}$ of a graph G is defined, in matrix notation, as $L^{RW} = D^{-1}L$ or, element-wise, as

$$L^{RW}_{ij} = \begin{cases} 1 & \text{if } i = j, \\ -\frac{1}{deg(i)} & \text{if } i \neq j \text{ and } (i,j) \in E, \\ 0 & otherwise. \end{cases} \tag{B.7}$$

It is named L^{RW} since it is closely related to a Random Walk (RW) on the graph.

B.3 Graph families

Path. A *path graph* (Figure B.1a) is a connected graph with n vertices and $n-1$ edges; the first and last vertex have degree 1 and all the others have degree 2.

Cycle. A *cycle graph* (Figure B.1b) is a path where the two external vertices are connected through an additional edge.

Star. A *star graph* (Figure B.1c) is a connected graph composed of $n+1$ vertices and n edges, where one central vertex has degree n and is connected to all the others, which have degree 1.

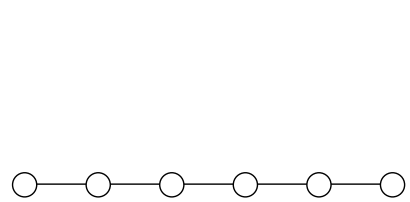
Complete. A *complete graph* (Figure B.1d), also known as *clique*, is the most connected graph: it has n vertices and each of them os connected to all the others, having a total of $\frac{n(n-1)}{2}$ edges.

Bipartite. A *bipartite graph* (Figure B.1e) is a graph of n vertices partitioned into two sets, such that nodes of the same set are not connected.

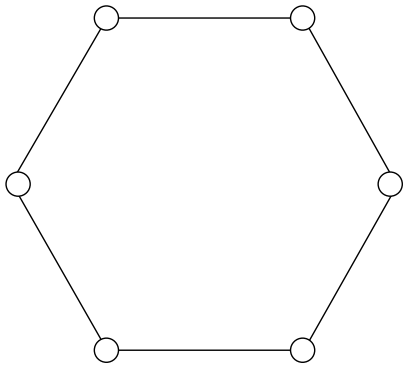
Complete bipartite. A *complete bipartite graph* (Figure B.1f) is a bipartite graph with each vertex of one set connected to all the vertices of the other.

B.4 Centrality measures

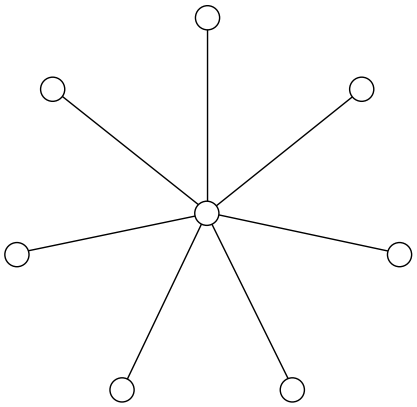
Centrality is an indicator of which are the most "central" and, in some sense, influence nodes of a graph. Such concept is not obvious to model and in fact it has several definitions. We will have an overview in this section, focusing on the ones that will be used more later.



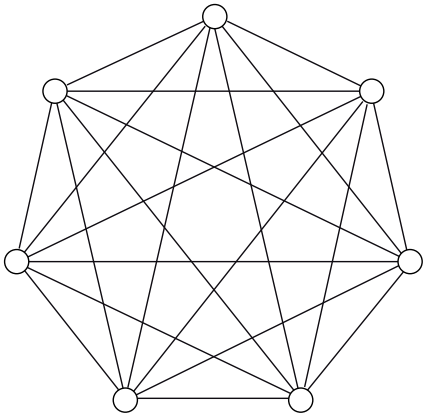
(a) A path graph.



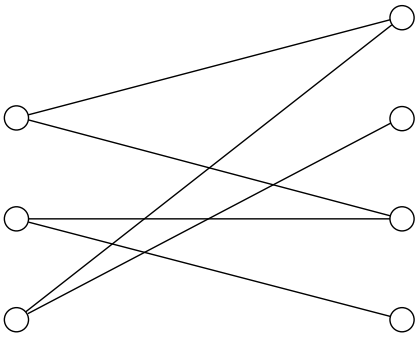
(b) A cycle graph.



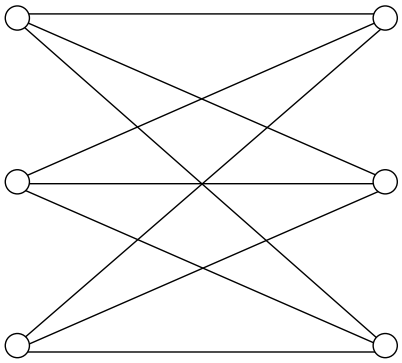
(c) A star graph.



(d) A clique.



(e) A bipartite graph.



(f) A complete bipartite graph.

Figure B.1: Some examples of graphs in figures (a), (b), (c), (d), (e), and (f).

Degree centrality. The first definition is that of *degree centrality*, which is simply represented by the degree of each node, i.e. $C_D(v) = deg(v)$ where v is a node of a graph, and it is able to tell which are the nodes more connected to others and able to catch some flow passing through the graph. It can be computed in linear time, just counting the number of edges incident to each node.

Eigenvector centrality. As the name itself suggest it involves operations with the spectrum of the graph and improves the degree centrality giving a different weight to each node, proportional to the combined score of their neighbors. The problem, for an undirected graph G with an associated adjacency matrix A, is equivalent to solve the eigenequation $A\mathbf{x} = \alpha^{-1}\mathbf{x}$, where the *eigenvector centrality* of node v is defined as $C_E(v) = \mathbf{x}_v$ and α^{-1} is the largest eigenvalue of A. Such value can be computed through the *power method*, which will be briefly described below.

The *power method* is an iterative algorithm for approximately solving the eigenvalue equation. It is especially good at finding the largest and smallest eigenvalues and their associated eigenvectors for the way it works. It starts with an initial vector $\mathbf{x}^{(0)} \in \mathbb{R}^n$ and proceeds applying over and over the adjacency matrix operator as follows

$$\mathbf{x}^{(t)} = A\mathbf{x}^{(t-1)}. \tag{B.8}$$

until convergence, i.e. until the vector $\mathbf{x}$ does not change significantly anymore. We will not discuss in details about the proof of convergence, the appropriate initial vector choice, and the running time to obtain a maximum error ε, but the idea is to iterate the above operation until the variation between $\frac{\mathbf{x}^{(t+1)}}{\|\mathbf{x}^{(t+1)}\|}$ and $\frac{\mathbf{x}^{(t)}}{\|\mathbf{x}^{(t)}\|}$ is under a certain threshold.

Katz centrality. *Katz centrality* is a generalization of degree centrality and can be seen as a variation of eigenvector centrality since can be computed in a way similar to the power method. It is defined as $\mathbf{x} = \alpha A^\intercal \mathbf{x} + \beta\mathbf{1}$, with α and β some positive constants. The *Katz centrality* of node v is then defined as $C_K(v) = \mathbf{x}_v$, where $\mathbf{x}$ can be found solving $\mathbf{x} = \beta(I - \alpha A^\intercal)^{-1} \cdot \mathbf{1}$.

Pagerank centrality. *Pagerank centrality* is a very well known measure since the algorithm that computes was developed by Google and is at the base of Google search engine. The measure tries to overcome an issue that we could face with Katz centrality: all the nodes linked by a node with high centrality get high centrality too, independently from how many they are. Formally

$$C_{PR} = \mathbf{x}_v = \sum_{u \in V} A_{uv} \frac{\mathbf{x}_u}{d_u^+} + \beta \tag{B.9}$$

where d_u^+ is the *out-degree* of node u.

Closeness centrality. Informally a node has a high *closeness centrality* if it is close to many other nodes. In fact it is defined as the reciprocal of the average length of the shortest path between a node towards all the others. Formally

$$C_C(v) = \frac{n}{\sum_{u \in V} d(u, v)} \tag{B.10}$$

with $d(u, v)$ being the distance or shortest path (minimum number of hops) between nodes u and v.

Betweenness centrality. The *betweenness centrality* of a node describes how often such nodes is in between the shortest path of two random nodes in a network. It is defined as follows

$$C_B(v) = \sum_{s,t \in V} \frac{\sigma_{st}^v}{\sigma_{st}} \tag{B.11}$$

where σ_{st} is the total number of shortest paths between nodes s and t, and σ_{st}^v is the fraction of shortest paths between nodes s and t passing through v. In such formula we consider the fact that if σ_{st} and σ_{st}^v are 0 then $\frac{\sigma_{st}^v}{\sigma_{st}} = 0$.

Another definition is the one of *edge betweenness* in terms of flow. For every pair of nodes s, t let s split a unit flow through all possible paths towards t. The edges betweenness of $e = (u, v)$ is defined as the combined flow on e. The betweenness of a node u is the sum of the betweenness of all the edges adjacent to it. An algorithm (more details can be found in [Che13]), which runs in $\mathcal{O}(n(n + m))$, to compute such metric is the following:

1. Starting from a node v, perform a Breadth First Search (BFS).
2. Count the number of shortest paths from v to all the other nodes.
3. Compute the flow from v to all the other nodes in each edge.

There are faster algorithms that compute an approximation of the betweenness centrality, such as the recent one proposed by Riondato and Kornaropoulos in [RK14].

C

Linear Algebra

Here we review some basic definitions and results from linear algebra and, more specifically, spectral theory.

C.1 Definitions and basic results

Definition C.1.1. *Given a complex number $x = a+ib$ then we define its* conjugate *as $\overline{x} = a - ib$. Given a matrix A, its conjugate is the matrix $\overline{A}$ such having all elements are the conjugate ones of A. We denote the transpose with $A^\intercal$ and the conjugate transpose with A^*.*

Definition C.1.2 (Inner product)**.** *Given two vectors $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$ their* inner product *is defined as*

$$\langle \mathbf{x}, \mathbf{y} \rangle = \mathbf{x}^* \mathbf{y} = \sum_{i=1}^{n} \overline{\mathbf{x}_i} \mathbf{y}_i \tag{C.1}$$

being $\mathbf{x}^*$ the conjugate transpose of $\mathbf{x}$.

Such product is called *dot product* or *scalar product* in Euclidean spaces, to emphasize the notation "$\cdot$", used in $\mathbb{R}^n$ to define it, or the fact that the result is a scalar. Notice that, by definition, $\langle \mathbf{x}, \mathbf{y} \rangle = \overline{\langle \mathbf{x}, \mathbf{y} \rangle}$; the inner product of a vector with itself is equal to its squared Euclidean norm; formally $\langle \mathbf{x}, \mathbf{x} \rangle = \|\mathbf{x}\|_2^2$.

Definition C.1.3. *Given a matrix $A \in \mathbb{C}^{n \times n}$, a scalar λ, and a non-zero vector $\mathbf{v} \in \mathbb{C}^n$, whenever*

$$A\mathbf{v} = \lambda \mathbf{v} \tag{C.2}$$

we say that λ is an eigenvalue *of A and $\mathbf{v}$ the* eigenvector *of A associated to λ. The pair $\lambda, \mathbf{v}$ is also referred as* eigenpair*.*

The above equation $A\mathbf{v} = \lambda \mathbf{v}$ is also known as *eigenvalue equation.*

Definition C.1.4 (Hermitian matrix)**.** *A matrix A is* Hermitian *if it is equal to its conjugate transpose.*

Theorem C.1.1. *Given an Hermitian matrix A all its eigenvalues are real.*

Proof. Let λ and $\mathbf{v}$ be an eigenpair of a matrix A. Then, $A\mathbf{v} = \lambda\mathbf{v}$. Conjugating both sides we get that $\mathbf{v}^* A^\intercal = \overline{\lambda}\mathbf{v}^*$ and thus it follows that $\mathbf{v}^* A\mathbf{v} = \overline{\lambda}\mathbf{v}^*\mathbf{v}$ exploiting the symmetry of A. Thus $\lambda\|\mathbf{v}\|_2^2 = \overline{\lambda}\|\mathbf{v}\|_2^2$ which implies $\lambda = \overline{\lambda}$ showing that $\lambda \in \mathbb{R}$. ■

Theorem C.1.2. *Let A be an Hermitian matrix and $\mathbf{u}$, $\mathbf{v}$ two eigenvectors of A associated to different eigenvalues. Then $\mathbf{u}$ and $\mathbf{v}$ are orthogonal.*

Proof. We have the following equalities: $A\mathbf{u} = \lambda_\mathbf{u}\mathbf{u}$ and $A\mathbf{v} = \lambda_\mathbf{v}\mathbf{v}$. A is symmetric, thus $\mathbf{u}^\intercal A\mathbf{v} = \lambda_\mathbf{u}\mathbf{u}^\intercal\mathbf{v}$ for the first equality and $\mathbf{u}^\intercal A\mathbf{v} = \lambda_\mathbf{v}\mathbf{u}^\intercal\mathbf{v}$ for the second. Then $\lambda_\mathbf{v}\mathbf{u}^\intercal\mathbf{v} = \lambda_\mathbf{u}\mathbf{u}^\intercal\mathbf{v}$ and since $\lambda_\mathbf{u} \neq \lambda_\mathbf{v}$ then $\mathbf{u}^\intercal\mathbf{v} = 0$. ■

Notice that the adjacency and transition matrices associated to every regular graph are Hermitian.

Theorem C.1.3 (Spectral decomposition)**.** *Let $\lambda_1 \leqslant \ldots \leqslant \lambda_n$ be the eigenvalues of a matrix A with associated eigenvectors $\mathbf{v}_1, \ldots, \mathbf{v}_n$. Then*

$$A = \sum_{i=1}^{n} \lambda_i \mathbf{v}_i \mathbf{v}_i^\intercal. \tag{C.3}$$

Proof. Let $B = \sum_{i=1}^{n} \lambda_i \mathbf{v}_i \mathbf{v}_i^\intercal$. Thus, for all j, it holds that

$$B\mathbf{v}_j = \sum_{i=1}^{n} \lambda_i \mathbf{v}_i \mathbf{v}_i^\intercal \mathbf{v}_j = \lambda_j \mathbf{v}_j = A\mathbf{v}_j. \tag{C.4}$$

Since the eigenvectors form an orthonormal basis of the space we have for all $\mathbf{v} \in \mathbb{R}^n$ that $A\mathbf{v} = B\mathbf{v}$ and thus $A = B$. ■

Here we present a variational characterization of the eigenvalues of Hermitian operators that will be used later which is known as *variational theorem*, *min-max theorem*, or *Courant-Fischer-Weyl min-max principle*.

Theorem C.1.4 (Courant-Fischer-Weyl min-max principle)**.** *Given an $n \times n$ real symmetric matrix A, $\lambda_1 \leqslant \ldots \leqslant \lambda_n$ its eigenvalues counted with multiplicities, and $\mathbf{v}_1, \ldots, \mathbf{v}_n$ the associated orthonormal eigenvectors. Then for all $1 \leqslant k \leqslant n$ it hold that*

$$\lambda_k(A) = \min_{\substack{\mathbf{v} \in \mathbb{R}^n \setminus 0,\, \mathbf{v}^\intercal \mathbf{u}_i = 0 \\ \forall i \in \{1,\ldots,k-1\}}} \frac{\mathbf{v}^\intercal A\mathbf{v}}{\mathbf{v}^\intercal\mathbf{v}} \tag{C.5}$$

and

$$\lambda_k(A) = \max_{\substack{\mathbf{v} \in \mathbb{R}^n \setminus 0,\, \mathbf{v}^\intercal \mathbf{u}_i = 0 \\ \forall i \in \{k+1,\ldots,n\}}} \frac{\mathbf{v}^\intercal A\mathbf{v}}{\mathbf{v}^\intercal\mathbf{v}}. \tag{C.6}$$

The proof of Theorem C.1.4 can be found in [V+13].

Theorem C.1.5 (Cauchy-Schwarz's inequality)**.** *For all vectors $\boldsymbol{u}, \boldsymbol{v}$ of an inner product space it holds that*

$$|\langle \boldsymbol{u}, \boldsymbol{v} \rangle|^2 \leqslant \langle \boldsymbol{u}, \boldsymbol{u} \rangle \cdot \langle \boldsymbol{v}, \boldsymbol{v} \rangle, \tag{C.7}$$

where $\langle \cdot, \cdot \rangle$ is the inner product.

Corollary C.1.1. *Given $M \in \mathbb{R}^{n \times m}$ it holds that*

$$\|M\|_2^2 \leqslant \|M\|_1 \cdot \|M\|_\infty. \tag{C.8}$$

Proof. We have that $\|M\|_2 := \sup_{\|\boldsymbol{x}\|_2 = 1} \|M\boldsymbol{x}\|_2$, with $M\boldsymbol{x} = (m_1, \ldots, m_n)^\intercal$. Notice that

$$\|M\|_2^2 = \sum_{k=1}^{n} |m_k|^2 = \sum_{k=1}^{n} (|m_k| \cdot |m_k|) \leqslant (\sup_i |m_i|) \cdot \sum_{k=1}^{n} |m_k| = \|M\|_\infty \cdot \|M\|_1. \tag{C.9}$$

■

D

Spectral Graph Theory

Spectral graph theory studies the relation between eigenvalues of graph matrices and combinatorial properties of the graphs they represent. The reported definitions and proofs are very well known results and are mostly taken from [V+13] and [Tre14].

D.1 Graph Laplacian and its eigenvalues

Lemma D.1.1. *Let $G = (V, E)$ be a graph and L its Laplacian matrix. Let's define the L_e matrices as follows: for every $e = (i, j) \in E$ let $L_e(i, j) = L_e(j, i) = -1$, $L_e(i, i) = L_e(j, j) = 1$, $L_e(x, y) = 0$, with $x, y \notin \{i, j\}$. Then $L = \sum_{e \in E} L_e$.*

Proof. It follows directly from the definitions of L_e's and L. ■

Lemma D.1.2. *Let L be the Laplacian matrix of a graph $G = (V, E)$. Then L is* positive semidefinite *(PSD), i.e. $\lambda_i \geqslant 0$, for all $i = 1, \ldots, n$.*

Proof. Given any vector $\mathbf{v} = (v_1, \ldots, v_n)$, then it holds that

$$\mathbf{v}^\intercal L \mathbf{v} = \mathbf{v}^\intercal \sum_{e \in E} L_e \mathbf{v} = \sum_{e \in E} \mathbf{v}^\intercal L_e \mathbf{v} = \sum_{e=(i,j) \in E} (v_i - v_j)^2 \geqslant 0. \tag{D.1}$$

Then $\min_{\mathbf{v} \neq 0} \mathbf{v}^\intercal L \mathbf{v}$ and it follows from Theorem C.1.4 that L is PSD. ■

Notice that any Laplacian L has $\lambda_1(L) = 0$. In fact considering $\mathbf{1}$ as the vector of all 1s it is easy to check that $L \cdot \mathbf{1} = 0$, that 0 is then an eigenvalue of L, and from Lemma D.1.2 that $\lambda_1(L) = 0$. The second eigenvalue of the Laplacian, as said before, has an important role in spectral graph theory and here we will see why.

Definition D.1.1. *Given a graph $G = (V, E)$ we define its* normalized Laplacian matrix *as $L = I - \frac{1}{d} A$.*

The above definition, which is more convenient, will be used in the following theorem.

Theorem D.1.1. *Let $G = (V, E)$ a d-regular graph, and L its normalized Laplacian matrix. Let $\lambda_1 \leqslant \ldots \leqslant \lambda_n$ its eigenvalues, counted with multiplicities. Then the following statements hold:*

1. $\lambda_1 = 0$ *and* $\lambda_n \leqslant 2$*;*

2. $\lambda_k = 0$ *if and only if* G *has at least* k *connected components;*

3. $\lambda_n = 2$ *if and only if one of the connected components of* G *is bipartite.*

The proof can be found in [Tre14].

As introduced before, the second smallest eigenvalue of the Laplacian of a graph is called *algebraic connectivity* and has an important role in measuring the connectivity of a graph. Fiedler showed its properties in [Fie73, Fie89] and to used the so called *Fiedler vector* to prove its theorem.

D.2 Relating cuts and eigenvalues

Theorem D.2.1 (Cheeger's inequality [Chu96])**.** *Let* P *be the transition matrix of a connected edge-weighted graph* $G = (V, E, w)$ *and let* λ_2 *be its second largest eigenvalue. Let* $|E(S, V\backslash S)| = \sum_{u\in S,\, v\in V\backslash S} w(u, v)$ *and* $h_G = \min_{S:\mathrm{vol}(S)\leqslant \frac{\mathrm{vol}(V)}{2}} \frac{|E(S,V\backslash S)|}{\mathrm{vol}(S)}$*. Then*

$$\frac{1-\lambda_2}{2} \leqslant h_G \leqslant \sqrt{2(1-\lambda_2)}. \tag{D.2}$$

Lemma D.2.1 (Expander Mixing Lemma)**.** *Let* $G = (V, E)$ *be a* d*-regular graph. Let* $S \subseteq V$ *and* $T \subseteq V$*. Then it holds that*

$$\left| e(S,T) - \frac{d|S||T|}{n} \right| \leqslant \lambda\sqrt{|S||T|}, \tag{D.3}$$

where λ *is the second largest eigenvalue, in absolute value, of the adjacency matrix of* G*.*

Lemma D.2.2 (Expander Mixing Lemma – Bipartite graphs)**.** *Let* $G = ((L, R), E)$ *be a* d*-regular bipartite graph. Let* $S \subseteq L$ *and* $T \subseteq R$ *with* $|S| = \alpha L$ *and* $|T| = \beta R$*. Then it holds that*

$$\left| \frac{e(S,T)}{|E|} - \alpha\beta \right| \leqslant \frac{\lambda}{d}\sqrt{\alpha\beta}, \tag{D.4}$$

where λ *is the second largest eigenvalue, in absolute value, of the adjacency matrix of* G*.*

E

Probability Theory and Stochastic Processes

Herein we introduce the basics of probability theory and of stochastic processes, with a focus Markov chains and random walks.

E.1 Events, probability, random variables

We start with the foundations of probability, including events, probability spaces, random variables, and some basic theorems that will be used later. All the definitions, theorems, and lemmas are taken from [MU17].

Axioms of probability

Definition E.1.1. *A probability space is composed of:*

1. *a sample space Ω, which is the set of all possible outcomes of the random process that the space models;*
2. *a family of sets $\mathcal{F}$ representing the possible events; each set in $\mathcal{F}$ is a subset of the sample space Ω;*
3. *a probability function $\mathbf{P} : \mathcal{F} \to \mathbb{R}$, satisfying Definition E.1.2.*

Definition E.1.2. *A probability function is any function $\mathbf{P} : \mathcal{F} \to \mathbb{R}$ that satisfies the following conditions:*

1. *for any event E, $0 \leqslant \mathbf{P}(E) \leqslant 1$;*
2. *$\mathbf{P}(\Omega) = 1$;*
3. *for any finite or countably infinite sequence of pairwise mutually disjoint events it holds that*

$$\mathbf{P}\left(\bigcup_{i \geqslant 1} E_i\right) = \sum_{i \geqslant 1} E_i. \tag{E.1}$$

Lemma E.1.1. *For any two events E_1 and E_2*

$$\mathbf{P}(E_1 \cup E_2) = \mathbf{P}(E_1) + \mathbf{P}(E_2) - \mathbf{P}(E_1 \cap E_2). \tag{E.2}$$

Lemma E.1.2 (Union bound)**.** *For any finite or countably infinite sequence of events*

$$\mathbf{P}\left(\bigcup_{i\geqslant 1} E_i\right) \leqslant \sum_{i\geqslant 1} E_i. \tag{E.3}$$

Proof. This intuitively follows from the generalization of Lemma E.1.1. The probability of the union, as in the previous case, is exactly equal to the sum when the events are disjoints. ■

Definition E.1.3 (Independent events)**.** *Two events E and F are independent if and only if*

$$\mathbf{P}(E \cap F) = \mathbf{P}(E) \cdot \mathbf{P}(F). \tag{E.4}$$

In general k events are mutually independent if and only if

$$\mathbf{P}\left(\bigcap_{i=1}^{k} E_i\right) = \prod_{i=1}^{k} \mathbf{P}(E_i). \tag{E.5}$$

Definition E.1.4 (Conditional probability)**.** *The conditional probability that event E occurs, given that event F occurs, is*

$$\mathbf{P}(E \,|\, F) = \frac{\mathbf{P}(E \cap F)}{\mathbf{P}(F)}. \tag{E.6}$$

Such probability, obviously, is well-defined only if $\mathbf{P}(F) > 0$.

Theorem E.1.1 (Law of total probability)**.** *Let $E_1, \ldots, E_n$ be mutually disjoint events in the sample space Ω and let $\bigcup_{i=1}^{n} E_i = \Omega$. Then*

$$\mathbf{P}(A) = \sum_{i=1}^{n} \mathbf{P}(A \cap E_i) = \sum_{i=1}^{n} \mathbf{P}(A \,|\, E_i)\mathbf{P}(E_i). \tag{E.7}$$

Theorem E.1.2 (Bayes' law)**.** *Assume that $E_1, \ldots, E_n$ are mutually disjoint sets such that $\bigcup_{i=1}^{n} E_i = E$. Then*

$$\mathbf{P}(E_j \,|\, A) = \frac{\mathbf{P}(E_j \cap A)}{\mathbf{P}(A)} = \frac{\mathbf{P}(A \,|\, E_j)\mathbf{P}(E_j)}{\sum_{i=1}^{n} \mathbf{P}(A \,|\, E_i)\mathbf{P}(E_i)}. \tag{E.8}$$

Definition E.1.5 (Event with high probability)**.** *We say that an event $\mathcal{E}_n$ occurs* with high probability *(*w.h.p.*, in short) if $\mathbf{P}(\mathcal{E}_n) \geqslant 1 - \mathcal{O}(n^{-\gamma})$, for some positive constant γ.*

Random variables and expectation

Definition E.1.6 (Random variable)**.** *A* random variable X *on a sample space* Ω *is a function* $X : \Omega \to F$*, where* F *is some set. A* discrete random variable *is a random variable such that* F *is finite or countably infinite.*

Definition E.1.7 (Expectation)**.** *The* expectation *or* expected value *of a discrete random variable* X *is given by*

$$\mathbf{E}[X] = \sum_{i \in X} i \, \mathbf{P}(X = i). \tag{E.9}$$

Theorem E.1.3 (Linearity of expectation)**.** *For any finite collection of discrete random variables*

$$\mathbf{E}\left[\sum_{i=1}^{n} X_i\right] = \sum_{i=1}^{n} \mathbf{E}[X_i]. \tag{E.10}$$

Lemma E.1.3. *For any constant* c *and discrete random variable* X

$$\mathbf{E}[cX] = c\mathbf{E}[X]. \tag{E.11}$$

E.2 Markov chains and random walks

Markov chains are an extremely useful tool for modeling random processes and are at the base of many algorithms, such as PageRank. In this section we report some basic definitions, which are mainly taken from [MU17] and [LP17].

Definition E.2.1. *A* stochastic process $\mathbf{X} = \{X_t : t \in T\}$ *is a collection of random variables, denoted with* X*, that change their state* X_t *over time* t*.*

If X assumes only discrete values we call $\mathbf{X}$ a *discrete space process*. If X_t can only assume a finite number of values, for all t, then we call the process *finite*. Whenever T is a countably infinite set, we call $\mathbf{X}$ a *discrete time process*.

Markov chains

Markov chains are discrete time stochastic processes that do not remember their past history and such that their evolution only depend on their actual state. Here we give a formal definition.

Definition E.2.2. *A discrete time stochastic process* $\mathbf{X}$ *is a Markov chain if*

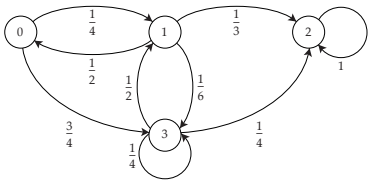
$$\begin{aligned} &\mathbf{P}(X_t = a_t \,|\, X_{t-1} = a_{t-1}, \ldots, X_0 = a_0) \\ &\qquad = \mathbf{P}(X_t = a_t \,|\, X_{t-1} = a_{t-1}) = \mathbf{P}_{a_{t-1}, a_t}. \end{aligned} \tag{E.12}$$

Such property, prerogative of Markov chain, is called *Markov property* or *memoryless property*.

Observe that every *Markovian process*, i.e., every process having such property, can be described by a transition matrix: Each of its elements, e.g., $P_{i,j}$, describes the probability of moving from state i to state j in one time step and is sufficient to describe the whole process because of the memoryless property. The rows of the transition matrix are probability distributions and make such matrix *stochastic*, meaning that its entries are non-negative, being probabilities, and sum to 1, since are probability distributions. Such matrix is the representation of a graph (more precisely a weighted digraph) and an example of it can be seen in Figure E.1.[1]

$$P = \begin{pmatrix} 0 & \frac{1}{4} & 0 & \frac{3}{4} \\ \frac{1}{2} & 0 & \frac{1}{3} & \frac{1}{6} \\ 0 & 0 & 1 & 0 \\ 0 & \frac{1}{2} & \frac{1}{4} & \frac{1}{4} \end{pmatrix}$$

(a) Transition matrix.



(b) Markov digraph.

Figure E.1: An example of Markov chain represented as transition matrix (a) and associated weighted digraph (b).

If we denote with $p_i^{(t)}$ the probability of the process of being at state i at time t, then we can compute such value as $p_i^{(t)} = \sum_{j \geqslant 0} p_j^{(t-1)} P_{j,i}$. Using vector notation we can define $\pi^{(t)} = (p_0^{(t)}, p_1^{(t)}, \ldots)$ and rewrite the above equation as follows

$$\pi^{(t)} = \pi^{(t-1)} P = \pi^{(0)} P^t. \tag{E.13}$$

So we can see how multiplying a row vector, representing a probability distribution, by P on the right side takes us from the current distribution (at time t) to the next one (time $t+1$). Instead if we consider a column vector μ, representing a function of the state space, we get that multiplying it by P on the left side gives us the expected value of such function on the next state.

Definition E.2.3 (Irreducibility)**.** *A Markov chain is* irreducible *if for any two states x, y there exists an integer t such that $P_{x,y}^t > 0$, i.e., there exists a directed path in the Markov digraph from x to y.*

Definition E.2.4. *Let $\mathcal{T}_x = \{t > 0 : P_{x,x}^t > 0\}$ the set of time for which a Markov chain P can return to its initial state x. The greatest common divisor of $\mathcal{T}_x$ is called the* period *of the state x.*

Definition E.2.5 (Aperiodicity)**.** *Let P be a Markov chain. If all its state have period 1, then we say that P is* aperiodic*.*

Lemma E.2.1. *Let P be a Markov chain. If P is irreducible then all the states have the same period.*

[1] Source: [MU17]

Theorem E.2.1. *If a Markov chain P is both irreducible and aperiodic, then there exists an integer r such that $P^r_{x,y} > 0$ for all states x, y.*

Definition E.2.6. *A state of a Markov chain is said to be* recurrent *whenever the chain will eventually return to it, once visited, with probability 1. Otherwise a state is said to be* transient*. A state is* positive recurrent *if the expected time of returning to it, after its first visit, is finite.*

Definition E.2.7 (Ergodicity)**.** *An aperiodic, positive recurrent state is said to be* ergodic*. If all the states of a Markov chain are* ergodic*, then the Markov chain is* ergodic*.*

Corollary E.2.1 (Ergodic Markov Chains)**.** *Any finite, irreducible, and aperiodic Markov chain is* ergodic*.*

Stationary distributions

Definition E.2.8 (Stationary distribution)**.** *A* stationary distribution *of a Markov chain is a row vector probability distribution $\bar{\pi}$ such that*

$$\bar{\pi} = \bar{\pi} P. \tag{E.14}$$

Such definition implies that whenever a Markov chain reaches such distributions it will stay on that forever. Hence, such distribution is a *steady-state* of the Markov chain. We call the time t in which a Markov chain reaches a stationary distribution *mixing time* of the Markov chain. Now we present a fundamental theorem which is at the basis of the analysis of Markov chains.

Theorem E.2.2. *For any ergodic Markov chain, the following statements hold:*

1. *it has a unique stationary distribution $\bar{\pi} = (\pi_0, \ldots, \pi_n)$;*
2. *for all i, j, the limit $\lim_{t\to\infty} P^t_{j,i}$ exists and is independent from j;*
3. *$\bar{\pi}_i = \lim_{t\to\infty} P^t_{j,i} = \frac{1}{h_{i,i}}$, where $h_{i,i}$ is the expected number of steps for the Markov chain, starting from state i, to return to state i.*

Random walks

Random walks on graphs are a particular type of Markov chains and are especially useful to analyze algorithms.

Definition E.2.9. *A* random walk *on a connected graph $G = (V, E)$ is a Markov chain defined by the sequence of positions among the vertices of G, which compose the state space. The transition matrix of a* random walk *is the transition matrix associated to G. Thus, if the state is vertex v the next state will be a neighbor of v and the probability of following any incident edge will be $\frac{1}{d_v}$.*

Lemma E.2.2. *A random walk on a graph G is aperiodic if and only if G is not bipartite.*

Theorem E.2.3. *A random walk on a not bipartite graph G converges to a stationary distribution $\bar{\pi}$, where*

$$\pi_v = \frac{d_v}{2|E|}. \tag{E.15}$$

Proof. Since $\sum_{v \in V} d_v = 2|E|$ it follows that

$$\sum_{v \in V} \pi_v = \sum_{v \in V} d_v = \frac{d_v}{2|E|} = 1. \tag{E.16}$$

Let's define $N(v)$ as the set of neighbors of node v. Using the stationary distribution equation $\bar{\pi} = \bar{\pi} P$ we get

$$\pi_v = \sum_{u \in N(v)} \frac{d_u}{2|E|} P_{u,v} = \sum_{u \in N(v)} \frac{d_u}{2|E|} \frac{1}{d_u} = \frac{d_v}{2|E|}. \tag{E.17}$$

■

E.3 Concentration bounds and other useful results

Theorem E.3.1 (Berry–Esseen [She14])**.** *Let $X_1, \ldots, X_n$ be independent and identically distributed random variables with mean $\mu = 0$, variance $\sigma^2 > 0$, and third absolute moment $\rho < \infty$. Let $Y_n = \frac{1}{n} \sum_{i=1}^{n} X_i$; let F_n be the cumulative distribution function of $\frac{Y_n \sqrt{n}}{\sigma}$; let Φ the cumulative distribution function of the standard normal distribution. Then, there exists a positive constant $C < 0.4748$ such that, for all x and for all n,*

$$|F_n(x) - \Phi(x)| \leqslant \frac{C\rho}{\sigma^3 \sqrt{n}}. \tag{E.18}$$

Theorem E.3.2 (Le Cam)**.** *Let $X_1, \ldots, X_n$ be independent Bernoulli random variables and let p_i the probability of having $X_i = 1$. Let $\lambda = \sum_{i=1}^{n} p_i$ and let $Y = \sum_{i=1}^{n} X_i$ be a Poisson random variable. Then*

$$\sum_{k=0}^{\infty} \left| \mathbf{P}\left(Y = k\right) - \frac{\lambda^k e^{-\lambda}}{k!} \right| < 2 \sum_{i=1}^{n} p_i^2. \tag{E.19}$$

Theorem E.3.3 (Littlewood–Offord's small ball [Erd45])**.** *Let x_i be a Rademacher random variable (taking values ± 1 with probability $p = \frac{1}{2}$), let a_i be real constants such that $|a_i| \geqslant 1$, and let $X = \sum_{i=1}^{n} a_i x_i$. Then, for any $r \in \mathbb{R}$, it holds that*

$$\mathbf{P}(|X - r| < 1) = \mathcal{O}\left(\frac{1}{\sqrt{n}}\right). \tag{E.20}$$

Theorem E.3.4 (Rademacher concentration bound [MS90])**.** *Let x_i be a Rademacher random variable (taking values ± 1 with probability $p = \frac{1}{2}$), let a_i be real constants, and let $X = \sum_{i=1}^{n} a_i x_i$. Then, it holds that*

$$\mathbf{P}\left(|X| > t \|\boldsymbol{a}\|_2\right) \leqslant 2e^{-\frac{t^2}{2}}. \tag{E.21}$$

where $\|\boldsymbol{a}\|_2$ is the Euclidean norm of the vector $\boldsymbol{a} = (a_1, \ldots, a_n)$.

Theorem E.3.5 (Chernoff — Additive)**.** *Let $X_1, \ldots, X_n$ be independent Bernoulli random variables, let $X = \sum_{i=1}^{n} X_i$, and let $\mathbf{E}[X] = \mu$. Then:*

$$\mathbf{P}(X \leqslant \mu - \lambda) \leqslant e^{-2\lambda^2/n}, 0 < \lambda < n - \mu; \tag{E.22}$$

$$\mathbf{P}(X \geqslant \mu + \lambda) \leqslant e^{-2\lambda^2/n}, 0 < \lambda < \mu. \tag{E.23}$$

Theorem E.3.6 (Chernoff — Multiplicative)**.** *Let $X_1, \ldots, X_n$ be n independent Bernoulli random variables, let $X = \sum_{i=1}^{n} X_i$, and let $\mathbf{E}[X] = \mu$. Then:*

$$\mathbf{P}(X \leqslant (1-\delta)\mu) \leqslant e^{-\delta^2\mu/2}, 0 \leqslant \delta \leqslant 1; \tag{E.24}$$

$$\mathbf{P}(X \geqslant (1+\delta)\mu) \leqslant e^{-\delta^2\mu/3}, 0 \leqslant \delta \leqslant 1. \tag{E.25}$$

Lemma E.3.1 (Lemma 4.5 [CGG+18])**.** *Let $\{X_t\}_{t\in\mathbb{N}}$ be a Markov Chain with finite state space Ω and let $f : \Omega \mapsto [0, n]$ be a function that maps states to integer values. Let c_3 be any positive constant and let $m = c_3\sqrt{n}\log n$ be a target value. Assume the following properties hold:*

1. *For any positive constant h, there exists a positive constant $c_1 < 1$ such that, for any $x \in \Omega : f(x) < m$,*

$$\mathbf{P}\left(f(X_{t+1}) < h\sqrt{n} \;\middle|\; X_t = x\right) < c_1. \tag{E.26}$$

2. *There exist two positive constants ε, c_2 such that, for any $x \in \Omega : h\sqrt{n} \leqslant f(x) < m$,*

$$\mathbf{P}(f(X_{t+1}) < (1+\varepsilon)f(X_t) \mid X_t = x) < e^{-c_2 f(x)^2/n}. \tag{E.27}$$

Then the process reaches a state x such that $f(x) \geqslant m$ within $\mathcal{O}(\log n)$ rounds, w.h.p.

Bibliography

[AAB+11] Yehuda Afek, Noga Alon, Omer Barad, Eran Hornstein, Naama Barkai, and Ziv Bar-Joseph. A biological solution to a fundamental distributed computing problem. *Science (New York, N.Y.)*, 331(6014):183–185, January 2011.

[AAD+06] Dana Angluin, James Aspnes, Zoë Diamadi, Michael J. Fischer, and René Peralta. Computation in networks of passively mobile finite-state sensors. *Distributed computing*, 18(4):235–253, 2006.

[AFJ06] Dana Angluin, Michael J. Fischer, and Hong Jiang. Stabilizing consensus in mobile networks. In *Proc. of Distributed Computing in Sensor Systems (DCOSS'06)*, volume 4026 of *LNCS*, pages 37–50, 2006.

[AFPP12] Vincenzo Auletta, Diodato Ferraioli, Francesco Pasquale, and Giuseppe Persiano. Metastability of Logit Dynamics for Coordination Games. In *33rd Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1006–1024, Kyoto, Japan, 2012. SIAM.

[ALP+14] Chen Avin, Zvi Lotker, David Peleg, Yvonne Anne Pignolet, and Itzik Turkel. Core-Periphery in Networks: An Axiomatic Approach. *arXiv:1411.2242 [physics]*, November 2014. arXiv: 1411.2242.

[BBV08] Alain Barrat, Marc Barthelemy, and Alessandro Vespignani. *Dynamical processes on complex networks*. Cambridge university press, 2008.

[BC09] Michael J. Barber and John W. Clark. Detecting network communities by propagating labels under constraints. *Phys. Rev. E*, 80:026129, Aug 2009.

[BCM+18] Luca Becchetti, Andrea E.F. Clementi, Pasin Manurangsi, Emanuele Natale, Francesco Pasquale, Prasad Raghavendra, and Luca Trevisan. Average whenever you meet: Opportunistic protocols for community detection. In *26th Annual European Symposium on Algorithms, ESA 2018, August 20-22, 2018, Helsinki, Finland*, pages 7:1–7:13, 2018.

[BCN+15] Luca Becchetti, Andrea Clementi, Emanuele Natale, Francesco Pasquale, and Riccardo Silvestri. Plurality consensus in the gossip

model. In *Proceedings of the 26th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA'15)*, pages 371–390. SIAM, 2015.

[BCN+16] Luca Becchetti, Andrea E. F. Clementi, Emanuele Natale, Francesco Pasquale, and Luca Trevisan. Stabilizing consensus with many opinions. In *Proceedings of the Twenty-27th Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2016, Arlington, VA, USA, January 10-12, 2016*, pages 620–635, 2016.

[BCN+17a] Luca Becchetti, Andrea Clementi, Emanuele Natale, Francesco Pasquale, Riccardo Silvestri, and Luca Trevisan. Simple dynamics for plurality consensus. *Distributed Computing*, 30(4), August 2017.

[BCN+17b] Luca Becchetti, Andrea Clementi, Emanuele Natale, Francesco Pasquale, and Luca Trevisan. Find your place: Simple distributed algorithms for community detection. In *Proceedings of the 28th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 940–959. Society for Industrial and Applied Mathematics, 2017.

[BCPR19] Luca Becchetti, Emilio Cruciani, Francesco Pasquale, and Sara Rizzo. Step-by-step community detection in volume-regular graphs. In *Proceedings of the 30th International Symposium on Algorithms and Computation, ISAAC 2019, Shanghai, China, December 8-11, 2019*, page to appear, 2019.

[BDSY99] Amir Ben-Dor, Ron Shamir, and Zohar Yakhini. Clustering gene expression patterns. *Journal of computational biology*, 6(3-4):281–297, 1999.

[BE00] Stephen P Borgatti and Martin G Everett. Models of core/periphery structures. *Social Networks*, 21(4):375 – 395, 2000.

[BF07] Daniel I. Bolnick and Benjamin M. Fitzpatrick. Sympatric Speciation: Models and Empirical Evidence. *Annual Review of Ecology, Evolution, and Systematics*, 38:459–487, 2007.

[BGKMT16] Petra Berenbrink, George Giakkoupis, Anne-Marie Kermarrec, and Frederik Mallmann-Trenn. Bounds on the Voter Model in Dynamic Networks. In Ioannis Chatzigiannakis, Michael Mitzenmacher, Yuval Rabani, and Davide Sangiorgi, editors, *43rd International Colloquium on Automata, Languages, and Programming (ICALP 2016)*, volume 55 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 146:1–146:15, Dagstuhl, Germany, 2016. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik.

[BGLL08] Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. Fast unfolding of communities in large networks. *Journal of statistical mechanics: theory and experiment*, 2008(10):P10008, 2008.

[BGPS06] Stephen Boyd, Arpita Ghosh, Balaji Prabhakar, and Devavrat Shah. Randomized gossip algorithms. *IEEE/ACM Transactions on Networking (TON)*, 14:2508–2530, 2006.

[BGW03] Ulrik Brandes, Marco Gaertler, and Dorothea Wagner. Experiments on graph clustering algorithms. In *Algorithms - ESA 2003, 11th Annual European Symposium, Budapest, Hungary, September 16-19, 2003, Proceedings*, pages 568–579, 2003.

[BKN17] Lucas Boczkowski, Amos Korman, and Emanuele Natale. Minimizing Message Size in Stochastic Communication Patterns: Fast Self-Stabilizing Protocols with 3 bits. In *28th Annual ACM-SIAM Symposium on Discrete Algorithms*, January 2017.

[Bol98] Béla Bollobás. Random graphs. In *Modern Graph Theory*, pages 215–252. Springer, 1998.

[BRSV11] Paolo Boldi, Marco Rosa, Massimo Santini, and Sebastiano Vigna. Layered label propagation: A multiresolution coordinate-free ordering for compressing social networks. In *Proceedings of the 20th international conference on World wide web*, pages 587–596. ACM, 2011.

[CCDP19a] Federico Corò, Emilio Cruciani, Gianlorenzo D'Angelo, and Stefano Ponziani. Exploiting social influence to control elections based on scoring rules. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence, IJCAI 2019, Macau, China, August 10-16, 2019*, page to appear, 2019.

[CCDP19b] Federico Corò, Emilio Cruciani, Gianlorenzo D'Angelo, and Stefano Ponziani. Vote for me! election control via social influence in arbitrary scoring rule voting systems. In *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems, AAMAS 2019, Montreal, QC, Canada, May 13-17, 2019*, pages 1895–1897, 2019.

[CDIG+15] Andrea Clementi, Miriam Di Ianni, Giorgio Gambosi, Emanuele Natale, and Riccardo Silvestri. Distributed community detection in dynamic graphs. *Theoretical Computer Science*, 2015.

[CDRR16] Sarah Cannon, Joshua J. Daymude, Dana Randall, and Andrea Richa. A markov chain algorithm for compression in self-organizing particle

systems. In *PODC 2016 - Proceedings of the 2016 ACM Symposium on Principles of Distributed Computing*, volume 25-28-July-2016, pages 279–288. Association for Computing Machinery, 7 2016.

[CER14] Colin Cooper, Robert Elsässer, and Tomasz Radzik. The power of two choices in distributed voting. In *Automata, Languages, and Programming - 41st International Colloquium, ICALP 2014, Copenhagen, Denmark, July 8-11, 2014, Proceedings, Part II*, pages 435–446, 2014.

[CER+15] Colin Cooper, Robert Elsässer, Tomasz Radzik, Nicolas Rivera, and Takeharu Shiraga. Fast consensus for voting on general expander graphs. In *Distributed Computing - 29th International Symposium, DISC 2015, Tokyo, Japan, October 7-9, 2015, Proceedings*, pages 248–262, 2015.

[CG10] Gennaro Cordasco and Luisa Gargano. Community detection via semi-synchronous label propagation algorithms. In *Business Applications of Social Network Analysis (BASNA), 2010 IEEE International Workshop on*, pages 1–8. IEEE, 2010.

[CGG+18] Andrea E. F. Clementi, Mohsen Ghaffari, Luciano Gualà, Emanuele Natale, Francesco Pasquale, and Giacomo Scornavacca. A Tight Analysis of the Parallel Undecided-State Dynamics with Two Colors. In *43st International Symposium on Mathematical Foundations of Computer Science*, 2018.

[Cha00] Moses Charikar. *Greedy Approximation Algorithms for Finding Dense Components in a Graph*, pages 84–95. Springer Berlin Heidelberg, Berlin, Heidelberg, 2000.

[Che13] Hongsong Chen. Networks, crowds, and markets: Reasoning about a highly connected world [book review]. *IEEE Technol. Soc. Mag.*, 32(3):10, 2013.

[Chu96] Fan R.K. Chung. Laplacians of graphs and Cheeger's inequalities. In *Combinatorics, Paul Erdös is eighty*, volume 2, pages 157–172. János Bolyai Math. Soc., 1996.

[CKW18] Luca Cardelli, Marta Kwiatkowska, and Max Whitby. Chemical reaction network designs for asynchronous logic circuits. *Natural Computing*, 17(1):109–130, 2018.

[CMVB19] Emilio Cruciani, Breno Miranda, Roberto Verdecchia, and Antonia Bertolino. Scalable approaches for test suite reduction. In *Proceedings of the 41st International Conference on Software Engineering, ICSE 2019, Montreal, QC, Canada, May 25-31, 2019*, pages 419–429, 2019.

[CNM04] Aaron Clauset, M. E. J. Newman, and Cristopher Moore. Finding community structure in very large networks. 70(6), 2004.

[CNNS18a] Emilio Cruciani, Emanuele Natale, André Nusser, and Giacomo Scornavacca. On the emergent behavior of the 2-choices dynamics. In *Proceedings of the 19th Italian Conference on Theoretical Computer Science, Urbino, Italy, September 18-20, 2018.*, pages 60–64, 2018.

[CNNS18b] Emilio Cruciani, Emanuele Natale, André Nusser, and Giacomo Scornavacca. Phase transition of the 2-choices dynamics on core-periphery networks. In *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems, AAMAS 2018, Stockholm, Sweden, July 10-15, 2018*, pages 777–785, 2018.

[CNNS18c] Emilio Cruciani, Emanuele Natale, André Nusser, and Giacomo Scornavacca. Phase transition of the 2-choices dynamics on core-periphery networks. *Bulletin of the EATCS*, 125, 2018.

[CNS18] Emilio Cruciani, Emanuele Natale, and Giacomo Scornavacca. On the metastability of quadratic majority dynamics on clustered graphs and its biological implications. *Bulletin of the EATCS*, 125, 2018.

[CNS19] Emilio Cruciani, Emanuele Natale, and Giacomo Scornavacca. Distributed community detection via metastability of the 2-choices dynamics. In *Proceedings of the 33rd AAAI Conference on Artificial Intelligence, AAAI 2019, Honolulu, HI, USA, January 27 - February 1, 2019*, page to appear, 2019.

[CO04] Jerry A Coyne and H Allen Orr. *Speciation*. Sinauer Associates, Inc, Sunderland, Mass, 1 edition edition, 2004.

[CRRS17] Colin Cooper, Tomasz Radzik, Nicolas Rivera, and Takeharu Shiraga. Fast plurality consensus in regular expanders. In *31st International Symposium on Distributed Computing, DISC 2017, October 16-20, 2017, Vienna, Austria*, pages 13:1–13:16, 2017.

[CSWB09] Matthew Cook, David Soloveichik, Erik Winfree, and Jehoshua Bruck. *Programmability of Chemical Reaction Networks*, pages 543–584. Springer Berlin Heidelberg, Berlin, Heidelberg, 2009.

[DDG+14] Zahra Derakhshandeh, Shlomi Dolev, Robert Gmyr, Andréa W. Richa, Christian Scheideler, and Thim Strothmann. Brief announcement: Amoebot – a new model for programmable matter. In *Proceedings of the 26th ACM Symposium on Parallelism in Algorithms and Architectures*, SPAA '14, pages 220–222, New York, NY, USA, 2014. ACM.

[DGM+11] Benjamin Doerr, Leslie Ann Goldberg, Lorenz Minder, Thomas Sauerwald, and Christian Scheideler. Stabilizing consensus with the power of two choices. In *Proceedings of the Twenty-third Annual ACM Symposium on Parallelism in Algorithms and Architectures*, SPAA '11, pages 149–158, New York, NY, USA, 2011. ACM.

[Dij74] Edsger W. Dijkstra. Self-stabilizing systems in spite of distributed control. *Commun. ACM*, 17(11):643–644, November 1974.

[DJ10] Easley David and Kleinberg Jon. *Networks, Crowds, and Markets: Reasoning About a Highly Connected World*. Cambridge University Press, New York, NY, USA, 2010.

[Dot14] David Doty. Timing in chemical reaction networks. In *Proc. of 25th Ann. ACM-SIAM Symp. on Discrete Algorithms (SODA'14)*, pages 772–784. SIAM, 2014.

[DP09] Devdatt Dubhashi and Alessandro Panconesi. *Concentration of Measure for the Analysis of Randomized Algorithms*. Cambridge University Press, New York, NY, USA, 1st edition, 2009.

[Dur11] Richard Durrett. *By Richard Durrett - Probability Models for DNA Sequence Evolution: 2nd (second) Edition*. Springer-Verlag New York, LLC, November 2011.

[EFK+16] Robert Elsässer, Tom Friedetzky, Dominik Kaaser, Frederik Mallmann-Trenn, and Horst Trinker. Rapid Asynchronous Plurality Consensus. *arXiv:1602.04667 [cs]*, February 2016. arXiv: 1602.04667.

[EJN02] Mark E. J. Newman. Spread of epidemic disease on networks. 66:016128, 08 2002.

[ER60] Paul Erdos and Alfréd Rényi. On the evolution of random graphs. *Publ. Math. Inst. Hung. Acad. Sci*, 5(1):17–60, 1960.

[Erd45] Paul Erdös. On a lemma of littlewood and offord. *Bulletin of the American Mathematical Society*, 51(12):898–902, 1945.

[FB07] Santo Fortunato and Marc Barthélemy. Resolution limit in community detection. *Proceedings of the National Academy of Sciences*, 104(1):36–41, 2007.

[FH16] Santo Fortunato and Darko Hric. Community detection in networks: A user guide. 659:1–44, 2016.

[Fie73] Miroslav Fiedler. Algebraic connectivity of graphs. *Czechoslovak mathematical journal*, 23(2):298–305, 1973.

[Fie89] Miroslav Fiedler. Laplacian of graphs and algebraic connectivity. *Banach Center Publications*, 25(1):57–70, 1989.

[FN16] Pierre Fraigniaud and Emanuele Natale. Noisy Rumor Spreading and Plurality Consensus. In *ACM Symposium on Principles of Distributed Computing*, pages 127–136, 2016.

[For10] Santo Fortunato. Community detection in graphs. *Physics Reports*, 486(3):75–174, 2010.

[FTT04] Gary William Flake, Robert E Tarjan, and Kostas Tsioutsiouliklis. Graph clustering and minimum cut trees. *Internet Mathematics*, 1(4):385–408, 2004.

[FV15] Diodato Ferraioli and Carmine Ventre. Metastability of Asymptotically Well-Behaved Potential Games. In *Mathematical Foundations of Computer Science 2015*, Lecture Notes in Computer Science, pages 311–323. Springer Berlin Heidelberg, 2015.

[GGG+17] Andreas Galanis, Andreas Göbel, Leslie Ann Goldberg, John Lapinskas, and David Richerby. Amplifiers for the Moran Process. *J. ACM*, 64(1):5:1–5:90, March 2017.

[GH61] Ralph E Gomory and Tien Chung Hu. Multi-terminal network flows. *Journal of the Society for Industrial and Applied Mathematics*, 9(4):551–570, 1961.

[Gia16] George Giakkoupis. Amplifiers and Suppressors of Selection for the Moran Process on Undirected Graphs. *arXiv:1611.01585 [cs, math, q-bio]*, November 2016. arXiv: 1611.01585.

[GL98] Wen-Biao Gan and Jeff W. Lichtman. Synaptic Segregation at the Developing Neuromuscular Junction. *Science*, 282(5393):1508–1511, November 1998.

[GL17] Mohsen Ghaffari and Johannes Lengler. Tight analysis for the 3-majority consensus dynamics. *CoRR*, abs/1705.05583, 2017.

[Gre10] Steve Gregory. Finding overlapping communities in networks by label propagation. *New Journal of Physics*, 12(10):103018, 2010.

[GS19] Sara E. Garza and Satu Elisa Schaeffer. Community detection with the Label Propagation Algorithm: A survey. *Physica A: Statistical Mechanics and its Applications*, page 122058, July 2019.

[Het00] Herbert W. Hethcote. The mathematics of infectious diseases. *SIAM Review*, 42(4):599–653, 2000.

[HLL83] Paul W. Holland, Kathryn Blackmond Laskey, and Samuel Leinhardt. Stochastic blockmodels: First steps. *Social Networks*, 5(2), June 1983.

[HP01] Yehuda Hassin and David Peleg. Distributed probabilistic polling and applications to proportionate agreement. *Information and Computation*, 171(2):248 – 268, 2001.

[HW04] Adel Hlaoui and Shengrui Wang. A direct approach to graph clustering. *Neural Networks and Computational Intelligence*, 4(8):158–163, 2004.

[KKT15] David Kempe, Jon Kleinberg, and Éva Tardos. Maximizing the spread of influence through a social network. *Theory of Computing*, 11(4):105–147, 2015.

[KL70] Brian W Kernighan and Shen Lin. An efficient heuristic procedure for partitioning graphs. *The Bell system technical journal*, 49(2):291–307, 1970.

[KPS13] Kishore Kothapalli, Sriram V. Pemmaraju, and Vivek Sardeshmukh. On the analysis of a label propagation algorithm for community detection. In *Proc. of the 14th Int. Conf. on Distributed Computing and Networking (ICDCN'13)*, pages 255–269, 2013.

[KS60] John G. Kemeny and J. Laurie Snell. *Finite Markov chains*. D. van Nostrand Company, inc., Princeton, N.J., 1960.

[Kun13] Jérôme Kunegis. Konect: the koblenz network collection. In *Proceedings of the 22nd International Conference on World Wide Web*, pages 1343–1350. ACM, 2013.

[LFR08] Andrea Lancichinetti, Santo Fortunato, and Filippo Radicchi. Benchmark graphs for testing community detection algorithms. *Physical review E*, 78(4):046110, 2008.

[LGT14] James R Lee, Shayan Oveis Gharan, and Luca Trevisan. Multiway spectral partitioning and higher-order cheeger inequalities. *Journal of the ACM (JACM)*, 61(6):37, 2014.

[LHLC09] Ian X. Y. Leung, Pan Hui, Pietro Liò, and Jon Crowcroft. Towards real-time community detection in large networks. *Phys. Rev. E*, 79:066107, Jun 2009.

[LHN05] Erez Lieberman, Christoph Hauert, and Martin A. Nowak. Evolutionary dynamics on graphs. *Nature*, 433(7023):312–316, January 2005.

[Lig06] Thomas M. Liggett. *Interacting Particle Systems*. Springer Science & Business Media, 2012-12-06. Google-Books-ID: 7JbqBwAAQBAJ.

[Lig09] Thomas M. Liggett. *Stochastic Interacting Systems: Contact, Voter and Exclusion Processes*. Springer Science & Business Media, 2013-03-09. Google-Books-ID: wRv2CAAAQBAJ.

[LK14] Jure Leskovec and Andrej Krevl. SNAP Datasets: Stanford large network dataset collection. http://snap.stanford.edu/data, June 2014.

[Llo82] Stuart Lloyd. Least squares quantization in pcm. *IEEE transactions on information theory*, 28(2):129–137, 1982.

[LM09] Xin Liu and Tsuyoshi Murata. How does label propagation algorithm work in bipartite networks? In *IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology - Volume 03*, pages 5–8, 2009.

[LM10] Xin Liu and Tsuyoshi Murata. Advanced modularity-specialized label propagation algorithm for detecting communities in networks. *Physica A: Statistical Mechanics and its Applications*, 389(7):1493 – 1500, 2010.

[LP17] David A Levin and Yuval Peres. *Markov chains and mixing times*, volume 107. American Mathematical Soc., 2017.

[LRU14] Jure Leskovec, Anand Rajaraman, and Jeffrey David Ullman. *Mining of massive datasets*. Cambridge university press, 2014.

[MCC+19] Mohammad Abouei Mehrizi, Federico Corò, Emilio Cruciani, Gianlorenzo D'Angelo, and Stefano Ponziani. Models and algorithms for election control via influence maximization. In *Proceedings of the 20th Italian Conference on Theoretical Computer Science, Como, Italy, September 9-11, 2019.*, page to appear, 2019.

[MCVB18] Breno Miranda, Emilio Cruciani, Roberto Verdecchia, and Antonia Bertolino. FAST approaches to scalable similarity-based test case prioritization. In *Proceedings of the 40th International Conference on Software Engineering, ICSE 2018, Gothenburg, Sweden, May 27 - June 03, 2018*, pages 222–232, 2018.

[MDN+13] Simon H. Martin, Kanchon K. Dasmahapatra, Nicola J. Nadeau, Camilo Salazar, James R. Walters, Fraser Simpson, Mark Blaxter, Andrea Manica, James Mallet, and Chris D. Jiggins. Genome-wide evidence for speciation with gene flow in Heliconius butterflies. *Genome Research*, 23(11):1817–1828, November 2013.

[MNRS17] George B Mertzios, Sotiris E Nikoletseas, Christoforos L Raptopoulos, and Paul G Spirakis. Determining majority in networks with local interactions and very small local memory. *Distributed Computing*, 30(1):1–16, 2017.

[MNT14] Elchanan Mossel, Joe Neeman, and Omer Tamuz. Majority dynamics and aggregation of information in social networks. *Autonomous Agents and Multi-Agent Systems*, 28(3):408–429, May 2014.

[MPG29] RV Mises and Hilda Pollaczek-Geiringer. Praktische verfahren der gleichungsauflösung. *ZAMM-Journal of Applied Mathematics and Mechanics/Zeitschrift für Angewandte Mathematik und Mechanik*, 9(1):58–77, 1929.

[MS90] Stephen J Montgomery-Smith. The distribution of rademacher sums. *Proceedings of the American Mathematical Society*, 109(2):517–522, 1990.

[MS01a] Marina Meila and Jianbo Shi. Learning segmentation by random walks. In *Advances in neural information processing systems*, pages 873–879, 2001.

[MS01b] Marina Meila and Jianbo Shi. A random walks view of spectral segmentation. 2001.

[MT17] Elchanan Mossel and Omer Tamuz. Opinion exchange dynamics. *Probability Surveys*, 14:155–204, 2017.

[MU17] Michael Mitzenmacher and Eli Upfal. *Probability and computing: randomization and probabilistic techniques in algorithms and data analysis*. Cambridge university press, 2017.

[Nat17] Emanuele Natale. *On the Computational Power of Simple Dynamics*. Phd thesis, 2017. Sapienza University of Rome.

[New04] M. E. J. Newman. Fast algorithm for detecting community structure in networks. 69(6), 2004.

[NG04] M. E. J. Newman and M. Girvan. Finding and evaluating community structure in networks. 69(2), 2004.

[PSZ17] Richard Peng, He Sun, and Luca Zanetti. Partitioning well-clustered graphs: Spectral clustering works! *SIAM Journal on Computing*, 46(2):710–743, 2017.

[Rab83] Michael O. Rabin. Randomized byzantine generals. In *Proc. of the 24th Ann. Symp. on Foundations of Computer Science (SFCS'83)*, pages 403–409. IEEE, 1983.

[RAB09] Martin Rosvall, Daniel Axelsson, and Carl T. Bergstrom. The map equation. *The European Physical Journal Special Topics*, 178:13–23, 2009.

[RAK07] Usha Nandini Raghavan, Réka Albert, and Soundar Kumara. Near linear time algorithm to detect community structures in large-scale networks. *Physical Review E*, 76(3):036106, 2007.

[RK14] Matteo Riondato and Evgenios M. Kornaropoulos. Fast approximation of betweenness centrality through sampling. In *Proceedings of the 7th ACM International Conference on Web Search and Data Mining*, WSDM '14, pages 413–422, New York, NY, USA, 2014. ACM.

[RMJ07] Matthew J Rattigan, Marc Maier, and David Jensen. Graph clustering with network structure indices. In *Proceedings of the 24th international conference on Machine learning*, pages 783–790. ACM, 2007.

[SAL+06] Vincent Savolainen, Marie-Charlotte Anstett, Christian Lexer, Ian Hutton, James J. Clarkson, Maria V. Norup, Martyn P. Powell, David Springate, Nicolas Salamin, and William J. Baker. Sympatric speciation in palms on an oceanic island. *Nature*, 441(7090):210–213, May 2006.

[ŠB11a] Lovro Šubelj and Marko Bajec. Robust network community detection using balanced propagation. *The European Physical Journal B*, 81(3):353–362, 2011.

[ŠB11b] Lovro Šubelj and Marko Bajec. Unfolding communities in large complex networks: Combining defensive and offensive label propagation for core extraction. *Physical Review E*, 83(3):036103, 2011.

[SC08] Philipp Schuetz and Amedeo Caflisch. Efficient modularity optimization by multistep greedy algorithm and vertex refinement. 2008.

[Sch07] Satu Elisa Schaeffer. Graph clustering. *Computer science review*, 1(1):27–64, 2007.

[Sha09] Devavrat Shah. *Gossip algorithms*. Now Publishers Inc, 2009.

[She14] Irina Shevtsova. On the absolute constants in the berry-esseen-type inequalities. *Doklady Mathematics*, 89(3):378–381, May 2014.

[SK88] Peter R Suaris and Gershon Kedem. An algorithm for quadrisection and its application to standard cell placement. *IEEE Transactions on Circuits and Systems*, 35(3):294–303, 1988.

[SLBK03] Adam Schenker, Mark Last, Horst Bunke, and Abraham Kandel. Clustering of web documents using a graph model. *SERIES IN MACHINE PERCEPTION AND ARTIFICIAL INTELLIGENCE*, 55:3–18, 2003.

[SM98] Jianbo Shi and Jitendra Malik. Motion segmentation and tracking using normalized cuts. In *Computer Vision, 1998. Sixth International Conference on*, pages 1154–1160. IEEE, 1998.

[SM00] Jianbo Shi and Jitendra Malik. Normalized cuts and image segmentation. *IEEE Transactions on pattern analysis and machine intelligence*, 22(8):888–905, 2000.

[SZ17] He Sun and Luca Zanetti. Distributed graph clustering and sparsification. *CoRR*, abs/1711.01262, 2017.

[TK06] Jianjun Paul Tian and D Kannan. Lumpability and commutativity of markov processes. *Stochastic analysis and Applications*, 24(3):685–702, 2006.

[TK08] Gergely Tibély and János Kertész. On the equivalence of the label propagation method of community detection and a potts model approach. *Physica A: Statistical Mechanics and its Applications*, 387(19):4982 – 4984, 2008.

[TL12] Stephen G. Turney and Jeff W. Lichtman. Reversing the Outcome of Synapse Elimination at Developing Neuromuscular Junctions In Vivo: Evidence for Synaptic Competition and Its Mechanism. *PLOS Biol*, 10(6):e1001352, June 2012.

[Tre14] Luca Trevisa. *Lecture Notes on Expansion, Sparsest Cut and Spectral Graph Theory*. 2014.

[TWK+12] Juan C. Tapia, John D. Wylie, Narayanan Kasthuri, Kenneth J. Hayworth, Richard Schalek, Daniel R. Berger, Cristina Guatimosim, H. Sebastian Seung, and Jeff W. Lichtman. Pervasive Synaptic Branch Removal in the Mammalian Neuromuscular System at Birth. *Neuron*, 74(5):816–829, June 2012.

[UB13] Johan Ugander and Lars Backstrom. Balanced label propagation for partitioning massive graphs. In *Proceedings of the sixth ACM international conference on Web search and data mining*, pages 507–516. ACM, 2013.

[V+13] Nisheeth K Vishnoi et al. Lx= b. *Foundations and Trends® in Theoretical Computer Science*, 8(1–2):1–141, 2013.

[VD01] Stijn Marinus Van Dongen. *Graph clustering by flow simulation*. PhD thesis, 2001.

[VL07] Ulrike Von Luxburg. A tutorial on spectral clustering. *Statistics and computing*, 17(4):395–416, 2007.

[WH04] Fang Wu and Bernardo A. Huberman. Finding communities in linear time: a physics approach. 38(2):331–338, 2004.

[WV18] Bryan Wilder and Yevgeniy Vorobeychik. Controlling elections through social influence. In *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems, AAMAS 2018, Stockholm, Sweden, July 10-15, 2018*, pages 265–273, 2018.

[XCS13] Jierui Xie, Mingming Chen, and Boleslaw K. Szymanski. Labelrankt: Incremental community detection in dynamic networks via label propagation. In *ACM Workshop on Dynamic Networks Management and Mining*, pages 25–32, 2013.

[XS11] Jierui Xie and Boleslaw K. Szymanski. Community detection using a neighborhood strength driven label propagation algorithm. In *Network Science Workshop (NSW), 2011 IEEE*, pages 188–195. IEEE, 2011.

[XS13] Jierui Xie and Boleslaw K. Szymanski. Labelrank: A stabilized label propagation algorithm for community detection in networks. In *Network Science Workshop (NSW), 2013 IEEE 2nd*, pages 138–143. IEEE, 2013.

[ZM04] Shi Zhou and Raúl J Mondragón. The rich-club phenomenon in the internet topology. *IEEE Communications Letters*, 8(3):180–182, 2004.

[ZRS+17] Xian-Kun Zhang, Jing Ren, Chen Song, Jia Jia, and Qian Zhang. Label propagation algorithm for community detection based on node importance and label influence. *Physics Letters A*, 381(33):2691–2698, 2017.