

Universidad ORT Uruguay
Facultad de Ingeniería

FROM RESOURCE ALLOCATION TO
NEIGHBOR SELECTION IN
PEER-TO-PEER NETWORKS

Entregado como requisito para la obtención del título de Master en Ingeniería

Martín Zubeldía #143253
Tutor: Fernando Paganini
Tutor: Andrés Ferragut

2014

DECLARACIÓN DE AUTORÍA

Yo, Martín Zubeldía, declaro que el trabajo que se presenta en esta obra es de mi propia mano. Puedo asegurar que:

- La obra fue producida en su totalidad mientras realizaba el Master en Ingeniería;
- Cuando he consultado el trabajo publicado por otros, lo he hecho con claridad;
- Cuando he citado obras de otros, he indicado las fuentes. Con excepción de estas citas, la obra es enteramente mía;
- En la obra, he acusado recibo de las ayudas recibidas;
- Cuando la obra se basa en trabajo realizado conjuntamente con otros, he explicado claramente qué fue contribuido por otros, y qué fue contribuido por mi;
- Ninguna parte de este trabajo ha sido publicada previamente a su entrega, excepto donde se han realizado las aclaraciones correspondientes.



Martín Zubeldía
25 de julio de 2014

AGRADECIMIENTOS

Primero que nada, quiero agradecerle a mis tutores Fernando y Andrés por guiarme, no sólo a través de este proyecto de tesis, sino en los últimos cuatro años que he tenido el privilegio de estar en el grupo MATE bajo su tutela. Tengo mucho que agradecerle a los dos. En particular le agradezco a Fernando haber confiado en mí y ofrecerme que me uniera al grupo y por todo lo que he aprendido de él en éstos años, él y Andrés me han contagiado su entusiasmo por la investigación teórica en ingeniería y me han introducido al mundo de la academia. También les agradezco a los dos por tenerme paciencia y haberme apoyado mientras continuaba mis estudios de matemática en la Universidad de la República.

También quiero agradecerle a Patricia Corbo, por el voto de confianza para darme la oportunidad de estudiar becado, que me llevó a tener los excelentes tutores que tuve.

Por último pero no menos importante, quiero agradecerle a mis padres que siempre me apoyaron para que siguiera estudiando y persiguiendo mis sueños.

RESUMEN

En las redes peer-to-peer (P2P) hay dos grandes problemas: la eficiencia en el uso del ancho de banda que contribuye cada peer, y los incentivos que la asignación global de recursos alcanzada provee a cada uno de ellos. Estos incentivos deben promover la cooperación de los usuarios, de manera de que estos contribuyan ancho de banda a la red, lo cual es la clave de la escalabilidad de la arquitectura. Existen algunas soluciones ad-hoc a estos problemas, como el popular protocolo BitTorrent, el cual constituye una primera aproximación al problema. En la tesis se estudia, primero desde un punto de vista teórico, las diferentes posibilidades de asignación de recursos en estas redes, y los incentivos que las mismas proveen a los peers. Luego, se realiza el diseño de un algoritmo que alcanza la asignación deseada manteniendo los incentivos para los peers a contribuir.

Analizando los incentivos aparece un compromiso entre eficiencia y justicia en la red. Se argumenta que una asignación proporcional es la más adecuada a estas redes y se analizan diferentes alternativas para alcanzarla. Sin embargo, las alternativas existentes presentan varios problemas a la hora de la implementación en redes actuales.

Se procede entonces a diseñar un algoritmo descentralizado de “selección de vecinos” (donde se elige con quién compartir contenido de manera de alcanzar un óptimo global). El algoritmo se basa en el uso de Cadenas de Markov de tiempo continuo que aparecen en el estudio de la mecánica estadística, en particular las distribuciones de Gibbs. El algoritmo consiste en un Gibbs Sampler, que alcanza la asignación deseada manteniendo sencillez en la implementación.

En la última parte de la tesis se extienden las propuestas al contexto de redes inalámbricas ad-hoc, en las cuales el compromiso de eficiencia y justicia cambia radicalmente debido a que la eficiencia de la red está asociada a qué vecinos podemos elegir para comunicar, ya que en las redes inalámbricas las restricciones de capacidad se vuelven par a par, en lugar de una única restricción de subida por peer. Las interferencias entre enlaces debido la comunicación inalámbrica complican aun más el problema. De todos modos, se propone una extensión al algoritmo que logra los objetivos deseados también en este tipo de redes, y que permite modular el compromiso entre eficiencia y justicia satisfactoriamente.

Palabras clave: Redes peer-to-peer; asignación de recursos; muestreo de Gibbs.

ABSTRACT

In peer-to-peer networks there are two main issues: the efficiency in the use of the upload bandwidth of the peers; and the incentives that the resource allocation provides to the peers, which should promote the upload of content which is key to its scalability. There are some ad-hoc solutions to these issues such as the popular BitTorrent protocol. In this thesis we pretend to study, first from a theoretical standpoint, the different choices for the resource allocation and their incentives for the peers. Then, we intend to design an algorithm that achieves a desired allocation that provides the proper incentives for the peers to contribute to the network.

Analyzing the incentives we note that there is a tradeoff between efficiency and fairness in the network and although a max-min allocation would impose minimal incentives, we argue that a “proportional allocation” is better suited for these networks. With that in mind, we propose a decentralized neighbor selection algorithm, based on a Gibbs sampler, which achieves the desired allocation while being easy to implement. Lastly, we turn our attention to ad-hoc wireless networks and analyze the choices for the resource allocation in this setting, which are no longer just the superposition of the allocation of every peer, as the interferences given by the shared channel complicate the analysis. Furthermore, we propose a new decentralized algorithm that achieves a proportional allocation for this type of networks.

Key words: Peer-to-peer networks; resource allocation; Gibbs sampler.

Content

1	Introduction	8
1.1	Peer-to-peer networks	8
1.2	Outline of the thesis	11
2	Resource allocation in wired P2P networks	13
2.1	Fluid model	13
2.2	Analysis of resource allocation choices	15
2.3	Introducing incentives	21
2.4	Proportional fairness	24
2.5	Conclusions	25
3	Proportional reciprocity and the Sinkhorn iteration	26
3.1	Resource allocation model	26
3.2	Original Sinkhorn approach	27
3.3	Applications to resource allocation in P2P networks	31
4	Neighbor selection for wired P2P networks	33
4.1	Implementation restrictions: discrete connections	34
4.2	Energy driven allocations	34

4.3	Deterministic approach	38
4.4	Introducing randomness	40
4.5	Simulations	48
4.6	Conclusions	51
5	Extension to wireless P2P networks	52
5.1	Network setting	53
5.2	Rate allocations	55
5.3	Maximizing efficiency	59
5.4	Decentralized algorithm for reciprocity	62
5.5	Simulations	70
5.6	Conclusions	75
6	Conclusions and future work	76
	Bibliography	78
	Appendices	
A.	Game theoretic approach to incentives	81
B.	Population dynamics with arbitrary upload capacities	86
C.	Markov chains	94
D.	Graph theory	99

Chapter 1

Introduction

In this chapter we begin by introducing some concepts about peer-to-peer networks and the peculiarities of content distribution. Then, we present the problem of resource allocation in this kind of networks and give an outline of the thesis.

1.1 PEER-TO-PEER NETWORKS

Traditionally, in order to distribute a file over the Internet one would have multiple servers which had the file, and the interested users would connect to them as clients to download it. In this case we have two distinct type of nodes in the network: servers and clients, and the information only flows from servers to clients (Figure 1.1(a)). The main drawback of this kind of network is that it does not scale with the number of interested clients, because if the number of servers remains fixed, the load on the servers increases with the number of clients.

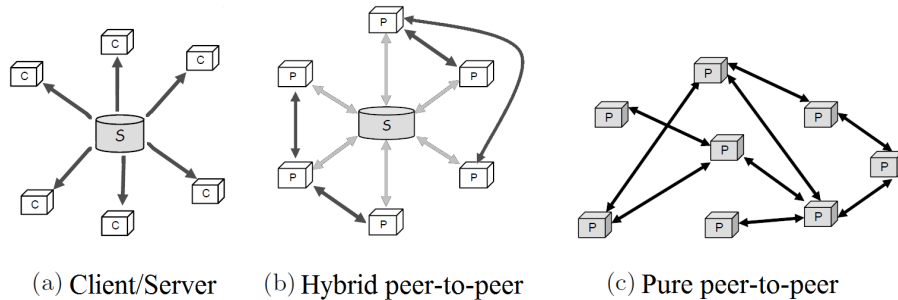


Figure 1.1. Different network configurations for the distribution of content.

To address this scalability problem in content distribution, a new kind of network arose, the peer-to-peer (P2P) network. In these networks, each node (peer) has the ability to download content from other peers and the responsibility of uploading content to others as well. There are two types of P2P networks: hybrid and pure (Figure 1.1(b,c)). The pure P2P networks are completely self organized and do not

rely in any centralized node or algorithm. On the other hand, in the hybrid networks there are still central servers that carry certain tasks, such as keeping track of the location of all peers with a determined file and sharing that information with new peers. We will focus our attention on hybrid P2P networks where the only task of the servers is to keep track of the peers, in order to provide this information to the new peers, all the actual file sharing in between peers.

1.1.1 Mechanisms to distribute content

In order to distribute a file throughout the network efficiently, it is divided into small pieces (or chunks) that are sent from peers who have them to others who do not. This enables peers to download different pieces from different peers at the same time, making a better use of network bandwidth, which is usually the scarce resource. The peers that are still downloading the file are called *leechers*, and the ones that already have the whole file are called *seeders*. Seeders play the role of servers, that have the whole file and can upload it to other peers. However, in order to have true scalability, the leechers also need to contribute to the network, uploading content to other leechers. As a result, it is essential that any protocol establishes incentives for the leechers to contribute in order to avoid free-riders, because if many peers decide not to upload content to the network, the burden of uploading the content to everyone will fall on the seeders and this will defeat the purpose of using P2P networks.

In every P2P system, there should be a mechanism that helps peers connect to each other. For example, in the BitTorrent protocol [1], there are centralized entities called *trackers* that have the location of all peers which have or are downloading a particular file. When a new peer arrives to the network, a tracker provides a random subset of peers that are interested in the same content. This builds a *neighbor overlay graph* for each peer which will impact the final resource allocation in the network by establishing who can connect to whom.

When a peer is finally connected to other interested peers, it starts making requests for desired pieces of the file to the connected peers. In order to decide which piece to request, one could easily start asking for the pieces in order, but this is far from efficient (each peer could only get new pieces from peers who downloaded more content than them). If we are downloading a file, the order of the pieces that we download is irrelevant, and we can choose to download them in any order that we see fit. For example, the popular BitTorrent protocol uses the *rarest first* rule, in which the peer asks for the piece that sees as the least popular. This maximizes the minimum amount of users that have a piece, promoting diversity and preventing the *missing piece syndrome* (a phenomenon studied in [2]).

Lastly, when a peer is being asked to upload pieces to several leechers, it has to decide to which peers to upload to (which peers to *unchoke*). This part is handled by the *neighbor selection algorithm*, and it is the main responsible for determining

the resource allocation in the network and imposing incentives to contribute.

1.1.2 Resource allocation problem

The main purpose of using P2P networks, as we said before, is its scalability, which depends heavily on the cooperation of other leechers that are downloading the same file. In order to ensure this, any protocol must set the proper incentives in place which can only be done effectively through the neighbor selection algorithm. This algorithm should be simple enough for implementation and must perform two tasks at the same time:

1. Provide incentives to other peers to upload content to others.
2. Discover new peering options.

For example, BitTorrent peers open a maximum amount of connections to other peers (usually four). Using TCP (Transport Control Protocol) connections, there is no control over the rates between the peers, but instead they depend on the RTTs (Round Trip Times) between them and on the bottlenecks of the network. Under normal circumstances (bottleneck in the upload and uniform RTTs) this leads to a uniform split of the upload bandwidths. For the sake of simplicity, we will suppose that we obtain a uniform split of the bandwidth between the selected neighbors. That being said, BitTorrent's neighbor selection algorithm has essentially two parts:

- The tit-for-tat part: in which each peer, every 10s, decides to upload to the three peers from which it downloaded the most in the last 20s.
- The optimistic unchoke: in which each peer, every 30s, opens a connection to a random peer for 30s.

The result is that every peer is at any time only connected to 4 peers at most, 3 of them that are chosen based on a ranking of the received bandwidths and the other one at random. This tit-for-tat reciprocity gives some incentive to the leechers to contribute to the network in order to increase the chance of getting pieces from other peers, but still has room for free riders.

The tit-for-tat part of the BitTorrent's neighbor selection algorithm tries to achieve the first objective but it falls short. In game theory, the name "tit-for-tat" usually means "equivalent retaliation" and it is a very effective strategy for the game of the repeated prisoner's dilemma. In the case in which all peers have the same upload capacity then they would all be equal in the game and this scheme would be appropriate. However, in the realistic scenario in which every peer has a different upload capacity, the players have different power in the game and instead

of trying to find another peer that just cooperates, the players want to find the peers that give them *more*. In this setting the game is similar to an auction, in which each player has a certain buying power (its upload capacity) and wants to obtain upload bandwidth from other peers in return, as seen in [3]. In an auction you do not have an incentive to contribute more, just to outbid the others, and if you cannot outbid them then you do not have any incentive to bid at all. This is one of the problems with BitTorrent's neighbor selection algorithm, as one should always have incentives to contribute more.

The other part of the algorithm is the *optimistic unchoke*, which uploads content to other peer at random. This is very easy to implement and performs the second task (discovering new peering options) impeccably. However, this part of the algorithm goes against the incentives that the tit-for-tat part was trying to achieve. When selecting a peer at random regardless of any other measure or situation, we are giving away a fraction of the upload capacity of the network without asking for anything in return. This enables free-riders in the network that can choose to upload nothing and still get a fraction of the upload capacity for free. This is the other problem of the BitTorrent's neighbor selection algorithm.

One of the limitations of this tit-for-tat scheme (or any reciprocity based scheme) is that the new peers with zero pieces cannot contribute to the network yet, and will not be selected for upload by the algorithm. In order to avoid this pitfall, there is a mechanism called *bootstrapping* that assigns a higher probability to the new peers for being selected at random until they have enough pieces to start contributing.

1.2 OUTLINE OF THE THESIS

The objective of this thesis is to analyze the resource allocation problem in P2P overlays and then propose simple decentralized algorithms that achieve a desired allocation of resources, first for a wired network and then for an ad-hoc wireless network.

Here is a brief outline of this thesis: In Chapter 2, we will present a fluid model for the leecher and seeder populations for a P2P overlay over a wired network and analyze the possible resource allocation choices, taking into account the average download time of the files and the incentives. A first contribution is showing that the resource allocation that minimizes the average download time and provides minimum incentives is the max-min allocation. However, we argue that a proportional allocation is better suited for this kind of networks as it provides stronger incentives. In Chapter 3 we present preliminary results and theory on a matrix renormalization algorithm, as a means to achieve proportionality. Then, in Chapter 4 we will develop a new decentralized neighbor selection algorithm that achieves the desired allocation of resources over a wired network, and it is shown to perform better than other alternatives through simulations. Finally, in Chapter 5 we focus on P2P overlays

over an ad-hoc wireless network and we develop a decentralized neighbor selection algorithm specially designed for this kind of networks.

Furthermore, in Appendices A and B we formalize and extend the results presented in Chapter 2. Lastly, in Appendices C and D we give some background on Markov chains and on graph theory respectively.

Chapter 2

Resource allocation in wired P2P networks

Consider a scenario where we have a set of peers with upload capacities $\mu_i, i = 1, \dots, N$; assume there are no other bottlenecks in the network, and enough diversity of pieces so that the total bandwidth $\sum_i \mu_i$ can be used for download; how should it be distributed among the same peers, now seen as clients?

This question has been addressed by many researchers; often, the discussion is combined with efforts to characterize the behavior of prevailing P2P protocols such as BitTorrent. In this chapter we are going to present a fluid model for the leecher and seeder population for a general bandwidth allocation. Then we will analyze the properties of the equilibrium looking for the allocation that minimizes the average download time, first with no restrictions and then imposing incentives in the allocation, finding the one that minimizes the average download time while maintaining proper incentives for the peers. Lastly we show that a proportional allocation provides stronger incentives to contribute and we argue that it is the best candidate for the resource allocation.

For further discussion of incentives, including resistance to attacks, we refer to [3]. See also [4, 5] for related game-theoretic studies.

2.1 FLUID MODEL

We present a fluid model of the population dynamics for a wired P2P network. First, we introduce some notation.

- $\mu_1 > \mu_2 > \dots > \mu_n > 0$ are the possible upload rates for the peers.

- $d_1 > d_2 > \dots > d_n > 0$ are the download capacities for the peers.
- $x_i(t)$ and $y_i(t)$ are the amount of leechers and seeders in the system at time t with upload rate μ_i .
- λ_i is the intensity of arrivals of peers with upload rate μ_i .
- $\frac{1}{\gamma_i}$ is the mean time that a seeder with upload capacity μ_i stays in the system, hence $y_i(t)\gamma_i$ is the rate which seeders with upload rate μ_i leave the system at time t .
- $r_i(x(t), y(t))$ is the download rate that a peer of upload capacity μ_i receives, with $x(t) = (x_1(t), \dots, x_n(t))$, $y(t) = (y_1(t), \dots, y_n(t))$.

We will assume that the size of the file is equal to 1 and that all the leechers that finish downloading, stay as seeders for some time.

Using the aforementioned notation, we define the dynamics of the seeders and leechers

$$\dot{x}_i(t) = \lambda_i - r_i(x(t), y(t))x_i(t) \quad \forall i \quad (2.1)$$

$$\dot{y}_i(t) = r_i(x(t), y(t))x_i(t) - \gamma_i y_i(t) \quad \forall i \quad (2.2)$$

with the “conservation of mass” restriction

$$\sum_{i=1}^n r_i(x(t), y(t))x_i(t) \leq \sum_{i=1}^n \mu_i [x_i(t) + y_i(t)] \quad (2.3)$$

which states that the total download throughput of the network must be less than or equal to the total upload capacity of the network. We will assume that there is strict inequality only when the system is saturated by the download capacities d_i . We want to characterize the equilibria of these dynamics.

Proposition 2.1.1. *If r_i are continuous with respect to x , bounded by above by the download capacity d_i and uniformly bounded by below by $\epsilon > 0$, then the previous dynamics defined by 2.1 and 2.2 have equilibria in x^* , y^* such that*

$$y_i^* = \frac{\lambda_i}{\gamma_i}$$

and x^* are fixed points of the equations

$$x_i^* r_i(x^*, y^*) = \lambda_i$$

Proof. In the equilibrium, the derivatives with respect to time must be all zero

$$0 = \lambda_i - r_i(x^*, y^*)x_i^*(t)$$

$$0 = r_i(x^*, y^*)x_i^* - \gamma_i y_i^*$$

Then for the seeders we have that

$$\lambda_i = \gamma_i y_i^*$$

Then

$$y_i^* = \frac{\lambda_i}{\gamma_i}$$

For the leechers, the equilibria (if they exist) have to satisfy

$$\lambda_i = r_i(x^*, y^*) x_i^*$$

or equivalently

$$x_i^* = \frac{\lambda_i}{r_i(x^*, y^*)}$$

In order to prove the existence of the equilibria, consider the function

$$F : I \rightarrow I / F(x)_i = \frac{\lambda_i}{r_i(x, y^*)}$$

where I is the convex compact set $\prod_{i=1}^n \left[0, \frac{\lambda_i}{\epsilon}\right]$. Note that F is continuous because the r_i are continuous and that it is well defined due to the fact that $r_i(x, y) \geq \epsilon \forall i$ and thus $\frac{\lambda_i}{r_i(x, y)} \leq \frac{\lambda_i}{\epsilon}$. As a result, F is a continuous function over a convex compact set onto itself, and hence by the Brouwer fixed point theorem, it has at least one fixed point. \square

From now on, we will focus on analyzing the equilibrium for these dynamics for different rate functions r_i .

2.2 ANALYSIS OF RESOURCE ALLOCATION CHOICES

Now we turn our attention to the design problem of selecting the proper rate functions r_i . At this point, we are going to suppose that there is a centralized entity that takes all the upload rate and redirects it according to a desired allocation. The problem of implementing the allocation in a decentralized manner will be postponed.

A first objective could be to minimize the average download time of a leecher regardless of the resulting incentives for the peers. Then we introduce the proper incentives and we find the resource allocation that minimizes the download time with these constraints. This part generalizes the work in [6], adding seeders and an analysis of the incentives of the resource allocations. In Appendix B we give a further generalization for the case where the upload capacities can take any positive value instead of just a finite set.

2.2.1 Minimizing the download time

In equilibrium, the average download time of a peer in the network with upload capacity μ_i would be

$$\bar{T}_i = \frac{1}{r_i^*}$$

where $r_i^* = r_i(x^*, y^*)$, as the file has size 1 and all the peers with upload capacity μ_i download at rate r_i^* . Using the fact that the fraction of leechers that arrive to the network with upload capacity μ_i is $\frac{\lambda_i}{\Lambda}$ (where $\Lambda = \sum_i \lambda_i$), we get that the average download time of a generic leecher is

$$\bar{T} = \sum_{i=1}^n \frac{\lambda_i}{\Lambda} \frac{1}{r_i^*}$$

Using the value of x_i^* in equilibrium, we can rewrite the average download time as

$$\bar{T} = \frac{1}{\Lambda} \sum_{i=1}^n x_i^*$$

which is essentially Little's law [7], a result of queueing theory which states that the average time that a customer stays in the system multiplied by the average rate of arrivals is equal to the average number of customers in the system. As a result, we can minimize the total number of leechers in equilibrium in order to minimize the average download time. The first constraint that we have is the one given by the "conservation of mass" (Equation (2.3)), which in equilibrium is equivalent to

$$\sum_{i=1}^n \mu_i x_i^* + \sum_{i=1}^n \mu_i \frac{\lambda_i}{\gamma_i} \geq \Lambda$$

This means that the total upload capacity of the leechers plus the upload capacity of the seeders (all measured in files per second) must be greater or equal to the total incoming leecher rate Λ . Let

$$C_y = \sum_{i=1}^n \mu_i \frac{\lambda_i}{\gamma_i}$$

be the total upload capacity of the seeders. Then, the previous restriction can be rewritten as

$$\sum_{i=1}^n \mu_i x_i^* \geq \Lambda - C_y$$

The last constraint that we will impose is that a peer with upload capacity μ_i has a maximum download capacity d_i , where $d_1 > d_2 > \dots > d_n$. That is

$$r_i(x, y) \leq d_i \quad \forall i \quad (2.4)$$

We can translate this into a restriction in x^*

$$x_i^* \geq \frac{\lambda_i}{d_i} \quad \forall i$$

Using equation (2.4) in equilibrium, we can now write the optimization problem of minimizing the average download time as

Problem 1

$$\begin{aligned} \inf \quad & \sum_{i=1}^n x_i^* \\ \text{s.t.} \quad & \sum_{i=1}^n \mu_i x_i^* \geq \Lambda - C_y \end{aligned} \tag{2.5}$$

$$x_i^* \geq \frac{\lambda_i}{d_i} \quad \forall i \tag{2.6}$$

Note that this is a linear programming problem whose constraints determine a polytope and thus its solution will be in the vertex of the polytope that is closest to the origin in the 1-norm. For different sets of parameters, we will have different sets of active restrictions, and this will actually have an interpretation from the network's viewpoint. The solution of this problem will depend on the rate of incoming leechers Λ and on the total upload capacity of the population of seeders in equilibrium C_y .

Case sustained by seeders

Note that if $C_y \geq \Lambda$, i.e. when the total number of peers that arrive per second is less or equal to the total upload capacity of the seeders (all measured in files per second) then the first restriction in the optimization (Equation (2.5)) is trivial, which means that the upload capacity of the leechers is not necessary and the system can be sustained by the seeders. As a result, the minimum is achieved when we have equality in (2.6)

$$x_i^* = \frac{\lambda_i}{d_i} \quad \forall i$$

which is the case of $r_i = d_i \forall i$. This is the minimum amount of leechers that we can possible have in the network, as everyone is saturated by their download capacity and stays the minimum amount of time possible.

Remark 2.2.1. If we forget about the restrictions on the download capacity in this case, then the minimum would be achieved when $x_i^* = 0 \forall i$, which is not a reasonable equilibrium as it would mean that $r_i^* = +\infty$.

Case sustained by seeders + leechers

Now we analyze the case in which a population of leechers that contribute to the network is actually needed in order to reach an equilibrium, because $C_y < \Lambda$.

Using equation (2.6), we have that

$$\sum_{i=1}^n \mu_i x_i^* \geq \sum_{i=1}^n \frac{\mu_i \lambda_i}{d_i} =: D$$

When $D \geq \Lambda - C_y$, restriction (2.5) always hold and then the system is still saturated by the download capacity. As a result, we achieve the same minimum for the optimization problem as in the previous case. This is summarized in the following proposition.

Proposition 2.2.1. *If the average arrival rate of leechers is less than or equal to the upload capacity of the population of seeders plus the upload capacity of the minimum amount of leechers that we could have, i.e.*

$$\Lambda \leq D + C_y$$

then the solution of the Problem 1 is achieved when all peers are saturated by download capacity, i.e.

$$x_i^* = \frac{\lambda_i}{d_i} \quad \forall i$$

Although we have the same result, now the upload capacity of the leechers is actually needed in order to cope with the arrival rate Λ . In both cases, the average download time for the leechers is minimum:

$$\bar{T} = \sum_{i=1}^n \frac{\lambda_i}{\Lambda} \frac{1}{d_i}$$

In order to see more clearly how the constraints interact, it is convenient to consider the case in which we have only two upload capacities, where we can see the result looking at the feasible set in \mathbb{R}^2 as in Figure 2.1. The region defined by constraints on the download capacity (2.6) is completely contained in the half-plane that defines the conservation of mass (2.5), and that is why the minimum is achieved when we have equality in (2.6).

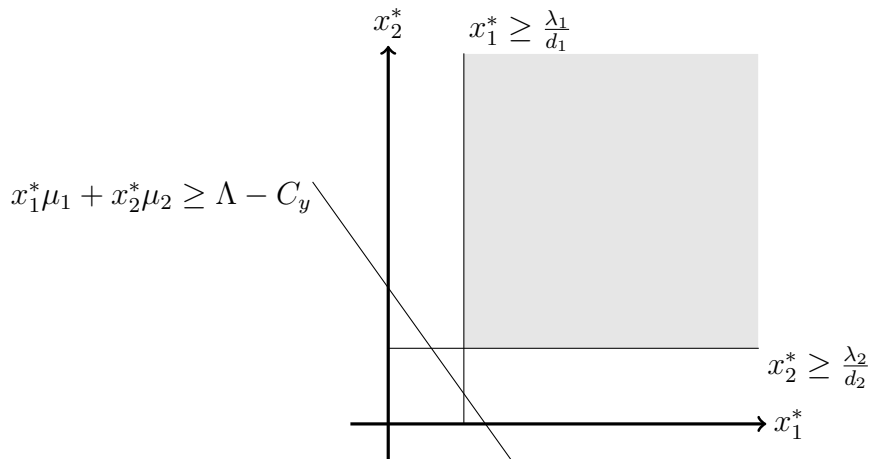


Figure 2.1. Feasible region for the optimization problem with two variables when the conservation of mass restriction is not active.

Network not saturated by download capacity

The most interesting case is when not all the peers are saturated by download capacity, in which the two dimensional case yields a feasible region as in Figure 2.2. Now the region defined by the constraints on the download capacity (2.6) is no longer contained in the half-plane that defines the conservation of mass (2.5) and the minimum is achieved elsewhere.

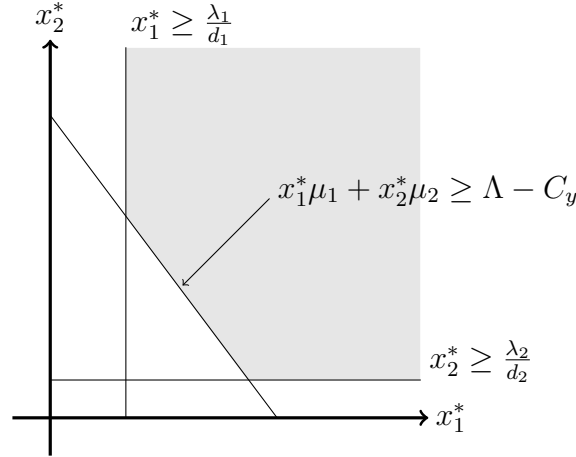


Figure 2.2. Feasible region for the optimization problem with two variables when the conservation of mass restriction is active.

In this case it is convenient to make the change of variables

$$\tilde{x}_i = x_i^* - \frac{\lambda_i}{d_i}$$

where the \tilde{x}_i represents the amount of leechers that there are over the minimum imposed by the download capacity restriction. Then we rewrite our optimization problem in \tilde{x} as

Problem 2

$$\begin{aligned} \inf \quad & \sum_{i=1}^n \tilde{x}_i \\ \text{s.t.} \quad & \sum_{i=1}^n \mu_i \tilde{x}_i \geq \Lambda - C_y - D \\ & \tilde{x}_i \geq 0 \quad \forall i \end{aligned} \tag{2.7}$$

$$\tag{2.8}$$

Now, we are trying to find a positive vector such that the sum of its components is minimum while verifying the inequality of equation (2.7). For example, if we have only two upload capacities, the feasible region would be as in Figure 2.3. Clearly, the solution lies in the vertex closest to the origin, which for an arbitrary number

of upload capacities $\mu_1 > \mu_2 > \dots > \mu_n > 0$ is

$$\begin{aligned}\tilde{x}_1 &= \frac{\Lambda - C_y - D}{\mu_1} \\ \tilde{x}_i &= 0 \quad \forall i > 1\end{aligned}$$

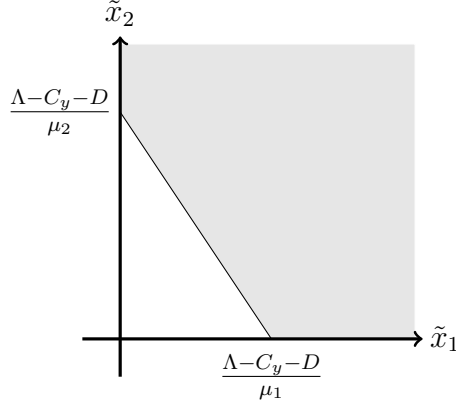


Figure 2.3. Feasible region for the optimization problem with two variables after the variable change.

This result means that the fast peers are accumulated in order to increase the total capacity of the network, thus minimizing the amount of peers in the network. The solution of the optimization problem in this case is summarized in the following proposition.

Proposition 2.2.2. *If the average arrival rate of leechers is greater than the upload capacity of the population of seeders plus the upload capacity of the minimum amount of leechers that we could have, i.e.*

$$\Lambda > D + C_y$$

then the solution of Problem 1 is achieved when all peers are saturated by download capacity except for the fastest ones, i.e.

$$\begin{aligned}x_1^* &= \frac{\Lambda - C_y - D + \frac{\mu_1 \lambda_1}{d_1}}{\mu_1} \\ x_i^* &= \frac{\lambda_i}{d_i} \quad \forall i > 1\end{aligned}$$

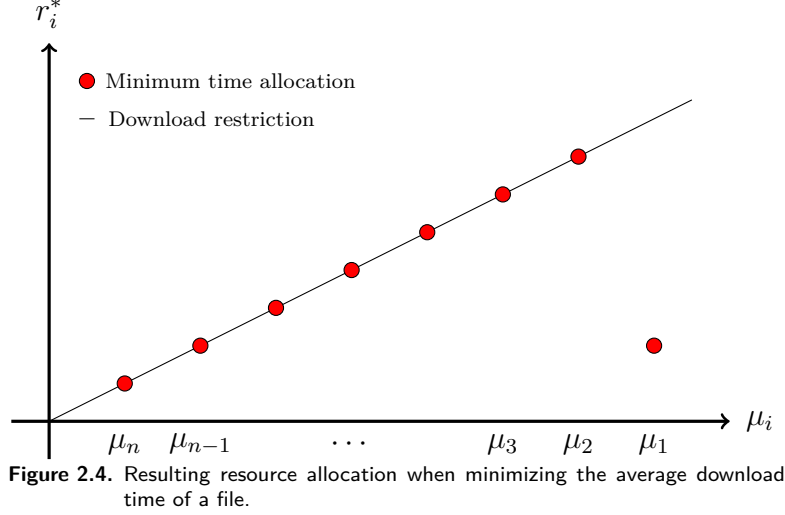
In this case the resource allocation is

$$\begin{aligned}r_1^* &= \frac{\lambda_1 \mu_1}{\Lambda - C_y - D + \frac{\mu_1 \lambda_1}{d_1}} \\ r_i^* &= d_i \quad \forall i > 1\end{aligned}$$

and average download time is

$$\bar{T} = \frac{1}{\Lambda} \left(\frac{\Lambda - C_y - D}{\mu_1} + \sum_{i=1}^n \frac{\lambda_i}{d_i} \right)$$

When $\Lambda > D + C_y$, this will be the minimum average download time for the leechers. Any other allocation will have a longer one. The allocation profile in this case is depicted in Figure 2.4.



2.3 INTRODUCING INCENTIVES

The previous resource allocation indeed minimizes the average download time of the peers, but it comes at the expense of the peers that contribute more to the network, as can be seen in Figure 2.4. In fact, the fastest peers may not be the ones with the highest download rate, since the allocation defined by r_i^* may not be increasing with the upload rate. This happens if

$$d_2 > \frac{\lambda_1 \mu_1}{\Lambda - C_y - \sum_{i=2}^n \frac{\mu_i \lambda_i}{d_i}}$$

In that case there is no gain in being one of the peers that contribute the most to the network. This condition is equivalent to the fact that if we lower the download capacity d_1 and make it equal to d_1 , the network would still not be saturated by download capacity. As most P2P networks are far from being saturated by download capacity, this condition holds in most practical cases and thus this allocation does not provide proper incentives to the fastest peers.

In most of the BitTorrent clients, the user has the ability to change its upload rate at will (always upper bounded by their actual download capacity). If we gave this power to the peers under the previous bandwidth allocation, then no peer would choose to upload at the maximum rate μ_1 , but instead they would choose the second largest rate (or the closest one). Intuitively, this would lead to a situation in which everyone would end up lowering its upload rate to a minimum. In order to prevent this undesirable scenario, it is enough to impose that the allocation should be increasing as the effective upload rate of the peers increases.

In appendix A we formalize this intuition by modeling this problem as a non-cooperative non-atomic game for the leechers.

2.3.1 Imposing an increasing resource allocation

First of all, we will consider only the case where $\Lambda > D + C_y$, as in the other case the only efficient rate allocation is $r_i^* = d_i \forall i$. In that case, we can include the increasing nature of the download rates as a new restriction Problem 1, but now it is convenient to present it as an optimization problem directly in the download rates r_i^* :

Problem 3

$$\begin{aligned} \inf \quad & \sum_{i=1}^n \frac{\lambda_i}{r_i^*} \\ \text{s.t.} \quad & \sum_{i=1}^n \mu_i \frac{\lambda_i}{r_i^*} \geq \Lambda - C_y \\ & r_i^* \leq d_i \quad \forall i \\ & r_i^* > r_{i+1}^* \quad \forall i \end{aligned}$$

Now, we can find a resource allocation vector r^* that minimizes the download times but that is non-increasing in i (non-decreasing with the upload rate). This would be the infimum of our problem, not the minimum as it is not strictly increasing with the upload rate. The solution is the max-min allocation depicted in Figure 2.5, in which the slowest peers will receive $r_i^* = d_i$ and then from some index i^* forward they will all receive a constant rate R . The rate R is obtained from the conservation of mass equation

$$\sum_{i=1}^{i^*} \mu_i \frac{\lambda_i}{R} + \sum_{i=i^*+1}^n \mu_i \frac{\lambda_i}{d_i} = \Lambda - C_y$$

which yields

$$R = \frac{\sum_{j=1}^{i^*} \lambda_j \mu_j}{\Lambda - C_y - \sum_{j=i^*+1}^n \frac{\lambda_j \mu_j}{d_j}}$$

The index i^* is the maximum index that produces an increasing allocation, which is greater than 0 because the network is not saturated by download capacity, but could take any value up to n , in which case we obtain a constant allocation. As a

result, the general bandwidth allocation will be

$$r_i^* = \frac{\sum_{j=1}^{i^*} \lambda_j \mu_j}{\Lambda - C_y - \sum_{j=i^*+1}^n \frac{\lambda_j \mu_j}{d_j}} \quad \text{if } i = 1, \dots, i^* \quad (2.9)$$

$$= d_i \quad \text{if } i = i^* + 1, \dots, n \quad (2.10)$$

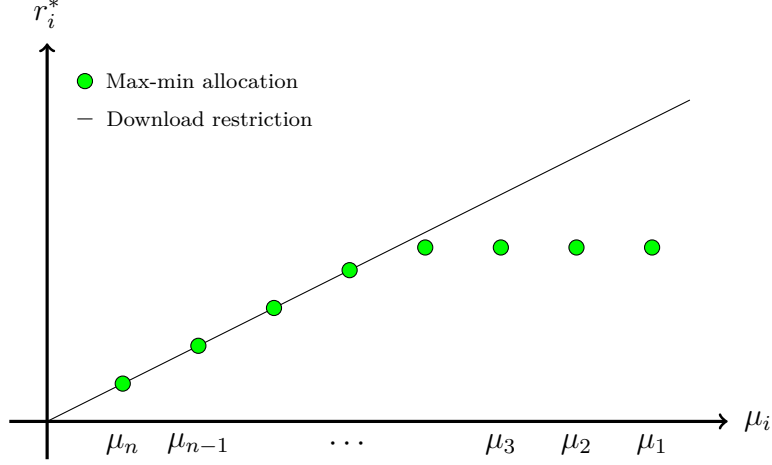


Figure 2.5. Resulting resource allocation with max-min fairness.

Note that for slower peers it is equivalent to the allocation that results in minimizing the average download times (Figure 2.4), but then it is nondecreasing. Typically the network is far from being saturated by download capacity, so we would get an allocation in which most of the peers get the same download rate.

Remark 2.3.1. We have a critical value of d_n^* for which we have $i^* = n$ if $d_n \geq d_n^*$. This is

$$d_n^* = \frac{\sum_{i=1}^n \mu_i \lambda_i}{\Lambda - C_y}$$

In that case the resource allocation is constant. The resulting resource allocation profile will be as in Figure 2.6.

In theory one could try to achieve max-min in order to provide incentives and minimize the average download time. This could be done in practice by connecting to other peers at random, as this would yield an uniform allocation for all peers (or saturate the download capacity). However, we argue that max-min is not really enough, because we need a strictly increasing allocation. In this regard, a natural, simple and fair resource allocation rule would be: you should get at least as much as you give.

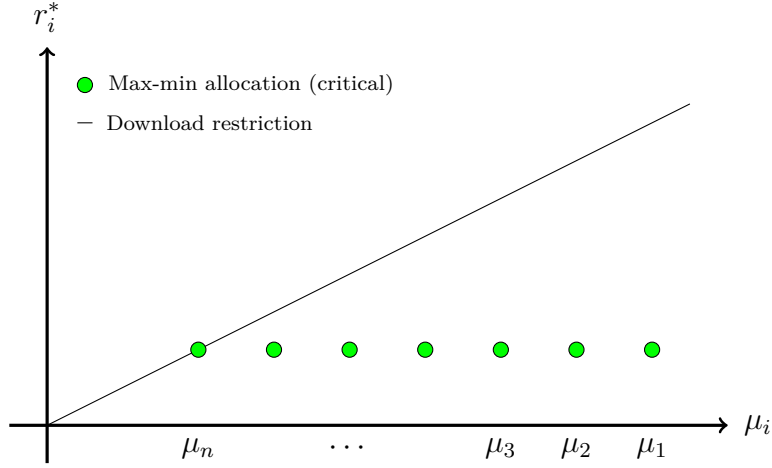


Figure 2.6. Constant max-min resource allocation.

2.4 PROPORTIONAL FAIRNESS

As we said before, a more appropriate resource allocation is obtained by imposing a notion of fairness, the proportional fairness. This allocation provides stronger incentives to peers to use their maximum upload rate, as their download rate will increase proportionally. With this notion of fairness, if the download capacities are high enough, each peer will have a download rate

$$r_i^* = \alpha \mu_i \quad \forall i$$

Using the conservation of mass (Equation (2.5)), we get that

$$\sum_{i=1}^n \frac{\lambda_i}{\alpha} \geq \Lambda - C_y$$

then if $\Lambda > C_y$ (i.e., if it is not sustained by seeders)

$$\alpha = \frac{\Lambda}{\Lambda - C_y}$$

and as a result we have the resource allocation

$$r_i^* = \mu_i \frac{\Lambda}{\Lambda - C_y}$$

The average time to download will be of course greater than in the previous case

$$\bar{T} = \sum_{i=1}^n \frac{\lambda_i}{\Lambda} \left(\frac{\Lambda - C_y}{\mu_i \Lambda} \right)$$

Note that if $d_i = K \mu_i$ then we will have only two possible situations: all peers are saturated by download capacity or none of them are. In fact, they will be saturated

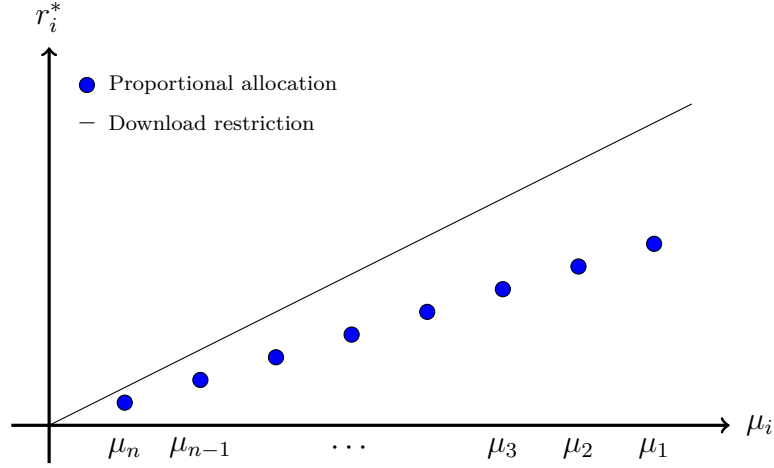


Figure 2.7. Resulting resource allocation when imposing proportional fairness.

by download capacity when $K \leq \frac{\Lambda}{\lambda - C_y}$. When it is not saturated by download capacity, a profile of the download rates is depicted in Figure 2.7.

This proportionally fair allocation provides direct, transparent incentives for peers to contribute and it is the one that we should aim to impose in practice.

The case in which the leechers produce a proportional allocation and in which the seeders produce a max-min allocation (which would also be incentive compatible) was studied in [8].

2.5 CONCLUSIONS

In this chapter we first presented a fluid model for the population dynamics of a wired P2P network for general resource allocation functions. Then we presented the resource allocation that minimizes the average download time and showed that in most cases it does not provide the proper incentives for the peers. Then we found that the max-min allocation is the best in terms of average download time that provides minimal incentives, but the stronger incentives of a proportional allocation are better suited for these networks.

A formalization of the incentives from a game theoretic point of view is given in Appendix A. Furthermore, a generalization of this analysis for when the upload capacities can take any positive value is given in Appendix B.

Chapter 3

Proportional reciprocity and the Sinkhorn iteration

In the previous chapter we concluded that a proportional allocation is a desirable choice for the resource allocation in a wired P2P network. However, we assumed that we could distribute the upload capacity of the peers at will, regardless of the network connectivity or any other restrictions (such as the fact that a peer cannot use its own upload capacity). Its practical feasibility is not at all obvious.

The feasibility of a proportional fair allocation can be studied by writing the mutual peer exchange bandwidths in matrix form: the question reduces to a problem of matrix row and column renormalization. Indeed, the natural iteration of renormalizing rows and columns leads to a *reciprocity* algorithm that can achieve the desired allocation, whenever feasible.

3.1 RESOURCE ALLOCATION MODEL

We begin by defining some notation. A set of N peers shares information through a connectivity graph G : two peers are neighbors in this graph if they can exchange information. Let $A = (a_{ij})$ be the adjacency matrix of the graph, assumed symmetric. Note that in BitTorrent parlance, in this chapter we are only modeling the behavior of *leechers*, who are both uploading and downloading content. We note that $a_{ii} = 0 \forall i$.

We model the bandwidth sharing by a matrix $Z \in \mathbb{R}_+^{N \times N}$ in which the entry z_{ij} corresponds to the throughput of the connection from peer i to peer j . The matrix Z has the following properties:

$$z_{ij} = 0 \text{ if } a_{ij} = 0, \quad \sum_j z_{ij} = \mu_i \quad \forall i, \quad (3.1)$$

where μ_i is the total upload rate of peer i .

On the other hand, the received bandwidth per peer is obtained through the column sums

$$r_j(Z) = \sum_i z_{ij} \quad \forall j.$$

We consider as target allocation the situation where each peer receives the same bandwidth that it gives to the network, that is $r_j = \mu_j \forall j$, which we will call a proportional allocation. This property was called *global proportional fairness* by De Veciana in [9]. In Chapter 2 we saw that his rule provides strong incentives for the peers to contribute to the network.

At this point there are two questions to be asked. The first question is whether a matrix Z exists satisfying $r_j(Z) = \mu_j$, i.e. with prescribed row and column sums; we will call this a *feasible* allocation. This problem is different from most graph-based resource allocation problems, as the bottlenecks are in the nodes instead of in the edges. The second question is whether the above allocation admits a decentralized implementation, i.e. a set of mutual exchange rules peers can follow to achieve it, without the intervention of a central authority.

In order to find such an allocation a natural discrete algorithm would be to upload to each other peer with a rate proportional to the download rate obtained from that peer in each step, which is called the *proportional response dynamics* and was first proposed in [10]. This is a decentralized algorithm that only uses local information about the current resource allocation of the network and can be expressed in mathematical terms (using the matrix representation of allocations) as

$$z_{ij}^{k+1} = \frac{z_{ji}^k}{\sum_k z_{ki}^k} \mu_i \quad \forall i, j$$

In order to study the convergence of this algorithm and whether the limit has the desired properties, we are going to review a very similar matrix scaling algorithm first developed by Sinkhorn in the 1960's.

3.2 ORIGINAL SINKHORN APPROACH

The algorithm developed by Sinkhorn was trying to solve a broader matrix problem:

Let B be a $m \times n$ nonnegative matrix and $f \in \mathbb{R}_+^m$, $c \in \mathbb{R}_+^n$ two fixed vectors. Find, if it exists, a $m \times n$ nonnegative matrix Z such that $z_{ij} = 0$ if $b_{ij} = 0$ and such that the row sums are f and the column sums are c .

Our problem is exactly the case in which the matrix B is the initial resource allocation of the network and the two vectors are equal to the upload capacity vector, i.e. $f = c = \mu$.

Initially the algorithm to solve this problem was introduced for doubly stochastic matrices ($f = c = (1, \dots, 1)$) in [11, 12, 13] and then refined for arbitrary nonnegative matrices and vectors (See [14] and references therein). We will focus on the latter version.

Definition 3.2.1 (Sinkhorn scaling algorithm). Given a nonnegative $m \times n$ matrix B , and specified vectors of the row sums ($f \in \mathbb{R}_+^m$) and column sums ($c \in \mathbb{R}_+^n$), we iterate the following algorithm with initial values $B_{ij}^0 = B_{ij}$. In the odd steps we normalize the rows to meet the desired row sums, and in the even steps we normalize the columns to meet their prescribed sums, yielding the two step algorithm

$$b_{ij}^{k+1} = \frac{b_{ij}^k}{\sum_{l=1}^n b_{il}^k} f_i$$

$$b_{ij}^{k+2} = \frac{b_{ij}^{k+1}}{\sum_{l=1}^m b_{lj}^{k+1}} c_j$$

This algorithm can be seen as a discrete dynamical system in which the state alternates between a matrix in the set of matrices with desired column sum and a matrix in the set of matrices with desired row sum, always with the same zeros as the initial condition. Note that in the limit there could be more zeros.

Ideally, this renormalization of rows and columns will converge to a matrix which has the prescribed row and column sums. Although this is a two step algorithm, we analyze the sequence of matrices $B^{(n)}$ after each step in the space of all $m \times n$ matrices. As with any sequence, we want to obtain the conditions under which this algorithm converges to a unique limit, and a characterization of this limit. These will depend mostly on zeros of the initial condition $B^{(0)}$ and on the row and column sum vectors f and c . Furthermore, even if the sequence has no limit, we can analyze the even and odd subsequences and find the limit for them.

Example 3.2.1. Suppose that we have the initial matrix

$$B^{(0)} = \begin{pmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{pmatrix}$$

and row and column sum vectors $f = c = (2, 3, 4)$. Then the iteration converges to the matrix

$$B^{(\infty)} = \begin{pmatrix} 0 & \frac{1}{2} & \frac{3}{2} \\ \frac{1}{2} & 0 & \frac{5}{2} \\ \frac{3}{2} & \frac{5}{2} & 0 \end{pmatrix}$$

Note that in this case the limit has the prescribed row and column sums and the zero structure is the same as the initial matrix. If we change the row and column sum vectors to $f = c = (1, 2, 3)$, the algorithm still converges but to the matrix

$$B^{(\infty)} = \begin{pmatrix} 0 & 0 & 1 \\ 0 & 0 & 2 \\ 1 & 2 & 0 \end{pmatrix}$$

In this second case, the limit matrix still has the desired row and column sums, but there are more zeros in the limit than in the original matrix. Note that in order for the third column to have sum equal to 3, all the “mass” from the first and second rows has to be concentrated in the third column. Lastly, if we take $f = c = (1, 2, 4)$ the algorithm does not converge but it oscillates between the matrices

$$B_1 = \begin{pmatrix} 0 & 0 & 1 \\ 0 & 0 & 2 \\ \frac{4}{3} & \frac{8}{3} & 0 \end{pmatrix}$$

and

$$B_2 = \begin{pmatrix} 0 & 0 & \frac{4}{3} \\ 0 & 0 & \frac{8}{3} \\ 1 & 2 & 0 \end{pmatrix}$$

Note that in this case a matrix with the desired sums does not exist, as even in the case of allocating all the “mass” of columns one and two in the third column, we cannot reach a column sum equal to four. This limitation comes from the vectors f and c and from the zeros of the initial matrix $B^{(0)}$. For instance, if the diagonal entries could be non zero, then we could have any row and column sum when $f = c$.

We now introduce some definitions in order to state the convergence theorem for the Sinkhorn algorithm which will explain the results of the previous example.

Definition 3.2.2 (Zero minor). Let B be a $m \times n$ matrix. Given the subsets of indices $Z \subset \{1, \dots, m\}$ and $L \subset \{1, \dots, n\}$, the sub matrix B_{ZL} which results from discarding rows indexed in Z^C and columns in L^C is called a minor of B , and in particular it is a zero minor if $B_{ZL} = 0$.

Definition 3.2.3 (Almost and exactly scalable matrices). A nonnegative matrix B is *almost scalable* to row and column sums $f \in \mathbb{R}_+^m$ and $c \in \mathbb{R}_+^n$, with $\sum_{j=1}^n c_j = \sum_{i=1}^m f_i$, if for every zero minor B_{ZL} of B ,

$$\sum_{i \in Z^c} f_i \geq \sum_{j \in L} c_j \tag{3.2}$$

Furthermore, if equality holds only if $B_{Z^c L^c}$ is a zero minor as well, then the matrix is *exactly scalable*.

Note that in Example 3.2.1, the first case is exactly scalable, the second one is almost scalable and the last one is neither. The following theorem states the convergence conditions of the Sinkhorn algorithm which formalizes the ideas drawn from the example.

Theorem 3.2.1 (Balakrishnan et al. [14]). *Consider a $m \times n$ nonnegative matrix B , and the desired row sums $f \in \mathbb{R}_+^m$ and column sums $c \in \mathbb{R}_+^n$ with $\sum_{j=1}^n c_j = \sum_{i=1}^m f_i$.*

- *There exists a $m \times n$ matrix Z which satisfies these prescribed row and column sums, where $Z = D_1 B D_2$ for some $D_1 \in M_{m \times m}(\mathbb{R})$ and $D_2 \in M_{n \times n}(\mathbb{R})$, both diagonal and positive definite $\Leftrightarrow B$ is exactly scalable.*
- *If the above is true, the Sinkhorn scaling of B will converge to such a matrix Z .*
- *If B is almost scalable but not exactly scalable, the algorithm will converge to a unique limit of the form $\lim_{n \rightarrow \infty} D_1^{(n)} B D_2^{(n)}$ which satisfies the row and column constraints. However, the individual matrix sequences $D_1^{(n)}$ and $D_2^{(n)}$ will not converge.*

This theorem gives us necessary and sufficient conditions for the convergence of the Sinkhorn algorithm to a unique limit. This limit of course has the desired row and column sums and depends strongly on the initial condition. Note that the convergence of the algorithm depends only on the zero structure of the initial matrix B and on the vectors f and c . We can now state a corollary on the zero structure of the limit matrix Z .

Corollary 3.2.1. *If the initial matrix B is almost scalable, then the Sinkhorn algorithm converges to Z such that if $b_{ij} = 0 \Rightarrow z_{ij} = 0$ (but there might be more zeros in Z than in B). If furthermore the initial matrix B is exactly scalable, then Z has exactly the same zero structure of B . That is $z_{ij} = 0 \Leftrightarrow b_{ij} = 0$.*

3.2.1 Sinkhorn scaling as an optimization problem

We now give a different approach to the convergence and to the characterization of the limit of the Sinkhorn algorithm. The main result follows.

Theorem 3.2.2. *Given an almost scalable matrix $B \in M_{m \times n}(\mathbb{R}_+)$ for $f \in \mathbb{R}_+^m$ and $c \in \mathbb{R}_+^n$, then the limit given by the Sinkhorn algorithm Z results in the minimum of the following optimization problem*

$$\begin{aligned}
 \min_{z_{ij}: b_{ij} \neq 0} \quad & \sum_{i,j: b_{ij} \neq 0} z_{ij} \log \left(\frac{z_{ij}}{b_{ij}} \right) \\
 \text{s.t.} \quad & \sum_{j=1}^n z_{ij} = f_i \quad \forall i = 1, \dots, m \\
 & \sum_{i=1}^m z_{ij} = c_j \quad \forall j = 1, \dots, n \\
 & z_{ij} \geq 0 \quad \forall i, j
 \end{aligned}$$

Corollary 3.2.2. *If $f = c$ and the initial matrix B is symmetric, then the limit matrix Z of the Sinkhorn iteration is symmetric.*

Proof. Let Z_{opt} be the optimal matrix in Theorem (3.2.2). Since B symmetric, the cost is a symmetric function of Z and so are the constraints, therefore Z_{opt}^T is also optimal. But the objective function is strictly convex and thus it has a unique minimum, therefore $Z_{opt} = Z_{opt}^T$. \square

3.3 APPLICATIONS TO RESOURCE ALLOCATION IN P2P NETWORKS

Recall that the proportional response dynamics, intended as a decentralized algorithm to achieve a proportional resource allocation, was defined by

$$z_{ij}^{k+1} = \frac{z_{ji}^k}{\sum_k z_{ki}^k} \mu_i \quad \forall i, j$$

In the language of matrix scaling, this means to transpose the matrix and renormalize rows to have sum μ . In this sense, except for the transpose operation, this iteration amounts to the Sinkhorn's iterative row and column renormalization. This connection makes it possible to obtain results for the proportional response iteration from the extensive literature that studied the Sinkhorn renormalization which was reviewed in this chapter. The main result follows.

Theorem 3.3.1 (Proportional response dynamics convergence). *Let A be the adjacency matrix of the connectivity graph of a P2P network, and let $\mu \in \mathbb{R}_+^N$ be the vector of upload capacities. If A is almost scalable for $f = c = \mu$, then for any initial resource allocation Z_0 in which every possible connection is active (i.e. $z_{ij} > 0$ if $a_{ij} > 0$), the even and odd subsequences of the proportional response dynamics converge to a proportional resource allocation Z and its transpose. If furthermore $Z_0 = Z_0^T$, then the limit is a unique symmetric resource allocation Z . In any case, the two step average of the resource allocations converge to a symmetric resource allocation.*

Remark 3.3.1. Note that even if our objective was to achieve an allocation in which the total upload rate of a peer μ_i was equal to the total download rate r_i , the proportional response dynamics goes one step further and produces symmetric allocations, which have a peerwise balance of bandwidth ($z_{ij} = z_{ji} \forall i, j$).

One of the key conditions for this theorem is that the adjacency matrix A has to be almost scalable, which is a condition on its zero minors and on the upload capacity vector μ . If we consider a special case of interest where the network is full mesh, then the adjacency matrix A is almost scalable if

$$\mu_i \leq \sum_{k \neq i} \mu_k \quad \forall i$$

This can be interpreted as if each node in the network can get as much bandwidth as it gives, which enables the kind of reciprocity that we are looking for. This condition is very easy to achieve in a P2P network, as they generally have a large amount of peers and it is very difficult that one can contribute more than the rest of the network combined. As a result, it is safe to assume that this condition always holds in practice.

The other key condition is that the initial condition has to have all possible connections active in order to ensure that Z_0 is also almost scalable. This one is harder to achieve in a P2P network, as usually the peers will only have a small amount of connections active in order to reduce overheads. Furthermore, in a dynamic network when new peers arrive, initially they will have no active connections, which would render this algorithm useless. In [3] they propose an algorithm that devotes 80% of the upload bandwidth to the proportional reciprocity scheme and the remaining 20% to an optimistic connection as the one used by BitTorrent's neighbor selection algorithm. This solves the problem of opening new connections but still has the complications of maintaining a large amount of connections active and fine tuning the actual throughput of those connections, which would make more difficult the use of transport layer protocols like TCP. Besides, the use of 20% of the bandwidth for an optimistic connection goes against the proportional allocation that the other part is trying to achieve.

Taking everything into account, we note that the proportional response dynamics is an excellent algorithm to achieve proportional allocations in a network, but it has several implementation issues which were only partially solved. In the next chapter we are going to design a neighbor selection algorithm which takes into account the implementation restrictions that a neighbor selection algorithm should have.

Chapter 4

Near selection for wired P2P networks

In this chapter we follow the route of progressively imposing the design constraints of practical systems for the proportional allocation and developing a decentralized algorithm that achieves it. The new proposal is analyzed mathematically and tested in simulations. Most of this chapter was already published in [15].

The ideal reciprocity scheme of the previous chapter is not easily taken to practice. We consider two important implementation restrictions:

- (i) For overhead reasons, peers must maintain simultaneous connection with only a small amount of peers.
- (ii) Due to the underlying TCP protocol, these connections will receive an amount of bandwidth that depends on the RTTs, which we will assume equal. This yields a uniform split of the peer's upload bandwidth.

To study these limitations we introduce an *energy function* which is zero under ideal reciprocity, and which a practical scheme should try to reduce. Although optimizing energy under constraints (i)-(ii) has combinatoric complexity, we identify cases where zero energy is indeed achievable; more generally, we characterize the algorithm in which each peer tries to myopically reduce its portion of the energy: a tit-for-tat structure similar to BitTorrent's comes out naturally from this procedure.

The final step in the implementation road is to introduce some randomness in peer selection, which avoids one of the pitfalls that appear in a deterministic myopic algorithm by exploring the set of peers, which will in practice vary in time. For this task we turn to a Gibbs' sampler, designing a Markov process guided by a potential defined in terms of our energy function. In this regard, we note that recent papers [16, 17] have introduced this technique in the study of P2P systems from a network

utility maximization perspective. As we will explain, there are differences between the two proposals, reflected in the potential functions used.

The resulting neighbor selection algorithm was tested in simulation and compared to other existing protocols, performing well against the alternatives in terms of reciprocity and fairness.

4.1 IMPLEMENTATION RESTRICTIONS: DISCRETE CONNECTIONS

The proportional response dynamics reviewed in Chapter 3 is a decentralized algorithm that achieves the desired allocation in a P2P network when possible. However, problems arise if we wish to implement such algorithm in practice. First of all, it needs a constant connection with each neighbor peer in the network; this is impractical as there would be too many active connections, which leads to more overhead than is desired. Secondly, each connection would have to be fine-tuned to a desired rate. This is difficult to achieve, specially if you are planning on using TCP as underlying protocol. Finally, this is a completely deterministic algorithm and as such lacks the necessary randomness to explore the different peering options as the network evolves.

The BitTorrent algorithm, although practical and easy to implement, yields an allocation which is not proportional in most cases and has some issues that we explained in Section 1.1. There is thus room left for exploring alternatives to the BitTorrent neighbor selection, that will more closely reflect our design objective of proportional allocation, within the practical constraints that have been identified. In this chapter we will address two of these constraints, which stem from the discrete nature of connections and reliance of transport protocols that impose bandwidth sharing:

1. Each peer can only open a maximum amount of N_0 connections.
2. The upload capacity of each peer is equally distributed between all outbound connections.

In the following, N will denote the number of peers, with their upload capacities in decreasing order: $\mu_1 \geq \mu_2 \geq \dots \geq \mu_N$. Again, it is assumed there are no other bottlenecks in the network.

4.2 ENERGY DRIVEN ALLOCATIONS

As a means to study the impact of the discrete constraints on the desired reciprocity, we will introduce an *energy function* $\mathcal{E}'(Z)$, sum of squares of the peerwise

discrepancies in exchange rates, as follows:

$$\mathcal{E}'(Z) = \frac{1}{2} \sum_{i,j} (z_{ij} - z_{ji})^2.$$

This function is defined over the set of allocation matrices

$$\mathcal{M} = \left\{ Z : z_{ij} = 0 \text{ if } a_{ij} = 0, \sum_j z_{ij} = \mu_i \forall i, z_{ij} \geq 0 \forall i, j \right\}$$

for a given vector of upload capacities μ and a given adjacency matrix A , assumed symmetric.

Proposition 4.2.1. *If the row and column scaling problem with adjacency matrix A and capacities μ is feasible, then the allocations of minimal energy $\mathcal{E}'(Z) = 0$ are precisely the symmetric allocations*

$$\mathcal{M}^* = \{ Z \in \mathcal{M} : Z = Z^T \} \neq \emptyset.$$

The above follows from the theory reviewed in Chapter 3. The set \mathcal{M}^* is convex and is the set of average allocations that can be obtained by the proportional response dynamics, which make this a proper energy for our purposes.

We now begin to incorporate the discrete restrictions imposed by the number N_0 of peer connections, and the equal bandwidth between them. At this point it is convenient to factor out the peer bandwidths and introduce a matrix X with coefficients in $\{0, \frac{1}{N_0}\}$ that stores the neighboring configurations in terms of the fractions x_{ij} of its own bandwidth that peer i allocates to each peer j . From it, the rate allocation can be obtained as

$$Z = \text{diag}(\mu_i) X.$$

Based on this, we can redefine the energy as a function of the neighboring configurations X

$$\mathcal{E}(X) = \frac{1}{2} \sum_{i,j} (\mu_i x_{ij} - \mu_j x_{ji})^2. \quad (4.1)$$

We would like to minimize the energy $\mathcal{E}(X)$ with the incorporated restrictions. The minimization now is over the subset of stochastic matrices

$$\Lambda^S = \left\{ X \in \left\{0, \frac{1}{N_0}\right\}^{N \times N} : x_{ij} = 0 \text{ if } a_{ij} = 0; \sum_{j \in S} x_{ij} = 1 \right\}$$

Although the minimization of the energy without the discrete restrictions yields symmetric matrices, the configuration that minimizes the energy (4.1) need not be symmetric.

Example 4.2.1. Suppose that we have five peers ($N = 5$) that can open two connections at a time ($N_0 = 2$) in a full mesh network. Furthermore, suppose that $\mu_1 = \mu_2 = \mu_3 = 10$ and $\mu_4 = \mu_5 = 1$. Then the configuration that minimizes the energy (Figure 4.1) is not symmetric:

$$X^* = \begin{pmatrix} 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 \\ \frac{1}{2} & 0 & \frac{1}{2} & 0 & 0 \\ \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{2} & 0 & \frac{1}{2} \\ \frac{1}{2} & 0 & 0 & \frac{1}{2} & 0 \end{pmatrix}$$

and the energy in this case is $\mathcal{E}(X^*) = \frac{1}{2}$.

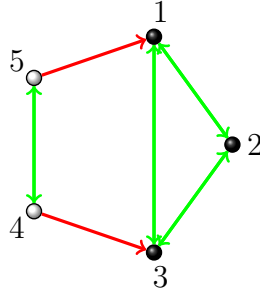


Figure 4.1. Example non symmetrical configuration of minimum energy. We have fast peers (black) and slow peers (white). Between them there are reciprocated links (green) and unreciprocated links (red).

The minimization of the energy is a very large combinatoric problem, so an exact solution is very hard to obtain. However, in certain cases we can obtain a configuration that minimizes the energy.

Proposition 4.2.2. *In a full mesh network, suppose that N_0 is even and that we have k groups of peers with the same upload bandwidth μ_i for each member of group i . Besides, each group has $N_i > N_0$ peers \Rightarrow There exists a configuration X^* such that $\mathcal{E}(X^*) = 0$.*

Proof. As we have groups of peers with the same bandwidth, we want to form sets of peers with the same bandwidth connected to each other, but disconnected from the rest, thus obtaining a configuration X^* with $\mathcal{E}(X^*) = 0$. Equivalently, for each group of peers we have to find a N_0 -regular graph with N_i vertices. Fortunately Theorem D.1.1 stated in Appendix D gives us the existence of such graphs because N_0 is even. As a result, every group of N_0 regular graphs would make the energy equal to 0. \square

Example 4.2.2. For example, given a set of peers distributed in three groups of sizes $N_1 = 7$, $N_2 = 11$ and $N_3 = 5$ and upload capacities μ_1 , μ_2 and μ_3 ; if we take $N_0 = 4$, we can build Cayley graphs (defined in Appendix D) and achieve minimum energy (Figure 4.2). Even though this graph is not connected, this is just an example of a configuration of minimum energy. The randomness that will be incorporated later will ensure that every link will have positive probability of being used.

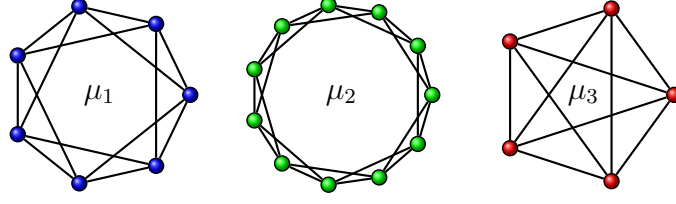


Figure 4.2. Example of minimum energy configuration

Even if the peers have not exactly the same upload bandwidth, but are close, we can find an upper bound for the minimum achievable energy.

Proposition 4.2.3. *Suppose that the network is full mesh and N_0 is even. Divide the set of peers into k groups, where the bandwidths $\{\mu_i\}$ for peers in each group occupy an interval of length δ . If every group has $N_i > N_0$ peers, there exists at least one configuration X^* such that $\mathcal{E}(X^*) \leq \delta^2 \frac{N}{2N_0}$.*

Proof. Consider the same X^* constructed in the previous proposition. Write the total energy as $\mathcal{E}(X^*) = \sum_k \mathcal{E}_i(X^*)$, adding the energy contributions of each disconnected group. For group i we have $N_i N_0$ mutual connections, each with energy

$$\frac{1}{2}(\mu_i x_{ij} - \mu_j x_{ji})^2 \leq \frac{\delta^2}{2N_0}.$$

Therefore $\mathcal{E}_i(X^*) \leq \delta^2 \frac{N_i}{2N_0}$ and the result follows from $\sum_i N_i = N$. \square

This suggests that grouping peers in subsets of similar bandwidth, of any size greater than N_0 , is a good strategy to approximate the goal of proportional reciprocity. The size of the classes will be a function of the existing set of μ_i 's.

Example 4.2.3. In the same situation as in Example 4.2.2, but now with just similar upload bandwidths, the same Cayley graph is a bound for the energy (Figure 4.3).

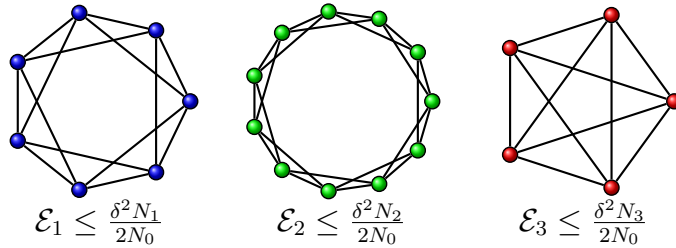


Figure 4.3. Example of minimum energy configuration

Note that this is fundamentally different to the formation of cliques that the BitTorrent tit-for-tat mechanism produces (as noted in [6]). The formation of cliques only depends on the ranking of the peer by its upload capacity and not on the capacity itself. This could lead to severe unfairness in the resulting resource allocation as portrayed in the following example.

Example 4.2.4. Suppose that $N_0 = 4$ and that we have seven peers with upload capacity $\mu_1 = 10$ and eight peers with upload capacity $\mu_2 = 1$. Then BitTorrent would form cliques as in Figure 4.4, yielding an unfair allocation.

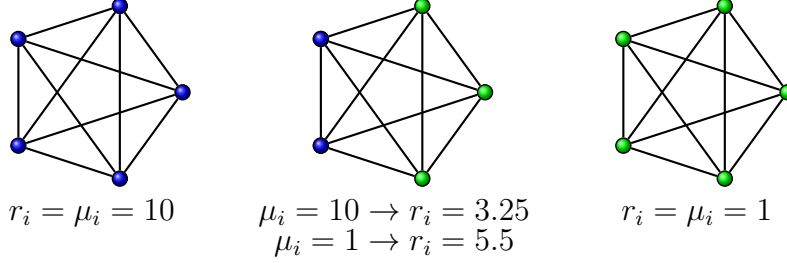


Figure 4.4. Formation of cliques with BitTorrent's neighbor selection algorithm.

This suggests that grouping peers in subsets of similar bandwidth, of any size greater than N_0 , is a good strategy to approximate the goal of proportional reciprocity. The size of the classes will be a function of the existing set of μ_i 's; the flexibility of going beyond cliques of size $N_0 + 1$ can lead to significant improvements.

4.3 DETERMINISTIC APPROACH

The question to ask at this point is: can the energy be minimized by a *decentralized* algorithm? Given the combinatoric nature of the problem we do not expect the global optimum to be computable, but a reasonable heuristic is to have each peer i choose its outgoing connections seeking to myopically reduce its own portion of the energy

$$\mathcal{E}_i(X) := \sum_j (\mu_i x_{ij} - \mu_j x_{ji})^2$$

In this minimization we assume as given the rates $\mu_j x_{ji}$ received by peer i , and we introduce the notation $J^{in} = \{j : x_{ji} \neq 0\}$ for the set of peers from which peer i is currently receiving data. Let N^{in} be the cardinality of this set, and note that there are no a priori constraints on it, in principle $0 \leq N^{in} \leq N - 1$.

Since peer i will divide its bandwidth uniformly among its N_0 outgoing connections, the myopic optimization is just to choose the set $J^{out} = \{j : x_{ij} \neq 0\}$, of cardinality N_0 , to minimize the energy portion $\mathcal{E}_i(X)$. The following proposition characterizes the optimal configuration.

Proposition 4.3.1. *Given a set J^{in} of peers uploading to i , a configuration X^* minimizes the local energy \mathcal{E}_i if and only if it solves*

$$\max_{J^{out}} \sum_{J^{in} \cap J^{out}} \mu_j \tag{4.2}$$

Proof. For convenience we will denote by $\tilde{\mu}_j := \frac{\mu_j}{N_0}$, the fraction of bandwidth allocated in a single connection from peer j . The local energy of a given configuration X can then be expressed as follows:

$$\mathcal{E}_i(X) = \sum_{j \in J^{in} \cap J^{out}} (\tilde{\mu}_i - \tilde{\mu}_j)^2 + \sum_{j \in J^{in} \setminus J^{out}} \tilde{\mu}_j^2 + \sum_{j \in J^{out} \setminus J^{in}} \tilde{\mu}_i^2$$

Expanding the square $(\tilde{\mu}_i - \tilde{\mu}_j)^2 = \tilde{\mu}_i^2 + \tilde{\mu}_j^2 - 2\tilde{\mu}_i\tilde{\mu}_j$ and rearranging terms leads to the equivalent expression

$$\mathcal{E}_i(X) = \sum_{j \in J^{in}} \tilde{\mu}_j^2 + \sum_{j \in J^{out}} \tilde{\mu}_i^2 - 2 \sum_{j \in J^{in} \cap J^{out}} \tilde{\mu}_i \tilde{\mu}_j$$

The first term above is given, and the second is fixed at $N_0 \tilde{\mu}_i^2$ for all allowable configurations, so only the third term can be minimized by choice of J^{out} ; noting that $\tilde{\mu}_i$ is fixed, and $\mu_j = N_0 \tilde{\mu}_j$, we arrive at the equivalent maximization (4.2). \square

To interpret the max-weight type condition (4.2), we distinguish two cases:

- (i) $N^{in} \leq N_0$. In this case it is clearly optimal in (4.2) to cover the entire set J^{in} with J^{out} , assigning any extra elements arbitrarily.
- (ii) $N^{in} > N_0$. In this case only a portion of the μ_j can be included. The maximum weight is achieved by assigning J^{out} to the largest N_0 values of $\{\mu_j, j \in J^{in}\}$.

So we see that the local reciprocity energy is minimized by picking N_0 peers that are currently giving the most bandwidth to peer i , and assigning any extra slots arbitrarily. Interestingly, this corresponds exactly to the tit-for-tat part of the BitTorrent algorithm. Therefore, the myopic optimization of our energy cost is consistent with this widespread reciprocity mechanism.

What happens if we iterate on the above deterministic algorithm, each peer successively updating its configuration based on the tit-for-tat like reciprocity scheme? In general, it is difficult to characterize the behavior of such dynamics over a discrete set of configurations. The trajectory will depend on initial conditions, and there is no reason to expect the global energy-minimizing configuration will be found.

For example, the initial file-exchange may break the graph into components, leaving some peers disconnected from their optimal neighbors; these will never be discovered by the above deterministic reciprocity. This suggests that a certain amount of random exploration is required. BitTorrent addresses this issue through the optimistic unchoke portion; however this egalitarian file-sharing implies an important deviation from proportionality. An alternative is studied in the following section.

4.4 INTRODUCING RANDOMNESS

In this section we want to develop a neighbor selection algorithm for the peers in which each peer will update its connections periodically, taking into account the download rates from other peers and introducing some randomness in order to explore the peering options. The idea is not to use an optimistic connection chosen at random, but instead to introduce a more directed randomness. In order to do this, we will present a probability distribution over the set of all configurations and construct a Markov chain whose invariant distribution is that one, with transitions that can be made in a decentralized fashion. In the end, the algorithm executed by the peers will be the one that produces the designed Markov chain and thus will present the desired invariant distribution, which will be concentrated around the symmetric allocations.

4.4.1 Choosing the invariant distribution

A good choice for the invariant distribution of our Markov chain is the so called Gibbs distribution for the energy function \mathcal{E} :

$$\pi_T(X) = \frac{\exp\left(-\frac{1}{T}\mathcal{E}(X)\right)}{C_T} = \frac{\exp\left(-\frac{1}{2T} \sum_{i,j \in S} (\mu_i x_{ij} - \mu_j x_{ji})^2\right)}{\sum_{X' \in \Lambda^S} \exp\left(-\frac{1}{2T} \sum_{i,j \in S} (\mu_i x'_{ij} - \mu_j x'_{ji})^2\right)}$$

where $T > 0$ is a parameter which is called the temperature of the distribution. The idea behind this type of distributions comes from the field of statistical mechanics, which studies physical systems (usually of particles) and their interactions. Those systems usually tend to be in the state of minimum energy. However, this minimum energy state can be altered by thermal energy that introduces randomness. In this case, we are still close to the minimum energy state if the “temperature” is low enough as is shown in the following proposition.

Proposition 4.4.1. $\lim_{T \rightarrow 0^+} \pi_T = \sum_{i=1}^K \frac{1}{K} \delta_{X_i^*}$, where $\{X_1^*, \dots, X_K^*\} = \arg \min_{X \in \Lambda^S} \{\mathcal{E}(X)\}$.

Proof. We have that

$$\pi_T(X) = \frac{\exp\left(-\frac{1}{T}\mathcal{E}(X)\right)}{\sum_{X'} \exp\left(-\frac{1}{T}\mathcal{E}(X')\right)}$$

When T goes to 0^+ , the sum in the denominator is equivalent to the sum of the biggest terms, i.e. the terms with minimum energy. Suppose that there are K

configurations with minimum energy (there is at least one because the sum is finite), then

$$\pi_T(X) \approx \frac{\exp\left(-\frac{1}{T}\mathcal{E}(X)\right)}{\sum_{i=1}^K \exp\left(-\frac{1}{T}\mathcal{E}(X_i^*)\right)}$$

When T goes to 0^+ , the right hand side goes to 0 if the numerator does not corresponds to a configuration of minimum energy and goes to 1 if it does. \square

With this proposition we see that the Gibbs distribution π_T is a thermodynamic approximation of a distribution that is concentrated around the configurations of minimum energy (Figure 4.5). Actually, as seen in [18], the Gibbs distribution with parameter $T > 0$ is the solution of the following optimization problem

$$\begin{aligned} \min_p \quad & \sum_{X \in \Lambda^S} p_X \mathcal{E}(X) + T \sum_{X \in \Lambda^S} p_X \log(p_X) \\ \text{s.t.} \quad & \sum_{X \in \Lambda^S} p_X = 1 \\ & p_X \geq 0 \quad \forall X \end{aligned}$$

The function that we are minimizing is the previous energy plus an entropy term, which has greater impact over the minimum as T increases. Note that the minimum in the entropy term is achieved when the probability distribution is uniform.

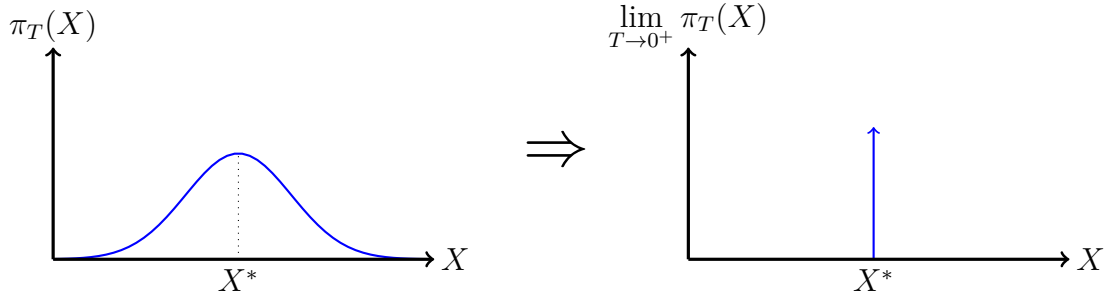


Figure 4.5. Thermodynamic approximation of the optimization problem.

Until now we showed that π_T is an appropriate probability distribution for the configurations since when the temperature T is close to zero, it is concentrated around the configurations of minimum energy, which are the symmetric allocations. At this point we set up to construct a Markov chain with invariant distribution π_T but with transitions that can be done in a decentralized fashion.

Remark 4.4.1. A Markov chain with a decentralized implementation exists due to the fact that the energy \mathcal{E} can be decomposed as a sum of local energies that only depend on local information. This type of energies are said to come from *nearest neighbors Gibbs potentials* in the literature of Gibbs distributions. See [19] for further reading on this subject.

We remark at this point that in recent work by [17, 16], it was proposed to use this kind of approach for a P2P network utility maximization problem, and it

was argued that this “reverse engineered” BitTorrent. In this regard, we make the following remarks:

- The energy function used in the approach of [17, 16] is defined in terms of a network utility, aimed more at performance than at fairness. This would have impact in a situation where the rate of upload of peer i is not equivalent for all peers j , due to other network bottlenecks.
- The dynamics proposed in these references implies *choking* one of the current peers and replacing by a new one; the peer most likely to be choked is the one with lowest current rate *to* it in the *upload* sense. Such a rule is in fact consistent with the algorithm for *seeders* in the BitTorrent protocol (peers who already own the file). It is different, however, to a reciprocity scheme based on *download* rates received *from* other peers, as in the tit-for-tat mechanism used by *leechers*. The latter is the focus of our work, and so our proposal will be complementary to these references.

4.4.2 Building the Markov chain

Now that we have a desired invariant distribution for the space of the configurations, we want to build a decentralized algorithm on the nodes that yields this distribution, i.e. an algorithm for the peers to follow which will yield a symmetric allocation (approximately) in the network. For this purpose we build a continuous time Markov chain that has the desired distribution and that can be implemented in a decentralized way.

The only transitions that are admissible in the chain are between configurations X and X' that only differ in one row, i.e., in the connections of one peer. In particular, if they differ in row i , the transition rates are

$$q_{X,X'}^T = \frac{\exp\left(-\frac{1}{T} \sum_{j \in S} (\mu_i x'_{ij} - \mu_j x_{ji})^2\right)}{\sum_{X'' \in \Lambda_i^S(X)} \exp\left(-\frac{1}{T} \sum_{j \in S} (\mu_i x''_{ij} - \mu_j x_{ji})^2\right)} \quad (4.3)$$

where $\Lambda_i^S(X) = \{X'' \in \Lambda^S : x''_{ij} = x_{ij}, \forall i \neq i, \forall j\}$ is the set of all possible configurations that can be reached from X changing only row i .

Proposition 4.4.2. *The Markov chain defined by 4.3 is reversible and the invariant distribution is π_T .*

Proof. We prove that the invariant distribution and the transition rates verify the detailed balance equations

$$q_{X,X'}^T \pi_T(X) = q_{X',X}^T \pi_T(X')$$

$$\begin{aligned}
q_{X,X'}^T \pi_T(X) &= \frac{\exp\left(-\frac{1}{T} \sum_{j \in S} (\mu_{i_0} x'_{i_0 j} - \mu_j x_{j i_0})^2\right) \exp\left(-\frac{1}{2T} \sum_{i,j \in S} (\mu_i x_{ij} - \mu_j x_{ji})^2\right)}{\sum_{X'' \in \Lambda_{i_0}^S(X)} \exp\left(-\frac{1}{T} \sum_{j \in S} (\mu_{i_0} x''_{i_0 j} - \mu_j x_{j i_0})^2\right) C_T} \\
&= \frac{\exp\left(-\frac{1}{2T} \left(2 \sum_{j \in S} (\mu_{i_0} x'_{i_0 j} - \mu_j x_{j i_0})^2 + 2 \sum_{j \in S} (\mu_{i_0} x_{i_0 j} - \mu_j x_{j i_0})^2 + \sum_{i \neq i_0} \sum_{j \neq i_0} (\mu_i x_{ij} - \mu_j x_{ji})^2\right)\right)}{\sum_{X'' \in \Lambda_{i_0}^S(X)} \exp\left(-\frac{1}{T} \sum_{j \in S} (\mu_{i_0} x''_{i_0 j} - \mu_j x_{j i_0})^2\right) C_T} \\
&= \frac{\exp\left(-\frac{1}{2T} \left(2 \sum_{j \in S} (\mu_{i_0} x'_{i_0 j} - \mu_j x'_{j i_0})^2 + 2 \sum_{j \in S} (\mu_{i_0} x_{i_0 j} - \mu_j x'_{j i_0})^2 + \sum_{i \neq i_0} \sum_{j \neq i_0} (\mu_i x'_{ij} - \mu_j x'_{ji})^2\right)\right)}{\sum_{X'' \in \Lambda_{i_0}^S(X)} \exp\left(-\frac{1}{T} \sum_{j \in S} (\mu_{i_0} x''_{i_0 j} - \mu_j x'_{j i_0})^2\right) C_T} \\
&= \frac{\exp\left(-\frac{1}{2T} \sum_{j \in S} (\mu_{i_0} x_{i_0 j} - \mu_j x'_{j i_0})^2\right) \exp\left(-\frac{1}{2T} \sum_{i,j \in S} (\mu_i x'_{ij} - \mu_j x'_{ji})^2\right)}{\sum_{X'' \in \Lambda_{i_0}^S(X)} \exp\left(-\frac{1}{T} \sum_{j \in S} (\mu_{i_0} x''_{i_0 j} - \mu_j x'_{j i_0})^2\right) C_T} = q_{X',X}^T \pi_T(X')
\end{aligned}$$

□

Furthermore, we have the following proposition.

Proposition 4.4.3. *The rows of the configurations are updated uniformly at random.*

Proof. We want to know the rate at which we make the transition $X \rightarrow X' \in \Lambda_i^S(X)$, that is, that site i is updated. As all the times are exponential, the distribution of the time of the updates at site i is another exponential whose parameter is simply the sum of all transition rates $q_{X,X'}^T$

$$q_i^T = \sum_{X' \in \Lambda_i^S(X)} q_{X,X'}^T = \sum_{X' \in \Lambda_i^S(X)} \frac{\exp\left(-\frac{1}{T} \sum_{j \in S} (\mu_i x'_{ij} - \mu_j x_{ji})^2\right)}{\sum_{X'' \in \Lambda_i^S(X)} \exp\left(-\frac{1}{T} \sum_{j \in S} (\mu_i x''_{ij} - \mu_j x_{ji})^2\right)} = 1$$

so every site has the same rate of updates and it is independent from the state X . This is equivalent to selecting a random site to update. □

Remark 4.4.2. Note that the transition rates (4.3) only depend on local information available to the peer that changes its connections. Besides, as the times between updates for all nodes are independent of the node which is updated and is independent of the configuration, the algorithm can be executed in a completely decentralized way. This type of Markov chains are also called *Gibbs samplers*, as they are used to draw samples from Gibbs distributions.

Note that this Gibbs sampler, which was constructed with the invariant distribution as a starting point, has transitions that are a thermodynamic approximation of the myopic policy considered in the previous section.

Proposition 4.4.4. $\lim_{T \rightarrow 0^+} q_{X, X'}^T = \sum_{k=1}^K \frac{1}{K} \delta_{X_k^*}$, where $\{X_1^*, \dots, X_K^*\} = \arg \min_{X' \in \Lambda_i^S(X)} \{\mathcal{E}_i(X') = \sum_{j \in S} (\mu_i x'_{ij} - \mu_j x_{ji})^2\}$.

Proof. It is analog to Proposition 4.4.1. □

This shows that when the temperature goes to zero, the transitions go to the deterministic myopic policy of minimizing the local energy. However, for every positive value of T , there is still some randomness involved that makes it work as expected. Eventually, as it has finite state space and is irreducible, the Markov chain will converge to the invariant distribution π_T .

Now we present the neighbor selection algorithm based on the previous Gibbs sampler. If every peer executes this algorithm, then the network as a whole will behave as the Markov chain and as a result we will achieve a symmetric resource allocation (approximately).

Gibbs neighbor selection algorithm 1

Suppose that we have a P2P network with N nodes and connectivity graph G . Then repeat the following:

- Step 1: Draw an exponential random time with mean 10 seconds (or any fixed amount of time) and wait that amount of time.
- Step 2: While waiting, measure the average throughputs of all incoming connections $\{z_j : j \text{ is a neighbor}\}$.
- Step 3: After the time expires, unchoke a set of N_0 neighbors (represented as the 0-1 vector $X_i = (x_{ij} : j \text{ is a neighbor})$) according to the probability distribution

$$p(X_i) = \frac{\exp\left(-\frac{1}{T} \sum_j \left(\frac{\mu_i}{N_0} x_{ij} - z_j\right)^2\right)}{C_T}$$

where μ_i is the upload capacity of the peer and C_T is a normalizing constant.

Note that every peer is choosing its N_0 neighbors to unchoke based on the local energy, and as a result the neighbors that have higher throughput connections have

higher probability of being unchoked. Besides, as $T > 0$ there is a positive probability that a peer chooses to connect to a peer which was not connected before. However, the advantage of this approach is that if a peer is in a state of very low energy, it has a very high probability of not changing connections; but if it is in a state of high energy, it has a higher probability of exploring other peering options. This directed randomness is what will make it perform better than the alternatives.

4.4.3 Deterministic update times

In practice, the times between updates may not be an exponential random time but a deterministic one. For example, in BitTorrent's neighbor selection algorithm, each peer updates its connections every 10 seconds. If we substitute the exponential times between updates by deterministic ones in the previous algorithm, the resulting stochastic process defined over the configurations is no longer a Markov chain. However, we can still analyze the outcome of such algorithm by a discrete time Markov chain whose transitions involve the connections of all the peers at the same time, after the sequential update of all the peers. This representation is appropriate as the time needed to update all peers is deterministic, and thus we can chose it as our time unit.

The transition matrix $P_T = (p_{X,X'}^T)$ for our new discrete time Markov chain is obtained by multiplying the transition probabilities of each peer as the sequence goes along

$$p_{X,X'}^T = \prod_{i=1}^N \frac{e^{-\frac{1}{T}\mathcal{E}_i(X,X')}}{Z_i^T},$$

where

$$\mathcal{E}_i(X, X') = \sum_{j=1}^{i-1} (\mu_i x'_{ij} - \mu_j x'_{ji})^2 + \sum_{j=i}^N (\mu_i x'_{ij} - \mu_j x_{ji})^2,$$

and Z_i^T are appropriate normalizing constants. $\mathcal{E}_i(X, X')$ reflects the local energy of the i -th intermediate configuration when transiting between X and X' , which now can be any pair of configurations. This process is also called *systematic sweep Gibbs sampler* and it is a standard variation of the previous Gibbs sampler (see [19] for further reading on Gibbs samplers).

The Markov chain defined before has a finite state space and is irreducible and aperiodic, thus it eventually converges to its invariant distribution, which can be shown to be equal to π_T . Furthermore, in this case we can bound the speed of convergence.

Proposition 4.4.5. *Let η be the initial distribution and $P = (p_{X,X'})$ be the transition matrix of the discrete-time Markov chain. Denote by $d_V(\cdot, \cdot)$ the total variation distance between two probability distributions. If the network is full mesh, then*

$$d_V(\eta P^n, \pi_T) \leq \frac{1}{2} d_V(\eta, \pi_T) \left(1 - \exp \left(- \sum_i \frac{2\mu_i \mu_{\max}}{T N_0} \right) \right)^n.$$

Proof. Using theorem 7.2 of chapter 6 in [19], we have

$$d_V(\eta P^n, \pi_T) \leq \frac{1}{2} d_V(\eta, \pi_T) \delta(P)^n$$

where $\delta(P)$ is the Dobrushin's ergodic coefficient

$$\begin{aligned} \delta(P) &= \frac{1}{2} \max_{X, X'} \left\{ \sum_{X''} |p_{X, X''} - p_{X', X''}| \right\} \\ &= \max_{X, X'} \left\{ 1 - \sum_{X''} \min\{p_{X, X''}, p_{X', X''}\} \right\} \\ &= 1 - \min_{X, X'} \left\{ \sum_{X''} \min\{p_{X, X''}, p_{X', X''}\} \right\} \end{aligned}$$

Now, let's define $\Lambda_i^S(X, X')$ as the set of all configurations $X'' \in \Lambda^S$ such that the row j of X'' , denoted X''_j satisfies $X''_j = X'_j$ if $j < i$ and $X''_j = X_j$ if $j > i$. That is, $\Lambda_i^S(X, X')$ is the set of all possible configurations that can be reached in step i in the partial transition between X and X' . Let

$$m_i(X, X') = \min_{X'' \in \Lambda_i^S(X, X')} \{\mathcal{E}_i(X, X'')\}$$

and

$$M_i(X, X') = \max_{X'' \in \Lambda_i^S(X, X')} \{\mathcal{E}_i(X, X'')\}$$

be the minimum and maximum local energy at step i when making the transition from X to X' . Then the transition probabilities of step i can be expressed as

$$p_{X, X'}^i = \frac{\exp\left(-\frac{1}{T}(\mathcal{E}_i(X, X') - m_i(X, X'))\right)}{\sum_{X'' \in \Lambda_i^S(X, X')} \exp\left(-\frac{1}{T}(\mathcal{E}_i(X, X'') - M_i(X, X'))\right)}$$

We can bound from above each term of the sum in the denominator by 1 and thus the sum by the number of terms that is $\#\Lambda$. Furthermore, we can bound from below the numerator by

$$\exp\left(-\frac{1}{T}(M_i(X, X') - m_i(X, X'))\right)$$

To find an appropriate bound, we need to find the maximum possible difference $M_i(X, X') - m_i(X, X')$. When there are no peers connecting to peer i , the local energy does not change choosing different peers to upload content and this difference would be zero. As a result, the biggest difference occurs when at least N_0 peers are connected to i . In this case, we have that $M_i(X, X') - m_i(X, X')$ is

$$\frac{1}{N_0^2} \left(\sum_{k=1}^{N_0} \mu_i^2 + \sum_{k=1}^{N^{in}} \mu_{j_k}^2 - \sum_{k=1}^{N_0} (\mu_i - \mu_{j_k})^2 - \sum_{k=N_0+1}^{N^{in}} \mu_{j_k}^2 \right)$$

and doing some algebra we obtain

$$\begin{aligned}
M_i(X, X') - m_i(X, X') &= \frac{1}{N_0^2} \sum_{k=1}^{N_0} 2\mu_i \mu_{j_k} \\
&\leq \frac{1}{N_0^2} \sum_{k=1}^{N_0} 2\mu_i \mu_{max} \\
&= \frac{2\mu_i \mu_{max}}{N_0}
\end{aligned}$$

This results in the following uniform bound

$$p_{X, X'}^i \geq \frac{\exp\left(-\frac{2\mu_i \mu_{max}}{TN_0}\right)}{\#\Lambda}$$

Then, the minimum element in the matrix P is

$$\begin{aligned}
\min_{X, X' \in \Lambda^S} \{p_{X, X'}\} &= \min_{X, X' \in \Lambda^S} \left\{ \prod_{i=1}^N p_{X, X'}^i \right\} \\
&\geq \prod_{i=1}^n \frac{\exp\left(-\frac{2\mu_i \mu_{max}}{TN_0}\right)}{\#\Lambda} \\
&\geq \frac{\exp\left(-\sum_{i=1}^N \frac{2\mu_i \mu_{max}}{TN_0}\right)}{(\#\Lambda)^N}
\end{aligned}$$

and with this we obtain the following expression

$$\begin{aligned}
\delta(P) &= 1 - (\#\Lambda)^N \frac{\exp\left(-\sum_{i=1}^N \frac{2\mu_i \mu_{max}}{TN_0}\right)}{(\#\Lambda)^N} \\
&= 1 - \exp\left(-\sum_{i=1}^N \frac{2\mu_i \mu_{max}}{TN_0}\right)
\end{aligned}$$

which concludes the proof. \square

Remark 4.4.3. Note that when T is closer to 0, the speed of convergence bounded by Proposition 4.4.5 gets slower but the invariant distribution π_T is more concentrated on the configurations of minimum energy (Proposition 4.4.1). On the other hand, when T goes to infinity, the speed of convergence also goes to infinity, but the invariant distribution becomes independent from the energy (uniform). This is the fundamental tradeoff that we have in this system.

Finally, we can present the second version of the neighbor selection algorithm with deterministic updates times, which is essentially systematic sweep Gibbs sampler. This is the one that will be implemented.

Suppose that we have a P2P network with N nodes and connectivity graph G . Then repeat the following:

- Step 1: Wait 10s (or any fixed amount of time).
- Step 2: While waiting, measure the average throughputs of all incoming connections $\{z_j : j \text{ is a neighbor}\}$.
- Step 3: After the time expires, unchoke a set of N_0 neighbors (represented as the 0-1 vector $X_i = (x_{ij} : j \text{ is a neighbor})$) according to the probability distribution

$$p(X_i) = \frac{\exp\left(-\frac{1}{T} \sum_j \left(\frac{\mu_i}{N_0} x_{ij} - z_j\right)^2\right)}{C_T}$$

where μ_i is the upload capacity of the peer and C_T is a normalizing constant.

4.5 SIMULATIONS

We now evaluate the devised Gibbs neighbor selection algorithm 2 as a means to achieve reciprocity and fairness. We implemented the algorithm in Matlab, and in order to perform comparisons, we also implemented idealized versions of the BitTorrent unchoking mechanism, as well as the ideal proportional reciprocity based on the Sinkhorn iteration discussed, the PropShare unchoking algorithm of [3] and the Markov approximation approach devised in [16].

Let us begin by briefly recalling the different algorithms. The standard BitTorrent unchoking mechanism maintains for each peer $N_0 = 4$ outgoing connections. Three of these connections are used to reciprocate other peers, and the remaining connection is an *optimistic* unchoke, designed to explore new peers. The latter is kept for several iterations in order to allow time for the optimistically unchoked peer to reciprocate. This algorithm has low overhead and enables peers to find appropriate partners [6], but it has two main disadvantages: the unchokes are based only on the ranking of better contributors, and not in the bandwidth they provided, which has incentives problems [3]. It also constantly searches for new peers, allocating a substantial proportion of the uplink bandwidth to this end, and possibly drifting away from good configurations.

The Sinkhorn algorithm, on the other hand, focuses on reciprocating only, by allocating proportional shares to each connected peer. To this end, it is the best one can do and achieves a fast convergence time. A pure proportional response however, has two main drawbacks from a practical perspective: it requires to keep a large amount of connections with several peers, as well as controlling exactly the amount

of bandwidth allocated to each unchoked peer, which may be difficult to implement in practice. More importantly, it can get stuck in bad configurations if the initial connectivity of peers is sparse.

The PropShare algorithm is based on the Sinkhorn iteration, and was devised to correct this last problem, among other optimizations. This algorithm allocates proportionally to the received contributions 80% of the uplink bandwidth of a given peer. It uses the remaining 20% to explore new peers through optimistic unchokes, much like BitTorrent. This exploration mechanism enables the algorithm to increase the number of connected peers. While this may achieve a higher level of fairness, the bandwidth committed to the optimistic search can make the algorithm drift away from good configurations. This algorithm still suffers from the burden of maintaining many connections and controlling the amount of bandwidth given to each. In our simulations we implemented an idealized version of PropShare based on these features, not taking into account these problems, and thus we expect our results to provide an upper bound on real life PropShare performance.

Finally, the Markov approximation algorithm of [16] has points in common with our proposal. However, the main emphasis is on achieving optimal throughput allocation by discovering the best neighbors to *upload* to. Reciprocity is not taken into account and therefore it suffers on the fairness side, as we will show.

To evaluate the algorithms, we simulated a scenario with $N = 100$ peers, which belong to two categories: half of the peers have a fast uplink connection, and the other half are 5 times slower. Ideally, all peers should get as much bandwidth as they give to achieve proportional fairness. All peers can potentially connect to each other, and in the case of our algorithm and BitTorrent, dividing bandwidth equally between all outgoing connections. All algorithms start from the same initial connectivity condition with $N_0 = 4$ outgoing connections per peer.

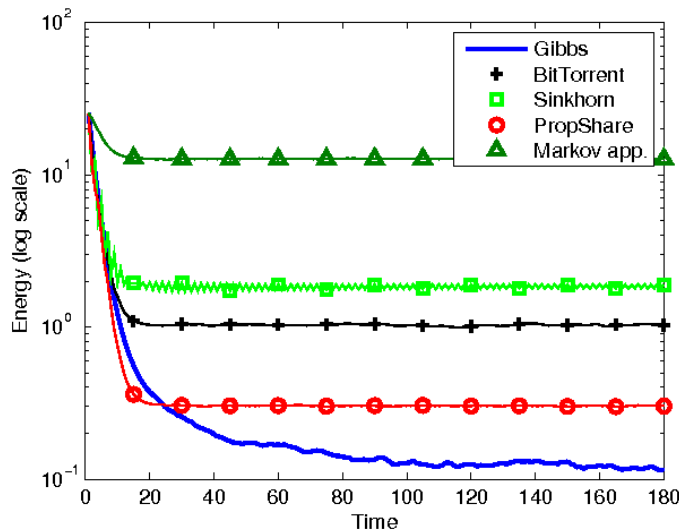


Figure 4.6. Gibbs energy evolution for the different algorithms.

As a measure of the achieved reciprocity and fairness, we evaluate two metrics:

the Gibbs energy $\mathcal{E}(X)$, which is intended to be minimized by our neighbor selection algorithm, and also the Kullback-Leibler (KL) divergence $D(\mu\|r)$ between the offered bandwidths $\mu = \{\mu_i\}$ and the rates received by each peer $r = \{r_i\}$, which is defined by

$$D_{KL}(\mu\|r) = \sum_i r_i \log \left(\frac{r_i}{\mu_i} \right)$$

Note that the KL divergence takes its minimum only when $r = \mu$, in which case $D_{KL}(\mu\|r) = 0$.

In order to correctly assess the performance of each algorithm, we simulate several replications of each one of them starting from a random initial condition, and plot the average results for each metric. In Figure 4.6, we plot the evolution of the Gibbs energy $\mathcal{E}(x)$ for the different algorithms in log scale.

Our neighbor selection algorithm (with $N_0 = 4$ in this case for comparison) is designed to find a minimum of the energy and it does so in a competitive number of iterations, at the expense of a convergence time somewhat slower than the remaining algorithms. The Markov approximation algorithm is not good at achieving reciprocity, remaining with a high energy. The (ideal) Sinkhorn iteration is theoretically the best, but when facing random initial conditions with sparse connectivity the algorithm cannot fully renormalize the allocation, and thus it does not reach minimum energy. The BitTorrent algorithm is assigning too many optimistic unchokes, and this reflects on the energy achieved. Finally, PropShare is the best alternative, at the expense of having a greater number of simultaneous connections for each peer, and controlling bandwidth on each one of them. Our neighbor selection algorithm achieves a better reciprocity while at the same time having only $N_0 = 4$ open connections per peer sharing the uplink rate equally.

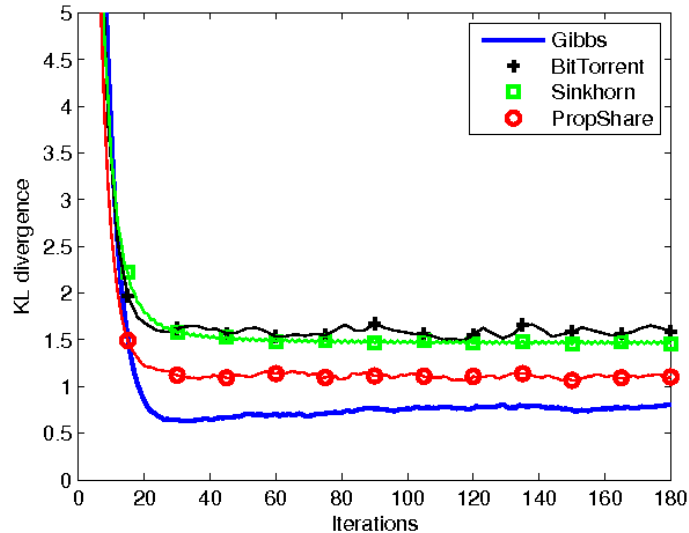


Figure 4.7. Evolution of the Kullback-Leibler divergence between uplink and received rate.

As for the fairness in the resulting allocation, in Figure 4.7 we plot the aforementioned KL divergence for each algorithm. In this case we omit the Markov

approximation algorithm since it does not pursue proportional fairness. Note that also for this metric the best algorithms are the proposed neighbor selection algorithm and the PropShare algorithm, with the proposed one achieving better results.

4.6 CONCLUSIONS

In this chapter we proposed a decentralized neighbor selection algorithm based on a Markov chain that approximates the proportional allocation while being easier to implement than the proportional response dynamics, reaching a middle ground between the simplicity of BitTorrent and the fairness and incentives of the proportional response dynamics. Moreover, we explored through simulations how the new algorithm compares to several alternatives in an heterogeneous P2P environment, showing good results.

Chapter 5

Extension to wireless P2P networks

In this chapter we will work with a P2P overlay over an ad-hoc wireless network, conceived as a system of nodes that establish transmission links based on proximity and communicate with each other only in single-hop fashion. The shared nature of the wireless medium accounts for the need of contention mechanisms by which nodes control their transmission attempts. We will focus on the rate allocations in this type of networks and specially in three deeply linked properties of these allocations:

- The efficiency in the use of the wireless channel measured as the total throughput of the network.
- The incentives for the peers to contribute to the network.
- The connection diversity of the peers.

To begin with, we present the wireless network setting and find all the feasible rate allocations in the network, showing that in this case we can always find reciprocated allocations (i.e. allocations in which each peers receives as much as it gives). Then we propose a decentralized algorithm, based on a continuous time backoff processes similar to that of CSMA in the 802.11 standard, that yields a symmetric rate allocation and whose efficiency-diversity tradeoff can be tuned by a single parameter. The idea of using continuous time backoff processes was previously used in [20] but with a different purpose.

One mechanism from 802.11 that we will take full advantage of, is its solution for the exposed/hidden node problem: RTS/CTS messages. The idea is that when a node wants to transmit to another one, it transmits a Request To Send (RTS) message to the other. If the latter can receive the message, it responds with a Clear To Send (CTS) message and the transmission begins. When another node listens to

a CTS message, it knows that the medium is occupied for a certain amount of time, and it will not participate in any other transmission during that time.

The problem of rate allocation for P2P content distribution over an ad-hoc wireless network has caught the attention of numerous researchers lately. In [21] the authors develop a token-passing based multi-point relays scheme that presents an efficient scheduling approach to disseminate content. Furthermore, in [22] they present a synchronous P2P wireless network architecture that focuses on finding the optimal allocation in terms of bandwidth efficiency. Lastly, in [23] the authors focus on the actual fairness of the rate allocations.

5.1 NETWORK SETTING

We will focus in the setting where peers are spatially distributed and want to share information or content through a wireless channel. The exchange of content is based in the interchange of pieces of the file, whose availability also plays a role in determining the connectivity graph. We will assume that there is a unique file to be downloaded, whose size is very large and is split in small pieces. As a result, we have a long download time and a very large amount of pieces. Furthermore, we assume that the peers use the “rarest first” approach when requesting pieces from other peers. This, combined with the large amount of pieces, gives us the possibility of ignoring the piece by piece exchange and focus on the rate allocation. Lastly, we assume that we can make a separation of time scales in which the change in the network parameters (such as the arrival or departure of peers) is much slower than the proposed exchange dynamics.

The constraints in this problem are quite different from the wired case. In the former, each peer has only to decide how to allocate its own upload bandwidth by choosing its neighbors and the resulting allocation is independent of the decision of other peers, because they do not compete with each other. Besides, the total upload capacity of the network is just the sum of the upload capacities of every peer (which is fixed) for the same reason. Now in the wireless case, the peers are sharing a common medium (the RF channel) and their decisions affect the other peers. Furthermore, the total upload capacity of the network is no longer fixed, but depends on the decisions of all the peers due to the interference between them. Lastly, now the bandwidth restrictions are in the links instead of in the nodes, as each pair of source-destination will have its own link capacity.

If we have spatially distributed nodes, we could build the connectivity graph G by connecting every pair of peers which can transmit directly to each other. Finally, the maximum possible transmission rate from peer i to peer j , which we denote by μ_{ij} , will depend on the transmission power, distance and fading in the channel. Note that the transmissions rates need not be symmetric. In general, we will assume the following hypotheses for the wireless network:

- Fixed number of wireless peers N .
- Fixed topology defined by an undirected connectivity graph G with adjacency matrix A .
- Single hop connectivity (each peer connects only to the ones within range).
- General bandwidth $N \times N$ matrix $\mu = (\mu_{ij})$, where μ_{ij} is the maximum possible transmission rate from peer i to peer j (it can be non symmetric due to different transmission powers).
- Use of RTS/CTS messages.

Note that the connectivity graph is undirected due to the use of RTS/CTS messages, which makes it necessary for both to be in range of each other. In this general network setting, we want to find all the possible rate allocations Z . In order to do this, we begin by defining the interference graph.

5.1.1 Interference graph

Interference plays a fundamental role in any wireless network. We will use an interference model based on proximity (e.g., a peer interferes another if they are within range of transmission). All the interference information under this model can be condensed in an interference graph.

Let i, j be a pair of nodes and $\mathcal{N}(i), \mathcal{N}(j)$ their respective neighbors under the connectivity graph G , i.e. $\mathcal{N}(i), \mathcal{N}(j)$ are all the peers that are within range of i and j respectively. Suppose that they are neighbors, i.e. $i \in \mathcal{N}(j)$ and $j \in \mathcal{N}(i)$. Then when the link ij is active (node i is transmitting to node j), nodes $k, l, o \in \mathcal{N}(i) \cup \mathcal{N}(j)$ cannot be transmitters nor receivers at that time (as in figure 5.1). That means that the link ij interferes with the links xy and yx for all $x \in \mathcal{N}(i) \cup \mathcal{N}(j)$ and $y \in \mathcal{N}(x)$. This is a little more restrictive than it needs to be, because the node o could be a receiver of a connection from peer m because m is not a neighbor of j (they will not interfere with each other). This is the interference pattern that arises when one uses RTS/CTS. Peer o will not send a CTS signal to m because it heard that node j sent a CTS signal.

Given the connectivity graph G and the aforementioned interference rules, we can build the interference graph G_I in which each node is a link, and two nodes are connected if the links interfere with each other (Figure 5.2). First of all, note that this is an undirected graph because the interference works both ways, if link ij interferes the link kl then link kl interferes with link ij due to the RTS/CTS messages.

An interesting fact about the interference in this wireless network is that when link ij is active, it interferes with exactly the same links as when link ji is active. We

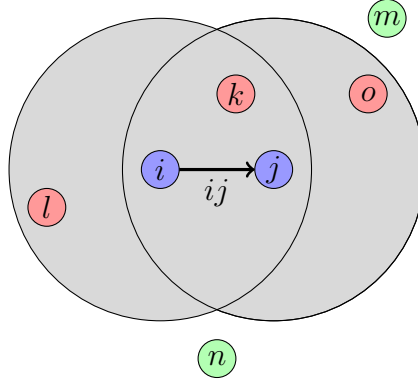


Figure 5.1. Interference diagram of link ij . Nodes in red cannot participate in other transmissions but nodes in green can be a transmitter or a receiver with other nodes.

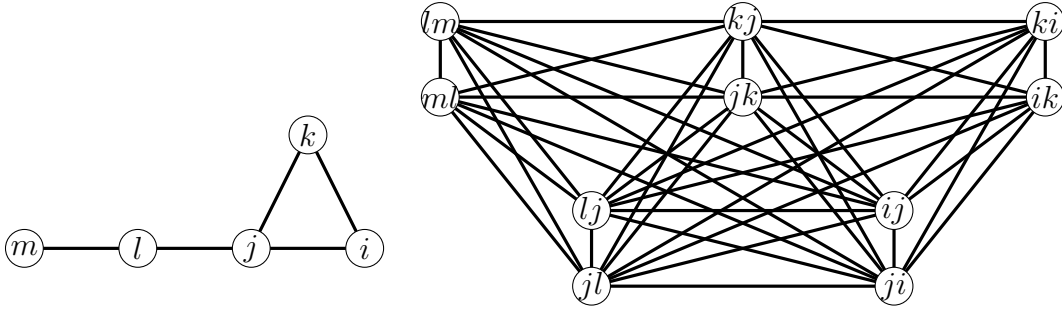


Figure 5.2. The connectivity graph (left) and the interference graph (right).

call this *symmetric interference*, which is a result of using the RTS/CTS messages. Using this fact, we could build a collapsed interference graph in which each pair of links between two peers is now a node, maintaining the same links between them (Figure 5.3). This is actually the resulting graph of deleting half of the nodes in the original interference graph and it is a much simpler representation of the interference between links. In this representation it is clearly seen that the only independent set (i.e. a set of peers that are not connected between them) with more than one link active is $\{(lm), (ik)\}$.

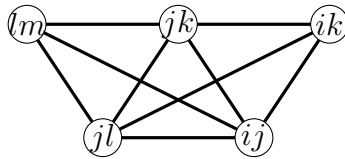


Figure 5.3. The collapsed interference graph.

5.2 RATE ALLOCATIONS

In the wired case, the possible resource allocations were easily given by the union of the allocations of each peer. With a wireless medium, we have to take into account the interference between the links in order to determine which rate allocations are feasible in the network. Note that in the wireless case the total upload rate (resource)

of the network is not fixed, and thus the term “rate allocation” is more appropriate than “resource allocation” as the amount of resources to be allocated is not fixed.

Firstly, we define a *configuration* of the network as a matrix $X \in \{0, 1\}^{N \times N}$ such that the set of entries that are 1 define an independent set in G_I . Note that this means that $x_{ij} = 0$ if $i \notin \mathcal{N}(j)$. These represent the possible active links than can coexist without interfering. Furthermore, we define a *rate allocation* in the network as a $N \times N$ matrix Z such that $z_{ij} \geq 0$ and for which the entry z_{ij} represents the average throughput of link ij . Then, a *feasible rate allocation* is an allocation that can be achieved by the network, e.g. all the allocations such that $Z = \mu \circ X$ where X is a configuration and \circ denotes the Hadamard product of matrices (i.e. the entry wise product). These would be the rate allocations that are a result of a static network with a fixed set of active links.

However, if we consider a dynamic network, we can achieve a greater span of rate allocations. We can consider the stochastic process that represents the dynamics of the network which state space is the set of configurations. If we have a probability p_k of finding the network in the state X_k , then the average rate allocation of the network is

$$\bar{Z} = \sum_k p_k (\mu \circ X_k)$$

As a result, the feasible allocations are the convex hull of the extremal rate allocations $Z_k = \mu \circ X_k$.

5.2.1 Desired rate allocations

Given the set of all feasible rate allocations, a natural question to be asked is which properties should they have. Our objective remains the same as before, to have reciprocated allocations such that the total upload rate of a peer is the same as the total download rate of that peer, i.e.

$$\sum_n z_{ni} = \sum_n z_{in} \quad \forall i$$

The difference is that now the total upload rate of a peer, given by $\sum_n z_{in}$ for peer i , is no longer fixed. In this case, two elements in the set of all feasible reciprocated allocations not only differ in the individual rates z_{ij} but also they may differ in the total upload rate of the network

$$\eta = \sum_{i,j} z_{ij}$$

In this setting it is convenient to introduce the notion of efficiency for a rate allocation as a measure of how close is that allocation of the maximum η possible. This

would be the solution of

$$\begin{aligned}
& \max_Z \quad \sum_{i,j} z_{ij} \\
& \text{s.t.} \quad Z = \sum_k p_k (\mu \circ X_k) \\
& \quad \sum_k p_k = 1 \\
& \quad X_k \text{ is an independent set of } G_I \quad \forall k
\end{aligned}$$

Note that there could be many allocations Z that maximize the efficiency, but they would all be convex combinations of independent sets with maximum weights.

The last desirable property that we will ask from a rate allocation is that it has to have diversity of connections, i.e. that the peers are not always connected to the same peers, to promote the diversity of pieces.

Existence of reciprocated allocations

First of all, we prove the existence of reciprocated allocations through the existence of symmetric ones in the following proposition.

Proposition 5.2.1. *For every rate matrix μ and connectivity graph G , there exists a symmetric resource allocation matrix Z that satisfies the network constraints.*

Proof. We begin with the case of a single link between two peers with rates μ_{12} and μ_{21} . If Z_{ij} is the allocation in which only link ij is active, then we can take the convex combination

$$Z = \frac{\mu_{12}}{\mu_{12} + \mu_{21}} Z_{21} + \frac{\mu_{21}}{\mu_{12} + \mu_{21}} Z_{12}$$

in which each link is active for a certain proportion of time and yields an allocation in which each peer has an average throughput of

$$\frac{\mu_{12}\mu_{21}}{\mu_{12} + \mu_{21}}$$

For general networks we can take convex combinations of this example and build a symmetric resource allocation Z . \square

From this proposition it is clear that finding a symmetric allocation is quite easy.

Example 5.2.1. Suppose that we have a 3 node network with the connectivity graph given by Figure 5.4. In the wired case, if every node had upload capacity 1,

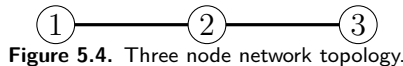


Figure 5.4. Three node network topology.

then it could not be possible to find a symmetric resource allocation as the center peer always receives 2 and cannot give more than 1 (its upload capacity) without introducing idle time. If we consider the wireless network with the same topology in which each directional link has capacity 1, we would have that every node has a maximum effective upload capacity of 1 (just like in the wired case). However, in this case we could let the center node transmit two times more than the others giving us the possibility of a symmetric resource allocation like this

$$\begin{pmatrix} 0 & \frac{1}{4} & 0 \\ \frac{1}{4} & 0 & \frac{1}{4} \\ 0 & \frac{1}{4} & 0 \end{pmatrix}$$

This example shows that despite the fact that the wireless case seems to have more complicated constraints, the shared RF channel helps us achieve symmetric resource allocations in any case without introducing idle time.

Efficiency of reciprocated allocations

Although finding a symmetric allocation is always possible, these allocations could be less efficient than just a reciprocated allocation and at the same time the reciprocated allocations might not achieve the maximum efficiency possible. After all, we would be maximizing the efficiency η over smaller sets of matrices.

Proposition 5.2.2. *In the case of symmetric rate matrices ($\mu = \mu^T$), there is no loss in efficiency if we consider only symmetric allocations.*

Proof. Suppose that Z^* an allocation that achieves the maximum efficiency. Then Z^{*T} is also an allocation that achieves the maximum efficiency because $\mu = \mu^T$ and the interference is symmetric. Then $\frac{Z^* + Z^{*T}}{2}$ is a symmetric allocation that achieves the maximum efficiency. \square

However, if $\mu \neq \mu^T$ we could have a loss in efficiency even by considering reciprocated allocations. For example, in the simple case of only two peers with asymmetrical rates, the unique configuration of maximum throughput is the one where only the fastest one transmits (which is clearly not a reciprocated allocation). Furthermore, there can be a further loss in efficiency by considering symmetric allocations as can be seen in the following example.

Example 5.2.2. Consider the case in which we have three nodes, all connected and with link capacities as in Figure 5.5. Then if each link in the blue clockwise loop is used one third of the time, the resulting allocation would be the most efficient and would be reciprocated as well, as each peer would have the same upload and download rates. However, a symmetric allocation would use the slower counterclockwise loop, which would yield a much less efficient allocation. The problem with the allocation which only uses the clockwise loop is that the peers have no

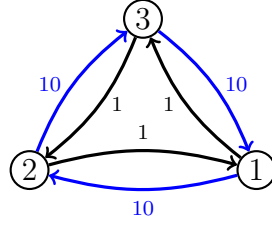


Figure 5.5. A three node network with an efficient loop.

incentive to operate at their full capacity, as they could be almost ten times slower and still have the same download rate as before. This problem comes from the fact that in a reciprocated allocation, the peer that uploads content to another may not know if the other is contributing back to the network. As a consequence, the only way for a peer to know without doubt that it is uploading content to a peer that is contributing back to the network, is having a symmetric allocation.

Taking everything into account, we are going to focus on symmetric allocations and their efficiency and connection diversity, as they are the ones that provide the proper incentives for the peers. In the following sections we are going to develop a decentralized algorithm, first focusing on efficiency and then on producing symmetric allocations.

5.3 MAXIMIZING EFFICIENCY

In this section we will develop a decentralized algorithm that focuses on efficiency for a P2P overlay over an ad-hoc wireless network. The main idea is to keep a continuous time backoff process for each outgoing connection, and regulate the backoff aggressiveness in order to obtain the desired rate allocation.

5.3.1 Markov chain algorithm

Suppose that each peer has knowledge of their local neighbors and of its rates to those peers. With this information, if peer i can transmit (i.e. no peer that interferes with the peer is transmitting), it draws an exponential random variable for each possible outgoing connection with parameter

$$R_{ij} = W_0 \exp\left(\frac{\mu_{ij}}{T}\right)$$

where W_0 is a common aggressiveness factor and T is a parameter that will regulate the efficiency of the resulting allocation. The peer waits for the minimum time drawn and, if no interference link is activated, it starts transmitting through that link. When a link is active, it stays active for a random exponential time of mean

1. During this time, it transmits an exponential amount of data of mean μ_{ij} (where ij is an activated link). We propose that everyone transmits for the same amount of time (in average) because in a wireless setting this ensures that the overheads do not have a stronger impact on faster peers (see [24] and references therein).

We can translate this algorithm into a continuous time Markov chain with state space

$$E = \{X \in \{0, 1\}^{N \times N} : X \text{ defines an independent set in } G_I\}$$

and transition rates

$$q(X, X + e_{ij}) = W_0 \exp\left(\frac{\mu_{ij}}{T}\right) \mathbb{1}_{\{X + e_{ij} \in E\}} \quad (5.1)$$

$$q(X, X - e_{ij}) = \mathbb{1}_{\{X - e_{ij} \in E\}} \quad (5.2)$$

Proposition 5.3.1. *The continuous time Markov chain defined by (5.1) and (5.2) is time reversible and has invariant distribution*

$$\pi_T(X) = \frac{\exp\left(\sum_{ij} x_{ij} \left(\frac{\mu_{ij}}{T} + \log(W_0)\right)\right)}{C_T}$$

where C_T is a normalizing constant.

Proof. We just have to show that π_T verifies the detailed balance equations

$$\pi_T(X)q(X, X + e_{ij}) = \pi_T(X + e_{ij})q(X + e_{ij}, X)$$

or equivalently

$$\frac{\pi_T(X + e_{ij})}{\pi_T(X)} = \exp\left(\frac{\mu_{ij}}{T} + \log(W_0)\right) = \frac{q(X, X + e_{ij})}{q(X + e_{ij}, X)}$$

□

5.3.2 Efficiency vs. diversity of connections

Note that as T approaches zero, the invariant distribution can be approximated by a Gibbs distribution with respect to the energy

$$\mathcal{E}(X) = - \sum_{ij} x_{ij} \mu_{ij}$$

as the constant $\log(W_0)$ becomes negligible. In this case, the configurations of minimum energy would be the independent sets with maximum throughput. As a consequence, when the parameter T goes to zero, the invariant distribution is concentrated in those configurations. Because we are dealing with a P2P content distribution network, in which it is important to connect to different peers, it may

not be a good idea to set T very close to zero, as the network would remain static most of the time and thus it will be disconnected. As a result, if we increase the efficiency we will experience a loss in connection diversity.

On the other hand, if we let T go to infinity, then every link will have the same access rate W_0 . This will lower the efficiency significantly, but at the same time it promotes diversity of connections by giving each link the same transmission aggressiveness. Consequently, we can use the parameter T as a design knob to determine the working point between efficiency and connection diversity.

5.3.3 Reducing idle time

One of the determining factors of the efficiency, as we saw before, is the throughputs of the connections that are active at any given time. The other determining factor is the amount of idle time between those connections. Note that there is idle time when the independent set of links that is active is not maximal. With the following example, we show that for a fixed T the idle time can be reduced to an arbitrarily small amount of time by increasing the common aggressiveness parameter W_0 .

Example 5.3.1. Consider the case in which we have two nodes with link rates μ_{12} and μ_{21} not necessarily equal. Then the network alternates between two states: a state where one of the links is active (which has an exponentially distributed duration of parameter 1), and a state where none of them are. The latter has an exponentially distributed duration of parameter

$$W_0 \left(\exp \left(\frac{\mu_{12}}{T} \right) + \exp \left(\frac{\mu_{21}}{T} \right) \right) = W_0 \beta$$

which is the result of the minimum of the exponential backoffs of both links. Then the average proportion of idle time is

$$\frac{\frac{1}{W_0 \beta}}{1 + \frac{1}{W_0 \beta}} = \frac{1}{1 + W_0 \beta}$$

So we can make the idle time arbitrarily small by increasing the common aggressiveness parameter W_0 . Essentially, this jointly reduces the mean time of all backoff processes (this would be equivalent to reducing the window size in the slotted backoff process of CSMA/CA of the 802.11 standard).

5.3.4 The case of $\mu = \mu^T$

In the particular case where the rate matrix μ is symmetric, we showed in Proposition 5.2.2 that there is no loss of efficiency by considering a symmetric allocation, and in fact this algorithm gives a symmetric allocation in that case.

Proposition 5.3.2. *If the rate matrix μ is symmetric, then the resulting allocation is symmetric for every value of T and W_0 and for every connectivity graph.*

Proof. First of all, note that for every configuration X , there is a complementary configuration X^T such that if link ij is active in X then link ji is active in X^T , due to the symmetric nature of the interference. Then it is enough to prove that each of those pairs of configurations have the same probability. That is, is enough to prove that

$$\frac{\pi_T(X)}{\pi_T(X^T)} = 1$$

for all configurations X .

$$\frac{\pi_T(X)}{\pi_T(X^T)} = \frac{W_0 \exp \left(\sum_{ij: x_{ij}=1} \frac{\mu_{ij}}{T} \right)}{W_0 \exp \left(\sum_{ij: x_{ij}=1} \frac{\mu_{ji}}{T} \right)}$$

and this is equal to one because $\mu = \mu^T$. □

We can conclude that this algorithm yields rate allocations that can be made extremely efficient or more diverse by tuning the parameter T . Furthermore, when the rate matrix μ is symmetric it yields a symmetric allocation independently of any other network parameter. However, when $\mu \neq \mu^T$ we obtain non symmetric allocations. This will be resolved in the following section by taking the peerwise imbalances into account.

5.4 DECENTRALIZED ALGORITHM FOR RECIPROCITY

In order to obtain symmetric allocations even in the case of asymmetric rate matrices, each peer will keep track of the imbalance between itself and all its neighbors through the variables

$$d_{ij}(t) = \int_0^t z_{ij}(u) - z_{ji}(u) du$$

Globally, we obtain a skew-symmetric matrix $D(t)$ with the differences between the total uploaded and downloaded bytes for each link. In practice, every peer only has the information of the column (and row) in which is involved. For the algorithm to be decentralized, it can only use that local information.

With this information, if peer i can transmit, it has to decide which neighbor to serve. Intuitively, when the peer has received less from a peer than it has given, it should decrease the probability of unchoking that peer next, to encourage it to upload more content. Alternatively, when the peer has received more from a peer

than it has given, it should increase the probability of unchoking that peer next, in order to restore the balance and to increase the probability of being unchoked again. Although this last part is not essential to introduce the proper incentives, it conveys the idea of cooperation between peers.

Remark 5.4.1. A way of interpreting the variables that keep track of imbalances is as a closed network of fluid queues in which the total amount of “fluid” is zero, and we can have positive and negative amounts in each queue. In this sense the scheduling that we are looking for is similar to a max-weight scheduling [25], in which we should serve the queue with the most amount of fluid (imbalance) with the highest probability.

Markovian exponential backoff algorithm

Using the same idea of an exponential backoff process as before, the transition rate to activate the link ij should increase when $d_{ij}(t)$ decreases and go to zero when $d_{ij}(t)$ increases. With this in mind, we define the time dependent link transmission aggressiveness as

$$R_{ij}(t) = W_0 \exp\left(\frac{\mu_{ij}}{T} - \alpha d_{ij}(t)\right)$$

where W_0 and T are the same as before and α is a parameter that regulates how forgiving is a peer of current peerwise imbalance.

We could model this as a Markov process augmenting the state to include the imbalance matrix D . Unfortunately this model is not very tractable. Instead we will use an argument of separation of time scales in order to study the long run behaviour of the system.

Dynamics with separation of time scales

Consider the stochastic closed loop defined by Figure 5.6. The transition rates of the Markov chain are regulated by the closed loop that has the integrator, which yields stochastic dynamics in $X(t)$, $Z(t)$ and $D(t)$.

In order to analyze the resulting dynamics, we make the assumption that we have two separate time scales. That is, we assume that the convergence of the Markov chain is much faster than the dynamics of the closed loop with the integrator. In that case, we can analyze the Markov chain for a fixed D which has transition rates

$$\begin{aligned} q(X, X + e_{ij}) &= W_0 \exp\left(\frac{\mu_{ij}}{T} - \alpha d_{ij}\right) \mathbb{1}_{\{X + e_{ij} \in M\}} \\ q(X, X - e_{ij}) &= \mathbb{1}_{\{X - e_{ij} \in M\}} \end{aligned}$$

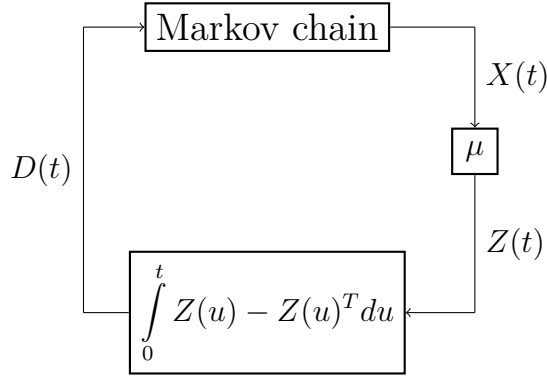


Figure 5.6. Stochastic closed loop for the imbalances.

Again we have that this is a time reversible Markov chain with invariant distribution

$$\pi(X) = \frac{\exp \left(\sum_{ij} x_{ij} \left(\frac{\mu_{ij}}{T} - \alpha d_{ij} + \log(W_0) \right) \right)}{C_T}$$

where C_T is a normalizing constant. The proof is analog to the one of Proposition 5.3.1.

Using the separation of time scales, we can take the average over the stochastic part (which is the same as the expected value in this case, because the chain is ergodic) and redefine d_{ij} as

$$d_{ij}(t) = \int_0^t \mathbb{E}[z_{ij}(u) - z_{ji}(u)] du$$

and we can analyze the equilibrium and convergence of the dynamics of $D(t) = (d_{ij}(t))$ given by the closed loop of Figure 5.7.

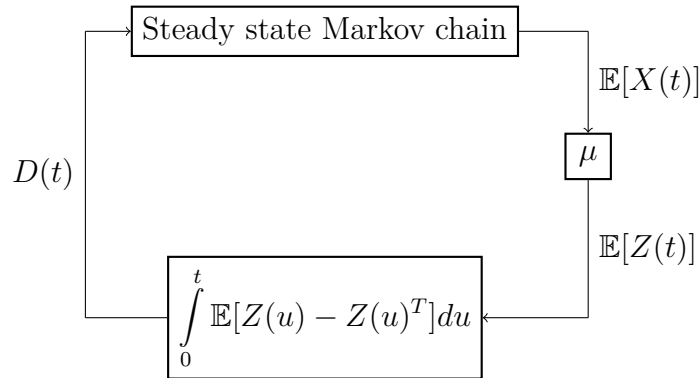


Figure 5.7. Closed loop for the imbalances after the separation of time scales.

Taking the derivative with respect to time, we get

$$\begin{aligned}
\dot{d}_{ij}(t) &= \mathbb{E}[z_{ij}(t) - z_{ji}(t)] \\
&= \sum_{X \in E} (\mu_{ij}x_{ij} - \mu_{ji}x_{ji})\pi(x) \\
&= \sum_{X \in E} (\mu_{ij}x_{ij} - \mu_{ji}x_{ji}) \frac{\exp\left(\sum_{mn} x_{mn} \left(\frac{\mu_{mn}}{T} - \alpha d_{mn}(t) + \log(W_0)\right)\right)}{C_T(D(t))}
\end{aligned}$$

Although the dynamics of D are highly non linear, we can prove that there is a unique globally stable equilibrium.

Theorem 5.4.1. *The dynamics for $D(t) = (d_{ij}(t))$ have a unique equilibrium in*

$$d_{ij}^* = \frac{1}{2\alpha} \left[\frac{\mu_{ij} - \mu_{ji}}{T} + \log\left(\frac{\mu_{ij}}{\mu_{ji}}\right) \right]$$

Proof. In the equilibrium, we must have that $\dot{d}_{ij}(t) = 0$. Then

$$0 = \sum_X (\mu_{ij}x_{ij} - \mu_{ji}x_{ji}) \frac{\exp\left(\sum_{mn} x_{mn} \left(\frac{\mu_{mn}}{T} - \alpha d_{mn}^* + \log(W_0)\right)\right)}{C_T(D^*)}$$

we can keep only the terms in the sum that have either $x_{ij} = 1$ or $x_{ji} = 1$

$$\begin{aligned}
0 &= \sum_{X:x_{ij}=1} \mu_{ij} \frac{\exp\left(\sum_{mn} x_{mn} \left(\frac{\mu_{mn}}{T} - \alpha d_{mn}^* + \log(W_0)\right)\right)}{C_T(D^*)} \\
&\quad - \sum_{X:x_{ji}=1} \mu_{ji} \frac{\exp\left(\sum_{mn} x_{mn} \left(\frac{\mu_{mn}}{T} - \alpha d_{mn}^* + \log(W_0)\right)\right)}{C_T(D^*)} \\
&= \mu_{ij} W_0 \exp\left(\frac{\mu_{ij}}{T} - \alpha d_{ij}^*\right) \sum_{X:x_{ij}=1} \frac{\exp\left(\sum_{mn \neq ij} x_{mn} \left(\frac{\mu_{mn}}{T} - \alpha d_{mn}^* + \log(W_0)\right)\right)}{C_T(D^*)} \\
&\quad - \mu_{ji} W_0 \exp\left(\frac{\mu_{ji}}{T} - \alpha d_{ji}^*\right) \sum_{X:x_{ji}=1} \frac{\exp\left(\sum_{mn \neq ji} x_{mn} \left(\frac{\mu_{mn}}{T} - \alpha d_{mn}^* + \log(W_0)\right)\right)}{C_T(D^*)}
\end{aligned}$$

The sums in both terms are equal due to the symmetric interference, because for each configuration X such that $x_{ij} = 1$, there exists a configuration X' such that $x_{ji} = 1$ and the rest of the values are equal. Thus we can take the aforementioned

sum as a common factor and obtain

$$0 = \left[\mu_{ij} \exp \left(\frac{\mu_{ij}}{T} - \alpha d_{ij}^* \right) - \mu_{ji} \exp \left(\frac{\mu_{ji}}{T} - \alpha d_{ji}^* \right) \right] W_0 \sum_{X: x_{ij}=1} \frac{\exp \left(\sum_{mn \neq ij} x_{mn} \left(\frac{\mu_{mn}}{T} - \alpha d_{mn}^* + \log(W_0) \right) \right)}{C_T(D^*)}$$

and as a consequence of the exponentials being always non zero we get that the first factor must be zero

$$0 = \mu_{ij} \exp \left(\frac{\mu_{ij}}{T} - \alpha d_{ij}^* \right) - \mu_{ji} \exp \left(\frac{\mu_{ji}}{T} - \alpha d_{ji}^* \right)$$

and using that $d_{ij} = -d_{ji}$ we can derive the equilibrium

$$d_{ij}^* = \frac{1}{2\alpha} \left[\frac{\mu_{ij} - \mu_{ji}}{T} + \log \left(\frac{\mu_{ij}}{\mu_{ji}} \right) \right]$$

This proves the uniqueness of the equilibrium. The existence is given by the fact that D^* satisfies the equilibrium condition of making all derivatives zero. \square

Before going forward with the stability, we prove the following lemma.

Lemma 5.4.1. *In the dynamics defined by the closed loop in Figure 5.7 we have that*

$$(d_{ij}(t) - d_{ij}^*)E[z_{ij}(t) - z_{ji}(t)] \leq 0 \quad (5.3)$$

Proof. In equilibrium

$$\mathbb{E}[z_{ij}^*] = \mathbb{E}[z_{ji}^*] \quad \forall ij$$

Furthermore for a fixed link ij we have that

$$\mathbb{E}[z_{ij}(t)] = \beta \mu_{ij} R_{ij}(t) = \beta \mu_{ij} W_0 \exp \left(\frac{\mu_{ij}}{T} - \alpha d_{ij}(t) \right)$$

where the constant β depends on the interference from other links and it is always non zero. Due to the symmetric interference generated by the RTS/CTS messages, the proportionality constant for the link ij is the same that for the link ji at all times. Then in equilibrium

$$\mathbb{E}[z_{ij}^*] = \beta \mu_{ij} W_0 \exp \left(\frac{\mu_{ij}}{T} - \alpha d_{ij}^* \right) = \beta \mu_{ji} W_0 \exp \left(\frac{\mu_{ji}}{T} - \alpha d_{ji}^* \right) = \mathbb{E}[z_{ji}^*]$$

If we have at some point the inequality $d_{ij}(t) > d_{ij}^*$, then the access rate of link ij is smaller than in equilibrium and the access rate of link ji is larger than in equilibrium. As a result

$$\mathbb{E}[z_{ij}(t)] < \mathbb{E}[z_{ji}(t)]$$

independently of the state of the rest of the system. The case where $d_{ij}(t) < d_{ij}^*$ is analog. \square

Intuitively, this means that if the imbalance of a link is positive, then it will be less aggressive than its counterpart and as a result we will get a higher throughput from its neighbor. With this we prove the stability of the equilibrium.

Theorem 5.4.2. *The equilibrium of the dynamics for $D(t) = (d_{ij}(t))$ is globally asymptotically stable.*

Proof. Lets introduce the straightforward Lyapunov function

$$V(D(t)) = \frac{1}{2} \sum_{ij} (d_{ij}(t) - d_{ij}^*)^2 > 0 \quad \forall D(t) \neq D^*$$

It is clear that is strictly positive in every point but the equilibrium and that is radially unbounded. Taking the derivative with respect to time we obtain

$$\dot{V}(D(t)) = \sum_{ij} (d_{ij}(t) - d_{ij}^*) E[z_{ij}(t) - z_{ij}(t)] < 0 \quad \forall D(t) \neq D^*$$

where the last inequality is due to Lemma 5.4.1. Then by Lyapunov stability theory [26], the equilibrium is globally asymptotically stable. \square

Now that we know the dynamics of the imbalance measures, we can derive what happens to the throughputs themselves. Note that in equilibrium, the access rate of link ij is

$$R_{ij}^* = W_0 \exp\left(\frac{\mu_{ij}}{T} - \alpha d_{ij}^*\right) = W_0 \sqrt{\frac{\mu_{ji}}{\mu_{ij}}} \exp\left(\frac{\mu_{ij} + \mu_{ji}}{2T}\right)$$

In the exponent, the term $\frac{\mu_{ij} + \mu_{ji}}{2T}$ accounts for the fact that we put emphasis on throughput efficiency, so the pair of links with higher access rates will be more aggressive. On the other hand, the factor $\sqrt{\frac{\mu_{ji}}{\mu_{ij}}}$ is the one that takes care of the reciprocity.

If we make the parameter T go to infinity, then the aggressiveness of link ij is just

$$R_{ij} = W_0 \sqrt{\frac{\mu_{ji}}{\mu_{ij}}}$$

which is enough to yield a symmetric allocation.

Corollary 5.4.1. *For the dynamics defined in Figure 5.7, we have that*

$$E[Z(t)] \rightarrow Z^*$$

where Z^* is a symmetric matrix such that

$$z_{ij}^* = \sum_x \mu_{ij} x_{ij} \frac{\exp\left(\sum_{mn} x_{mn} \left(\frac{\mu_{mn} + \mu_{nm}}{2T} + \frac{1}{2} \log\left(\frac{\mu_{nm}}{\mu_{mn}}\right) + \log(W_0)\right)\right)}{\sum_{x'} \exp\left(\sum_{mn} x'_{mn} \left(\frac{\mu_{mn} + \mu_{nm}}{2T} + \frac{1}{2} \log\left(\frac{\mu_{nm}}{\mu_{mn}}\right) + \log(W_0)\right)\right)}$$

This means that the proposed algorithm converges to a symmetric allocation independently of the chosen parameters α , T and W_0 .

Remark 5.4.2. In the closed loop given by figure 5.7, it is clear that we need the input of the integrator to be zero, but that does not mean that the output will be zero. From the network perspective, it means that the actual number of bytes sent in a direction need not be equal to the number of bytes sent in the other direction. On the contrary, the imbalance between the total number of bytes sent is necessary to reach an equilibrium.

If we start with $D(0) = D^*$, we no longer have dynamics in D and as a result we get again a reversible continuous time Markov chain with invariant distribution

$$\pi^*(X) = \frac{\exp\left(\sum_{ij} x_{ij} \left(\frac{\mu_{ij} + \mu_{ji}}{2T} + \frac{1}{2} \log\left(\frac{\mu_{ji}}{\mu_{ij}}\right) + \log(W_0)\right)\right)}{C_T} \quad (5.4)$$

Remark 5.4.3. Note that we could try to approximate the distribution given by (5.4) by the Gibbs distribution with respect to the energy

$$\mathcal{E}(X) = \sum_{ij} x_{ij} \frac{\mu_{ij} + \mu_{ji}}{2}$$

when T goes to zero. In the limit, the distribution would be concentrated on the configurations

$$X^* = \arg \max_X \sum_{ij} x_{ij} \frac{\mu_{ij} + \mu_{ji}}{2}$$

which are actually symmetric. As a consequence, if the rate matrix μ is not symmetric, we would get a non symmetric allocation Z . The problem with this reasoning is that when we bring down the parameter T , the convergence of the Markov chain is slower. As a result, our hypothesis of separation of time scales can not be made and thus this result is not valid.

Example 5.4.1. Suppose that the connectivity graph is full mesh. In that case, the only possible configurations are the ones that activate one link at a time. Let X_{ij} be the configuration in where only link ij is active. Then, when the dynamics of the imbalances D are in equilibrium, we get the invariant distribution

$$\pi(X_{ij}) = \frac{W_0 \exp\left(\frac{\mu_{ij} + \mu_{ji}}{2T} + \frac{1}{2} \log\left(\frac{\mu_{ji}}{\mu_{ij}}\right)\right)}{\sum_{mn} W_0 \exp\left(\frac{\mu_{mn} + \mu_{nm}}{2T} + \frac{1}{2} \log\left(\frac{\mu_{nm}}{\mu_{mn}}\right)\right)}$$

from this we get

$$\begin{aligned} E[z_{ij}] &= \mu_{ij} \frac{\exp\left(\frac{\mu_{ij} + \mu_{ji}}{2T} + \frac{1}{2} \log\left(\frac{\mu_{ji}}{\mu_{ij}}\right)\right)}{\sum_{mn} \exp\left(\frac{\mu_{mn} + \mu_{nm}}{2T} + \frac{1}{2} \log\left(\frac{\mu_{nm}}{\mu_{mn}}\right)\right)} \\ &= \mu_{ij} \frac{\sqrt{\frac{\mu_{ji}}{\mu_{ij}}} \exp\left(\frac{\mu_{ij} + \mu_{ji}}{2T}\right)}{\sum_{mn} \sqrt{\frac{\mu_{nm}}{\mu_{mn}}} \exp\left(\frac{\mu_{mn} + \mu_{nm}}{2T}\right)} \end{aligned}$$

If we make T go to infinity, we obtain

$$E[z_{ij}] = \mu_{ij} \frac{\sqrt{\frac{\mu_{ji}}{\mu_{ij}}}}{\sum_{mn} \sqrt{\frac{\mu_{nm}}{\mu_{mn}}}}$$

and the relation between the throughput of two links would be

$$\frac{E[z_{ij}]}{E[z_{kl}]} = \frac{\sqrt{\mu_{ij}\mu_{ji}}}{\sqrt{\mu_{kl}\mu_{lk}}}$$

With this we can conclude that in a network with full connectivity, if we make T go to infinity, the average throughput of a bidirectional link in a full mesh will be proportional to the geometric mean of the unidirectional link capacities. In this case we get good connection diversity while sacrificing the efficiency.

Wireless neighbor selection algorithm

Now we give a step by step implementation of the designed neighbor selection algorithm for ad-hoc wireless networks.

Suppose that we have a P2P overlay over an ad-hoc wireless network with connectivity graph G . Then repeat the following algorithm:

- Step 1: For each possible outgoing link ij , draw an exponentially distributed random time with parameter

$$R_{ij} = W_0 \exp\left(\frac{\mu_{ij}}{T} - \alpha d_{ij}\right)$$

and wait that time.

- Step 2: If another link transmits before one of the local links ij , update the imbalances d_{ij} to reflect the amount of data received, draw the exponentially distributed random times with the updated imbalances and wait again.
 - Step 3: When the time of one of the links expires, transmit during an exponentially distributed random time of mean one.
 - Step 4: Update the imbalances d_{ij} with the neighbors to reflect the amount of data transmitted.
-

5.5 SIMULATIONS

5.5.1 Implementation and performance metrics

We now evaluate the devised wireless neighbor selection algorithm as a means to achieve symmetric allocations and different levels of efficiency and connection diversity. We implemented the algorithm in Matlab and in order to evaluate it, we simulate two scenarios. Every simulation starts from the initial condition where no one is transmitting.

As a measure of the reciprocity and efficiency, we evaluate three metrics:

1. The Kullback-Leibler divergence between the average upload rate and the average download rate defined by

$$D_{KL}(\mu(Z)||r(Z)) = \sum_i r_i(Z) \log \left(\frac{r_i(Z)}{\mu_i(Z)} \right)$$

where $\mu_i(Z) = \sum_n z_{in}$ is the average upload rate of peer i and $r_i(Z) = \sum_n z_{ni}$ is the average download rate of peer i .

2. The Jain index of the quotients of the average upload and download rates defined by

$$J(Z) = \frac{\left(\sum_i \frac{\mu_i(Z)}{r_i(Z)} \right)^2}{N \sum_i \left(\frac{\mu_i(Z)}{r_i(Z)} \right)^2}$$

3. The ratio between the average throughput of the network and the maximum possible throughput achievable defined by

$$Efficiency = \frac{\sum_{ij} z_{ij}}{\max_{ij} \sum_{ij} z_{ij}}$$

5.5.2 Full mesh topology

The first scenario has a full mesh topology with 50 nodes and with a random matrix of rates μ , where all the rates were drawn as a uniform random variable over $[0, 1]$. This would correspond for example to a crowded classroom with 50 students in it. In Figure 5.8 we can see that the Jain index is extremely close to one, achieving an almost perfect reciprocity. The Kullback-Leibler divergence is also very small as seen in figure 5.9. Furthermore, it can be seen in both figures that the temperature has a small effect in the convergence speed, as lower temperatures yield somewhat slower convergence rates.

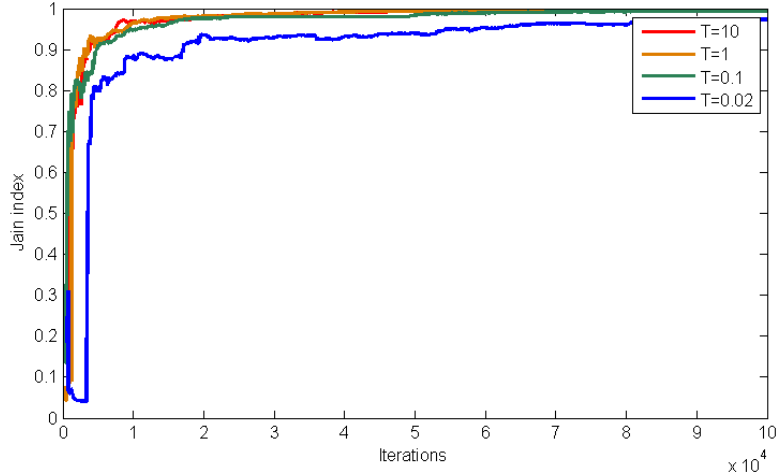


Figure 5.8. Jain index of the upload and download throughput of the peers in a full mesh network with 50 nodes, $W_0 = 1$ and $\alpha = 1$ for different temperatures.

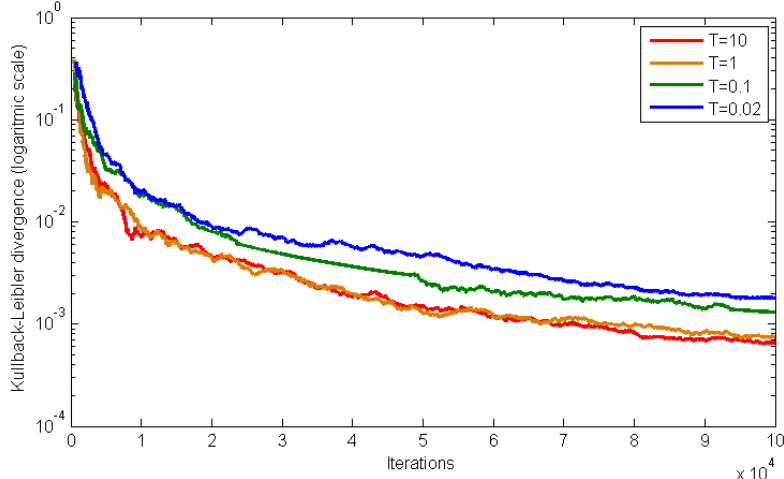


Figure 5.9. Kullback-Leibler divergence between the upload and download throughput of the peers in a full mesh network with 50 nodes, $W_0 = 1$ and $\alpha = 1$ for different temperatures.

As a measure of connection diversity, we will use the minimum sum of the throughputs of links that disconnect the connectivity graph, defined by

$$\begin{aligned} \text{Min cut} = & \min_{L \text{ subset of links}} \sum_{ij \in L} z_{ij} \\ \text{s.t.} & \quad G_I \setminus L \text{ is disconnected} \end{aligned}$$

In table 5.1 we can see the tradeoff between the efficiency and the connection diversity.

Temperature	Efficiency	Min-cut
0.01	96%	0
0.02	93%	2.5×10^{-5}
0.05	88%	2.3×10^{-4}
0.1	66%	4.8×10^{-3}
1	41%	5.1×10^{-3}
10	37%	5.2×10^{-3}

Table 5.1. Efficiency and connection diversity in the case of a full mesh network.

5.5.3 Poisson network topology

In the second scenario we have the peers spatially distributed as a Poisson process over the bounded set $[0, 1]^2$, where in average we have 50 peers. Then two nodes are connected if they are at a distance less than the radius $R = 0.3$. Furthermore, we give each node a uniform random transmission power P_i in $[0, 0.1]$, and define the rate of each link as

$$\mu_{ij} = \min \left\{ 1, \frac{P_i}{d(i, j)^2} \right\}$$

In Figure 5.11 we can see that the Jain index very close to one, achieving a very good reciprocity. The Kullback-Leibler divergence is also small as seen in figure 5.10. Furthermore, it can be seen in both figures that the temperature has some effect on the convergence speed. However, the different temperatures produce a smaller difference in the efficiency in this scenario as seen in table 5.2.

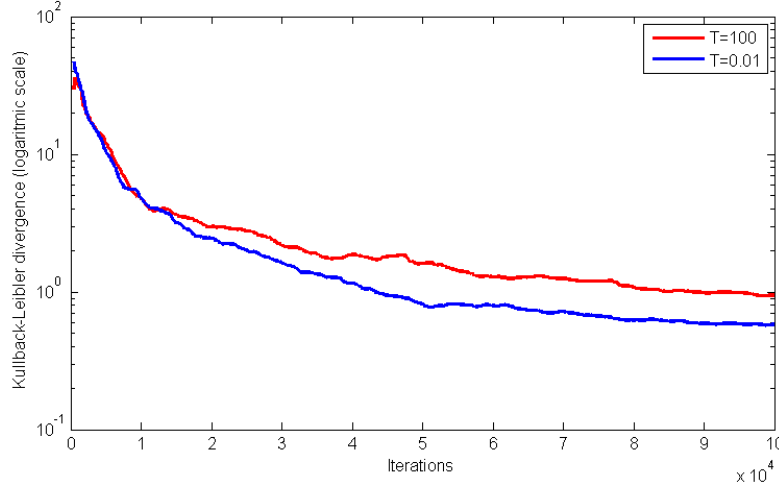


Figure 5.10. Kullback-Leibler divergence between the upload and download throughput of the peers in a Poisson network topology with 50 nodes, $W_0 = 10$ and $\alpha = 10$ for different temperatures.

Temperature	Efficiency
0.01	88%
100	73%

Table 5.2. Efficiency for the Poisson network.

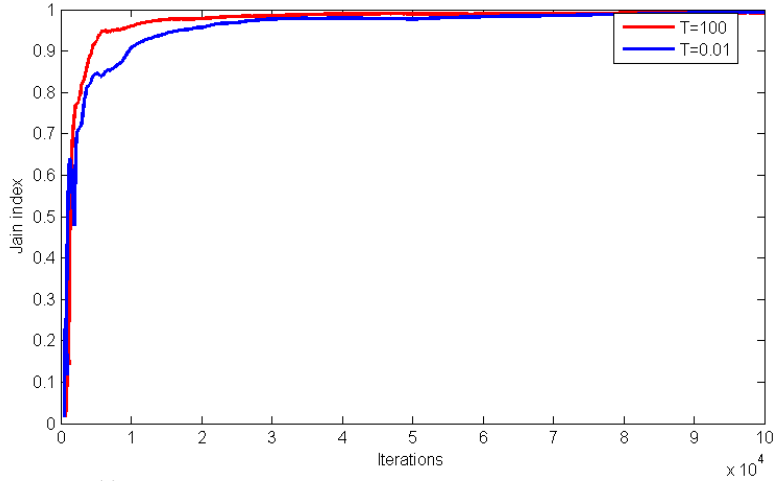


Figure 5.11. Jain index of the upload and download throughput of the peers in a Poisson network topology with 50 nodes, $W_0 = 10$ and $\alpha = 10$ for different temperatures.

A visual representation of the network, with the throughput of each link is given in Figure 5.12 for $T = 100$, in Figure 5.13 for $T = 0.05$ and in Figure 5.14 for $T = 0.001$. Note that when the parameter is higher, the throughputs of the links are more homogeneous, whether for $T = 0.05$ there are links that are significantly more active (darker gray) and others that are significantly less active (almost white). Furthermore, when $T = 0.001$ there is a clear disconnection of the network as it becomes too static. This shows that there is still a tradeoff in this case between the efficiency that increases as T goes to zero, and the connection diversity, that gets worse as T goes to zero.

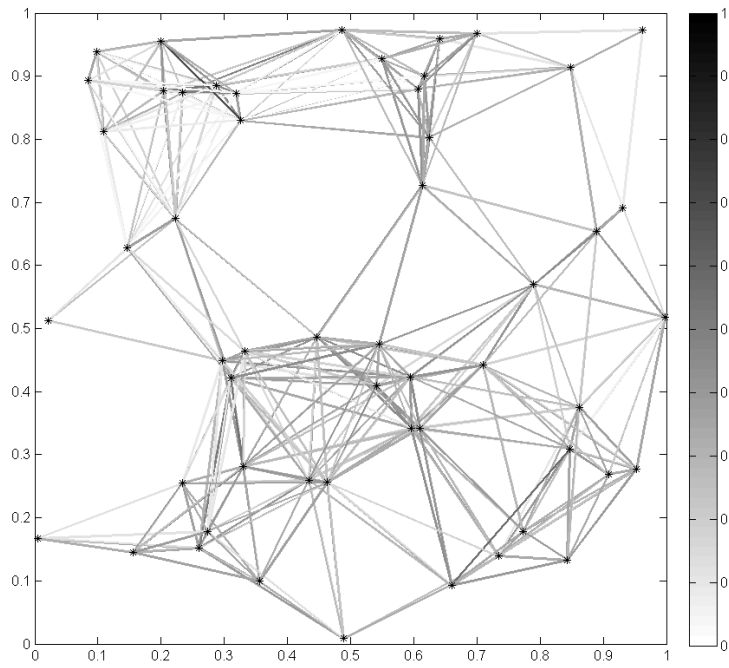


Figure 5.12. Connectivity graph and throughputs when $T = 100$.

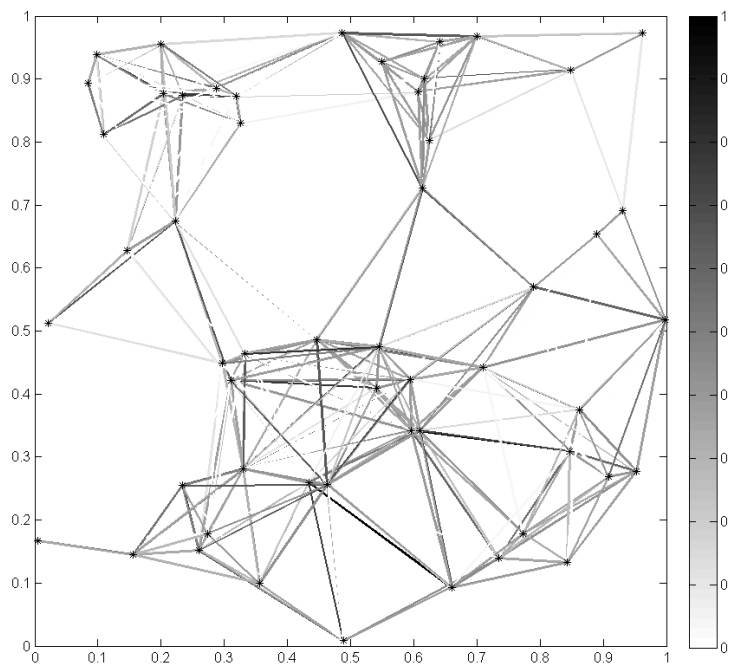


Figure 5.13. Connectivity graph and throughputs when $T = 0.05$.

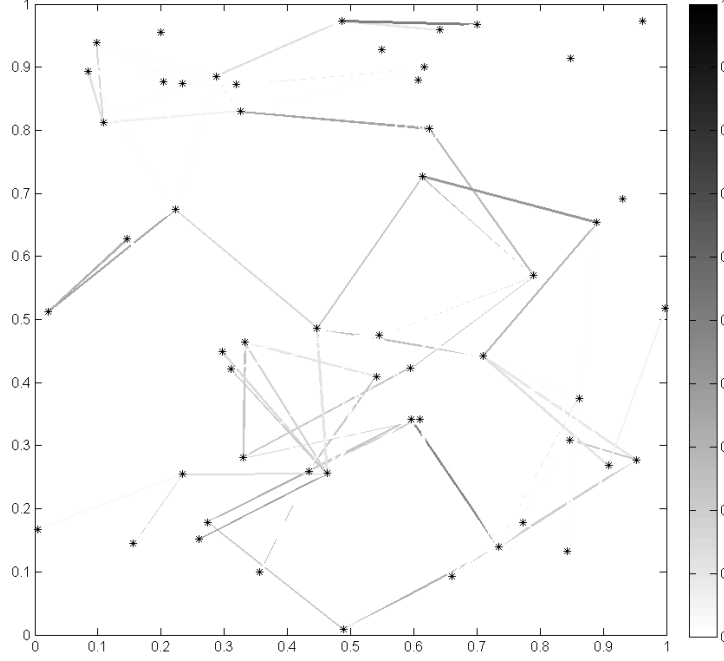


Figure 5.14. Connectivity graph and throughputs when $T = 0.001$.

5.6 CONCLUSIONS

In this chapter we characterized the rate allocations in an ad-hoc wireless network and showed that it is easy to find a symmetric allocation and as a result poses the correct incentives to the peers. Furthermore, we can explore the design space of the tradeoff between connection diversity and throughput efficiency, without sacrificing the peerwise reciprocity of symmetric allocations at any time. This results in a decentralized algorithm with a policy similar to the “backpressure” for queues, that has the proper incentives for any network topology. This was validated through simulations.

Chapter 6

Conclusions and future work

In this thesis we analyzed the problem of the resource allocation in a peer-to-peer system making emphasis on the incentives and fairness. Using a fluid model for the population dynamics, we showed that the one that minimizes the average download time does not provide the proper incentives and would not work if implemented in practice. Introducing conditions for the correct incentives on the download rates, we arrived at the conclusion that the max-min allocation is the most efficient while presenting minimal incentives to the peers. However, we argued that a proportional allocation provides stronger incentives and it is more appropriate for these networks.

Having a clear resource allocation in mind, we set up to design a decentralized neighbor selection algorithm that achieves such allocation. In order to do this, first we reviewed a matrix renormalization algorithm devised by Sinkhorn in the 1960's that achieves the desired allocations, but note that its practical implementation would fail under a dynamic network and would need a fine control of the throughputs that is hard to achieve. We proposed a solution based on a Gibbs sampler that achieves an approximately proportional allocation and that is simple to implement. Furthermore, we showed that there is a tradeoff between fairness and rate of convergence and also showed through simulations that this algorithm performs better than several alternatives.

Lastly, we turned our attention to P2P overlays over an ad-hoc wireless network. In this challenging setting we observed that there are other metrics to take into account: the efficiency and the connection diversity. We proposed a specialized neighbor selection algorithm that imposes a proportional allocation as before and can achieve different levels of efficiency and connection diversity.

There are several lines of work that can be pursued in the future. In the wired case, a next logical step would be to implement and test the algorithms in a packet level simulator or a real test bed, in order to validate the results obtained in the Matlab simulations. On the other hand, another open line of work would be to

analyze the performance of the algorithm in a dynamic network, when there are arrivals and departure of peers.

In the wireless case, we observed that ultimately the improvement in efficiency comes at the expense of reduced connection diversity, which seems to be a consequence of the wireless nature of the network. Our algorithm for the wireless case explores this tradeoff using a single parameter, but it is still an open question which metric should be used to correctly measure connection diversity and in which point should the network operate with respect to this tradeoff.

Bibliography

- [1] B. Cohen, “Incentives build robustness in BitTorrent,” in *Proc. of 1st Workshop on the Economics of Peer-2-Peer Systems*, 2003, pp. 1–5.
- [2] B. Hajek and J. Zhu, “The Missing Piece Syndrome in Peer-to-Peer Communication,” in *Proc. of the IEEE International Symposium on Information Theory*, 2010, pp. 1748–1752.
- [3] D. Levin, K. LaCurts, N. Spring, and B. Bhattacharjee, “BitTorrent is an Auction: Analyzing and Improving BitTorrent’s Incentives,” in *Proc. of the ACM SIGCOMM Conference*, 2008, pp. 243–254.
- [4] C. Aperjis, R. Johari, and M. Freedman, “Bilateral and Multilateral Exchanges for Peer-Assisted Content Distribution,” *IEEE/ACM Transactions on Networking*, vol. 19, pp. 1290–1303, Oct. 2011.
- [5] R. Ma, S. Lee, J. Lui, and D. Yau, “Incentive and Service Differentiation in P2P Networks: A Game Theoretic Approach,” *IEEE/ACM Transactions on Networking*, vol. 14, pp. 978–991, Oct. 2006.
- [6] B. Fan, J. Lui, and D. Chiu, “The Design Trade-offs of BitTorrent-like File Sharing Protocols,” *IEEE/ACM Transactions on Networking*, vol. 17, pp. 365–376, Apr. 2009.
- [7] J. Little, “A proof of the queueing formula $L = \lambda W$,” *Operations research*, vol. 9, pp. 383–387, Jun. 1961.
- [8] F. Paganini, A. Ferragut, and M. Zubeldía, “Dynamics of heterogeneous peer-to-peer networks,” in *Proc. of the 52nd IEEE Conference on Decision and Control*, 2013, pp. 3293–3298.
- [9] X. Yang and G. de Veciana, “Performance of peer-to-peer networks: Service capacity and role of resource sharing policies,” *Performance evaluation*, vol. 63, pp. 175–194, Mar. 2006.
- [10] F. Wu and L. Zhang, “Proportional response dynamics leads to market equilibrium,” in *Proc. of the 39th ACM Symposium on Theory of Computing*, 2007, pp. 354–363.

- [11] R. Sinkhorn, “A relationship between arbitrary positive matrices and doubly stochastic matrices,” *Annals of Mathematical statistics*, vol. 35, pp. 876–876, Jun. 1964.
- [12] R. Sinkhorn and P. Knopp, “Concerning nonnegative matrices and doubly stochastic matrices,” *Pacific Journal of Mathematics*, vol. 21, pp. 343–348, Feb. 1967.
- [13] R. Sinkhorn, “Diagonal equivalence to matrices with prescribed row and column sums II,” *Proceedings of the American Mathematical Society*, vol. 45, pp. 195–198, Aug. 1974.
- [14] H. Balakrishnan, I. Hwang, and C. Tomlin, “Polynomial Approximation Algorithm for Belief Matrix Maintenance in Identity Management,” in *Proc. of the 43rd IEEE Conference on Decision and Control*, 2004, pp. 4874–4879.
- [15] M. Zubeldía, A. Ferragut, and F. Paganini, “Proportional fairness in heterogeneous peer-to-peer networks through reciprocity and Gibbs sampling,” in *Proc. of the 51st Allerton conference on Communication, Control and Computing*, 2013, pp. 123–130.
- [16] Z. Shao, H. Zhang, M. Chen, and K. Ramachandran, “Reverse-Engineering BitTorrent: A Markov Approximation Perspective,” in *Proc. of the 31st IEEE International Conference on Computer Communications*, 2012, pp. 2996–3000.
- [17] H. Zhang, Z. Shao, M. Chen, and K. Ramachandran, “Optimal Neighbor selection in BitTorrent-like Peer-to-Peer Networks,” in *Proc. of the ACM SIGMETRICS/RICS*, 2011, pp. 141–142.
- [18] M. Chen, S. Liew, Z. Shao, and C. Kai, “Markov Approximation for Combinatorial Network Optimization,” in *Proc. of the 29th IEEE International Conference on Computer Communications*, 2010, pp. 1–9.
- [19] P. Brémaud, *Markov Chains: Gibbs Fields, Monte Carlo simulation and queues*. New York, NY: Springer, 1999.
- [20] L. Jiang and J. Walrand, “A Distributed CSMA Algorithm for Throughput and Utility Maximization in Wireless Networks,” *IEEE/ACM Transactions on Networking*, vol. 18, pp. 960–972, Jun. 2010.
- [21] A. Waluyo, D. Taniar, W. Rahayu, A. Aikebaier, M. Takizawa, and B. Srinivasan, “Mobile Peer-to-Peer data dissemination in wireless ad-hoc networks,” *Information Sciences*, vol. 20, pp. 3–20, May 2013.
- [22] X. Wu, S. Tavildar, S. Shakkottai, T. Richardson, J. Li, R. Laroia, and A. Jovicic, “FlashLinQ: A Synchronous Distributed Scheduler for Peer-to-Peer Ad Hoc Networks,” *IEEE/ACM Transactions on Networking*, vol. 21, pp. 1215–1228, Aug. 2013.

- [23] M. Neely, “Optimal Peer-to-Peer Scheduling for Mobile Wireless Networks with Redundantly Distributed Data,” *IEEE Transactions on mobile computing*, to be published.
- [24] M. Zubeldía, A. Ferragut, and F. Paganini, “Overcoming performance pitfalls in rate-diverse high speed WLANs,” *Computer Networks*, vol. 57, pp. 3673–3685, Dec. 2013.
- [25] L. Tassiulas and A. Ephremides, “Stability properties of constrained queueing systems and scheduling policies for maximum throughput in multihop radio networks,” *IEEE Transactions on Automatic Control*, vol. 37, pp. 1936–1948, Dec. 1992.
- [26] A. Lyapunov, “The general problem of stability of motion,” Ph.D. dissertation, University of Kharkiv, Ukraine, 1892.
- [27] G. Carmona and K. Podczeck. (2010) Approximation and Characterization of Nash Equilibria of Large Games. [Online]. Available: <http://homepage.univie.ac.at/konrad.podczeck/material/ch100115.pdf>
- [28] J. Norris, *Markov Chains*. Cambridge, UK: Cambridge University Press, 1998.
- [29] C. Godsil and G. Royle, *Algebraic graph theory*. New York, NY: Springer, 2001.
- [30] P. Erdős and T. Gallai, “Graphs with prescribed degrees of vertices (Hungarian),” *Matematikai Lapok*, vol. 11, pp. 264–274, 1960.

Appendix A

Game theoretic approach to incentives

In chapter 2 we presented three possible resource allocations in a wired P2P network: the allocation for minimum download time, the max-min allocation and the proportional allocation. In order to analyze the incentives provided by different resource allocations, we are going to model the problem as a non-cooperative non-atomic game. We use these type of games as they are a tractable approximation of games with many users [27].

First of all, we introduce the three main components of any game: the players, the strategies and the payoffs.

Players

The players are only the leechers (not the seeders), and they are non-atomic: they can be represented as the points in a set with mass Λ . Besides, there are n classes of leechers with different upload capacities $\mu_1 > \mu_2 > \dots > \mu_n$ which can be represented as a vector $\lambda = (\lambda_1, \lambda_2, \dots, \lambda_n)$ of masses such that $\sum_{i=1}^n \lambda_i = \Lambda$, i.e. the sum of all is equal to the total mass Λ . This means that there is a fraction $\frac{\lambda_i}{\Lambda}$ of the peers that have upload capacity μ_i .

Strategies

If a leecher has upload capacity μ_k , then it can choose an upload rate $u \in \{\mu_i : \mu_i \leq \mu_k\}$, i.e., any upload rate slower than its upload capacity. The strategies of all players can be represented as the vector $\lambda' = (\lambda'_1, \lambda'_2, \dots, \lambda'_n)$ such that $\sum_{i=1}^n \lambda'_i = \Lambda$.

This means that a fraction $\frac{\lambda'_i}{\Lambda}$ of peers chose the upload rate μ_i . Due to the restriction in the upload rate that a peer can choose (slower or equal to its upload capacity), then the possible strategies are the ones that satisfy the inequalities

$$\lambda'_i \leq \sum_{j=1}^i \lambda_j \quad \forall i$$

These restrictions say that the total amount of peers that select the upload capacity μ_i are less than or equal to the amount of peers that have upload capacity higher than or equal to μ_i .

Payoffs

Given the joint strategies of all players represented by λ' , the players that chose the upload rate μ_i will have r_i^* as their payoff, which is obtained as the one in the equilibrium of the dynamics defined by (2.1) and (2.2) when the arrival rates are λ' , but with the seeders fixed (as they do not participate in the game). Note that the payoffs depend on the choice of all peers.

Of course the game only makes sense in the case in which the whole network is not saturated by download capacity, so we will focus only on that case. We present a definition that will be used throughout this appendix.

Definition A.0.1 (Nash equilibrium). A strategy λ' is a Nash equilibrium for a non-atomic game if for any other strategy $\lambda'' = (\lambda'_1, \dots, \lambda'_i - \epsilon, \dots, \lambda'_j + \epsilon, \dots, \lambda'_n)$, the peers with upload capacity μ_j achieve a smaller payoff than the peers of upload capacity μ_i under λ' .

The desirable situation is the one in which each peer selects its upload capacity as its upload rate (i.e. $\lambda' = \lambda$). We would like it to be a Nash equilibrium for the game whose payoffs are defined by the resource allocation.

A.1. MINIMUM DOWNLOAD TIME GAME

First we turn our attention to the resulting resource allocation of minimizing the average download time. The part of the game that needs explaining is the payoff, as the players and strategies are the same as before. In a given strategy, suppose that the maximum upload capacity selected by a set of peers of positive measure is μ_k . Then we will have that $\lambda'_i = 0 \quad \forall i < k$. If we take λ'_i as the arrival rates of our system, then by proposition 2.2.2, the payoff will be the download rates d_i for all peers who selected an upload capacity $\mu_i < \mu_k$ and $\frac{\lambda'_k \mu_k}{\Lambda - C_y - \sum_{i=k+1}^n \frac{\mu_i \lambda'_i}{d_i}}$ for all peers who

selected the upload rate μ_k .

Now, if we look at the desired strategy of selecting the highest possible upload capacity in the game, we have the following result.

Proposition A.1.1. *If*

$$d_2 > \frac{\lambda_1 \mu_1}{\Lambda - C_y - \sum_{i=2}^n \frac{\mu_i \lambda_i}{d_i}} > 0$$

then setting the upload rate equal to the upload capacity (i.e. $\lambda' = \lambda$) is not a Nash equilibrium for the minimum download time game.

Proof. For the strategy $\lambda' = \lambda$, the payoff of the peers in the first group is

$$\frac{\lambda_1 \mu_1}{\Lambda - C_y - \sum_{i=2}^n \frac{\mu_i \lambda_i}{d_i}}$$

Now consider the strategy

$$\lambda' = (\lambda_1 - \epsilon, \lambda_2 + \epsilon, \lambda_3, \dots, \lambda_n)$$

where a set of peers decided to select an upload rate of μ_2 instead of μ_1 . Then the payoff of the peers that switched strategies is d_2 , which by hypothesis is larger than before. As a result $\lambda' = \lambda$ is not a Nash equilibrium for this game. \square

This proposition shows us, in a game theoretic framework, that the resource allocation does not provide the proper incentives and it will not work if it is implemented in a network in which the users have a choice on their upload rate.

A.2. MAX-MIN ALLOCATION GAME

In Chapter 2, we intuitively reached the conclusion that the max-min allocation is the one that minimizes the average download time while maintaining the proper incentives. In the max-min allocation game, every peer chooses its upload rate as before and then the payoffs are the max-min allocation rates

$$\begin{aligned} r_i^* &= \frac{\sum_{j=1}^{i^*} \lambda_j \mu_j}{\Lambda - C_y - \sum_{j=i^*+1}^n \frac{\lambda_j \mu_j}{d_j}} & \text{if } i = 1, \dots, i^* \\ &= d_i & \text{if } i = i^* + 1, \dots, n \end{aligned}$$

For this game we have the following result.

Proposition A.2.1. *Setting the upload rate equal to the upload capacity (i.e. $\lambda' = \lambda$) is a Nash equilibrium for the max-min allocation game.*

Proof. Suppose that a peer with upload capacity μ_i with $i > i^*$ chose an upload rate $\mu_j < \mu_i$. Then its payoff will decrease from d_i to d_j in the best case (because $d_i > d_j \forall i < j$), or to a lower level if the upload capacity of the network deteriorates so much that the new cutoff index i'^* is smaller than j .

If a peer with upload capacity μ_i with $i \leq i^*$ chose an upload rate $\mu_j < \mu_i$ with $j > i^*$, then its payoff will decrease from r_i^* to d_j in the best case, or to a lower level if the upload capacity of the network deteriorates so much that the new cutoff index i'^* is smaller than j . On the other hand, if it chose an upload rate $\mu_j < \mu_i$ with $j \leq i^*$, then its payoff will go from

$$\frac{\sum_{j=1}^{i^*} \lambda_j \mu_j}{\Lambda - C_y - \sum_{j=i^*+1}^n \frac{\lambda_j \mu_j}{d_j}}$$

to

$$\frac{\sum_{j=1}^{i'^*} \lambda'_j \mu_j}{\Lambda - C_y - \sum_{j=i'^*+1}^n \frac{\lambda'_j \mu_j}{d_j}}$$

Suppose that $i'^* = i^*$. In that case note that $\sum_{j=1}^{i^*} \lambda_j \mu_j > \sum_{j=1}^{i^*} \lambda'_j \mu_j$ and that

$$\frac{d}{dx} \left(\frac{x}{\Lambda - C_y - x} \right) = \frac{\Lambda - C_y}{(\Lambda - C_y - x)^2} > 0$$

As a result, its payoff will decrease by changing the upload rate. If $i'^* > i^*$, then the payoff will be less than d_{i^*} which was also smaller.

Taking everything into account, we have that the payoff of any peer is always larger under λ than under any other strategy λ' . Hence λ is a Nash equilibrium for this game. \square

A.3. PROPORTIONAL FAIRNESS GAME

In the proportional fairness game, if a peer chooses the upload rate μ_i , the its payoff will be

$$r_i^* = \mu_i \frac{\Lambda}{\Lambda - C_y}$$

For this game, we have the following result:

Proposition A.3.1. *Setting the upload rate equal to the upload capacity (i.e. $\lambda' = \lambda$) is Nash equilibrium for the proportional fairness game.*

Proof. Note that in this case, a peer that chose an upload rate μ_i will have a payoff $r_i^* = \mu_i \frac{\Lambda}{\Lambda - C_y}$ which is an increasing function of the upload rate μ_i and it is independent of the decision of the other players. If we have a strategy

$$\lambda' = (\lambda_1, \dots, \lambda_i - \epsilon, \dots, \lambda_j + \epsilon, \dots, \lambda_n)$$

then the peers that chose a lower upload rate than their upload capacity will obtain a smaller payoff. Consequently, deviating from the strategy λ only decreases the payoffs of the peers, hence it is a Nash equilibrium. \square

Remark A.3.1. Note that for this choice of resource allocation, the payoff of a peer is independent of the upload rates that chose the other players, which is a desirable property. This is other reason why we should use the proportional allocation.

Appendix B

Population dynamics with arbitrary upload capacities

In Chapter 2 we introduced a fluid model for the dynamics of a population under a general resource allocation r , with the restriction that there is only a finite set of upload capacities. As the actual number of different upload capacities is also very large, we can go one step further and consider a fluid model where that parameter is also a fluid. In general, we will admit an arrival density $\lambda(\mu)$ over the continuum of the upload capacities.

B.1. FLUID MODEL

We present a fluid model of the population dynamics for a wired P2P network. First, we introduce some notation.

- Let $E = [\mu_{min}, \mu_{max}]$ be the set of possible upload capacities.
- The functions $x, y : E \times [0, +\infty) \rightarrow [0, +\infty)$ are such that $x(\mu, t)$ and $y(\mu, t)$ are the densities of the number of leechers and seeders respectively in the network at time t .
- $\lambda : E \rightarrow [0, +\infty)$ is the density of the intensity of arrivals.
- $\gamma : E \rightarrow (0, +\infty)$ is the exit rate of the seeders.
- $r(\mu, x(\mu, t), y(\mu, t))$ is download rate of a peer with upload capacity μ for densities $x(\mu, t)$ of leechers and $y(\mu, t)$ of seeders.

We will assume that the size of the file is equal to 1 and that the leechers that finish downloading stay as seeders, as in Chapter 2. With this, we define the

dynamics of the seeders and leechers

$$\begin{aligned}\frac{\partial x(\mu, t)}{\partial t} &= \lambda(\mu) - r(\mu, x(\mu, t), y(\mu, t)) x(\mu, t) \\ \frac{\partial y(\mu, t)}{\partial t} &= r(\mu, x(\mu, t), y(\mu, t)) x(\mu, t) - \gamma(\mu) y(\mu, t)\end{aligned}$$

with the “conservation of mass” restriction

$$\int_E r(\mu, x(\mu, t), y(\mu, t)) x(\mu, t) d\mu \leq \int_E \mu [x(\mu, t) + y(\mu, t)] d\mu$$

which is no more than the density based version of equation (2.3).

In order to find the equilibrium function $x^*(\mu)$, we have to solve the fixed point equation

$$\lambda(\mu) = r(\mu, x^*(\mu), y^*(\mu)) x^*(\mu)$$

for which there is no analytic solution in general. However, in some cases we can find an explicit formula for the equilibrium functions.

Proposition B.1.1. *If the rate function is of the form*

$$r(\mu, x(\mu, t), y(\mu, t)) = \rho(\mu) \int_E \mu [x(\mu, t) + y(\mu, t)] d\mu$$

then the previous dynamics have a unique equilibrium in

$$\begin{aligned}x^*(\mu) &= \frac{\lambda(\mu)}{C^* \rho(\mu)} \\ y^*(\mu) &= \frac{\lambda(\mu)}{\gamma(\mu)}\end{aligned}$$

where C^ is the total capacity of the network in equilibrium*

$$C^* = \int_E \frac{\mu \lambda(\mu)}{2\gamma(\mu)} d\mu + \sqrt{\left[\int_E \frac{\mu \lambda(\mu)}{2\gamma(\mu)} d\mu \right]^2 + \int_E \frac{\mu \lambda(\mu)}{\rho(\mu)} d\mu}$$

Proof. In the equilibrium, the derivatives with respect to time must be all zero

$$\begin{aligned}0 &= \lambda(\mu) - r(\mu, x^*(\mu), y^*(\mu)) x^*(\mu) \\ 0 &= r(\mu, x^*(\mu), y^*(\mu)) x^*(\mu) - \gamma(\mu) y^*(\mu)\end{aligned}$$

Then for the seeders we have that

$$\lambda(\mu) = \gamma(\mu) y^*(\mu)$$

As γ is non zero, we have that

$$y^*(\mu) = \frac{\lambda(\mu)}{\gamma(\mu)}$$

Now for the leechers

$$\lambda(\mu) = r(\mu, x^*(\mu), y^*(\mu)) x^*(\mu)$$

Given that $r(\mu, x^*(\mu), y^*(\mu)) = C^* \rho(\mu)$, we only have to find the total capacity of the network in the equilibrium

$$C^* = \int_E \mu [x^*(\mu) + y^*(\mu)] d\mu$$

substituting the equilibrium functions for the leechers and seeders we obtain

$$C^* = \int_E \frac{\mu \lambda(\mu)}{C^* \rho(\mu)} d\mu + \int_E \frac{\mu \lambda(\mu)}{\gamma(\mu)} d\mu$$

From this we get a second order equation in C^*

$$C^{*2} - C^* \int_E \frac{\mu \lambda(\mu)}{\gamma(\mu)} d\mu - \int_E \frac{\mu \lambda(\mu)}{\rho(\mu)} d\mu = 0$$

which yields

$$C^* = \frac{C_y + \sqrt{C_y^2 + 4\beta}}{2}$$

where C_y is the total upload capacity of the seeders

$$C_y = \int_E \frac{\mu \lambda(\mu)}{\gamma(\mu)} d\mu$$

and

$$\beta = \int_E \frac{\mu \lambda(\mu)}{\rho(\mu)} d\mu$$

□

B.2. ANALYSIS OF RESOURCE ALLOCATIONS

We will try to minimize the total number of leechers in the network in equilibrium, that is minimize

$$X^* = \int_E x^*(\mu) d\mu = \int_E \frac{\lambda(\mu)}{r(\mu, x^*(\mu), y^*(\mu))} d\mu$$

The first constraint that we have is the one given by the “conservation of mass”, which in equilibrium is equivalent to

$$\int_E \mu x^*(\mu) d\mu + \int_E \frac{\lambda(\mu) \mu}{\gamma(\mu)} d\mu \geq \Lambda$$

Let

$$C_y = \int_E \frac{\lambda(\mu)\mu}{\gamma(\mu)} d\mu$$

be the total upload capacity of the seeders. Then, the previous restriction can be rewritten as

$$\int_E \mu x^*(\mu) d\mu \geq \Lambda - C_y$$

The last restriction that we will impose is that a peer with upload capacity μ has a maximum download capacity $d(\mu)$, where the function d is strictly increasing. That is

$$r(\mu, x(\mu, t), y(\mu, t)) \leq d(\mu) \quad \forall \mu$$

We can translate this into a restriction in the function $x^*(\mu)$

$$x^*(\mu) \geq \frac{\lambda(\mu)}{d(\mu)}$$

Now we can write the optimization problem as

$$\begin{aligned} \inf \quad & X^* = \int_E x^*(\mu) d\mu \\ \text{s.t.} \quad & \int_E \mu x^*(\mu) d\mu \geq \Lambda - C_y \\ & x^*(\mu) \geq \frac{\lambda(\mu)}{d(\mu)} \end{aligned}$$

Case sustained by seeders

When $C_y > \Lambda$, we have again that the restriction of the conservation of mass is not active and thus the minimum is achieved when we have equality in the download capacity restrictions. That is

$$x^*(\mu) = \frac{\lambda(\mu)}{d(\mu)}$$

Case sustained by leechers

For $C_y < \Lambda$, we do the same reasoning as in Chapter 4. We plug the second restriction of the optimization problem into the first one, we get that

$$\int_E \mu x^*(\mu) d\mu \geq \int_E \frac{\mu \lambda(\mu)}{d(\mu)} d\mu =: D$$

So when $D \geq \Lambda - C_y$, the system is saturated by the download capacity and as a result we achieve the same minimum for the optimization problem

$$x^*(\mu) = \frac{\lambda(\mu)}{d(\mu)}$$

Now when the population of leechers has to be big enough to cope with the download demand of new leechers, we can make the variable change

$$\tilde{x}(\mu) = x^*(\mu) - \frac{\lambda(\mu)}{d(\mu)}$$

where \tilde{x} is a new function that gives the density for the amount of leechers that there are over the minimum imposed by the download capacity restriction. The optimization problem in \tilde{x} is

$$\begin{aligned} \inf \quad & \int_E \tilde{x}(\mu) d\mu \\ \text{s.t.} \quad & \int_E \mu \tilde{x}(\mu) d\mu \geq \Lambda - C_y - D \\ & \tilde{x}(\mu) \geq 0 \quad \forall \mu \end{aligned}$$

Now, we are trying to find a positive function such that $\int_E \tilde{x}(\mu) d\mu$ is minimum while verifying the first inequality. Clearly the minimum is achieved not by a function, but by the distribution

$$\tilde{x}(\mu) = C \delta_{\mu_{max}}(\mu)$$

In that case, the first restriction gives us that

$$C \mu_{max} \geq \Lambda - C_y - D$$

and as we were trying to find the “smallest” function (or distribution in this case) in some sense, the minimum will be achieved for

$$\tilde{x}(\mu) = \frac{\Lambda - C_y - D}{\mu_{max}} \delta_{\mu_{max}}(\mu)$$

Remark B.2.1. Note that even though the minimum is only achieved by a distribution, we can build a function $r^*(\mu) = r(\mu, x^*(\mu), y^*(\mu))$ that is arbitrarily close to that minimum as in Figure B.1.

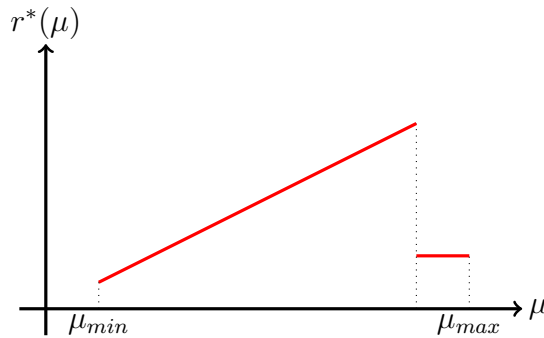


Figure B.1. Approximation for the resource allocation with minimum time.

Introducing incentives

The problem with the previous allocation is that it does not give the proper incentives to the peers. The allocation purposely gives less resources to the ones that contribute the most to the network, this is reflected in the fact that the function r is not an increasing function. We can put this as a new restriction in our optimization problem, but as it is not easy to translate into a restriction for the function $x^*(\mu)$, we will state our optimization problem directly as the variational problem in $r^*(\mu) = r(\mu, x^*(\mu), y^*(\mu))$.

$$\begin{aligned} \inf \quad & \int_E \frac{\lambda(\mu)}{r^*(\mu)} d\mu \\ \text{s.t.} \quad & \int_E \frac{\mu\lambda(\mu)}{r^*(\mu)} d\mu \geq \Lambda - C_y \\ & r^*(\mu) \leq d(\mu) \quad \forall \mu \\ & r^*(\mu) \text{ is an increasing function} \end{aligned}$$

We will assume that we are working again under the hypothesis that $D \leq \Lambda - C_y$, so the network is not entirely saturated by its download capacity. Now, we can find a resource allocation function r that minimizes the download times but is only non decreasing

$$r^*(\mu) = d(\mu)1_{\{\mu < \mu^*\}} + d(\mu^*)1_{\{\mu \geq \mu^*\}}$$

where μ^* is the one that verifies the “conservation of mass” equation

$$\int_{\mu_{min}}^{\mu^*} \frac{\mu\lambda(\mu)}{d(\mu)} d\mu + \int_{\mu^*}^{\mu_{max}} \frac{\mu\lambda(\mu)}{d(\mu^*)} d\mu = \Lambda - C_y$$

This is analogous to the case with finite number of upload capacities, and the allocation profile is also very similar (Figure B.2).

Remark B.2.2. We have a critical value of $d(\mu_{min})$ for which we have $\mu^* = \mu_{min}$. This value is

$$d(\mu_{min}) = \frac{1}{\Lambda - C_y} \int_E \mu\lambda(\mu) d\mu$$

In this case, the allocation is just constant.

B.2.1 Case with uniform arrivals and seed departures

Lets analyze with further detail the cases when the arrival rate is uniform over the possible upload capacities

$$\lambda(\mu) = \frac{\Lambda}{\mu_{max} - \mu_{min}} \quad \forall \mu$$

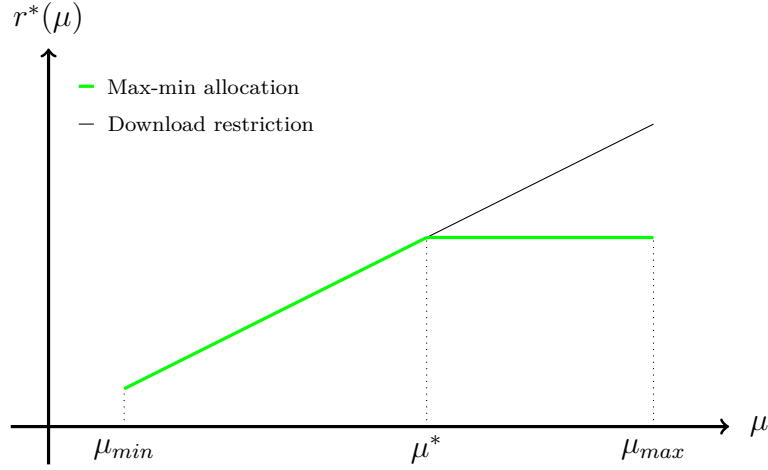


Figure B.2. Profile of a resource allocation with max-min fairness.

Also, we take the rate of departures of seeders also constant

$$\gamma(\mu) = \gamma \quad \forall \mu$$

Lastly, we take the download capacity function to be

$$d(\mu) = K\mu$$

with $K \geq 1$.

We want to compare the average downloading times for the peers in different scenarios, which by Little's law is

$$\bar{T} = \frac{1}{\Lambda} \int_E \frac{\lambda(\mu)}{r(\mu)} d\mu$$

Saturated by download capacity

In this case, the resource allocation function is $r(\mu) = K\mu$. Then the average download time is

$$\bar{T} = \frac{1}{\mu_{max} - \mu_{min}} \int_{\mu_{min}}^{\mu_{max}} \frac{1}{K\mu} d\mu = \frac{\log\left(\frac{\mu_{max}}{\mu_{min}}\right)}{(\mu_{max} - \mu_{min})K}$$

which is the minimum that can be achieved by the network at any time.

Sustained by leechers

If we use the resource allocation policy that has no regard for incentives, we get the average download time is

$$\bar{T} > \frac{\log\left(\frac{\mu_{max}}{\mu_{min}}\right)}{(\mu_{max} - \mu_{min})K} + \frac{\Lambda\left(1 - \frac{1}{K}\right) - C_y}{\Lambda\mu_{max}}$$

Note that we get the same base download time plus a term that comes from the added time from the fast peers.

Recall that when we introduced incentives, the download rate should be

$$r^*(\mu) = d(\mu)1_{\{\mu < \mu^*\}} + d(\mu^*)1_{\{\mu \geq \mu^*\}}$$

In this case, if $\mu_{min} > 0$, we have a critical value of K for which we have $\mu^* = \mu_{min}$. This value is

$$K_c = \frac{\gamma(\mu_{max} + \mu_{min})}{2\mu_{min} [\gamma - p(\mu_{max} + \mu_{min})]}$$

and

$$\mu^* = b + \sqrt{b^2 - \mu_{max}^2}$$

where

$$b = \mu_{min} + K \left[\mu_{max} - \mu_{min} - \frac{p}{2\gamma} (\mu_{max}^2 - \mu_{min}^2) \right]$$

We have two distinct cases for the average download time. When $K \leq K_c$

$$\bar{T} = \frac{1}{K(\mu_{max} - \mu_{min})} \left[\log\left(\frac{\mu^*}{\mu_{min}}\right) + \frac{\mu_{max}}{\mu^*} - 1 \right]$$

and when $K > K_c$ we have that

$$r^*(\mu) = K_c \mu_{min}$$

then

$$\bar{T} = \frac{1}{K_c \mu_{min}} = \frac{2[\gamma - p(\mu_{max} + \mu_{min})]}{\gamma(\mu_{max} + \mu_{min})}$$

In any case, note that this corresponds to a notion of max-min fairness in the download rates.

Appendix C

Markov chains

A Markov chain is a stochastic process named after the Russian mathematician Andrey Markov, who published the first results on these processes in 1906. After that, Kolmogorov generalized his results for countably infinite state spaces. It one of the simplest stochastic processes and their applications are everywhere, from Engineering to Biology. Here we will present some basic definitions and properties. An introduction to this subject can be found in [28].

Definition C.0.1 (Markov process). A family of random variables $\{X(t)\}_{t \geq 0}$ with values over a measurable state space (E, \mathcal{A}) is a Markov process if

$$P(X(t_{n+1}) \in A_{n+1} | X(t_n) \in A_n, \dots, X(t_0) \in A_0) = P(X(t_{n+1}) \in A_{n+1} | X(t_n) \in A_n) \quad (\text{C.1})$$

for all measurable sets A_k and for all increasing sequences $t_0 < \dots < t_n < t_{n+1}$.

This property gives the memoryless nature to the Markov processes, as the conditional probabilities only depend on the last known state of the process. In some sense, the process “forgets” what happened earlier.

It is worth specializing the above to discrete and continuous time cases.

C.1. DISCRETE TIME

Definition C.1.1 (Discrete time Markov chain (DTMC)). A DTMC is a Markov process $X(k)$ with $k \in \mathbb{N}$ and countable state space E . Furthermore, the process is said to be homogeneous if

$$p_{ij}(k) = P(X(k+1) = j | X(k) = i)$$

does not depend on k .

As the state space is countable, verifying this condition on the points of the state space is enough, as they generate the whole σ -algebra of measurable sets.

The transitions of a homogeneous chain is completely determined by the transition matrix (or transition kernel) $P = (p_{ij})$.

In order to determine the trajectory, apart from the matrix P , we need the initial condition. In this case, it is a probability distribution over the state space E , which is represented by a row vector π_0 such that

$$\pi_0(i) = P(X(0) = i)$$

Although the actual trajectory of the process is random, we can define a discrete dynamical system that perfectly describes the probability distribution for the process at time n as:

$$\pi_n = \pi_0 P^n$$

where $\pi_n(i) = P(X(n) = i)$.

Definition C.1.2 (Invariant distribution). A probability distribution π over E is invariant for the DTMC with transition matrix P if $\pi P = \pi$.

If the initial condition is invariant with respect to P , then it is clear that

$$\lim_{n \rightarrow \infty} \pi P^n = \pi$$

which means that the limit distribution of the chain starting with distribution π is π . It is also clear to see that any limit distribution must be an invariant distribution. Furthermore, under some conditions on the transition matrix P it can be proved that there exists a unique invariant distribution for the chain, and that the dynamics converge to it regardless of the initial condition π_0 .

Definition C.1.3 (Irreducible chain). A DTMC with transition matrix P is said to be irreducible if for every $i, j \in E$, there exists a positive integer n_{ij} such that $p_{ij}^{(n)} > 0$, where $p_{ij}^{(n)}$ is the ij entry of the matrix P^n .

Definition C.1.4 (Aperiodic chain). Let T_i^{min} be the minimum return time to state i starting from state i . A DTMC is said to be aperiodic if $\gcd\{T_i^{min} : i \in E\} = 1$.

Definition C.1.5 (Recurrence). Let T_i be the random variable of the return time to state i starting from state i . A DTMC is said to be recurrent $P(T_i < +\infty) = 1 \forall i \in E$. Furthermore it is said to be positive recurrent if $E[T_i] < +\infty \forall i \in E$.

Theorem C.1.1. *Let $X(k)$ be a homogeneous DTMC with transition matrix P which is irreducible, aperiodic and positive recurrent \Rightarrow There exists a unique invariant distribution π for P and*

$$\pi = \lim_{n \rightarrow \infty} \pi_0 P^n$$

for all initial conditions π_0 . In that case, it is said to be ergodic.

Definition C.1.6 (Reversibility). A homogeneous DTMC with transition matrix P is said to be reversible if there exists an invariant distribution π such that verifies the so called *detailed balance equations*:

$$p_{ij}\pi(i) = p_{ji}\pi(j) \quad \forall i, j \in E \quad (\text{C.2})$$

A reversible Markov chain is statistically indistinguishable from the process obtained by reversing the time. But what makes them specially appealing is that it is much easier to find an invariant distribution for them using (C.2), instead of using that $\pi P = \pi$.

C.2. CONTINUOUS TIME

Definition C.2.1 (Continuous time Markov chain (CTMC)). A CTMC is a Markov process $X(t)$ with $t \in [0, +\infty)$ and countable state space E . Furthermore, the process is said to be homogeneous if

$$p_{ij}(t, s) = P(X(t + s) = j | X(t) = i)$$

does not depend on t .

Definition C.2.2 (Transition semigroup). The transitions of a homogeneous chain is completely determined by the transition semigroup $P(t) = (p_{ij}(t))$.

By the Chapman-Kolmogorov equation

$$P(s + t) = P(s)P(t)$$

it is seen that $P(t)$ is actually a semigroup with $P(0) = Id$. Then, if we assume that $P(t)$ is differentiable at $t = 0$, we get that

$$P'(t) = \left. \frac{\partial P(s + t)}{\partial s} \right|_{s=0} = \left. \frac{\partial P(s)P(t)}{\partial s} \right|_{s=0} = P'(0).P(t)$$

equivalently we get that

$$P'(t) = \left. \frac{\partial P(s + t)}{\partial s} \right|_{s=0} = \left. \frac{\partial P(t)P(s)}{\partial s} \right|_{s=0} = P(t).P'(0)$$

These are known as backward and forward Kolmogorov equations respectively.

We can find an expression for $P(t)$ as a function of time solving the differential equations

$$P'(t) = Q.P(t)$$

where $Q = P'(0)$ and it has initial condition $P(0) = Id$. Solving this matrix differential equation yields

$$P(t) = e^{Qt}$$

when e^Q is well defined.

An equivalent definition in this case is that the time that the chain stays in state i is an exponential random variable with parameter $-q_{ii}$. Furthermore, given that it leaves state i , it jumps to state j with probability $-\frac{q_{ij}}{q_{ii}}$. As a result, the dynamics of the CTMC is completely determined by the matrix Q .

Now given the initial condition π_0 , the distribution at time t is

$$\pi(t) = \pi_0 P(t)$$

then

$$\pi(t) = \pi_0 e^{Qt}$$

Remark C.2.1. The matrix Q is called the *transition rates matrix* and it has the following structure:

- $-\infty < q_{ii} < 0$
- $-q_{ii} = \sum_{j \in E, j \neq i} q_{ij}$

As a result, $\sum_{j \in E} q_{ij} = 0 \forall i$ and thus Q has 0 as an eigenvalue.

Definition C.2.3 (Invariant distribution). A probability distribution π over E is invariant for the CTMC with transition rates matrix Q if $\pi Q = 0$.

Effectively, if $\pi = \pi e^{Qt}$, taking the derivative with respect to time we obtain $0 = \pi Q e^{Qt}$ for all t . As a result it must be $\pi Q = 0$.

The definitions of irreducibility and recurrence are analogous as in the discrete time case, but the notion of periodicity does not make sense in continuous time, as the probability of returning to a state in a fixed time (or a multiple of it) is zero. We then have the following ergodic theorem.

Theorem C.2.1 (Ergodic theorem). *Let $X(t)$ be a homogeneous CTMC with transition rates matrix Q which is irreducible and positive recurrent \Rightarrow There exists a unique invariant distribution π for Q and*

$$\pi = \lim_{t \rightarrow \infty} \pi_0 P(t)$$

for all initial conditions π_0 .

As in the discrete time case, we also have a notion of reversibility for CTMC.

Definition C.2.4 (Reversibility). A homogeneous CTMC with transition rates matrix Q is said to be reversible if there exists an invariant distribution π such that

$$q_{ij}\pi(i) = q_{ji}\pi(j) \quad \forall i, j \in E$$

Appendix D

Graph theory

In this appendix we will present some basic definitions and some important examples of graphs, which are used throughout this thesis. An introduction can be found in [29].

D.1. DEFINITIONS

There are several definitions of graph depending on the context and intended applications. Here we will give the definition that better suits our needs.

Definition D.1.1 (Graph). A graph G consists of a *vertex* set V and an *edge* set E , where an edge is a pair of different vertices of G . Note that there can be either zero or one edge between two vertices in a graph and there cannot be an edge between a vertex and itself.

If ij is an edge, then we say that i and j are *adjacent* or that i is a *neighbor* of j , and denote this by writing $i \sim j$. Also a vertex is said to be *incident* with an edge if it is one of the two vertices of the edge.

We will focus in finite graphs, in which the number of vertices is finite. Furthermore, if the edges are an ordered pair of vertices, then the graph is said to be *directed*. In other case it is said to be *undirected*.

Definition D.1.2 (Subgraph). Given a graph $G = (V, E)$ and a subset of vertices $U \subset V$, the subgraph of G generated by U is the graph $H = (U, F)$, where $F \subset E$ is the set of edges which are determined by a pair of vertices in U .

Now we define two important cases of subgraphs.

Definition D.1.3 (Independent set). Given a graph $G = (V, E)$, an independent set is a subset of the vertices $S \subset V$ such that it has no edges as a subgraph.

As there are no edges in an independent set, we use the term *independent set* when we are referring to the set of vertices and when we are referring to the induced subgraph.

Definition D.1.4 (Clique). Given a graph $G = (V, E)$, a clique is a subgraph of G such that all vertices are joined by an edge.

When the whole graph is a clique, the graph is called *complete* or *full mesh*.

The *degree* of a vertex is the number of neighbor that it has. For instance, the degree of every vertex in the subgraph generated by an independent set is zero.

Definition D.1.5 (Regular graph). A graph $G = (V, E)$ is said to be regular or k -regular if the degree of all vertex is equal to k .

An independent set, a clique and a complete graph are examples of regular graphs.

Now we state a theorem for the existence of regular graphs for a fixed finite amount of vertices.

Theorem D.1.1 (Erdős and Gallai [30]). *A m -regular graph of N vertices exists $\Leftrightarrow m \cdot N$ is even and $N > m$.*

Despite the abstract nature of the definition of a graph, it can be fully characterized by a 0 – 1 matrix.

Definition D.1.6 (Adjacency matrix). The adjacency matrix $A = (a_{ij})$ of a graph $G = (V, E)$ is the 0 – 1 matrix with rows and columns indexed by V , such that $a_{ij} = 1$ if and only if $ij \in E$.

Note that in a undirected graph the adjacency matrix is symmetric, as the edges are undirected and thus if $ij \in E$ then $ji \in E$. Conveniently the adjacency matrix of a subgraph is just the submatrix of the adjacency matrix of the original graph.

D.2. CAYLEY GRAPH

There is an example of a graph that deserves particular attention.

Definition D.2.1 (Cayley graph). Consider the algebraic group \mathbb{Z}_n and a subset $B \subset \mathbb{Z}_n$ such that $0 \notin B$ and B is closed under opposites. Then the Cayley graph $C(\mathbb{Z}_n, B)$ is the graph with the elements of \mathbb{Z}_n as vertices such that $i \sim j$ if and only if $i - j \in B$.

Note that the Cayley graphs are always $2k$ -regular for some k (every vertex has the same even degree).