POLITECNICO
DI TORINO

UNIVERSITÀ
DEGLI STUDI
DI TORINO

Doctoral Dissertation
Doctoral Program in Pure and Applied Mathematics (30^{th}cycle)

Network Cooperative Models

By

Barbara Franci

******

Supervisor:
Prof. Fabio Fagnani

Doctoral Examination Committee:
Prof. E. Bini, Università di Torino
Prof. F. Pellerey, Politecnico di Torino
Prof. S. Zampieri, Università di Padova

Università di Torino - Politecnico di Torino
October, 9^{th}, 2018

Declaration

I hereby declare that, the contents and organization of this dissertation constitute my own original work and does not compromise in any way the rights of third parties, including those relating to the security of personal data.

Barbara Franci
October, 9^{th}, 2018

* This dissertation is presented in partial fulfillment of the requirements for **Ph.D. degree**.

Alla mia famiglia.
Tutta.

Acknowledgements

I would like to acknowledge that my PhD fellowship is sponsored by a TIM/Telecom Italia grant.

Abstract

In this thesis, we study two network cooperative models in the fields of evolutionary game theory and opinion dynamics. First, we consider an allocation problem motivated by peer-to-peer cloud storage models. The setting is that of a network of units (e.g. computers) that collaborate and offer each other space for the back up of the data of each unit. We formulate the problem as an optimization problem, we cast it into a game theoretic setting and we then propose a decentralized allocation algorithm based on the log-linear learning rule. Our main technical result is to prove the convergence of the algorithm to the optimal allocation. Moreover, we include some interest variants including a mechanism to avoid selfish behaviors and the possibility to allocate more copies of the date, enforcing the security of the storage. We give some bounds on the allocation time and present some simulations that show the feasibility of our solution and corroborate the theoretical results.

On the other hand, the second part of the thesis is devoted to the wisdom of a crowd. Naive learning, in particular, is a recent elegant model for the wisdom of crowds phenomenon. In this model, a large population loses its wisdom when a group of individuals has outsize influence in the consensus outcome of an opinion formation process. In this thesis we introduce and characterize a finite-time version of the naive learning model. Instead of requiring complete convergence to consensus, we study a finite-time opinion formation process and establish the notion of prominent families in this context. Surprisingly, finite-time wisdom is strictly distinct from its infinite-time version. We provide a comprehensive treatment of various finite-time wisdom notions, counterexamples to meaningful conjectures, and a complete characterization of equal-neighbor averaging models.

Contents

List of Figures

List of Tables

Nomenclature

Cloud Storage Problem

α_x	Amount of data that x needs to allocate
β_x	Amount of available storage of agent x
γ	Noise parameter of the Gibbs measure
λ_x	Reliability agent x
$\Lambda_{(W,\xi),(W',\xi)}$	Transition rates of the Markovian process
$\Lambda_{(W,\xi),(W,\xi')}$	Transition rates of the Markovian process
λ_{xy}	Estimated reliability of agent y by x
$\mu(\xi, W)$	Invariant probability measure
ν_x^{act}	Activation rate of agent x
ν_x^{off}	Rate of agent x for switching from on to off
ν_x^{on}	Rate of agent x for switching from off to on
ξ_x	Functional state of agent x
$\rho(W, W')$	Probability of acceptance under the states W and W'
$\rho_{yx}(t)$	Probability of acceptance at time t, i.e. probability that y accepts data from x
σ	Number of copies
τ_{abs}	Average absorption time

Ψ	Potential function
$\mathcal{A}$	Set of all the atoms of all the users to be allocated
$\mathcal{A}_x$	Set of all the atoms (x,a) of user x to be allocated
$\mathcal{B}$	Set of all the atoms of available storage of all the resources
$\mathcal{B}_y$	Set of all the atoms (y,b) of available space of resource y
C^{agg}	Aggregation constant
C^{all}	Allocation constant
C_y^{con}	Congestion constant
$\mathcal{E}$	Edges of the underlying graph
$\mathcal{E}_D$	Outgoing edges of a set D
$\mathcal{E}_W$	Outgoing edges of the graph $\mathcal{H}_W$
$\mathcal{E}_{\mathcal{P}}$	Edges of the bipartite graph
$\mathcal{G}$	Underlying graph
$\mathcal{H}_W$	Underlying graph of potential moves
$\mathcal{L}_p$	Underlying graph of the partial allocation set
$\mathcal{M}$	Matching
$\mathcal{M}_x(W,\xi)$	Admissible family of allocation states
n	Number of agents
$N(D)$	Set of out-neighbor of a set D in $\mathcal{G}$
$N^{\mathcal{P}}(A)$	Set of out-neighbor of a set A in $\mathcal{P}$
N_x	Set of out-neighbor of a node x
$N_x(W,\xi)$	Set of available neighbor of node x under W and ξ
N_y^-	Set of in-neighbor of a node y

$N_{(x,a)}(W,\xi)$	Set of available neighbor of node x under W and ξ
$\mathcal{P}$	Bipartite graph
$P_x^{(W,\xi)}(W')$	Probability of switching to state W' from W under ξ
$P_x^{(W,\xi)}(W)$	Probability of remaining in state W under ξ
$\mathcal{Q}$	Set of allocations
$\mathcal{Q}_p$	Set of partial allocations
$\tilde{Q}$	Injective allocation
Q	Allocation, a map $Q : \mathcal{A} \to \mathcal{X}$
$\mathcal{R}_{(x,a)}$	Set of resources used to allocate the copies of the atom (x,a)
$\mathcal{S}$	Set of all the copies of all the atoms of all the users to be allocated
$\mathcal{S}_x$	Set of all the copies (x,a,s) of the atoms of user x to be allocated
$U_x(W)$	Utility function of agent x
$\mathcal{W}$	Set of allocation states
$\mathcal{W}_p^{sat}$	Set of allocation states where no agents can make a move
$\mathcal{W}_p$	Set of partial allocation states
$\mathcal{W}_x(W,\xi)$	Possible partial allocation states obtainable from W
$\mathcal{W}_{(x,a)}(W,\xi)$	Possible partial allocation states obtainable from W
W	Partial allocation state
W^σ	Matching allocation state
W^x	Total amount of data allocated by x under W
W_y	Total amount of data stored in y under W
$W_{\cdot y}$	Column vector of W labelled by y
$W_{x\cdot}$	Row vector of W labelled by x

W_{xy}	Number of pieces of data that x has allocated in y under W
$\mathcal{X}$	Set of agents
$\mathcal{X}^f(W)$	Set of fully allocated units
$\mathcal{X}^{sat}(W)$	Set of saturated units
$Z_x^{(W,\xi)}(\gamma)$	Normalizing factor of the Gibbs measure
$Z_x^{(W,W',\xi)}(\gamma)$	Normalizing factor of the Gibbs measure

Wisdom of Crowds

Δ_n	Set of stochastic vectors of dimension n
μ	Unknown parameter
$\xi_i(0)$	Family of independent gaussian distributed noisy terms
$\pi^{[n]}$	Left dominant eigenvector of $P^{[n]}$
π	Left dominant eigenvector of P
τ_{mix}	Mixing time of a matrix P
Φ_P	Maximum son-time influence of a set of nodes
$\chi^{[n]}(k)$	Vector of the sums over the column of $(P^{[n]})^k$
$ave(x)$	Average of the vector $x \in \mathbb{R}^n$
$\mathbb{B}^{[n]}$	Prominent family
d_i	Out-degree of a node
d_{max}	Maximum degree of a graph
d_{min}	Minimum degree of a graph
$^{[n]}$	Superscript that indicates the dimension
$P^{[n]}$	Row-stochastic matrix defining the interaction graph of dimension n
P	Row-stochastic matrix defining the interaction graph

$x_i^{[n]}(k)$	Opinion of agent i at time k of dimension n
$x_i(0)$	Initial opinion of agent i
$x_i(k)$	Opinion of agent i at time k

Introduction

A multi-agent system is a system composed of multiple interacting intelligent agents within a certain environment. Topics in which the multi agents system approach can be applied include online trading, disaster response, modeling social structures, cooperation and coordination, distributed constraint optimization and multi-agent learning. The fields in which we study multi agents system are Game theory and network dynamics.
Game Theory is the field of mathematics that studies situations in which decision-makers interact with one another, and in which the behavior and the welfare of each of them depends also on the decisions made by the other participants. On the other hand, social network are primary conduits of opinion and informations, they carry news about products, influence decisions and drive political opinions toward other groups. In both these research topics it is important to understand how beliefs change through time, how these changes depend on the network structure and whether the final outcome is reasonable.
In this thesis we include two different contributions. In the first part, we study an application of evolutionary game theory to a network cloud storage model while, in the very final chapters, we focus on the research of conditions for a network to be wise. In this introduction chapter we discuss the motivations that lead us to these studies and the main theoretical results obtained in both the subjects. The analysis of the cloud storage model was previously presented in [9] while the work on the wisdom of crowds is presented in [6].

The decentralized cloud storage model. Cloud storage on the Internet has come to rely almost exclusively on data providers serving as trusted third parties to transfer and store the data. While the system works well enough in most cases, it still suffers from the inherent weaknesses of the trust-based model. The traditional cloud is open

to a variety of security threats, including man-in-the-middle attacks, and malware that expose sensitive and private consumer and corporate data. Furthermore, current cloud storage applications are charging large premiums on data storage facilities to business clients. Moreover, these cloud storage providers may have technical failures that can cause data breaches and unavailability, much to the distress of the users and applications that depend on them.
To address the aforementioned shortcomings, a decentralized peer-to-peer cloud storage model would be the right answer. On the wake of the successful peer-to-peer file sharing model of applications like BitTorrent and its lookalike, the same philosophy may well be leveraged on a different but very similar application service like storage. Indeed a slew of fledgling and somehow successful startups are entering in this market niche. Among the most noteworthy examples are:

- Storj: www.youtube.com/channel/UC-cTEqWwZV5Rl-h0RZsp2Qw [40],
- BitTorrent Sync: www.getsync.com/ [32],
- Ethos: www.youtube.com/watch?v=qUftGCQ5dqo,
- SpaceMonkeys: www.spacemonkey.com/,
- Sia: http://sia.tech [39].

Clearly, a completely decentralized peer-to-peer model must account for some challenging technical difficulties that are absent in a centralized cloud model. Firstly, security and privacy must be carefully implemented by ensuring end-to-end encryption resistant to attackers. In addition, the model must account for the latency, performance, and downtime of average user devices.
Albeit the above technical issues are challenging, they can be addressed with the right tools and architectures available at current state of the art technology and will not be considered in this thesis. What remains an open question up to now is how to endow the system with the right incentives for the end users to collaborate and share their storage commodity with each other.
We believe that the answer to that question comes from the formal framework of game-theory. First, it can be applied to the real world. Second, it provides the mathematical tools to study an ongoing phenomenon (this is the typical setting of social and psychological sciences). Third, it allows to design specific mechanisms,

i.e. set rules of the game to ensure successful cooperation among users/players (more engineering approach).

In order to solve the optimization problem in a scalable decentralized fashion we cast the allocation problem into a game theoretic framework and we make use of evolutionary game theory to design the algorithm.

In the last decade, game theory has emerged as a new fundamental paradigma to solve distributed optimization problems and, more in general, in the design and control of large scale networked engineering systems [15, 20, 25, 22]. The basic idea is that of modeling system units as rational agents whose behavior consists in a selfish search for their maximum utility. The goal is to design both the agents utility functions and an adaptation rule (evolutionary game dynamics) in such a way that the resulting global behavior is desirable. One of the advantages of this approach with respect to classical global optimization techniques is that it naturally leads to scalable decentralized solutions that can easily incorporate constraints on the computational power of the units as well as the information flow determined by the network architecture.

Two are the challenging issues when (evolutionary) game theory is used as a design mechanism: first, designing admissible agent utility functions such that the resulting game possesses desirable Nash equilibria; second, designing adaptation rules that guarantee the solution's convergence to such Nash equilibria.

Recently, cooperative storage cloud models based on peer-to-peer architectures have been proposed as valid alternatives to traditional centralized cloud storage services. The idea is quite simple: instead of using dedicated servers for the storage of data, the participants themselves offer space available on their connected devices to host data from other users. In this way, each participant has two distinct roles: that of a unit that needs external storage to securely back up its data, and that of a resource available for the back up of data of other users. This approach has in principle a number of relevant advantages with respect to traditional cloud storage models. First, it eliminates the need for a significant dedicated hardware investment so that the service should be available at (order of magnitude) lower cost. Second, it overcomes the typical problems related to the use of a single external provider as security threats or fragility with respect to technical failures.
In this thesis, we consider a network of units (PC's but possibly also smartphones or other devices possessing storage capabilities) which need to store externally a

back up of their data and, at the same time, can offer space available to store data of other connected units. In this set up, we cast the peer-to-peer storage model to an allocation Potential Game [27] and we propose an original decentralized algorithm which make units interact, cooperate, and store a complete back up of their data on their connected neighbors.
Units are assumed to be connected through a network and, autonomously, at random time, to activate and allocate or move their data pieces among the neighboring units. Formally, each unit has a utility function which gives a value to their neighbors on the basis of their reliability, their current congestion (resources have bounded storage capabilities), and the amount of data the unit has already stored in them.

Allocation games have been extensively studied [38, 18]. One of the key features of our model is the presence of hard constraints as a consequence of the bounded storage capabilities of the units. This property is non-classical in game theory and has remarkable consequences on the structure of Nash equilibria and the behavior of the algorithm. Even though the proposed game is shown to be a potential game, the convergence of the best response dynamics is here a subtle issue that does not follow from classical results. In fact, we propose an algorithm based on a noisy best response action: each time a unit activates, it decides the neighbor to use on the basis of a Gibbs probability distribution having its peak on the maxima of the utility function.
In this model, there is no need for central supervision and it can easily incorporate features that we want the system to possess depending on the application, as for instance, enforcing structure on the way data of each unit is treated (aggregate or rather disgregate in the back up process), avoiding congestion phenomena in the use of the resources, differentiate among resources on the basis of their reliability.

We want to remark that the type of functionals considered are typically non-convex (even when relaxed to continuous variables) so that many algorithms for distributed optimization may fail to converge to the global maximum. In addition, the proposed algorithm presents a number of interesting features. The algorithm is decentralized and adapted to any predefined graph. For the cloud application we have in mind, the choice of the graph topology can be seen as a design parameter that allows to control the computational complexity at the units level. It is asyncronous and it is robust with respect to temporary disconnection of units. Moreover, it is intrinsically open-ended: if new data or new units enters into the system, a new run of the algorithm will automatically permit the allocation of the new data and,

possibly, the redistribution of the data stored by the old units to take advantage of new available space.

For an important class of games known as potential games, a popular adaptation rule is the so-called noisy best-response where units choose their state by following a Gibbs-Boltzmann distribution having its peak on the maxima of the utility function [20]. For optimization purposes, an interesting strategy [22, 26] is to design utility functions so to yield a potential game whose potential coincide with the reward functional of the problem and then consider noisy best response dynamics (also known as log-linear learning dynamics) [25, 24]. This rule can be proven to yield convergence, asymptotically in time and in the limit of the noise approaching zero, to maxima of the potential that form a subclass of the Nash equilibria. In this context, the goal is to design utility functions in such a way that the potential (in particular, its maxima) have the desired global properties. The noisy best-response turns out to be a random decentralized scheme for the maximization of the potential. Following classical evolutionary game theory [35], we propose an algorithm based on a noisy best response action: each time a unit activates, it decides the neighbor to use on the basis of a Gibbs probability distribution having its peak on the maxima of the utility function.

Under certain assumptions, this rule is known to lead to a time-reversible ergodic Markov chain whose invariant probability distribution is a Gibbs measure with energy function described by the potential. For a very general family of functionals having an additive separable form, namely that can be expressed as sums of terms depending on the various units, we define a game by setting the utility function of each unit as simply the sum of those addends in the functional involving the unit itself and its neighbors, while the action set of a unit consists of the vectors describing the allocation among its various neighbors. The game so defined is easily shown to be potential with potential given by the original functional. The game, however, possesses a key critical feature: because of the hard storage constraints of the various resources, units are not free to choose their actions as they want, but they are constrained from the choice made by other units. For instance, if a unit is saturating the space available in a certain resource, other units connected to the same resource will not be able to use it. In cooperative cloud storage models where resources are common users, this hard storage constraint is a very natural assumption and can not be relaxed. This property is non-classical in game theory and has remarkable consequences on the structure of Nash equilibria and the behavior of

the noisy best response dynamics that is not guaranteed in general to approximate the optimum.

Since the first description of the algorithm include only the realistic assumption that resources can be in functional state off, we study some variations of the algorithm that make the approach more interesting and trustworthy.

First, we are interested in the behavior of the units so that a selfish user (who never accept the allocation from others) can be penalized or even excluded from the network. To this aim, the algorithm will include a reciprocity process in which users evaluate themselves and, as resources, have the possibility to deny the allocation from a unit. Indeed, we show that a selfish user who always deny allocation from others, remains excluded and cannot complete his/her backup. Clearly this phenomena happen in a reasonable amount of time; for longer amount of time the algorithm still guarantee the full allocation. The importance of reputation and trust is out of question in both human and virtual societies: reputation-based systems are used to establish trust among agents on a network in a wide set of applications [36, 30, 19]. In our approach the agents themselves are capable of punishing non-desirable behaviors, by for instance, not selecting certain resources. For this variation we describe the possible probabilities of acceptance, meaning the probability that a resource will accept the storage from a user; these probabilities represent different aspects of the reliability of the users and take into consideration also the previous interactions. Moreover, we prove the convergence of this algorithm with reciprocity to an allocation state with probability 1 following the analysis of the original case.

A second variation of the algorithm instead deals with the fact that units may want to allocate multiple copies of the data to enforce the security of the allocated data from external attacks. The multiple copies allocation leads to a more secure storage of the data and increase the probability of recovering them. The difficulties in this approach are related to the fact that a diversification of the storage is needed. Namely, since we want different copies of the same data to be stored in different resources, the algorithm has a constraint on the resource choice. This result in the addition of a constraint in the model that influence not only the allocation conditions and the characterization of the Nash equilibria but also the analysis of the algorithm.

Since we are studying an allocation problem, not only it is important to understand whether the allocation is possible but also if it is doable in a reasonable amount of time. For this reason, we will focus on the estimation of the allocation time. We

analyze the different algorithms in order to give the average allocation time (when possible) or at least some bounds. What makes this task hard is the fact that standard techniques [21] to compute the absorbing time of the underlying Markov Chain are not always applicable. This is why we resort to the Coupon Collector's problem [21] to find an upper bound on the allocation time.

Given our algorithm and its variation, the main theoretical results of this model are the following.

- We show the equivalence of the allocation problem with a classical matching problem on a graph. This allows us to use the celebrated Hall's theorem and give a necessary and sufficient condition for the allocation problem to be solvable. While this condition is true also in the algorithm with reciprocity, it does not hold in the multiple copies algorithm. For this last case, we state and motivate a conjecture with an equivalent condition.
- We characterize (when possible) the Nash equilibria of the game taking into account desirable features as the level of fragmentation of the data
- We prove that all units complete their allocation in finite time with probability one and that the allocation configuration converges, when the noise parameter approaches 0, to a maximum of the potential (that is also a Nash equilibria). This guarantees that the solution will indeed be close to the global welfare of the community.

For its definition, our problem fits into the wide class of distributed resource allocation problems. Among the many applications where such problems arise in a similar form to the one proposed in this thesis, we can cite cloud computing [18], network routing [34], vehicle target assignment [2], content distribution [13], graph coloring [29]. The game theoretic approach to allocation problems and the consequent design of distributed algorithms has been systematically addressed in [26, 22, 25, 24] where general techniques for the choice of the utility functions and of the dynamic learning rule have been proposed.

The model proposed in this thesis and the algorithm based on noisy best response dynamics, is inspired by this literature. A key aspect of our model that makes it different from the models treated in the above literature is the fact that resources have hard storage limitations. This is a natural feature of the distributed cloud storage

problem considered in this work and that, to our knowledge, had not been widely analyzed [28, 33].

Finite-time wisdom of crowds and naive learning. The last chapter of this thesis is devoted to the wisdom of crowds. This topic has recently become of particular interest. While most of the work related to social interaction are focused on the network properties of the underlying graph [8, 14], in this work we will focus on the outcome of the dynamic.

Social networks typically consist of a graph where each node possesses a state variable; the interconnection between the individuals depends on the edges in the underlying graph [8]. Imagine that a number of individuals possess an information represented by a real number (for instance an opinion on a given fact); agents interact and change their opinion by averaging with the opinions of other individuals. Under certain assumptions this will lead the community to converge to a consensus opinion. In social sciences, empiric evidence [12] has shown how such aggregate opinion may give a very good estimation of unknown quantities: such phenomenon has been proposed in the literature as *wisdom of crowds* [37].

Social scientists describe as *wisdom of crowds* the phenomenon whereby large groups of people are collectively smarter than single individuals. Much research has focused on understanding the conditions under which such a phenomenon occurs. The assumption on the required condition generally include diversity of opinions, independence of opinions, and a system that aggregates individual opinions without introducing bias. Indeed, one plausible explanation for this phenomenon in certain estimation tasks is that: each individual opinion is influenced by a independent zero-mean noise and, therefore, taking averages of large numbers of individual opinions will reduce the effect of noise simply due to the law of large numbers.

So it is natural to study the conditions under which an influence system enhances or diminishes the wisdom of crowd effect. Motivated precisely by such reasoning, an insightful *naive learning model* was recently proposed by Golub and Jackson [14]; in this model, a crowd is *wise* if, starting from individual estimates influenced by independent zero-mean noise, its asymptotic consensus value is equal to the unknown parameter.

Our work is a contribution to the naive learning model; instead of requiring that the opinion formation process is allowed infinite time in order for the crowd

to reach consensus, we here investigate the crowd's *finite-time wisdom*. For such finite-time settings, we aim to define a number of wisdom notions, provide rigorous characterizations, and work out instructive examples.

Aristotle is thought to be the first one who wrote about the 'Wisdom of the crowd' in his work titled 'Politics' while in recent years, Surowiecki's book [37] has widely popularized the concept. The most famous example is not quite recent and concerns the weight of an ox: the experiment took place at a 1906 country fair in Plymouth where 800 people participated in a contest to estimate the weight of a slaughtered and dressed ox. Statistician Francis Galton [12] observed that the median guess was indeed remarkably close to its correct value.

This has contributed to the insight in cognitive science that a crowd's individual judgments can be modeled as a probability distribution of responses with the median centered near the true value of the quantity to be estimated. On the other hand, it is possible to find in literature experiments in which the aggregate opinion is not close to the real value, depending on how the agents are susceptible to the social influence [23].

The literature on opinion dynamics and influence systems is very rich. The classic French-DeGroot averaging model is discussed in the text on influence systems by Friedkin and Johnsen [11], the text on social networks by Jackson [16], and the recent survey by Proskurnikov and Tempo [31]. Inextricably linked with influence systems is the concept of centrality in its various incarnations. Especially relevant to this work is the notion of the eigenvector centrality, e.g., see the foundational works by Bonacich [4] and Friedkin [10].

The phenomenon of wisdom of crowds is related also to the topic of social learning. We refer to the original work [14] for an insightful comparison between the naive learning approach and other distinct methods. We here only outline two alternative approaches. First, Acemoglu *et al* [1] present a game-theoretical model for sequential learning; even in this setup the presence of "excessively influential agents" is an impediment to social learning. Second, the social learning model proposed by Jadbabaie *et al* [17] is based on the constant arrival of new information (i.e., a feature our model does not have) and shows that new information together with basic connectivity properties is sufficient to overcome the influence of any individual agent.

The understanding of whether social influence processes enhance or diminishes the wisdom of a crowd continues to be of very recent interest and debate. Lorenz *et al* [23] describe an empirical study in which the influence system diminishes the accuracy of the collective estimate by diminishing diversity of the crowd (even while conducing the opinions closer to consensus). In contrast, Becker *et al* [3] present theoretical and empirical evidence about settings in which the influence system enhances the wisdom of the crowd.

In Chapter 7, we start by reviewing the naive learning model for large populations proposed by Golub and Jackson [14] and based on the French-DeGroot opinion formation process. In this model, the population is wise if the final (in the limit as time $k \to \infty$) consensus opinion is equal to the correct value. This property is cast in terms of the left dominant eigenvector of the influence matrix in large populations (i.e., in the limit as the number of individual $n \to \infty$).

In this thesis we consider a variation of the naive learning setting in which the French-DeGroot opinion formation process is allowed only in finite time and does not, therefore, reach completion. The social structure of the DeGroot model is described by a weighted and possibly directed network. Agents have beliefs about some common question of interest, communicate with their neighbors in the network and update their opinion. An agent's new belief is the weighted average of her neighbors' opinion from the previous instant of time. Over time, provided some connectivity and aperiodicity condition on the underlying graph [7, 5], beliefs converge to a consensus. Our individuals do not reach consensus and, in this thesis, a population is finite-time wise if the average of the individuals opinion remains correct as time progresses along the opinion dynamics process. In other words, [14] considers wisdom in the limit in which $n \to \infty$ after $k \to \infty$, we here consider the limit in which $n \to \infty$ after at $k = 1$, k fixed, and uniformly over k. We argue that these finite-time variation are especially relevant for large population, since it is known that the time required for consensus to be approximately achieved typically diverges as the population size diverges. It is therefore of interest to assume that no enough time is provided to the opinion formation process to achieve complete consensus and inquire what are wise sequences under this relaxed condition.

Given this premise, the last chapter of this thesis makes four main contributions. First, for our proposed finite-time setting we introduce the notions of one-time wisdom, finite-time wisdom, uniform wisdom, and pre-uniform wisdom. We provide

necessary and sufficient characterizations of one-time and finite-time wisdom in terms of the limiting value of the maximum column averages of the matrix sequence. We also provide a sufficient condition for uniform wisdom involving the same limiting value as well as the limiting value of the matrices' mixing time.

Second, we provide numerous detailed examples of graph families to establish, among other relationships, that finite-time wisdom neither implies nor is implied by wisdom, as previously defined. Our examples illustrate how various general implications among the various wisdom notions are tight. Our examples also demonstrate that the proposed notions of one-time and finite-time wisdom are meaningful and strictly distinct from the notion of (infinite-time) wisdom as defined by Golub and Jackson [14].

Third, we provide a sufficient condition to ensure that a sequence of row-stochastic matrices is indeed finite-time wise. We introduce an appropriate novel notion of prominent family and show how its absence implies finite-time wisdom in general. Roughly speaking, a collection of nodes (a family) is prominent if, in the limit of large population, its size is negligible (that is, of order $o(n)$) but its total accorded one-time influence is not (that is, of order 1 in n).

Fourth, we then extend our sufficient condition to the setting of equal-neighbor sequences of row-stochastic matrices. For such highly-structured case, we show that the absence of prominent individuals is a necessary and sufficient condition for finite-time wisdom, pre-uniform wisdom, and wisdom in the equal-neighbor setting. In other words, we completely characterize finite-time wisdom in the equal-neighbor setting.

Thesis organization. The thesis is structured as follows. Chapter 1 collect the mathematical notions necessary for understanding this work. In particular, it is focused on Game Theory and Graph Theory with their basic definitions. In Chapter 2, we describe the cloud storage model and give the details of the algorithm in its primary form. The reciprocity process is introduced in Chapter 3 where we give more details on the reputation based algorithms present in literature. The analysis of the allocation time is done in Chapter 4. This is the only chapter that include a set of simulations that verifies the obtained estimations. Chapter 5 is devoted to the formulation of the problem with multiple copies and its analysis. The last chapter of the cloud storage problem is Chapter 6: it is devoted to the presentation of an

extensive set of simulations that corroborate the theoretical results, prove the good scalability properties of the algorithm in terms of speed and complexity, and illustrate the influence of the parameters of the utility functions in the solution reached by the algorithm. The last two chapters deal with the wisdom of crowds; it includes the main results and a wide set of examples. A conclusion chapter ends the thesis.

References

[1] ACEMOGLU, D., DAHLEH, M. A., LOBEL, I., AND OZDAGLAR, A. Bayesian learning in social networks. *Review of Economic Studies 78*, 4 (2011), 1201–1236.

[2] ARSLAN, G., MARDEN, J. R., AND SHAMMA, J. S. Autonomous vehicle-target assignment: A game-theoretical formulation. *Journal of Dynamic Systems, Measurement, and Control 129*, 5 (2007), 584–596.

[3] BECKER, J., BRACKBILL, D., AND CENTOLA, D. Network dynamics of social influence in the wisdom of crowds. E5070–E5076.

[4] BONACICH, P. Factoring and weighting approaches to status scores and clique identification. *Journal of Mathematical Sociology 2*, 1 (1972), 113–120.

[5] BULLO, F. Lectures on network systems. *Version 0.86, November* (2016).

[6] BULLO, F., F.FAGNANI, AND FRANCI, B. Finite-time wisdom of crowds and naive learning. *Submitted to IEEE Transactions on Network Science and Engineering* (2018).

[7] DEGROOT, M. H. Reaching a consensus. *Journal of the American Statistical Association 69*, 345 (1974), 118–121.

[8] FAGNANI, F. Consensus dynamics over networks. Lecture notes for Winter School on Complex Networks, INRIA. Downloaded on 12/23/2016, Jan. 2014.

[9] FAGNANI, F., FRANCI, B., AND GRASSO, E. A game theoretic approach to a peer-to-peer cloud storage model.

[10] FRIEDKIN, N. E. Theoretical foundations for centrality measures. 1478–1504.

[11] FRIEDKIN, N. E., AND JOHNSEN, E. C. *Social Influence Network Theory: A Sociological Examination of Small Group Dynamics*. 2011.

[12] GALTON, F. Vox populi. 450–451.

[13] GOEMANS, M. X., LI, L., MIRROKNI, V. S., AND THOTTAN, M. Market sharing games applied to content distribution in ad hoc networks. *IEEE Journal on Selected areas in Communications 24*, 5 (2006), 1020–1033.

[14] GOLUB, B., AND JACKSON, M. O. Naive learning in social networks and the wisdom of crowds. *American Economic Journal: Microeconomics 2*, 1 (2010), 112–149.

[15] HAN, Z., NIYATO, D., SAAD, W., BASAR, T., AND HJORUNGNES, A. Game theory in wireless and communication networks: Theory. *Models and Applications* (2011).

[16] JACKSON, M. O. *Social and Economic Networks*. 2010.

[17] JADBABAIE, A., SANDRONI, A., AND TAHBAZ-SALEHI, A. Non-Bayesian social learning. *Games and Economic Behavior 76*, 1 (2012), 210–225.

[18] JALAPARTI, V., AND NGUYEN, G. D. Cloud resource allocation games. Tech. rep., 2010.

[19] KOUTROULI, E., AND TSALGATIDOU, A. Reputation-based trust systems for p2p applications: design issues and comparison framework. *Trust and privacy in digital business* (2006), 152–161.

[20] LASAULCE, S., AND TEMBINE, H. *Game theory and learning for wireless networks: fundamentals and applications*. Academic Press, 2011.

[21] LEVIN, D. A., PERES, Y., AND WILMER, E. L. *Markov chains and mixing times*. American Mathematical Soc., 2009.

[22] LI, N., AND MARDEN, J. R. Designing games for distributed optimization. *IEEE Journal of Selected Topics in Signal Processing 7*, 2 (2013), 230–242.

[23] LORENZ, J., RAUHUT, H., SCHWEITZER, F., AND HELBING, D. How social influence can undermine the wisdom of crowd effect. 9020–9025.

[24] MARDEN, J. R., AND SHAMMA, J. S. Revisiting log-linear learning: Asynchrony, completeness and payoff-based implementation. In *Communication, Control, and Computing (Allerton), 2010 48th Annual Allerton Conference on* (2010), IEEE, pp. 1171–1172.

[25] MARDEN, J. R., SHAMMA, J. S., ET AL. Game theory and distributed control. *Handbook of game theory 4* (2012), 861–900.

[26] MARDEN, J. R., AND WIERMAN, A. Distributed welfare games. *Operations Research 61*, 1 (2013), 155–168.

[27] MONDERER, D., AND SHAPLEY, L. S. Potential games. *Games and economic behavior 14*, 1 (1996), 124–143.

[28] PACCAGNAN, D., AND MARDEN, J. R. The importance of system-level information in multiagent systems design: Cardinality and covering problems. *arXiv preprint arXiv:1710.07460* (2017).

[29] PANAGOPOULOU, P. N., AND SPIRAKIS, P. G. A game theoretic approach for efficient graph coloring. In *ISAAC* (2008), vol. 8, Springer, pp. 183–195.

[30] PINYOL, I., AND SABATER-MIR, J. Computational trust and reputation models for open multi-agent systems: a review. *Artificial Intelligence Review 40*, 1 (2013), 1–25.

[31] PROSKURNIKOV, A. V., AND TEMPO, R. A tutorial on modeling and analysis of dynamic social networks. Part I. *Annual Reviews in Control 43* (2017), 65–79.

[32] QIU, D., AND SRIKANT, R. Modeling and performance analysis of bittorrent-like peer-to-peer networks. In *ACM SIGCOMM computer communication review* (2004), vol. 34, ACM, pp. 367–378.

[33] RAMASWAMY, V., PACCAGNAN, D., AND MARDEN, J. R. The impact of local information on the performance of multiagent systems. *arXiv preprint arXiv:1710.01409* (2017).

[34] ROUGHGARDEN, T. *Selfish routing and the price of anarchy*, vol. 174. MIT press Cambridge, 2005.

[35] SANDHOLM, W. H. *Population games and evolutionary dynamics*. MIT press, 2010.

[36] SELCUK, A. A., UZUN, E., AND PARIENTE, M. R. A reputation-based trust management system for p2p networks. In *Cluster Computing and the Grid, 2004. CCGrid 2004. IEEE International Symposium on* (2004), IEEE, pp. 251–258.

[37] SUROWIECKI, J. *The Wisdom Of Crowds*. Anchor, 2004.

[38] TEKIN, C., LIU, M., SOUTHWELL, R., HUANG, J., AND AHMAD, S. H. A. Atomic congestion games on graphs and their applications in networking. *IEEE/ACM Transactions on Networking (TON) 20*, 5 (2012), 1541–1552.

[39] VORICK, D., AND CHAMPINE, L. Sia: Simple decentralized storage.

[40] WILKINSON, S., BOSHEVSKI, T., BRANDOFF, J., AND BUTERIN, V. Storj a peer-to-peer cloud storage network.

Chapter 1

Preliminaries

In this chapter we give some basic definitions to introduce the reader to the terminology and notation of the thesis. We start with Game Theory and we end the chapter with some definitions from Graph Theory and Markov chains.

1.1 Game Theory

Game Theory is the field of mathematics that studies situations in which decision-makers interact with one another, and in which the happiness of each of them depends also on the decisions made by the other participants. One of the first subject in which Game Theory have found application is economic theory [18]. The approach of game theory in this field is fundamental because it allows the analysis of the behavior of the individuals of the economic community.

In general, a game is described by the following three features [3]:

- There is a set of participants, who are usually called the *players*. In this thesis we will also use the terms agents and users. The set of players is indicated with $\mathcal{X} = \{1, \ldots, n\}$ where $n \in \mathbb{N}$ is the number of agents. In this case, the game is called n-person game.
- Each player has a set of action from which he/she can choose; we will refer to these as the possible *strategies* of a player. In general, the actions sets are indicated with $A_1, \ldots, A_n$ being $1, \ldots, n$ the users; the sets of strategies are

supposed to be non-empty. The strategy of player i is usually indicate with a_i; we use a_{-i} to indicate the actions played by all the players except i.

- For each choice of strategies, each player receives a *payoff* that can depend on the strategies selected by everyone. The payoffs are generally numbers but depending on the case, it can be preferable to maximize or minimize it. In general, we denote with the symbol U_i the payoff of player $i \in \mathcal{X}$.

Since this aspects characterize the game, modeling the interaction framework as a strategic form game involves specifying the decision makers, their respective strategies, and a payoff function for each agent.
In general, we assume that everything a player cares about is summarized in the utility. However, nothing in the framework of game theory requires that players care only about their own personal rewards. Anyway, once the payoffs have been defined, they should constitute a complete description of the happiness of each player for all the possible outcomes of the game.
In some cases it seems reasonable to assume that each player also knows the structure of the network, the other players and their strategies or their payoffs. However, this knowledge is not essential. In this thesis for example, the players will have knowledge only of a part of the network and of their own strategies. Moreover, each player is assumed to be rational, meaning that he/she acts to maximize his/her payoff given what he/she knows. Clearly, Game theory has many elements in common with an optimization problem [18]. The difference is in the fact that in game theory to obtain an optimum result a player must relate with others. The problem is in fact a mixture of conflicting optimization problems (one for each player). Furthermore, we suppose that each player actually succeeds in selecting the optimal strategy. This seems reasonable in simple settings or for experienced players, while it is not in complex games, or games played by inexperienced players. The part of game theory that relates with players who make mistakes and learn during the game is know as Evolutionary Game Theory and we will talk about this later in this section.
Now we give some basic notions, starting from the definition of a one-shot game. A game is called *one-shot game* [5] if it is played once. In general, it can be represented by a matrix. We say that a player has a *strictly dominant strategy* if he/she has a strategy that is strictly better than all other options regardless of what the other player choose. When a player has a strictly dominant strategy, we should expect that they will play it (it follows from the fact that the players are rational).

In 1950, John Nash [13] proposed a simple but powerful principle for reasoning about behavior in general games. Normally, we should expect players to use strategies that are best responses to each other. More precisely, suppose that Player 1 chooses a strategy S and Player 2 chooses a strategy T. We say that this pair of strategies (S,T) is a Nash equilibrium if S is a best response to T, and T is a best response to S [3]. The idea is that no player has an incentive to deviate to a different strategy, so that the system is in an equilibrium state. If a pair of strategies are not best responses to each other, the players cannot both believe that these strategies will be actually played, as they know that at least one player would have an incentive to deviate from it. Therefore, a Nash equilibrium can be thought of as an equilibrium in beliefs. If each player believes that the other players will actually play a strategy that is part of a Nash equilibrium, then he/she is willing to play his/her part of the Nash equilibrium. Now we present an example, known as the Prisoner's Dilemma, to clarify the defined notions.

Example 1.1.1. *The Prisoner's Dilemma is presented as follows. Two members of a criminal gang are arrested and imprisoned. Each prisoner is in solitary confinement and they cannot comunicate with each other. The police lack sufficient evidence to convict the two suspects but they hope to get both sentenced to prison. Simultaneously, the policemen offer each prisoner a bargain. Each prisoner is given the opportunity to: betray the other by testifying that the other committed the crime, or confess. The offer is:*

- *If Player 1 and Player 2 each betray the other, each of them serves 2 years in prison;*
- *If Player 1 betrays Player 2 and Player 2 confess, Player 1 will be set free and Player 2 will serve 3 years in prison (and vice versa);*
- *If Player 1 and Player 2 both confess, both of them will only serve 1 year in prison.*

The payoff matrix can be written as follows; the utilities are taken negative since the more years in prison they get, the worse it is.

		Player 1	
		$Confess$	$Betray$
Player 2	$Confess$	$-1,-1$	$-3,0$
	$Betray$	$0,-3$	$-2,-2$

Since betraying offers a greater reward than cooperating, all rational self-interested prisoners would betray the other. So the only possible outcome for two prisoners is for them to betray each other. The interesting part of this result is that pursuing individual reward logically leads both of the prisoners to betray, when they would get a better reward if they both confess. Regardless of what the other decides, each prisoner gets a higher reward by betraying the other ("defecting"). In fact, if Player 2 cooperates, Player 1 should defect, because going free is better than serving 1 year. If Player 2 defects, Player 1 should also defect, because serving 2 years is better than serving 3. Therefore betraying is a dominant strategy. The same reasoning holds for Player 2.
Mutual defection is the only strong Nash equilibrium in the game. The dilemma then is that mutual cooperation yields a better outcome than mutual defection but it is not the rational outcome because the choice to cooperate, at the individual level, is irrational.

Now we propose some classes of games that will be useful in this thesis. These classes are population games, congestion games and potential games.

One can imagine many economic, social, and technological environments in which large collections of small agents make interdependent decisions. For example consider the traffic network where drivers commute from a place to another: the delay each driver experiences depends not only on the route he/she selects, but also on the other agents' route. Otherwise, consider the market where a large number of buyers and sellers participate in an exchange: each individual specifies the terms of trade and hopes of obtaining the greatest benefit.
While the two example above are quite different, they have some features in common. First, each environment contains a large number of agents capable of making independent decisions. Second, each agent is small, i.e., his/her choices have only a minor impact on the outcomes of other agents. Third, agents are anonymous: the outcome of an agent from the interaction depends on his own strategy and the distribution of the other strategies. Interactions with these three properties can be modeled using *population games* [16]. The population may have finite of infinite

agents. For our purpose, we assume that there are a finite number of players, who choose from a finite number of strategies. While some basic results, including existence of Nash equilibrium, can be proved at this level of generality, obtaining more specific conclusions requires certain structural properties. The subclasses of population games that we are going to study are the class of congestion games which provide a model of, for example, the traffic network and the class of potential games. Any game where a set of agents have to choose from a finite set of alternatives, and where the payoff of a player depends on the number of players choosing the same alternative, is a congestion game. Congestion games were first proposed by Rosenthal in 1973 [14]. Formally, a congestion game is a game where there is a set X of n players and a set of resources R. Each player has a finite set of strategies A_i that is a subset of R. Each resource has a cost (or delay) function which is to be interpreted as the payoff for using a certain resource.

Example 1.1.2. *Consider the traffic network where two players start at point O and need to get to point T. Suppose that node O is connected to node T via two roads A and B, where A is a little shorter than B. However, the more players pass through a point the greater the delay of each player becomes, so having both players go through the same connection point causes extra delay.*

Monderer and Shapley, in 1996, created the concept of a potential game and proved that every congestion game is a potential game [12].
In game theory, a game is said to be a potential game if the incentive of all players to change their strategy can be expressed using a single global function called the potential function. Formally, let n be the number of players, A the set of action profiles over the action sets A_i of each player $i \in X$ and U_i be the payoff function of user $i \in X$. A game is an *exact potential game* if there exists a function $\Phi : A \to \mathbb{R}$ such that $\forall a_{-i} \in A_{-i},\ \forall a_i',\ a_i'' \in A_i$,

$$\Phi(a_i', a_{-i}) - \Phi(a_i'', a_{-i}) = u_i(a_i', a_{-i}) - u_i(a_i'', a_{-i})$$

This means that when player i switches from strategy a' to strategy a'', the change in the potential equals the change in the utility of that player. This definition can be relaxed. An *ordinal potential game* is a game where there exists a function $\Phi : A \to \mathbb{R}$ such that $\forall a_{-i} \in A_{-i},\ \forall a_i',\ a_i'' \in A_i$,

$$\Phi(a_i', a_{-i}) - \Phi(a_i'', a_{-i}) > 0 \Leftrightarrow u_i(a_i', a_{-i}) - u_i(a_i'', a_{-i}) > 0.$$

There are other definitions of potential games [12] that we are not discussing in this thesis because they are not necessary for our propose.
The potential function is a useful tool to analyze equilibrium properties of games, since the incentives of all players are mapped into one function, and the set of pure Nash equilibria can be found by locating the local optima of the potential function. It is clear that since the potential games always have a Nash equilibrium, also the congestion games have one.

1.1.1 Evolutionary Game Theory

In the previous part of this section, we developed the basic ideas of game theory, in which individual players make decisions, and the payoff to each player depends on the decisions made by all.

In this subsection, on the other hand, game-theoretic analysis will be applied to settings in which individuals can exhibit different forms of behavior. As its name suggests, this approach has been applied most widely in the area of evolutionary biology [17] but the possible applications cross many discipline, from economics to social studies [5, 3, 16].
Talking about evolutionary game theory, we now describe two learning dynamics - best response and noisy best response - that we will use in the rest of the thesis.
Consider a one-shot game described with its payoffs matrix. If S is the strategy chosen by Player 1, and T is the strategy chosen by Player 2, then there is an entry in the payoff matrix corresponding to the pair (S,T). We will write $U_i(S,T)$ to denote the payoff to Player i, $i = 1,2$, as a result of this pair of strategies. Now, we say that a strategy S for Player 1 is a *best response* to a strategy T for Player 2 if S produces a payoff as good as any other strategy paired with T, that is

$$U_1(S,T) \geq U_1(S_0,T)$$

for all other strategies S_0 of Player 1. Naturally, there is an equivalent definition for Player 2 and strategy T. Notice that this definition allows for multiple different strategies of Player 1 to be tied as the best response to strategy T. This can make it difficult to predict which strategy Player 1 will use. For this reason, we introduce the notion of strict best response. A strategy S of Player 1 is a *strict best response* to a strategy T for Player 2 if S produces a strictly higher payoff than any other strategy

paired with T:

$$U_1(S,T) > U_1(S_0,T)$$

for all other strategies S_0 of Player 1. When a player has a strict best response to T, this is clearly the strategy she should play when faced with T.
In evolutionary game theory, best response dynamics represents a class of strategy updating rules, where players strategies in the next round are determined by their best responses to the strategies of the other agents. Players only choose the best response that would give them the highest payoff on the next round. Formally, we can describe the asynchronous best response dynamics in the following way [6]. Let $a(t) = (a_1(t), a_2(t), \ldots, a_n(t))$ be the action profile at time $t = 1, 2, \ldots$ At each time $t > 0$ player $i \in \mathcal{X}$ is chosen at random to change his/her current action. We suppose the the other player are not allowed to change their strategy, i.e., $a_i(t) = a_i(t-1)$. player i select an action $a_i(t)$ by choosing his/her best response to $a_{-i}(0)$; if there is more than one best response action, he/she choose one of them randomly. Then, another player is chosen to updates his/her action and so on.
Best response was first presented by Cournot in 1838 [2] and it is proven that it leads to a Nash equilibrium in finite potential games [12].
Given the definition of best response, we can redefine the dominant strategy. A dominant strategy for a player is a strategy that is a best response to every strategy of the other players. Analogously, a strictly dominant strategy for a player is a strategy that is a strict best response to every strategy of the other players.

An interesting and useful variation of the dynamics just described is the Noisy Best Response [9, 6, 10]. It is an iterative learning algorithm and it is known also as log-linear learning. Let $a(t)$ represent the action profile at time $t = 0, 1, 2, \ldots$. At each time $t > 0$, one player $i \in \mathcal{X}$ is randomly chosen and allowed to alter his/her current action. We suppose that the other players do not change their strategy, i.e. $a_{-i}(t) = a_{-i}(t-1)$. At time t, player i selects an action according the the following probabilistic strategy

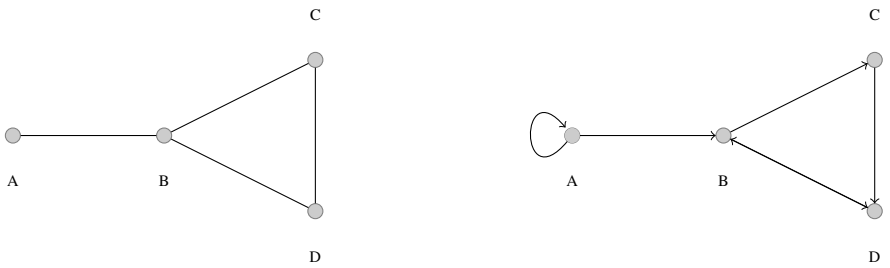
$$\mathbb{P}[a_i(t) = a_i] = \frac{e^{\frac{1}{\tau}U_i(a_i, a_{-i}(t-1))}}{\sum_{\bar{a}_i \in A_i} e^{\frac{1}{\tau}U_i(\bar{a}_i, a_{-i}(t-1))}}$$

for any action $a_i \in A_i$ and temperature $\tau > 0$. The temperature τ determines how likely player i is to select a suboptimal action. As $\tau \to \infty$, player i will select any

action $a_i \in A_i$ with equal probability. As $\tau \to 0$, player i will select a best response to the action profile $a_{-i}(t-1)$, with arbitrarily high probability.

1.2 Graph Theory

A graph is a way of specifying relationships among a collection of items. It consists of a set of objects with certain pairs of these objects connected by links. To make some examples consider that social networks (as Facebook, Twitter, ...) and information networks (as the Web) can be modeled by a graph. In the following part of the section we give some basic definitions, mostly taken from [1, 3].
An *undirected graph* consists of a set V of elements called vertices (or nodes) and of a set E of unordered pairs of vertices, called edges. For $u, v \in V$, $u \neq v$, the set $\{u, v\}$ denotes an unordered edge. In Figure 1.1a it is shown a very simple example with four nodes ($V = \{A, B, C, D\}$). To consider a bigger example, the Facebook network can be seen as an undirected graph where the users are the nodes and the edges represent the "friendship" relation.



(a) Undirected Graph (b) Directed Graph

Fig. 1.1 Examples of graphs

A *directed graph* of order n is a pair $G = (V, E)$, where V is a set with n vertices and E is a set of ordered pairs of vertices (the edges). For $u, v \in V$, the ordered pair (u, v) denotes an edge from u to v. Therefore, $E \subseteq V \times V$. A directed graph is undirected if $(v, u) \in E$ anytime $(u, v) \in E$. Moreover, in a directed graph, a *self-loop* is an edge from a node to itself. Figure 1.1b shows an example of a direct graph for the set of nodes $V = \{A, B, C, D\}$ which also include a self-loop in A. Another example of direct graph is the Web where the nodes are the pages and the edges are

the links from one page to another.
In a directed graph G with an edge $(u,v) \in E$, u is called an in-neighbor of v, and v is called an out-neighbor of u. We let $N^{in}(v)$ (respectively $N^{out}(v)$) denote the set of in-neighbors (out-neighbors) of v. The in-degree $d_{in}(v)$ and out-degree $d_{out}(v)$ of v are the number of in-neighbors and out-neighbors of v, respectively. Note that a self-loop at a node v makes v both an in-neighbor and an out-neighbor of itself.

A *weighted directed graph* is a graph $G = (V,E,W)$, where the pair (V,E) is a directed graph with nodes set V and edges set E, and where W is the weight matrix, i.e., w_{ij} is a positive weight for the edge $(i,j) \in E$. A directed graph $G = (V,E)$ can be regarded as a weighted directed graph by defining its set of weights to be all equal to 1, that is, setting $w_{ij} = 1$ for all $(i,j) \in E$. A weighted directed graph is undirected if $w_{ij} = w_{ji}$ for all $i, j \in V$. The notions of in- and out-degree can be generalized to the weighted graph as follows:

$$d_{out}(i) = \sum_{j \in V} w_{ij} \text{ and } d_{in}(i) = \sum_{j \in V} w_{ji}$$

In other words, d_{out} is the sum of the weights of al the out-edges and d_{in} is the sum of the weights of the in-edges.

Finally, a complete graph is an *undirected graph* where any pair of distinct nodes is connected by an edge; if it is a direct graph, any pair of nodes is connected by an edge in both directions. A graph is regular if all the nodes have the same degree. In Figure 1.2a there is a complete graph with six nodes while in Figure 1.2b there is a cycle which is a regular graph of degree 2; notice that a complete graph is a regular graph of degree $n-1$.

(a) Complete Graph (b) Cycle Graph

Fig. 1.2 Examples of graphs

A path is an ordered sequence of vertices such that any pair of consecutive vertices in the sequence is an edge of the graph. A path is simple if no vertex appears

more than once in it (except possibly for the initial and final vertex). A graph is *connected* if there exists a path between any two vertices. Similar definitions holds for directed graphs. A directed path is an ordered sequence of vertices such that any pair of consecutive vertices in the sequence is a directed edge and a directed path is simple if no vertex appears more than once in it (except possibly for the initial and final vertex). Moreover, we say that G is *strongly connected* if there exists a directed path from any node to any other node and that G is *weakly connected* if the undirected version of the directed graph is connected.

1.2.1 Matching Theory

In this section we define two notions, that is, bipartite graph and matching, useful to state the Hall's Marriage Theorem [4] which will be cited later in the thesis.

Given a graph $G = (V,E)$, a *matching* M in G is a set of pairwise non-adjacent edges, none of which are loops; that is, no two edges share a common vertex [8]. A *perfect matching* of a graph is a matching in which every vertex of the graph is incident to exactly one edge of the matching. A perfect matching is therefore a matching containing $n/2$ edges (the largest possible), meaning perfect matchings are only possible on graphs with an even number of vertices. A perfect matching is also called a complete matching. Matching problems are often concerned with bipartite graphs.

A *bipartite graph* is a graph whose vertices can be divided into two disjoint and independent sets A and B such that every edge connects a vertex in A to one in B. Vertex sets A and B are usually called the parts of the graph. Examples of bipartite graphs are every tree graph and any cycle graph with an even number of vertices.

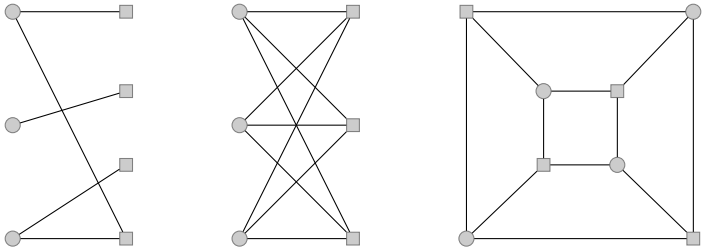


Fig. 1.3 Examples of bipartite graphs

Given a bipartite graph $G = (V,E)$ with parts A and B, for any subset $X \subseteq A$ we indicate with

$$N(X) = \{y \in B : (x,y) \in E \text{ for some } x \in A\}.$$

We are now ready to state the theorem.

Theorem 1.2.1 (Hall)**.** *Let $G = (V,E)$ be a finite bipartite graph with parts A and B and such that $|A| = |B|$. Suppose that for all subsets $X \subseteq A$ we have $|N(X)| \geq |X|$. Then G has a perfect matching.*

The theorem takes its name for one of its possible interpretation. Imagine a group of n men and a group of n women. For each woman, there is a subset of the men that she would marry and any man would be engaged with a woman who wants to marry him. The aim is to pair men and women so that every person is happy. Let A be the set of men and B the set of women and let A_x be the the set of men that a woman x would marry. The marriage theorem states that each woman can be engaged with a man if and only if the marriage condition holds. The marriage condition is that, for any subset X of the women, the number of men whom at least one of the women would marry, $|N(X)|$ is at least as big as the number of women in that subset, $|X|$. It is obvious that this condition is necessary, as if it does not hold, there are not enough men to share among the women but it is also a sufficient condition.

1.2.2 Markov chains

In 1907 Markov [11] began the study of a process where the outcome of an experiment can affect the outcome of the next experiment. This type of process was later called Markov chain.
Markov chains can be represented with graphs whose nodes set represent the state space and the edges represent the possible state transitions [15]. We suppose that whenever the process is in state i there is a fixed probability P_{ij} to jump in state j. Clearly, the matrix P can be seen as the weights matrix of the graph. Formally, a Markov chain is a stochastic process X(t), for $t = 0,1,\ldots$, with state space $\mathcal{X}$ coinciding with the node set V, and such that, for any states i and j in $\mathcal{X}$

$$\mathbb{P}[X(t+1) = j | X(0) = i_0, X(1) = i_1, \ldots, X(t-1) = i_t 1, X(t) = i] = P_{ij}. \quad (1.1)$$

Equation (1.1) states that the future state $X(t+1) = j$ is independent from the past, i.e., from the trajectory $X(0) = i_0, X(1) = i_1, \ldots, X(t-1) = i_{t-1}$, and that, given the present state $X(t) = i$, the probability of moving from node i to node j equals P_{ij}. The fact that the future state depends only on the present state is known as the *Markov property*, and the process $X(t)$ is usually called discrete-time Markov chain with transition probability matrix P.
Since the probabilities are nonnegative and the process must jump in some state, we have that P is a stochastic matrix, that is, a nonnegative square matrix with entries corresponding to the elements of X and all row sums equal to 1:

$$P_{ij} \geq 0 \text{ and } \sum_{j \in V} P_{ij} = 1.$$

Moreover a discrete-time Markoc chain can be associated to an initial probability distribution $\pi(0)$ whose entries correspond to the elements of $\mathcal{X}$ and such that

$$P(X(0) = i) = \pi_i(0), i \in X.$$

In order to determine the probability distribution of the trajectories of a Markov chain $X(t)$, it is necessary to specify both a transition probability matrix P and an initial probability distribution $\pi(0)$. Then, the trajectory satisfies

$$P(X(0) = i_0, X(1) = i_1, \ldots, X(t) = i_t) = \pi_{i_0}(0) \prod_{i \leq s \leq t} P_{i_{s-1} i_s}, t \geq 0. \tag{1.2}$$

From equation (1.2), for every time $t \geq 0$, recursive formulas for the marginal probability distribution $\pi(t)$ of $X(t)$ can be derived. The entry

$$\pi_i(t) := \mathbb{P}(X(t) = i), i \in X,$$

is the probability that the Markov chain is in node i at time t. The same holds for the t-step transition probability matrix $P(t)$, whose entries

$$P_{ij}(t) = \mathbb{P}(X(t) = j | X(0) = i), i, j \in X,$$

indicate the conditional probability that the chain is in node j at time t given that it was in node i at time 0. These recursive formulas can be written as

$$\pi(t+1) = P^T\pi(t) \text{ and } P(t+1) = P(t)P, P(t+1) = PP(t)$$

which are known as the Chapman-Kolmogorov equations. From these, it follows that

$$P(t) = P^t \text{ and } \pi(t) = \pi(0)P^t, t \geq 0.$$

Moreover, we call a probability π satisfying

$$\pi = \pi P \tag{1.3}$$

a *stationary distribution* of the Markov chain. Clearly, if π is a stationary distribution and $\pi(0) = \pi$, then $\pi(t) = \pi$ for all $t \geq 0$. In this case, π is the long-term limiting distribution of the chain. This is why we sometimes say that π is an invariant probability distribution for P.
We say that a probability π on X satisfies the detailed balance equations if

$$\pi_i P_{ij} = \pi_j P_{ji} \text{ for all } i, j \in X. \tag{1.4}$$

Let P be the transition matrix of a Markov chain with state space X. It can be proven [7] that any distribution π satisfying the detailed balance equations is stationary for P. Therefore, checking detailed balance equations is often the simplest way to verify that a particular distribution is stationary.
We now present an example, taken from [7], to show a very simple Markov chain.

Example 1.2.2. *A frog lives in a pond with two lily pads, east and west. One day, it finds two coins at the bottom of the pond and decide to bring one up to each lily pad. From that day, every morning, the frog decides whether to jump by tossing the coin of the current lili pad. If the coin lands heads up, the frog jumps to the other lily pad. Otherwise, he remains where it is. Let* $\{e,w\}$ *be the states set, and let* $(X_0, X_1, \dots)$ *be the sequence of lily pads occupied by the frog. Since the coins were found at the bottom of the pond, we should not suppose that they are fair. Therefore, the coin on the east pad has probability p of landing heads up, while the coin on the west pad has probability q of landing heads up. The rules of the frog for jumping imply that if*

we set

$$P = \begin{pmatrix} P_{ee} & P_{ew} \\ P_{we} & P_{ww} \end{pmatrix} = \begin{pmatrix} 1-p & p \\ q & 1-q \end{pmatrix},$$

then $(X_0, X_1, \dots)$ *is a Markov chain with transition matrix P. Doing some computations and since the probability distribution must be stationary, we have*

$$\pi_e = \frac{q}{p+q}, \pi_w = \frac{p}{p+q}.$$

When a probability distribution π satisfy the detailed balance equations, it hold that

$$\pi_{i_0} P_{i_0 i_1} \cdots P_{i_{n-1} i_n} = \pi_{i_n} P_{i_n i_{n-1}} \cdots P_{i_1 i_0}.$$

In other words, if a Markov chain satisfies (1.4) and has stationary initial distribution, then the distribution of $(X_0, X_1, \dots, X_n)$ is the same as the distribution of $(X_n, X_{n-1}, \dots, X_0)$. For this reason, a chain satisfying (1.4) is called *reversible*.

The time reversal of an irreducible Markov chain with transition matrix P and stationary distribution π is the chain with matrix

$$\bar{P}_{ij} := \frac{\pi_j P_{ji}}{\pi_i}.$$

The stationary equation $\pi = \pi P$ implies that $\bar{P}$ is a stochastic matrix.

Moreover, we say that a Markov chain is *irreducible* [7] if for any two states $i, j \in \mathcal{X}$ there exists an integer t (possibly depending on i and j) such that $P^t_{ij} > 0$. This means that it is possible to get from any state to any other state using only transitions of positive probability.

Let f_i be the probability that, starting in state i, the process will reenter in state i. State i is said to be *recurrent* if $f_i = 1$ and *transient* if $f_i < 1$. If state i is recurrent, the Markov property implies that the process will visit i infinitely many times. On the other hand, if state i is transient, there is a positive probability $1 - f_i$ of never entering again that state. Consequently, state i is recurrent if and only if the expected number of visits to i is infinite while it is transient if the expected number of visits to i is finite.

For an irreducible chain, the period of the chain is defined to be the period which is common to all states. In other words [15], a state j is said to be periodic if there is an integer $d > 1$ such that $P^t_{jj} = 0$ whenever d does not divide t. The minimal such

d is called period (if this hold for all $t > 0$ then he period of j is infinite). State j is aperiodic if there is no such a $d > 1$. The chain will be called *aperiodic* if all states have period 1. If a chain is not aperiodic, we call it *periodic*.

A state i is said to be *ergodic* if it is aperiodic and positive recurrent (recurrent and with finite mean recurrence time). If all states in an irreducible Markov chain are ergodic, then the chain is said to be ergodic. It can be shown that a finite state irreducible Markov chain is ergodic if it has an aperiodic state. More generally, a Markov chain is ergodic if there is a number N such that any state can be reached from any other state in a number of steps greater than or equal to N.
One of the most famous example of Markov chain is the birth and death chain that we describe, following [7], in the next example.

Example 1.2.3. *A birth-and-death chain has state space $X = \{0,1,2,\ldots,n\}$. In one step the state can increase or decrease by at most 1. The current state can be thought of as the size of a population. In a single step of the chain there can be at most one birth or death. The transition probabilities can be described by:*

- *p_k, the probability of moving from k to $k+1$ when $0 \leq k < n$,*
- *q_k, the probability of moving from k to $k-1$ when $0 < k \leq n$,*
- *r_k, the probability of remaining at k when $0 \leq k \leq n$,*

Moreover, $q_0 = p_n = 0$ and $p_k + q_k + r_k = 1$ for all k. It is well known that every birth-and-death chain is reversible.

References

[1] BULLO, F. Lectures on network systems. *Version 0.86, November* (2016).

[2] COURNOT, A.-A. *Recherches sur les principes mathématiques de la théorie des richesses par Augustin Cournot*. chez L. Hachette, 1838.

[3] EASLEY, D., AND KLEINBERG, J. *Networks, crowds, and markets: Reasoning about a highly connected world*. Cambridge University Press, 2010.

[4] HALL, P. On representatives of subsets. *Journal of the London Mathematical Society 1*, 1 (1935), 26–30.

[5] HAMIDOU, T., ALTMAN, E., EL-AZOUZI, R., AND HAYEL, Y. Evolutionary games in wireless networks. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics* (2009).

[6] LASAULCE, S., AND TEMBINE, H. *Game theory and learning for wireless networks: fundamentals and applications*. Academic Press, 2011.

[7] LEVIN, D. A., PERES, Y., AND WILMER, E. L. *Markov chains and mixing times*. American Mathematical Soc., 2009.

[8] LOVÁSZ, L., AND PLUMMER, M. D. *Matching theory*, vol. 367. American Mathematical Soc., 2009.

[9] MARDEN, J. R., AND SHAMMA, J. S. Revisiting log-linear learning: Asynchrony, completeness and payoff-based implementation. In *Communication, Control, and Computing (Allerton), 2010 48th Annual Allerton Conference on* (2010), IEEE, pp. 1171–1172.

[10] MARDEN, J. R., SHAMMA, J. S., ET AL. Game theory and distributed control. *Handbook of game theory 4* (2012), 861–900.

[11] MARKOV, A. A. Rasprostranenie zakona bolshih chisel na velichiny, zavisyaschie drug ot druga. *Izvestiya Fiziko-matematicheskogo obschestva pri Kazanskom universitete 15*, 135-156 (1906), 18.

[12] MONDERER, D., AND SHAPLEY, L. S. Potential games. *Games and economic behavior 14*, 1 (1996), 124–143.

[13] NASH, J. F., ET AL. Equilibrium points in n-person games. *Proceedings of the national academy of sciences 36*, 1 (1950), 48–49.

[14] ROSENTHAL, R. W. A class of games possessing pure-strategy nash equilibria. *International Journal of Game Theory 2*, 1 (1973), 65–67.

[15] ROSS, S. M. Stochastic processes. 1996, 1996.

[16] SANDHOLM, W. H. *Population games and evolutionary dynamics*. MIT press, 2010.

[17] SMITH, J. M., AND PRICE, G. R. The logic of animal conflict. *Nature 246*, 5427 (1973), 15–18.

[18] VON NEUMANN, J., AND MORGENSTERN, O. Theory of games and economic behavior.

Chapter 2

A network cloud storage model

In this chapter we present a cooperative fully distributed algorithm through which a network of units (e.g. computers) can collaborate and offer each other space for the back up of the data of each unit. In this model, there is no need for central supervision and it can easily incorporate features that we want the system to possess depending on the application: enforcing structure on the way data of each unit is treated (aggregate or rather disgregate), avoiding congestion phenomena, differentiate among resources on the basis of their reliability. The algorithm was presented in a different version without analytical results in [4].

For a very general family of functionals having an additive separable form, namely that can be expressed as sums of terms depending on the various units, we define a game by setting the utility function of each unit as simply the sum of those addends in the functional involving the unit itself and its neighbors, while the action set of a unit consists of the vectors describing the allocation among its various neighbors. The game so defined is easily shown to be potential with potential given by the original functional. The game, however, possesses a key critical feature: because of the hard storage constraints of the various resources, units are not free to choose their actions as they want, but they are constrained from the choice made by other units.

The first result in this chapter is the proof of the existence of an allocation with the analysis of some particular cases. The main technical contribution is to show that, despite the hard constraints, under mild technical conditions, a family of dynamics having their core on the log-linear learning rule converge to the desired

solution. More precisely, by a careful analysis of the connectivity properties of the transition graph associated to the Markov process, we will obtain two results, stated in Theorem 2.3.2 and Corollary 2.3.8. Theorem 2.3.2 ensures that the algorithm reaches a complete allocation with probability one, if a complete allocation is indeed possible. Corollary 2.3.8 studies the asymptotic behavior of the algorithm and explicitly exhibits the invariant probability distribution. Consequence of Corollary 2.3.8 is that in the double limit when time goes to infinity and the noise parameter goes to 0, the algorithm converges to a Nash equilibrium that is, in particular, a global maximum of the potential function. This guarantees that the solution will indeed be close to the global welfare of the community. At the best of our knowledge, this analysis is new in game theory.

The remaining part of this chapter is structured as follow. In Section 2.1 we formally define the network allocation problem and prove a necessary and sufficient condition for the allocation problem to be solvable (the proof first appear in [4]). We then introduce a family of functionals and define the optimal allocation problem. Section 2.2 is devoted to cast the problem to a potential game theoretic framework [8, 7] and to propose a distributed algorithm that is an instance of a noisy best response dynamics. The main technical part of the chapter is Section 2.3 where the fundamental results Theorem 2.3.2 and Corollary 2.3.8 are stated and proven. We do not include simulations in this chapter, but we present some examples. Simulations will be presented in Chapter 6.

2.1 The cooperative storage model

Consider a set $\mathcal{X}$ of units that play the double role of users who have to allocate externally a back up of their data, as well resources where data from other units can be allocated. Generically, an element of $\mathcal{X}$ will be called a unit, while the terms user and resource will be used when the unit is considered in the two possible roles of, respectively, a source or a recipient of data. We assume units to be connected through a directed graph $\mathcal{G} = (\mathcal{X}, \mathcal{E})$ where a link $(x,y) \in \mathcal{E}$ means that unit x is allowed to store data in unit y. We denote by

$$N_x := \{y \in \mathcal{X} \,|\, (x,y) \in \mathcal{E}\}, \quad N_y^- := \{x \in \mathcal{X} \,|\, (x,y) \in \mathcal{E}\}$$

respectively, the out- and the in-neighborhood of a node. Note the important different interpretation in our context: N_x represents the set of resources available to unit x while N_y^- is the set of units having access to resource y. If $D \subseteq \mathcal{X}$, we put $N(D) = \cup_{x\in D} N_x$ and $N^-(D) = \cup_{x\in D} N_x^-$.

We assume the data possessed by the units to be quantized atoms of the same size. Each unit x is characterized by two non negative integers:

- α_x is the number of data atoms that unit x needs to back up into his neighbors,
- β_x is the number of data atoms that unit x can accept and store from his neighbors.

The numbers $\{\alpha_x\}$ and $\{\beta_x\}$ will be assembled into two vectors denoted, respectively, α and β. Given the triple $(\mathcal{G}, \alpha, \beta)$, we define a *partial state allocation* as any matrix $W \in \mathbb{N}^{\mathcal{X}\times\mathcal{X}}$ that satisfies the following conditions

(P1) $W_{xy} \geq 0$ for all x, y and $W_{xy} = 0$ if $(x,y) \notin \mathcal{E}$.

(P2) $W^x := \sum_{y\in\mathcal{X}} W_{xy} \leq \alpha_x$ for all $x \in \mathcal{X}$.

(P3) $W_y := \sum_{x\in\mathcal{X}} W_{xy} \leq \beta_y$ for all $y \in \mathcal{X}$.

We interpret W_{xy} as the number of pieces of data that x has allocated in y under W. Property (P1) enforces the graph constraint: x can allocate in y iff $(x,y) \in \mathcal{E}$. Property (P2) says that a unit can not allocate more data than the one it owns, and, finally, (P3) describes the storage constraint at the level of units considered as resources. Whenever W satisfies (P2) with equality for all $x \in \mathcal{X}$, we say that W is an *allocation state*. The set of partial allocation states and the set of allocation states are denoted, respectively, with the symbols $\mathcal{W}_p$ and $\mathcal{W}$. We will say that the allocation problem is solvable if a state allocation $W \in \mathcal{W}$ exists.

2.1.1 The allocation problem as a matching problem

We define

$$\mathcal{A}_x = \{(x,a) \,|\, a \in \{1,\ldots,\alpha_x\}\}, \quad \mathcal{A} = \bigcup_{x\in\mathcal{X}} \mathcal{A}_x$$

$$\mathcal{B}_y = \{(y,b)\,|\,b \in \{1,\ldots,\beta_y\}\}, \quad \mathcal{B} = \bigcup_{y\in\mathcal{X}} \mathcal{B}_y$$

$\mathcal{A}$ and $\mathcal{B}$ represent, respectively, the sets of all the atoms to be allocated and the set of atoms of available space.

Given the triple $(\mathcal{G},\alpha,\beta)$, we define an allocation as any map $Q : \mathcal{A} \to \mathcal{X}$ ($Q(x,a) = y$ means that x has allocated the data atom a into resource y) satisfying the properties expressed below.

(C1) **Graph constraint** $Q(x,a) \in N_x$ for all $x \in \mathcal{X}$ and $a \in \{1,\ldots,\alpha_x\}$;

(C2) **Storage limitation** For every $y \in \mathcal{X}$, it holds, $|Q^{-1}(y)| \leq \beta_y$.

We will say that the allocation problem is solvable if an allocation Q exists. We denote by $\mathcal{Q}$ the set of allocations. We will also need to consider partial allocations, namely functions $Q : D \to \mathcal{X}$ where $D \subseteq \mathcal{A}$ and satisfying, where defined, conditions (C1) and (C2). We denote by $\mathcal{Q}_p$ the set of partial allocations.

It is clear that, given a partial allocation Q, the matrix $W(Q) \in \mathbb{N}^{\mathcal{X}\times\mathcal{X}}$ where $W(Q)_{xy}$ is the number of atomic data that x has allocated in y under Q, namely,

$$W(Q)_{xy} := |Q^{-1}(y) \cap \mathcal{A}_x| \tag{2.1}$$

is an allocation state. It is immediate to see that, conversely, if there exists W satisfying the properties (P1)-(P3), then, from it, we can construct an allocation Q such that $W = W(Q)$. Clearly, under this correspondence, we have that $Q \in \mathcal{Q}$ iff W satisfies (P2) with equality for all $x \in \mathcal{X}$. It is clear that two allocations Q^1 and Q^2 such that $W(Q^1) = W(Q^2)$, only differ for a permutation of the data atoms of the various units and for many porpouses can be considered as equivalent.

Consider now the bipartite graph $\mathcal{P} = (\mathcal{A}\times\mathcal{B}, \mathcal{E}_\mathcal{P})$ where $((x,a),(y,b)) \in \mathcal{E}_\mathcal{P}$ iff $(x,y) \in \mathcal{E}$. An allocation naturally induces a matching on $\mathcal{P}$ which is complete on $\mathcal{A}$. To this aim, notice that, from $Q \in \mathcal{Q}$ and using condition (C2), we can construct an injective mapping $\tilde{Q} : \mathcal{A} \to \mathcal{B}$ such that $\tilde{Q}(x,a) = (Q(x,a),b)$ for every $x \in \mathcal{X}$ and for all a.

We then define

$$\mathcal{M} := \bigcup_{x\in\mathcal{X}} \{((x,a),(y,b)) \in \mathcal{A}\times\mathcal{B}\,|\,(y,b) = \tilde{Q}(x,a)\}$$

It is clear that this procedure can be inverted and that from any matching of $\mathcal{M}$ complete on $\mathcal{A}$ we can associate an allocation for $(\mathcal{G}, \alpha, \beta)$. This equivalence allows to use classical results like the Hall's marriage theorem to characterize the existence of allocations. Precisely we have the following result

Theorem 2.1.1. *Given $(\mathcal{G}, \alpha, \beta)$, there exists an allocation iff the following condition is satisfied:*

$$\sum_{x \in D} \alpha_x \leq \sum_{y \in N(D)} \beta_y \quad \forall D \subseteq \mathcal{X} \tag{2.2}$$

Proof. By Hall's theorem, the existence of a matching in $\mathcal{P}$ complete on A is equivalent to the condition

$$|A| \leq |N^{\mathcal{P}}(A)| \quad \forall A \subseteq \mathcal{A} \tag{2.3}$$

where $N^{\mathcal{P}}(A) \subseteq \mathcal{B}$ is the out-neighborhood of A in $\mathcal{P}$. Given $A \subseteq \mathcal{A}$ let $\bar{A}$ be the union of those $\mathcal{A}_x$'s for which $\mathcal{A}_x \cap A \neq \emptyset$. By the way the bipartite graph $\mathcal{P}$ has been defined and by the fact that any atom $(x,a) \in \mathcal{A}$ is connected in $\mathcal{P}$ to all the atoms $(y,b) \in \mathcal{B}$ such that $y \in N_x$, it follows that $N^{\mathcal{P}}(A) = N^{\mathcal{P}}(\bar{A})$. Therefore it is sufficient to restrict condition (2.3) to subsets A such that $\mathcal{A}_x \cap A \neq \emptyset$ yield $\mathcal{A}_x \subseteq A$. Given such an A, if we consider $D = \{x \mid \mathcal{A}_x \subseteq A\}$, we immediately obtain that (2.2) coincides with (2.3). □

From the practical point of view, the equivalence of our problem with a classical matching problem, is, however, of little utility, as the number of nodes of $\mathcal{P}$ is of the size $\sum \alpha_x + \sum \beta_y$ which will in general be very large.

In general, it is not necessary to check the validity of (2.2) for every subset D. We say that $D \subseteq \mathcal{X}$ is maximal if for any $D' \supsetneq D$, it holds $N(D') \supsetneq N(D)$. We say that $D_1, D_2 \subseteq \mathcal{X}$ are independent if $N(D_1) \cap N(D_2) = \emptyset$ and $D \subseteq \mathcal{X}$ is called irreducible if it can not be decomposed into the union of two non empty independent subsets. Clearly, it is sufficient to verify (2.2) for the subclass of maximal irreducible subsets.

Example 2.1.2. *If $\mathcal{G}$ is complete, we have that $N(\{x\}) = \mathcal{X} \setminus \{x\}$ while $N(D) = \mathcal{X}$ for all D such that $|D| \geq 2$. Hence, the only maximal irreducible subsets are the singletons $\{x\}$ and the set $\mathcal{X}$. Condition (2.2) in this case reduces to*

$$\alpha_x \leq \sum_{y \neq x} \beta_y, \ \forall x \in \mathcal{X} \qquad \sum_{x \in \mathcal{X}} \alpha_x \leq \sum_{y \in \mathcal{X}} \beta_y \tag{2.4}$$

In general, the class of maximal irreducible subsets can be large and grow more than linearly in the size of $\mathcal{X}$, as the following example shows.

Example 2.1.3. *If $\mathcal{G} = (\mathcal{X}, \mathcal{E})$ is a line graph ($\mathcal{X} = \{1, 2, \ldots, n\}$ and $\mathcal{E} = \{(i, i+1), i = 1, \ldots, n-1\}$) it can be checked that the maximal irreducible subsets are those of the form $\{i, i+2, \ldots, i+2s\}$.*

We now focus on the special but intersting case when all units have the same amount of data to be stored and the same space available, namely, $\alpha_x = a$, $\beta_x = b$ for every $x \in \mathcal{X}$. In this case, condition (2.4) that characterizes the existence of allocations for the complete graph, simply reduces to $a \leq b$. In this case, among the possible allocation states there are those where each unit uses only one resource: given any permutation $\sigma : \mathcal{X} \to \mathcal{X}$ without fixed points, we can consider

$$W^{\sigma}_{xy} = \begin{cases} a & \text{if } \sigma(x) = y \\ 0 & \text{otherwise} \end{cases} \tag{2.5}$$

In general, an allocation state as W^{σ} in (2.5) of the example above where each unit uses just one resource and each resource is only used by one unit, is called a *matching allocation state*. Existence of matching allocation states is guaranteed for more general graphs than the complete ones.

Proposition 2.1.4. *Let $\mathcal{G} = (\mathcal{X}, \mathcal{E})$ be any graph and assume that $\alpha_x = a$, $\beta_x = b$ for every $x \in \mathcal{X}$. The following conditions are equivalent:*

(i) There exists an allocation state W;

(ii) There exists a matching allocation state;

(iii) $a \leq b$ and $|D| \leq |N(D)|$ for every subset $D \subseteq \mathcal{X}$.

Proof. (ii) $\Rightarrow$ (i) is trivial. Notice that (iii) is, in this case, equivalent to condition (2.2). Therefore (i) $\Rightarrow$ (iii) follows from Theorem 2.1.1. What remains to be shown is that (iii) $\Rightarrow$ (ii). To this aim, notice that when (iii) is verified and we consider the bipartite graph $\tilde{\mathcal{G}} = (\mathcal{X} \times \mathcal{X}, \tilde{\mathcal{E}})$ where $(x, y) \in \tilde{\mathcal{E}}$ iff $(x, y) \in E$, Hall's theorem guarantees the existence of a matching in $\tilde{\mathcal{G}}$ complete on the first set, namely a permutation $\sigma : \mathcal{X} \to \mathcal{X}$ such that $(x, \sigma(x)) \in \tilde{\mathcal{E}}$ for every $x \in \mathcal{X}$. The corresponding state allocation W^{σ} defined as in (2.5) is a matching allocation state. □

We can now extend the result contained in Example 2.1.2.

Corollary 2.1.5. *Suppose that $\mathcal{G} = (\mathcal{X}, \mathcal{E})$ is any undirected regular graph and that $\alpha_x = a$, $\beta_x = b$ for every $x \in \mathcal{X}$ with $a \leq b$. Then, there exists a (matching) allocation state.*

Proof. Let s be the degree of each node in the graph. Fix any subset $D \subseteq \mathcal{X}$. If $\mathcal{E}_D$ is the set of directed edges starting from a node in D, we have that $s|D| = |\mathcal{E}_D| \geq s|N(D)|$. This implies that that $|D| \leq |N(D)|$. We conclude using Proposition 2.1.4. □

Not necessarily a matching allocation state is the desirable one. In certain applications, security issues may rather require to fragment the data of each unit as much as possible. Suppose we are under the same assumptions than in previous result, namely $\mathcal{G} = (\mathcal{X}, \mathcal{E})$ is an undirected regular graph with degree s, $\alpha_x = a$, $\beta_x = b$ for every $x \in \mathcal{X}$ with $a \leq b$. If moreover s divides a we can also consider the 'diffused' allocation state given by

$$W_{xy} = \frac{a}{s} A_{xy} \tag{2.6}$$

where A is the adjacency matrix of $\mathcal{G}$. Notice that all these matrices W can also be interpreted as valid allocation states for the case when the underlying graph is complete.

For graphs that are not regular, simple characterizations of the existence of allocations in general do not exist. However, sufficient conditions can be obtained as the result below shows and whose proof follows along the same line than the proof of Corollary 2.1.5.

Proposition 2.1.6. *Let $\mathcal{G}$ be any graph with minimal out-degree d_{min} and maximum in-degree d^-_{max}. Let $a = \max_x \alpha_x$, $b = \min_x \beta_x$ and assume that $a \leq b d_{min}/d^-_{max}$. Then, there exists an allocation state.*

The above result can not be improved: indeed in a star graph with $\alpha_x = a$, $\beta_x = b$ for every $x \in \mathcal{X}$, it is immediate to see that the condition $a \leq b d_{min}/d^-_{max}$ is necessary for an allocation to exist.

2.1.2 The optimal allocation problem

On the set of allocation states, we define a reward functional measuring qualitative and realistic features that we desire the solution to possess, i.e., congestion and aggragation. Functionals considered in this paper have a separable structure that is a standard assumption in allocation problems [6].

We start with a notation. Given a (partial) allocation state $W \in \mathcal{W}_p$, we denote with the symbols $(W_{x\cdot})$ and $(W_{\cdot y})$, respectively, the row vector of W with label x, and the column vector of W with label y. We consider functionals $\Psi : \mathcal{W}_p \to \mathbb{R}$ of the type:

$$\Psi(W) = \sum_{x \in \mathcal{X}} f_x(W_{x\cdot}) + \sum_{y \in \mathcal{X}} g_y(W_{\cdot y}) \tag{2.7}$$

consisting of two parts: one that takes into account the way each unit is succeeding in allocating its data and another that is typical a congestion term and considers the amount of data present in the various resources. Our goal is to maximize the functional Ψ over the set of allocation states $\mathcal{W}$. The reason for defining Ψ in the larger set of partial allocation states $\mathcal{W}_p$ will be clearer later when we present the game theoretic set up and the algorithm.

Examples and simulations in this thesis will focus on the following cases:

$$\begin{aligned} f_x(W_{x\cdot}) &= C^{all} \textstyle\sum_y W_{xy} + C^{agg} \sum_{y \in \mathcal{X}} W_{xy}^2, \\ g_y(W_y) &= -C_y^{con} (W_y)^2 \end{aligned} \tag{2.8}$$

We now explain the meaning of the various terms:

- the term $C^{all} \sum_x \sum_y W_{xy}$ where $C^{all} > 0$ is sufficiently large, has the effect of pushing the optimum to be an allocation state (a configuration where all units have stored their entire set of data);
- the term $C^{agg} \sum_x \sum_{y \in \mathcal{X}} W_{xy}^2$ has different significance depending on the sign of C^{agg}. If $C^{agg} > 0$ plays the role of an aggregation term, it pushes units not to use many different resources for their allocation. If instead $C^{agg} < 0$, the term has the opposite effect as it pushes towards fragmentation of the data.
- the term $-\sum_y C_y^{con} (W_y)^2$ is a classical congestion term: the constants $-C_y^{con} < 0$ for all y measure the reliability of the various resources and pushes the use of more reliable resources

An alternative choice for the resource congestion term is the following. Put $|W_{\cdot y}|_H = |\{x \in \mathcal{X}, W_{xy} > 0\}|$ the number of units that are using resource y and consider

$$g_y(W_{\cdot y}) = -C_y^{con}|W_{\cdot y}|_H \tag{2.9}$$

This might be useful in contexts where it is necessary the control the number of units accessing the same resource, to avoid communication burden.

The functionals (2.8) and (2.9) reflects the features that we wanted to enforce: congestion and aggregation. The reason for the latter feature comes from the fact that an exceeding fragmentation of the stored data will cause a blow up in the number of communications among the units, both in the storage and recovery phases. This feature should be considered against another feature, the diversification of back ups, which is going to be addressed in Chapter 5. In real applications, units will need to store multiple copies of their data in order to cope with security and failure phenomena. In that case, these multiple copies will need to be stored in different units. On the other hand, the congestion term is represented by a classical cost function that each user possibly experiences, for instance, as a delay in the storage/recovery actions.
The above desired features may be contradictory in general and we want to have tunable parameters to make the algorithm converge towards a desired compromised solution. The choice of this functionals has been made on the basis of simple realistic considerations and on the fact that, as exploited below, this leads to a potential game. In principle, different terms in the utility function can be introduced in order to make units to take into considerations other desired features (e.g multiple back up).

While our theory and algorithms will be formulated for a generic Ψ as defined in (2.7), the example proposed and the numerical simulations will be restricted to the specific cases we have described.

Below we present a couple of examples of explicit computation of the maxima of Ψ. We assume Ψ to be of the form described in (2.7) and (2.8) with $C_y^{con} = C^{con}$ for every $y \in \mathcal{X}$. We also assume that $\alpha_x = a$, $\beta_x = b$ for every $x \in \mathcal{X}$ with $a \leq b$.

Example 2.1.7. *Suppose that $\mathcal{G} = (\mathcal{X}, \mathcal{E})$ is any undirected regular graph and assume that $\alpha_x = a$, $\beta_x = b$ for every $x \in \mathcal{X}$ with $a \leq b$. Take Ψ to be of the form described in (2.7) and (2.8) with $C_y^{con} = C^{con} > 0$ for every $y \in \mathcal{X}$. There are two cases:*

- *$C^{agg} > 0$. In this case, the maxima of Ψ coincide with the matching allocation states. Indeed notice that any matching allocation state W (whose existence is guaranteed by Proposition 2.1.4) separately maximizes, for each y, the two expressions $\sum_{y\in\mathcal{X}} W_{xy}$ and $\sum_{y\in\mathcal{X}} W_{xy}^2$. Moreover, considering that $W_y = a$ for every resource y, simultaneously, minimize the congestion expression $\sum_{y\in\mathcal{X}}(W_y)^2$. The fact that these are the only possible maxima is evident from these considerations.*

- *$C^{agg} < 0$. If the degree s of $\mathcal{G}$ divides a, arguing like above, we see that the unique maximum is given by the diffused allocation state (2.6). When s does not divide a, such a simple solution does not exist. In this case, maxima can be characterized as follows. Put $a = sk + r = (s-r)k + r(k+1)$ (with $r < s$) and consider a regular subgraph $\tilde{G}$ of degree r. An optimal allocation is obtained by letting units allocate $k+1$ atoms of their data in each of their neighbors in $\tilde{G}$ and k atoms of their data in each of the remaining neighbors.*

2.2 The game theoretic set-up and the algorithm

In this section we recast the optimization problem into a game theoretic context and we then use learning dynamics to derive decentralized algorithms adapted to the given graph topology that solve the allocation problem and maximize the functional Ψ.

Assume that a functional Ψ as in (2.7) has been fixed. We associate a game to Ψ according to the ideas developed in [6, 2].

The set of actions $\mathcal{A}_x$ of a unit x is given by all possible row vectors $(W_{x\cdot})$ such that $\sum_x W_{xy} \leq \alpha_x$. In this way the product set of actions $\prod_x \mathcal{A}_x$ can be made to coincide with the space of non-negative matrices $W \in \mathbb{R}^{\mathcal{X}\times\mathcal{X}}$ such that $\sum_x W_{xy} \leq \alpha_x$ for every $x \in \mathcal{X}$. Such a W in general is not a partial allocation. Indeed, such a W will automatically only possess properties (P1) and (P2). We have that $W \in \mathcal{W}_p$ if the extra conditions (P3), $\sum_y W_{xy} \leq \beta_y$ for every $y \in \mathcal{X}$, is satisfied. This is a key non classical feature of the game associated to our model: the storage limitations make the available actions of a unit depend on the choice made be the other ones.

Now, for each unit x, we define its utility function $U_x : \mathcal{W}_p \to \mathbb{R}$ as

$$U_x(W) = f_x(W_{x\cdot}) + \sum_{y \in N_x^-} g_y(W_{\cdot y})\,. \tag{2.10}$$

Note that, in order to compute $U_x(W)$, unit x needs to know, besides the state of its own data allocation $\{W_{x\cdot}\}$, the congestion state $g_y(W_{\cdot y})$ of the neighboring resources. Different utility functions, with different levels of informations required, can be found in [2].

We now recall some basic facts of game theory. A *Nash equilibrium* is any allocation state $W \in \mathcal{W}_p$ such that, for every $\bar{x} \in \mathcal{X}$ and for every $W' \in \mathcal{W}_p$ such that $W_{xy} = W'_{xy}$ for every $x \neq \bar{x}$ and for every y, it holds

$$U_{\bar{x}}(W) \geq U_{\bar{x}}(W') \tag{2.11}$$

If $W, W' \in \mathcal{W}_p$ are two allocation states such that $W_{xy} = W'_{xy}$ for every $x \neq \bar{x}$ and for every y, it is straightforward to see that the following equality holds

$$\begin{aligned}
U_{\bar{x}}(W') - U_{\bar{x}}(W) &= f_{\bar{x}}(W'_{\bar{x}\cdot}) + \sum_{y \in N_{\bar{x}}^-} g_y(W'_{\cdot y}) - \left(f_{\bar{x}}(W_{\bar{x}\cdot}) + \sum_{y \in N_{\bar{x}}^-} g_y(W_{\cdot y}) \right) \\
&= f_{\bar{x}}(W'_{\bar{x}\cdot}) - f_{\bar{x}}(W_{\bar{x}\cdot}) + \sum_{y \in N_{\bar{x}}^-} g_y(W'_{\cdot y}) - g_y(W_{\cdot y}) \\
&= \sum_{x \in \mathcal{X}} f_x(W'_{x\cdot}) - f_x(W_{x\cdot}) + \sum_{y \in \mathcal{X}} g_y(W'_{\cdot y}) - g_y(W_{\cdot y}) \\
&= \sum_{x \in \mathcal{X}} f_x(W'_{x\cdot}) + \sum_{y \in \mathcal{X}} g_y(W'_{\cdot y}) - \left(\sum_{x \in \mathcal{X}} f_x(W_{x\cdot}) + \sum_{y \in \mathcal{X}} g_y(W_{\cdot y}) \right) \\
&= \Psi(W') - \Psi(W)
\end{aligned} \tag{2.12}$$

This says, in the language of game theory [7], that the game is *potential* with *potential function* given by Ψ itself. A simple classical result says that maxima of the potential are Nash equilibria for the game. In general the game will possess extra Nash equilibria.

The choice (2.10) is not the only one to lead to a potential game with potential Ψ. Other possibilities can be constructed following [6].

As far as our theory is concerned, the specific form of the utility functions is not important as far as it leads to a potential game with potential Ψ. On the utility functions, (2.10) or its possible alternatives, we impose a monotonicity condition that essentially says that no unit will ever have a vantage to remove data already allocated. Precisely, we assume that for every $W, W' \in \mathcal{W}_p$, $\bar{x} \in \mathcal{X}$ such that $W_{xy} = W'_{xy}$ for every $x \neq \bar{x}$ and for every y, the following holds

$$W'^{\bar{x}} < W^{\bar{x}} \Rightarrow U_{\bar{x}}(W') < U_{\bar{x}}(W) \tag{2.13}$$

This condition is not strictly necessary for our results (as our algorithm actually will not allow units to remove data), it is however a meaningful assumption and simulations show that helps to speed up the algorithm.

We now focus on the case when Ψ is of the form given by (2.8) with $C^{con}_y = C^{con}$ for every $y \in \mathcal{X}$. In this case, a simple check shows that the monotonicity condition (2.13) is guaranteed if we impose the condition

$$C^{all} > 2(||\alpha||_\infty |C^{agg}| + ||\beta||_\infty C^{con}) \tag{2.14}$$

where $||v||_\infty = \max v_i$ is the infinity norm of a vector.

We conclude this section, computing the Nash equilibria in a couple of simple examples and discussing the relation with the maxima of Ψ.

Example 2.2.1. *Suppose that $\mathcal{G}$ is the complete graph with three units and that $\alpha_x = a = 2$ and $\beta_x = b \geq 2$ for $x = 1,2,3$. Consider Ψ to be of the form (2.8) with $C^{con}_y = C^{con}$ for every $y \in \mathcal{X}$ and that condition (2.14) holds. Consider the following allocation states*

$$W^1 = \begin{pmatrix} 0 & 2 & 0 \\ 0 & 0 & 2 \\ 2 & 0 & 0 \end{pmatrix}, W^2 = \begin{pmatrix} 0 & 0 & 2 \\ 2 & 0 & 0 \\ 0 & 2 & 0 \end{pmatrix}, W^3 = \begin{pmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{pmatrix}$$

We know from the considerations in Example 2.1.7 that in the case when $C^{agg} > 0$, the matching allocation states W^1 and W^2 are the (only) two maxima of Ψ and thus Nash equilibria. Instead, if $C^{agg} < 0$, the diffused allocation state W^3 is the only maximum of Ψ and is in this case a Nash equilibrium.

Notice now that if $b < 3$, the only three possible allocation states are W^i for $i = 1,2,3$. Since any two of these matrices differ in more than one row and condition (2.14) yields (2.13), we deduce that all three of them are in this case Nash equilibria, independently on the sign of C^{agg}.

Suppose now that $b \geq 3$. Explicit simple computations show that, if $C^{agg} \leq C^{con}$, W^3 is a Nash equilibrium and if $C^{agg} \geq -6C^{con}$, W^1 and W^2 are Nash equilibria. In summary, if $b < 3$ or if $b \geq 3$ and $-6C^{con} \leq C^{agg} \leq C^{con}$, the three matrices W^i for $i = 1,2,3$ are Nash equilibria.

The next example shows that also partial allocations may be Nash equilibria.

Example 2.2.2. *$\mathcal{G}$ 5-cycle, $\alpha_x = a = 4$ and $\beta_1 = 7$, $\beta_2 = 2$, $\beta_3 = 4$, and $\beta_4 = \beta_5 = 6$. It can be checked that the two matrices below are both Nash equilibria:*

$$W = \begin{pmatrix} 0 & 0 & 0 & 0 & 4 \\ 3 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 4 & 0 \\ 0 & 0 & 3 & 0 & 1 \\ 4 & 0 & 0 & 0 & 0 \end{pmatrix}, \qquad W = \begin{pmatrix} 0 & 0 & 0 & 0 & 4 \\ 3 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 4 & 0 \\ 0 & 0 & 4 & 0 & 0 \\ 4 & 0 & 0 & 0 & 0 \end{pmatrix}$$

The one of the right is a maximum of Ψ, the one on the left is instead a partial allocation.

2.2.1 The algorithm

The allocation algorithm we are proposing is fully distributed and asynchronous and is only based on communications between units, taking place along the links of the graph $\mathcal{G} = (\mathcal{X}, \mathcal{E})$. It is based on the ideas of learning dynamics where, randomly, units activate and modify their action (allocation state) in order to increase their utility. The most popular of these dynamics is the so-called *best response* where units at every step choose the action maximizing their utility. This dynamics is proven to converge almost surely, in finite time, to a Nash equilibrium. In presence of Nash equilibria that are not maxima of the potential (as it is in our case) best response dynamics is not guaranteed to converge to a maximum. This is simply because Nash Equilibria are always equilibrium points for the dynamics.

A popular variation of the best response is the so-called *noisy* best response (also known as *log-linear learning*) where maximization of utility is relaxed to a random choice dictated by a Gibbs probability distribution [3].

We now illustrate the details of our algorithm. For the sake of proposing a realistic model we assume that units may temporarily be shut down or in any case disconnected from the network. We model this assuming that, at every instant of time, a unit is either in functional state on or off: units in functional state off are not available for communication and for any action including storage and data retrieval. A unit, which is currently in state on, can activate and either newly allocate or move some data among the available resources (e.g. those neighbors that still have place available and that are on at that time).

The functional state of the network at a certain time will be denoted by $\xi \in \{0,1\}^{\mathcal{X}}$: $\xi_x = 1$ means that the unit x is on. The times when units modify their functional state (off to on or on to off) and the times when units in functional state on activate are modeled as a family of independent Poisson clocks whose rates will be denoted (for unit x), respectively, ν_x^{on}, ν_x^{off}, and ν_x^{act}. The functional state of the network as a function of time $\xi(t)$ is thus a continuous time Markov process whose components are independent Bernoulli processes.

We now describe the core of the algorithm, namely the rules under which activated units can modify their allocation state.

We start with some notation. Given a (possibly partial) allocation state $W \in \mathcal{W}_p$, a functional state $\xi \in \{0,1\}^{\mathcal{X}}$, and a unit $\bar{x} \in \mathcal{X}$ such that $\xi_{\bar{x}} = 1$, define:

$$\mathcal{W}_{\bar{x}}(W,\xi) = \left\{ W' \in \mathcal{W}_p : \begin{array}{l} W'_{xy} = W_{xy} \text{ if } x \neq \bar{x} \text{ or } \xi_y = 0 \\ W'^{\bar{x}} \geq W^{\bar{x}},\ W' \neq W \end{array} \right\}. \tag{2.15}$$

$\mathcal{W}_{\bar{x}}(W,\xi)$ describes the possible partial allocation states obtainable from W by modifications done by the unit $\bar{x}$: only the terms $W_{\bar{x}y}$ where y is on can be modified and the total amount of allocated data $W^{\bar{x}}$ can only increase or remain equal. Since the sets $\mathcal{W}_{\bar{x}}(W,\xi)$ can in general be very large, it is convenient to consider the possibility that the algorithm might use a smaller set of actions where units either allocate new data or simply move data from one resource to another one.

Given $(W,\xi) \in \mathcal{W}_p \times \{0,1\}^{\mathcal{X}}$ and a unit $\bar{x}$, define

$$N_{\bar{x}}(W,\xi) := \{y \in N_{\bar{x}} \,|\, W_{\bar{x}y} < \beta_y,\, \xi_y = 1\} \tag{2.16}$$

the set of available neighbor resources for $\bar{x}$ under the allocation state W and the functional state ξ: those that are on and still have space available.

A family of sets $\mathcal{M}_{\bar{x}}(W,\xi) \subseteq \mathcal{W}_{\bar{x}}(W,\xi)$, defined for each $\bar{x} \in \mathcal{X}$ and each $(W,\xi) \in \mathcal{W}_p \times \{0,1\}^{\mathcal{X}}$, is called *admissible* if

(i) $W^{\bar{x}} < \alpha_{\bar{x}},\, y \in N_{\bar{x}}(W,\xi), \Rightarrow \exists n : W' = W + ne_{\bar{x}y} \in \mathcal{M}_{\bar{x}}(W,\xi)$;

(ii) $W_{\bar{x}y'} > 0,\, \xi_{y'} = 1,\, y'' \in N_{\bar{x}}(W,\xi) \Rightarrow W' = W + (e_{\bar{x}y''} - e_{\bar{x}y'}) \in \mathcal{M}_{\bar{x}}(W,\xi)$;

(iii) $W' \in \mathcal{M}_{\bar{x}}(W,\xi)$ iff $W \in \mathcal{M}_{\bar{x}}(W',\xi)$ for every $W \in \mathcal{W}$.

Conditions (i) and (ii) essentially asserts that when a unit has an available neighbor resource not yet saturated, then $\mathcal{M}_{\bar{x}}(W,\xi)$ must incorporate the possibility to newly allocate or transfer already allocated data into it. Condition (iii) instead simply says that when the functional state does not change and we are in an allocation state, any transformation can be reversed.

Examples of admissible families $\mathcal{M}_{\bar{x}}(W,\xi)$ are the following

1. $\mathcal{M}_{\bar{x}}(W,\xi) = \mathcal{W}_{\bar{x}}(W,\xi)$
2. $\mathcal{M}_{\bar{x}}(W,\xi) = \{W' \in \mathcal{W}_{\bar{x}}(W,\xi) : \exists y, \exists n\, W' = W + ne_{\bar{x}y})\} \cup \{W' \in \mathcal{W}_{\bar{x}}(W,\xi) : \exists y', y'', \exists n\, W' = W + n(e_{\bar{x}y''} - e_{\bar{x}y'})\}$
3. $\mathcal{M}_{\bar{x}}(W,\xi) = \{W' \in \mathcal{W}_{\bar{x}}(W,\xi) : \exists y, \exists n \in Q\, W' = W + ne_{\bar{x}y})\} \cup \{W' \in \mathcal{W}_{\bar{x}}(W,\xi) : \exists y', y'', \exists n \in Q\, W' = W + n(e_{\bar{x}y''} - e_{\bar{x}y'})\}$ where $Q \subseteq N$ and $1 \in Q$.

In the second case, modifications allowed are those where a unit either allocate a certain amount of new data into a single resource or it moves data from one resource to another one. The third case puts an extra constraint on the amount of data allocated or moved: the simplest case is $Q = \{1\}$, just an atomic piece of data is newly allocated or moved. Simulation presented in this thesis all fit in this third case with various possible sets Q.

Given an admissible family $\mathcal{M}_{\bar{x}}(W,\xi)$, we now define a *Gibbs measure* on it as follows. Given a parameter $\gamma > 0$, put

$$\begin{aligned} Z_{\bar{x}}^{(W,\xi)}(\gamma) &= \sum_{\tilde{W}\in\mathcal{M}_{\bar{x}}(W,\xi)} e^{\gamma U_{\bar{x}}(\tilde{W})} \\ Z_{\bar{x}}^{(W,W',\xi)}(\gamma) &= \max\left\{Z_{\bar{x}}^{(W,\xi)}(\gamma), Z_{\bar{x}}^{(W',\xi)}(\gamma)\right\} \end{aligned} \tag{2.17}$$

Now define, for $W' \in \mathcal{M}_{\bar{x}}(W,\xi)$,

$$P_{\bar{x}}^{(W,\xi)}(W') = \begin{cases} \frac{e^{\gamma U_{\bar{x}}(W')}}{Z_{\bar{x}}^{(W,\xi)}(\gamma)}, & \text{if } ||W|| < ||W'|| \\ \frac{e^{\gamma U_{\bar{x}}(W')}}{Z_{\bar{x}}^{(W,W',\xi)}(\gamma)}, & \text{if } ||W|| = ||W'|| \end{cases} \tag{2.18}$$

where $||W|| = \sum_{xy} W_{xy}$, and complete it to a probability by putting

$$P_{\bar{x}}^{(W,\xi)}(W) = 1 - \sum_{W'\in\mathcal{M}_{\bar{x}}(W,\xi)} P_{\bar{x}}^{(W,\xi)}(W')$$

The algorithm is completely determined by the choice of the admissible family $\mathcal{M}_{\bar{x}}(W,\xi)$ and of the probabilities (2.18). If unit $\bar{x}$ activates at time t, the systems is in partial allocation state $W(t)$, and in functional state $\xi(t)$, it will jump to the new partial allocation state W' with probabilities given by

$$P(W(t+) = W') = P_{\bar{x}}^{(W(t),\xi(t))}(W'),\ W' \in \mathcal{M}_{\bar{x}}(W(t),\xi(t)) \tag{2.19}$$

If unit $\bar{x}$ chooses a W' such that $||W'|| > ||W||$ we say that it makes an *allocation move*, otherwise, if $||W'|| = ||W||$, we talk of a *distribution move*.

2.3 Analysis of the algorithm

In this section we analyze the behavior of the algorithm introduced above. We will essentially show two results:

1. first, we prove that if the set of state allocation $\mathcal{W}$ is not empty (i.e. condition (2.2) is satisfied), the algorithm above will reach such an allocation in bounded time with probability 1 (e.g $W(t) \in \mathcal{W}$ for t sufficiently large);

2. second, we show that, under a slightly stronger assumption than (2.2), in the double limit $t \to +\infty$ and then $\gamma \to +\infty$, the process $W(t)$ induced by the algorithm will always converge, in law, to a Nash equilibrium that is a global maximum of the potential function Ψ.

In order to prove such results, it will be necessary to go through a number of intermediate technical steps. Other possible techniques can be found in [1, 9, 5].

In the sequel we assume we have fixed a triple $(\mathcal{G}, \alpha, \beta)$ satisfying the existence condition (2.2), an admissible family of sets $\mathcal{M}_{\bar{x}}(W, \xi)$ and we consider the allocation process $W(t)$ described by (2.19) with any possible initial condition $W(0)$.

By the way it has been defined, the process $W(t)$ is Markovian conditioned to the functional state process $\xi(t)$. If we consider the augmented process $(W(t), \xi(t))$, this is Markovian and its only non zero transition rates are described below:

$$\begin{aligned} \Lambda_{(W,\xi),(W,\xi')} &= \begin{cases} \nu_{\bar{x}}^{on} & \text{if } \xi_{\bar{x}} = 0,\ \xi'_{\bar{x}} = 1,\ \xi_x = \xi'_x \forall x \neq \bar{x} \\ \nu_{\bar{x}}^{off} & \text{if } \xi_{\bar{x}} = 1,\ \xi'_{\bar{x}} = 0,\ \xi_x = \xi'_x \forall x \neq \bar{x} \end{cases} \\ \Lambda_{(W,\xi),(W',\xi)} &= \nu_{\bar{x}}^{act} P_{\bar{x}}^{(W,\xi)}(W') \quad \text{if } \xi_{\bar{x}} = 1,\ W' \in N_{\bar{x}}(W, \xi) \end{aligned} \tag{2.20}$$

We now introduce a graph on $\mathcal{W}_p$ that will be denoted by $\mathcal{L}_p$: an edge (W, W') is present in $\mathcal{L}_p$ if and only if $W' \in \mathcal{M}_{\bar{x}}(W, \mathbb{1})$. Notice that, if $\nu_{\bar{x}}^{act} > 0$ for every $\bar{x}$, this can be equivalently described as $\Lambda_{(W,\mathbb{1}),(W',\mathbb{1})} > 0$. The graph $\mathcal{L}_p$ thus describes the possible jumps of the process $W(t)$ conditioned to the fact that all resources are in functional state on. We want to stress the fact that the graph $\mathcal{L}_p$ depends on the triple $(\mathcal{G}, \alpha, \beta)$ as well on the choice of the admissible family $\mathcal{M}_{\bar{x}}(W, \mathbb{1})$ but not on the particular choice of the functional Ψ or of the utility functions $U_{\bar{x}}$.

Our strategy, in order to prove our first claim, will be to show that from any element $W \in \mathcal{W}_p$ there is a path in $\mathcal{L}_p$ to some element $W' \in \mathcal{W}$.

Given $W \in \mathcal{W}_p$ we define the following subsets of units

$$\mathcal{X}^f(W) := \{x \in \mathcal{X} \mid W^x = \alpha_x\},$$

$$\mathcal{X}^{sat}(W) := \{x \in \mathcal{X} \setminus \mathcal{X}^f(W) \mid \nexists y \in N_x \text{ s.t } W_y < \beta_y\}$$

Units in $\mathcal{X}^f(W)$ are called *fully allocated*: these units have completed the allocation of their data under the state W. Units in $\mathcal{X}^{sat}(W)$ are called *saturated*: they have

not yet completed their allocation, however, under the current state W, they can not make any action, neither allocate, nor distribute.

Finally, define

$$\mathcal{W}_p^{sat} := \{W \in \mathcal{W}_p \setminus \mathcal{W} \mid \mathcal{X} = \mathcal{X}^f(W) \cup \mathcal{X}^{sat}(W)\}$$

It is clear that from any $W \in \mathcal{W}_p \setminus \mathcal{W}_p^{sat}$, there exists units that can make either an allocation or a distribution move. Instead, if we are in a state $W \in \mathcal{W}_p^{sat}$, there are units that are not fully allocated and all these units con not make any move. The only units that can possibly make a move are the fully allocated ones. Notice that, because of condition (2.2), for sure there exist resources y such that $W_y < \beta_y$ and these resources are indeed exclusively connected to fully allocated units. The key point is to show that in a finite number of distribution moves, performed by fully allocated units, it is always possible to move some data atoms from resources connected to saturated units to resources with available space: this will then make possible a further allocation move.

For any fixed $W \in \mathcal{W}_p$, we can consider the following graph structure on $\mathcal{X}$ thought as set of resources: $\mathcal{H}_W = (\mathcal{X}, \mathcal{E}_W)$. Given $y_1, y_2 \in \mathcal{X}$, there is an edge from y_1 to y_2 if and only if there exists $x \in \mathcal{X}$ for which

$$W_{xy_1} > 0, \quad (x, y_2) \in \mathcal{E}$$

The edge from y_1 to y_2 will be indicated with the symbol $y_1 \to_x y_2$ (to also recall the unit x involved). The presence of the edge means that the two resources y_1 and y_2 are in the neighborhood of a common unit x that is using y_1 under W. This indicates that x can in principle move some of its data currently stored in y_1 into resource y_2 if this last one is available. We have the following technical result

Lemma 2.3.1. *Suppose $(\mathcal{G}, \alpha, \beta)$ satisfies (2.2). Fix $W \in \mathcal{W}_p$ and let $\bar{y} \in \mathcal{X}$ be such that there exists $\bar{x} \in N_{\bar{y}}$ with $W^{\bar{x}} < \alpha_{\bar{x}}$. Then, there exists a sequence*

$$\bar{y} = y_0, x_0, y_1, \ldots, y_{t-1}, x_{t-1}, y_t \tag{2.21}$$

satisfying the following conditions

(Sa) Both families of the y_k's and of the x_k's are each made of distinct elements;

(Sb) $y_k \rightarrow_{x_k} y_{k+1}$ *for every* $k = 0, \ldots, t-1$*;*

(Sc) $W_{y_k} = \beta_{y_k}$ *for every* $k = 0, \ldots, t-1$*, and* $W_{y_t} < \beta_{y_t}$.

Proof. Let $\mathcal{Y} \subseteq \mathcal{X}$ be the subset of nodes that can be reached from $\bar{y}$ in $\mathcal{H}_W$. Preliminarily, we prove that there exists $y' \in \mathcal{Y}$ such that $W_{y'} < \beta_{y'}$. Let

$$\mathcal{Z} := \{x \in \mathcal{X} \mid \exists y \in \mathcal{Y}, W_{xy} > 0\}$$

and notice that, by the way $\mathcal{Y}$ and $\mathcal{Z}$ have been defined,

$$x \in \mathcal{Z},\ (x,y) \in \mathcal{E} \ \Rightarrow\ y \in \mathcal{Y} \tag{2.22}$$

Suppose now that, contrarily to the thesis, $W_y \geq \beta_y$ for all $y \in \mathcal{Y}$. Then,

$$\begin{aligned} \sum_{x \in \mathcal{Z}} \alpha_x &\leq \sum_{y \in \mathcal{Y}} \beta_y \\ &= \sum_{y \in \mathcal{Y}} W_y \\ &= \sum_{y \in \mathcal{Y}} \sum_{x \in \mathcal{Z}} W_{xy} \\ &= \sum_{x \in \mathcal{Z}} W^x \\ &< \sum_{x \in \mathcal{Z}} \alpha_x \end{aligned} \tag{2.23}$$

where the first inequality follows from (2.22) and (2.2), the first equality from the contradiction hypothesis, the second equality from the definition of $\mathcal{Z}$, the third equality again from (2.22) and, finally, last inequality from the existence of $\bar{x}$. This is absurd and thus proves our claim.

Consider now a path of minimal length from $\bar{y}$ to $\mathcal{Y}$ in $\mathcal{H}_W$:

$$\bar{y} = y_0 \rightarrow_{x_0} y_1 \rightarrow_{x_1} \cdots \rightarrow_{x_{t-2}} y_{t-1} \rightarrow_{x_{t-1}} y_t$$

and notice that the sequence $\bar{y} = y_0, x_0, y_1, \ldots, y_{t-1}, x_{t-1}, y_t$ will automatically satisfy properties (Sa) to (Sc).

□

We are now ready to prove the first main result.

Theorem 2.3.2. *Assume that the following conditions hold*

1. $(\mathcal{G}, \alpha, \beta)$ *satisfies (2.2).*
2. $\mathcal{M}_{\bar{x}}(W, \xi)$ *is an admissible family.*

Then, for every $W \in \mathcal{W}_p$ there is a path in $\mathcal{L}_p$ to some element $W' \in \mathcal{W}$.

Proof. We will prove the claim by a double induction process. To this aim we consider two indices associated to any $W \in \mathcal{W}_p \setminus \mathcal{W}$. The first one is defined by

$$m_W = \sum_{x \in \mathcal{X}} (\alpha_x - W^x) \geq 1$$

To define the second index, consider any $\bar{x} \in \mathcal{X} \setminus \mathcal{X}^f(W)$. We can apply Lemma 2.3.1 to W and any $\bar{y} \in N_{\bar{x}}$ and obtain that we can find a sequence of agents $\bar{y} = y_0, x_0, y_1, \ldots, y_{t-1}, x_{t-1}, y_t$ satisfying the properties (Sa), (Sb), and (Sc) above. Among all the possible choices of $\bar{x} \in \mathcal{X}$, $\bar{y} \in N_{\bar{x}}$ and of the corresponding sequence, assume we have chosen the one minimizing t and denote such minimal t by t_W. The induction process will be performed with respect to the lexicographic order induced by the pair (m_W, t_W).

In the case when $t_W = 0$, it means we can find $\bar{x} \in \mathcal{X}$ and $\bar{y} \in N_{\bar{x}}$ such that $W_{\bar{y}} < \beta_{\bar{y}}$. This yields $\bar{y} \in N_{\bar{x}}(W, \mathbb{1})$. Hence, by property (i) in the definition of an admissible family, it follows that there exists n such that $W' = W + ne_{\bar{x}\bar{y}} \in \mathcal{M}_{\bar{x}}(W, \mathbb{1})$. Notice that $m_{W'} < m_W$. In case $m_W = 1$, this means that $W' \in \mathcal{W}$.

Consider now any $W \in \mathcal{W}_p \setminus \mathcal{W}$ such that $t = t_W > 1$. Let $\bar{x} \in \mathcal{X}$, $\bar{y} \in N_{\bar{x}}$ and the sequence $\bar{y} = y_0, x_0, y_1, \ldots, y_{t-1}, x_{t-1}, y_t$ satisfying the properties (Sa), (Sb), and (Sc) above. Since $W_{x_{t-1}y_{t-1}} > 0$ and $y_t \in N_{x_{t-1}}(W, \mathbb{1})$, it follows from property (ii) in the definition of admissible families that $W' = W - (e_{x_{t-1}y_{t-1}} - e_{x_{t-1}y_t}) \in \mathcal{M}_{x_{t-1}}(W, \mathbb{1})$. Since $W'_{y_{t-1}} < \beta_{y_{t-1}}$, for sure $t_{W'} < t_W$. The induction argument is thus complete.

□

Corollary 2.3.3. *Consider the process $W(t)$ as defined in (2.19) and assume that the following conditions hold*

1. $(\mathcal{G}, \alpha, \beta)$ *satisfies (2.2).*

2. $\nu_x^{on} > 0$ *and* $\nu_x^{act} > 0$ *for every* $x \in \mathcal{X}$,

3. $\mathcal{M}_{\bar{x}}(W,\xi)$ *is an admissible family.*

Then,

$$\mathbb{P}(\exists t_0 \,|\, W(t) \in \mathcal{W} \forall t \geq t_0) = 1$$

Proof. It follows from the form of the transition rates (2.20) and assumption 2), that the process $(W(t),\xi(t))$, starting from any initial condition (W,ξ), will reach $(W,\mathbb{1})$ in bounded time with positive probability. Combining with Theorem 2.3.2 and using again 2), it then follows that $(W(t),\xi(t))$ reaches a couple $(W',\mathbb{1})$ for some $W' \in \mathcal{W}$ in bounded time with positive probability. Since, by definition of an admissible family, the set $\{(W',\xi), W' \in \mathcal{W}\}$ is invariant by the process $(W(t),\xi(t))$, standard results on Markov processes yield the thesis. □

We are now left with studying the process $W(t)$ on $\mathcal{W}$. Noisy best response dynamics are known to yield reversible Markov processes. This is indeed the case also in our case once the process has reached the set of allocations $\mathcal{W}$. Precisely, the following result holds:

Proposition 2.3.4. *Suppose that* $\nu_x^{on}, \nu_x^{off} > 0$ *for all* $x \in \mathcal{X}$. *Then,* $(W(t),\xi(t))$, *restricted to* $\mathcal{W} \times \{0,1\}^{\mathcal{X}}$, *is a time-reversible Markov process. More precisely, for every* $(W,\xi),(W',\xi') \in \mathcal{W} \times \{0,1\}^{\mathcal{X}}$ *it holds*

$$\rho_{(W,\xi)}\Lambda_{(W,\xi),(W',\xi')} = \rho_{(W',\xi')}\Lambda_{(W',\xi'),(W,\xi)} \tag{2.24}$$

where

$$\rho_{(W,\xi)} = \left[\prod_{x:\,\xi_x=1} \nu_x^{\text{on}} \prod_{x:\,\xi_x=0} \nu_x^{\text{off}}\right] e^{\gamma\Psi(W)} \tag{2.25}$$

Proof. It follows from relations (2.20) and the definition of admissible families, that the only cases when $\Lambda_{(W,\xi),(W',\xi')}$ and $\Lambda_{(W',\xi'),(W,\xi)}$ are not both equal to zero are the following:

(i) $W' = W$, $\xi_{\bar{x}} = 0$, $\xi'_{\bar{x}} = 1$, $\xi_x = \xi'_x \forall x \neq \bar{x}$

(ii) $W' = W$, $\xi_{\bar{x}} = 1$, $\xi'_{\bar{x}} = 0$, $\xi_x = \xi'_x \forall x \neq \bar{x}$

(iii) $\xi' = \xi$, $W' \in N_{\bar{x}}(W,\xi)$.

In case (i), we have that

$$\frac{\rho_{(W,\xi)}}{\rho_{(W,\xi')}} = \frac{\prod\limits_{x:\,\xi_x=1} \nu_x^{\text{on}} \prod\limits_{x:\,\xi_x=0} \nu_x^{\text{off}}}{\prod\limits_{x:\,\xi'_x=1} \nu_x^{\text{on}} \prod\limits_{x:\,\xi'_x=0} \nu_x^{\text{off}}} = \frac{\nu_{\bar{x}}^{\text{off}}}{\nu_{\bar{x}}^{\text{on}}} = \frac{\Lambda_{(W,\xi'),(W,\xi)}}{\Lambda_{(W,\xi),(W,\xi')}}$$

Case (ii) can be analogously verified. Consider now case (iii). Using relations (2.12), (2.20), and (2.18), we obtain

$$\begin{aligned} \frac{\rho_{(W,\xi)}}{\rho_{(W',\xi)}} &= e^{\gamma(\Psi(W)-\Psi(W'))} = e^{\gamma(U_{\bar{x}}(W)-U_{\bar{x}}(W'))} \\ &= \frac{\Lambda_{(W',\xi),(W,\xi)}}{\Lambda_{(W,\xi),(W',\xi)}} \end{aligned}$$

□ □

We now show that under a slight stronger assumption than (2.2), namely,

$$\sum_{x\in A} \alpha_x < \sum_{y\in N(A)} \beta_y \quad \forall A \subseteq \mathcal{X}\,, \tag{2.26}$$

the process $(W(t),\xi(t))$ restricted to $\mathcal{W}\times\{0,1\}^{\mathcal{X}}$ is ergodic. Denote by $\mathcal{L}$ the subgraph of $\mathcal{L}_p$ restricted to the set $\mathcal{W}$. Notice that, as a consequence of time-reversibility, $\mathcal{L}$ is an undirected graph. Ergodicity is equivalent to proving that $\mathcal{L}$ is connected. We start with a lemma analogous to previous Lemma 2.3.1.

Lemma 2.3.5. *Suppose $(\mathcal{G},\alpha,\beta)$ satisfies (2.26) and let $W\in\mathcal{W}$. Then, for every $\bar{y}\in\mathcal{X}$, there exists a sequence (2.21) satisfying the conditions (Sa), (Sb), and (Sc) as in Lemma 2.3.1.*

Proof. It is sufficient to follow the steps of to the proof of Lemma 2.3.1 noticing that in (2.23) the first equality is now a strict inequality, while the last strict inequality becomes an equality. □

If $W,W'\in\mathcal{W}$ are connected through a path in $\mathcal{L}$, we write that $W\sim W'$. Introduce the following distance on $\mathcal{W}$: if $W^1,W^2\in\mathcal{W}$

$$\delta(W^1,W^2) = \sum_{x,y} |W^1_{xy} - W^2_{xy}|$$

A pair $\{W^1, W^2\} \in \mathcal{W}$ is said to be minimal if

$$\delta(W^1, W^2) \leq \delta(W^{1'}, W^{2'}) \ \forall W^{1'} \sim W^1, \forall W^{2'} \sim W^2$$

Notice that $\mathcal{L}$ is connected if and only if for any minimal pair $\{W^1, W^2\}$, it holds $W^1 = W^2$.

Lemma 2.3.6. *Let $\{W^1, W^2\}$ be a minimal pair. Suppose $y \in \mathcal{X}$ is such that $W^1_y < \beta_y$. Then, $W^1_{xy} = W^2_{xy}$ for all $x \in \mathcal{X}$.*

Proof. Suppose by contradiction that $W^1_{xy} < W^2_{xy}$ for some $x \in \mathcal{X}$. Then, necessarily, there exists $y' \neq y$ such that $W^1_{xy'} > W^2_{xy'}$. Consider then $W^{1'} = W^1 - e^{xy'} + e^{xy}$. Since $\delta(W^{1'}, W^2) < \delta(W^1, W^2)$, this contradicts the minimality assumption. Thus $W^1_{xy} \geq W^2_{xy}$ for all $x \in \mathcal{X}$. This yields $W^2_y < \beta_y$. Exchanging the role of W^1 and W^2 we obtain the thesis. □

Proposition 2.3.7. *If condition (2.26) holds true, the graph $\mathcal{L}$ is connected.*

Proof. Let $\{W^1, W^2\}$ be any minimal pair. We will prove that W^1 and W^2 are necessarily identical. Consider any resource y. It follows from Lemma 2.3.5 that we can find a sequence $y = y_0, x_0, y_1 \cdots, y_{t-1}, x_{t-1}, y_t$ satisfying the same (Sa), (Sb), and (Sc) with respect to the state allocation W^1. Among all the possible sequences, choose one with t minimal for given y. We will prove by induction on t that $W^1_{xy} = W^2_{xy}$ for all $x \in \mathcal{X}$.

If $t = 0$, it means that $W^1_y < \beta_y$. It then follows from Lemma 2.3.6 that $W^1_{xy} = W^2_{xy}$ for all $x \in \mathcal{X}$. Suppose now that the claim has been proven for all minimal pairs $\{W^1, W^2\}$ and any $y \in \mathcal{X}$ for which $t < \bar{t}$ (w.r. to W^1) and assume that $y = y_0, x_0, y_1 \cdots, y_{\bar{t}-1}, x_{\bar{t}-1}, y_{\bar{t}}$ satisfies the properties (Sa), (Sb), and (Sc) with respect to W^1.

Since $W_{x_{\bar{t}-1}y_{\bar{t}-1}} > 0$ and $y_{\bar{t}} \in N_{x_{\bar{t}-1}}(W^1, \mathbb{1})$, it follows from property (ii) in the definition of admissible families that $W'^1 = W^1 - (e_{x_{\bar{t}-1}y_{\bar{t}-1}} - e_{x_{\bar{t}-1}y_{\bar{t}}}) \in \mathcal{M}_{x_{\bar{t}-1}}(W^1, \mathbb{1})$. In other words, $W'^1 \sim W^1$.

Consider now W^2 and notice that Lemma 2.3.6 yields $W^2_{x_{\bar{t}-1}y_{\bar{t}}} = W^1_{x_{\bar{t}-1}y_{\bar{t}}} < \beta_{y_{\bar{t}}}$. Define

$$W^{2'} = \begin{cases} W^2 & \text{if} W^2_{x_{\bar{t}-1}y_{\bar{t}-1}} = 0 \\ W^2 - e_{x_{\bar{t}-1}y_{\bar{t}-1}} + e_{x_{\bar{t}-1}y_{\bar{t}}} & \text{if} W^2_{x_{\bar{t}-1}y_{\bar{t}-1}} > 0 \end{cases}$$

Again, by property (ii) in the definition of admissible families, it follows that $W'^2 \sim W^2$. Since $\delta(W^{1'}, W^{2'}) \leq \delta(W^1, W^2)$, this implies that also $\{W'^1, W'^2\}$ is a minimal pair. Notice that $y = y_0, x_0, y_1 \cdots, y_{\bar{t}-1}$ satisfies (Sa), (Sb), and (Sc) with respect to $W^{1'}$. Therefore, by the induction hypotheses, it follows that $W^{1'}_{xy} = W^{2'}_{xy}$ for all $x \in \mathcal{X}$. Since $W^1_{xy} = W^{1'}_{xy}$ and $W^2_{xy} = W^{2'}_{xy}$, result follows immediately. □

We can now state our final result.

Corollary 2.3.8. *Assume that the following conditions hold*

1. $(\mathcal{G}, \alpha, \beta)$ *satisfies (2.26).*
2. $\nu_x^{on} > 0$, $\nu_x^{off} > 0$, *and* $\nu_x^{act} > 0$ *for every* $x \in \mathcal{X}$,
3. $\mathcal{M}_{\bar{x}}(W, \xi)$ *is an admissible family.*

Then, $(W(t), \xi(t))$, *restricted to* $\mathcal{W} \times \{0,1\}^{\mathcal{X}}$, *is an ergodic time-reversible Markov process whose unique invariant probability measure is given by*

$$\mu_\gamma(\xi, W) = Z_\gamma^{-1} \left[\prod_{x:\, \xi_x = 1} \nu_x^{\text{on}} \prod_{x:\, \xi_x = 0} \nu_x^{\text{off}} \right] e^{\gamma \Psi(W)}$$

where Z_γ *is the normalizing constant.*

Proof. Let $(W, \xi), (W', \xi') \in \mathcal{W} \times \{0,1\}^{\mathcal{X}}$. It follows from the form of the transition rates (2.20) and the fact that $\nu_x^{on} > 0$ for all x, that the process $(W(t), \xi(t))$, starting from (W, ξ), will reach $(W, \mathbb{1})$ in bounded time with positive probability. Combining with Proposition 2.3.7 and using the fact that $\nu_x^{act} > 0$ for all x, it then follows that $(W(t), \xi(t))$ reaches $(W', \mathbb{1})$ in bounded time with positive probability. Finally, from $(W', \mathbb{1})$ again the process reaches (W', ξ') in bounded time with positive probability. This says that the process is ergodic and it thus possesses a unique invariant measure whose form can be derived by the time-reversibility property characterized in Proposition 2.3.4. □

Remark: It follows from previous result that the process $W(t)$ converges in law to the probability distribution

$$\tilde{\mu}_\gamma(W) := \tilde{Z}_\gamma^{-1} e^{\gamma \Psi(W)}$$

Notice that when $\gamma \to +\infty$, the probability $\tilde{\mu}_\gamma$ converges to a probability concentrated on the set $\operatorname{argmax}_{W \in \mathcal{W}} \Psi(W)$ of state allocations maximizing the potential. Thus, if γ is large, the distribution of the process $W(t)$ for t sufficiently large will be close to a maximum of Ψ.

Remark: Condition (2.26) is necessary for ergodicity. Notice indeed that in the case when $\mathcal{G}$ is complete and $\alpha_x = \beta_x = a$ for all $x \in \mathcal{X}$, under every allocation W such that $W_y = a$ for every y, all resources will be saturated and, consequently, no distribution move will be allowed in W. Such allocations W are thus all sinks in the graph $\mathcal{L}$ that is therefore not connected.

2.3.1 An example

We include in this chapter only the following, very easy but substantial example to validate the theoretical analysis. A wide set of simulation is shown in Chapter 6 where a number of parameter are calculated to measure the performance on different aspects as, for instance, the distance from the optimum or the number of mover per atom. In this example instead, we show the final state reached by the algorithm varying the aggregation parameter. We believe that these matrix are very expressive of the performance, since it is easy to recognize the equilibria. The very same example will be recover in Chapter 6 to show its behavior parameters.

Example 2.3.9. *Consider to have* $n = 10$ *users on a complete graph such that* $\alpha_x = a = 45$ *and* $\beta_x = b = 50$ *for every unit x. We consider the cases:* $C^{agg} = -7, -1, 1/2, 3$. *First, we show the final states reached by the dynamics for a single*

run of the algorithm

$$W_{-7} = \begin{pmatrix} 0 & 5 & 5 & 5 & 5 & 5 & 5 & 5 & 5 & 5 \\ 5 & 0 & 5 & 5 & 5 & 5 & 5 & 5 & 5 & 5 \\ 5 & 5 & 0 & 5 & 5 & 5 & 5 & 5 & 5 & 5 \\ 6 & 5 & 5 & 0 & 6 & 5 & 6 & 6 & 0 & 6 \\ 5 & 5 & 5 & 5 & 0 & 5 & 5 & 5 & 5 & 5 \\ 5 & 5 & 5 & 5 & 5 & 0 & 5 & 5 & 5 & 5 \\ 5 & 5 & 5 & 5 & 5 & 5 & 0 & 5 & 5 & 5 \\ 5 & 5 & 5 & 5 & 5 & 5 & 5 & 0 & 5 & 5 \\ 5 & 5 & 5 & 5 & 5 & 5 & 5 & 5 & 0 & 5 \\ 5 & 5 & 5 & 5 & 5 & 5 & 5 & 5 & 5 & 0 \end{pmatrix}$$

$$W_{-1} = \begin{pmatrix} 0 & 5 & 3 & 5 & 5 & 4 & 8 & 7 & 6 & 2 \\ 9 & 0 & 3 & 4 & 4 & 1 & 6 & 7 & 6 & 5 \\ 1 & 3 & 0 & 6 & 5 & 6 & 6 & 3 & 8 & 7 \\ 6 & 1 & 6 & 0 & 6 & 10 & 7 & 0 & 5 & 4 \\ 3 & 6 & 7 & 8 & 0 & 6 & 2 & 8 & 0 & 5 \\ 2 & 8 & 9 & 6 & 4 & 0 & 3 & 7 & 5 & 1 \\ 8 & 3 & 7 & 5 & 9 & 7 & 0 & 0 & 0 & 6 \\ 4 & 5 & 4 & 4 & 3 & 4 & 5 & 0 & 8 & 8 \\ 1 & 9 & 2 & 3 & 8 & 1 & 7 & 8 & 0 & 6 \\ 8 & 10 & 5 & 3 & 3 & 7 & 3 & 4 & 2 & 0 \end{pmatrix}$$

$$W_{1/2} = \begin{pmatrix} 0 & 0 & 23 & 7 & 0 & 4 & 0 & 0 & 10 & 1 \\ 3 & 0 & 12 & 7 & 2 & 7 & 2 & 5 & 0 & 7 \\ 1 & 4 & 0 & 2 & 15 & 2 & 7 & 1 & 12 & 1 \\ 15 & 6 & 2 & 0 & 2 & 14 & 0 & 1 & 4 & 1 \\ 4 & 4 & 0 & 1 & 0 & 4 & 13 & 0 & 3 & 16 \\ 18 & 0 & 0 & 1 & 3 & 0 & 8 & 2 & 8 & 5 \\ 2 & 7 & 6 & 12 & 2 & 3 & 0 & 0 & 5 & 8 \\ 1 & 4 & 1 & 9 & 12 & 1 & 9 & 0 & 4 & 4 \\ 0 & 17 & 1 & 1 & 4 & 1 & 0 & 16 & 0 & 5 \\ 2 & 5 & 0 & 1 & 6 & 10 & 2 & 17 & 2 & 0 \end{pmatrix}$$

$$W_3 = \begin{pmatrix} 0 & 0 & 45 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 45 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 45 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 45 & 0 & 0 & 0 \\ 0 & 0 & 0 & 45 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 45 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 45 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 45 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 45 \\ 0 & 0 & 0 & 0 & 45 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

For $C^{agg} = 3$ a matching allocation state is reached and it is a maximum of Ψ in this case. For $C^{agg} = -7$ the solution is also very close to the maximum that is the diffused allocation state. For $C^{agg} = 1/2, -1$, the presence of Nash equilibria that are not maxima of Ψ slows down the dynamics and the algorithm does not reach the maximum at time T. Increasing in this case the time horizon to $T = 20 * \sum \alpha$, the final state of the system gets quite close to the maximum (this will be confirmed in Chapter 6).

References

[1] ALÓS-FERRER, C., AND NETZER, N. The logit-response dynamics. *Games and Economic Behavior 68*, 2 (2010), 413–427.

[2] ARSLAN, G., MARDEN, J. R., AND SHAMMA, J. S. Autonomous vehicle-target assignment: A game-theoretical formulation. *Journal of Dynamic Systems, Measurement, and Control 129*, 5 (2007), 584–596.

[3] BLUME, L. E. The statistical mechanics of strategic interaction. *Games and economic behavior 5*, 3 (1993), 387–424.

[4] FAGNANI, F., FRANCI, B., AND GRASSO, E. A game theoretic approach to a peer-to-peer cloud storage model.

[5] MARDEN, J. R., AND SHAMMA, J. S. Revisiting log-linear learning: Asynchrony, completeness and payoff-based implementation. In *Communication,*

Control, and Computing (Allerton), 2010 48th Annual Allerton Conference on (2010), IEEE, pp. 1171–1172.

[6] MARDEN, J. R., AND WIERMAN, A. Distributed welfare games. *Operations Research 61*, 1 (2013), 155–168.

[7] MONDERER, D., AND SHAPLEY, L. S. Potential games. *Games and economic behavior 14*, 1 (1996), 124–143.

[8] ROSENTHAL, R. W. A class of games possessing pure-strategy nash equilibria. *International Journal of Game Theory 2*, 1 (1973), 65–67.

[9] YOUNG, H. P. The evolution of conventions. *Econometrica: Journal of the Econometric Society* (1993), 57–84.

Chapter 3

Inducing reciprocity in the model

3.1 Introduction

In this chapter, we consider a variant of the algorithm proposed in Chapter 2, giving the resources the possibility to deny the allocation from a unit. We suppose that the framework is the same as in Chapter 2 namely that there is a network of user who are at the same time a source of data and a resource for storage.

The agents autonomously and at random activate and allocate or move their data pieces among the neighboring units. Each unit has a utility function which gives a value to their neighbors on the basis of their reliability, their current congestion, and the amount of data the unit has already stored in them. Since the users are evaluated by their reliability, we want to emphasize this aspect giving them a mechanism to eventually exclude non trustworthy agents. To do that, we allow the resources to evaluate the users, depending on his/her previous behavior, and to decide whether to accept or deny the allocation. Including this mechanism induce a sort of reciprocity in the dynamic forcing the mutual exchange of data. Intuitively, this is because more reliable users are interested in using resources that are trustworthy as well. On the other hand, resources are willing to accept data from agents that accepted stortage already. This aspect is fundamental to avoid selfish behaviors.

In this chapter we also present a learning algorithm so that the agents have the possibility to know the reliability of the neighbors during the allocation. This is a realistic approach since in principle, when the users connect to the network, they are not aware of the trustworthiness of the other agents.

Reputation based algorithms, i.e. those process in which the dynamic is strictly related to the reliability of the agents, are widely studied in literature. The remaining part of this section is devoted to their revision. In the next section we recall some details of the model and we describe the reciprocity process and the learning algorithm. In Section 3.3 we briefly describe the algorithm and prove the convergence results in this case with reciprocity.

3.1.1 Reliability and reputation

In literature, there can be found many works on reputation-based systems but, as far as we know, there are no paper where this mechanism is purposely related to an allocation problem. On the other hand, some of the application of this methods are on peer-to-peer systems [2, 4] and on the game theoretic approach [1, 5].

In this thesis we measure the reputation through the reliability parameter, i.e., a quantity that express how much a resource is trustworthy. The level of reliability in general depends on many factors (usually related to the quality of services), for instance:

- the probability that, if contacted at random time, the resource is available to give access to the stored data,
- the safety of the stored data from internal or external attacks,
- the speed of the transfer.

Reputation-based systems are used to establish trust among agent on a network. The idea is that the behavior of past interactions determines the reliability of the other users. Every agent maintains its own opinion on every other user he/she has dealt with in the past and uses this information for possible interactions in the future. This same idea of evaluating the agents on the past interaction is also used in [1].

The reason of this studies on reciprocity is that the intentions of the agents are generally unknown. Hence, due to the uncertainty of their potential behavior we need mechanisms to control the interactions among the agents, and protect good agents from the selfish ones. Agents themselves can be capable of punishing non-desirable behaviors, by for instance, not selecting certain resources.

There are plenty of ways through which agents can obtain informations about a user [3], among these: direct experiences, information gathered from other agents or prejudice (not necessarily with a negative connotation).

The information can be stored by each agent (distributed) or by a global third-party (centralized). There are no clear strategies due to their high dependency on the context and both the cases are valuable. Clearly, in our case agents themselves will store their own informations.

3.2 Learning reliability and inducing reciprocity

In this section, we describe two processes: (i) a learning process from which the users get to know the reliability of their neighbors (ii) a reciprocity process that enforce the mutual exchange of data. While the first one is independent on the algorithm and do not directly affect it, the latter can alter the allocation/distribution modifying the cooperation between the agents. In fact, the second is a part of the algorithm and it influences the behavior of the users and the theoretical analysis.

Indicate the reliability parameter with $\lambda_x, x \in \mathcal{X}$. This quantity express how much resource x is trustworthy. We take λ_x to be positive for all $x \in \mathcal{X}$ imaging that a resource with negative reliability would never be chosen by the users. Recall that the utility function of each user x is:

$$U_x(W) = f_x(W_{x\cdot}) + \sum_{y \in N_x^-} g_y(W_{\cdot y})$$

where W is the state of the system as defined in (2.10) and the two functionals are

$$f_x(W_{x\cdot}) = C^{all} \sum_y W_{xy} + C^{agg} \sum_{y \in \mathcal{X}} W_{xy}^2,$$

$$g_y(W_y) = -C_y^{con}(W_y)^2 \quad \text{or} \quad g_y(W_{\cdot y}) = -C_y^{con}|W_{\cdot y}|_H$$

As we already explained in Chapter 2, the reliability parameter can be included in the congestion term of the utility function. Depending on the interpretation of the reliability parameter, we have the following cases.

- If we give the user an incentive to use more reliable resources, we set

$$-C_y^{con} = \mathcal{C}(\lambda_y) > 0$$

for some positive continuous function $\mathcal{C}$ increasing in λ_y, $y \in \mathcal{X}$. In this case, a very easy example is to consider the reliability to be the probability of a resource to be on;

- On the other hand, to have a classical congestion term one can see the reliability as a negative parameter to discourage the use of more congested and less reliable resources. In this case we can consider the reliability to be the probability of the resource to be off.

When it is necessary, we call *homogenous* the case where all the users have the same reliability. From now on, for simplicity, we consider the reliability as the probability of finding a resource in functional state on, therefore $\lambda_x \in [0,1]$ for all $x \in \mathcal{X}$. We recall that the functional state is indicated with $\xi \in \{0,1\}^{\mathcal{X}}$ and when we refer to a single user x we write ξ_x. Now we can describe the learning algorithm.

3.2.1 A learning mechanism

Suppose the users do not know a priori if a resource is reliable and they learn it during the game. In general, the choice of an initial trust value depends on the strategy followed: it can represent complete distrust, complete trust or neutral trust [2]. We suppose that the initial value of λ_x is supposed to be equal 1 for each resource $x \in \mathcal{X}$. This assumption means that the units consider every resource as trustworthy. Moreover, it ensure that at the beginning some atoms are allocated. Each user learn the reliability of the resources he/she is allowed to use and each unit updates his/her own opinion independently from the other users and only when the unit is on. The details of the process are described below.

Each time a unit x wakes up, he/she is allowed to see the functional state ξ_y of his/her neighbor y. We call λ_{xy} the *estimated reliability*, i.e., how x estimates the reliability of resource y. We can describe the learning process as

$$\lambda_{xy}(t+1) = \frac{t}{t+1}\lambda_{xy}(t) + \frac{1}{t+1}\xi_y(t+1) \tag{3.1}$$

where $\xi_y(t+1)$ is the functional state of y at time $t+1$. To be more precise we should write t_{xy} to indicate the dependency of the time variable on the agents. This time variable counts the number of times that users x interacts with resource y. Since $\xi_y \in \{0,1\}$, (3.1) represent the fraction of time that user x found y in functional state on.

Lemma 3.2.1. *The sequences of* $\{\lambda_{xy}(t)\}_{t\in\mathbb{R}}$ *converges to* λ_y *for* $t \to \infty$ *for all* $x, y \in \mathcal{X}$.

Proof. Follow from the Law of Large Numbers. □

3.2.2 The Reciprocity Process

In this section we describe the reciprocity process. From now on, when we consider a couple of agents, we suppose that both the units are in functional state on. An interesting point of view on the choice of probabilities is the one from [1]: the agents measure the size of the gifts received in the previous interactions and weight the other agents with these informations.

On the other hand, our reciprocity mechanism is guided by a quantity, called from now on *probability of acceptance* that influence the decisions of the resources. In fact the probability of acceptance $\rho_{yx}(t)$ represent the probability that resource y accept data from user x. The time variable t is a user dependent variable since it counts the number of interaction between user x and resource y. We avoid to write t_{xy} to lighten the reading.

We propose three different probabilities of acceptance: the first one take into consideration the number of times that a resource refused the allocation; the second emphasizes on the mutual behavior considering the reliability of the two users and the quantities already allocated; the latter evaluates the behavior of the units as a source of data and as a resource. For simplicity in the following definitions we suppose that λ_y is fixed and know by all the agents a priori. Clearly, if we introduce the learning algorithm also this parameter will be time and user dependent. The three probabilities are defined as follow.

1. (**Number of refusal**) First, we define the *estimated probability of acceptance*

$$\omega_{yx}(t+1) = \frac{t}{t+1}\omega_{yx}(t) + \frac{1}{t+1}\mu(t+1).$$

This quantity is learned in time and depends on the variable $\mu(t)$ that take values

$$\mu(t+1) = \begin{cases} 1 \text{ if } x \text{ accept} \\ 0 \text{ otherwise.} \end{cases}$$

ω_{yx} counts the fraction of time that x accepted data from y therefore it represent an estimation of the probability that resource x will accept the storage in the future. Notice that in this case the time variable counts the number of times that user y tried to allocate in resource x. We set

$$\eta_x^y(t) = \lambda_x * \omega_{yx}(t).$$

to be a weighted estimation that include also the reliability of the user who wants to allocate. Taking into consideration λ_x is in the interest of resource y for his/her future allocation since it gives preference to more reliable resources. Finally, the probability of acceptance is defined as

$$\rho_{yx}(t) = \min\left\{\frac{\eta_x^y(t) + \lambda_x W_{yx}(t)}{\lambda_y(W_{xy}(t)+1)}, 1\right\} \tag{3.2}$$

Taking the minimum with 1 is to guarantee that ρ is a probability; we will prove this result later in this chapter.

2. (**Mutual behavior**) A more intuitive approach is to emphasize the mutual behavior, considering the reliability and the quantities already allocated. For this reason the probability of acceptance depend on the reliabilities of the two agents, λ_x and λ_y, and the quantities $W_{yx}(t)$ and $W_{yx}(t)$. In fact, we define

$$\rho_{yx}(t) = \min\left\{\frac{\lambda_x(W_{yx}(t)+1)}{\lambda_y(W_{xy}(t)+1)}, 1\right\}. \tag{3.3}$$

Since (3.3) is always positive, it is a probability. Moreover, it is a particular case of the previous probability of acceptance ($\omega_{yx}(t) = 1$ for all $t \geq 0$ or, equivalently, $\eta_x^y(t) = \lambda_x$).

3. (**Source/Resource**) The last case evaluates a unit comparing his/her behavior as a source and as a resource:

$$\begin{aligned}\rho_{yx}(t) &= \delta\frac{\lambda_x W_{yx}(t)}{\sum\limits_{x\in N_y}\lambda_x W_{yx}(t)} + (1-\delta)\frac{\lambda_y W_{yx}(t)}{\sum\limits_{x\in N_y}\lambda_y W_{yx}(t)} \\ &= \delta\frac{\lambda_x W_{yx}(t)}{\sum\limits_{x\in N_y}\lambda_x W_{yx}(t)} + (1-\delta)\frac{W_{yx}(t)}{\sum\limits_{x\in N_y} W_{yx}(t)}\end{aligned} \tag{3.4}$$

where $\delta \in [0,1]$ is a tuning parameter. The first term evaluates user y, who acts as a resource, as a source comparing the quantity allocated in x with respect to the other resources in his/her neighborhood; the second term evaluates y as a resource counting the data allocated by x in relation with the data allocate by the other sources in the neighborhood. In other words, this probability compares the behavior of the resource with respect to the agent who wants to allocate and to the resources in the neighborhood.

The following lemma gives a condition for the probability of acceptance as defined in equation (3.2) to be positive. From this lemma it follows that (3.2) is a probability.

Lemma 3.2.2. *At any time t, the probability of acceptance defined in equation (3.2) is positive if the initial estimated probability $\omega_{yx}(0) > 0$.*

Proof. Since the probability of acceptance depends on the allocated data, we study when those quantities are zero. Suppose that at time $t > 0$ units x and y are interacting.

- If $W_{yx}(t) \neq 0$ and $W_{yx}(t) \neq 0$

$$\frac{\lambda_x \omega_{yx}(t) + \lambda_x W_{yx}(t)}{\lambda_y (W_{yx}(t)+1)} > 0$$

- If $W_{yx}(t) \neq 0$ and $W_{yx}(t) = 0$

$$\frac{\lambda_x \omega_{yx}(t) + \lambda_x W_{yx}(t)}{W_{yx}(t)+1} = \frac{\lambda_x \omega_{yx}(t)}{\lambda_y (W_{yx}(t)+1)}$$

- If $W_{yx}(t) = 0$ and $W_{yx}(t) = 0$

$$\frac{\lambda_x \omega_{yx}(t) + \lambda_x W_{yx}(t)}{\lambda_y (W_{yx}(t) + 1)} = \frac{\lambda_x \omega_{yx}(t)}{\lambda_y}$$

In the last two cases, the fraction is positive if $\omega_{yx}(t) > 0$ and this follow by the definition assuming $\omega_{yx}(0) > 0$. □

Taking $\omega_{yx}(0) > 0$ is a realistic choice: it means that at the beginning agents trust each other and suppose that other neighbors will accept their their data for storage.

3.3 The algorithm with reciprocity

The allocation algorithm we are proposing is a variation of the algorithm proposed in Chapter 2. It is fully distributed, asynchronous and based on communications between units connected on the graph $\mathcal{G} = (\mathcal{X}, \mathcal{E})$.

At every instant of time, a unit is either in functional state on or off. A unit, which is currently in state on, can activate and either allocate or move an atomic piece of its data among the available resources. The choice of the available resource where to allocate will be following a noisy best response dynamics where the utility maximization is based on a Gibbs probability distribution. A resource accepts the allocation following one of the probabilities (3.2)-(3.4). The functional state of the network at a certain time will be denoted by $\xi \in \{0,1\}^{\mathcal{X}}$: $\xi_x = 1$ means that the unit x is on. The times when units modify their functional state (off to on or on to off) and the times when units in functional state on activate are modeled as a family of independent Poisson clocks whose rates will be denoted (for unit x), respectively, ν_x^{on}, ν_x^{off}, and ν_x^{act}. The functional state of the network as a function of time $\xi(t)$ is thus a continuous time Markov process whose components are independent Bernoulli processes.

We now recall some notation and definitions from Chapter 2. Given a (possibly partial) allocation state $W \in \mathcal{W}_p$, a functional state $\xi \in \{0,1\}^{\mathcal{X}}$, and a unit $\bar{x} \in \mathcal{X}$ such that $\xi_{\bar{x}} = 1$, the possible partial allocation states obtainable from W by modifications

done by the unit $\bar{x}$ are:

$$\mathcal{W}_{\bar{x}}(W,\xi) = \left\{ W' \in \mathcal{W}_p : \begin{array}{l} W'_{xy} = W_{xy} \text{ if } x \neq \bar{x} \text{ or } \xi_y = 0 \\ W'^{\bar{x}} \geq W^{\bar{x}},\ W' \neq W \end{array} \right\}. \tag{3.5}$$

Since the sets $\mathcal{W}_{\bar{x}}(W,\xi)$ can in general be very large, it is convenient to consider a smaller set of actions where units either allocate new data or simply move data from one resource to another one. Given $(W,\xi) \in \mathcal{W}_p \times \{0,1\}^{\mathcal{X}}$ and a unit $\bar{x}$, set

$$N_{\bar{x}}(W,\xi) := \{ y \in N_{\bar{x}} \,|\, W_{\bar{x}y} < \beta_y,\ \xi_y = 1 \} \tag{3.6}$$

the set of available neighbor resources for $\bar{x}$ under the allocation state W and the functional state ξ: those that are on and still have space available.

A family of sets $\mathcal{M}_{\bar{x}}(W,\xi) \subseteq \mathcal{W}_{\bar{x}}(W,\xi)$, defined for each $\bar{x} \in \mathcal{X}$ and each $(W,\xi) \in \mathcal{W}_p \times \{0,1\}^{\mathcal{X}}$, is called *admissible* if

(i) $W^{\bar{x}} < \alpha_{\bar{x}},\ y \in N_{\bar{x}}(W,\xi), \Rightarrow \exists n : W' = W + n e_{\bar{x}y} \in \mathcal{M}_{\bar{x}}(W,\xi)$;

(ii) $W_{\bar{x}y'} > 0,\ \xi_{y'} = 1,\ y'' \in N_{\bar{x}}(W,\xi) \Rightarrow W' = W + (e_{\bar{x}y''} - e_{\bar{x}y'}) \in \mathcal{M}_{\bar{x}}(W,\xi)$;

(iii) $W' \in \mathcal{M}_{\bar{x}}(W,\xi)$ iff $W \in \mathcal{M}_{\bar{x}}(W',\xi)$ for every $W \in \mathcal{W}$.

Examples of admissible families $\mathcal{M}_{\bar{x}}(W,\xi)$ can be found in Chapter 2, Section 2.2.1.

Given an admissible family $\mathcal{M}_{\bar{x}}(W,\xi)$, we recall the Gibbs measure on it. Given a parameter $\gamma > 0$, put

$$\begin{aligned} Z_{\bar{x}}^{(W,\xi)}(\gamma) &= \textstyle\sum_{\tilde{W} \in \mathcal{M}_{\bar{x}}(W,\xi)} e^{\gamma U_{\bar{x}}(\tilde{W})} \\ Z_{\bar{x}}^{(W,W',\xi)}(\gamma) &= \max\left\{ Z_{\bar{x}}^{(W,\xi)}(\gamma), Z_{\bar{x}}^{(W',\xi)}(\gamma) \right\} \end{aligned} \tag{3.7}$$

Now define, for $W' \in \mathcal{M}_{\bar{x}}(W,\xi)$,

$$P_{\bar{x}}^{(W,\xi)}(W') = \begin{cases} \frac{e^{\gamma U_{\bar{x}}(W')}}{Z_{\bar{x}}^{(W,\xi)}(\gamma)}, & \text{if } ||W|| < ||W'|| \\ \frac{e^{\gamma U_{\bar{x}}(W')}}{Z_{\bar{x}}^{(W,W',\xi)}(\gamma)}, & \text{if } ||W|| = ||W'|| \end{cases} \tag{3.8}$$

where $||W|| = \sum_{xy} W_{xy}$, and complete it to a probability by putting

$$P_{\bar{x}}^{(W,\xi)}(W) = 1 - \sum_{W' \in \mathcal{M}_{\bar{x}}(W,\xi)} P_{\bar{x}}^{(W,\xi)}(W')$$

The algorithm is completely determined by the choice of the admissible family $\mathcal{M}_{\bar{x}}(W,\xi)$, the probabilities (3.8) and the probability of acceptance (3.2)-(3.4). If unit $\bar{x}$ activates at time t, the systems is in partial allocation state $W(t)$, and in functional state $\xi(t)$, it will jump to the new partial allocation state W' with probabilities given by

$$P(W(t+) = W') = P_{\bar{x}}^{(W(t),\xi(t))}(W')\rho(W(t),W'),\ W' \in \mathcal{M}_{\bar{x}}(W(t),\xi(t)) \qquad (3.9)$$

With an abuse of notation we write $\rho(W,W')$ to indicate the probability of acceptance in relation with the allocation states:

$$\rho(W(t),W') = \rho_{y\bar{x}}(t) \quad \text{if } \xi_{\bar{x}} = 1, W' \in \mathcal{M}_{\bar{x}}(W(t),\xi), W'_{\bar{x}y} > W_{\bar{x}y}(t) \qquad (3.10)$$

that is, $\bar{x}$ is trying to allocate/move in resource y. If unit $\bar{x}$ chooses a W' such that $||W'|| > ||W||$ we say that it makes an *allocation move*, otherwise, if $||W'|| = ||W||$, we talk of a *distribution move*.

3.3.1 Analysis of the algorithm

We now show the convergence of the algorithm with reciprocity. We prove that if the allocation is possible, the algorithm will reach such an allocation in bounded time with probability 1. We recall that a necessary and sufficient condition for the allocation is that

$$\sum_{x \in D} \alpha_x \leq \sum_{y \in N(D)} \beta_y \quad \forall D \subseteq \mathcal{X} \qquad (3.11)$$

This result was proven in Theorem 2.1.1 in Chapter 2.

Notice that, also in this case, the process $W(t)$ is a Markov process conditioned to $\xi(t)$, the functional state process. Therefore, if we consider the process $(W(t),\xi(t))$ including the reciprocity probabilities, we have that the process is Markovian and

the transition rates are:

$$\Lambda_{(\xi,W),(\xi',W)} = \begin{cases} \nu_{\bar{x}}^{on} & \text{if } \xi_{\bar{x}} = 0,\ \xi'_{\bar{x}} = 1,\ \xi_x = \xi'_x \forall x \neq \bar{x} \\ \nu_{\bar{x}}^{off} & \text{if } \xi_{\bar{x}} = 1,\ \xi'_{\bar{x}} = 0,\ \xi_x = \xi'_x \forall x \neq \bar{x} \end{cases} \tag{3.12}$$

$$\Lambda_{(\xi,W),(\xi,W')} = \nu_{\bar{x}}^{act} P_{\bar{x}}^{(W,\xi)}(W')\rho(W,W') \quad \text{if } \xi_x = 1, W' \in \mathcal{M}_{\bar{x}}(W,\xi) \tag{3.13}$$

The statements proven in Chapter 2 holds also in this case. In fact, considering that there is a non-zero probability that the process $(\xi(t), W(t))$ starting from (ξ, W) will reach $(\mathbb{1}, W)$ in bounded time (because of the presence of transitions (3.12)), standard probabilistic arguments will then allow to conclude that allocation will be achieved in bounded time with probability 1.

Lemma 2.3.1 that we proved in Section 2.3 holds also in this case regardless of the reciprocity process. Given the sequence of Lemma 2.3.1, we can prove the equivalent of Theorem 2.3.2 that is, we can prove that it is possible to reach a full allocation state on the graph $\mathcal{L}_p$ where an edge (W, W') is present if and only if $W' \in \mathcal{M}_{\bar{x}}(W, \mathbb{1})$. From this result it follows the convergence result with probability of acceptance defined as in (3.2) and (3.3).

Theorem 3.3.1. *Assume that*

1. $\nu_x^{on} > 0$ *and* $\nu_x^{act} > 0$ *for every* $x \in \mathcal{X}$,
2. $\mathcal{M}_{\bar{x}}(W, \xi)$ *is an admissible family,*
3. $(\mathcal{G}, \alpha, \beta)$ *satisfies (3.11)*
4. $\rho(W, W')$ *is defined as in (3.2) and* $\omega_{yx}(0) > 0$ *for all* $y, x \in \mathcal{X}$

Then,

$$\mathbb{P}(\exists t_0 \,|\, W(t) \in \mathcal{W} \forall t \geq t_0) = 1$$

Proof. Follows with the same reasoning of Theorem 2.3.2 in Chapter 2, considering that there is a positive probability of acceptance. □

Theorem 3.3.2. *Assume that*

1. $\nu_x^{on} > 0$ *and* $\nu_x^{act} > 0$ *for every* $x \in \mathcal{X}$,
2. $\mathcal{M}_{\bar{x}}(W, \xi)$ *is an admissible family,*
3. $(\mathcal{G}, \alpha, \beta)$ *satisfies (3.11)*
4. $\rho(W, W')$ *is defined as in equation (3.3).*

Then,

$$\mathbb{P}(\exists t_0 \,|\, W(t) \in \mathcal{W} \forall t \geq t_0) = 1.$$

Proof. Follow because (3.3) is a particular case of (3.2). □

3.4 Conclusion

In this chapter we described a variant of the allocation algorithm proposed in Chapter 2. We introduce a learning process to allow the agents to know the reliability of his/her neighbors proving the convergence to the exact value of λ_x for all $x \in \mathcal{X}$. Moreover we study the algorithm with reciprocity including a process that influence the allocation depending on the previous behavior of the agents. For this algorithm we prove that an allocation state can be reached in bounded time with probability 1.

References

[1] BONACICH, P., AND LIGGETT, T. M. Asymptotics of a matrix valued markov chain arising in sociology. *Stochastic Processes and their Applications 104*, 1 (2003), 155–171.

[2] KOUTROULI, E., AND TSALGATIDOU, A. Reputation-based trust systems for p2p applications: design issues and comparison framework. *Trust and privacy in digital business* (2006), 152–161.

[3] PINYOL, I., AND SABATER-MIR, J. Computational trust and reputation models for open multi-agent systems: a review. *Artificial Intelligence Review 40*, 1 (2013), 1–25.

[4] SELCUK, A. A., UZUN, E., AND PARIENTE, M. R. A reputation-based trust management system for p2p networks. In *Cluster Computing and the Grid, 2004. CCGrid 2004. IEEE International Symposium on* (2004), IEEE, pp. 251–258.

[5] SKYRMS, B., AND PEMANTLE, R. A dynamic model of social network formation. In *Adaptive networks*. Springer, 2009, pp. 231–251.

Chapter 4

Estimation of the allocation time

4.1 Introduction

In this chapter, we study the convergence time of the allocation algorithms described in the previous chapters. Recall that in our allocation problem a population of units is connected through a network and each of the units possesses a number of items that need to be allocated among the neighboring resources.
We focus on estimating the allocation time, analyzing the different algorithms. The goal is to estimate the average time for the allocation or at least to give some bounds. Indeed, we give an upper bound using the Coupon Collector's problem [4] and using the properties of the associated Markov chain [3] we find the average allocation time of the algorithms described in Chapter 2 and in Chapter 3.

In the next section we recall the algorithms studied, then we analyze the convergence time. Section 4.3 is devoted to computing the upper bound and the average time for allocation is calculated in Section 4.4. The chapter ends with a simulation section, that validates the estimations, and some conclusions.

4.2 The model and the algorithms

In this section we briefly describe the model and we recall some notation. Consider a set $\mathcal{X}$ of units which play the double role of users who have to allocate externally a back up of their data, as well resources where data from other units can be allocated.

We assume units to be connected through a directed graph $\mathcal{G} = (\mathcal{X}, \mathcal{E})$. Each unit x is characterized by two non negative integers: α_x which is the number of data atoms that unit x needs to back up, and β_x which is the number of data atoms that unit x can accept and store from his neighbors; they are assembled in two vectors α and β. Recall that the data possessed by the units can be seen as quantized atoms of the same size. In fact, $\mathcal{A}$ is the set of all atoms to be allocated:

$$\mathcal{A}_x = \{(x,a) \,|\, a \in \{1,\dots,\alpha_x\}\}, \quad \mathcal{A} = \bigcup_{x\in\mathcal{X}} \mathcal{A}_x.$$

Finally, each agent has a reliability λ_x that for simplicity, from now on, we interpret as the probability of finding a resource in functional state on following the considerations in Chapter 3. The functional state of the network at a certain time is denoted by $\xi \in \{0,1\}^{\mathcal{X}}$: $\xi_x = 1$ means that the unit x is on.

The allocation algorithm we are proposing is fully distributed, asynchronous and based on communications between units connected on the graph. We consider the fact that units can be disconnected from the network and that resources can accept or deny the allocation from other users.

From now on, we call *homogenous case* the case where all users have the same amount of data to allocate $\alpha_x = a$ for all $x \in \mathcal{X}$. If moreover the users have also the same reliability $\lambda_x = l$ for all $x \in \mathcal{X}$, we call the case *uniform*.

4.3 An upper bound for the homogeneous case

Consider the homogenous case and, without loss of generality, suppose they also have the same amount of available space $\beta_x = b$ for all $x \in \mathcal{X}$. We suppose that resources are always on and we do not include the reciprocity process.

Consider the set $\mathcal{A}$ of all atoms; since all the users have the same amount of data, the probability to pick an atom (and allocate it) is equal for all atoms (x,a). For this reason, the problem can be interpreted as the Coupon Collector's Problem [4]. The Coupon Collector's Problem answer the question of how many trials are necessary to collect all the types of coupons. In the following subsection we describe in details the Coupon Collector's Problem and recall some results on the expected number of trials, then we show how this estimation can be useful in the allocation problem.

4.3.1 The Coupon Collector's Problem

The CEO of Bubbleburst bubble gum company decides to include a picture card of a famous football player in each pack of bubble gum to be sold. A complete set of cards consists of ten players and the distribution of the cards is uniform: a pack of gum is just as likely to contain a picture of anyone of the ten players. How many packs of bubble gum does a football fan have to buy, on the average, to complete the set? This example, taken from [2], is only one of the possible applications of the Coupon Collector's Problem. In the following we consider a more general case, following the approach used in [4].

Suppose to have k types of coupons and that at each trial a coupon is chosen at random. Each type of coupon is equally likely and the random choices are mutually independent. The aim of the problem is to study the relation between the number of trials and the probability of collecting at least one copy of each type of coupon.

Let T be a random variable representing the number of trials required to collect at least one copy of each type of coupon. We first determine the expected value of T. Let $C_1, C_2, \ldots, C_n$ be the sequence of trials where $C_i \in \{1, \ldots, k\}$ denotes the type of the coupon drawn in the i-th trial. We call a trial C_i a *success* if the type C_i was not drawn in any of the previous $i-1$ selections. Clearly C_1 and C_T are always successes. Divide the sequence into epochs: each epoch i begins with the i-th successful trial and ends with the trial of the $(i+1)$-th success. Let T_i, for $0 \leq i \leq n-1$, be the random variable representing the number of trials in the i-th epoch, so that

$$T = \sum_{i=0}^{n-1} T_i.$$

Let p_i denote the probability of success on any trial of the i-th epoch, i.e.,

$$p_i = \frac{k-i}{k},$$

the probability of drawing one of the $n-i$ remaining coupon types. Since the random variable T_i is geometrically distributed with parameter p_i, it holds that

$$\mathbb{E}[T_i] = \frac{1}{p_i} \quad \text{and} \quad Var(T_i) = \frac{1-p_i}{p_i}$$

By linearity of the expected value,

$$\mathbb{E}[T] = \mathbb{E}\left[\sum_{i=0}^{k-1} T_i\right] = \sum_{i=0}^{k-1} \mathbb{E}[T] = \sum_{i=0}^{k-1} \frac{k}{k-i} = k\sum_{i=1}^{k} \frac{1}{i} = kH_k.$$

H_k is the k-th Harmonic number and it is asymptotically equal to $\ln k + \Theta(1)$, then

$$\mathbb{E}[T] = k\ln k + O(k)$$

We now recall some results from [4] that show that the value of T is sharply concentrated around its expected value, that is, it is unlikely to deviate from its expected value.

Theorem 4.3.1 (Theorem 3.8 in [4])**.** *Let the random variable T denote the number of trials for collecting each of the k types of coupons. Then, for any constant $c \in \mathbb{R}$, and $m = k\ln k + ck$,*

$$\lim_{k\to\infty} \mathbb{P}[T > m] = 1 - e^{-e^{-c}}.$$

In the following corollary we summarize some other estimation that follows form Theorem 4.3.1.

Corollary 4.3.2. *Let the random variable T denote the number of trials for collecting each of the k types of coupons. Then, for any constant $c \in \mathbb{R}$, it holds*

- $\lim_{k\to\infty} \mathbb{P}[T < k(\ln k - c)] = e^{-e^{c}}$
- $\lim_{k\to\infty} \mathbb{P}[T > k(\ln k + c)] = 1 - e^{-e^{-c}}$
- $\lim_{k\to\infty} \mathbb{P}[k(\ln k - c) < T < k(\ln k + c)] = e^{-e^{c}} - e^{-e^{-c}}$

From Corollary 4.3.2 it follows that, with high probability, the number of trials for collecting all k coupon lies in a small interval centered in its expected value.

4.3.2 Allocation as a Coupon Collector's Problem

Since we consider the homogenous case with also $\beta_x = b$ for all $x \in \mathcal{X}$ and we suppose the all resources are always on (i.e., $\lambda_x = 1$ for all $x \in \mathcal{X}$), we have that the

probability to pick an atom (and allocate it) is equal for all atoms $(x,a) \in \mathcal{A}$. We can apply the estimation on the Coupon Collector's Problem in our case, considering the coupons as the atoms to be allocated and the number of types as the total number of atoms. This lead to the following proposition.

Proposition 4.3.3. *The allocation time in the uniform case is of order $|\mathcal{A}|\ln|\mathcal{A}|$ where $|\mathcal{A}|$ is the cardinality of the set of atoms to be allocated.*

Remark 4.3.4. *Notice that the same estimate holds also in the case where resources are on with a fixed probability equal for all the resources ($\lambda_x = l$ for all $x \in \mathcal{X}$) and in the case with reciprocity.*
In the first case the probability of allocating an atom depends on the probability of finding the resource on. If the agents have different reliabilities it is sufficient to take

$$\lambda_{min} = \min_{x\in\mathcal{X}} \lambda_x$$

to have the same probability for all the atoms.
In the latter case, it is sufficient to consider the worst case scenario taking the minimum probability of acceptance that rises in the case where user x has already allocated her entire amount of data α while agent y has not allocated anything. The probability of acceptance (3.2) - (3.4) reduce to:

- *(number of refusal)*
$$\rho_{yx} = \frac{\omega}{\alpha+1} \text{ for all } x,y \in \mathcal{X}$$
- *(mutual behavior)*
$$\rho_{yx} = \frac{1}{\alpha+1} \text{ for all } x,y \in \mathcal{X}$$
- *(source/resource)*
$$\rho_{yx} = (1-\delta)\frac{\alpha}{\beta}$$

Since in both the cases the probability is the same for all the resources, the Coupons Collector's Problem gives the same estimation.

4.4 Average allocation time

Since the previous estimation is an overestimation, in this section we focus on the average time of allocation. We begin the analysis with the study of the associated Markov chain. We consider the case where resources can be on with a certain probability and the case with reciprocity. To describe the process we reduce to a case with only two agents, then we generalize.

First, we state some definitions (taken from [1, 3]) useful for the estimations. Given a Markov chain, the *hitting time* for a state is the first time at which the state is visited. A state of a Markov chain is called an *absorbing state* if the probability of jumping in another state is equal zero or, in other words, the probability of remaining in the same state is equal one. The *absorbing time* is the expected number of steps before the chain is absorbed, that is, before it reaches an absorbing state.

The associated Markov chain in the case with two agents is described as follow. Let the states of the system be the couples of allocated data $(W_{xy}(t), W_{yx}(t))$, where x and y are the two units. Suppose that they have to allocate the quantities α_x and α_y respectively and that they have enough available space $\beta_x = \beta_y = b$ for the allocation. Notice that the underlying graph of the Markov chain is a grid whose absorption state is the state corresponding to the full allocation of both users (α_x, α_y).

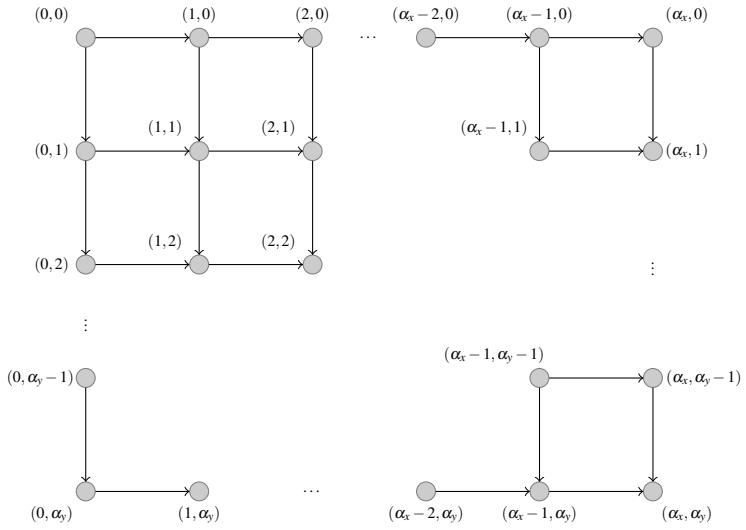


Fig. 4.1 Underlying graph of the associated Markov chain

Remark 4.4.1. *Notice that the possibile path, in the uniform case, from* $(0,0)$ *to* (α,α) *are bijective with the Grand-Dyck paths. This means that there are*

$$\binom{2\alpha}{\alpha} \approx \frac{4^{\alpha}}{\sqrt{\pi\alpha}}$$

possible ways to the complete allocation. Moreover, depending on the algorithm there could be self loops.

4.4.1 Average time in the network cloud storage model

The first case study is the case without reciprocity. In this case, the probability of allocation depends on the probability of finding the resource in functional state on (for simplicity we suppose it is the reliability itself), therefore the transition probabilities are, if $W_{xy} \neq \alpha_x, W_{yx} \neq \alpha_y$:

$$\Lambda_{(W_{xy},W_{yx}),(W'_{xy},W'_{yx})} = \begin{cases} \frac{\lambda_x}{2} & \text{if } W'_{xy} = W_{xy}, W'_{yx} = W_{yx}+1 \\ \frac{\lambda_y}{2} & \text{if } W'_{xy} = W_{xy}+1, W'_{yx} = W_{yx} \\ 1-\frac{\lambda_x+\lambda_y}{2} & \text{if } W'_{xy} = W_{xy}, W'_{yx} = W_{yx}. \end{cases}$$

On the other hand, if one of the two users has already allocated his/her entire amount of data, the transition probabilities are:

$$\Lambda_{(W_{xy},\alpha_y),(W'_{xy},\alpha_y)} = \begin{cases} \lambda_y & \text{if } W'_{xy} = W_{xy}+1 \\ 1-\lambda_y & \text{if } W'_{xy} = W_{xy}, \end{cases}$$

$$\Lambda_{(\alpha_x,W_{yx}),(\alpha_x,W'_{yx})} = \begin{cases} \lambda_x & \text{if } W_{yx} = W'_{yx}+1 \\ 1-\lambda_x & \text{if } W_{yx} = W'_{yx}, \end{cases}$$

The expected absorbing time can be obtained solving the following system of equations [3]:

$$\begin{cases} \mathbb{E}[\alpha_x, \alpha_y] = 0 \\ \mathbb{E}[W_{xy}, \alpha_y] = 1 + (1-\lambda_y)\mathbb{E}[W_{xy}, \alpha_y] + \lambda_y \mathbb{E}[W_{xy}+1, \alpha_y] & \text{if } W_{xy} \neq \alpha_x \\ \mathbb{E}[\alpha_x, W_{yx}] = 1 + (1-\lambda_x)\mathbb{E}[\alpha_x, W_{yx}] + \lambda_x \mathbb{E}[\alpha_x, W_{yx}+1] & \text{if } W_{yx} \neq \alpha_y \\ \mathbb{E}[W_{xy}, W_{yx}] = 1 + \dfrac{\lambda_x}{2}\mathbb{E}[W_{xy}, W_{yx}+1] + \dfrac{\lambda_y}{2}\mathbb{E}[W_{xy}+1, W_{yx}] \\ \qquad + \left(1 - \dfrac{\lambda_x+\lambda_y}{2}\right)\mathbb{E}[W_{xy}, W_{yx}] & \text{if } W_{xy} \neq \alpha_x, W_{yx} \neq \alpha_y \end{cases}$$

Notice that here, the absorbing time coincide with the allocation time, therefore we set τ_{abs} to indicate the average allocation (absorbing) time. Solving the system we have:

$$\tau_{abs} = \mathbb{E}[0,0] = \frac{\alpha_x\lambda_x + \alpha_y\lambda_y}{\lambda_x\lambda_y}. \tag{4.1}$$

Remark 4.4.2. *This approach holds also in the homogeneous case. Suppose that the users have a quantity $\alpha_x = \alpha_y = a$ to allocate, then:*

$$\tau_{abs} = \mathbb{E}[0,0] = a\left(\frac{\lambda_x+\lambda_y}{\lambda_x\lambda_y}\right).$$

The following theorem is a generalization of result (4.1) to the uniform case. Notice that in the case with $|\mathcal{X}| = n$ it is more appropriate to talk about hitting time since we are interested in te first time that the system reaches a full allocation state. On the other hand, we still use the notation τ_{abs} because whenever the system reaches a full allocation state, it can never exit form the set $\mathcal{W}$. Therefore, the allocation time is

$$\tau_{abs} = \min\{t \geq 0 : W \in \mathcal{W}\}$$

that is, the first time that one of the set of full allocation states is visited by the chain.

Theorem 4.4.3. *In the uniform case with n users it holds that*

$$\tau_{abs} = \frac{n\alpha}{\lambda}.$$

Proof. The states of the systems are the amounts of data allocated by each user $W^x = \sum_{y \in \mathcal{X}} W_{xy}$ and the transition probabilities are:

$$\Lambda_{(W^x),(W'^x)} = \begin{cases} \frac{\lambda}{k} & \text{if } W'^x = W^x + 1 \text{ and } W^x \neq \alpha \quad \forall x \in \mathcal{X} \\ 1 - \lambda & \text{if } W'^x = W^x, \end{cases}$$

where k is the number of possible allocation. The result follows as in (4.1). □

4.4.2 Average time of the algorithm with reciprocity

Now, we focus on the case with reciprocity. We limit our analysis to the *mutual behavior* probability of acceptance, that is,

$$\rho_{yx}(t) = \min \left\{ \frac{\lambda_x (W_{yx}(t) + 1)}{\lambda_y (W_{xy}(t) + 1)}, 1 \right\}.$$

If we consider the uniform case, then the probability of allocation is given by the probability of acceptance. Also in this case it is possible to write the transition probabilities of the associated Markov chain:

$$\Lambda_{(W_{xy},W_{yx}),(W'_{xy},W'_{yx})} = \begin{cases} \frac{1}{2} & \text{if } W_{xy} \leq W_{yx} \text{ and } W'_{xy} = W_{xy} + 1, W'_{yx} = W_{yx} \\ & \text{or } W_{xy} \geq W_{yx} \text{ and } W'_{xy} = W_{xy}, W'_{yx} = W_{yx} + 1 \\ \frac{1}{2} - \frac{W_{yx}+1}{2(W_{xy}+1)} & \text{if } W_{xy} > W_{yx} \text{ and } W'_{xy} = W_{xy}, W'_{yx} = W_{yx} \\ \frac{1}{2} - \frac{W_{xy}+1}{2(W_{yx}+1)} & \text{if } W_{xy} < W_{yx} \text{ and } W'_{xy} = W_{xy}, W'_{yx} = W_{yx} \\ \frac{W_{yx}+1}{2(W_{xy}+1)} & \text{if } W_{xy} > W_{yx} \text{ and } W'_{xy} = W_{xy} + 1, W'_{yx} = W_{yx} \\ \frac{W_{xy}+1}{2(W_{yx}+1)} & \text{if } W_{xy} < W_{yx} \text{ and } W'_{xy} = W_{xy}, W'_{yx} = W_{yx} + 1 \end{cases}$$

Again, the expected time of absorption can be obtained solving the system:

$$
\begin{cases}
\mathbb{E}[\alpha,\alpha] = 0 & \\
\mathbb{E}[W_{xy},\alpha] = \alpha - W_{xy} & \\
\mathbb{E}[\alpha,W_{yx}] = \alpha - W_{yx} & \\
\mathbb{E}[W_{xy},W_{yx}] = 1 + \frac{1}{2}\mathbb{E}[W_{xy}+1,W_{yx}] + \frac{1}{2}\mathbb{E}[W_{xy},W_{yx}+1] & \text{if } W_{xy} = W_{yx} \\
\mathbb{E}[W_{xy},W_{yx}] = 1 + \frac{W_{yx}+1}{2(W_{xy}+1)}\mathbb{E}[W_{xy}+1,W_{yx}] + \frac{1}{2}\mathbb{E}[W_{xy},W_{yx}+1] + & \\
\quad + \left(\frac{1}{2} - \frac{W_{yx}+1}{2(W_{xy}+1)}\right)\mathbb{E}[W_{xy},W_{yx}] & \text{if } W_{xy} > W_{yx} \\
\mathbb{E}[W_{xy},W_{yx}] = 1 + \frac{W_{xy}+1}{2(W_{yx}+1)}\mathbb{E}[W_{xy},W_{yx}+1] + \frac{1}{2}\mathbb{E}[W_{xy}+1,W_{yx}] + & \\
\quad + \left(\frac{1}{2} - \frac{W_{xy}+1}{2(W_{yx}+1)}\right)\mathbb{E}[W_{xy},W_{yx}] & \text{if } W_{xy} < W_{yx}
\end{cases}
$$

In this case, the analytical solution is not easy to obtain, since the nodes of the chain have different degrees, but in the simulation section a number of examples validate the approach.

An interesting thing about the chain associated to the algorithms is that it is possibile to compute the probability of a path. In particular we are interested in the best and the worst path from $(0,0)$ to (α,α) since we want the worst case to happen with a very low probability. Notice that the worst case is when the algorithm moves on the border of the grid while the best choice is moving on the diagonal (since the grid is symmetric we study only the upper-right path).

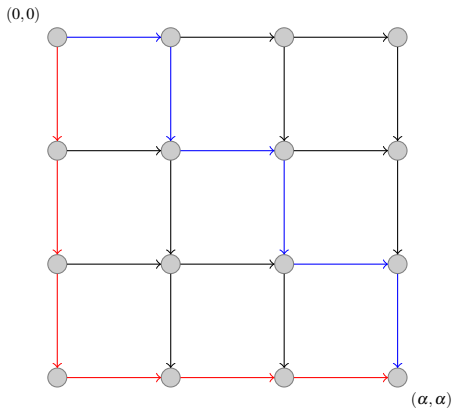


Fig. 4.2 Paths on the underlying graph of the Markov chain

Doing some computation we have that the worst case happen with probability

$$P_{border} = \frac{1}{2^{\alpha}(\alpha!)},$$

while the diagonal path is more likely to happen:

$$P_{diag} = \frac{1}{2^{2\alpha-1}}.$$

4.5 Simulations

In this section, we present a number of numerical simulations that validate our theoretical results and show the behavior of the proposed algorithm.

Below we report the basic assumptions of our simulations.

- Most of our simulations will be for $n = 2$ or $n = 50$ units.
- We consider two possible underlying networks, a complete graph and a regular graph with degree 10.
- We consider the case of homogeneous populations where all units have the same α_x, β_x and λ_x, $x \in \mathcal{X}$, as well heterogeneous cases with two subpopulations $\mathcal{X}_1$ and $\mathcal{X}_2$ sharing different parameters.
- In the utility function we set the congestion parameter to be $C^{con} = 1$ while we consider different values for the aggregation parameter C^{agg}. The allocation constant is fixed

  $$C^{all} = 3(||\alpha||_\infty |C^{agg}| + ||\beta||_\infty C^{con})$$

  according to Chapter 2.
- In the implementation of the algorithm, time is supposed to be discrete, assuming that at each instant of time a unit x wakes up with a probability proportional to α_x.

- The parameter γ in the Gibbs distribution is chosen to be time-varying with the law:
$$\gamma(t) = \gamma(t-1) + \frac{1}{n * 1000}$$
where n is the number of agents.

For all the examples, the performance of the algorithm will be analyzed considering the following parameters computed, in a Montecarlo style, by averaging over 10 running of the algorithm.

- **Time horizon**: for every example we note the upper bound given by the Coupon Collector's Problem to make a comparison with the average time of allocation. Therefore, the time horizon is given by Proposition 4.3.3, i.e.,
$$T = |\mathcal{A}| \log |\mathcal{A}|$$
where $|\mathcal{A}|$ is the total amount of atoms to be allocated.
- **Absorbing time**: τ_{abs} is the average absorbing (hitting) time computed solving the systems of Sections 4.4.1 and 4.4.2 or using Theorem 4.4.3. Indeed, in the case without reciprocity we compute τ_{abs} the the formula (4.1) while in the other case we use the Matlab software to solve the system.
- **Average time of allocation**: τ_{all} is computed counting the number of iteration needed to allocate all the data atoms and it is compared to the value estimated with the Markov chain absorbing time τ_{abs}.

The first example is for two users and it compare different settings. Later we consider more users and different topologies.

Example 4.5.1. *Consider to have two users, each of them having a quantity $\beta_x = \beta_y = 50$ of available space. We study the behavior of the algorithm without reciprocity varying the amount of data to allocate and the reliability of the resources. In particular we consider:*

1. $\alpha_x = \alpha_y = 45$ *and* $\lambda_x = \lambda_y = 0.8$,
2. $\alpha_x = \alpha_y = 45$ *and* $\lambda_x = 0.8$, $\lambda_y = 0.5$,

3. $\alpha_x = 45$, $\alpha_y = 40$ *and* $\lambda_x = \lambda_y = 0.8$,

4. $\alpha_x = 45$, $\alpha_y = 40$ *and* $\lambda_x = 0.8$, $\lambda_y = 0.5$.

Table 4.1 shows the parameters in the cases listed above. As one can see, the numbers of iterations needed is close to the esteemed absorbing time, confirming that the estimation is correct.

Table 4.1 Allocation time with varying reliability

	Case 1	Case 2	Case 3	Case 4
T	404.9829	404.9829	377.6254	377.6254
τ_{abs}	112.5000	146.2500	106.2500	140.0000
τ_{all}	110.000	142.6000	105.9000	138.1000

Example 4.5.2. *Consider to have two users, each of them having a quantity* $\alpha_x = \alpha_y = 45$ *to be allocated and suppose* $\beta_x = \beta_y = 50$ *and* $\lambda_x = \lambda_y = 1$. *We now show the time parameter for the algorithm with reciprocity. In this case we have:*

Table 4.2 Allocation time with reciprocity

T	404.9829
τ_{abs}	96.8000
τ_{all}	96.5000

Figure 4.3 shows how the time of allocation vary trough the samples. The blue line indicates τ_{abs}*, the dotted line is the average and the stars are the single allocation times.*

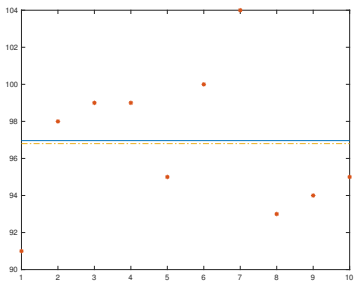


Fig. 4.3 Average allocation time and samples

The last example is for 50 users to evaluate the estimations also for higher numbers of agents.

Example 4.5.3. *Consider to have $n = 50$ users on a complete graph and on a regular graph with degree 10. Each of them has $\alpha_x = 45$ and $\beta_x = 50$. We choose $\lambda_x = 1$ and later $\lambda_x = 0.8$ for all $x \in \mathcal{X}$. The following tables shows the time parameters.*

Table 4.3 Performance parameters for $n = 50$

(a) $\lambda_x = 1$

	Complete	Regular
T	7815200	7815200
τ_{abs}	2250	2250
τ_{all}	2263.3000	2335.9000

(b) $\lambda_x = 0.8$

	Complete	Regular
T	7815200	7815200
τ_{abs}	2812.5000	2812.5000
τ_{all}	2821.7000	2816.4000

4.6 Conclusions

In this section we give an upper bound for the allocation time and we compute, in some particular cases, the average time needed to complete the allocation.

Even if a final section with simulations confirms the estimations, some more work in the estimation is needed. Indeed, a more tight bound would be of high interest because it would give a reasonable horizon for practical applications. On the other hand, the average allocation time is a good measure of the speed of the algorithm but it must be studied in more general frameworks.

References

[1] BAILEY, N. T. *The elements of stochastic processes with applications to the natural sciences*, vol. 25. John Wiley & Sons, 1990.

[2] ISAAC, R. The pleasures of probability. undergraduate texts in mathematics, 1995.

[3] LEVIN, D. A., PERES, Y., AND WILMER, E. L. *Markov chains and mixing times*. American Mathematical Soc., 2009.

[4] MOTWANI, R., AND RAGHAVAN, P. *Randomized algorithms*. Chapman & Hall/CRC, 2010.

Chapter 5

A diversified allocation algorithm

5.1 Introduction

In this chapter we focus on a more secure version of the algorithm. In particular we are interested in the possibility for the agents to recover the data. To this aim we allow the user to allocate more than one copy of their atoms. This diversification increase the probability to recover the stored data even though some resources are in functional state off and guarantee a higher safety in case of a breakdown. The optimum degree of diversification depend on the case and we suppose it is fixed a priori.

The game theoretic background is the same as in Chapter 2. The aim of the user is to maximize his/her payoff and at the same time to complete the allocation of all the copies of his/her backup. The game is potential and following standard optimization analysis [2] we show that the maximum of the potential function are indeed Nash Equilibria.

In the next section we describe the details of the problem, following Chapter 2 and expanding, where necessary, the definitions. Later we describe the allocation condition and study the equilibria of the game. Finally, we describe and analyze the allocation algorithm.

5.2 The allocation problem

Consider a set $\mathcal{X}$ of units which will play the double role of users who have to allocate a back up of their data, as well resources where data from other units can be allocated. Each unit $x \in \mathcal{X}$ is characterized by three parameters:

- α_x which is the amount of data she needs to allocate outside
- β_x which is the amount of memory available to allocate data from other units
- λ_x which is the reliability

Recall that the quantities $\{\alpha_x\}$ and $\{\beta_x\}$ are assembled in the vectors α and β respectively. The number of copies to be stored is indicated with σ, fixed a priori and equal for all the agents. We assume units to be connected through a directed graph $\mathcal{G} = (\mathcal{X}, \mathcal{E})$ where a link $(x,y) \in \mathcal{E}$ means that unit x is allowed to storage data in unit y. We denote by

$$N_x := \{y \in \mathcal{X} \,|\, (x,y) \in \mathcal{E}\}, \quad N_y^- := \{x \in \mathcal{X} \,|\, (x,y) \in \mathcal{E}\} \tag{5.1}$$

respectively, the out- and the in-neighborhood of a node. Goal of every unit x is to allocate σ distinct copies of its data into other units available storage with the constraint of putting different copies into distinct resources. Recall that $\mathcal{A}$ is the set of all the atoms to be allocated:

$$\mathcal{A}_x = \{(x,a) \,|\, a \in \{1,\dots,\alpha_x\}\}, \quad \mathcal{A} = \bigcup_{x \in \mathcal{X}} \mathcal{A}_x$$

and define

$$\mathcal{S}_x = \{(x,a,s) \,|\, a \in \{1,\dots,\alpha_x\}, s = 1,\dots,\sigma\}, \quad \mathcal{S} = \bigcup_{x \in \mathcal{X}} \mathcal{S}_x$$

the set of all the copies of all the atoms to be allocated. Analogously the set of all the atoms of available storage is indicated with

$$\mathcal{B}_y = \{(y,b) \,|\, b \in \{1,\dots,\beta_y\}\}, \quad \mathcal{B} = \bigcup_{y \in \mathcal{X}} \mathcal{B}_y.$$

Given the quadruple $(\mathcal{G}, \alpha, \beta, \sigma)$, we define an allocation as any map

$$Q : \mathcal{S} \to \mathcal{X}$$

satisfying the properties described below.

(QC1) **Graph constraint** If $(x,y) \notin \mathcal{E}$, then $Q(x,a,s) \neq y$ for every $(x,a,s) \in S$.

(QC2) **Storage limitation** For every $y \in \mathcal{X}$,

$$|Q^{-1}(y)| \leq \beta_y$$

(QC3) **Diversification** $Q(x,a,s) \neq Q(x,a,s')$ for any $s \neq s'$, any $x \in \mathcal{X}$ and any $(x,a,s) \in \mathcal{S}$.

The fact that $Q(x,a,s) = y$ means that user x has allocated the copy s of the data atom a into resource y. Clearly, if $\sigma = 1$ we have the original allocation problem as defined in Chapter 2. We will say that the allocation problem is solvable if an allocation Q exists. We denote by $\mathcal{Q}$ the set of allocations. We will also need to consider partial allocations, namely maps $Q : D \to \mathcal{X}$ where $D \subseteq \mathcal{S}$ satisfying, where defined, conditions (QC1) to (QC3). We denote by $\mathcal{Q}_p$ the set of partial allocations.

Since the diversification constraint is fundamental in this approach, we define

$$\mathcal{R}_{(x,a)} = \{y \in \mathcal{X} \,|\, \exists s : Q(x,a,s) = y\} \tag{5.2}$$

the set of resources where the atom (x,a) is allocated. Notice that $|\mathcal{R}_{(x,a)}| \leq \sigma$ for all $(x,a) \in \mathcal{A}$ and the cardinality of this set is exactly σ when all the copies of the atoms are allocated.

To each allocation Q we can associate a matrix W. More generally, given any partial allocation $Q \in \mathcal{Q}_p$, we put

$$W(Q)_{xy} = |Q^{-1}(y) \cap \mathcal{S}_x|. \tag{5.3}$$

$W(Q)_{xy} = W_{xy}$ is the total amount of data that x is allocating into y under the allocation Q; when not necessary, we avoid to indicate the dependency on Q.

The allocation problem can be reformulated in terms of the matrix W.

Proposition 5.2.1. *The allocation problem is solvable if and only if there exists a matrix $W \in \mathbb{R}^{\mathcal{X}\times\mathcal{X}}$ satisfying the following properties:*

(P1) $W_{xy} \geq 0$ *for all* x, y *and* $W_{xy} = 0$ *whenever* $(x,y) \notin \mathcal{E}$.

(P2) $W_{xy} \leq \alpha_x$ *for all* x, y.

(P3) $\sum_{y\in\mathcal{X}} W_{xy} = \sigma\alpha_x$ *for all* $x \in \mathcal{X}$.

(P4) $\sum_{x\in\mathcal{X}} W_{xy} \leq \beta_y$ *for all* $y \in \mathcal{X}$.

Proof. Only if: Suppose the allocation problem is solvable and that Q is an allocation. Consider $W = W(Q)$ as defined in (5.3). Property (P1) immediately follows from (QC1). Since two copies of the same atom cannot be allocated in the same resource, as a consequence of (QC3), it follows that

$$W_{xy} = |Q^{-1}(y) \cap \mathcal{S}_x| \leq \alpha_x$$

This proves (P2). Regarding (P3), we have that

$$\sum_{y\in\mathcal{X}} W_{xy} = \sum_{y\in\mathcal{X}} |Q^{-1}(y) \cap \mathcal{S}_x| = \sigma\alpha_x.$$

Finally, (P4) is a straightforward consequence of (QC2).

If: For a fixed unit $x \in \mathcal{X}$, order the resources in N_x as $y_1, \ldots, y_m$ and suppose, without loss of generality, that the atoms of S_x are numbered from 0 to $\sigma\alpha_x$ in groups of α_x. Define $Q_x : \mathcal{S}_x \to \mathcal{X}$ by putting $Q_x(a) = y_k$ if $a \in \left[\sum_{j=1}^{k-1} W_{xy_j}, \sum_{j=1}^{k} W_{xy_j}\right[$. Now define $Q : \mathcal{S} \to \mathcal{X}$ by putting $Q(x,a,s) = Q_x(a + (s-1)\alpha_x)$. By construction property (QC1) holds. (QC2) follows from

$$|Q^{-1}(y)| = \sum_{x\in\mathcal{X}} Q_x^{-1}(y) \leq \sum_{x\in\mathcal{X}} W_{xy} \leq \beta_y$$

where the last inequality follows from (P4). Finally, notice that $a + (s-1)\alpha_x$ and $a + (s'-1)\alpha_x$ differ of at least α_x. Since $W_{xy} \leq \alpha_x$, necessarily, $Q_x(a + (s-1)\alpha_x) \neq Q_x(A + (s'-1)\alpha_x)$. This yields (C3).

□

A matrix W satisfying properties (P1) to (P4) will be called a state of the system. The set of states is a simplex which will be denoted by $\mathcal{W}$. It will be useful to consider matrices W satisfying properties (P1), (P2), (P4), and the following relation

(P3') $\sum_{y\in\mathcal{X}} W_{xy} \leq \sigma\alpha_x$ for all $x \in \mathcal{X}$.

replacing (P3). They correspond to situations where not all data have been allocated and they are called partial states; with the symbol $\mathcal{W}_p$ we denote their family.

5.2.1 Study of the allocation conditions

An important aspect to study is the allocation condition, i.e., understanding which relation exists between the quantity to be allocated and the available storage.
The allocation condition proven in Chapter 2

$$\sum_{x\in D} \alpha_x \leq \sum_{y\in N(D)} \beta_y \text{ for all } D \subseteq \mathcal{X} \tag{5.4}$$

does not hold in its generalization

$$\sigma \sum_{x\in D} \alpha_x \leq \sum_{y\in N(D)} \beta_y \text{ for all } D \subseteq \mathcal{X}$$

in this case with multiple copies. The reason is the diversification constraint that forbid the allocation of different copies of the same atom in the same resource. Furthermore, due to the diversification constraint it is not possible to build an allocation that correspond to a matching therefore we cannot use the same argument of Theorem 2.1.1. To be more specific, consider a graph $\mathcal{P} = (\mathcal{S}\times\mathcal{B}, \mathcal{E}_P)$ where $((x,a,s),(y,b)) \in \mathcal{E}_P$ iff $(x,y) \in \mathcal{E}$. Given an allocation $Q \in \mathcal{Q}$, build a map $\tilde{Q} : \mathcal{S} \to \mathcal{B}$ such that $\tilde{Q}(x,a,s) = (Q(x,a,s),b)$ for all $x \in \mathcal{X}$ and for all atoms and copies a and s. The induced matching is given by

$$\mathcal{M} = \bigcup_{x\in\mathcal{X}} \{((x,a,s),(y,b) \in \mathcal{S}\times\mathcal{B} \,|\, (y,b) = \tilde{Q}(x,a,s)\}.$$

Unfortunately, given a matching, it is not possible to associate it to an allocation because it can happen that $\tilde{Q}(x,a,s) = (y,b)$ and $\tilde{Q}(x,a,s') = (y,b')$.

This is the reason why we can only formulate a conjecture on the allocation condition.

Conjecture 5.2.2. *Given $(\mathcal{G}, \alpha, \beta)$, there exists an allocation iff*

$$\sigma \sum_{x \in D} \alpha_x \leq \sum_{y \in N(D)} \min \left\{ \beta_y, \sum_{x \in N^-(y)} \alpha_x \right\} \quad \forall D \subseteq \mathcal{X}. \tag{5.5}$$

Remark 5.2.3. *Notice that this condition include $\sigma \leq |N^x|$ for all x which is a fundamental hypothesis for the allocation to guarantee the diversification.*

The following example shows a case where the generalization of condition (5.4) is not enough to guarantee the allocation while it satisfies condition (5.5).

Example 5.2.4. *Consider the following graph and for simplicity suppose that there are two users who are a sources of data (indicated with $U = \{1,2\}$) and five shared resources $R = \{3,4,5,6,7\}$. Suppose that $\alpha_1 = 2$ and $\alpha_2 = 1$ and fix $\sigma = 3$; suppose moreover that $\beta_i = 2$ for $i = 3,\ldots,7$. The underlying graph is in the following figure.*

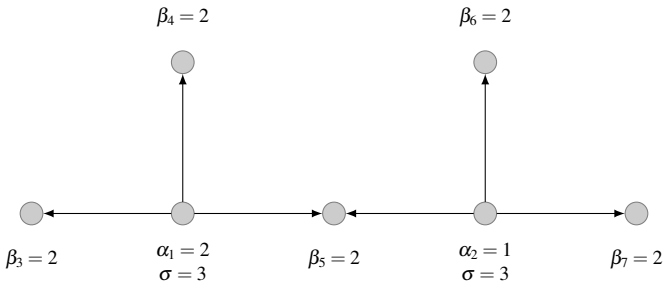


Fig. 5.1 Underlying graph of Example 5.2.4

Condition (5.4) holds:

$$\sigma \sum_{x \in U} \alpha_x = 9 \leq 10 = \sum_{y \in R} \beta_y$$
$$\sigma\alpha_1 = 6 = \sum_{y \in N(1)} \beta_y$$
$$\sigma\alpha_2 = 3 \leq 6 = \sum_{y \in N(2)} \beta_y$$

but one of the two users cannot complete the allocation. In fact, if $D = U$

$$\sigma \sum_{x \in U} \alpha_x = 9 \text{ and } \sum_{y \in N(D)} \min \left\{ \beta_y, \sum_{x \in N^-(y)} \alpha_x \right\} = 8.$$

The conjecture is motivated by the fact that, given condition (5.5), it is possible to find an allocation state in a particular case. The following example is explicative.

Example 5.2.5. *Consider to have a complete underlying graph and suppose* $\alpha_x = a$ *and* $\beta_x = b$ *for all* $x \in \mathcal{X}$*. In this case, the condition reduce to*

$$\sigma a \leq b \text{ and } \sigma \leq n-1 \tag{5.6}$$

To find an allocation state it is sufficient to take an $n \times n$ *circulant matrix on the vector*

$$[0, \underbrace{a, \ldots, a}_{\sigma}, \underbrace{0, \ldots, 0}_{n-\sigma-1}].$$

Notice that it is necessary to set the first entrance to be zero to guarantee that no users allocate into themselves. The matrix that we obtain guarantee the diversification since each user allocate exactly σ *copies and it also satisfy the storage limitation because there are exactly* σa *atoms in each resource. Clearly any permutation of this matrix that guarantees that* $W_{xx} = 0$ *for all* $x \in \mathcal{X}$ *is a possible allocation state.*

5.2.2 Feasible allocation states

Example 5.2.5 suggest that it is possible to find an allocation state at least in the complete case. Here, we show that there are other solutions for regular graphs. Consider an undirected regular graph $\mathcal{G} = (\mathcal{X}, \mathcal{E})$ with degree $s \geq \sigma$ and adjacency matrix A, and suppose $\alpha_x = a$ and $\beta_x = b$ for all $x \in \mathcal{X}$. The diffused allocation state defined in Chapter 2, if s divides a

$$W_{xy} = \frac{a}{s} A_{xy}$$

is still a desirable allocation state when the fragmentation of data is needed. On the other hand, the matching allocation state

$$W_{xy}^{\rho} = \begin{cases} \sigma a & \text{if } \rho(x) = y \\ 0 & \text{otherwise} \end{cases}$$

where $\rho : \mathcal{X} \to \mathcal{X}$ is a permutation without any fixed points, is not a possible state because of the diversification constraint. However, it seem reasonable to consider the following generalization

$$W_{xy}^{\rho} = \begin{cases} a & \text{if } \rho(x) = y_i,\ i = 1, \ldots, \sigma \\ 0 & \text{otherwise} \end{cases}$$

where $\rho : \mathcal{X} \to \mathcal{X}^{\sigma}$ assign to each user a σ-upla of resources. In this allocation state each user choose exactly σ resources and it is the most aggregate state we can obtain to ensure the diversification. The allocation state given in Example 5.2.5 is of this type.

5.3 The game theoretic structure

Given a (partial) allocation state $W \in \mathcal{W}_p$, we use the symbols $W_{x\cdot}$ and $W_{\cdot y}$ to indicate, respectively, the row x and the column y of the matrix W. The payoff for unit x in the state W is defined by a function $U_x : \mathcal{W}_p \to \mathbb{R}$ such that

$$U_x(W) = f_x(W_{\cdot x}) + \sum_{y \in N_x} g_y(W_{y\cdot})$$

following the description in Chapter 2. The first addend considers the amount of data that x has already allocated. The second addend instead takes into consideration the resource x is using and its congestion situation. Clearly, the fragmentation of the allocated data depends on the functionals. Possible utility functions are described in Section 2.2 of Chapter 2.

A Nash equilibrium is any allocation state $W \in \mathcal{W}_p$ such that for every $x \in \mathcal{X}$ and for every $W' \in \mathcal{W}_p$ such that $W_{xy} = W'_{xy}$ for every $x \neq \bar{x}$ and for every y, it holds

$$U_x(W) \geq U_x(W').$$

In words, in a Nash equilibrium, a unit x has no convenience to change its allocation moving data from one resource to another which is available. Following the analysis in [1] as in Chapter 2, we can write the functional

$$\Psi(W) = \sum_{x\in\mathcal{X}} f_x(W_{x\cdot}) + \sum_{y\in\mathcal{X}} g_y(W_{\cdot y})$$

that represent the discrete potential function of the game. The characteristics and properties of the potential and utility functions are widely described in Sections 2.1.2 and 2.2 of Chapter 2. Clearly the diversification do not affect the analysis and the description of the utility since it depends on the allocated quantities.

5.3.1 Existence of Nash Equilibria

The model above can be interpreted as a population game [2] where the population consists of atoms of different types as many as the units $\mathcal{X}$. To every unit $x \in \mathcal{X}$ we associate a continuum of atoms of type x consisting of σ distinct intervals of size α_x who have to split themselves into the available resources $y \in N_x$ satisfying the constraints (QC2) and (QC3). The population games considered in the literature do not present the constraints (P2) and (P4). As we will see this addiction is not entirely innocuous; in particular it makes some of the classical results [2] not automatically extendable to our context. We detail this below.

Define the function $\Psi : \mathcal{W} \to \mathbb{R}$:

$$\Psi(W) = \sum_{x\in\mathcal{X}} \int_0^{W_x} f_x(s)\mathrm{d}s + \sum_{y\in\mathcal{X}} \int_0^{W_y} g_y(s)\mathrm{d}s \tag{5.7}$$

and notice that

$$\frac{\partial\Psi}{\partial W_{xy}} = f_x(W_{x\cdot}) + g_y(W_{\cdot y}) \tag{5.8}$$

It follows that the game we are considering is a potential game. In classical population game theory, Nash equilibria coincide with the Kuhn-Tucker points of Ψ with respect to the constraints (P1) to (P4), namely state $W \in \mathcal{W}$ such that there exist coefficients (Kuhn-Tucker weights) μ'_{xy}, μ''_{xy} for $x,y \in \mathcal{E}$ with $(x,y) \in \mathcal{X}$, μ'_x, μ''_x for $x \in \mathcal{X}$

satisfying the relations

$$\begin{aligned} &\mu'_{xy} \geq 0, \quad \mu'_{xy} W_{xy} = 0 \\ &\mu''_{xy} \geq 0, \quad \mu''_{xy}(W_{xy} - \alpha_x) = 0 \\ &\mu''_{y} \geq 0, \quad \mu''_{y}(\textstyle\sum_x (W_{xy} - \beta_y)) = 0 \end{aligned} \tag{5.9}$$

(for all $x, y \in \mathcal{X}$ such that $(x,y) \in \mathcal{E}$ and for $y \in \mathcal{X}$) and such that the auxiliary function

$$\tilde{\Psi} = \Psi + \sum_{x,y} \mu'_{xy} W_{xy} - \sum_{x,y} \mu''_{xy}(W_{xy} - \alpha_x) - \sum_x \mu'_x (\sum_y W_{xy} - \sigma\alpha_x) - \sum_y \mu''_y (\sum_x (W_{xy} - \beta_y)) \tag{5.10}$$

is stationary in W, namely $\nabla\tilde{\Psi}(W) = 0$.

Proposition 5.3.1. *Kuhn-Tucker points for Ψ are Nash equilibria for the game. In particular, local maxima of Ψ are Nash equilibria.*

Proof. Suppose W is a Kuhn-Tucker point with corresponding weights μ'_{xy}, μ''_{xy}, μ'_x, μ''_x. Stationarity condition yields

$$U_x(W) = -\sum_y \mu'_{xy} + \sum_y \mu''_{xy} + \mu'_x + \sum_y \mu''_y \tag{5.11}$$

for all x, y such that $(x,y) \in \mathcal{E}$. To prove that W is Nash, consider a pair x, y_1 such that $W_{xy_1} > 0$. Consider than another state W' such that $W'_{y_2} < \beta_{y_2}$ and $W'_{xy_2} < \alpha_x$ for some y_2 in the neighborhood of x. From (5.11) using the conditions (5.9), we obtain

$$U_x(W) = -\sum_{y \neq y_1} \mu'_{xy} + \sum_y \mu''_{xy} + \mu'_x + \sum_y \mu''_y$$

$$U_x(W') = -\sum_y \mu'_{xy} + \sum_{y \neq y_2} \mu''_{xy} + \mu'_x + \sum_{y \neq y_2} \mu''_y$$

Since μ''_{xy_2}, μ''_{y_1}, μ'_{xy_1} are non-negative terms, this immediately implies that $U_x(W) \geq U_x(W')$. Thus this shows that W is Nash equilibria. Last assertion simply follows from the fact that local maxima are Kuhn-Tucker points. □

In the special case when $f_x(W_x) = 0$, our population game is a congestion game. Notice that in this case the payoff $U_x(W)$ does not explicitly depend on x and its

dependence on W is exclusively through W_y. Also the potential function Ψ is only function of the variables W_y as y varies in $\mathcal{X}$. If we define $\phi : \prod_{y\in\mathcal{X}} [0,\beta_y] \to \mathbb{R}$ by

$$\phi(v) = \sum_{y\in\mathcal{X}} \int_0^{v_y} g_y(s)\mathrm{d}s \tag{5.12}$$

and we put $v : \mathcal{W} \to \mathbb{R}$ by $v(W)_y = W_y$, we have that $\Psi(W) = \phi(v(W))$. Now, if we suppose that $g_y(s)$ is a strict increasing C^1 function, we have that ϕ is strictly convex so it possesses just one local maximum v^0 which is also global. Being $v(W)$ linear, we thus have that Ψ is convex on $\mathcal{W}$. However, in general the convexity is not strict since the function $v(W)$ is many-to-one. To the global maximum v^0 will thus correspond, in general, an affine set of global maxima of Ψ. Notice that if $W \in \mathcal{W}$ is any global maximum of Ψ, then, $W_y = v_y^0$ for every resource y. This shows that all these Nash equilibria are characterized by giving the same congestion levels on the various resources; in other terms the only thing that can possibly change is the way the units split their atoms among the resources. Of course convexity implies that no other Kuhn-Tucker point for Ψ may exist. Even in this case, however, the set of Nash equilibria of the underlying game is in general larger than the set of global maxima. Clearly, in the case when also the aggregation term is present, also local maxima and other stationary points can show up in the potential function.

5.4 The allocation algorithm

The allocation algorithm we are proposing is fully distributed and asynchronous and is only based on communications between units, taking place along the links of the graph $\mathcal{G} = (\mathcal{X}, \mathcal{E})$. It is based on the ideas of learning dynamics where, randomly, units activate and modify their action (allocation state) in order to increase their utility.

We now illustrate the details of our algorithm following the reasoning of Chapter 2. The functional state of the network at a certain time will be denoted by $\xi \in \{0,1\}^{\mathcal{X}}$: $\xi_x = 1$ means that the unit x is on. The times when units modify their functional state (off to on or on to off) and the times when units in functional state on activate are modeled as a family of independent Poisson clocks whose rates will be denoted (for unit x), respectively, ν_x^{on}, ν_x^{off}, and ν_x^{act}. The functional state of the network as

a function of time $\xi(t)$ is thus a continuous time Markov process whose components are independent Bernoulli processes.

We now describe the core of the algorithm, namely the rules under which activated units can modify their allocation state.

We start with some notation. Given a (possibly partial) allocation state $W \in \mathcal{W}_p$, a functional state $\xi \in \{0,1\}^{\mathcal{X}}$, and an atom $(\bar{x},\bar{a})$ such that $\xi_{\bar{x}} = 1$, define:

$$\mathcal{W}_{(\bar{x},\bar{a})}(W,\xi) = \left\{ W' \in \mathcal{W}_p : \begin{array}{l} W'_{xy} = W_{xy}, x \neq \bar{x} \text{ or } \xi_y = 0 \text{ or } y \in \mathcal{R}_{(\bar{x},\bar{a})} \\ W'^{\bar{x}} \geq W^{\bar{x}}, W' \neq W \end{array} \right\} \tag{5.13}$$

$$\mathcal{W}_{\bar{x}}(W,\xi) = \bigcup_{\bar{a} \in \mathcal{A}_x} \mathcal{W}_{(\bar{x},\bar{a})}(W,\xi) \tag{5.14}$$

$\mathcal{W}_{\bar{x}}(W,\xi)$ describes the possible partial allocation states obtainable from W by modifications done by the unit $\bar{x}$. Only the terms $W_{\bar{x}y}$ where y is on and free from repetition of copies can be modified; the total amount of allocated data $W^{\bar{x}}$ can only increase or remain equal. Since the sets $\mathcal{W}_{\bar{x}}(W,\xi)$ can in general be very large, it is convenient to consider the possibility that the algorithm might use a smaller set of actions where units either allocate new data or simply move data from one resource to another one.

Given $(W,\xi) \in \mathcal{W}_p \times \{0,1\}^{\mathcal{X}}$ and an atom $(\bar{x},\bar{a})$, define

$$N_{(\bar{x},\bar{a})}(W,\xi) = \left\{ y \in N_x : W_y < \beta_y, \xi_y = 1, y \notin \mathcal{R}_{(\bar{x},\bar{a})} \right\} \tag{5.15}$$

and

$$N_{\bar{x}}(W,\xi) = \bigcup_{a \in \mathcal{A}_x} N_{(\bar{x},\bar{a})}(W,\xi) \tag{5.16}$$

the set of available neighbor resources for $\bar{x}$ under the allocation state W and the functional state ξ: those that are on and still have space available and guarantee the diversification.

A family of sets $\mathcal{M}_{\bar{x}}(W,\xi) \subseteq \mathcal{W}_{\bar{x}}(W,\xi)$, defined for each $\bar{x} \in \mathcal{X}$ and each $(W,\xi) \in \mathcal{W}_p \times \{0,1\}^{\mathcal{X}}$, is called *admissible* if

(i) $W^{\bar{x}} < \sigma\alpha_{\bar{x}}$, $y \in N_{(\bar{x},\bar{a})}(W,\xi), \Rightarrow W' = W + e_{\bar{x}y} \in \mathcal{M}_{\bar{x}}(W,\xi)$;

(ii) $W_{\bar{x}y'} > 0$, $\xi_{y'} = 1$, $y \in \mathcal{R}_{(x,a)}$, $y'' \in N_{(\bar{x},\bar{a})}(W,\xi), \Rightarrow W' = W + \sum_{i=1}^{\sigma} (e_{\bar{x}y''} - e_{\bar{x}y'}) \in \mathcal{M}_{\bar{x}}(W,\xi)$;

(iii) $W' \in \mathcal{M}_{\bar{x}}(W,\xi)$ iff $W \in \mathcal{M}_{\bar{x}}(W',\xi)$ for every $W \in \mathcal{W}$.

Conditions (i) and (ii) essentially asserts that when a unit has an available neighbor resource not yet saturated, then $\mathcal{M}_{\bar{x}}(W,\xi)$ must incorporate the possibility to newly allocate or transfer already allocated data into it; this hold for all the copies of the atom. Condition (iii) instead simply says that when the functional state does not change and we are in an allocation state, any transformation can be reversed.
Examples of admissible families $\mathcal{M}_{\bar{x}}(W,\xi)$ are the following

1. $\mathcal{M}_{\bar{x}}(W,\xi) = \mathcal{W}_{\bar{x}}(W,\xi)$
2. $\mathcal{M}_{\bar{x}}(W,\xi) = \{W' \in \mathcal{W}_{\bar{x}}(W,\xi) : \exists y, \exists n\ W' = W + ne_{\bar{x}y})\} \cup \{W' \in \mathcal{W}_{\bar{x}}(W,\xi) : \exists y', y'', \exists n\ W' = W + n(e_{\bar{x}y''} - e_{\bar{x}y'})\}$
3. $\mathcal{M}_{\bar{x}}(W,\xi) = \{W' \in \mathcal{W}_{\bar{x}}(W,\xi) : \exists y_i, i = 1,\dots,\sigma\ W' = W + \sum_{i=1}^{\sigma} e_{xy_i}\} \cup \{W' \in \mathcal{W}_{\bar{x}}(W,\xi) : \exists y'_i, \exists y''_i, i = 1,\dots,\sigma\ W' = W + \sum_{i=1}^{\sigma}(e_{xy''_i} - e_{xy'_i})\}$

In the second case, modifications allowed are those where a unit either allocate a certain amount of new data into a single resource or it moves data from one resource to another one. The third case allow the user to allocate or move all the copies of an atom at the same time.

Given an admissible family $\mathcal{M}_{\bar{x}}(W,\xi)$, we now define a *Gibbs measure* on it as follows. Given a parameter $\gamma > 0$, put

$$\begin{aligned} Z_{\bar{x}}^{(W,\xi)}(\gamma) &= \textstyle\sum_{\tilde{W} \in \mathcal{M}_{\bar{x}}(W,\xi)} e^{\gamma U_{\bar{x}}(\tilde{W})} \\ Z_{\bar{x}}^{(W,W',\xi)}(\gamma) &= \max\left\{ Z_{\bar{x}}^{(W,\xi)}(\gamma), Z_{\bar{x}}^{(W',\xi)}(\gamma) \right\} \end{aligned} \tag{5.17}$$

Now define, for $W' \in \mathcal{M}_{\bar{x}}(W,\xi)$,

$$P_{\bar{x}}^{(W,\xi)}(W') = \begin{cases} \frac{e^{\gamma U_{\bar{x}}(W')}}{Z_{\bar{x}}^{(W,\xi)}(\gamma)}, & \text{if } ||W|| < ||W'|| \\ \frac{e^{\gamma U_{\bar{x}}(W')}}{Z_{\bar{x}}^{(W,W',\xi)}(\gamma)}, & \text{if } ||W|| = ||W'|| \end{cases} \tag{5.18}$$

where $||W|| = \sum_{xy} W_{xy}$, and complete it to a probability by putting

$$P_{\bar{x}}^{(W,\xi)}(W) = 1 - \sum_{W' \in \mathcal{M}_{\bar{x}}(W,\xi)} P_{\bar{x}}^{(W,\xi)}(W')$$

The algorithm is completely determined by the choice of the admissible family $\mathcal{M}_{\bar{x}}(W,\xi)$ and of the probabilities (5.18). If unit $\bar{x}$ activates at time t, the systems is in partial allocation state $W(t)$, and in functional state $\xi(t)$, it will jump to the new partial allocation state W' with probabilities given by

$$P(W(t+) = W') = P_{\bar{x}}^{(W(t),\xi(t))}(W'),\ W' \in \mathcal{M}_{\bar{x}}(W(t),\xi(t)) \tag{5.19}$$

If unit $\bar{x}$ chooses a W' such that $||W'|| > ||W||$ we say that it makes an *allocation move*, otherwise, if $||W'|| = ||W||$, we talk of a *distribution move*.

5.4.1 Analysis of the algorithm

The nice convergence properties of the original algorithm cannot be obtained in this model with the same reasoning of Chapter 2. The impossibility to prove those results depends on the diversification constraint that invalidate Lemma 2.3.1.

Now, assume we have fixed a quadruple $(\mathcal{G},\alpha,\beta,\sigma)$ satisfying the existence condition (2.2), an admissible family of sets $\mathcal{M}_{\bar{x}}(W,\xi)$ and we consider the allocation process $W(t)$ described by (2.19) with any possible initial condition $W(0)$.

By the way it has been defined, the process $W(t)$ is Markovian conditioned to the functional state process $\xi(t)$. If we consider the augmented process $(W(t),\xi(t))$, this is Markovian and its only non zero transition rates are described below:

$$\begin{aligned}
\Lambda_{(W,\xi),(W,\xi')} &= \begin{cases} \nu_{\bar{x}}^{on} & \text{if } \xi_{\bar{x}} = 0,\ \xi'_{\bar{x}} = 1,\ \xi_x = \xi'_x \forall x \neq \bar{x} \\ \nu_{\bar{x}}^{off} & \text{if } \xi_{\bar{x}} = 1,\ \xi'_{\bar{x}} = 0,\ \xi_x = \xi'_x \forall x \neq \bar{x} \end{cases} \\
\Lambda_{(W,\xi),(W',\xi)} &= \nu_{\bar{x}}^{act} P_{\bar{x}}^{(W,\xi)}(W') \quad \text{if } \xi_{\bar{x}} = 1,\ W' \in N_{\bar{x}}(W,\xi)
\end{aligned} \tag{5.20}$$

We now introduce a graph on $\mathcal{W}_p$ that will be denoted by $\mathcal{L}_p$: an edge (W,W') is present in $\mathcal{L}_p$ if and only if $W' \in \mathcal{M}_{\bar{x}}(W,\mathbb{1})$. Notice that, if $\nu_{\bar{x}}^{act} > 0$ for every $\bar{x}$, this can be equivalently described as $\Lambda_{(W,\mathbb{1}),(W',\mathbb{1})} > 0$. The graph $\mathcal{L}_p$ thus describes the possible jumps of the process $W(t)$ conditioned to the fact that all resources are

in functional state on. We want to stress the fact that the graph $\mathcal{L}_p$ depends on the triple $(\mathcal{G}, \alpha, \beta)$ as well on the choice of the admissible family $\mathcal{M}_{\bar{x}}(W, \mathbb{1})$ but not on the particular choice of the functional Ψ or of the utility functions $U_{\bar{x}}$.

Our strategy, in order to prove our first claim, will be to show that from any element $W \in \mathcal{W}_p$ there is a path in $\mathcal{L}_p$ to some element $W' \in \mathcal{W}$.

Given $W \in \mathcal{W}_p$ we define the following subsets of units

$$\mathcal{X}^f(W) := \{x \in \mathcal{X} \mid W^x = \sigma\alpha_x\},$$

$$\mathcal{X}^{sat}(W) := \{x \in \mathcal{X} \setminus \mathcal{X}^f(W) \mid \nexists y \in N_x \text{ s.t } W_y < \beta_y\}$$

Units in $\mathcal{X}^f(W)$ are called *fully allocated*: these units have completed the allocation of their data under the state W. Units in $\mathcal{X}^{sat}(W)$ are called *saturated*: they have not yet completed their allocation, however, under the current state W, they can not make any action, neither allocate, nor distribute.

Finally, define

$$\mathcal{W}_p^{sat} := \{W \in \mathcal{W}_p \setminus \mathcal{W} \mid \mathcal{X} = \mathcal{X}^f(W) \cup \mathcal{X}^{sat}(W)\}$$

It is clear that from any $W \in \mathcal{W}_p \setminus \mathcal{W}_p^{sat}$, there exists units that can make either an allocation or a distribution move. Instead, if we are in a state $W \in \mathcal{W}_p^{sat}$, there are units that are not fully allocated and all these units con not make any move. The only units that can possibly make a move are the fully allocated ones. Notice that, because of condition (2.2), for sure there exist resources y such that $W_y < \beta_y$ and these resources are indeed exclusively connected to fully allocated units. The key point is to show that in a finite number of distribution moves, performed by fully allocated units, it is always possible to move some data atoms from resources connected to saturated units to resources with available space: this will then make possible a further allocation move.

For any fixed $W \in \mathcal{W}_p$, we can consider the following graph structure on $\mathcal{X}$ thought as set of resources: $\mathcal{H}_W = (\mathcal{A}, \mathcal{E}_W)$. Given $y_1, y_2 \in \mathcal{X}$, there is an edge from y_1 to y_2 if and only if there exists $(x,a) \in \mathcal{A}$ for which

$$y_1 \in \mathcal{R}_{(x,a)}, \quad y_2 \in N_x \setminus \mathcal{R}_{(x,a)}.$$

The edge from y_1 to y_2 will be indicated with the symbol $y_1 \rightarrow_{(x,a)} y_2$ (to also recall the atom (x,a) involved). The presence of the edge means that the two resources y_1 and y_2 are in the neighborhood of a common unit x that is using y_1 under W. This indicates that x can in principle move some copies of its data currently stored in y_1 into resource y_2 if this last one is available.

As one can see, most of the feature used in Chapter 2 can be defined in this case but we cannot riformulate Lemma 2.3.1. What is not clear here is that even though there are available resources with $W_y < \beta_y$, there is no guarantee that such resources satisfy the diversification constraint. This problem is surely related to the absence of an allocation condition that was instead given and used in the original case and that is here only conjectured. Notice now that the existence of a sequence as in Lemma 2.3.1 is fundamental to guarantee the allocation and distribution moves therefore the convergence cannot be proven. On the other hand, in Chapter 6, some simulations validate the guess that the algorithm reaches a Nash Equilibrium also in this case of multiple copies.

5.5 Conclusions

In this chapter, we present the extension of our allocation algorithm to a case with multiple copies of the back up data. The interest on this problem comes from the realistic assumption that users may want to allocate more copies of the backup data to increase their security. Therefore, we describe the generalized model including a new parameter σ indicating the number of copies to be stored for each data. For this new model we conjectured an allocation condition motivated by Example 5.2.4 and prove the existence of Nash equilibra.

Future analysis should include, firstly, the proof that the allocation condition is the one proposed in Conjecture 5.2.2 and to give a proper characterization of the Nash equilibria. Moreover, the convergence properties are not easy to prove because of the diversification constraint. Therefore a deep analysis in this direction is not only needed but also fundamental to make our algorithm more interesting, general and valuable.

References

[1] MARDEN, J. R., AND WIERMAN, A. Distributed welfare games. *Operations Research 61*, 1 (2013), 155–168.

[2] SANDHOLM, W. H. *Population games and evolutionary dynamics*. MIT press, 2010.

Chapter 6

Simulations

6.1 Introduction

In this section we present a wide set of simulations that validate the theoretical analysis of all the proposed variations. We analyze the structure of the asymptotic Nash configuration reached and the effect of the variations of the parameters in the utility function. The algorithms are tested for their convergence properties and evaluated through some performance indices measuring the computational and topological complexity of the solution found. We study some particular cases and show many plots. Moreover, we compute a number of parameters to measure the performance as, for instance, the satisfaction level of the agents, the average in and out degree and the symmetry of the final state of the system; nonetheless, we compute also the distance from the optimum, the number of moves per each atom and the distance from the full allocation.

We aim to show that the algorithm is feasible for practical implementation and that it has good performance and scaling properties. Example presented are admittedly simple: our goal here is not to work out codes with optimized performance, neither to present exhaustive sets of simulations.

In the next section we describe the parameters that we need in our evaluation and we explain the basic assumption made for the following simulations. The remaining sections follow the structure of this thesis, starting from the original algorithm, through the variant with reciprocity to conclude with the multiple copies model.

6.1.1 Preliminaries

In this section we describe in details the calculated parameters that measure the performance. We present all the parameter at once but depending on the case we will show only the ones needed.

Since the agents activates as a family of independent Poisson clocks, we will assume that $\nu_x = \nu\alpha_x$ for some $\nu > 0$, namely that units activation rates is proportional to the amount of data they need to back up.

The number of units is denoted by n and assumed to be even. Most of our simulations are for $n = 50$ but we have also studied scalability issues by considering higher numbers of agents. We have considered two possible interconnection topologies: the complete graph and a random regular graph of degree 10 randomly constructed according to the classical configuration model.

For all examples, the performance of the algorithm will be analyzed considering the following parameters computed, in a Montecarlo style, by averaging over 10 running of the algorithm.

- **Allocation complexity:** Moving data from one resource to another one is an expensive task which must be carefully monitored in real applications. To this aim, we have introduced the index ν_{moves} which computes the number of allocation or distribution moves per piece of data throughout the dynamics. In formula, if m_i is the total number of moves performed by agent i during the run of the algorithm, we put

$$\nu_{moves} = \frac{1}{n}\sum_{x\in\mathcal{X}}\frac{m_i}{\alpha_i}.$$

- **Level of satisfaction:** In some cases we will divide the resource in two subsets $\mathcal{X}_1$ and $\mathcal{X}_2$ depending on their reliability (we suppose that one of the two subset is more trustworthy). We define the mean and the variance of the satisfaction level as

$$\Lambda_x = \frac{\sum_y \frac{W^\infty_{xy}}{\alpha_x}\lambda_y}{\lambda_x}$$

$$\bar{\Lambda} := \frac{1}{n}\sum_{x\in\mathcal{X}}\sum_{y\in N_x}\frac{W^\infty_{xy}}{\alpha_x}\lambda_y,$$

$$\Lambda_{var} := \frac{1}{n} \sum_{x \in \mathcal{X}} \left(\sum_{y \in \mathcal{X}} \frac{W_{xy}^{\infty}}{\alpha_x} \lambda_y - \bar{\Lambda} \right)^2$$

If λ_1 and λ_2 are taken to be the probability that if contacted at a random time the resource is available to give access to the stored data, $\bar{\Lambda}$ can be interpreted as the probability that a piece of data can be recovered when requested at some random time.

- **Number of unsatisfied users:** let

$$n_{unsat} = \frac{|x \in \mathcal{X} : \Lambda_x < 1|}{|\mathcal{X}|}$$

be the fraction of unsatisfied user with respect to the mutual satisfaction level. This parameter is interesting especially in the heterogeneous case to understand the behavior of the algorithm.

- **Average congestion:** The presence of the congestion term in the utility function should insure that all resources with the same reliability should in principle be used equally. We measure the mean and variance of the congestion level of resources in $\mathcal{X}_i$ by

$$\bar{C}^i := \frac{1}{n\beta} \sum_{y \in \mathcal{X}_i} W_y^{\infty},$$

$$C_{var} := \frac{1}{n} \sum_{y \in \mathcal{X}_i} \left(\beta^{-1} W_y^{\infty} - \bar{C} \right)^2$$

- **Topological Complexity:** We consider the in and out mean degrees measuring the topological complexity of the subgraph consisting of the edges (x,y) for which $W_{xy}^{\infty} > 0$:

$$d^+ := \frac{1}{n} \sum_{x \in \mathcal{X}} \sum_{y \in \mathcal{X}} \mathbb{1}_{\{W_{xy} > 0\}},$$

$$d_i^- := \frac{2}{n} \sum_{x \in \mathcal{X}} \sum_{y \in \mathcal{X}_i} \mathbb{1}_{\{W_{xy} > 0\}},\ i = 1,2$$

- **Distance from optimum:** $\psi = \frac{\Psi(W^T)}{\Psi_{opt}}$ where Ψ_{opt} is the maximum of the potential. This allows to measure how the final allocation state $W(T)$ is far from a Nash configuration maximizing the potential Ψ. We will use the exact value of Ψ_{opt} in those cases for which the optimum Nash is explicitly known.

In some cases we will also present a plot of the function $\psi(t) = \frac{\Psi(W(t))}{\Psi_{opt}}$, in this case for a single running of the algorithm.

- **Symmetry:** we study how far is the final system from being a symmetric matrix computing the parameter

$$\zeta = \left\| W - W^T \right\|_\infty$$

This parameter is meaningful in the case with reciprocity because it indicates the mutual behavior between users.

- **Distance from Full Allocation:**

$$\Delta = \sum_{x \in \mathcal{X}} \alpha_x - \sum_{x,y \in \mathcal{X}} W_{xy}$$

counts the quantity of atoms not yet allocated. If the allocation is complete, this parameter is 0.

We assume the admissible family $\mathcal{M}_{\bar{x}}(W, \xi)$ to be of type

$$\begin{aligned} \mathcal{M}_{\bar{x}}(W, \xi) = \quad & \{W' \in \mathcal{W}_{\bar{x}}(W, \xi) : \exists y, \exists n \in Q \; W' = W + n e_{\bar{x}y})\} \cup \\ & \{W' \in \mathcal{W}_{\bar{x}}(W, \xi) : \exists y', y'', \exists n \in Q \; W' = W + n(e_{\bar{x}y''} - e_{\bar{x}y'})\} \end{aligned}$$

where $Q \subseteq N$ and $1 \in Q$, i.e., the modifications allowed are those where a unit either allocate or move a number of data constrained in a set Q. Most of the examples are for $Q = \{1\}$: just one data is allocated or moved each time.

On the basis of our theoretical analysis, the algorithm, in the limit when $t \to +\infty$ and the inverse noisy parameter $\gamma \to +\infty$, is known to converge to the optimum. In practical implementations, a typical choice in these cases is to take the parameter γ, time-varying and diverging to $+\infty$. The tuning of the divergence rate is known to be critical to obtain good results. Here we have chosen

$$\gamma(t+1) = \gamma(t) + \frac{1}{n * 1000}$$

Most of the time, we suppose the units to be always on (otherwise things get simply slowed down). The time horizon is fixed $T = 5 * \sum_{x \in \mathcal{X}} \alpha_x$: in this way a unit x will activate, on average, a number of times equal to 5 times the number of data it needs

to allocate. As we will see, this time range is sufficient for the allocation to be completed and to get close to the optimum (this has been checked in those cases when the optimum is analytically known).

6.2 The network cloud storage model

In this section, we present some numerical results to validate the theoretical approach discussed in Chapter 2.

All the examples in this section are for the case when the functional has the form

$$U_x(W) = C^{all} \sum_{y \in \mathcal{X}} W_{xy} + C^{agg} \sum_{y \in \mathcal{X}} W_{xy}^2 - \sum_{y \in N_x^-} C_y^{con} (W_y)^2, \tag{6.1}$$

but the last one where instead we consider the form

$$U_x(W) = C^{all} \sum_{y} W_{xy} + C^{agg} \sum_{y \in \mathcal{X}} W_{xy}^2 - \sum_{y \in N_x^-} C_y^{con} |W_{\cdot y}|_H. \tag{6.2}$$

We always take

$$C^{all} = 3(||\alpha||_\infty |C^{agg}| + ||\beta||_\infty C^{con}). \tag{6.3}$$

This choice is motivated by the considerations in Chapter 2. We assume that $C_y^{con} = C^{con}$ for all units and we consider both the case when the aggregation parameter C^{agg} is positive or negative.

We now present a number of simulations for the case when the functional has the form defined in (6.1). We first consider the case when $Q = \{1\}$: just one data is allocated or moved each time a unit activates. We always take $C^{con} = 1$ and C^{all} chosen according to (6.3) and different values for C^{agg}. The first example is the same presented in Chapter 2.

Example 6.2.1. *Consider to have $n = 10$ users on a complete graph such that $\alpha_x = a = 45$ and $\beta_x = b = 50$ for every unit x. We consider the cases: $C^{agg} = -7, -1, 1/2, 3$. We already showed the final state of the system reached by the algorithm in Example 2.3.9. We plot in Figure 6.1 the time evolution of the potentials and we confront it with the optimal potential represented by the red line. For*

$C^{agg} = 3$ a matching allocation state is reached and it is a maximum of Ψ in this case. For $C^{agg} = -7$ the solution is also very close to the maximum that is the diffused allocation state. For $C^{agg} = 1/2, -1$, the presence of Nash equilibria that are not maxima of Ψ slows down the dynamics and the algorithm does not reach the maximum at time T (this particularly evident for the case $C^{agg} = 1/2$). Increasing in this case the time horizon to $T = 20 * \sum \alpha$, the final state of the system gets quite close to the maximum as confirmed by the plots in Figure 6.2. The following table shows the performance parameters in the Montecarlo simulation for the usual T.

Table 6.1 Performance parameters for $n = 10$

	$C^{agg} = -7$	$C^{agg} = -1$	$C^{agg} = 1/2$	$C^{agg} = 3$
d	9	8.7400	6.6200	1
ψ	1	0.9944	0.9156	1
Δ	0	0	0	0
ν_{moves}	3.1669	4.9389	4.9331	3.2449

From now on we focus on the cases $C^{agg} = -7, 3$, $C^{con} = 1$ and C^{all} chosen according to (6.3), showing that reasonably good properties are maintained for larger communities and different topologies.

Example 6.2.2. *Consider to have $n = 50$ users on a complete graph and on a regular graph of degree 10 such that $\alpha_x = a = 45$ and $\beta_x = b = 50$ for every unit x.*

Table 6.2 Performance parameters for $n = 50$

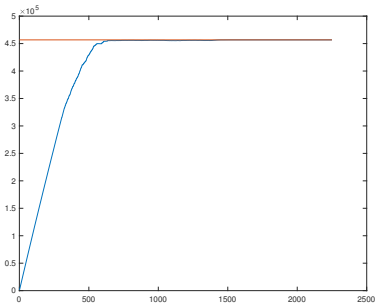
(a) $C^{agg} = 3$

	Complete	Regular
d	1.2400	1.2280
ψ	0.9794	0.9872
Δ	0	0
ν_{moves}	1.8238	2.4538

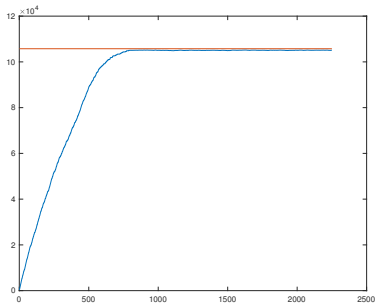
(b) $C^{agg} = -7$

	Complete	Regular
d	45	10
ψ	1	1
Δ	0	0
ν_{moves}	1.3746	1.2898

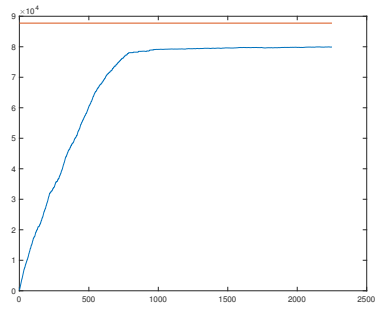
Fig. 6.1 Time evolution of the Potential



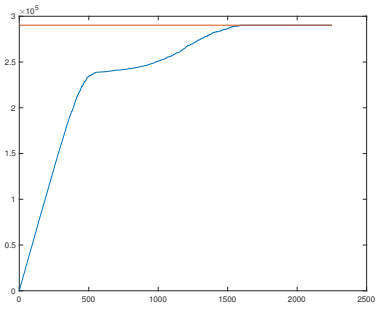
(a) $C^{agg} = -7$



(b) $C^{agg} = -1$

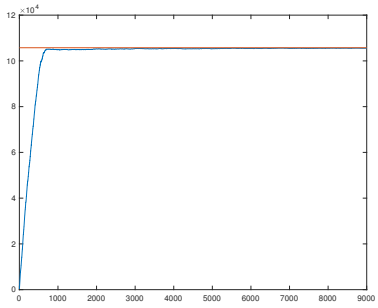


(c) $C^{agg} = 1/2$

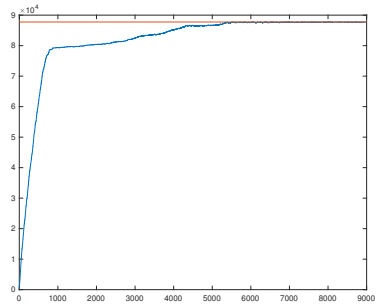


(d) $C^{agg} = 3$

Fig. 6.2 Evolution of the Potential with $T = 20 * \sum_{x \in \mathcal{X}} \alpha_x$



(a) $C^{agg} = -1$



(b) $C^{agg} = 1/2$

While a matching allocation state is not reached when $C^{agg} = 3$, the value of the average degree shows that the solution is quite concentrated with most of units allocating in just one resource. Instead, for $C^{agg} = -7$ we have reached an optimum diffused allocation state. In Figure 6.3 we show how the aggregation parameter influence the allocation: in Figure 6.3a it is shown the underlying graph in the regular case while in Figure 6.3b it is represented the final allocation state.

The next example shows how the presence of heterogeneous resources does not alter much the performance of the algorithm

Example 6.2.3. *Consider to have $n = 50$ users on a complete graph and on a regular graph of degree 10 such that $\alpha_x = a = 43$ for every x. Assume that half of the units have $\beta_x = 40$ and half or them instead $\beta_x = 50$. Notice that, in this case, for the regular graph topology, there is no a-priori guarantee that allocation is feasible. Simulations show however that allocation is reached in all cases.*

Table 6.3 Performance parameters for $n = 50$, different storage capabilities

(a) $C^{agg} = 3$

	Complete	Regular
d	2.0040	2.2760
Δ	0	0
ν_{moves}	2.1540	4.1273

(b) $C^{agg} = -7$

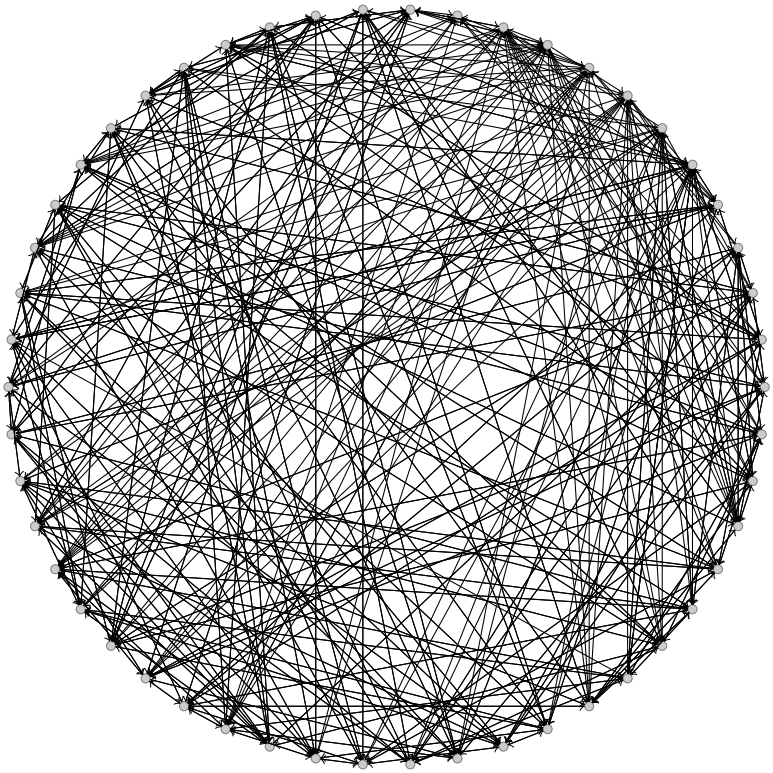
	Complete	Regular
d	42.628	10
Δ	0	0
ν_{moves}	1.9754	1.2862

In the following example we consider larger families of units connected through a regular graph of degree 10. Numerical results show the good scalability properties of the algorithm.

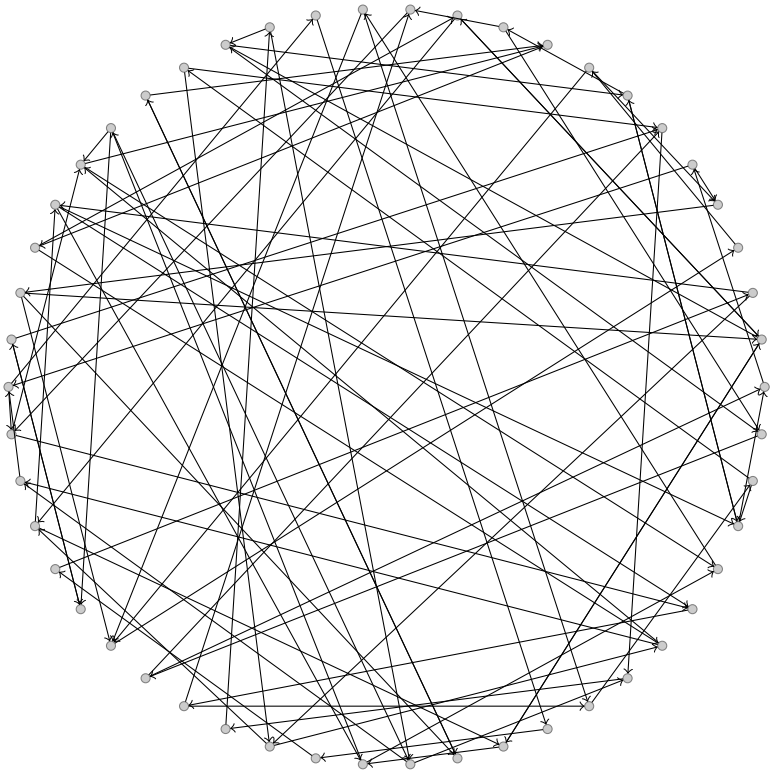
Example 6.2.4. *Suppose to have $n = 100, 200, 300$ users on a regular graph of degree 10 with $\alpha_x = a = 45$ and $\beta_x = b = 50$. The parameters table is presented in the next page.*

Next example consider the case when allocations and distributions are allowed with different granularity Q.

Fig. 6.3 Underlying graph for $n = 50$ users, regular with degree 10



(a) Underlying Network



(b) Used Edges

Table 6.4 Performance parameters for $n = 100, 200, 300$

(a) $C^{agg} = -7$

	$n = 100$	$n = 200$	$n = 300$
d	10	10	10
ψ	1	1	1
Δ	0	0	0
ν_{moves}	1.2535	1.2832	1.2897

(b) $C^{agg} = 3$

	$n = 100$	$n = 200$	$n = 300$
d	1.3980	1.4040	1.4017
ψ	0.9751	0.9753	0.9748
Δ	0	0	0
ν_{moves}	2.0125	1.6346	1.5114

Example 6.2.5. *Consider to have $n = 10$ users on a complete graph such that $\alpha_x = a = 45$ and $\beta_x = b = 50$ for every unit x. We assume that units can allocate or move each time a quantity of data belonging to either $Q_1 = \{1, 5, 10\}$ or $Q_2 = \{1, 25, 45\}$. We also report the case $Q_0 = \{1\}$ for the sake of comparison.*

Table 6.5 Performance parameters for $n = 10$, different granularity

(a) $C^{agg} = -7$

	Q_0	Q_1	Q_2
d	9	9	3.8500
ψ	1	0.9999	0.8902
Δ	0	0	0
ν_{moves}	3.1669	0.2311	0.1767

(b) $C^{agg} = 3$

	Q_0	Q_1	Q_2
d	1	1.0100	1.0100
ψ	1	0.9996	0.9999
Δ	0	0	0
ν_{moves}	3.2449	0.1224	0.0229

As expected, the possibility to allocate at one time larger sets of data drastically reduces the number of allocation and distribution moves and speeds up the algorithm. Notice however that in one case, using the set Q_2, the algorithm does not reach the maximum. This phenomenon is probably due to the fact that allocating large set of data at once can lead to allocation states quite far from the optimum and thus require longer time to converge. This says that the choice of the set Q is likely to play a crucial role in order to optimize the speed of convergence of the algorithm-

Finally, the last example is for the objective functional with the alternative congestion term. Therefore, the utility for each user $x \in \mathcal{X}$ is of the form (6.2).

Example 6.2.6. *Consider to have $n = 50$ users on a complete graph and on regular graph of degree 10 such that $\alpha_x = a = 45$ and $\beta_x = b = 50$ for every unit x. We take $C^{con} = 1$ and C^{all} chosen according to (6.3) while we take different values for*

C^{agg}. As expected, in this case, varying the aggregation parameter C^{agg}, we obtain solutions with a different degree of fragmentation.

Table 6.6 Performance parameters for $n = 50$, alternative congestion term

(a) Regular graph

	$C^{agg} = 0$	$C^{agg} = -1/2$	$C^{agg} = -2$	$C^{agg} = -5$
d	4.7680	8.2849	9.8240	10
Δ	0	0	0	0
ν_{moves}	4.8604	4.4729	4.3748	5

(b) Complete graph

	$C^{agg} = -2$	$C^{agg} = -10$	$C^{agg} = -20$	$C^{agg} = -100$
d	13.9640	21.6400	23	45
Δ	0	0	0	0
ν_{moves}	4.7415	4.9951	5	1.3547

This is particularly evident in the case of a complete graph. The choice of the topology and of the functional parameters can be seen, in this case, as alternative or complementary ways to prescribe the complexity of the allocation in terms of links used.

6.3 Reliability and reciprocity

In this section, we consider the reciprocity algorithm and the reliability learning process of Chapter 3.

The allocation constant is fixed

$$C^{all} = 3(||\alpha||_\infty |C^{agg}| + ||\beta||_\infty C^{con}).$$

Since in this approach the reliability of the users is fundamental, we include this parameter in the utility function that it is now defined as

$$U_x(W) = C^{all} \sum_{y \in \mathcal{X}} W_{xy} + C^{agg} \sum_{y \in \mathcal{X}} W_{xy}^2 - \sum_{y \in N_x^-} C_y^{con} (W_y)^2,$$

with $C_y^{con} = 1 - \lambda_y$. This choice is motivated by the fact that we want users to prefer the most reliable resources even if it disadvantage the congestion.

The first example is to show how the reliability parameter influence the allocation. For this example, we consider the agents to be divided in two families $\mathcal{X}_1$ and $\mathcal{X}_2$ of equal cardinality depending on their reliability. In this part we do not include the reciprocity process but we consider the learning algorithm.

Example 6.3.1. *Suppose there are $n = 10$ users divided in two families $\mathcal{X}_1$ with reliability $\lambda_1 = 0.5$ and $\mathcal{X}_2$ with $\lambda_2 = 0.8$ (we call the families bad and good, respectively). Each agent has $\alpha_x = a = 45$ and $\beta_x = b = 50$. We choose $C_y^{con} = 1 - \lambda_y$ and C^{all} fixed according to (6.3) while we let vary C^{agg}. The following table shows the performance parameters.*

Table 6.7 Performance parameters for n=10 with different reliabilities

	C^{agg} =3	$C^{agg} = -1$	$C^{agg} = -7$
ν_{moves}	1.9297	1.5151	1.0329
d	1.6900	8.9500	9
$\bar{C}^1$	0.8652	0.8000	0.8000
$\bar{C}^2$	0.9348	1.0000	1.0000
$\bar{C}_{var}$	0.0322	0.0100	0.0100
$\bar{C}$	0.9000	0.9000	0.9000
Δ	0	0	0

As expected, more reliable resources are used more than the bad ones. In terms of performance, the final allocations states are quite similar to the classic case. Figure 6.4 shows the learning process through time. In particular it shows the average reliability learned by the users (starting from $\lambda_y = 1$ for all $y \in \mathcal{X}$) during the entire allocation algorithm.

6.3.1 The algorithm with reciprocity

The examples in this section include the reciprocity process. Recall that we analyzed, in Chapter 3, three different approaches to define the probability of acceptance.

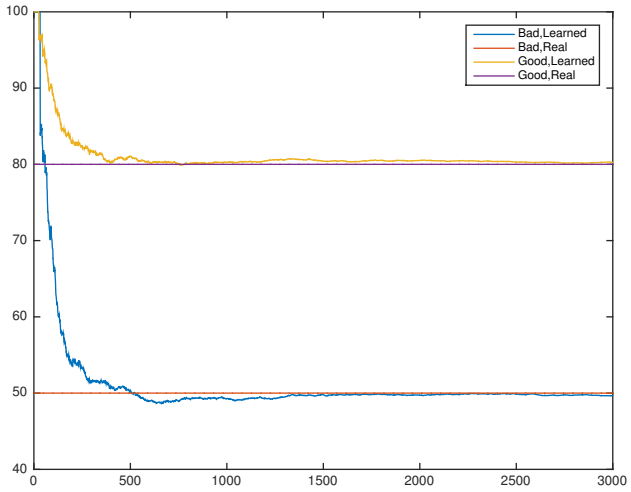


Fig. 6.4 Learning process

1. (**Number of refusal**) This take into consideration the number of times the user who wants to allocate have refused to store the data of the resource. The probability of acceptance in this case is defines as

$$\rho(y,x) = \min\left\{\frac{\eta_x^y + \lambda_x W_{yx}}{\lambda_y(W_{xy}+1)}, 1\right\} \tag{6.4}$$

where η_x^y is the estimation that the user will accept the storage in future.

2. (**Mutual behavior**) Taking into consideration the mutual reliabilities and the quantities allocate we set

$$\rho(y,x) = \min\left\{\frac{\lambda_x(W_{yx}+1)}{\lambda_y(W_{xy}+1)}, 1\right\}. \tag{6.5}$$

3. (**Source/Resource**) This evaluate the behavior of the users as resources with respect to the rest of the neighbor; the probability is defined as

$$\rho(y,x) = \delta\frac{\lambda_x W_{yx}}{\sum\limits_{x\in N_y}\lambda_x W_{yx}} + (1-\delta)\frac{W_{xy}}{\sum\limits_{x\in N_y} W_{xy}} \tag{6.6}$$

where $\delta \in [0,1]$ is a tuning parameter.

In the following set of simulation we want to show how this reciprocity influences the final allocation state as well as the behavior of the users. Moreover, we show how this behavior varies depending on the probability of acceptance and that in certain cases it is possible to exclude from the network a selfish user. By selfish user we mean an agent that, as a resource always deny the allocation from the neighbors but at the same time aims to complete his/her own.

Number of refusal.
We start with some example using the probability of acceptance (6.4). We first present an example with $n = 10$ users and show the final states of the system. Later we will show also some examples for higher numbers of players.

Example 6.3.2. *Consider to have $n = 10$ users on a complete graph. Fix $\alpha_x = a = 45$ and $\beta_x = b = 50$ for all $x \in \mathcal{X}$ and let $C^{agg} = -7, -1, 3$. Suppose that there are no selfish users. The reliability is $\lambda_x = \lambda = 1$. The following matrices represent a sample of the final allocation states reached by the algorithm.*

$$W_3 = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 45 & 0 & 0 & 0 & 0 \\ 0 & 0 & 45 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 45 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 45 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 45 \\ 45 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 45 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 45 & 0 & 0 & 0 \\ 0 & 0 & 0 & 45 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 45 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

$$W_{-1} = \begin{pmatrix} 0 & 4 & 5 & 4 & 4 & 3 & 4 & 6 & 9 & 6 \\ 5 & 0 & 5 & 5 & 4 & 9 & 3 & 6 & 5 & 3 \\ 4 & 4 & 0 & 6 & 5 & 2 & 9 & 2 & 7 & 6 \\ 7 & 3 & 4 & 0 & 3 & 9 & 6 & 5 & 3 & 5 \\ 3 & 3 & 3 & 6 & 0 & 8 & 5 & 4 & 5 & 8 \\ 5 & 4 & 2 & 8 & 5 & 0 & 6 & 2 & 6 & 7 \\ 3 & 3 & 6 & 5 & 3 & 5 & 0 & 9 & 7 & 4 \\ 7 & 6 & 5 & 4 & 5 & 1 & 7 & 0 & 5 & 5 \\ 6 & 3 & 8 & 6 & 6 & 4 & 4 & 4 & 0 & 4 \\ 3 & 7 & 5 & 5 & 4 & 6 & 3 & 10 & 2 & 0 \end{pmatrix}$$

$$W_{-7} = \begin{pmatrix} 0 & 5 & 5 & 5 & 5 & 5 & 5 & 5 & 5 & 5 \\ 5 & 0 & 5 & 5 & 5 & 5 & 5 & 5 & 5 & 5 \\ 5 & 5 & 0 & 5 & 5 & 5 & 5 & 5 & 5 & 5 \\ 5 & 5 & 5 & 0 & 5 & 5 & 5 & 5 & 5 & 5 \\ 5 & 5 & 5 & 5 & 0 & 5 & 5 & 5 & 5 & 5 \\ 5 & 5 & 5 & 5 & 5 & 0 & 5 & 5 & 5 & 5 \\ 5 & 5 & 5 & 5 & 5 & 5 & 0 & 5 & 5 & 5 \\ 5 & 5 & 5 & 5 & 5 & 5 & 5 & 0 & 5 & 5 \\ 5 & 5 & 5 & 5 & 5 & 5 & 5 & 5 & 0 & 5 \\ 5 & 5 & 5 & 5 & 5 & 5 & 5 & 5 & 5 & 0 \end{pmatrix}$$

Even if the allocation state W_3 is a perfect matching, the parameters shows that it is not always the case. In the first column of the table we show also the parameter of the algorithm without reciprocity ($C^{agg} = 3$, reported in the first column of the table) to make a comparison.

Table 6.8 Performance parameters for n=10 with reciprocity

	C^{agg} =3	C^{agg} =3	$C^{agg} = -1$	$C^{agg} = -7$
d	1.7400	1.9700	9	9
ζ	94.5000	53	0	0
ψ	0.9402	0.9349	1	1
Δ	0	0	0	0
ν_{moves}	1.6862	1.6427	2.2749	1.0002

Clearly, for the negative aggregation parameters the symmetry follows straightforward but for the positive value the algorithm with reciprocity is way more symmetric.

For the sake of completeness we also show an example with agents having a certain probability to be on.

Example 6.3.3. *Suppose there are $n = 10$ users with $\alpha_x = 45$, $\beta_x = 50$ and $\lambda_x = 0.8$ for all $x \in \mathcal{X}$. The congestion e allocation parameter are fixed and we vary the aggregation parameter. The table shows the performance; the first column is for the case with our reciprocity.*

Table 6.9 Performance parameters for n=10 with reciprocity and reliability

	C^{agg} =3	C^{agg} =3	$C^{agg} = -1$	$C^{agg} = -7$
d	1.5100	1.9400	9	9
ζ	94	51.8000	0	0
ψ	0.9478	0.9351	1	1
Δ	0	0	0	0
ν_{moves}	1.7651	1.6447	2.5871	1.0053

As one can see, there are no substantial differences from this performance to in the one with $\lambda_x = 1$ for all $x \in \mathcal{X}$.

The following example is for higher number of agents and different topologies.

Example 6.3.4. *Consider to have $n = 50$ users on a complete graph and on a regular graph of degree 10. Fix $\alpha_x = a = 45$, $\beta_x = b = 50$ and $\lambda_x = \lambda = 1$ for all $x \in \mathcal{X}$. Let $C^{agg} = -7,3$ and suppose that there are no selfish users. The following tables (Tables 6.10a and 6.10b) show the parameters.*

The following example shows how our reciprocity process discourage selfish behaviors. With selfish we mean a user who always refuse the allocation of the other players.

Example 6.3.5. *Consider to have $n = 10$ users on a complete graph. Fix $\alpha_x = a = 45$ and $\beta_x = b = 50$ for all $x \in \mathcal{X}$ and let $C^{agg} = -7,-1,3$. The reliability is $\lambda_x = \lambda = 1$.*

Table 6.10 Performance parameters for $n = 50$ with reciprocity

(a) Complete case

	3	-7
d	9.8620	45
ζ	93.2000	8
Δ	0	0
ν_{moves}	1.9146	1.3115

(b) Regular case

	3	-7
d	1.5900	10
ζ	83.2000	8.2000
Δ	0	0
ν_{moves}	2.2485	1.2519

Suppose that user $x = 3$ is selfish. First, we show the final allocation states; on the side we put the allocated quantities for each user.

$$W_3 = \begin{pmatrix} 0 & 0 & 0 & 37 & 0 & 0 & 0 & 0 & 0 & 8 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 43 & 2 \\ 1 & 3 & 0 & 0 & 2 & 1 & 3 & 3 & 4 & 1 \\ 45 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 45 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 41 & 0 & 2 & 2 \\ 0 & 0 & 0 & 0 & 0 & 45 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 2 & 17 & 2 & 6 & 0 & 1 & 15 \\ 0 & 13 & 0 & 0 & 31 & 0 & 0 & 0 & 0 & 1 \\ 1 & 31 & 0 & 11 & 0 & 1 & 0 & 1 & 0 & 0 \end{pmatrix} \begin{matrix} 45 \\ 45 \\ 18 \\ 45 \\ 45 \\ 45 \\ 45 \\ 45 \\ 45 \\ 45 \end{matrix}$$

$$W_{-1} = \begin{pmatrix} 0 & 3 & 0 & 5 & 6 & 4 & 7 & 7 & 5 & 8 \\ 6 & 0 & 0 & 5 & 7 & 7 & 8 & 1 & 5 & 6 \\ 3 & 1 & 0 & 2 & 2 & 1 & 1 & 1 & 3 & 1 \\ 4 & 3 & 0 & 0 & 9 & 6 & 4 & 10 & 5 & 4 \\ 8 & 6 & 0 & 7 & 0 & 3 & 3 & 6 & 7 & 5 \\ 1 & 11 & 0 & 6 & 6 & 0 & 3 & 7 & 5 & 6 \\ 6 & 7 & 0 & 6 & 5 & 8 & 0 & 4 & 3 & 6 \\ 6 & 0 & 0 & 7 & 5 & 6 & 6 & 0 & 10 & 5 \\ 8 & 6 & 0 & 5 & 1 & 5 & 6 & 7 & 0 & 7 \\ 5 & 6 & 0 & 2 & 6 & 5 & 8 & 6 & 7 & 0 \end{pmatrix} \begin{matrix} 45 \\ 45 \\ 15 \\ 45 \\ 45 \\ 45 \\ 45 \\ 45 \\ 45 \\ 45 \end{matrix}$$

$$W_{-7} = \begin{pmatrix} 0 & 5 & 0 & 3 & 6 & 6 & 5 & 7 & 6 & 7 \\ 7 & 0 & 0 & 7 & 5 & 6 & 4 & 6 & 5 & 5 \\ 1 & 3 & 0 & 2 & 2 & 2 & 2 & 2 & 3 & 2 \\ 5 & 6 & 0 & 0 & 8 & 5 & 5 & 5 & 5 & 6 \\ 3 & 6 & 0 & 5 & 0 & 7 & 6 & 7 & 5 & 6 \\ 6 & 5 & 0 & 5 & 7 & 0 & 5 & 3 & 7 & 7 \\ 6 & 4 & 0 & 6 & 4 & 7 & 0 & 6 & 8 & 4 \\ 6 & 7 & 0 & 4 & 7 & 5 & 5 & 0 & 6 & 5 \\ 6 & 5 & 0 & 5 & 6 & 5 & 6 & 6 & 0 & 6 \\ 6 & 5 & 0 & 7 & 4 & 5 & 8 & 7 & 3 & 0 \end{pmatrix} \begin{matrix} 45 \\ 45 \\ 19 \\ 45 \\ 45 \\ 45 \\ 45 \\ 45 \\ 45 \\ 45 \end{matrix}$$

Independently on the aggregation parameter, the allocation of user $x = 3$ is never completed. The parameter Δ (distance from allocation) confirms this behavior.

Table 6.11 Performance parameters for n=10 and a selfish user

	$C^{agg} = 3$	$C^{agg} = -1$	$C^{agg} = -7$
d	3.1800	7.9400	8.0100
ζ	52.6000	22.5000	16
Δ	28	31	31.2000
ν_{moves}	2.3104	1.9491	1.1002

The allocation is not completed even for higher time horizon. These results do not contradicts our analysis since, in Section 3.3.1, we always suppose that there should be a positive probability of acceptance to complete the allocation while here $\rho(3,x) = 0$ for all $x \in \mathcal{X}$.

Mutual behavior.

Now we show some examples with the probability of acceptance (6.5). When needed we include the results of the classical algorithm to make a comparison. The first example is for $n = 10$ users, later we consider higher number of agents and different topologies.

Example 6.3.6. *Consider to have $n = 10$ users on a complete graph. Each of them has $\alpha_x = a = 45$, $\beta_x = b = 50$ and reliability $\lambda_x = 1$. We let vary the aggregation*

parameter $C^{agg} = -7, -1, 3$ *and fix* C_y^{con} *for all* $y \in \mathcal{X}$ *and* C^{all}. *First we show the final allocation states*

$$W_{-1} = \begin{pmatrix} 0 & 5 & 5 & 5 & 5 & 5 & 5 & 5 & 5 & 5 \\ 5 & 0 & 5 & 5 & 5 & 5 & 5 & 5 & 5 & 5 \\ 5 & 5 & 0 & 5 & 5 & 5 & 5 & 5 & 5 & 5 \\ 5 & 5 & 5 & 0 & 5 & 5 & 5 & 5 & 5 & 5 \\ 5 & 5 & 5 & 5 & 0 & 5 & 5 & 5 & 5 & 5 \\ 5 & 5 & 5 & 5 & 5 & 0 & 5 & 5 & 5 & 5 \\ 5 & 5 & 5 & 5 & 5 & 5 & 0 & 5 & 5 & 5 \\ 5 & 5 & 5 & 5 & 5 & 5 & 5 & 0 & 5 & 5 \\ 5 & 5 & 5 & 5 & 5 & 5 & 5 & 5 & 0 & 5 \\ 5 & 5 & 5 & 5 & 5 & 5 & 5 & 5 & 5 & 0 \end{pmatrix}$$

$$W_3 = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 45 \\ 0 & 0 & 0 & 0 & 0 & 45 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 45 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 45 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 45 & 0 & 0 & 0 \\ 0 & 45 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 45 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 45 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 45 & 0 & 0 & 0 & 0 & 0 & 0 \\ 45 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

Table 6.12 Performance parameters for n=10 with mutual probability of acceptance

	$C^{agg} = 3$	$C^{agg} = 3$	$C^{agg} = -1$	$C^{agg} = -7$
d	1.7400	1.8600	9	9
ζ	94.5000	46.9	0	0
ψ	0.9402	0.9381	1	1
Δ	0	0	0	0
ν_{moves}	1.6862	1.7160	2.2784	1

Example 6.3.7. *Suppose there are* $n = 10$ *users with agent* 3 *selfish. Each of the user has* $\alpha_x = a = 45$, $\beta_x = b = 50$ *and* $\lambda_x = 1$. *We let vary the aggregation parameter*

$C^{agg} = -7,-1,3$ and fix C_y^{con} for all $y \in \mathcal{X}$ and C^{all}. We do not include the final allocation states but we show the parameters in the following table.

Table 6.13 Performance parameters for n=10 and a selfish user

	$C^{agg}=3$	$C^{agg}=-1$	$C^{agg}=-7$
d	3.2500	8.1000	8.1000
ζ	71.9000	44.9000	45
Δ	3.3000	0.1000	65.8000
ν_{moves}	1.4062	1.5707	0.8578

Comparing this results with Example 6.3.5 we have that this probability of acceptance give worse performance parameters in terms of selfishness. In fact, for $C^{agg} = 3,-1$ user 3 allocates almost all the data while for $C^{agg} = -7$ not only user 3 does not complete the allocation but also some other agents are unable to store their data.

Example 6.3.8. *Consider to have $n = 50$ users on a complete graph and on a regular graph of degree 10. Suppose $\alpha_x = a = 45$, $\beta_x = b = 50$ and $\lambda_x = \lambda = 1$ for all $x \in \mathcal{X}$. Let $C^{agg} = -7,3$ while C^{con} and C^{agg} are fixed; suppose that there are no selfish users. The following tables shows the parameters.*

Table 6.14 Performance parameters for $n = 50$ with mutual probability of acceptance

(a) Complete case

	$C^{agg}=3$	$C^{agg}=-1$	$C^{agg}=-7$
d	5.5880	45	45
ζ	94.8000	27.4000	30.8000
ψ	0.8486	0.9994	0.9994
Δ	0	0	0
ν_{moves}	1.6910	4.7175	4.9665

(b) Regular case

	$C^{agg}=3$	$C^{agg}=-1$	$C^{agg}=-7$
d	2.3100	9.9980	10
ζ	61.5000	13.3000	8.5000
ψ	0.9436	0.9925	1
Δ	0	0	0
ν_{moves}	3.4886	4.4200	4.9000

Source/Resource.
In this last paragraph we show some simulation with the probability of acceptance (6.6). As we will see, this probability behave worse than the previous two. For this reason we include only an example with 10 users.

Example 6.3.9. *Consider to have $n = 10$ users, each of them with $\alpha_x = a = 45$, $\beta_x = b = 50$ and $\lambda_x = 1$. The congestion and allocation parameter are fixed while we let vary the aggregation constant. The first results are for the usual time horizon $T = 5\sum_x \alpha_x$; below we show the final states of the system.*

$$W_3 = \begin{pmatrix} 0 & 0 & 45 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 45 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 45 & 0 & 0 & 0 \\ 0 & 45 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 45 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 45 \\ 0 & 0 & 0 & 0 & 0 & 45 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 45 & 0 & 0 & 0 & 0 & 0 \\ 45 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 45 & 0 & 0 \end{pmatrix}$$

$$W_{-1} = \begin{pmatrix} 0 & 3 & 4 & 3 & 5 & 3 & 2 & 5 & 4 & 3 \\ 5 & 0 & 3 & 3 & 7 & 2 & 2 & 2 & 2 & 3 \\ 5 & 3 & 0 & 3 & 3 & 4 & 2 & 3 & 2 & 4 \\ 4 & 4 & 4 & 0 & 4 & 5 & 5 & 3 & 4 & 3 \\ 5 & 7 & 5 & 5 & 0 & 4 & 4 & 4 & 4 & 4 \\ 3 & 3 & 3 & 7 & 3 & 0 & 3 & 3 & 4 & 2 \\ 2 & 2 & 2 & 4 & 3 & 2 & 0 & 2 & 2 & 4 \\ 5 & 5 & 5 & 5 & 5 & 5 & 5 & 0 & 5 & 5 \\ 3 & 5 & 3 & 4 & 3 & 3 & 3 & 3 & 0 & 4 \\ 5 & 5 & 6 & 3 & 3 & 3 & 3 & 3 & 5 & 0 \end{pmatrix}$$

$$W_{-7} = \begin{pmatrix} 0 & 3 & 3 & 3 & 3 & 3 & 3 & 2 & 2 & 2 \\ 3 & 0 & 3 & 3 & 3 & 3 & 3 & 3 & 2 & 2 \\ 3 & 3 & 0 & 3 & 3 & 3 & 3 & 3 & 3 & 2 \\ 5 & 5 & 5 & 0 & 4 & 4 & 4 & 4 & 4 & 4 \\ 4 & 4 & 4 & 4 & 0 & 4 & 4 & 4 & 4 & 3 \\ 5 & 4 & 4 & 4 & 4 & 0 & 4 & 4 & 4 & 4 \\ 5 & 5 & 5 & 5 & 5 & 5 & 0 & 4 & 4 & 4 \\ 4 & 4 & 4 & 4 & 4 & 3 & 3 & 0 & 3 & 3 \\ 4 & 4 & 4 & 4 & 4 & 4 & 4 & 3 & 0 & 3 \\ 3 & 3 & 3 & 2 & 2 & 2 & 2 & 2 & 2 & 0 \end{pmatrix}$$

Even though the final state of the system for $C^{agg} = 3$ is a perfect matching, it is clear that it is not symmetric; this means that the probability of acceptance (6.6) does not encourage the reciprocity. The performance parameters show that in general the final allocation state is not symmetric.

Table 6.15 Performance parameters for n=10, Source/Resource, $T = 5\sum_x \alpha_x$

	C^{agg} =3	C^{agg} =3	$C^{agg} = -1$	$C^{agg} = -7$
d	1.7400	2.3700	9	9
ζ	94.5000	94.5000	15.7000	11.6000
ψ	0.9402	0.9223	0.7416	0.7285
Δ	0	5.7000	118.6000	125.2000
ν_{moves}	1.6862	1.3333	0.7678	0.7285

Unfortunately, in none of the cases the allocation is completed. This is because the probability of acceptance is very small in this approach, especially when the resources are almost full but the allocated quantities per each user are small. Increasing the time horizon to $T = 20\sum_x \alpha_x$ the algorithm gives a better performance (shown in Table 6.16).

Clearly, if one consider the time horizon to be approximated with the estimations of Chapter 4 (in particular with the upper bound given by Proposition 4.3.3) such

Table 6.16 Performance parameters for n=10, Source/Resource, $T = 20\sum_x \alpha_x$

	C^{agg} =3	$C^{agg} = -1$	$C^{agg} = -7$
d	1.9500	9	9
ζ	93	0	0
ψ	0.9593	1	1
Δ	0	0	0
ν_{moves}	2.0847	1.0584	1.0004

allocation problem will not be shown. Since this approach require a higher amount of time and therefore gives worse performances than the other cases, we do not include other examples that will show the same behavior.

6.4 Multiple copies algorithm

In this section we show the results for the variant with multiple copies. We will simulate two possible allocation algorithm. The first one allows the agents to allocate all the copies at once. The second suppose to allocate only a single copy at a time. The first approach is the fastest since in the second case we have to consider at least σ iterations for each atom. However, in both the cases the Nash equilibrium is reached.

As in the classical model and following the theoretical analysis, we take the parameter γ time-varying and diverging to $+\infty$. The tuning of the divergence rate is known to be critical to obtain good results. Here we have chosen

$$\gamma(t+1) = \gamma(t) + \frac{1}{n * 1000}$$

Most of the time, we suppose the units to be always on as we already show that there are no substantial differences.

The congestion constant in the utility function is fixed and equal for all users, i.e. $C_y^{con} = C^{con} = 1$ for all $y \in \mathcal{X}$ since we do not include the reliability here. Indeed, the utility function is of the form expressed in (6.1).

6.4.1 Allocation of all copies at once

Since the users allocate all the copies at once, the time horizon can be taken as in Section 6.2, i.e., $T = 5\sum_{x\in\mathcal{X}} \alpha_x$: in this way a unit x will activate, on average, a number of times equal to 5 times the number of data it needs to allocate. The first example is for $n = 10$ users, later we consider higher numbers of player and different topologies

Example 6.4.1. *Suppose there are $n = 10$ users on a complete graph. Each of them has a quantity $\alpha = 18$ to be allocated in $\sigma = 3$ copies; the resource capability is $\beta = 60$. C^{all} and C^{con} are fixed as usual while we let vary $C^{agg} = 3, 1/2, -1$. First, we show the final allocation states.*

$$W_3 = \begin{pmatrix} 0 & 18 & 0 & 0 & 0 & 0 & 18 & 0 & 0 & 18 \\ 18 & 0 & 0 & 0 & 18 & 0 & 0 & 0 & 18 & 0 \\ 18 & 0 & 0 & 0 & 0 & 0 & 0 & 18 & 18 & 0 \\ 0 & 0 & 18 & 0 & 0 & 18 & 0 & 0 & 0 & 18 \\ 18 & 18 & 0 & 0 & 0 & 0 & 18 & 0 & 0 & 0 \\ 0 & 0 & 18 & 18 & 0 & 0 & 0 & 18 & 0 & 0 \\ 0 & 0 & 0 & 18 & 18 & 0 & 0 & 0 & 18 & 0 \\ 0 & 18 & 0 & 0 & 0 & 0 & 18 & 0 & 0 & 18 \\ 0 & 0 & 18 & 18 & 0 & 18 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 18 & 18 & 0 & 18 & 0 & 0 \end{pmatrix}$$

$$W_{\frac{1}{2}} = \begin{pmatrix} 0 & 18 & 0 & 1 & 18 & 0 & 0 & 17 & 0 & 0 \\ 17 & 0 & 17 & 0 & 0 & 17 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 18 & 0 & 18 & 18 \\ 0 & 0 & 1 & 0 & 18 & 1 & 0 & 17 & 0 & 17 \\ 0 & 0 & 0 & 18 & 0 & 0 & 18 & 0 & 18 & 0 \\ 0 & 18 & 0 & 0 & 0 & 0 & 0 & 18 & 0 & 18 \\ 17 & 16 & 0 & 16 & 1 & 1 & 0 & 1 & 1 & 1 \\ 18 & 0 & 0 & 0 & 18 & 0 & 18 & 0 & 0 & 0 \\ 1 & 0 & 18 & 17 & 0 & 16 & 0 & 2 & 0 & 0 \\ 0 & 0 & 18 & 0 & 0 & 18 & 0 & 0 & 18 & 0 \end{pmatrix}$$

$$W_{-1} = \begin{pmatrix} 0 & 6 & 6 & 6 & 6 & 6 & 6 & 6 & 6 & 6 \\ 6 & 0 & 6 & 6 & 6 & 6 & 6 & 6 & 6 & 6 \\ 6 & 6 & 0 & 6 & 6 & 6 & 6 & 6 & 6 & 6 \\ 6 & 6 & 6 & 0 & 6 & 6 & 6 & 6 & 6 & 6 \\ 6 & 6 & 6 & 6 & 0 & 6 & 6 & 6 & 6 & 6 \\ 6 & 6 & 6 & 6 & 6 & 0 & 6 & 6 & 6 & 6 \\ 6 & 6 & 6 & 6 & 6 & 6 & 0 & 6 & 6 & 6 \\ 6 & 6 & 6 & 6 & 6 & 6 & 6 & 0 & 6 & 6 \\ 6 & 6 & 6 & 6 & 6 & 6 & 6 & 6 & 0 & 6 \\ 6 & 6 & 6 & 6 & 6 & 6 & 6 & 6 & 6 & 0 \end{pmatrix}$$

While the diffused allocation state seems equal to the classical case, in the aggregated state W_3 it is evident how the diversification constraint influence the allocation. The following table shows the parameters.

Table 6.17 Performance parameters for $n = 10$, all copies at once

	$C^{agg} = -1$	$C^{agg} = 1/2$	$C^{agg} = 3$
d	9	3.5300	3.5300
ψ	1	0.9988	0.9933
Δ	0	0	0
ν_{moves}	1.1389	2.0922	1.6294

The fact that the final degree is not exactly 3 is due to the fact that also for the distribution moves, agents have to re-store all the three copies.

Example 6.4.2. *Cosider $n = 50$ users on a complete graph and on a regular graph of degree 10. Each of them has $\beta_x = b = 60$ and a quantity $\alpha_x = a = 18$ to be allocated in $\sigma = 3$ copies. C^{all} and C^{con} are fixed as usual while we let vary $C^{agg} = 3, 1/2, -1$. The table shows the performance parameters in both the cases.*

6.4.2 Allocation of one copy at a time

In this section we consider each copy as a single atom, in the sense that the agents have to remember where the stored copies of an atom are allocated to guarantee the

Table 6.18 Performance parameters for $n = 50$, all copies at once

(a) Complete case

	$C^{agg} = -1$	$C^{agg} = 1/2$	$C^{agg} = 3$
d	48.976	3.7480	3.9540
ψ	0.9995	0.9980	0.9936
Δ	0	0	0
ν_{moves}	1.1673	2.0867	1.6047

(b) Regular case

	$C^{agg} = -1$	$C^{agg} = 1/2$	$C^{agg} = 3$
d	10	3.9020	3.9320
ψ	1	0.9976	0.9897
Δ	0	0	0
ν_{moves}	1.2613	2.3792	1.9793

diversification. Notice that in this case, since users allocate one copy at a time, the total amount of data to be allocated is $\sigma \sum_{x \in \mathcal{X}} \alpha_x$ therefore the time horizon is fixed $T = 5 * \sigma \sum_{x \in \mathcal{X}} \alpha_x$. Moreover, the parameter ν_{moves} represent the number of moves per each copy while we introduce ν^{σ}_{moves} to indicate the number of moves per atom.

As usual, we start with an example with $n = 10$ users.

Example 6.4.3. *Consider $n = 10$ users on a complete graph, each of them having a quantity $\alpha_x = a = 18$ to be allocated in $\sigma = 3$ copies. The resource capability is $\beta_x = b = 60$ for all $x \in \mathcal{X}$. C^{all} and C^{con} are fixed as usual while we let vary $C^{agg} = 3, 1/2, -1$. First, we show the final allocation states.*

$$W_{-1} = \begin{pmatrix} 0 & 6 & 6 & 6 & 6 & 6 & 6 & 6 & 6 & 6 \\ 6 & 0 & 6 & 6 & 6 & 6 & 6 & 6 & 6 & 6 \\ 6 & 6 & 0 & 6 & 6 & 6 & 6 & 6 & 6 & 6 \\ 6 & 6 & 6 & 0 & 6 & 6 & 6 & 6 & 6 & 6 \\ 6 & 6 & 6 & 6 & 0 & 6 & 6 & 6 & 6 & 6 \\ 6 & 6 & 6 & 6 & 6 & 0 & 6 & 6 & 6 & 6 \\ 6 & 6 & 6 & 6 & 6 & 6 & 0 & 6 & 6 & 6 \\ 6 & 6 & 6 & 6 & 6 & 6 & 6 & 0 & 6 & 6 \\ 6 & 6 & 6 & 6 & 6 & 6 & 6 & 6 & 0 & 6 \\ 6 & 6 & 6 & 6 & 6 & 6 & 6 & 6 & 6 & 0 \end{pmatrix}$$

$$W_{1/2} = \begin{pmatrix} 0 & 0 & 0 & 18 & 18 & 0 & 0 & 0 & 0 & 18 \\ 18 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 18 & 18 \\ 0 & 0 & 0 & 0 & 0 & 18 & 18 & 0 & 18 & 0 \\ 18 & 0 & 18 & 0 & 0 & 0 & 0 & 18 & 0 & 0 \\ 0 & 0 & 0 & 18 & 0 & 18 & 18 & 0 & 0 & 0 \\ 0 & 18 & 0 & 0 & 18 & 0 & 0 & 18 & 0 & 0 \\ 18 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 18 & 18 \\ 0 & 18 & 18 & 0 & 18 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 18 & 0 & 0 & 18 & 18 & 0 & 0 & 0 \\ 0 & 18 & 0 & 18 & 0 & 0 & 0 & 18 & 0 & 0 \end{pmatrix}$$

From the table it emerges that this approach is better, in terms of convergence, than the previous one with the allocation of all the copies at once.

Table 6.19 Performance parameters for $n = 10$, one copy at a time

	$C^{agg} = -1$	$C^{agg} = 1/2$
d	9	3
ψ	1	1
Δ	0	0
ν_{moves}	3.2694	4.8139
ν^{σ}_{moves}	1.0898	1.6046

Example 6.4.4. *Suppose there are $n = 50$ users on a complete graph and on a regular graph of degree 10. Each of them has a quantity $\alpha_x = a = 18$ to be allocated in $\sigma = 3$ copies; the resource capability is $\beta_x = b = 60$. We let vary $C^{agg} = 3, -1$ while C^{all} and C^{con} are fixed as usual. The performance parameters are in Tables 6.20a and 6.20b.*

6.5 Conclusions

In this section we presented a wide set of simulation that validate the theoretical analysis. In the fist part we show the convergence properties of the algorithm, varying not only the congestion, aggregation and allocation parameters but also the users

Table 6.20 Performance parameters for $n = 50$, all copies at once

(a) Complete case

	$C^{agg} = -1$	$C^{agg} = 3$
d	49	3.5640
ψ	0.9995	0.9981
Δ	0	0
ν_{moves}	3.7631	4.6373
ν^{σ}_{moves}	1.2677	1.5458

(b) Regular case

	$C^{agg} = -1$	$C^{agg} = 3$
d	9	3.4300
ψ	1	0.9922
Δ	0	0
ν_{moves}	3.2694	3.4174
ν^{σ}_{moves}	1.0898	1.1391

quantities α_x and β_x. Moreover we show how the reliability influence the allocation, encouraging the agents in the choice of more trustable resources. On the other hand, the plots of Figure 6.4 show the good properties of the learning process that lead the users to know the reliability of their neighbors.

Section 6.3 is devoted to the reciprocity algorithm. We present the different probabilities of acceptance and their influence on the final configuration. This set of simulation, not only confirms the convergence properties of the algorithm with reciprocity, but also offers the possibility to compare the probabilities of acceptance. Indeed, while the "Number of Refusal" approach has some nice symmetry properties, the "Source/Resource" probability is too small to guarantee the allocation in a reasonable amount of time. From this section emerges that the first probability of acceptance is the best choice since it gives the agents the possibility to exclude a selfish user from completing the allocation.

The last part of this chapter include the simulation for the algorithm with multiple copies. Even though the theoretical analysis is not completed yet, the simulation suggest that the algorithm reaches convergence to a Nash equilibrium that is also a maximum of the potential. A more careful analysis of the theory is needed but these results represent an interesting starting point.

Chapter 7

Wisdom of crowds and naive learning

7.1 Introduction

Since social network are primary conduits of opinion and informations, carry news about products, influence decisions and drive political opinions toward other groups, it is important to understand how beliefs change through time, how this changes depend on the network structure and whether the final outcome is reasonable.

Social networks in mathematics typically consist of a graph where each node possesses a state variable; the interconnection between the individuals depends on the edges in the underlying graph [8]. Imagine that a number of independent individuals possess an information represented by a real number (for instance an opinion on a given fact). Agents interact and change their opinion by averaging with the opinions of other individuals. A natural system that aggregates individual opinions is an influence system, that is, a social dynamic process whereby individual opinions, starting from their respective initial conditions, evolve possibly towards consensus. In social sciences, empiric evidence [10] has shown how such aggregate opinion may give a very good estimation of unknown quantities: such phenomenon has been proposed in the literature as *wisdom of crowds* [15].

The wisdom of the crowd is the collective opinion of a group of individuals rather than that of a single expert. The term crowd refers to any group of people and it is not supposed to be cohesive. A large group answering to questions has been often found to be better than the answer given by any of the individuals within the group. An explanation for this phenomenon is that each individual opinion is influenced by

a noise and, taking the average over a large number of responses, its effect will be in some way canceled.

In this chapter we study a well known model of social interaction: the DeGroot model [5, 9], which is known to reach a consensus (under proper assumptions). The social structure of the DeGroot model is described by a weighted and possibly directed network. Agents have beliefs about some common question of interest, communicate with their neighbors in the network and update their opinion. An agent's new belief is the weighted average of her neighbors' opinion from the previous instant of time. Over time, provided some connectivity and aperiodicity condition on the underlying graph, beliefs converge to a consensus [5, 4].

The purpose of our work is to show under which condition the population is finite-time wise. We start by reviewing the naive learning model for large populations proposed by Golub and Jackson [11] and based on the DeGroot opinion formation process. In this model, the population is wise if the final (in the limit as time grows) consensus opinion is equal to the correct value. This property is cast in terms of the left dominant eigenvector of the influence matrix in large populations (i.e., in the limit as the number of individual diverges). Specifically, a wise population is a sequences of row-stochastic matrices whose left dominant eigenvector satisfies a "vanishing maximum entry condition." This condition ensures that no individual's initial erraneous opinion has an outside impact on the final consensus opinion, thereby rendering the population unwise.

In this chapter we consider a variation of the naive learning setting in which the DeGroot opinion formation process is allowed only finite time and does not, therefore, reach completion. Our individuals do not reach consensus and we say that a population is finite-time wise if the average of the individuals opinion remains correct as time progresses along the opinion dynamics process. In other words, Golub and Jackson [11] consider wisdom in the limit in which $n \to \infty$ after $k \to \infty$, while we consider the limit in which $n \to \infty$ after at $k = 1$, k fixed, and uniformly over k. We argue that these finite-time variation are especially relevant for large population, since it is known that the time required for consensus to be approximately achieved typically diverges as the population size diverges.

We apply these findings in two cases: the Erdös-Rényimodel and the preferential attachment model. The wisdom in all possible senses (finite-time, uniformly and infinite-time) follows quite easily in the first model. On the other hand, we show

how the classic Barabási-Albert model (with attachment probability linearly proportional to degree) is wise in all possible senses, whereas a super-linear preferential attachment model loses all its wisdom properties.

The chapter is organized as follow. Section 7.2 contains various wisdom definitions and corresponding general conditions. Section 7.3 contains the analysis of several counterexamples showing the general difference among the various wisdom notions. A sufficient condition for finite-time wisdom is given in Section 7.4 and in Section 7.5 we characterize pre-uniform wisdom in sequences of equal-neighbor matrices. The last section contains some concluding remarks.

7.1.1 Review of notation and preliminary concepts

Let $\mathbb{1}_n \in \mathbb{R}^{n\times 1}$ denote the vector of all ones. Given $x \in \mathbb{R}^n$, we define its *average* by $\text{ave}(x) = \frac{1}{n}\mathbb{1}_n^\top x = \frac{1}{n}\sum_{i=1}^n x_i$. Given $x \in \mathbb{R}^n$, its *norms* by $\|x\|_1 = \sum_i |x_i|$, $\|x\|_2 = (\sum_i x_i^2)^{1/2}$ and $\|x\|_\infty = \max_i |x_i|$. Let $\Delta_n = \{x \in \mathbb{R}^n \mid x \geq 0, \mathbb{1}_n^\top x = 1\}$. Given $x \in \Delta_n$, the useful inequality

$$x^\top x \leq \|x\|_\infty \tag{7.1}$$

follows from $x^\top x \leq \|x\|_\infty \|x\|_1 = \|x\|_\infty$.

Stochastic matrices A matrix $P \in \mathbb{R}^{n\times n}$ is non-negative ($P \geq 0$) if $P_{ij} \geq 0$ for each i, j. For such a matrix, the maximum column sum is

$$\|P\|_1 = \max_{j\in\{1,\ldots,n\}} \sum_{i=1}^n P_{ij} = \max \mathbb{1}_n^\top P.$$

Note that $\|\frac{1}{n}P\|_1$ is the maximum column average of P. To a nonnegative matrix P we associate an unweighted directed graph $G = (\{1,\ldots,n\}, E)$ where $E = \{(i,j) \in \{1,\ldots,n\}^2 \mid P_{ij} > 0\}$. Given a node j, let N_j denote the set of out-neighbors of j. The nonnegative matrix P is *irreducible* if G is strongly connected and *primitive* if G is strongly connected and aperiodic.

A matrix $P \in \mathbb{R}^{n\times n}$ is *stochastic* if $P \geq 0$ and $P\mathbb{1}_n = \mathbb{1}_n$.

Given a stochastic irreducible matrix P, its *left dominant eigenvector* $\pi \in \Delta_n$ is unique and well defined by the Perron-Frobenious Theorem and satisfies $\pi^\top P = \pi^\top$.

Given a primitive stochastic matrix P, its *mixing time* is defined by

$$\tau_{\mathrm{mix}}(P) := \inf\left\{ t \in \mathbb{N} \;\middle|\; \max_{i,j} \sum_k |(P)^t_{ik} - P)^t_{jk}| \leq \frac{1}{2\mathrm{e}} \right\}. \tag{7.2}$$

Equal-neighbor models

Definition 7.1.1 (Equal-neighbor, directed equal-neighbor, and weighted-neighbor matrices)**.** *Given a nonnegative matrix $W \in \mathbb{R}^{n\times n}_{\geq 0}$ with at least one positive entry in each row, define the stochastic matrix $P \in \mathbb{R}^{n\times n}$ by*

$$P = \mathrm{diag}(W\mathbb{1}_n)^{-1} W. \tag{7.3}$$

Then the matrix P is said to be

1. equal-neighbor *if $W \in \{0,1\}^{n\times n}$ and $W = W^\top$,*
2. directed equal-neighbor *if $W \in \{0,1\}^{n\times n}$ and $W \neq W^\top$, and*
3. weighted-neighbor *if $W \in \mathbb{R}^{n\times n}_{\geq 0}$ and $W = W^\top$.*

In these three definitions, the graph associated to W is undirected unweighted, directed unweighted, and undirected weighted respectively.

Note that, given a non-negative weight matrix $W \in \mathbb{R}^{n\times n}_{\geq 0}$, the *weighted out-degree* of a node i in the weighted digraph associated to W is $d_i = (W\mathbb{1}_n)_i = \sum_{j=1}^n W_{ij}$. Because we assume $d_i > 0$ for all i, equation (7.3) is well-posed and equivalent to:

$$P_{ij} = d_i^{-1} W_{ij}.$$

If W is symmetric, then d_i is called the *weighted degree*. If additionally W is binary, then d_i is called the *degree* and

$$P_{ij} = d_i^{-1} W_{ij} = \begin{cases} d_i^{-1}, & \text{if } j \in N_i, \\ 0, & \text{otherwise .} \end{cases} \tag{7.4}$$

We conclude with a known useful result. The left dominant eigenvector π of an irreducible equal-neighbor matrix P satisfies, for all $i \in \{1,\dots,n\}$,

$$\pi_i = \frac{d_i}{d_1 + \cdots + d_n}. \tag{7.5}$$

7.2 Definitions and basic results

We consider the classic French-DeGroot model [9, 5] of opinion dynamics

$$x_i(k+1) = \sum_{j=1}^{n} P_{ij} x_j(k), \tag{7.6}$$

where $x_i(k)$ denotes the opinion of individual i, $i \in \{1,\dots,n\}$ at time $k \in \mathbb{Z}_{\geq 0}$. The coefficient P_{ij} denotes the weight that individual i assigns to individual j when carrying out this revision. The matrix P is assumed row-stochastic and defines a weighted directed graph G.

We assume

$$x_i(0) = \mu + \xi_i(0), \tag{7.7}$$

where the constant $\mu \in \mathbb{R}$ is unknown parameter and the noisy terms $\xi_i(0)$ are a family of independent Gaussian-distributed variables such that $\mathbb{E}[\xi_i(0)] = 0$ and $\mathrm{Var}[\xi_i(0)] = \sigma^2 < \infty$ for all $i \in \{1,\dots,n\}$.

We will consider sequences of DeGroot models (7.6) with increasing dimensions n and denote all relative quantities with a superscript $[n]$. In particular, the state of the n-dimensional model is $x^{[n]}$. Given assumption (7.7) and assuming μ is kept constant, the law of large numbers implies that, almost surely,

$$\lim_{n\to\infty} \mathrm{ave}(x^{[n]}(0))) = \mu.$$

Definition 7.2.1 (Wisdom notions)**.** *Given a* sequence of stochastic matrices of increasing dimensions $\{P^{[n]} \in \mathbb{R}^{n\times n}\}_{n\in\mathbb{N}}$*, define a sequence of opinion dynamics problems with initial state* $\{x^{[n]}(0) \in \mathbb{R}^n\}_{n\in\mathbb{N}}$ *satisfying assumption* (7.7) *and evolution* $\{x^{[n]}(k) \in \mathbb{R}^n\}_{n\in\mathbb{N}}$ *at times* $k \in \mathbb{N}$*. The sequence* $\{P^{[n]} \in \mathbb{R}^{n\times n}\}_{n\in\mathbb{N}}$*, is*

1. one-time wise *if* $\lim_{n\to\infty} \mathrm{ave}(x^{[n]}(1))) = \mu$*;*

2. finite-time wise *if* $\lim_{n\to\infty} \text{ave}(x^{[n]}(k))) = \mu$, *for all* $k \in \mathbb{N}$;

3. wise *if* $\lim_{n\to\infty} \lim_{k\to\infty} \text{ave}(x^{[n]}(k))) = \mu$;

4. uniformly wise *if* $\lim_{n\to\infty} \sup_{k\in\mathbb{N}} |\text{ave}(x^{[n]}(k))) - \mu| = 0$.

Here all limits are meant in the probability sense.

The following theorem provides necessary and sufficient characterizations of the notions 1, 2, and 3 of the above definition. The proof of these characterizations is immediate from [11, 14].

Theorem 7.2.2 (Necessary and sufficient conditions for finite-time wisdom and wisdom)**.** *Consider a* sequence of stochastic matrices of increasing dimensions $\{P^{[n]} \in \mathbb{R}^{n\times n}\}_{n\in\mathbb{N}}$. *The sequence is*

1. one-time wise *if and only if* $\lim_{n\to\infty} \|\frac{1}{n}P^{[n]}\|_1 = 0$;

2. finite-time wise *if and only if, for all* $k \in \mathbb{N}$, $\lim_{n\to\infty} \|\frac{1}{n}(P^{[n]})^k\|_1 = 0$.

Moreover, assuming $\{P^{[n]}\}_{n\in\mathbb{N}}$ *are primitive, the sequence is*

3. wise *if and only if* $\lim_{n\to\infty} \|\pi^{[n]}\|_\infty = 0$, *where* $\pi^{[n]} \in \Delta_n$ *is the left dominant eigenvector of* $P^{[n]}$, *for* $n \in \mathbb{N}$.

Proof. Define

$$\chi^{[n]}(k) = \frac{1}{n}\mathbb{1}_n^\top (P^{[n]})^k \tag{7.8}$$

and notice that

$$\text{ave}(x^{[n]}(k)) = \mu + (\chi^{[n]}(k))^\top \xi^{[n]}(0).$$

Thus,

$$\begin{aligned}\mathbb{E}[\text{ave}(x^{[n]}(k))] &= \mu,\\ \text{Var}[\text{ave}(x^{[n]}(k))] &= \sigma^2(\chi^{[n]}(k))^\top \chi^{[n]}(k).\end{aligned}$$

By Chebyshev's inequality, the convergence in probability of $\text{ave}(x^{[n]}(k))$ to μ is equivalent to the convergence to 0 of the variance. Statements 1 and 2 can

now be proven by applying the inequality (7.1) with $x = \chi^{[n]}(k)$ and noting that $\|\chi^{[n]}(k)\|_\infty = \|\frac{1}{n}(P^{[n]})^k\|_1$.

Regarding statement 3, the primitivity assumption implies $\lim_{k\to+\infty} x^{[n]}(k) = (\pi^{[n]})^\top \big(\mu \mathbb{1}_n + \xi^{[n]}(0)\big)$ where $\pi^{[n]}$ is the left dominant eigenvector of $P^{[n]}$. Therefore,

$$\lim_{k\to+\infty} \operatorname{ave}(x^{[n]}(k)) = \mu + (\pi^{[n]})^\top \xi^{[n]}(0),$$

and statement 3 follows by applying the inequality (7.1) with $x = \pi^{[n]}$. □

There is a natural property that, in analogy to the other characterizations presented in Theorem 7.2.2, is related to uniform wisdom. We present it as a separated concept.

Definition 7.2.3 (Pre-uniform wisdom)**.** *A sequence of stochastic matrices of increasing dimensions* $\{P^{[n]} \in \mathbb{R}^{n\times n}\}_{n\in\mathbb{N}}$ *is* pre-uniformly wise *if* $\lim_{n\to\infty} \sup_{k\in\mathbb{N}} \|\frac{1}{n}(P^{[n]})^k\|_1 = 0$.

While pre-uniform wisdom is not sufficient to guarantee uniform wisdom, it surely yields wisdom and finite-time wisdom, as an immediate consequence of Theorem 7.2.2.

Finally, for what concerns uniform wisdom, we present a sufficient condition that turns out to be useful in many applications. Recall the notion of mixing time from equation (7.2).

Theorem 7.2.4 (A sufficient condition for uniform wisdom)**.** *A* sequence of primitive stochastic matrices of increasing dimensions $\{P^{[n]} \in \mathbb{R}^{n\times n}\}_{n\in\mathbb{N}}$ *is uniformly wise if*

$$\lim_{n\to+\infty} \sup_{k\in\mathbb{N}} \left\| \frac{1}{n}(P^{[n]})^k \right\|_1 \tau_{\text{mix}}(P^{[n]}) = 0.$$

Proof. Define

$$Y^{[n]}(k) = \operatorname{ave}(x^{[n]}(k))) - \mu = \chi^{[n]}(k)^\top \xi^{[n]}(0),$$

where $\chi^{[n]}(k)$ is defined in (7.8). Notice that

$$\lim_{k\to+\infty} Y^{[n]}(k) = Y^{[n]} := (\pi^{[n]})^\top \xi^{[n]}(0).$$

Fix $\delta > 0$ and notice that

$$\mathbb{P}\Big[\sup_{k\in\mathbb{N}}|Y^{[n]}(k)| > \delta\Big] \leq \mathbb{P}\Big[\sup_{k\in\mathbb{N}}|Y^{[n]}(k) - Y^{[n]}| > \delta/2\Big] + \mathbb{P}\left[|Y^{[n]}| > \delta/2\right]. \quad (7.9)$$

We now estimate the two right-end-side terms of equation (7.9). Regarding the first term, we fix k and we use Chebyshev's inequality to obtain

$$\begin{aligned}
&\mathbb{P}[|Y^{[n]}(k) - Y^{[n]}| > \delta/2] \\
&\qquad \leq \frac{4}{\delta^2}\operatorname{Var}[Y^{[n]}(k) - Y^{[n]}] \\
&\qquad \leq \frac{4}{\delta^2}(\chi^{[n]}(k) - \pi^{[n]})^\top(\chi^{[n]}(k) - \pi^{[n]}) \\
&\qquad \leq \frac{4}{\delta^2}\|\chi^{[n]}(k) - \pi^{[n]}\|_\infty\|\chi^{[n]}(k) - \pi^{[n]}\|_1 \\
&\qquad \leq \frac{8}{\delta^2}\sup_{k\in\mathbb{N}}\left\|\frac{1}{n}(P^{[n]})^k\right\|_1\|\chi^{[n]}(k) - \pi^{[n]}\|_1, \qquad (7.10)
\end{aligned}$$

where last step follows from the relation $\|\chi^{[n]}(k)\|_\infty = \|\frac{1}{n}(P^{[n]})^k\|_1$ and the limit $\chi^{[n]}(k) \to \pi^{[n]}$ for $k \to +\infty$. The final 1-norm factor in equation (7.10) can be estimated in terms of the mixing time [12, Section 4.5]

$$\|\chi^{[n]}(k) - \pi^{[n]}\|_1 \leq \exp\left(-\left\lfloor\frac{k}{\tau_{\mathrm{mix}}(P^{[n]})}\right\rfloor\right). \qquad (7.11)$$

Substituting into inequality (7.10) and using a union bound estimation, we now obtain

$$\begin{aligned}
\mathbb{P}\Big[\sup_{k\in\mathbb{N}}|Y^{[n]}(k) - Y^{[n]}| > \delta/2\Big] &\leq \frac{8}{\delta^2}\sup_{k\in\mathbb{N}}\left\|\frac{1}{n}(P^{[n]})^k\right\|_1\sum_k\exp\left(-\left\lfloor\frac{k}{\tau_{\mathrm{mix}}(P^{[n]})}\right\rfloor\right) \\
&= \frac{8e}{\delta^2}\sup_{k\in\mathbb{N}}\left\|\frac{1}{n}(P^{[n]})^k\right\|_1\frac{1}{1-\exp\left(-\frac{1}{\tau_{\mathrm{mix}}(P^{[n]})}\right)} \\
&\leq \frac{8e}{\delta^2}\sup_{k\in\mathbb{N}}\left\|\frac{1}{n}(P^{[n]})^k\right\|_1\tau_{\mathrm{mix}}(P^{[n]}). \qquad (7.12)
\end{aligned}$$

The second term in the right-hand-side of equation (7.9) can be easily bounded using Chebyshev's inequality and the simple bound (7.1):

$$\begin{aligned}\mathbb{P}[|Y^{[n]}| > \delta/2] &\leq \frac{4}{\delta^2}\operatorname{Var}[Y^{[n]}] = \frac{4}{\delta^2}(\pi^{[n]})^\top(\pi^{[n]}) \\ &\leq \frac{4}{\delta^2}\|\pi^{[n]}\|_\infty \\ &\leq \frac{4}{\delta^2}\sup_{k\in\mathbb{N}}\left\|\frac{1}{n}(P^{[n]})^k\right\|_1. \end{aligned} \tag{7.13}$$

The theorem statement follows from putting together the inequalities (7.9), (7.12), and (7.13). □

7.3 Basic implications and counterexamples

In the following lemma we show how the basic wisdom notions in Definitions 7.2.1 and 7.2.3 are related by a simple implication and how four counterexamples show that no additional statements may be made in general. To construct our counterexamples, we will rely upon the equal neighbor models as in Definition 7.1.1.

Lemma 7.3.1 (Basic implications and counterexamples)**.** *If a sequence of primitive stochastic matrices of increasing dimensions $\{P^{[n]} \in \mathbb{R}^{n\times n}\}_{n\in\mathbb{N}}$ is pre-uniformly wise, then it is wise and finite-time wise. Moreover, there exist sequences that*

1. *are neither wise nor finite-time wise (see Example 7.3.2),*
2. *are wise, but not one-time wise (see Example 7.3.3),*
3. *are finite-time wise, but not wise (see Example 7.3.4), and*
4. *are wise and finite-time wise, but not pre-uniformly wise (see Example 7.3.5).*

We illustrate this lemma in Figure 7.1. The proof of the first statement in the lemma is an immediate consequence of Definition 7.2.3 and the characterizations in Theorem 7.2.2. The four existence statement are established by providing explicit examples later in this section.

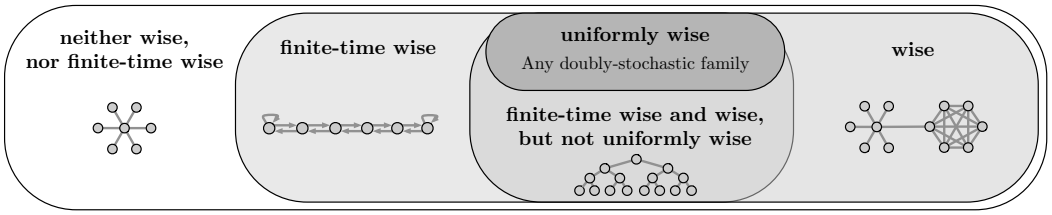


Fig. 7.1 Logical relations among sets of sequences with varying degree of wisdom. For each set we provide an example sequence that belongs to that set, but not to any strict subset of it.

Example 7.3.2 (The star graph is neither wise nor finite-time wise)**.** *For $n \in \mathbb{N}$, define $\{P^{[n]} \in \mathbb{R}^{n\times n}\}_{n\in\mathbb{N}}$ by*

$$P^{[n]} = \begin{bmatrix} 1/n & 1/n & 1/n & \cdots & 1/n \\ 1 & 0 & 0 & \vdots & 0 \\ 1 & 0 & 0 & \vdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & 0 & 0 & \cdots & 0 \end{bmatrix}.$$

In other words, consider the sequence of primitive equal-neighbor matrices defined by increasing-dimension star graphs (with a self-loop at the central node). It is easy to see that the dominant left eigenvector of $P^{[n]}$ is $\pi^{[n]} = \begin{bmatrix} \frac{n}{2n-1} & \frac{1}{(2n-1)} & \cdots & \frac{1}{(2n-1)} \end{bmatrix}^\top$. The sequence $\{P^{[n]}\}_{n\in\mathbb{N}}$ is neither wise, nor one-time wise because

$$\begin{aligned} \lim_{n\to\infty} \|\pi^{[n]}\|_\infty &= \lim_{n\to\infty} \pi_1^{[n]} = \frac{1}{2}, \\ \lim_{n\to\infty} \|\tfrac{1}{n}P^{[n]}\|_1 &= \lim_{n\to\infty} \max_{j\in\{1,\ldots,n\}} \frac{1}{n}\sum_{i=1}^{n} P_{ij}^{[n]} \\ &= \lim_{n\to\infty} \frac{1}{n}\sum_{i=1}^{n} P_{i1}^{[n]} = \lim_{n\to\infty} \frac{n-1+1/n}{n} = 1. \end{aligned}$$

□

Example 7.3.3 (The union/contraction of star and complete graph is wise, but not one-time wise)**.** *For $n \in \mathbb{N}$, let $S_n \sqcup K_n$ denote the undirected graph with $2n$ nodes obtained by (i) computing the union of the star S_n (one center node and n leafs) and a complete graph K_n, and (ii) identifying/contracting one leaf of S_n with a node of K_n. Accordingly, let $\{W^{[n]}\}_{n\in\mathbb{N}}$ be the sequence of adjacency matrices of $\{S_n \sqcup K_n\}_{n\in\mathbb{N}}$ and $\{P^{[n]}\}_{n\in\mathbb{N}}$ be the corresponding sequence of equal-neighbor matrices. These*

matrices are primitive because K_n contains cycles with co-prime length. Note the slight abuse of notation: $P^{[n]}$ has dimension $2n$.

The graph $S_n \sqcup K_n$, for $n = 6$, is depicted in Figure 7.2, where nodes are numbered as follows: node 1 is always the center of the star and node $n+1$ is the node belonging to both the star and the complete graph.

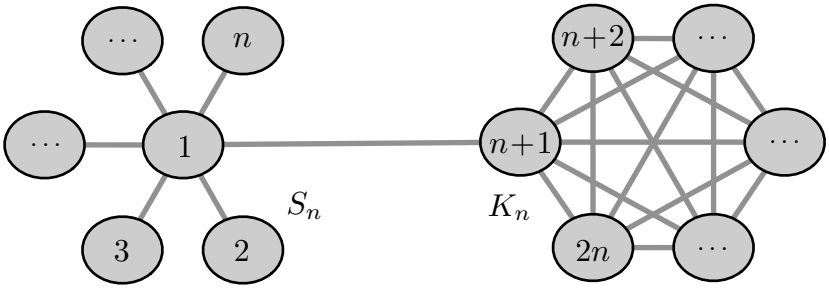


Fig. 7.2 The union and contraction of a star graph with a complete graph.

For this graph $S_n \sqcup K_n$, from equation (7.5)*, we compute the left dominant eigenvector of $P^{[n]}$. First, we observe that: $d_1 = n$, $d_j = 1$ for a generic leaf $j \in \{2,\dots,n\}$, $d_{n+1} = n$, and $d_h = n-1$ a generic node $h \in \{n+2,\dots,2n\}$ in K_n. Hence, the sum of all degrees is $n+1\cdot(n-1)+n+(n-1)^2 = n^2+n$ and the left dominant eigenvector satisfies:*

$$\pi_1^{[n]} = \frac{n}{n^2+n}, \quad \pi_j^{[n]} = \frac{1}{n^2+n},$$
$$\pi_{n+1}^{[n]} = \frac{n}{n^2+n}, \quad \pi_h^{[n]} = \frac{n-1}{n^2+n}.$$

Next, we compute the column sums of $P^{[n]}$. For the two special nodes we have

$$\sum_{i=1}^{2n} P_{i1}^{[n]} = \sum_{i=2}^{n} P_{i1}^{[n]} + P_{n+1,1}^{[n]} = (n-1) + \frac{1}{n},$$
$$\sum_{i=1}^{2n} P_{i,n+1}^{[n]} = P_{1,n+1}^{[n]} + \sum_{i=n+2}^{2n} P_{i,n+1}^{[n]} = 1 + \frac{1}{n},$$

and, for a generic leaf $j \in \{2,\ldots,n\}$ in S_n and a generic $h \in \{n+2,\ldots,2n\}$ in K_n, we have

$$\sum_{i=1}^{2n} P_{ij}^{[n]} = \frac{1}{n},$$
$$\sum_{i=1}^{2n} P_{ih}^{[n]} = P_{n+1,h}^{[n]} + \sum_{i=n+2}^{2n} P_{ih}^{[n]} = \frac{1}{n} + (n-2)\frac{1}{n-1}.$$

In summary, we note that the sequence $\{P^{[n]}\}_{n\in\mathbb{N}}$ is wise but not one-time wise because

$$\lim_{n\to\infty} \|\pi^{[n]}\|_\infty = \lim_{n\to\infty} \pi_1^{[n]} = \lim_{n\to\infty} \frac{n}{n^2+n} = 0,$$
$$\lim_{n\to\infty} \|\tfrac{1}{2n}P^{[n]}\|_1 = \lim_{n\to\infty} \max_{j\in\{1,\ldots,2n\}} \frac{1}{2n}\sum_{i=1}^{2n} P_{ij}^{[n]}$$
$$= \lim_{n\to\infty} \frac{1}{2n}\sum_{i=1}^{2n} P_{i1}^{[n]} = \lim_{n\to\infty} \frac{n-1+\frac{1}{n}}{2n} = \frac{1}{2}. \qquad \square$$

Example 7.3.4 (The path graph with biased weights is finite-time wise, but not wise)**.** *Consider a sequence of increasing-dimension path graphs whose biased weights are selected as follows: (i) pick a constant $\nu > 1$ and define the unique scalars $q > p > 0$ satisfying $q/p = \nu$ and $p+q = 1$, and (ii) define the n-dimensional weighted digraph as in Figure 7.3 (self-loops at node 1 and n). Let $\{P^{[n]}\}_{n\in\mathbb{N}}$ denote the sequence of primitive stochastic adjacency matrices of the the path graphs with biased weights. We have:*

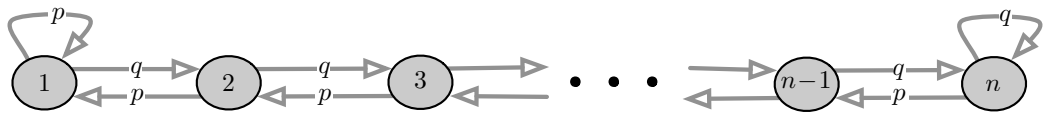


Fig. 7.3 Directed path with biased weights

$$P^{[n]} = \begin{bmatrix} p & q & \cdots & 0 & 0 \\ p & 0 & q & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & 0 \\ 0 & \ddots & p & 0 & q \\ 0 & 0 & \cdots & p & q \end{bmatrix}.$$

Let $\pi^{[n]}$ be the dominant left eigenvector of $P^{[n]}$. We claim that

$$\pi_1^{[n]} = \frac{\nu - 1}{\nu^n - 1}, \qquad \pi_i^{[n]} = \nu^{i-1}\pi_1^{[n]}, \text{ for } i \in \{2, \ldots, n\}. \tag{7.14}$$

We prove this claim as follows. From $(\pi^{[n]})^\top = (\pi^{[n]})^\top P^{[n]}$, we get

$$\begin{aligned} \pi_1^{[n]} p + \pi_2^{[n]} p &= \pi_1^{[n]}, \\ \pi_{i-1}^{[n]} q + \pi_{i+1}^{[n]} p &= \pi_i^{[n]}, \quad \text{for } i \in \{2, \ldots, n-1\}, \\ \pi_{n-1}^{[n]} q + \pi_n^{[n]} q &= \pi_n^{[n]}. \end{aligned}$$

From the first equality we immediately have $\pi_2^{[n]} = \nu\pi_1^{[n]}$. Next we prove by recursion that $\pi_i^{[n]} = \nu^{i-1}\pi_1^{[n]}$. This statement is true for $i = 2$. Assuming it is true for arbitrary i, we compute

$$\begin{aligned} \pi_{i+1}^{[n]} &= \frac{1}{p}\pi_i^{[n]} - \frac{q}{p}\pi_{i-1}^{[n]} = \frac{1}{p}\nu^{i-1}\pi_1^{[n]} - \frac{q}{p}\nu^{i-2}\pi_1^{[n]} \\ &= \left(\frac{1}{p}\nu - \frac{q}{p}\right)\nu^{i-2}\pi_1^{[n]} = \frac{1-p}{p}\nu^{i-1}\pi_1^{[n]} = \frac{q}{p}\nu^{i-1}\pi_1^{[n]}. \end{aligned}$$

The value of $\pi_1^{[n]}$ in (7.14) *follows from requiring $1 = \sum_{i=1}^n \pi_i^{[n]} = \sum_{i=1}^n \nu^{i-1}\pi_1^{[n]} = \frac{\nu^n - 1}{\nu - 1}\pi_1^{[n]}$. This concludes our proof of the formula* (7.14) *for $\pi^{[n]}$.*

Note that from formula (7.14)*, we know* $\pi_n^{[n]} = \frac{\nu-1}{\nu^n-1}\nu^{n-1} \asymp \frac{\nu-1}{\nu}$ *for* $n \to \infty$*. In summary, we note that the sequence* $\{P^{[n]}\}_{n\in\mathbb{N}}$ *is not wise but finite-time wise because*

$$\begin{aligned}
\lim_{n\to\infty} \|\pi^{[n]}\|_\infty &= \lim_{n\to\infty} \pi_n^{[n]} = \frac{\nu-1}{\nu} > 0, \\
\lim_{n\to\infty} \|\tfrac{1}{n}(P^{[n]})^k\|_1 &\le \lim_{n\to\infty} \|\tfrac{1}{n}P^{[n]}\|_1^k \\
&= \lim_{n\to\infty} \Big(\max_{j\in\{1,\dots,n\}} \frac{1}{n}\sum_{i=1}^n P_{ij}^{[n]} \Big)^k \\
&\le \lim_{n\to\infty} \frac{2^k}{n} = 0, \quad \textit{for all } k \in \mathbb{N},
\end{aligned}$$

where we used the fact that the in-degree of each node is upper bounded by 2. □

Example 7.3.5 (The reversed binary tree with root-leaves edges is wise and finite–time wise, but not pre-uniformly wise)**.** *We define a sequence of directed equal-neighbor model (see Definition 7.1.1) by defining the corresponding sequence of binary (not symmetric)* $W^{[n]}$ *matrices. We consider a binary tree with* $k-1$ *layers and* n *nodes, where* $n = 1+\cdots+2^{k-1} = \frac{2^k-1}{2-1} = 2^k-1$*. The edge direction is opposite the usual convention (from root to leaves), there is a self loop in the root node, and we add a directed edge from root to all leaves; see Figure 7.4. As in Figure 7.4, we label the nodes as follows:* $v_1^{(1)}$ *at layer* 1*,* $v_1^{(2)}, v_2^{(2)}$ *at layer* 2*, and, more generally,* $v_1^{(\ell)}, \cdots, v_{2^{\ell-1}}^{(\ell)}$ *at layer* ℓ*, for* $\ell \in \{1,\dots,k-1\}$*.*

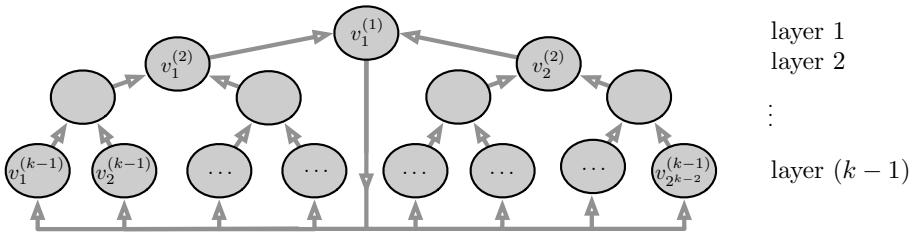


Fig. 7.4 A reversed binary directed tree with root-leaves edges.

We now compute the left dominant eigenvector $\pi^{[n]}$*. For symmetry reasons, at each layer* $\ell \in \{1,\dots,k-1\}$*, we have* $\pi^{[n]}(v_1^{(\ell)}) = \cdots = \pi^{[n]}(v_{2^{\ell-1}}^{(\ell)})$*. Next, define* $\pi^{(\ell)} = \pi^{[n]}(v_1^{(\ell)}) + \cdots + \pi^{[n]}(v_{2^{\ell-1}}^{(\ell)}) = 2^{\ell-1}\pi^{[n]}(v_1^{(\ell)})$ *and note that an aggregation argument leads to* $\pi^{(2)} = \cdots = \pi^{(k-1)} = \frac{1}{k}$ *and* $\pi^{(1)} = 2\pi^{(2)} = \frac{2}{k}$*. Hence, in summary,*

for each node $h \in \{1,\dots,2^{\ell-1}\}$ *at layer* ℓ,

$$\pi^{[n]}(v_h^{(\ell)}) = \begin{cases} \frac{2}{k}, & \text{if } \ell = 1, \\ \frac{1}{2^{\ell-1}}\frac{1}{k}, & \text{if } \ell > 1. \end{cases}$$

In summary, the sequence is wise since

$$\begin{aligned} \|\pi^{[n]}\|_\infty &= \max_{\ell\in\{1,\dots,k-1\}} \max_{h\in\{1,\dots,2^{\ell-1}\}} \pi^{[n]}(v_h^{(\ell)}) \\ &= \pi^{[n]}(v_1^{(1)}) = \frac{2}{k} = \frac{1}{\log_2(n+1)}, \end{aligned}$$

and therefore $\lim_{n\to\infty} \|\pi^{[n]}\|_\infty = 0$.

Next, we note $\sum_{i=1}^n P_{ij} \le \sum_{i=1}^n W_{ij} \le 3$, *where we used* $P_{ij} = d_i^{-1} W_{ij}$, $d_i \ge 1$ *and knowledge of the fact that each node has at most 3 in-edges. Therefore, the sequence is finite-time wise since, for all fixed time h:*

$$\begin{aligned} \lim_{n\to\infty} \|\tfrac{1}{n}(P^{[n]})^h\|_1 &\le \lim_{n\to\infty} \|\tfrac{1}{n}P^{[n]}\|_1^h \\ &= \lim_{n\to\infty} \Big(\max_{j\in\{1,\dots,n\}} \frac{1}{n}\sum_{i=1}^n P_{ij}^{[n]} \Big)^h \\ &\le \lim_{n\to\infty} \frac{2^h}{n} = 0. \end{aligned}$$

Finally, the sequence is not pre-uniformly wise because, when k and n grow as $n = 2^k - 1$, *we estimate*

$$\lim_{n\to\infty} \frac{1}{n}\sum_{i=1}^n P^{k-1}_{iv_1^{(1)}} \ge \lim_{n\to\infty} \frac{1}{n} 2^{k-1} = \lim_{n\to\infty} \frac{n+1}{2n} = \frac{1}{2},$$

where the first inequality follows if one deletes the term relative to the self loop in the matrix P. □

These four examples complete the list of counterexamples in Lemma 7.3.1. We may make one obvious additional statement: finite-time wisdom implies one-time wisdom. The converse is not true, as established by the following counterexample.

Example 7.3.6 (The weighted double-star graph is one-time wise, but not two-time wise)**.** *We define a sequence of weighted-neighbor matrices (recall Definition 7.1.1)*

as follows. For each n, as illustrated in Figure 7.5, the graph is a double star with one root node (labelled j), $\sqrt{n}$ intermediate nodes (a representative node is labelled h) and n leafs (a representative node is labelled i). For simplicity we assume $\sqrt{n}$ is a natural number. The symmetric edge weights are selected as follows: 1 *between root and the intermediate nodes and* $1/\sqrt{n}$ *between the intermediate nodes and the leafs. Note that the total number of nodes is* $n+\sqrt{n}+1 \asymp n$*. We add a self-loop with unit weight at the root node j so that each matrix in the sequence is primitive.*

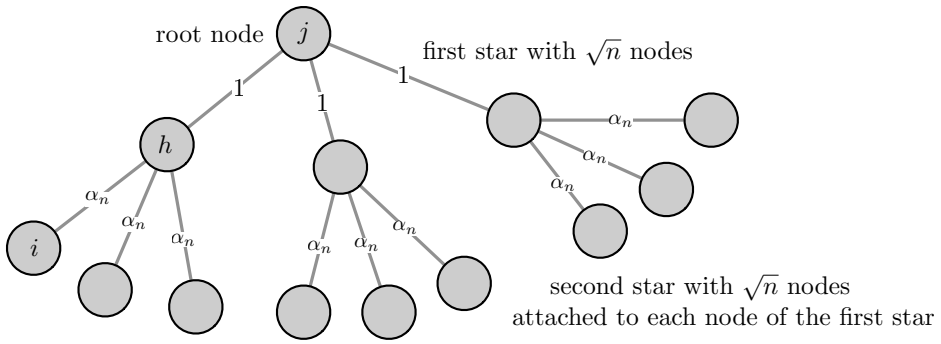


Fig. 7.5 A weighted double-star graph; we select $\alpha_n = 1/\sqrt{n}$.

With these definitions, we have $W_{hj} = W_{jh} = 1$ *and, therefore, the weighted degree of the root note j is* $d_j = \sqrt{n}+1$*. We also have* $W_{ih} = W_{hi} = \alpha_n = 1/\sqrt{n}$ *and, therefore, the weighted degree of each intermediate node h is* $d_h = 1+\sqrt{n}\alpha_n = 2$ *and the weighted degree of each leaf i is* $d_i = \alpha_n = 1/\sqrt{n}$*.*

From the equality $P = \mathrm{diag}(W\mathbb{1}_n)^{-1}W$ *defining a weighted-neighbor model, we compute* $P_{ih} = d_i^{-1}W_{ih} = 1$*, and* $P_{jh} = 1/(\sqrt{n}+1)$ *so that the column sum of P corresponding to an intermediate node h is* $\sqrt{n}\times 1+1\times 1/(\sqrt{n}+1) \asymp \sqrt{n}$*. Similarly, we compute* $P_{hj} = d_h^{-1}W_{hj} = 1/2$ *so that the column sum of P corresponding to the root node j is* $\sqrt{n}/2+1/(\sqrt{n}+1)$*. Therefore, the sequence is one-time wise.*

Finally, for each leaf i, we have $(P^2)_{ij} = P_{ih}P_{hj} = 1/2$*. Since there are n leafs, the column sum of* P^2 *corresponding to the root node j is at least* $n/2 \asymp n$*. The matrix sequence is therefore not wise at time two.* □

7.4 A sufficient condition for finite-time wisdom

In this section we provide an insightful sufficient condition guaranteeing finite-time-wisdom for general sequences of matrices. The main proof idea is to introduce a recursive bound on the total influence of families of nodes. This bound amounts to a stronger property than finite-time wisdom and, unlike what occurs for finite-time wisdom, can be easily checked in terms of the matrix sequence.

We start with the following useful definition.

Definition 7.4.1 (Prominent families)**.** *Given a sequence of stochastic matrices of increasing dimensions* $\{P^{[n]} \in \mathbb{R}^{n\times n}\}_{n\in\mathbb{N}}$*, a sequence of sets of nodes* $\{\mathbb{B}^{[n]} \subset \{1,\dots,n\}\}_{n\in\mathbb{N}}$ *is said to be a* $P^{[n]}$-prominent family *if*

1. *its size is negligible, that is,* $|\mathbb{B}^{[n]}| = o(n)$*, but*
2. *its* total one-time influence *is order* 1*, that is*

$$\frac{1}{n}\sum_{i=1}^{n}\sum_{j\in\mathbb{B}^{[n]}} P_{ij}^{[n]} \asymp 1.$$

While one-time wisdom does not imply finite-time wisdom in general sequences (see Example 7.3.6), the following key result shows how the absence of prominent families is inherited by the powers of $P^{[n]}$.

Theorem 7.4.2 (The absence of prominent families is persistent and implies finite–time wisdom)**.** *Consider a sequence of stochastic matrices of increasing dimensions* $\{P^{[n]} \in \mathbb{R}^{n\times n}\}_{n\in\mathbb{N}}$*. If there is no* $P^{[n]}$*-prominent family, then*

1. *for every k, there is no* $(P^{[n]})^k$*-prominent family, and*
2. $P^{[n]}$ *is finite-time wise.*

In order to prove Theorem 7.4.2 we provide some general notions and then establish the recursive influence bound. Given an $n\times n$ stochastic matrix P, define the *maximum one-time influence of a set of nodes* $\Phi_P : \{0,\dots,n\} \to \mathbb{R}_{\geq 0}$ by

$$\begin{cases} \Phi_P(0) = 0, \\ \Phi_P(s) = \max\limits_{\mathbb{B}\subseteq\{1,\dots,n\},|\mathbb{B}|=s} \sum\limits_{i=1}^{n}\sum\limits_{j\in\mathbb{B}} P_{ij}, \quad s\in\{1,\dots,n\}. \end{cases}$$

Clearly, $\Phi_P(s)$ is non-decreasing in s. Notice moreover that

$$\Phi_P(1) = \max_{i\in\{1,\dots,n\}} \sum_{i=1}^{n} P_{ij} = \|P\|_1,$$
$$\Phi_P(n) = \sum_{i=1}^{n}\sum_{j=1}^{n} P_{ij} = n.$$

We can extend the definition of Φ_P and define $\Phi_P : \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$ by $\Phi_P(x) = \Phi_P(\lfloor x \rfloor)$. Note that Φ_P remains a non-decreasing function.

Remark 7.4.3. *We present two simple statements without proof.*

1. *The absence of $P^{[n]}$-prominent families is equivalent to the following fact:*

$$\forall \{a_n \in \mathbb{R}_{\geq 0}\}_{n\in\mathbb{N}} \text{ satisfying } a_n = o(n),\ \Phi_{P^{[n]}}(a_n) = o(n). \tag{7.15}$$

2. *Since $n^{-1}\|P^{[n]}\|_1 = n^{-1}\Phi_{P^{[n]}}(1)$, the previous statement and Theorem 7.2.2 together imply that, if there is no $P^{[n]}$-prominent family, then $P^{[n]}$ is one-time wise.* □

The following technical result will play a crucial role in our derivations.

Lemma 7.4.4 (Recursive influence bound)**.** *Let P, Q be $n \times n$ stochastic matrices. For every $\delta > 0$ and $s \in \{0,\dots,n\}$, it holds*

$$\Phi_{PQ}(s) \leq \Phi_P(\delta\Phi_Q(s)) + \frac{n}{\delta}.$$

Proof. Define the shorthand $\mathcal{V} = \{1,\dots,n\}$. Consider any $\mathbb{B} \subseteq \mathcal{V}$ and define

$$N_{\mathbb{B}}^{Q}(\delta) = \left\{h \in \mathcal{V} \;\middle|\; \textstyle\sum_{j\in\mathbb{B}} Q_{hj} \geq \delta^{-1}\right\}.$$

Notice that

$$\begin{aligned}\sum_{i\in\mathcal{V}}\sum_{j\in\mathbb{B}}(PQ)_{ij} &= \sum_{i\in\mathcal{V}}\sum_{h\in\mathcal{V}}\sum_{j\in\mathbb{B}}P_{ih}Q_{hj}\\ &= \sum_{i\in\mathcal{V}}\sum_{h\in N_{\mathbb{B}}^{Q}(\delta)}P_{ih}\sum_{j\in\mathbb{B}}Q_{hj}+\sum_{i\in\mathcal{V}}\sum_{h\notin N_{\mathbb{B}}^{Q}(\delta)}P_{ih}\sum_{j\in\mathbb{B}}Q_{hj}\\ &\leq \sum_{i\in\mathcal{V}}\sum_{h\in N_{\mathbb{B}}^{Q}(\delta)}P_{ih}+\sum_{i\in\mathcal{V}}\sum_{h\notin N_{\mathbb{B}}^{Q}(\delta)}P_{ih}\delta^{-1}\\ &\leq \Phi_P\left(|N_{\mathbb{B}}^{Q}(\delta)|\right)+\frac{n}{\delta},\end{aligned} \tag{7.16}$$

where last inequality follows from the definition of Φ_P and the trivial bound

$$\sum_{i\in\mathcal{V}}\sum_{h\notin N_{\mathbb{B}}^{Q}(\delta)}P_{ih}\leq 1.$$

Note that

$$\Phi_Q(|\mathbb{B}|)\geq\sum_{h\in\mathcal{V}}\sum_{j\in\mathbb{B}}Q_{hj}\geq\sum_{h\in N_{\mathbb{B}}^{Q}(\delta)}\sum_{j\in\mathbb{B}}Q_{hj}\geq|N_{\mathbb{B}}^{Q}(\delta)|\cdot\delta^{-1},$$

which implies

$$|N_{\mathbb{B}}^{Q}(\delta)|\leq\delta\Phi_Q(|\mathbb{B}|).$$

Therefore,

$$\sum_{i\in\mathcal{V}}\sum_{j\in\mathbb{B}}(PQ)_{ij}\leq\Phi_P\left(\delta\Phi_Q(|\mathbb{B}|)\right)+\frac{n}{\delta}.$$

Now, fix a size s and the proof is complete by computing the maximum value of the left and right-hand side over all subsets $\mathbb{B}\subseteq\mathcal{V}$ with $|\mathbb{B}|=s$. □

We are finally ready to prove the main result in this section.

Proof of Theorem 7.4.2. We start by proving statement 1. Given Remark 7.4.31, it suffices to to show that

$$\forall\{a_n\in\mathbb{R}_{\geq 0}\}_{n\in\mathbb{N}}\text{ satisfying }a_n=o(n),\ \ \Phi_{(P^{[n]})^k}(a_n)=o(n). \tag{7.17}$$

We proceed by induction on k. Indeed we know from our assumption that the statement in equation (7.17) is true for $k=1$. We now suppose it is true for k and we

prove it for $k+1$. We fix a sequence $\{a_n\}$ such that $a_n = o(n)$. Lemma 7.4.4 implies

$$\Phi_{(P^{[n]})^{k+1}}(a_n) \leq \Phi_{P^{[n]}}\left(\delta\Phi_{(P^{[n]})^k}(a_n)\right) + \frac{n}{\delta}.$$

The induction assumption implies that $\Phi_{(P^{[n]})^k}(a_n) = o(n)$ and, by Remark 7.4.31, we have that

$$\Phi_{P^{[n]}}\left(\delta\Phi_{(P^{[n]})^k}(a_n)\right) = o(n).$$

Therefore

$$\limsup_{n\to\infty} \frac{\Phi_{(P^{[n]})^{k+1})}(a_n)}{n} \leq \frac{1}{\delta}$$

and, because δ is arbitrary,

$$\limsup_{n\to\infty} \frac{\Phi_{(P^{[n]})^{k+1})}(a_n)}{n} = 0.$$

This equality completes the proof by induction and proves statement 1. Statement 2 follow from statement 1 considering that

$$\frac{1}{n}\|(P^{[n]})^k\|_1 = \frac{1}{n}\Phi_{(P^{[n]})^k}(1).$$

□

7.5 A necessary and sufficient condition for pre-uniform wisdom

In this section we focus on sequence of equal-neighbor stochastic matrices (recall Definition 7.1.1). For this setting we are able to provide a complete characterization of one-time, finite-time and pre-uniform wisdom. We start with a revised notion of prominence.

Definition 7.5.1 (Prominent individuals)**.** *Given a sequence of stochastic matrices of increasing dimensions* $\{P^{[n]} \in \mathbb{R}^{n\times n}\}_{n\in\mathbb{N}}$*, a sequence of individuals* $\{k^{[n]} \in \{1,\dots,n\}\}_{n\in\mathbb{N}}$ *is said to be* $P^{[n]}$-prominent *if its* total one-time influence *is order* 1*, that is*

$$\frac{1}{n}\sum_{i=1}^{n} P^{[n]}_{ik^{[n]}} \asymp 1.$$

In other words, a prominent individual is a prominent family composed of a single individual at each n.

Remark 7.5.2. *Some comments are in order.*

1. *The absence of prominent families implies the absence of prominent individuals; but that the absence of prominent individual does not imply the absence of prominent family (e.g., take $\sqrt{n}$ nodes each with $\sqrt{n}$ neighbors with degree* 1 *in an equal neighbor sequence).*

2. *The existence of prominent individuals means that there exists a sequence of nodes $\{k^{[n]} \in \{1,\ldots,n\}\}_{n\in\mathbb{N}}$ such that node $k^{[n]}$ possesses order n neighbors with order* 1 *degree.* □

We next show how the absence of prominent individuals and the notion of one-time wisdom play a key role in equal-neighbor sequences.

Theorem 7.5.3 (The absence of prominent individuals is necessary and sufficient for pre-uniform wisdom in equal-neighbor sequences)**.** *Consider a sequence of equal-neighbor matrices of increasing dimensions $\{P^{[n]}\}_{n\in\mathbb{N}}$. The following statements are equivalent:*

1. *there is no $P^{[n]}$-prominent individual,*

2. *the sequence is one-time wise, and*

3. *the sequence is pre-uniformly wise.*

In other words, one-time wisdom implies finite-time wisdom and pre-uniform wisdom for the setting of equal-neighbor sequences as well as wisdom for the setting of primitive equal-neighbor sequences. Recall that for more general sequences, e.g., the setting of weighted-neighbor models, one-time wisdom does not imply two-time wisdom (see Example 7.3.6).

Based on Theorem 7.5.3 (and specifically on the fact that one-time wisdom implies wisdom) and Example 7.3.3 (showing an example of a wise but not one-time wise sequence), we classify the equal-neighbor sequences as shown in Figure 7.6.

In order to prove Theorem 7.5.3 we establish some useful bounds.

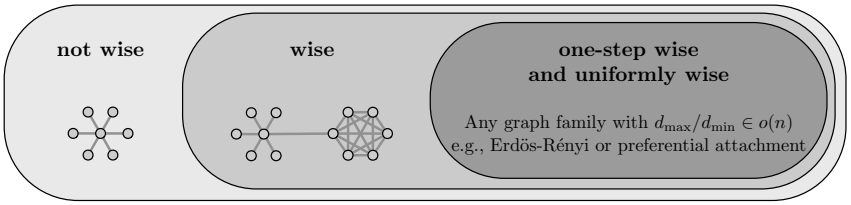


Fig. 7.6 Logical relations among sets of equal-neighbor sequences with varying degree of wisdom. For each set we provide an example of a equal-neighbor sequence.

Lemma 7.5.4 (One-norm bounds)**.** *Any equal-neighbor matrix $P \in \mathbb{R}^{n\times n}$ satisfies, for all $k \in \mathbb{N}$,*

$$\left\|\tfrac{1}{n}P^k\right\|_1 \leq 2\left\|\tfrac{1}{n}P\right\|_1^{1/2}. \tag{7.18}$$

Remark 7.5.5. *The square root is unavoidable, i.e., it is not possible to obtain a bound similar to* (7.18) *for equal-neighbor matrices that does not involve the square root of the one-norm of P. For example, the double-star with $\sqrt{n}$ aperture is an example where $\|\frac{1}{n}P\|_1 = 1/n$ and $\|\frac{1}{n}P^2\|_1 = 1/\sqrt{n}$.* □

Proof of Lemma 7.5.4. Let $P = \operatorname{diag}(W\mathbb{1}_n)^{-1}W$ for a binary and symmetric W, let G denote the undirected graph defined by W, pick a node $j \in \{1,\dots,n\}$ and a constant $\delta > 0$. Define

$$N_j(\delta) = \left\{h \in \{1,\dots,n\} \;\middle|\; P_{hj} \geq \frac{1}{\delta}\right\} = \{h \in N_j \mid d_h \leq \delta\}. \tag{7.19}$$

We now compute

$$\begin{aligned} P_{ij}^k &= \sum_{h=1}^{n} P_{ih}^{k-1}P_{hj} = \sum_{h\in N_j(\delta)} P_{ih}^{k-1}P_{hj} + \sum_{h\notin N_j(\delta)} P_{ih}^{k-1}P_{hj} \\ &\leq \sum_{h\in N_j(\delta)} P_{ih}^{k-1}P_{hj} + \sum_{h\notin N_j(\delta)} P_{ih}^{k-1}\frac{1}{\delta} \\ &= \sum_{h\in N_j(\delta)} P_{ih}^{k-1}P_{hj} + \frac{1}{\delta}. \end{aligned}$$

We now introduce the shorthand $\gamma_j^{(k)} = \sum_{i=1}^n P_{ij}^k$ for the j-th column sum of P^k and obtain

$$\gamma_j^{(k)} = \sum_{i=1}^n P_{ij}^k \leq \sum_{i=1}^n \sum_{h \in N_j(\delta)} P_{ih}^{k-1} P_{hj} + \frac{n}{\delta}. \tag{7.20}$$

We change the order of summation and compute, for any $h \in N_j(\delta)$,

$$\begin{aligned}\sum_{i=1}^n P_{ih}^{k-1} &= \sum_{i=1}^n \sum_{\ell_1=1}^n \cdots \sum_{\ell_{k-2}=1}^n (d_i^{-1} W_{i\ell_1})(d_{\ell_1}^{-1} W_{\ell_1\ell_2}) \cdots (d_{\ell_{k-2}}^{-1} W_{\ell_{k-2}h}) \\ &= \sum_{i=1}^n \sum_{\ell_1=1}^n \cdots \sum_{\ell_{k-2}=1}^n d_i^{-1} (W_{\ell_1 i} d_{\ell_1}^{-1})(W_{\ell_2\ell_1} d_{\ell_2}^{-1}) \cdots (W_{\ell_{k-2}\ell_{k-3}} d_{\ell_{k-2}}^{-1}) W_{h\ell_{k-2}},\end{aligned}$$

where we use the symmetry of W, and reorganize the products inside the summation. We now upper bound d_i^{-1} with 1, note that $\sum_{i=1}^n W_{\ell i} d_\ell^{-1} = 1$, and change the order of summation so that

$$\begin{aligned}\sum_{i=1}^n P_{ih}^{k-1} &\leq \sum_{\ell_{k-2}=1}^n \cdots \sum_{\ell_1=1}^n \sum_{i=1}^n (W_{\ell_1 i} d_{\ell_1}^{-1})(W_{\ell_2\ell_1} d_{\ell_2}^{-1}) \cdots (W_{\ell_{k-2}\ell_{k-3}} d_{\ell_{k-2}}^{-1}) W_{h\ell_{k-2}} \\ &= \sum_{\ell_{k-2}=1}^n W_{h\ell_{k-2}} = d_h.\end{aligned}$$

We plug this inequality into (7.20) and adopt some additional bounds to obtain

$$\gamma_j^{(k)} \leq \sum_{h \in N_j(\delta)} d_h P_{hj} + \frac{n}{\delta} \leq \Big(\max_{h \in N_j(\delta)} d_h \Big) \gamma_j^{(1)} + \frac{n}{\delta}.$$

Because W is binary, we know that $h \in N_j(\delta)$ implies $d_h \leq \delta$ (see also definition (7.19)). The last inequality becomes:

$$\frac{1}{n}\gamma_j^{(k)} \leq \delta \frac{1}{n}\gamma_j^{(1)} + \frac{1}{\delta}.$$

By selecting $\delta = \left(\frac{1}{n}\gamma_j^{(1)}\right)^{1/2}$, we obtain that each column average of the P^k satisfies

$$\frac{1}{n}\gamma_j^{(k)} \leq 2\Big(\frac{1}{n}\gamma_j^{(1)}\Big)^{1/2}.$$

This inequality immediately implies inequality (7.18) in the theorem statement. □

We are now ready to prove the main result of this section.

Proof of Theorem 7.5.3. We start by showing the equivalence 2 $\iff$ 3. The fact that one-time wisdom (statement 2) implies pre-uniform wisdom (statement 3) for equal-neighbor sequences is an immediate consequence of inequality (7.18) in Lemma 7.5.4. The converse implication 3 $\implies$ 2 is trivially true and stated in Lemma 7.3.1.

Next, we show the equivalence between the absence of prominent individuals (statement 1) and one-time wisdom (statement 2). To do so, we prove the equivalence between the existence of prominent individuals and the lack of one-time wisdom. First, assume there exists a prominent individual, that is, as mentioned in Remark 7.5.2, a sequence of nodes $\{k^{[n]} \in \{1,\dots,n\}\}_{n\in\mathbb{N}}$ such that node $k^{[n]}$ possesses order n neighbors in $G^{[n]}$ with order 1 degree. In other words there exists a constant $\delta > 0$ such that, recalling the definition in equation (7.19), we have $|N_{k^{[n]}}(\delta)|$ is of order n. Then, from the equality (7.4),

$$\begin{aligned}\frac{1}{n}\|P^{[n]}\|_1 &= \frac{1}{n}\max_{j\in\{1,\dots,n\}}\sum_{i=1}^{n} P_{ij}^{[n]} \geq \frac{1}{n}\sum_{i=1}^{n} P_{ik^{[n]}}^{[n]} \\ &= \frac{1}{n}\sum_{i\in N_{k^{[n]}}} d_i^{-1} W_{ik^{[n]}}^{[n]} \geq \frac{1}{n\delta}\sum_{i\in N_{k^{[n]}}(\delta)} W_{ik^{[n]}}^{[n]} \\ &= \frac{1}{n\delta}|N_{k^{[n]}}(\delta)|.\end{aligned}$$

This inequality shows that $\lim_{n\to\infty}\|\frac{1}{n}P^{[n]}\|_1$ cannot vanish and, therefore, the sequence of equal-neighbor matrices fails to be one-time wise.

Second, assume the sequence of equal-neighbor matrices fails to be one-time wise. Then there exist a constant $\alpha > 0$ and a time N such that, for all $n > N$,

$$\frac{1}{n}\max_{j\in\{1,\dots,n\}}\sum_{i=1}^{n} P_{ij}^{[n]} \geq \alpha.$$

In other words, there must exist a sequence of indices $\{k^{[n]} \in \{1,\dots,n\}\}_{n\in\mathbb{N}}$ such that $\sum_{i\in N_{k^{[n]}}} d_i^{-1} W_{ik^{[n]}}^{[n]} > \alpha n$. Because each degree is lower bounded by 1, this inequality implies the existence of a prominent individual. □

7.5.1 The equal-neighbor model over prototypical random graphs

For equal-neighbor models, a sufficient condition to guarantee one-time and pre-uniformly wisdom is:

$$\frac{d_{\max}(n)}{d_{\min}(n)} = o(n), \tag{7.21}$$

where $d_{\max}(n)$ and $d_{\min}(n)$ denote, respectively, the maximum and minimum degree of the graph as a function of the network size n. Indeed, let $k^{[n]}$ be a vertex of the graph and compute:

$$\frac{1}{n}\sum_{i=1}^{n} P_{ik^{[n]}}^{[n]} \leq \frac{1}{n}\sum_{i=1}^{n} \frac{1}{d_{\min}(n)} \leq \frac{1}{n}\frac{d_{\max}(n)}{d_{\min}(n)},$$

as $n \to \infty$. If condition (7.21) holds, then $\frac{1}{n}\sum_{i=1}^{n} P_{ik^{[n]}}^{[n]} \to 0$ as $n \to \infty$ so that there exist no prominent individuals.

In this section we study the Erdös-Rényi and the Barabási-Albert preferential attachment models of random graph and we show that, using condition (7.21), they are both one-time wise with high probability. In contexts where we have a sequence of probability spaces labelled by parameter n (in our case the number of nodes in the graph), the locution with high probability (w.h.p.) means with probability converging to 1 as $n \to +\infty$. In the case of a limit property, as the case of one-time wise, to assert that it holds w.h.p. means that, for every $\varepsilon > 0$, $\mathbb{P}[\|\frac{1}{n}P^{[n]}\|_1 < \varepsilon]$ converges to 1 for $n \to +\infty$.

Example 7.5.6 (The equal-neighbor model over Erdös-Rényi graphs)**.** *An Erdös-Rényi graph $G(n,p)$ is a graph with n vertices and with each possible edge having, independently, probability p of existing [7]. We focus on the case when $p = c\log(n)/n$ with $c > 1$. In this regime $G(n,p)$ is known to be connected and aperiodic w.h.p.. Moreover, [6, Lemma 6.5.2] implies that there exists a constant $b > 0$ such that $b\log n \leq d_i \leq 4c\log n$ for every node i w.h.p.. This bound immediately implies that condition* (7.21) *holds w.h.p. and thus the Erdös-Rényi model is one-time wise (and pre-uniformly wise and wise) w.h.p.. Using the fact that [6] $\tau_{\text{mix}}(P^{[n]}) = O(\log(n))$ w.h.p., we now prove that this model is also uniformly wise. Indeed, w.h.p.*

$$\begin{aligned}\left\|\tfrac{1}{n}(P^{[n]})^k\right\|_1 \tau_{\text{mix}}(P^{[n]}) &\leq 2\left\|\tfrac{1}{n}P^{[n]}\right\|_1^{1/2} \tau_{\text{mix}}(P^{[n]}) \\ &= O\left(\frac{1}{n^{\frac{1}{2}}}\log n\right), \quad \text{as } n \to \infty,\end{aligned}$$

where the first inequality follow from equation (7.18). □

We now present the preferential attachment model. Also in this case, it is also possible to prove that the equal-neighbor models is one time wise and uniformly wise w.h.p..

Example 7.5.7 (The equal-neighbor model over preferential attachment graphs)**.** *The Barabási-Albert preferential attachment model is a random graph generation model described as follows: vertices are added sequentially to the graph, new vertices are connected to a fixed number of earlier vertices, that are selected with probabilities proportional to their degrees. Specifically, assume a fixed number m_0 of initial vertices is given and, at every step, a new vertex is added and m ($m \leq m_0$) new edges are added, whereby the new vertex is connected to a prior node i with a probability proportional to the degree d_i of i, that is $d_i / \sum_j d_j$. After t time steps, the model leads to a random graph with $t + m_0$ vertices and mt edges. It is known [2, 3] that the degree distribution follows a power-law, that the minimum degree is $d_{\min}(n) = m$ (by construction), and that the maximum degree is of order $d_{\max}(n) \in \Theta(\sqrt{n})$ w.h.p..*

The equal-neighbor Barabási-Albert model is one-time wise (and, therefore, also pre-uniformly wise and wise) w.h.p.. This follows again by checking condition (7.21)*:*

$$\frac{1}{n}\frac{d_{\max}(n)}{d_{\min}(n)} = O\left(\frac{1}{n}\frac{\sqrt{n}}{m}\right) = O\left(\frac{1}{m\sqrt{n}}\right), \quad \text{as } n \to \infty.$$

Moreover, the equal-neighbor Barabási-Albert model is uniformly wise. Indeed, recall from [1] that the mixing time of the Barabási-Albert model is w.h.p. $\tau_{\text{mix}}(P^{[n]}) = O(\log(n))$, so that w.h.p.

$$\begin{aligned}\left\|\tfrac{1}{n}(P^{[n]})^k\right\|_1 \tau_{\text{mix}}(P^{[n]}) &\leq 2\left\|\tfrac{1}{n}P^{[n]}\right\|_1^{1/2} \tau_{\text{mix}}(P^{[n]}) \\ &= O\left(\frac{1}{n^{1/4}}\log n\right), \quad \text{as } n \to \infty,\end{aligned}$$

where the first inequality follows from inequality (7.18)*.*

Finally, we consider a super-linear preferential attachment model. Notice that the Barabási-Albert model is a linear preferential attachment in the sense that the probability of choosing a node in the network is linear in the degree of the nodes. If we consider a super-linear model *with a probability of the form $\sim x^p$ with $p > 1$, then it is known [13] that there exists, w.h.p., a node with degree of order n, while all*

other nodes have finite degrees. It follows that a sequence of prominent individuals exists in large populations and that, by Theorem 7.5.3, the super-linear preferential attachment model is neither wise nor finite-time wise. □

7.6 Conclusions

This work furthers the study of learning phenomena and influence systems in large populations. Our results provide an alternative and, arguably, a bit more realistic characterization of wise populations in terms of the absence of prominently influential individuals and groups. Future work includes extending these concepts to influence systems with time-varying and concept-dependent interpersonal weights and to other opinion dynamic models.

References

[1] ACEMOGLU, D., COMO, G., FAGNANI, F., AND OZDAGLAR, A. Opinion fluctuations and disagreement in social networks. *Mathematics of Operation Research 38*, 1 (2013), 1–27.

[2] BARABÁSI, A.-L., AND ALBERT, R. Emergence of scaling in random networks. 509–512.

[3] BOLLOBÁS, B., RIORDAN, O., SPENCER, J., AND TUSNÁDY, G. The degree sequence of a scale-free random graph process. *Random Structures & Algorithms 18*, 3 (2001), 279–290.

[4] BULLO, F. Lectures on network systems. *Version 0.86, November* (2016).

[5] DEGROOT, M. H. Reaching a consensus. *Journal of the American Statistical Association 69*, 345 (1974), 118–121.

[6] DURRETT, R. *Random Graph Dynamics*. 2006.

[7] ERDÖS, P., AND RÉNYI, A. On the evolution of random graphs. *Publication of the Mathematical Institute of the Hungarian Academy of Science 5*, 1 (1960), 17–60.

[8] FAGNANI, F. Consensus dynamics over networks. Lecture notes for Winter School on Complex Networks, INRIA. Downloaded on 12/23/2016, Jan. 2014.

[9] FRENCH, J. R. P. A formal theory of social power. *Psychological Review 63*, 3 (1956), 181–194.

[10] GALTON, F. Vox populi. 450–451.

[11] GOLUB, B., AND JACKSON, M. O. Naive learning in social networks and the wisdom of crowds. *American Economic Journal: Microeconomics 2*, 1 (2010), 112–149.

[12] LEVIN, D. A., PERES, Y., AND WILMER, E. L. *Markov chains and mixing times*. American Mathematical Soc., 2009.

[13] OLIVEIRA, R., AND SPENCER, J. Connectivity transitions in networks with super-linear preferential attachment. *Internet Mathematics 2*, 2 (2005), 121–163.

[14] PRUITT, W. E. Summability of independent random variables. *Journal of Mathematics and Mechanics 15* (1966), 769–776.

[15] SUROWIECKI, J. *The Wisdom Of Crowds*. Anchor, 2004.

Conclusions

In this thesis we studied two cooperative networks for multi agents systems. The first one is a decentralized cloud storage model for which we give an algorithm that ensure the complete storage of the data of the users through the use of evolutionary game theory. For this algorithm we prove its convergence properties and study some variants including the reciprocity process and the possibility to allocate multiple copies of the data. On the other hand, the second part of this thesis deals with the definition of new concepts of wisdom of crowds: finite-time wisdom and uniform wisdom. For this notions we give a characterization and a wide set of examples that corroborate the theoretical analysis.

In the next to paragraphs we emphasize on our results and we present some further steps in both the topics.

The decentralized cloud storage model. In this thesis, we have analyzed an allocation algorithm which allows a network of units (computers, smartphones or any other device with storage capabilities) to use the memory space available in their neighbors to store a back up of their data. The interest for such an algorithm comes from the goal of creating a peer-to-peer decentralized cloud storage model to be thought as an alternative to classical centralized cloud storage services.

We have proposed, in a game theoretic framework, a peer-to-peer decentralized storage model where a network of units are, at the same time, end users needing to allocate externally a back up of their data, as well storage resources for other users. We have proposed a novel fully distributed algorithm where units, connected through a network, activate autonomously at random time and either allocate or move pieces of their data among the neighboring available resources. Actions taken by the units are noisy best response actions with respect to utility functions which incorporate

the congestion of the resources, their reliability, as well possibly, the fragmentation of the stored data.

The proposed algorithm is based on evolutionary game theory. The main theoretical contribution has been to prove convergence and to give en efficient characterization of the asymptotic in terms of an explicit invariant probability distribution picked on optimal Nash equilibria.

We have presented and mathematically analyzed a decentralized allocation algorithm, motivated by the recent interest in cooperative cloud storage models. We have proved convergence and we have shown the practical implementability of the algorithm. The tuning of its parameters to optimize performance will be considered elsewhere. In this direction, it will also be useful to investigate the possibility to use different utility functions in the definition of the algorithm, following the ideas in [4] and [6].

We have also carried on a number of simulations showing the good scalability properties of the algorithm, its reduced complexity and illustrating the nice features of the final allocation state.

Moreover we find explicit bounds on the convergence time of the algorithm and extend to the case when multiple back up copies are needed to be stored with the further security constraint of using different resources for different copies.

We believe that there are several challenges related to the peer-to-peer storage model which have not yet been satisfactorily addressed by pure mathematical model. Among these:

- The structure of the Nash equilibria for general topologies is not completely clear; a more deepened study is needed.
- Units, in our model, are completely anonymous and resources do not make any filter on new allocation requests. More interesting models should incorporate trust formation mechanisms where more trusted units (when thought as resources) should have a vantage in finding place to store their data.
- In our approach the quantities to be allocated are fixed but a reasonable extension should be to include time variable amounts of data. Clearly if this is the case, the allocation condition must remain satisfied.

- Another aspect that worth the study is to consider a time varying graph: for instance, it can happen that users decide to explore the graph to choose more reliable resources. In this case, it would be of interest to answer the question of how the algorithm would behave.

- A natural extension of our analysis in this game theory setting is to study the price of anarchy and the price of stability to ensure that the solution is close to the global welfare. These quantities are widely studied in literature [3, 7] and represent a way to measure the goodness of an algorithm.

- Our approach is strictly case dependent but the evolutionary game theory approach is quite general. To make the algorithm more valuable different utility function or different case study should be studied and analyzed.

- Regarding our case-dependent approach, it would also worth studying a more general approach for instance including this work in the asset of generalized Nash equilibrium problems [2]. This class of games concern constrained problems for which neither a classical game theoretic approach nor an optimization algorithm are easily applicable. We believe that our noisy best response dynamic can be an answer to this kind of problems.

Wisdom of crowds. This work furthers the study of learning phenomena and influence systems in large populations. We introduce and characterize new notions of wisdom in finite time and uniformly in time. Moreover, our results provide an alternative and, arguably, a bit more realistic characterization of wise populations in terms of the absence of prominently influential individuals and groups. Future work includes extending these concepts to influence systems with time-varying and concept-dependent interpersonal weights and to other opinion dynamic models. To this aim, we already start the analysis of the Friedkin-Johnsen model that clearly needs a more accurate characterization. Moreover there are some other steps to be done, both in the analysis and in the extension of this concepts. For instance:

- Complete the characterization of the notions of finite time wise and uniformly wise for direct equal neighbor model and find more realistic examples in this direction.

- Since there can be found in literature [1, 5] ways to influence the PageRank in a web network it would be of interest to understand under which conditions this influence affect also the wisdom of a crowd (both in finite time and in its asymptotics).

References

[1] DE KERCHOVE, C., NINOVE, L., AND VAN DOOREN, P. Maximizing pagerank via outlinks. *Linear Algebra and its Applications 429*, 5-6 (2008), 1254–1276.

[2] FACCHINEI, F., AND KANZOW, C. Generalized nash equilibrium problems. *4OR: A Quarterly Journal of Operations Research 5*, 3 (2007), 173–210.

[3] KOUTSOUPIAS, E., AND PAPADIMITRIOU, C. Worst-case equilibria. *Computer science review 3*, 2 (2009), 65–69.

[4] LI, N., AND MARDEN, J. R. Designing games for distributed optimization. *IEEE Journal of Selected Topics in Signal Processing 7*, 2 (2013), 230–242.

[5] LITVAK, N., AND AVRACHENKOV, K. *The effect of new links on google pagerank*. PhD thesis, INRIA, 2004.

[6] MARDEN, J. R., AND WIERMAN, A. Distributed welfare games. *Operations Research 61*, 1 (2013), 155–168.

[7] NISAN, N., ROUGHGARDEN, T., TARDOS, E., AND VAZIRANI, V. V. *Algorithmic game theory*, vol. 1. Cambridge University Press Cambridge, 2007.