QLectives – Socially Intelligent Systems for Quality
Project no. 231200

Instrument: Large-scale integrating project (IP)
Programme: FP7-ICT

Deliverable D1.4.1

Modelling the dynamics of quality collectives

Submission date: 2012-03-01

Start date of project: 2009-03-01 Duration: 48 months

Organisation name of lead contractor for this deliverable: ETH Zurich

Project co-funded by the European Commission within the Seventh Framework Programme (2007-2013)		
Dissemination level		
PU	Public	X
PP	Restricted to other programme participants (including the Commission Services)	
RE	Restricted to a group specified by the consortium (including the Commission Services)	
CO	Confidental, only for members of the consortium (including the Commission Services)	

Document information

1.1 Author(s)

Author	Organisation	E-mail
Steve Genoud	ETHZ Zurich	sgenoud@ethz.ch
Nigel Gilbert	UniS	n.gilbert@surrey.ac.uk

1.2 Other contributors

Name	Organisation	E-mail
Tamás Vinkó	TUD	T.Vinko@tudelft.nl
Sergi Lozano	ETH Zurich	slozano@ethz.ch
Andrzej Nowak	UWAR	andrzejn232@gmail.com
Michal Ziembowicz	UWAR	ziembowicz@gmail.com

1.3 Document history

Version#	Date	Change
V0.1	07-01-2012	First pre-alpha draft internal consortium version
V0.2	20-01-2012	First draft internal consortium version
V1.0	17-02-2012	Approved version to be submitted to EU

1.4 Document data

Keywords	QLectives, peer-to-peer, user behavior
Editor address data	sgenoud@ethz.ch
Delivery date	17 February, 2012

1.5 Distribution list

Date	Issue	E-mail
	Consortium members	QLECTIVES@list.surrey.ac.uk
	Project officer	Jose.FERNANDEZ-VILLACANAS@ec.europa.eu
	EC archive	INFSO-ICT-231200@ec.europa.eu

QLectives Consortium

This document is part of a research project funded by the ICT Programme of the Commission of the European Communities as grant number ICT-2009-231200.

University of Surrey (Coordinator)
Department of Sociology/Centre for Research in Social Simulation
Guildford GU2 7XH
Surrey
United Kingdom
Contact person: Prof. Nigel Gilbert
E-mail: n.gilbert@surrey.ac.uk

University of Fribourg
Department of Physics
Fribourg 1700
Switzerland
Contact person: Prof. Yi-Cheng Zhang
E-mail: yi-cheng.zhang@unifr.ch

Technical University of Delft
Department of Software Technology
Delft, 2628 CN
Netherlands
Contact Person: Dr Johan Pouwelse
E-mail: j.a.pouwelse@tudelft.nl

University of Warsaw
Faculty of Psychology
Warsaw 00927
Poland
Contact Person: Prof. Andrzej Nowak
E-mail: nowak@fau.edu

ETH Zurich
Chair of Sociology, in particular Modelling and Simulation
Zurich, CH-8092
Switzerland
Contact person: Prof. Dirk Helbing
E-mail: dhelbing@ethz.ch

Centre National de la Recherche Scientifique, CNRS
Paris 75006,
France
Contact person: Dr. Camille ROTH
E-mail: camille.roth@polytechnique.edu

University of Szeged
MTA-SZTE Research Group on Artificial Intelligence
Szeged 6720, Hungary
Contact person: Dr Mark Jelasity
E-mail: jelasity@inf.u-szeged.hu

Institut für Rundfunktechnik GmbH
Munich 80939
Germany
Contact person: Dr. Christoph Dosch
E-mail: dosch@irt.de

QLectives introduction

QLectives is a project bringing together top social modelers, peer-to-peer engineers and physicists to design and deploy next generation self-organising socially intelligent information systems. The project aims to combine three recent trends within information systems:

- **Social networks** - in which people link to others over the Internet to gain value and facilitate collaboration
- **Peer production** - in which people collectively produce informational products and experiences without traditional hierarchies or market incentives
- **Peer-to-Peer systems** - in which software clients running on user machines distribute media and other information without a central server or administrative control

QLectives aims to bring these together to form Quality Collectives, i.e. functional decentralised communities that self-organise and self-maintain for the benefit of the people who comprise them. We aim to generate theory at the social level, design algorithms and deploy prototypes targeted towards two application domains:

- **QMedia** - an interactive peer-to-peer media distribution system (including live streaming), providing fully distributed social filtering and recommendation for quality
- **QScience** - a distributed platform for scientists allowing them to locate or form new communities and quality reviewing mechanisms, which are transparent and promote quality

The approach of the QLectives project is unique in that it brings together a highly inter-disciplinary team applied to specific real world problems. The project applies a scientific approach to research by formulating theories, applying them to real systems and then performing detailed measurements of system and user behaviour to validate or modify our theories if necessary. The two applications will be based on two existing user communities comprising several thousand people - so-called "Living labs", media sharing community tribler.org; and the scientific collaboration forum EconoPhysics.

Executive summary

BitTorrent as an internet based technology is relatively old. In the decade since its appearance, research has focused on its technical strengths and its weaknesses to attacks, aiming to improve the protocol. This literature is not much concerned with the behaviour of users. Most of the models consider a static user whose behaviour remains unchanged despite changes in the system's performance. In this deliverable we put the focus on two dynamical effects that a more complex approach to users in the BitTorrent system introduces. First, we consider the possible changes of social roles that users can take in the BitTorrent community. The second model attempts to reproduce the sharing behaviours of users as the performance they experience evolves.

The first agent-based model introduces behaviours that are more related to the role that users play in BitTorrent as a community. Every user has a main role related to his attitude towards content (from pure producer to pure consumer). In parallel, users can also take a role in the community, giving respect to users who produce content and share. Then, the decision rules for the main role are partially based on the respect generated by the community.

This first model is very abstract and can only represent reality in a metaphoric way. Nevertheless, we see that it approximates reality. More significantly, it allows us to identify some important structural facts. We highlight the crucial importance of the content creators, for without them the system cannot run. Also, we see that there must be enough community activity to motivate them or motivate others to become content creators.

The second model consists of an agent based model of BitTorrent as a techno-social system. A detailed model of the protocol is composed with a decision model of the users. The model of the protocol focuses only on the technical aspect of BitTorrent – the exchange of data – and needs to be informed by the behaviour of users. While we use a data driven method as the source of choices that users make relative to the content they provide or consume, we model their sharing behaviour. We assume two general types of sharing behaviour, either users contribute a lot to the community or they contribute as little data as possible. The decision process of users is supposed to follow an aspiration-satisfaction mechanism. This assumption is similar to what was used in Stream 2 research, in particular in deliverables D2.1.1 and D2.1.3. A dynamic effect is introduced as the sharing behaviour influences the performance of the system, which then has repercussions on the behaviour of users.

The resulting model should obey the macro social behaviour we observe in real communities. We use measurements of the performance of actual systems, an upgraded version of a dataset from deliverable D3.1.1, to calibrate the parameters of the model. Unfortunately, the comparison shows that there is no region of the parameter space where our model reproduces reality. We identify two possible sources of the failure. The assumption – two single behavioural types – might be too simple. Or the decision model might be too sensitive to random fluctuations. Our next step will be to gain insight in the system from data. This analysis of user behaviour will be presented in Stream 3 deliverable D3.3.1. Then, these new insights should allow us to reformulate our hypotheses and build a better model.

Contents

Chapter 1

Introduction

The current literature on BitTorrent is not much concerned by the behaviour of the users of the system. The reseach focuses on the strength and weaknesses of BitTorrent as a protocol. Often, researcher use measurements of the behaviour of actual users to test their new protocols. This data driven methodology has an important drawback — it ignores dynamical effects the changes of the protocol induce on the user behaviour. In this deliverable, we try to provide two different approaches that complements purely data driven models with models of user behaviour.

The first model focuses on the role that the sense of community might have on the motivation of users. The model is built sucht that users have a type of behaviour in relation to BitTorrent as a community. His type or role determines if he user shares the content he downloaded, creates new content and is active not only the file sharing aspect of BitTorrent, but also in the discussion board, the community at large. The dynamic aspect appears as users can change their type depending on the behaviour of other users.

The second model ignores the influence that BitTorrent as a community has on the behaviour of users. BitTorrent is described there as a pure techno-social system where the only interactions users have are mediated by the BitTorrent protocol. As the interaction in forums is ignored, users behaviour is supposed to depend only on the performance of the system they experience. Also, a detailed model of the protocol is used to reproduce the performance accurately. The aim of this model is to offer a dynamical framework for the evaluation of new reputation systems that can be added to BitTorrent.

Chapter 2

BitTorrent as an evolving community

This chapter introduces the first model of user behaviour that we studied. In this model, the focus is on user roles, that is a classification of users according to their behaviour, and the consequences of the activity of the users on the dynamics of peer-to-peer communities. In this preliminary model, we are not so much concerned with accurately reproducing any empirical example of peer-to-peer file sharing, but rather investigating the emergent properties of the model. In later chapters, we describe a more sophisticated version that incorporates a model of the BitTorrent protocol, which here is glossed over, and that is capable of being calibrated against network data.

2.1 Objectives

The aim of this initial model was to gain a better understanding of three issues:

1. Who (what type of user) is most important for the dynamics of peer to peer systems?
2. How can we increase altruism i.e. seeding?
3. What are the dynamics and interrelation of swarms?

To begin, we assume that those involved in peer-to-peer (p2p) networks may be classified into four roles: 'pros', 'seeders', 'lurkers' and 'leeches'. 'Leeches' are those who download but, because of technical limitations or ignorance, do not upload, even while they are downloading (the BitTorrent protocol penalises such users, as noted in the next chapter). 'Lurkers' are the great majority of users, those who download, and while doing so and possibly for some time thereafter, uploads content to the rest of the swarm. Most work on p2p protocols is aimed at maximising the time that lurkers spend uploading, an activity that gives them no direct benefit and can cost them convenience or even bills for bandwidth. 'Seeders' in the model are those who actively participate in making content available, even when they are not downloading; they may for example have a server that is always on and available. Pros are the experts who initiate and supply the network with content and we hypothesise that their presence, even if only in small numbers, is crucial to the functioning of the network.

In this chapter, we are primarily interested in the 'Pros' and 'Seeders'. Leeches are the main centre of attention in the subsequent chapter. Here, we are interested in what could motivate Pros and Seeders. For Leeches, the motive for allowing uploads is that the BitTorrent protocol ensures that they then get faster downloads. But this motive does not apply to Pros and Seeders, who are not in general interested in downloading content. Instead, we propose that these users are motivated by the 'respect' that other users offer them[1].

Not all users offer respect; to do so, they need to be knowledgeable about p2p networks and interested in those who provide content. But some users, whom we call 'Clubbers', do. They may identify themselves with the network, recognise the nicknames of prominent seeders and pros, contribute comments to bulletin boards and feel themselves to be 'members of the club' (hence the name). Holders of any of the roles may be Clubbers, although Leeches are very unlikely to be Clubbers.

In the following section, we specify the outline of the model more precisely.

2.2 Model specification

2.2.1 Objects

There are three basic types of objects: people ('agents'), collections of peer to peer transactions ('swarms') and items for transfer ('files').

Agents

Agents represent the people who engage in p2p activity. They can take on one of several roles:

- Pro (also known in the literature as a 'first seeder'): someone who introduces new material into the network from outside. A Pro creates a file and then offers it for download in a new swarm (i.e. seeds the swarm). The

[1] Note the contrast between this and the assumption often made that it is necessary to design the underlying protocol in a way that seeders will be incentivised.

Pro continues seeding until interest in the file fades away (it's attractiveness has dropped to zero). Then after a random interval (controlled by the `ripping-rate`), they introduce another new file, and so on.

- Seeder: someone who uploads a file from their collection of files (obtained previously by downloading the file from a swarm, since seeders start with empty collections). They continue to upload until the file loses its attractiveness. They may then either upload another file, or download a file from another swarm to add to their collection.

- Lurker: someone who downloads a file from a swarm. After it has been downloaded, the Lurker uploads the file again to the swarm, for the benefit of other members of the swarm, until `seeding-ratio` per cent of the file has been uploaded. The seeding-ratio that an agent uses is drawn at random from a Gaussian distribution with mean `mean-seeding-ratio` and variance 1. The Lurker then waits for an interval controlled by `activity-rate` before joining a swarm and downloading another file.

- Leech: someone who just downloads files, contributing nothing to a swarm, or waits doing nothing (the proportion of each is controlled by `activity-rate`).

Agents are either not members of any swarm, or a member of just one. Agents cannot interact with more than one swarm at a time (although they can join several swarms one after the other).

As well as a role, agents know about the swarm they belong to (if any), the files that they have gathered into their collection, and the proportion of the current file that they have downloaded or uploaded (if any). Each agent has a maximum bandwidth, which controls how quickly data can be uploaded to the swarm (it is assumed that download speeds are not bandwidth limited, although upload speeds are, which is the case for most users).

Some agents are Clubbers, and all agents have a motivation level, although this is only important for Pros, Seeders and Lurkers.

Swarms

A swarm is represented by an object that tracks the file that is the subject of the swarm, the agents that are currently uploading the file, and those that are downloading it. A swarm is created by a Seeder or Pro, at which time there is just one uploader (that agent) registered with it. Other agents may join the swarm to upload or download the file. A swarm dies when there are no longer any agents uploading to or downloading from it. The same content may be the subject of more than one swarm.

Files

Files are created by Pro agents, for example, by ripping a movie or a song. Each file has a volume (the size of the file in some units), a creation date, and an associated attractiveness. The file's attractiveness (a number greater than or equal to zero) depends on a number of factors, including how many times it has been downloaded, the number of 'seeders', i.e. those uploading the file, and how old it is. Its attractiveness is also affected by exogenous shocks (e.g. the movie winning an Oscar). The attractiveness of a file varies through time as these factors change.

2.2.2 Agent behaviours

Agents are generally 'busy' up or downloading files to and from swarms and while there is still some content to up or download, they continue doing that. What agents do when they are not busy depends on their role. If they are Pros, they create new files and swarms, at the `ripping-rate`. If they are seeders, they start a new swarm with one of the files in their collection. If they are Lurkers or Leeches, they join a swarm with a chance of `activity-rate` at each time step.

The swarm chosen by Lurkers and Leeches to join depends on the attractiveness of the file that they can download from swarms. This is done randomly, but with a bias towards more attractive files.

Leeches and Lurkers continue downloading a file until they have the whole file (the speed of download depends on the number of uploaders and downloaders in the swarm: the greater the sum of the bandwidth provided by uploaders and the fewer downloaders there are, the faster an agent is able to download). Seeders continue to upload until they observe that the attractiveness of the file has decreased to zero. At this point they get discouraged and exit from the swarm. It is possible for the number of uploaders to a swarm to drop to zero while there are still some partially completed downloads in progress. In this case, the downloaders remains in the swarm in the hope that sometime, an uploader will rejoin the swarm.

Agents change roles according to their current motivation level. Their motivation is increased by:

1. downloading, because they are gaining a file;
2. if they are seeding, having a large number of Clubbers in the swarm, because they get 'respect' from the Clubbers.

Motivation is reduced when:

1. they are uploading to a swarm, because the uploading costs them time and bandwidth that they could be using for downloading;

2. and to a lesser degree, while they are waiting, neither uploading nor downloading.

As the agents' motivations increase, they become successively a Lurker, a Seeder and then a Pro. Correspondingly, as motivation decreases, they drop through this hierarchy of roles, with those with the lowest motivation becoming a Leech.

2.3 The model

The model was developed using NetLogo 4.1.2, an open source programming environment designed for prototyping agent-based models. The code is available on the QLectives web site. The user interface is shown in Figure 2.1

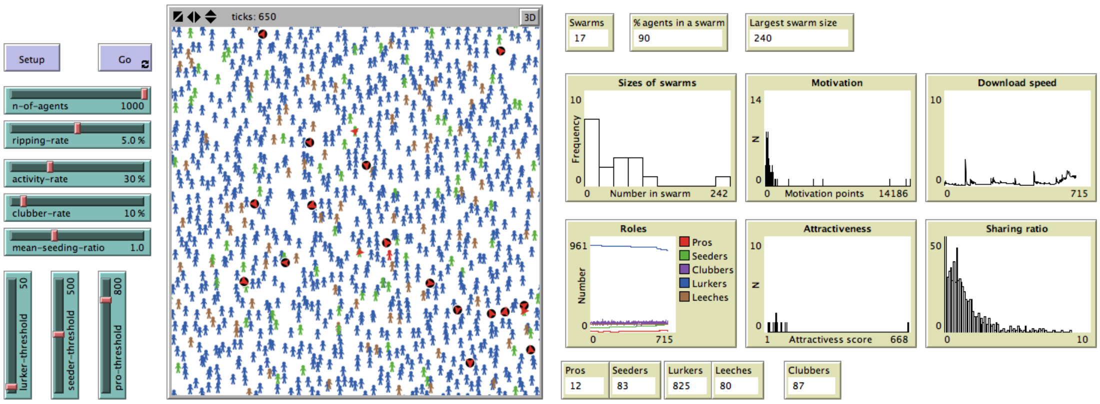

Figure 2.1: The user interface of the model

The Figure shows sliders that can be adjusted to set the various `rates`, a panel showing the agents, coloured according to their roles, with swarms shown as black dots and files as red triangles (the location of agents, swarms and files is of no consequence in this model; swarms and files are placed on top of the Pro agent that started them for convenience). On the right are plots and monitors showing (clockwise from top left):

1. a histogram of the size of swarms
2. a histogram of the motivations of agents
3. a plot of the average instantaneous download speed achieved (this depends on the number of those downloading from a swarm and the bandwidth of those uploading)
4. the sharing ratio: the total each agent has uploaded divided by the total that the agent has downloaded (0 for those who have not up or downloaded anything)

5. a histogram of the attractive scores of files
6. a plot of the changes in roles over simulated time.

Below these are monitors showing the number of each agent in the model and above, the number of swarms and the proportion of agents in a swarm.

2.4 Results

With the parameter settings shown in Figure 2.1 and Table 2.1, the system quickly reaches a steady state, with around 90 Leaches, 750 Lurkers, 155 Seeders and 20 Pros. This occurs by tick 500, a 'tick' being an arbitrary time unit such that the average file takes 50 ticks to upload by the average agent. Most agents (around 90%) find a swarm to join. The swarms vary in size from the largest at around 1205 members down to those with less than 10 (the most frequent). Pros are well rewarded for their activity by Clubbers and their motivation scores steadily increase.

Parameter		Value
Number of agents	n-of-agents	1000
Rate at which Pros input files	ripping-rate	5%
Activity rate	activity-rate	30%
Likelihood of becoming a Clubber	clubber-rate	30%
Proportion that Lurkers upload		
after downloading a file	mean-seeding-ratio	1.0
Motivation to become a:		
Lurker	lurker-threshold	50
Seeder	seeder-threshold	500
Pro	pro-threshold	800

Table 2.1: Baseline parameter settings

After a substantial time (more than 10,000 ticks), some Lurkers become Seeders and thus the number of Seeders gradually increases, while the number of Lurkers decreases.

Given this baseline, we can compare the behaviour of the model when some of the parameters are changed.

2.4.1 The importance of Pros

If the `pro-threshold` is increased to 1000, the initial state of the simulation includes no Pros, because no agents are created at the start with a sufficiently large motivation score. The result is easily predictable: with no Pros, there are no files entering the system, and so no uploading or downloading takes place and there are no swarms created. If the threshold is reduced so that there is just one Pro, activity starts but its continuation relies on whether Seeders 'graduate' to become Pros before the solitary Pro loses motivation and is transformed to a Seeder, and Seeders (with only one file to upload) become discouraged and become Lurkers. When that happens, all activity ceases as there are no swarms to join.

2.4.2 The importance of motivation

If the number of Clubbers is reduced (by reducing the `clubber-rate` from 10% to 5% and therefore the number of Clubbers from about 100 to about 50), thus reducing the motivation afforded to Pros and Seeders, the numbers of Pros and Seeders rapidly drops to zero. Again, this has the effect of removing all active swarms from the system and stifling any activity.

2.4.3 The dynamics of swarms

The sharing ratio (the ratio of uploaded to downloaded content per agent) depends in a complicated way on the individual agent's seeding ratio (the time a Lurker spends uploading after downloading a file), the relative upload and download speeds, and the number of upholders in a swarm. Empirical evidence taken from a 'private' p2p network (see Figure 3.3) suggests that the sharing ratios should be roughly Poisson distributed, with $\lambda = 1$. Figure 2.1 shows that the model does approximately have such a distribution.

2.5 Extensions

The model was designed to be a simple way of developing intuitions about the roles of various types of user behaviour, as encapsulated in the model's agent roles. In contrast to much research on p2p, including the BitTorrent protocol itself, in this model the motivation of agents is not determined by some kind of social dilemma game (e.g. the Tit-for-Tat strategy employed by BitTorrent), but rather by the simpler but also possibly more realistic assumption that some participants are motivated by a desire for respect from others. Of course, both this and the more usual game theoretic assumptions are ideal-types, that is, simplifications of the social world to assist in analysis and understanding. Neither set of assumptions can be said to be 'correct'—indeed, both must be wrong if we expect them to apply exactly and equally to everyone involved in peer-to-peer activities.

The advantage of the simple model described in this chapter is that it can easily be modified to experiment with different assumptions, including, for example, game theoretic strategies. Other possibilities are to make the seeding ratio depend on the agent's role and history, instead of bring fixed as in the current model, and to try various rules for penalising or encouraging uploading, including various designs for increasing the number of Clubbers and their influence.

2.6 Conclusions

An agent-based model of users' behaviour in p2p networks has been formulated to examine the efficacy of behavioural assumptions that do not depend on game theoretic strategies. A set of roles and their behaviours was defined and the model showed that with these, network dynamics that approximated to those found in the target p2p networks could be obtained. However, this model is merely a proof-of-concept. If work on it were to be continued, it should be compared more closely with empirical data, subjected to sensitivity analysis, and then used to study the effect of alternative behavioural designs.

Rather than do this, we decided to implement a more sophisticated model that is able to simulate an actual BitTorrent protocol, as used in QPlatform, and this is described in the next chapter.

Chapter 3

BitTorrent as a techno-social system

In the previous chapter we explored the possible influence of the community of the evolution of a system similar to BitTorrent. This chapter focuses on the *sharing behaviour* of users, how much data users give to the community compared to how much they use the system. This work results from a close collaboration between TUD and ETH Zurich. Each of the partner provided knowledge either in the BitTorrent protocol (TUD) or in human behaviour modelling (ETH Zurich).

In the first section we explain the basic reasoning behind this study. Then we expose a detailed agent based model of BitTorrent as a *techno-social system*. The resulting model should reproduce the macro social behaviour we observe in real communities. In the fourth section we discuss how the use measurements of the performance of actual systems can calibrate the parameters of the model, as well as the first results we obtained.

3.1 Motivation and theoretical considerations

This section introduces the general problem our model attempts to solve. We suppose the reader already has a good understanding of peer-to-peer systems, and more precisely BitTorrent. We first distinguish the concepts of *peers* and *users*. Then, we sketch a model articulated around those concepts. We finally discuss how this projects integrates in QLectives.

We give here the necessary definitions of BitTorrent for the further discussion. The BitTorrent protocol allows users to join a *swarm* of hosts (or *torrent*) to download and upload from each other simultaneously. Each swarm consists of two type of users: *seeders* who are only uploading (they do have the full content) and *leechers* who are downloading (and uploading at the same time to other leechers as well). Content files are split into *pieces*, allowing leechers which are still downloading to serve the pieces they already have to others.

3.1.1 Users and peers

The BitTorrent protocol is complex, and most of its users have a mental model different to what the protocol actually does. We want to distinguish two types of agents who correspond to two separate levels of functionality in BitTorrent as a system:

- The User Interface
- The Protocol

On one hand, the User Interface corresponds to how the agent we call the **user** interacts with BitTorrent. The current mental model of BitTorrent users is simple. We start with a user who is interested in some content. She finds a torrent file that corresponds to the content she is interested in. She loads this file in the BitTorrent client, which starts the download of the content. When the download finishes she chooses to continue sharing (cooperation) or to close the connection to this swarm (defection). The information the client gives the user consists mostly of the download speed and her sharing ratio (total upload amount divided by the total download amount).

The Protocol, on the other hand, describes how the client (we call this agent the **peer**) deals with other peers on the network. In the standard BitTorrent protocol, a Tit for Tat (TFT) strategy is implemented to establish the priorities of the peers to be shared to. The decisions of the peers do not depend on an explicit input of a user. The user is not asked who to share to. The details of the sharing exchange depends only on how the software is implemented.

Note the difference of level at which *users* and *peers* interact. Users are humans, who can have many ways of making decisions. They all have different preferences, which results in variability of behaviour. On the other hand, peers are bits of software. They only implement the algorithm the designer of the protocol intended.

3.1.2 Users, peers and Bartercast

A part of the QLectives project focuses on improving the cooperation in P2P systems. Particularly, the Bartercast reputation system and its improvements [1,2] have been implemented to enhance cooperation in P2P systems. In the original BitTorrent TFT strategy users are not incentivised for seeding. Most of the effort has been made to change the protocol, adding mechanisms to fill this gap. The proposed incentive scheme implemented in Bartercast is peer indirect reciprocity. A decentralised system of reputation informs peers about the behaviour of other peers. Based on this information peers can decide whom to upload to. For instance, it can be based on a simple rule: peers with higher reputation get higher preference. It has been shown that using this modified protocol, peers who share more have a better performance (download speed) than the freeriders.

The efficiency of this family of new protocols has been tested thoroughly. Nevertheless, the tests have focused so far on the effects that Bartercast have on peers. The efficiency question focuses on the difference in download speed that peers experience (by opposition to users). Also, these tests use a static user behaviour. In the test models, the user's decisions (when a download starts, if the user is a seeder or only a leecher) are a given[1]. In other words, the user behaviour is completely data driven. The choices made by the agents representing the users do not change with the introduction of incentives.

The aim of introducing an incentive scheme such as Bartercast is to change the user's behaviour. But, as we have argued here, the current literature does not give any information on the influence of incentives. It is only assumed that they induce a change in behaviour, but the scope of this change is not quantified. Moreover, if this happens, it is not clear how the induced dynamics could alter the general behaviour of the system.

3.1.3 Introducing "the social" in BitTorrent models

In our previous section a difference between the *peers* and *users* has been elaborated. The structure of the model of BitTorrent we propose here is also based on this distinction. We build a *techno-social* model, by integrating two submodels, a model of *peers* and a model of *users* behaviour.

First, a model of BitTorrent built by TUD renders the *peer* level. We use the simplified model of the BitTorrent protocol that was developed in the context of QLectives [3]. A similar model has been used for testing Bartercast [2]. The simplified model has been chosen because it allows a faster computation. Section 3.2.1 discusses with more detail the model and the reason for our choice.

Then, in addition to the model proposed by TUD, which focuses on the peer-level, our extension explicitly incorporates the changing behaviour of users. The starting assumption underlying our considerations concerns users sharing behaviour (whether they contribute to the public good or not). We assume it depends on the download speed previously experienced by the user. Also, we suppose that users do not exchange directly information. These assumptions lead us to pick an aspiration-satisfaction model of user behaviour [4,5]. Users choose to change their behaviour according to a learned aspiration towards the system. We discuss this model in detail in section 3.2.2.

[1] Data extracted from measured traces informs the simulation of the user behaviour.

3.1.4 Integration in QLectives

The general QLectives plan introduces a loop between data and models (stream 4 to stream 1). We aim to use such a loop in our model. We use data from from measurements of actual systems (we call them *traces*) to calibrate the model using these traces.

The QLectives project also plans that models should inform the development of new algorithms for quality collectives (stream 1 to 2). Once parameters of the model are calibrated to reflect the observed amount of user sharing behaviour well enough, we can change the underlying protocol. The new model, with, for instance, Bartercast as the peer protocol should evolve differently from the model with the original protocol.

One of the greatest difficulties of an interdisciplinary project like QLectives concerns the interest that different partners can have in a same question. Even when the broad project appeals to everyone, particular questions can often have applications only for one partner. In this dynamic model of BitTorrent communities we spent a special care on this aspect. Both main partners, TUD and ETH Zurich, can get interesting results for their own fields.

3.2 Implementation details

In the first section we described our model as a composition of two parts, representing the *peer* level as well as the *user* level. In this section we spend some time explaining the detailed implementation of the models of both parts, as well as how they fit together.

3.2.1 The BitTorrent model

First, a model of the BitTorrent protocol stands for the *peer* level. In order to speed up the computation, we use a simplified model of the exchange of pieces, as described by Meulpolder et al. [3]. An important assumption of the model is that peers always find a piece to download in other peers[2]. We use an implementation of the analytical model for our needs.

Model inputs

The implementation of the model, the simulation, needs specific information about the general behaviour of users. We can distinguish two kinds of decisions users take, decisions about:

[2] This hypothesis is valid for files with a high number of pieces. See [6] for a justification of the hypothesis.

- which file to download and when (torrent choices)
- when to stop seeding a file (sharing behaviour).

We are not interested equally in both kinds of decisions. Our goal is to improve the sharing behaviour of users. We assume that the torrents choices are independent of any mechanism that influences sharing behaviour. Also, we suppose that the dynamic effects of incentive schemes should not influence the choice of content[3]. This means that the model we build focuses on the sharing behaviour of users related to seeding and not providing new content.

Two different sources of information provide the two kinds of inputs. On one had, torrents choices, as in the classical implementations, are completely data driven. We use traces, measurements of actual BitTorrent communities. They define the existing swarms, which users join them, and at which time in the simulation.

Furthermore, the sharing behaviour of users is driven by a user model we develop. The original implementation of the simulation has been adapted in order to allow various types of sharing behaviour. In the first model we present here, the sharing behaviour allowed to users is simple: freeriding or sharing[4]. A *freerider* has a hit and run strategy, she leaves the swarm and stops sharing as soon as her download is finished. A *sharer* stays in the swarm for 18 hours after his download ends[5].

Finally, we need to specify the technical limitations on speed of download and upload. As the user sharing behaviour does not influence his technical limitations, we do not need to model them. In first approximation, we fixed the upload speed at 50kB/s and the download speed at 200kB/s. This corresponds to the standard values for a DSL connexion in 2006[6].

Use of traces

As input for the torrent choices we use a four month long trace of the FileList.org BitTorrent tracker collected at the beginning of 2006. The detailed description of the measurement method can be found in Roozenburg [7]. It is important to note that FileList is a closed community, with a ratio enforcement system — users

[3]These assumptions differ totally from the approach of chapter 1.

[4]The sharing behaviour will also be called strategies or types in this chapter.

[5]The 18 hours number is a rough estimate from the data. The sharing times form a continuous distribution. As we have only one type of user who shares, we can approximate the time for sharer from a central measure of the distribution. The mean seeding time of all downloads in the FileList trace is around 30 hours, the median 10 hours. We choose the arbitrary number of 18 hours.

[6]The traces we used have been measured between December 2005 and March 2006 for the FileList trace.

are forced to share to a minimum sharing ratio (upload volume divided download volume). The system also requires users to have a common identity across swarms. Because of the nature of the protocol, measurements of open BitTorrent communities cannot have user based information, only peer based. Because our model needs input about the user behaviour, we need to use a measurement of a close community such as FileList.org.

A great technical difficulty of modelling the evolution of a BitTorrent system derives from the number of users of the system at anytime. In both traces we have access to, the simultaneous number of peers at anytime is between 40 000 and 65 000. The computation of the whole system is therefore very slow. For our practical purposes we therefore need to work with subsets of the trace. We build subsets of about 100 users interacting over one week.

The creation of these subsets requires caution. As a first selection algorithm we would choose randomly a subset of users active during a period of interest. This naive algorithm has an important drawback: it does not reproduce the actual structure of the system. Imagine that the system is composed of 200 swarms with 2 peers in each and another unique swarm with 100 peers. A random selection of 50 peers will, on average, select only 10 peers in the main swarm while most of the 40 other will be alone in their own swarm. The performance of this system will be very different from the original system (most of the peers will not exchange anything). The non-uniform shape of the distribution of number of peers in swarms makes this effect important in the normal system and forces us to choose a better algorithm.

Selected users, and therefore their peers, must interact with each other. Our algorithm should provide a few torrents where most of the users are active. The algorithm we have developed is the following, for obtaining a subset of N users (for instance $N = 100$):

1. select a subset of torrents (for instance 10 torrents) which has a cumulative number of different user greater or equal to N.
2. choose one of the torrents, add all the users of this torrent
3. do the previous step until N users are selected.
4. remove all torrents with less than a minimum number of agents (for instance 4).

We test the validity of the algorithm with the Q-Q plot presented in Figure 3.1. The distributions of average download speeds for subtraces and the full trace are compared. We see that both distribution are similar. There is an important difference between the distributions for small download speeds. Also, the smaller traces tend to have more users who have a speed multiple of 50kB/s. This is due to the fact that the subtraces do not have enough users to have many cases where

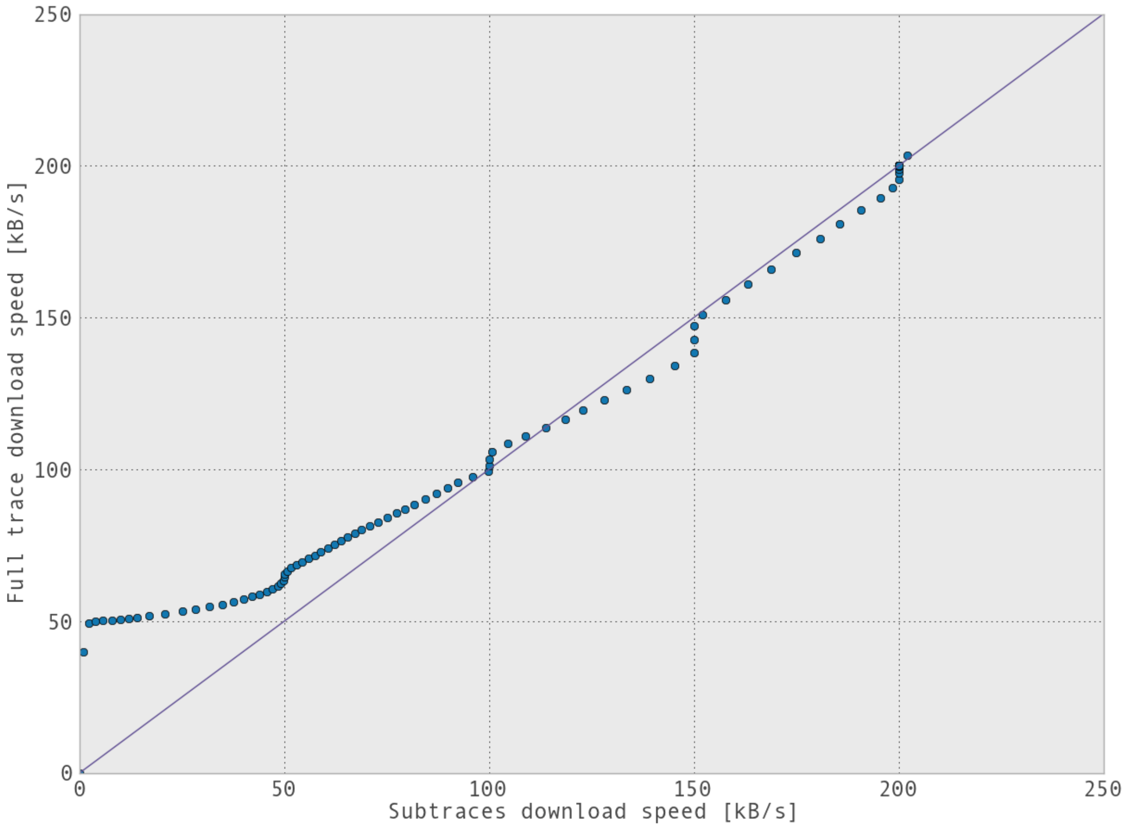

Figure 3.1: Comparison of the distributions of download speeds between 100 runs of the BitTorrent simulation with subtraces of 100 agents versus one simulation driven by the full trace. Both simulations are run with half of the population constituted by freeriders.

many users share to many other users (many-to-many). We have a integer multiples of the maximum upload speed, derived mostly from one users to many others (one to many) of a few users (some to many). We consider nevertheless that the general distribution is similar enough for our purposes.

3.2.2 The user decision model

Modelling the decision process of BitTorrent users is non-trivial. Different users certainly have different processes. One can think in terms of sharing ratio and an other in seeding time[7]. Also, an occasional user do not have the same expectations from the system than a heavy user. The technical sophistication of users is also important. For instance, not all users know what a sharing ratio is, and that their BitTorrent client has an option to automatically seed until a default ratio is obtained. Only if he does, has he the possibility to always seed to a certain ratio.

In the model we propose, we decide to ignore most of these complex factors. We want to build a model that captures enough of the general behaviour of the system as a whole. We hope that the choices we make are sensible enough in order to capture the evolution of a real system. So, our aim is not to reproduce the detailed process of all the users.

An analysis of some communities shows that users with a medium use of the system (1–5 downloads a week) are responsible for more than half of the volume of downloaded data[8]. We suppose that these users are generally sophisticated enough to decide if they want to to be a sharer or freerider. Sharers stay 18 hours in the swarm after the download finished, freeriders use a hit and run strategy. We further suppose that the decision is taken on the basis of performance. In other words, the past speed of downloads determines the future behaviour. Finally, in BitTorrent systems users interact only rarely directly, we therefore assume that the performance of the system is the only input users rely upon.

The different assumptions we have made in the previous paragraph, orient us towards aspiration based decision models. Recently, one implementation of this model has been used by Roca and Helbing [5]. This decision model is part of a greater class of decisions, learning rules [4,8]. Users learn what they can expect for the system, we call this their aspiration. With time, this aspiration evolves. The learning process is parametrised by α such that:

$$A_i(t) = \alpha A_i(t-1) + (1-\alpha)\pi_i(t-1) \tag{3.1}$$

with $\pi_i(t)$ the performance of the system at time t. The parameter α is understood

[7]The sharing ratio is defined as the division of volume of uploaded data by the volume downloaded. The seeding time is the time a user stays connected (and shares) while his download is finished.

[8]This follows a study of FileList dataset we used for the analysis in section 3.3.1.

as a memory factor. The higher the parameter, the more important the past is to the expected performance.

Aspiration then drives the behaviour of users. They change their behaviour when they are dissatisfied. We define satisfaction the following way:

$$s_i(t) = \pi_i(t) - A_i(t) + \eta_i(t) \tag{3.2}$$

where $\eta_i(t)$ is a normally distributed noise factor. When the satisfaction is greater than 0, users keep their strategies. When the satisfaction is lower, the strategy is flipped with probability:

$$p_i = \tanh(-s_i)$$

This model is very simple and treats all users identically. We add a term in the expression of the satisfaction, the individual bias β_i. This term captures the variability in the preferences of the user. For instance, some users are more altruistic and share even with conditions lower then their aspiration. Other consider the costs of sharing as more important and prefer to leech. The revised equation 3.2 is then:

$$s_i(t) = \pi_i(t) - A_i(t) + \eta_i(t) + f(\beta_i, \tau_i) \tag{3.3}$$

Note that the bias function f depends on τ_i, the type of the user. For the case of $\beta_i > 0:$

$$f = \begin{cases} \beta_i & \tau_i \text{ is sharer} \\ 0 & \text{otherwise} \end{cases} \tag{3.4}$$

In the case where $\beta_i < 0$ we have:

$$f = \begin{cases} -\beta_i & \tau_i \text{ is freerider} \\ 0 & \text{otherwise} \end{cases} \tag{3.5}$$

3.2.3 Putting it all together

Many models of artificial societies rest on simplifying assumptions about both the game structure (considering exchanges similar to a public goods game) and on agent behaviour[9]. In the model of BitTorrent we propose, we keep simplifying assumptions mostly for the user (and therefore agent) behaviour. But the game is modelled with detail, agents play a *BitTorrent game*.

[9] See for instance [5,9,10] for research produced in the context of QLectives or also [11-13].

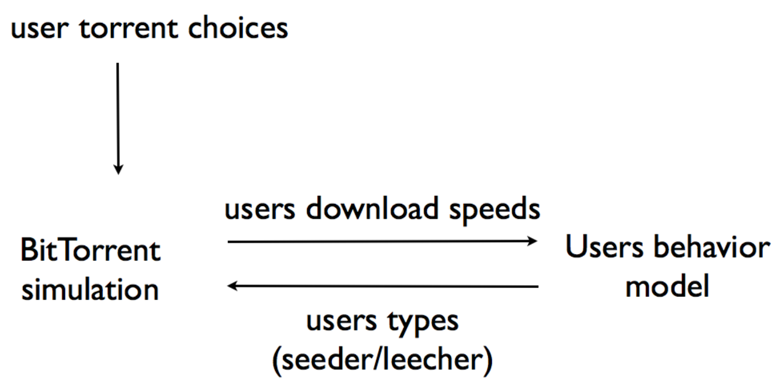

Figure 3.2: Representation of the full structure of the model.

As we have described in this section, the model is composed in part by a simulation of the BitTorrent protocol. This detailed simulation provides a certain performance (download speed) to users. They learn what to expect from the system and decide to keep their behaviour (sharer or freerider) according to their aspiration. The population of users, more precisely their types, informs (and therefore influences) the computation of the performance of the system.

The composition of the model, described in Figure 3.2.3, divides each turn in two steps. The first step corresponds to the model of *peers*. The simulation of BitTorrent is run, for a time corresponding to a week of data exchange between peers. Data extracted from the traces determines the torrent choices of all users. In the second step, we run the decision model of *users*. They are informed of their average performance during the week. Their decision provides the sharing behaviour to the peer simulation for the next turn.

3.3 Analysis of the model

This section focuses on the analysis of the result of the model. In a first section we discuss the features of the system we want to use as a comparison means. We will then use visual techniques to show that the model we proposed here does not reproduce the real world behaviour of BitTorrent systems.

3.3.1 Analysis of user behaviour

The data we use comes from the same trace that drives the torrent choices, the FileList trace [7]. This trace comes from information provided by peers to a central system. Some users may use modified software in which the peer lies. The volume of data shared might have been inflated in some cases. In consequence, we use a trace cleaned by the team in USZ in a collaboration with TUD [14].

The system of BitTorrent is complex enough to be characterised by a large set of quantities. We will limit ourselves to quantities that matter for our model. It has a twofold aim. First, it should captures most user sharing behaviour. Then, we want to be able to estimate the influence the introduction of a new protocol on the performance. We will discuss three quantities that characterize these aspects (or *features*) of the system:

- the *sharing ratio* is an indicator of the tendency of users to share
- the *time spend seeding* is also related to sharing behaviour
- the *download speed* corresponds to the performance of the system.

First, we focus on the sharing ratio of the users. It is defined as the total volume of data uploaded by a user divided by the volume he downloaded. Two different approaches can be taken. We could use ratios to distinguish between sharers and freeriders. If the distribution of ratios is bimodal, each mode corresponds to a type. We find the model parameters for which the share of seeders is identical to the measured one. This approach is not a good one. As shown on Figure 3.3, the distribution is not clearly bimodal. It looks like two convolved distributions. It is not possible to identify the two individual distributions without introducing hypotheses about the distribution of ratios within a type.

On the other hand, we need more than a single point to match the model. In order to have some confidence in the result of the calibration, we need to compare the distributions of ratios and not a single value, being the share of freeriders, the mean or the median. We can therefore see if the distribution of sharing ratios by user reproduces the distribution of Figure 3.3.

The seeding time corresponds to the length of time that a user stays in the system once his download is finished[10]. However, as presented in section 3.2.1, we use a simplified model of users' behaviours that rests on a fixed seeding time. The distribution of seeding time as a result of the model will consist of two peaks, one at 1 minute for freeriders and one at 18 hours for sharers. We therefore cannot use this value as a comparison quantity.

Finally, the average download speed of any user represents the performance he experiences. As the protocol does not distinguish between sharers and freeriders,

[10]We can obtain the volume of uploaded data by multiplying the seeding time by the average upload speed. This concept is therefore related to the sharing ratio.

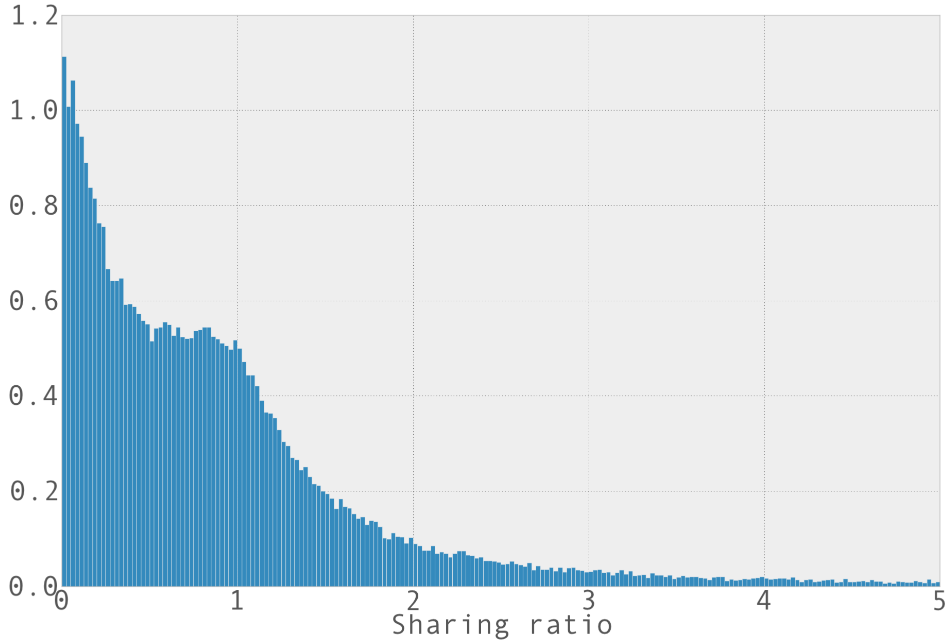

Figure 3.3: Normalised distribution of the sharing ratios of users in the FileList trace. The total surface, the sum of width times height of the rectangles of the histogram amounts to 1 (as the width is smaller than 1, the height can be higher).

the shape of the distribution should not present a convolution of distributions as in Figure 3.3. The distribution in Figure 3.4 follows the single peak behaviour we expect. Note that the distribution has speeds over 200kB/s. In the model we limit the download speed to this value. We can expect some inconsistencies for high average download speeds.

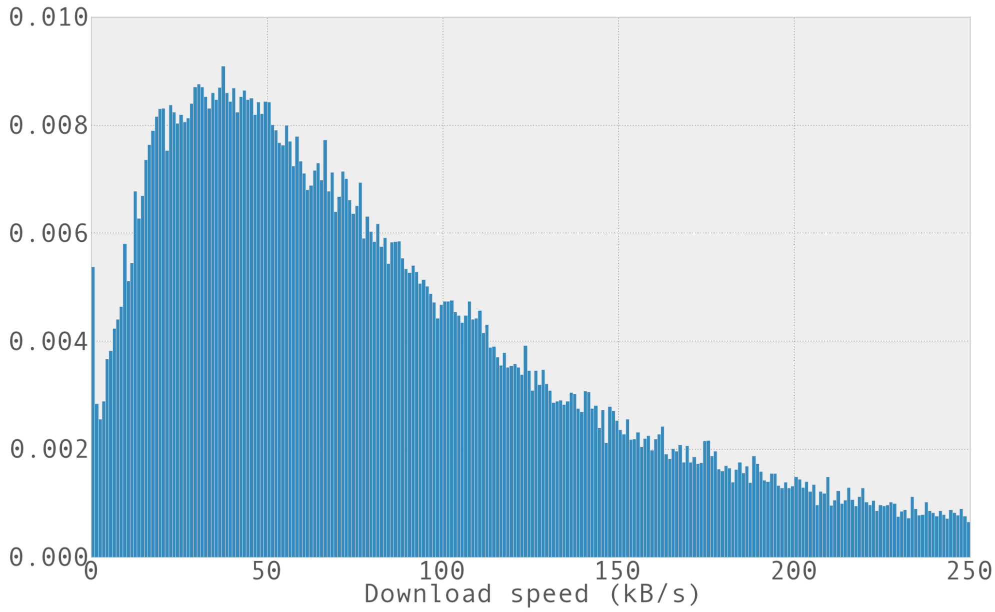

Figure 3.4: Normalised distribution of the average download speed of users in the FileList trace. The total surface, the sum of width times height of the rectangles of the histogram amounts to 1 (as the width is smaller than 1, the height can be higher).

3.3.2 Results

Unfortunately, the model we built does not reproduce actual systems. Actually it fails in two different ways. First, as we can see on Figure 3.6 and Figure 3.5 the parameters of the model do not play an important part in its behaviour. Secondly, and most importantly, Figure 3.7 shows that the distribution we are interested in is not captured by the model.

Figure 3.5 consists of Q-Q plots of distributions of the sharing ratios of agents with different parameters. The model in the ordinate always has parameters $\alpha = 0.1$ $bias = -0.9$, while the values in the abscissa are explained in the title. On one hand, we see that the points in top left graph are exactly on the 45 degree line, as expected (we compare the model with itself). More surprisingly, variation in the α parameter does not have an important influence on the sharing ratios at all. We see that for higher values of the parameter, agents tend to freeride slightly more. On the other hand, the sign of the bias does not have any influence on the distribution of ratios. The introduction of any bias changes the distribution, making it broader.

The analysis of the distributions of download speeds is simpler. As shown on Figure 3.6, they are not influenced by the parameters at all. We computed then in the same manner as the sharing ratios.

The Figure 3.7 is very explicit about the fact that our model does not reproduce the behaviour of the real system. In Q-Q plots between the distributions obtained from the model and the measured trace, a perfect match would mean the points should be on the line. We see a completely different graph, which indicates a falsification of the model.

The negative results can be interpreted as resulting from a decision model that is perhaps too sensitive, and therefore generates noise. A hint in that direction is provided by the evolution of the share of freeriders in the population. It does not converge in time, but the values have a nearly normal distribution around 0.5. The basic BitTorrent system does not provide a stable performance. Two different downloads can obtain different speeds just based on chance. The aspiration based model interprets these variations as more than randomness. Therefore most user behaviour changes are based on the noise generated by the BitTorrent system.

Also, the choice of strategies offered to the user might be too narrow. As we discussed earlier, in real systems not all users think in terms of seeding time. By forcing only two types (freerider or sharer) we introduce the big restriction that users seed for either 1 minute or 18 hours and no other values between these two. This might be too rough for the model to evolve towards a realistic behaviour.

In any case, the model is too flawed to be saved by some small, non-conceptual modifications. We need a complete rethink of the users' behaviour as well as their decision model. There are several possibilities. We could:

- add new parameters on which users base their sharing behaviour. Analysis of traces established that heavy users of the system share more than casual users. The rate of utilization of the system (the number of downloads each week) might be introduced in the decision process[11].
- change the aspiration mechanism to make it less sensitive to small, random variations of the performance.
- use continuous strategies, where users are not only sharers or freeriders but share more or less on a predefined scale.
- introduce a *fair* type of user who tries to share as much as he downloads (keep a sharing ratio close to 1).

[11] In our model, this value is driven by data and not modeled.

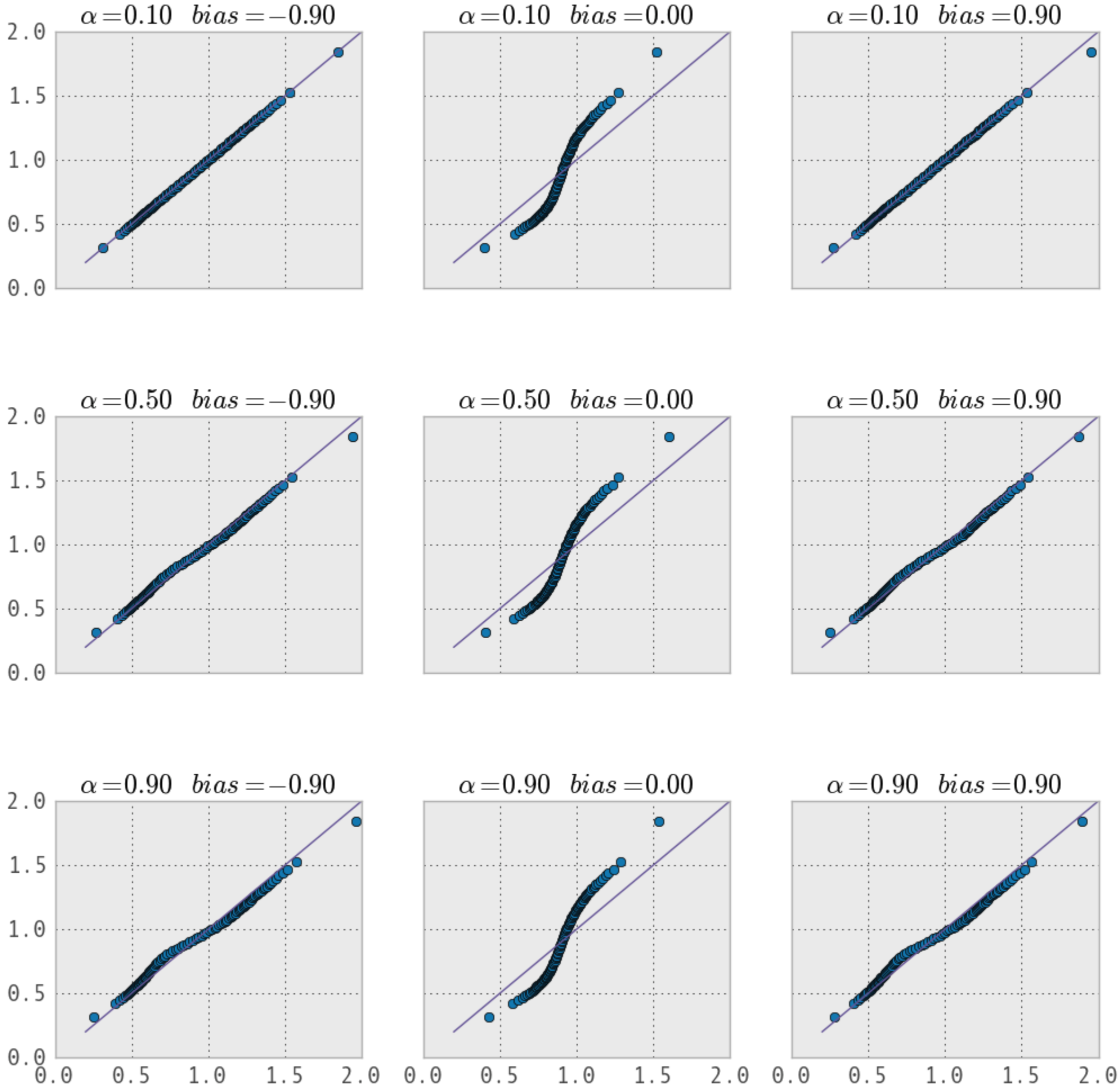

Figure 3.5: Comparison of the distributions of sharing ratios for the simulation with a wide range of parameters. All the distributions are compared with the first distribution ($\alpha = 0.1$ $bias = -0.9$). Each plot consists in the distribution of the agents of 100 runs with 100 agents.

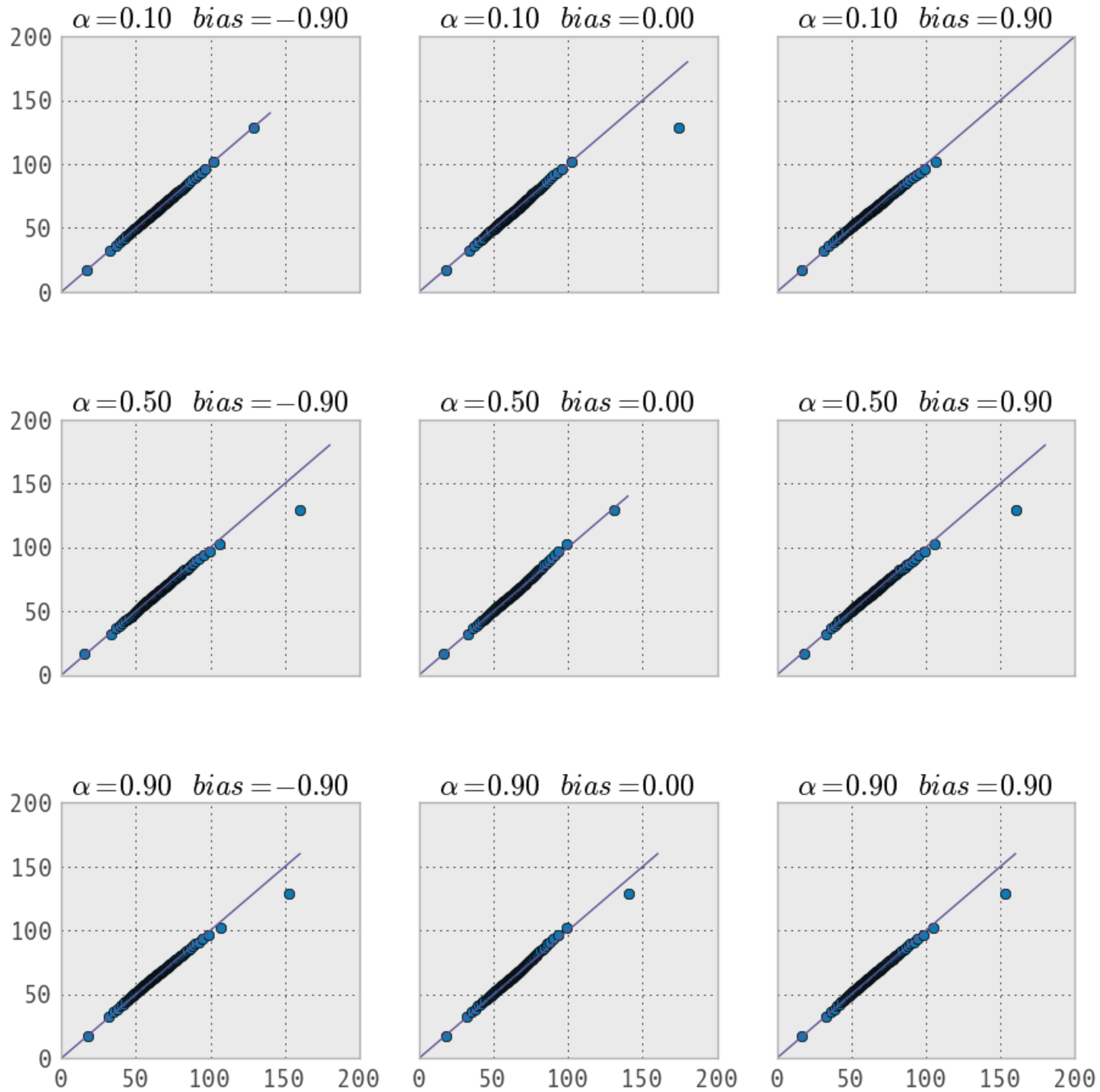

Figure 3.6: Comparison of the distributions of download speeds (in kB/s) for the simulation with a wide range of parameters. All the distributions are compared with the first distribution ($\alpha = 0.1$ $bias = -0.9$). Each plot consists in the distribution of the agents of 100 runs with 100 agents.

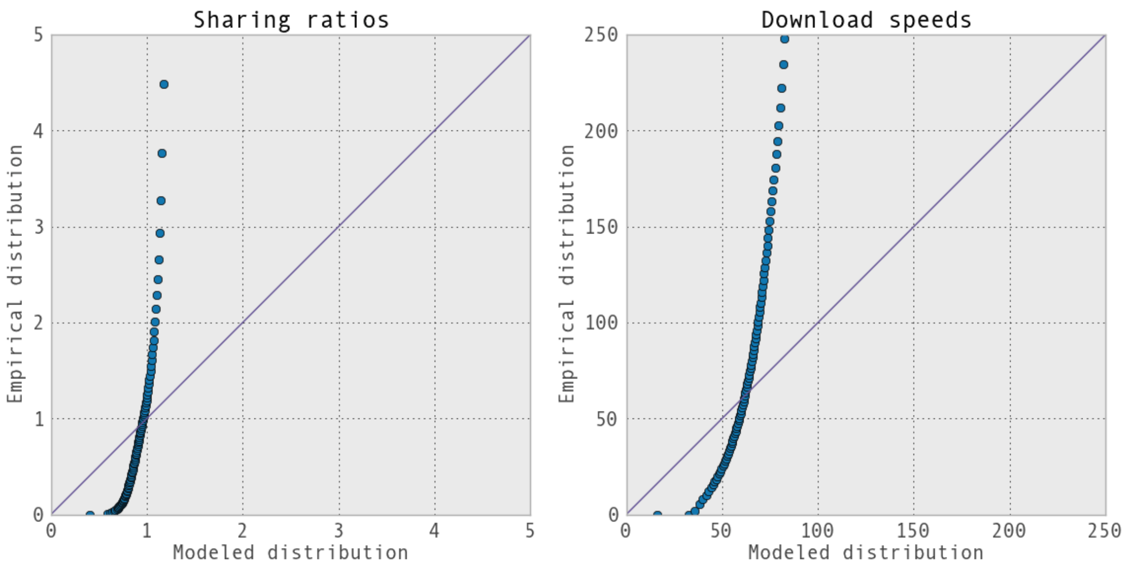

Figure 3.7: Comparison of the distribution of the important features between measured data (from the Filelist trace) and 100 runs of the model with 100 agents ($\alpha = 0.5\ bias = 0$). The features are computed for each agent over the whole period (4 months in the measured data and about 1 year in the simulated data).

Chapter 4

Further work

In reality BitTorrent users' behaviours are not stable. Some users stop using the system while other increase their consumption. The two models we presented in this deliverable aimed for modeling how an evolving behaviour can introduce dynamical effects in the system. Nevertheless none of them provided a model that used these effects and reproduced well reality.

The intuitions we had about the decision process of BitTorrent users and, more precisely, the assumptions we made are insufficient. They do not allow us to capture enough of the reality to model, and even less predict the behaviour of the BitTorrent system. Our next step will be to gain insight in the system from data. This analysis of user behaviour will be presented in another work package (deliverable 3.3.1). Then, these new insights should allow us to reformulate our hypotheses and build a better model.

Once we find a model that reproduces measurements well, we can implement the next step in the QLectives loop. As we sketched earlier in this document, we will introduce a reputation system in the BitTorrent protocol. If the QMedia community becomes important enough[1], Bartercast could be implemented. We could then estimate the predictive power of the method presented here in the case of the BitTorrent protocol.

[1] The Bartercast protocol is a reputation system. As such it needs to be run on an large enough population to work. If the population is small the number of exchanges between users of the reputation system will be low. As a consequence, the reputation mechanism will not have any influence on the performance of users.

Chapter 5

Bibliography

[1] M. Meulpolder, J. Pouwelse, and D. Epema, Parallel and Distributed Systems Report Series (2008).

[2] M. Meulpolder, J. A. Pouwelse, D. H. J. Epema, and H. J. Sips, in *Parallel & Distributed Processing, 2009. IPDPS 2009. IEEE International Symposium on* (IEEE, 2009), pp. 1-8.

[3] M. Meulpolder, J. a Pouwelse, D. H. J. Epema, and H. J. Sips, 2009 IEEE Ninth International Conference on Peer-to-Peer Computing 232-241 (2009).

[4] M. W. Macy and A. Flache, Proceedings of the National Academy of Sciences of the United States of America **99 Suppl 3**, 7229-36 (2002).

[5] C. P. Roca and D. Helbing, Proceedings of the National Academy of Sciences **108**, 11370-11374 (2011).

[6] D. Qiu and R. Srikant, Proceedings of the 2004 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications - SIGCOMM '04 367 (2004).

[7] J. Roozenburg, *Secure Decentralized Swarm Discovery in Tribler* (2006).

[8] A. E. Roth and I. Erev, Games and Economic Behavior **8**, 164-212 (1995).

[9] D. Helbing, A. Szolnoki, M. Perc, and G. Szab, New Journal of Physics **12**, 083005 (2010).

[10] D. Helbing and S. Lozano, Physical Review E **81**, 2-5 (2010).

[11] M. Milinski, D. Semmann, and H.-J. Krambeck, Nature **415**, 424-6 (2002).

[12] K. Sigmund, H. De Silva, A. Traulsen, and C. Hauert, Nature **466**, 861-3 (2010).

[13] M. A. Nowak and K. Sigmund, Nature **393**, 573-577 (1998).

[14] K. Csernai, M. Jelasity, J. Pouwelse, and T. Vinko, (2011).