5-13-2011

Enabling Peer-to-Peer Swarming for Multi-Commodity Dissemination

Daniel Sadoc Menasche
University of Massachusetts Amherst, sadoc@cas.umass.edu

ENABLING PEER-TO-PEER SWARMING FOR MULTI-COMMODITY DISSEMINATION

A Dissertation Presented

by

DANIEL SADOC MENASCHÉ

Submitted to the Graduate School of the
University of Massachusetts Amherst in partial fulfillment
of the requirements for the degree of

DOCTOR OF PHILOSOPHY

May 2011

Computer Science

ENABLING PEER-TO-PEER SWARMING FOR MULTI-COMMODITY DISSEMINATION

A Dissertation Presented

by

DANIEL SADOC MENASCHÉ

Approved as to style and content by:

Donald F. Towsley, Chair

Edmundo A. de Souza e Silva, Member

Weibo Gong, Member

James F. Kurose, Member

Laurent Massoulié, Member

Arun Venkataramani, Member

Andrew G. Barto, Department Chair
Computer Science

I dedicate this thesis to my parents, sister, family, professors and friends, and to everybody that contributed directly or indirectly to make this work concrete.

ACKNOWLEDGMENTS

Everyday I remind myself that my inner and outer life
are based on the labors of other men, living and dead,
and that I must exert myself in order to give in the same
measure as I have received and am still receiving.

Albert Einstein

This work is fruit of many collaborations. I am extremely grateful to my advisor, Distinguished Professor Don Towsley. His ideas, support, dedication and patience during the last four years are really inspiring to me. Many of the lessons that I learned from him, about how to tackle the right problems as well as on how to meticulously handle details, will stay with me for the rest of my life, and his influence in this thesis is pervasive.

Chapter 2 is the result of joint work with Professors Antonio Rocha, Don Towsley and Arun Venkataramani and Dr. Bin Li. I am very thankful to Professor Arun Venkataramani for his support and mentorship during my PhD years. Among many other things, I learned a lot from him about how to devise experiments, how to write clearly and how to present ideas. This work was very collaborative, and I cannot overstate how important and enriching it was to interact with Prof. Antonio Rocha, who executed most of the controlled PlanetLab experiments, and Dr. Bin Li, who collected measurements in the real BitTorrent network.

Chapter 3 is based on collaboration with Professors Antonio Rocha, Edmundo de Souza e Silva, Rosa Maria Leao, Don Towsley and Arun Venkataramani. Professors

Edmundo de Souza e Silva and Rosa M. Leao were my former advisors at UFRJ, and it was a huge pleasure to have a chance to work with them during the PhD thesis preparation. The model proposed in Chapter 4 was first presented by Professors Don Towsley, Edmundo de Souza e Silva and Rosa M. Leao. Professor Arun Venkataramani suggested the notion of *self-sustainability*. Prof. Antonio Rocha was responsible for the simulations.

Chapter 4 is based on joint work with Dr. Laurent Massoulié and Professor Don Towsley. I am extremely lucky for having a chance to work with Dr. Laurent Massoulié, who is always so generous with insights and dedication. Dr. Laurent Massoulié proposed the conjecture on the loss of efficiency due to direct reciprocity (whose proof in its more general setting we are still working on, jointly with Aaron Coté from UCLA) as well as the public board. This work was done in part while I was visiting Thomson Labs in the summer of 2007, and I am very grateful for their financial support during that period.

Chapter 5 is based on joint work with Dr. Giovanni Neglia and Professors Don Towsley and Shlomo Zilberstein. In particular, Dr. Giovanni Neglia proposed the existence and uniqueness of an equilibrium between publishers and users, that we are currently adapting to more general settings. This work was done in part while Dr. Giovanni Neglia was visiting UMass, and in part while I visited INRIA. I am very thankful to Dr. Giovanni Neglia for sponsoring my stay at INRIA.

I would also like to thank the members of my thesis committee, Dr. Laurent Massoulié, and Professors Don Towsley, Edmundo de Souza e Silva, Arun Venkataramani, Weibo Gong and Jim Kurose, for their time and suggestions. This work was sponsored in part by NSF under award numbers CNS-0519922 and CNS-0721779 and by a scholarship from CAPES/Fulbright (Brazil).

Finally, I am very grateful to my parents. The emphasis of my parents on the importance of learning, and the help of my sister with writing skills, provided me

strong foundations to grow. As invaluable is the attention and warmness of my friends, who make it much easier for me to have pleasure and enjoy everything I do.

ABSTRACT

ENABLING PEER-TO-PEER SWARMING FOR MULTI-COMMODITY DISSEMINATION

MAY 2011

DANIEL SADOC MENASCHÉ

B.Sc., FEDERAL UNIVERSITY OF RIO DE JANEIRO

M.Sc., FEDERAL UNIVERSITY OF RIO DE JANEIRO

M.Sc., UNIVERSITY OF MASSACHUSETTS AMHERST

Ph.D., UNIVERSITY OF MASSACHUSETTS AMHERST

Directed by: Professor Donald F. Towsley

Peer-to-peer swarming, as used by BitTorrent, is one of the *de facto* solutions for content dissemination in today's Internet. By leveraging resources provided by users, peer-to-peer swarming is a simple, scalable and efficient mechanism for content distribution. Although peer-to-peer swarming has been widely studied for a decade, prior work has focused on the dissemination of one commodity (a single file). This thesis focuses on the multi-commodity case.

We have discovered through measurements that a vast number of publishers currently disseminate multiple files in a single swarm (bundle). The first contribution of this thesis is a model for content availability. We use the model to show that, when publishers are intermittent, bundling K files increases content availability exponentially as function of K. When there is a stable publisher, we consider content

availability among peers (excluding the publisher). Our second contribution is the estimate of the dependency of peers on the stable publisher, which is useful for provisioning purposes as well as in deciding how to bundle. To this goal, we propose a new metric, swarm self-sustainability, and present a model that yields swarm self-sustainability as a function of the file size, popularity and service capacity of peers. Then, we investigate reciprocity and the use of barter that occurs among peers. As our third contribution, we prove that the loss of efficiency due to the download of unrequested content to enforce direct reciprocity, as opposed to indirect reciprocity, is at most two in a class of networks without relays. Finally, we study algorithmic and economic problems faced by enterprises who leverage swarming systems and who control prices and bundling strategies. As our fourth contribution, we present two formulations of the optimal bundling problem, and prove that one is NP hard whereas the other is solvable by a greedy strategy. From an economic standpoint, we present conditions for the existence and uniqueness of an equilibrium between publishers and peers.

TABLE OF CONTENTS

LIST OF TABLES

Table **Page**

LIST OF FIGURES

Figure **Page**

TERMINOLOGY

swarm	set of users interested in a given content
publisher	user interested in disseminating a content
peer	user interested in download of a content
seed	state of a peer after completing its download and before leaving the system
leecher	state of a peer before completing its download
chunk or block	minimum file division
self-sustainability	fraction of time at which all chunks are available among peers
availability	fraction of time at which all chunks are available among peers and publishers
active download time	period at which content is actively downloaded
idle waiting	period at which peers wait for content to become available
download time	active download time plus idle waiting

CHAPTER 1

INTRODUCTION

1.1 Swarming Systems

How would you disseminate a collection (bundle) of files, if you needed to? Some people would probably either send copies of each of the files to their friends, or post them online in a website and announce its link. However, these strategies might be too costly, specially if the number of files and the demand are substantial – each file needs to be replicated multiple times by a single server in order to reach all of its requesters. In order to mitigate such costs, one option consists of splitting the files into pieces, offering different pieces to different requesters. Requesters then complete their downloads by *cooperating with each other* in order to gather all pieces. By leveraging resources provided by users, such as bandwidth and disk space, this solution, also known as *peer-to-peer swarming*, decreases costs, such as bandwidth and energy, for the server. It also builds robustness since users can complete their downloads even in the absence of the server. But what if many of the files in the collection are unpopular? Could it be the case that there are no opportunities for users to cooperate, since they are all interested in different content? In this case, is it still possible to take advantage of peer-to-peer swarming in order to mitigate costs and build robustness?

In this thesis, we show how peer-to-peer swarming can be leveraged for the dissemination of multiple files, with different sizes and popularities. We are particularly interested in understanding the impact of users who, possibly influenced by recommendation systems, download *unrequested content*, or content that they had not orig-

inally sought. Our contributions are broadly classified into two areas: (a) we present models which allow us to quantify the impacts of the download of unrequested content on metrics such as mean download time and content availability and (b) we investigate the reciprocity that occurs among peers, bounding the loss of efficiency due to the download of unrequested content for bartering purposes. Although we will focus on peer-to-peer swarming for the dissemination of stored content in a wired network, in Chapter 6 we will indicate how our results might be extended to peer-to-peer networks for live content dissemination [102] or to delay-tolerant peer-to-peer networks where contacts are influenced by the mobility of users [18].

1.2 Motivation

Peer-to-peer systems are one of the *de facto* solutions for content dissemination in today's Internet. Daily, millions of users utilize peer-to-peer systems to exchange files in the Internet, and the use of peer-to-peer systems by enterprises is steadily growing [4, 92, 94, 8, 88]. As mentioned in the previous section, peer-to-peer systems decrease costs to publishers and provide scalability and robustness by leveraging resources provided by clients. As demand for content increases, system capacity scales accordingly, as the clients collaborate with each other while downloading the desired content. As the demand for multimedia files and the sizes of these files increase, peer-to-peer systems become an important content dissemination solution for many content providers [4, 94].

Peer-to-peer swarming, as illustrated by BitTorrent [20], is a simple and popular peer-to-peer content delivery mechanism [46]. A swarm is a set of peers interested in the same content (file or bundle of files) that exchange blocks of the files among themselves. Peers periodically query a tracker to obtain a random subset of other peers in the swarm. The fact that, in existing swarming systems, trackers do not

exchange control information with each other allows any user to publish its files independently, at any time.

Prior work on peer-to-peer systems focused on the dissemination of one file (commodity). However, a publisher may be responsible for the dissemination of multiple files and users may be interested in multiple files as well. This gives rise to novel and important research questions, especially in light of the long tail phenomenon, empirically observed in some entertainment businesses [5].

1.2.1 The Long Tail

There is evidence suggesting that in today's entertainment market there is rising demand for obscure titles, also known as niche content [5]. According to Chris Andersen, the future of entertainment is in the millions of niche markets, each of which is possibly associated to a small demand, but together representing a significant fraction of the revenue of enterprises such as Rhapsody, Amazon.com and Netflix. In 2004, products not available in traditional stores were responsible for 22%, 57% and 20% of the sales of Rhapsody, Amazon.com and Netflix. In November of 2008, Page [67] reported that 20% of the revenue of Raphysody came from songs that are not in the top 52,000. This reduction in the blockbuster effect indicates that in the field of content delivery the enterprises that can "make everything available", at a low cost, can have a significant advantage.

Peer-to-peer swarming scales to tolerate massive flash crowds, but its applicability to serve less popular content is limited. That is because, for less popular content, there might not be opportunities for peers to collaborate in downloading content. In this thesis, we deal with strategies to cope with such a challenge.

Key principle: In a market where enterprises that can make "everything available, with small costs" thrive, customers seek niche content. The demand for each item might be small, but the aggregate demand can be significant. The use

of peer-to-peer swarming to disseminate such kind of content poses novel challenges, which can be in part addressed by letting users download unrequested content. In Chapter 2 we will show that a linear increase in the arrival rate of requesters for a content yields exponential gains in availability.

1.3 Swarming Systems Primer

A swarm is a set of peers concurrently sharing content of common interest. A content might be a file or a bundle of files that are distributed together. The content is divided into blocks that peers upload to and download from each other. Since there is no interaction between peers across swarms, each swarm can be studied separately.

BitTorrent is one of the most popular applications that uses peer-to-peer swarming for content dissemination, and we will use it to illustrate how swarming works. Unlike a traditional server-based system, BitTorrent includes a *tracker* that promotes the interaction of participating peers. The identities of the trackers are announced to peers in *torrent* files, which can be found and downloaded through search engines such as Torrent Finder [91]. Peers periodically query the tracker to obtain a random subset of other peers in the swarm in order to exchange (upload and download) blocks with them. Peers also discover new neighbors from other peers, in addition to the tracker, when the Peer Exchange (PEX) extension is enabled.

There are two kinds of peers in the system: *seeds* and *leechers*. Seeds are peers that have completed their downloads and are available to upload blocks. Leechers are peers that have not completed their downloads and are actively downloading (and uploading) blocks of the file. Thus, leechers turn into seeds upon completing their downloads. Leechers adopt a tit-for-tat incentive strategy while downloading the file, i.e., leechers preferentially upload content to other leechers that reciprocate likewise, and "choke" or ignore leechers that do not reciprocate.

As many leechers leave the system immediately after completing the download, content publishers often support a seed that we refer to as the *publisher*. A publisher is guaranteed to have all of the blocks constituting the file. In the rest of this thesis we assume that each swarm includes at least one (possibly intermittent) publisher.

BitTorrent peers adopt the rarest first policy to decide which blocks to request and download from their neighbors. According to the rarest first policy, a peer prioritizes the rarest blocks when selecting the ones to download next. We say that a peer is *interested* in another peer if the latter can provide blocks to the former. Since rarest first guarantees a high diversity of blocks in the system, any peer is almost always interested in any other peer, which in general yields high system performance [50].

Note, however, that in BitTorrent peers only have local information about the system. Hence, they can only implement a *local* rarest first policy. The intrinsic limit on the number of connections that a user can establish naturally provides each of them with only a myopic view of the system. Firewalls, NATs and other exogenous factors may also prevent users from establishing connections among themselves.

1.4 Objectives

The objectives of this thesis are to develop models, measurements, tools and algorithms to understand and improve peer-to-peer swarming with respect to the following criteria:

- **Bundling implications**: multiple files can be disseminated in a single swarm through bundles. Pure bundling occurs when a user can only download the entire bundle or nothing; mixed bundling occurs when users are offered a choice between downloading the entire bundle or separate parts of the bundle. One of our aims is to study the implications of bundling on content availability, when publishers are intermittent, and on efficiency, when publishers are intermittent or always available.

- **Content availability**: when publishers are intermittent, content can become unavailable. Our second goal is to quantify content availability as a function of the popularity and size of the contents (files or bundles), arrival rate of publishers and capacity of publishers and peers.

- **Efficiency in distributing content**: if publishers are always available, we evaluate the dependence of peers on publishers as a function of the popularity and size of the files. For this purpose, we propose a new metric, swarm self-sustainability, and show how bundling affects this metric.

- **Bartering implications**: the dissemination of multiple files in an integrated way yields new opportunities for users. Users can download content that they had not originally sought, for bartering purposes. Another aim of this thesis is to evaluate the implications of different forms of bartering and reciprocity that occur among peers. We also propose strategies for match-making between users, and to issue recommendations on content value for bartering.

1.5 Contributions

The main contributions of our research are the following:

- **Modeling content availability and bundling implications [60, 58]:** The first contribution of this thesis is a model of content availability and its usage to quantify the implications of bundling. We have discovered through measurements that a vast number of publishers currently disseminate multiple files in a single swarm (bundle). We use our model to show that, when publishers are intermittent, bundling K files increases content availability exponentially as function of K.

- **Estimating the self-sustainability of swarms [59, 61]:** Our second contribution is the development and analysis of a model to estimate the dependency

of peers on a stable publisher. This model is useful for provisioning purposes as well as in deciding how to bundle. We propose a new metric, swarm self-sustainability, which measures the fraction of time that all blocks of a file are available among peers (excluding the stable publisher). Our model yields swarm self-sustainability as a function of the file size, popularity and service capacity of peers.

- **Bounding the loss of efficiency due to direct reciprocity [57]:** We investigate reciprocity and the use of barter that occurs among peers. As our third contribution, we prove that the loss of efficiency due to direct reciprocity, as opposed to indirect reciprocity, is at most two in a class of networks without relays. Our simulations indicate simple strategies for content recommendation and match-making between users allowing most user to download the demanded content, incurring a system loss of efficiency smaller than two, in scenarios of practical interest.

- **Addressing algorithmic and economic issues faced by enterprise swarming systems [58]:** We study algorithmic and economic problems faced by enterprises who leverage swarming systems and who control prices and bundling strategies. As our fourth contribution, we present two formulations of the optimal bundling problem, and prove that one is NP hard whereas the other is solvable by a greedy strategy. From an economic standpoint, we present conditions for the existence and uniqueness of an equilibrium between publishers and peers.

1.6 Previous Work

In what follows, we present a broad overview of previous work related to the topics discussed in this thesis. More comprehensive bibliographic review is presented at each chapter.

Peer-to-Peer Modeling

This thesis proposes techniques, models and algorithms applicable to the analysis of peer-to-peer swarming for multi-commodity dissemination. Guo et al. [39] were the first to identify, through measurements, potential benefits of leveraging collaboration among multiple swarms. Related work [56, 101, 71] that considered transmission of multiple contents across users interested in different files did so either in a real time setting [56, 101] or focusing on aspects complementary to those presented in this thesis, such as reputation mechanisms [71].

Availability and Bundling

In this thesis we analyze bundling as a mechanism to allow users to share different files in a single swarm. Recently, bundling has been suggested as a way to efficiently serve unpopular content in peer-assisted content delivery systems [17] and automatic ways to bundle files have been proposed [42]. More generally, in the music industry there has been increased interest on the implications of bundling from an economic standpoint [77, 29].

Incentives

The literature on incentives in peer-to-peer systems focusing on the relation between incentives and freeriding [82], clustering [49] and system design [71] is vast. Our work is the first to compare the *fundamental* benefits and disadvantages of direct and indirect reciprocity for the dissemination of digital goods.

In the economics literature, direct reciprocity bartering schemes are considered precursors of more sophisticated economies involving money and indirect reciprocity. Arpejis et al. [6] argue that prices can also play an important role in peer-to-peer sys-

tems. The authors compare implicit prices in bilateral exchanges (direct reciprocity) with explicit prices in multilateral exchanges (indirect reciprocity) and propose a system that implements the latter. Our work, in contrast, suggests that in the context of peer-to-peer file sharing, direct reciprocity has its own advantages and might suffice to implement an efficient system. More generally, our work on direct and indirect reciprocity might be extensible to innovation networks [72] and to biological networks [66].

Self-Sustainability

Although we were the first to study self-sustainability, our model is similar in spirit to the one presented by Hajek and Zhu [40] which in turn is based on the one by Massoulié and Vojnovic [56]. However, whereas these previous works analyzed the infinite population case (see also [31]), we are also interested in the small population regime. For small populations, Markov Chain (MC) models have been proposed by Veciana et al. [95], providing insights on the performance of the system but not dealing with the problem of availability of blocks among peers.

1.7 Bibliographic Notes

Most of the content of this thesis appears in [60, 58, 57, 59, 61]. All the works are fruits of collaborations with Professor Don Towsley. [60] is joint work with Professors Arun Venkataramani and Antonio Rocha and Dr. Bin Li. [58] is joint work with Dr. Giovanni Neglia and Professor Shlomo Zilberstein. [57] is joint work with Dr. Laurent Massoulie. Finally, [59, 61] were conducted together with Professors Edmundo de Souza e Silva, Rosa M. Leao, Antonio Rocha and Arun Venkataramani. Chapter 5 is also partially based on results first presented in my synthesis project at the University of Massachusetts at Amherst, prepared under the supervision of Professors Don Towsley and Shlomo Zilberstein, and available upon request.

1.8 Availability of This Thesis

This thesis is available in eletronic format at

`http://www-net.cs.umass.edu/~sadoc/thesis`

Possible corrections and additional material will also be made available in the above URL.

1.9 Thesis Overview

This section concludes our introduction. The remainder of this thesis is organized as shown in Figure 1.1. In Chapter 2, we propose a model for content availability, and use it to evaluate the implications of bundling. In Chapter 3 we study the dependence of peers on a stable publisher, by evaluating the content availability among peers (excluding the stable publisher). In Chapters 4 and 5 we discuss economic aspects. Chapter 4 deals with reciprocity and bartering and Chapter 5 deals with competition and equilibrium. Finally, Chapter 6 concludes this dissertation, pointing out some directions of future work.

Note that as we progress from Chapter 2 to Chapter 5, the proposed models are more abstract and less *hands on*. Accordingly, the validation of the models is grounded qualitatively on real data and quantitatively on controlled experiments in Chapter 2, on simulations of BitTorrent in Chapter 3 and on simulations of a stylized peer-to-peer system in Chapter 4. In Chapter 5, insights on the proposed model are obtained through its numerical evaluation.

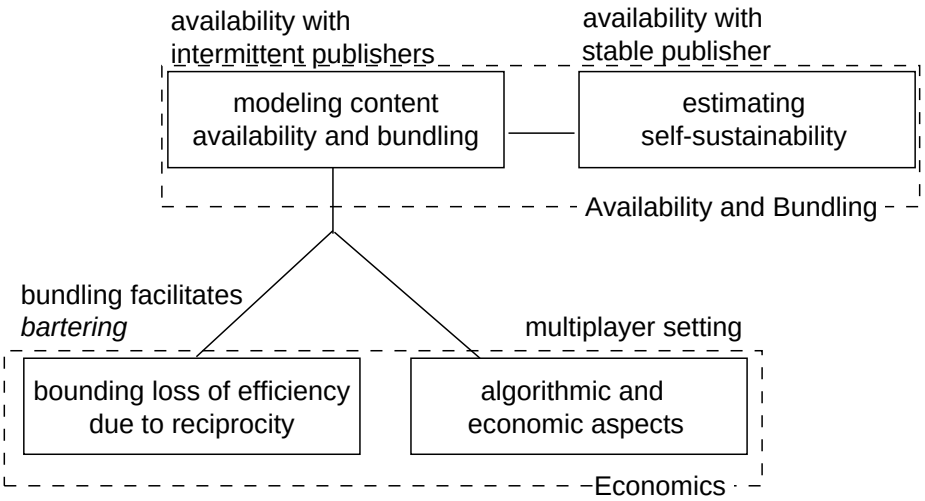


Figure 1.1. Structure of the thesis

CHAPTER 2

MODELING CONTENT AVAILABILITY AND IMPLICATIONS OF BUNDLING: THE INTERMITTENT PUBLISHERS CASE

This chapter and the next focus on the problem of content availability. In this chapter we consider a scenario with intermittent publishers, and present a model to quantify content availability and the implications of bundling. Using the model, we obtain bounds and approximations for content availability and average download times, as well as asymptotic results concerning the implications of bundling on these two metrics.

2.1 Introduction

In this chapter we present a mathematical model to study content availability and the implications of bundling in swarming systems such as BitTorrent. We use an extension of an M/G/∞ queue to model the self-scaling property of BitTorrent swarms, i.e., more peers bring in more capacity to the system. The key insight is to model uninterrupted intervals during which the content is available as *busy periods* of that queue. The busy period increases exponentially with the arrival rate of peers and publishers, and the time spent by peers and publishers in the swarm.

Our model enables us to analyze the impact of *bundling*, a common strategy adopted by BitTorrent publishers. The basic idea is that instead of disseminating individual files via isolated swarms, a publisher packages a number of related files into a bundle and disseminates it via a single larger swarm. Bundling is of interest because it leads to greater content availability. To appreciate why, consider a bundle

of K equal size, equally popular files. Equating the popularity of a swarm to the arrival rate of peers to that swarm, requests arrive to the bundle at a rate K times greater than for individual files. As we will observe our model implies that the busy period of the bundled swarm equals the busy period of the individual swarm times $e^{\Theta(K^2)}$, i.e., the busy period of the bundled swarm is exponentially larger than that of an individual swarm.

In some cases, the increased availability can reduce the average download time experienced by peers. The *download time* of a peer consists of a possibly non-zero *waiting time* spent while content is unavailable and a *service time* spent in actively downloading content. If the reduction in waiting time due to bundling is greater than the corresponding increase in service time, the download time decreases. We validate this conclusion in §2.4 through large-scale controlled experiments on PlanetLab using the Mainline BitTorrent client [50]. Our experiments also show that the conclusions of our model qualitatively hold even with more realistic request patterns, peer upload capacities, and heterogeneous popularities.

In summary, we make the following contributions in this chapter.

Measurement: We present a large-scale measurement study of real BitTorrent swarms that shows that (1) content availability is a serious problem due to publisher unavailability, (2) bundling of content is widely prevalent, and (3) bundled content is more available than unbundled content.

Availability model: We present a queuing-theoretic model to analyze content availability and download times in BitTorrent-like swarming systems. To our knowledge, this is the first model that relates content availability to arrivals and departures of peers as well as publishers.

Implications of bundling on performance: We use the model to analyze the implications of bundling, a widely prevalent yet little studied phenomenon, and show

that (1) bundling improves availability, and (2) bundling can reduce download times for unpopular content when publishers are highly unavailable.

Experimental validation: We validate the model using large-scale controlled experiments with the Mainline BitTorrent client [50] on PlanetLab showing that the model accurately predicts download times in swarms with intermittently available publishers for both bundled and individual content.

2.2 Measuring content availability and bundling in BitTorrent

In this section, we present a large-scale measurement study of BitTorrent that shows that 1) content unavailability is a serious problem in BitTorrent today, and 2) bundling of content is widely prevalent and that bundled content exhibits greater availability.

In the rest of this section, we classify users between *leechers* and *seeds* (see §1.3), since in our measurements we do not distinguish users who arrive to the network with content (publishers) from users that remain in the network after completing their downloads (seeds).

2.2.1 Content Availability in BitTorrent

We start with a definition of content availability. Recall that content is divided into blocks to be distributed among peers (refer to §1.3 for a primer on swarming systems).

Definition 2.2.1 (System Definition of Content Availability)**.** *Content is available if either at least one seed is present or leechers collectively have all blocks. Content availability is the fraction of time at which content is available.*

Seeds may become unavailable in practice due to several reasons. Publishers serving a large number of files may close down seeds after the initial popularity wave

subsides in order to reduce bandwidth costs. A seed may also be a user publishing home-generated content that cannot afford to stay online all the time. Seeds illegally uploading copyrighted material often disappear quickly for obvious reasons [26]. Even for legitimate content, maintaining highly available seeds entails administrative effort and cost, which runs counter to the goals of content seeds that value BitTorrent as a cheap alternative to a client-server approach, according to which clients download content exclusively from servers.

In this section we assume that content is available only if a seed is available, so as to tradeoff between the number of monitored swarms and the amount of processing per swarm. Kaune et al. [47] shows through measurements that, in the BitTorrent network, not more than 24% of peers can complete their downloads in the absence of seeds, which indicates that seeds have a significant impact on content availability. In the next section, we model content availability resulting both from seeds as well as from leechers alone.

2.2.2 Measuring unavailability

How available is content in BitTorrent swarms? To answer this question, we conducted a seven-month long measurement study of BitTorrent swarms as follows. We developed and deployed BitTorrent monitoring agents at 300 nodes on Planetlab from August 3, 2008 to March 6, 2009. Once every hour, a host at the University of Massachusetts Amherst receives an RSS feed advertised by GoogleReader of recently created torrent URLs from Mininova (a large torrent hosting site), and sends each URL to a subset of the monitoring agents on Planetlab. The agents join the swarms and begin to monitor their peers. Our agents leverage the Peer Exchange (PEX) protocol extension, which enables them to discover new neighbors from other peers in addition to the tracker. To avoid copyright issues, our agents collect only control

plane information without actually uploading or downloading content, which suffices for our purposes as in this section we equate content availability with seed availability.

To distinguish seeds from leechers, our agents record the bitmaps received from connected peers. The bitmaps are part of the BitTorrent protocol and a peer uses them to convey the blocks it possesses to its neighbors. Each entry in the trace collected by the agents consists of a swarm identifier, a peer identifier (IP address and port number) and its bitmap recorded roughly periodically for each discovered peer in the swarm. Our traces consist of more than 14 million distinct IP addresses and 66K distinct swarms.

Figure 2.1 shows the distribution of seed availability for the monitored swarms. The solid curve shows the availability in the first month after the creation of the swarm, when we expect the content to be the most popular. The extent of seed unavailability is severe: less than 35% of the swarms had at least one seed available all the time. The availability of seeds in the swarms over the entire duration of the measurements is even lower as shown by the dotted curve: almost 80% of the swarms are had no seeds 80% of the time.

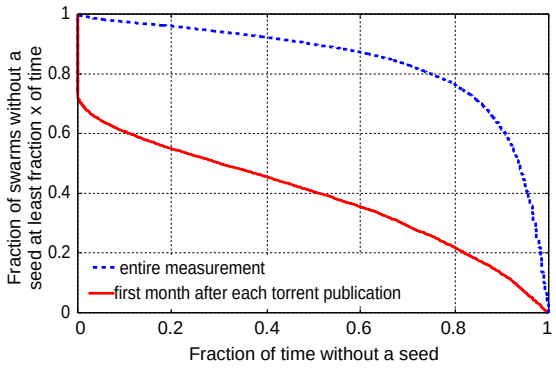


Figure 2.1. CDF of seed availability in 45,693 swarms each monitored for at least one month.

2.2.3 Content bundling

Bundling of content is a common practice in BitTorrent today. In this section, we study the extent of bundling and its impact on availability. The trace used in this section is a snapshot of BitTorrent swarms taken on May 6, 2009. For each of the 1,087,933 swarms in this snapshot, we record its content category (e.g., movies, TV, books etc.), names and sizes of constituent files, creation date, and instantaneous number of seeds and leechers. Note that we could afford to monitor many more swarms in this dataset than the previous one as we did not have to record details of peer arrivals and departures inside each swarm.

2.2.3.1 Extent of bundling

We analyze the extent of bundling in three of nine categories present in Mininova, namely, music, TV shows and books. These three categories together account for 45.98% of the swarms and 31.93% of the peers in the system. We chose these three categories because it is easier to automatically detect bundling by checking for the presence of multiple files with known extensions (e.g., `.mp3` for songs, `.mpg` for TV shows and `.pdf` for books). Detecting bundling is nontrivial in some categories, e.g., a DVD for a single movie is often organized as a collection of video files that are never distributed individually, making it difficult to check for the presence of multiple movies without manual inspection.

Among music swarms, albums are common. We classify a music swarm as a bundle if it has two or more files with common audio file extensions such as `.mp3`, `.mid` and `.wav`, which results in 193,491 of the 267,117 monitored swarms being classified as bundles.

Among TV show swarms, many bundles consist of sets of episodes in a season. We classify swarms that have two or more files with common video file extensions

such as `.mpg` and `.avi` as bundles, which results in 25,990 of the 164,930 monitored swarms being classified as bundles.

Among book swarms, we observe that collections, i.e., torrents containing the keyword "collection" in their titles, usually consist of a bundle of books connected by a broad theme, e.g., the "Ultimate Math Collection (1)" of size 5.81 GB has 642 books. We classified 841 of the 66,387 monitored swarms as collections. Classifying swarms that contain 2 or more files with common file extensions such as `.pdf` and `.djvu` as bundles results in an additional 6,270 bundles.

2.2.3.2 Bundled content is more available

In this section, we present evidence that suggests bundling is correlated with availability. We first consider book swarms. We find that 62% of all book swarms had no seed available on May 6, 2009, whereas that number drops to 36% if we consider only book collections.

We next analyze our traces more closely for content that is available both in isolation and as part of a larger bundle. We observe that among the unavailable collections, some of them were subsets of bigger collections, e.g., the 23 swarms consisting of collections of Garfield comics from 1978 to 2000 had no seeds. However, each of these collections can be found in a single super-collection aggregating all Garfield comics. The super-collection had seven seeds. After a manual inspection of all 841 book collections, we concluded that 210 had no seeds and were not subsets of other collections, which results in $210/841 = 25\%$ unavailability for content disseminated through collections (compared to 62% above for a typical swarm).

As another example, we consider swarms for the popular TV show "Friends". There was a total of 52 swarms associated with this show. Among them, 23 had one or more seeds available, and the remaining 29 had no seeds. The 23 available swarms consisted of 21 bundles (and 2 single episodes), whereas the 29 unavailable swarms

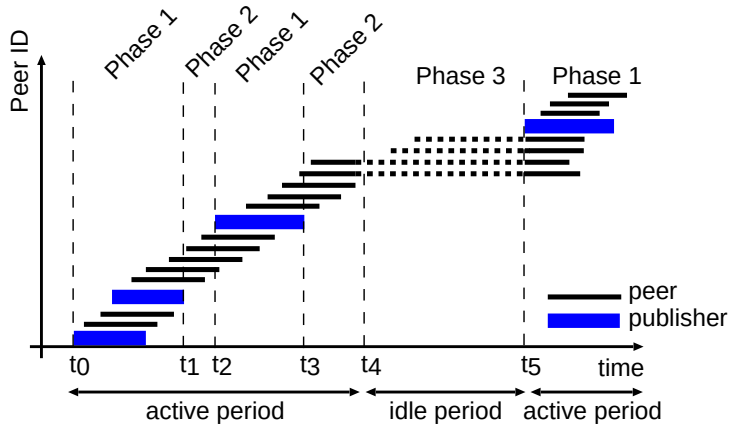


Figure 2.2. Illustration of the dynamics of a swarm: active and idle periods are represented by plain and dotted lines, respectively.

included only 7 bundles. These observations suggest a strong correlation between bundling and higher availability. The next section presents an analytic model that quantifies the causal relationship between the two.

2.3 Model

In this section, we develop a model for content availability in BitTorrent. The key insight underlying the model is to view BitTorrent as a *coverage process* or equivalently an extension of an M/G/∞ queuing system. The model shows that 1) bundling improves availability, and 2) for swarms with highly unavailable publishers, the availability benefit of bundling more than offsets the increased time to actively download more content, resulting in a net decrease in user-perceived download times.

2.3.1 Model overview

Requests for content arrive according to a Poisson process with rate λ. Publishers arrive according to a Poisson process with rate r. The content has size s, and is assumed to be infinitesimally divisible. We also assume that the service capacity of the system scales with the number of peers, the mean download rate of peers being μ.

We analyze content availability in two settings that differ according to whether peers are impatient or patient. Impatient peers leave the system immediately when content becomes unavailable. The residence time of patient peers, in contrast, comprises their active download times as well as their waiting times during periods in which content is unavailable. Unless otherwise stated, we assume that residence time of publishers and active file download times are exponentially distributed, with means $1/u$ and s/μ, respectively.

Figure 2.2 illustrates how content availability in BitTorrent depends upon the arrivals and departures of publishers and peers. Each horizontal line segment represents the time interval during which a peer (represented by a thin line) or a publisher (represented by a thick line) stays online. A swarm is initiated by the arrival of a publisher, which also marks the start of the first *active period*. The swarm's lifetime is divided into alternating active and *idle periods*. Content is available during active periods and unavailable during idle periods. If a publisher is always online, the first active period lasts forever and content remains always available.

So long as we have one or more publishers, or sufficient number of online peers, content is available, and downloads can progress. An active period ends when the following two conditions are satisfied: 1) there are no publishers online, and 2) the *coverage*, i.e., the number of peers currently online, drops to a fixed *threshold coverage* m (causing some blocks to become unavailable). By assuming a fixed threshold coverage, the active period approximately captures a contiguous period in which content is available. For example, Figure 2.2 shows that after all publishers leave at time t_1, the active period continues with the help of peers alone until a publisher reappears at time t_2. An active period may alternate any number of times between a phase consisting of one or more publishers (Phase 1) and a phase consisting of peers alone (Phase 2). Peers arriving during either phase in a active period find the content available. At t_4, there are no publishers and the number of peers drops to the coverage threshold

(assumed 2 in this example). This initiates an idle period that lasts until a publisher reappears at time t_5. Extant peers at the end of an active period as well as peers arriving during the idle period find the content unavailable (represented by dotted lines). Because of the time spent waiting, these peers experience longer *download times* defined as the times since a peer arrives until it completes the download.

Next, we introduce a definition of content availability, which allows us to build a tractable model for the system content availability (see Definition 2.2.1).

Definition 2.3.1 (Model Definition of Content Availability)**.** *Content availability is the fraction of time at which 1) there is at least one publisher online or 2) after all publishers leave the system, the population size contiguously remains above a fixed threshold m.*

Note that assuming that content is unavailable, i.e., there are missings blocks, when the number of peers in the network decreases to some threshold m is a simplification. In practice, content might become unavailable even when there are more than m peers in the system. Nevertheless, in §2.4 we will show the predictive power of our model in scenarios of practical interest, parameterizing the threshold coverage m using results to be presented in the upcoming chapter.

Our goal is to understand how content availability and the download times experienced by peers in a swarm depend upon 1) content popularity or the peer arrival rate λ; 2) the mean service time s/μ; and 3) the publisher arrival rate r and the mean time u that a publisher stays online. For simplicity, we also assume that peers are selfish and leave as soon as they complete their downloads; §2.3.3.4 extends the model to incorporate altruistic lingering.

To appreciate why bundling improves content availability, consider the special case of a highly unavailable publisher, i.e., its arrival rate r and mean residence time u are small. Then, the length of an active period is determined primarily by peers. Assuming a Poisson peer arrival process and a coverage threshold of one, the length of

an active period equals the busy period of an M/G/∞ queue, $(e^{\lambda s/\mu}-1)/\lambda$. Bundling K files increases the peer arrival rate for the bundle to $K\lambda$ as each peer desiring any of the constituent files requests the entire bundle, and increases the time spent by each peer in the swarm to Ks/μ. As a result, the length of the active period for the bundled swarm is $(e^{K^2\lambda s/\mu}-1)/(K\lambda)$. Let P be the unavailability of the individual swarm and $P^{(b)}$ the unavailability of the bundle. Then, $P^{(b)}/P = e^{\Theta(K^2)}$. For highly unavailable publishers, the availability gains of bundling can outweigh the cost of the increased time to download K times as much content resulting in a reduction in the download time, i.e., peers obtain more content in less time.

The rest of this section formalizes the above claims and derives closed-form expressions for the mean total download time experienced by peers with and without bundling.

2.3.2 Bounds on content availability and average download time

Next, we derive bounds for content availability and download time. To this goal, in §2.3.2.1 we equate content availability to seed availability, i.e., we set the threshold coverage to infinity, which yields a lower bound on content availability (see Definition 2.3.1) and an upper bound on average download time. In §2.3.2.2 we assume that the download of a content does not span more than one active period, i.e., we set the threshold coverage to zero, which yields an upper bound on content availability and a lower bound on download time.

2.3.2.1 Download time upper bound and content availability lower bound

Next we establish an upper bound on the download time and a lower bound on content availability. To this goal, we set the threshold coverage to infinity. Content is available if and only if there is at least one publisher online. A peer arriving during an idle period finds the content unavailable and immediately leaves, i.e., it does not queue up until a publisher arrives.

Variable	Description (units)
λ_k	peer arrival rate (1/s)
$\Lambda = \sum_{i=1}^{K} \lambda_k$	bundled peer arrival rate (1/s)
s_k	file size (bits)
$S = \sum_{i=1}^{K} s_k$	bundle size (bits)
μ	mean download rate of peers (bits/s)
r_k	arrival rate of publishers (1/s)
R	arrival rate of publishers for the bundle (1/s)
u_k	mean publisher residence time (s)
U	mean bundled publisher residence time (s)
Metric	**Description (units)**
P_k	unavailability
$P^{(b)}$	unavailability of bundle
T_k	download time (s)
$T^{(b)}$	bundle download time (s)

Table 2.1. Table of notation. Variables denoted by lower case characterize swarm $k \in \{1, 2, \cdots, K\}$ in isolation, while variables denoted by capital letters characterize the bundle of K files. Subscripts are dropped when homogeneous files are considered.

In swarm k ($k = 1, \ldots, K$), let r_k and u_k denote the arrival rate and mean residence time of publishers (refer to Table 2.1 for notation). Swarm k cycles through active and idle periods, with mean length $E[B_k]$ and $1/r_k$, respectively. The probability P_k that a peer arrives to swarm k and finds the content unavailable is

$$P_k = \frac{1/r_k}{E[B_k] + 1/r_k}, \qquad k = 1, \ldots, K \tag{2.1}$$

and

$$E[B_k] = \frac{e^{r_k u_k} - 1}{r_k} \tag{2.2}$$

The above follows from classical results for the busy period of an M/G/∞ queue [87].

Let R and U denote the arrival rate and residence time of publishers for the bundle, respectively. The probability $P^{(b)}$ that a peer arrives to find the content unavailable in the bundled swarm is

$$P^{(b)} = \frac{1/R}{E[B^{(b)}] + 1/R} \tag{2.3}$$

where the mean active period length for a bundle of K files is

$$E[B^{(b)}] = \frac{e^{RU} - 1}{R} \tag{2.4}$$

Replacing (2.2) into (2.1) and (2.4) into (2.3) yield a lower bounded on content availability for swarm k and for the bundled swarm, respectively.

To derive an upper bound on the download time, we use arguments similar to those in [24, §3] regarding birth-death systems with catastrophes. Let $\rho_k = r_k u_k$. Noting that birth, deaths and catastrophes in [24, Proposition 3.1] correspond to publisher arrivals, publisher departures and peer download completions in our setting, yields

Theorem 2.3.1. *The average download time for swarm k, $k = 1, \ldots, K$, is upper bounded by*

$$\frac{1}{\mu/s_k}\left(1 + \sum_{j=0}^{\infty} \frac{e^{-\rho_k}/\rho_k \sum_{i=j+1}^{\infty} \rho_k^i/i!}{e^{\rho_k} - (\mu/s_k)u_k \sum_{l=0}^{\infty} \rho_k^l/(l!(l + u_k\mu/s_k))}\right) \tag{2.5}$$

The content availability is lower bounded by $1 - e^{-r_k u_k}$.

The corresponding bounds for the bundled swarm are obtained by replacing s_k, r_k and u_k by $\max_k(s_k)$, R and U, respectively, in the expressions above.

The reader can find the derivation of Theorem 2.3.1, as well as of the other results that follow, in Appendix A.

2.3.2.2 Download time lower bound and content availability upper bound

Next, we aim at establishing a lower bound on the average download time and an upper bound on content availability. To this goal, we consider a threshold coverage equal to zero, i.e., a peer arriving during an active period always finishes the download during that active period and the last peer to finish ends the active period. In order

to obtain our bounds, we also assume that publishers can introduce the content in the system before leaving.

Let the aggregate arrival rate of peers and publishers to the individual swarm and to the bundle be $\lambda_k + r_k$ and $\Lambda + R$, respectively. For simplicity, we first consider the special scenario in which $u_k = s_k/\mu$ (we relax this assumption in the following section). Then

$$E[B_k] = \frac{e^{(r_k+\lambda_k)s_k/\mu} - 1}{r_k + \lambda_k}, \qquad k = 1, \ldots, K \tag{2.6}$$

and

$$E[B^{(b)}] = \frac{e^{(R+\Lambda)S/\mu} - 1}{\Lambda + R} \tag{2.7}$$

If, for all K files, $\lambda_k = \lambda$ and $s_k = s$ then $\Lambda = K\lambda$ and $S = Ks$. The bundled active period is $E[B^{(b)}] = e^{\Theta(K^2)}$. Thus, accounting for the availability gains brought by peers, the unavailability of the bundled swarm is $P^{(b)} = e^{-\Theta(K^2)}P_k$.

The next theorem uses the observations above to establish bounds on the average download time and content availability.

Theorem 2.3.2. *The average download time for swarm k, $k = 1, \ldots, K$, is lower bounded by*

$$(1 - \overline{A_k})/r_k + s_k/\mu \tag{2.8}$$

where

$$\overline{A_k} = \frac{E[B_k]}{1/r_k + E[B_k]} = \frac{(e^{(r_k+\lambda_k)(s_k/\mu)} - 1)/(r_k + \lambda_k)}{1/r_k + (e^{(r_k+\lambda_k)(s_k/\mu)} - 1)/(r_k + \lambda_k)} \tag{2.9}$$

The corresponding bounds for the bundled swarm are obtained by replacing s_k, r_k and λ_k by $\min_k(s_k)$, R and Λ, respectively, in (2.8) *and* (2.9).

2.3.3 A model for content availability and download time

Next, we quantify content availability and the mean download time experienced by peers when 1) the mean residence time of the publisher may differ from the service time of peers and 2) the coverage threshold may be finite and greater than zero.

We begin by presenting the theoretical background required by our model. Our results rely on the busy period analysis of the M/G/∞ queue [15], where the customer initiating the busy period has an exceptional residence time.

The workload consists of customers arriving according to a Poisson process with rate β. The residence time of the customer initiating a busy period is drawn from an exponential distribution with mean θ. The residence time of all other customers, X, takes the form of one of two exponentially distributed random variables, X_1 or X_2, corresponding to peers and publishers, with averages α_1 and α_2, respectively; $X = X_1$ with probability q_1 and $X = X_2$ with probability $q_2 = 1 - q_1$. The expected busy period is

$$E[B] = \theta + \sum_{i=1}^{\infty} \frac{\beta^i}{i!} \sum_{j=0}^{i} \binom{i}{j} \frac{q_1^j q_2^{i-j} \alpha_1^{1+j} \alpha_2^{1-j+i} \theta}{\alpha_1\alpha_2 + j\theta\alpha_2 + \theta\alpha_1 i - \theta\alpha_1 j} \tag{2.10}$$

In the rest of this section, unless otherwise stated, we assume that all files have the same size and request rates, and that the publisher arrival rates and residence times are the same across all swarms. Assuming homogeneous swarms allows us to drop the subscripts of variables referring to individual swarms. In Appendix A we show that most of our results extend to the case where different swarms have different file sizes and peer arrival rates.

2.3.3.1 Availability with impatient peers

We are now interested in determining the probability that a request leaves without being served. To this goal, we consider impatient peers that leave immediately the system if content is unavailable at their arrival.

Assumptions: Publishers arrive to individual swarms at rate r and stay in the system for a mean time u. For the bundled swarm, publishers arrive with rate R and stay for a mean time U. Peers that arrive during an idle period leave immediately

without being received service. The threshold coverage is assumed to be equal to zero, therefore active periods are referred to as busy periods.

Denote the probability that a request leaves without being served as P and $P^{(b)}$ for the individual and bundled systems, respectively. Then

$$P = \frac{1/r}{E[B] + 1/r} \qquad P^{(b)} = \frac{1/R}{E[B^{(b)}] + 1/R} \tag{2.11}$$

The average busy period for each individual swarm, $E[B]$, is obtained from (2.10) by setting the parameters as follows: $\beta = \lambda + r$, $\theta = u$, $\alpha_1 = s/\mu$, $q_1 = \lambda/(\lambda + r)$, $\alpha_2 = u$.

For the bundled swarm, the aggregate peer arrival rate is $\Lambda = K\lambda$ and the size is $S = Ks$. The average busy period, $E[B^{(b)}]$, is obtained from (2.10) as follows: $\beta = \Lambda + R$, $\theta = U$, $\alpha_1 = S/\mu$, $q_1 = \Lambda/(\Lambda + R)$, $\alpha_2 = U$.

The following lemma concerns the number of peers served in a busy period. Assuming that both the bundle publisher arrival rate, R, and mean publisher residence time, U, are functionally independent of K, we have

Lemma 2.3.1. *Bundling K files together yields a mean number of peers served in a busy period that is $E[N^{(b)}] = e^{\Theta(K^2)}$.*

Note that this result is qualitatively similar to the case when publishers and peers stay online for the same mean time (see paragraph following (2.7) in §2.3.2).

We now consider the scenario where peers have skewed preferences. Given K contents, let p_k denote the probability that a request is for content k, $k = 1, \ldots, K$. Assume that $p_k = c/k^\delta$, $\delta > 0$ (Zipf's law). Let Λ denote the aggregate peer arrival rate for all K swarms. Then, the arrival rate for swarm k is $\lambda_k = p_k\Lambda$. Under the assumption that the mean time to download the bundle scales as K/μ, one can show that the lemma above still holds (details in Appendix A).

In the theorem below we relate the asymptotics of the busy period to the probability that a request is not served. Under the same assumptions of Lemma 2.3.1 we have,

Theorem 2.3.3 (**Availability theorem).** *Bundling K files together yields unavailability $P^{(b)} = Pe^{-\Theta(K^2)}$.*

Although the above result assumes the publisher arrival rate for the bundle, R, is constant and independent of K, we show in Appendix A that even if $R = \Omega(e^{-cK^2})$, $c > 0$, the unavailability of the bundled swarm, $P^{(b)}$, is such that $P^{(b)} = Pe^{-\Theta(K^2)}$.

2.3.3.2 Mean download time with patient peers

We wish to compare the mean peer download time with and without bundling.

Assumptions: Peers that arrive during an idle period wait for a publisher to become available. The other assumptions are the same as in §2.3.3.1.

We first compute the average busy period length in an individual swarm, $E[B]$. When content is unavailable and a publisher arrives to start a busy period, the group of waiting peers immediately begins to be served. In the Appendix A we provide an expression for $E[B]$ accounting for the possible impact of this group of peers on the duration of the busy period. We also show that the theorem presented in the previous section holds under this setting. However, to simplify presentation, in what follows we neglect the possible impact of this group of peers on the duration of the busy period.

Approximation 2.3.1. *The impact of the group of waiting peers on the duration of the busy period is negligible.*

The above approximation is reasonable, for instance, if publishers reside in the swarm long enough to mitigate such impact. In this case, the average busy period

$E[B]$ is obtained from (2.10) by setting $\beta = \lambda + r$, $\alpha_1 = s/\mu$, $q_1 = \lambda/(\lambda + r)$, $\alpha_2 = \theta = u$.

The mean download time of a file when peers are patient, $E[T]$, is constituted by the mean active download time and mean idle waiting,

$$E[T] \approx \frac{s}{\mu} + \frac{1}{r}P \tag{2.12}$$

where $P = 1/(1 + rE[B])$.

For the bundled swarm, the mean busy period length, $E[B^{(b)}]$, can be obtained from (2.10) by setting $\beta = \Lambda + R$, $\alpha_1 = S/\mu$, $q_1 = \Lambda/(\Lambda + R)$, $\alpha_2 = \theta = U$. Once $E[B^{(b)}]$ is obtained, the mean download time for the bundle, $E[T^{(b)}]$, can be derived from (2.12) replacing s, r and $E[B]$ by their bundle counterparts S, R and $E[B^{(b)}]$. Bundling K files yields an averge download time of

$$E[T^{(b)}] \approx \frac{Ks}{\mu} + \frac{1}{R}P^{(b)} \tag{2.13}$$

where $P^{(b)} = 1/(1 + RE[B^{(b)}])$ and $E[B^{(b)}] = e^{\Theta(K^2)}$.

When service times dominate download times, bundling can increase the download time by up to a factor of K as peers download K times as much content. When wait times dominate download times, though, in Appendix A we show that after bundling $K = \sqrt{\log(1/R)}$ files the mean download time of the individual files, $E[T]$, is such that $E[T] = E[T^{(b)}]\Theta(R^{-c}K^{-b})$ $(b \geq 0, c > 0)$. Note that the factor $R^{-c}K^{-b}$ grows unbounded as $R \to 0$. This means that, when wait times dominate download times as is the case with highly unavailable publishers, peers may experience arbitrarily smaller download times when downloading bundles.

2.3.3.3 Threshold coverage

If a peer leaves the system carrying the last copy of a block of a file, content may become unavailable even if the number of peers online, i.e., the coverage, is greater than one. Our aim now is to determine the availability and the mean download time experienced by peers in the general case where content becomes unavailable when no publisher is online and the coverage reaches a threshold m.

Assumptions: Same as those described in §2.3.1.

Let $B(n, m)$ be the expected length of a period that begins with n peers and ends as soon as the population size reaches m ($m < n$). In the queueing theory literature, such a period is known as a *congestion period* [79]. For all n, $B(n, 0)$ is given by

$$B(n,0) = \sum_{i=1}^{n} \frac{s}{i\mu} + \frac{s}{\mu} \sum_{i=1}^{\infty} \left(\frac{s\lambda}{\mu}\right)^{i} \frac{(n+i)! - n!i!}{i!(n+i)!i} \tag{2.14}$$

For $m < n$, $B(n, m)$ is obtained using the recursion $B(n, m) = B(n, 0) - B(m, 0)$.

We use the expression above to estimate the unavailability probability and the expected download time of peers in the scenario described in §2.3.1 and depicted in Figure 2.2. Let $B(m)$ be the mean congestion period starting when the system transitions to Phase 2. Assuming that at this point the system is in steady state,

$$B(m) = \sum_{i=0}^{\infty} \frac{e^{-\frac{\lambda s}{\mu}} \left(\frac{\lambda s}{\mu}\right)^{i}}{i!} B(i, m) \tag{2.15}$$

The following two approximations allow us to solve the proposed model.

Approximation 2.3.2. *The duration of the congestion period is modeled through its mean.*

Approximation 2.3.3. *Publishers stay long enough in the system so that, when Phase 2 begins, i.e., when the number of publishers goes from one to zero, the population of peers is in steady state.*

Modeling the duration of the congestion period through its mean and assuming the system in steady state when Phase 2 begins are clearly strong simplifications. Nevertheless, in §2.4.3.1 we show that the effect of such assumptions on mean download times is negligible in many interesting scenarios. Therefore, we proceed with our analysis under such assumptions.

Under such assumptions, the number of times that the system cycles through Phases 1 and 2 before transitioning to Phase 3 is described by a geometric random variable with mean $e^{rB(m)}$. It follows that, for a threshold coverage of m, the mean download time of a file when peers are patient is

$$E[T] = s/\mu + P/r, \qquad \text{where } P \approx \exp(-r(u + B(m))) \tag{2.16}$$

The corresponding expression for bundled swarms is obtained by replacing s, λ, r and u by their bundled counterparts, S, Λ, R and U. In particular, if $R = Kr$ and $U = Ku$ the availability theorem (Theorem 2.3.3) still holds. In §2.4.3.1, we validate the mean download time estimated using our model against experiments and give recommendations on how to set the threshold coverage m.

2.3.3.4 Altruistic lingering

Peers may remain online as seeds after completing their downloads, either because they are altruistic or because publishers provide them incentives to do so. We now aim to analyze the impact of altruistic lingering on the mean download time of peers.

Assumptions: Peers remain in the system for an average amount of time $1/\gamma$ after completing their downloads. The other assumptions are the same as in §2.3.3.3.

In the Appendix A we show how to parameterize a general version of equation (2.10) to derive the availability probability and the mean download time of peers

that stay online as seeds after completing their downloads. Furthermore, we show that the availability theorem still holds.

To illustrate the consequences of peers staying longer in the system, consider two swarms with file sizes s_1 and s_2 and popularities λ_1 and λ_2. Peers remain in the swarm after completing their downloads in individual swarms, but not in the bundled swarm. We consider the scenario in which the mean number of users in swarm 1 and the bundle are equal, $[s_1\lambda_1/\mu + \lambda_1/\gamma = (\lambda_1 + \lambda_2)(s_1 + s_2)/\mu]$.

Let the mean residence time be defined as the download time plus altruistic lingering time. The mean residence time for requestors of content 1 equals

$$\frac{s_1}{\mu} + \frac{1}{\gamma} = \frac{(\lambda_1 + \lambda_2)(s_1 + s_2)}{\mu\lambda_1} = \frac{s_1 + s_2}{\mu}\Big(1 + \frac{\lambda_2}{\lambda_1}\Big) \tag{2.17}$$

For the bundled swarm, the mean residence time of peers is given by $(s_1 + s_2)/\mu$.

Assume the swarm 1 content is small and unpopular while the swarm 2 content is large and popular, $s_1 \ll s_2$, $\lambda_1 \ll 1 \ll \lambda_2$. Since content 1 is very unpopular (peer interarrival time very large), high availability depends on peers staying for a long time in the system after concluding their downloads (in equation (2.17), $1 + \lambda_2/\lambda_1 \to \infty$ as $\lambda_1 \to 0$). If swarm 1 is bundled with swarm 2, on the other hand, the overhead incurred by the peers only interested in content 2 is marginal (since $s_1 \ll s_2$) but the gains for peers interested in content 1 is remarkable, since requestors for content 1 experience the same availability and performance as those requesting file 2.

2.3.4 When does bundling reduce download time?

In this section we use the proposed model to illustrate when bundling reduces mean download time. We numerically evaluate equations (2.12) and (2.10) using the parameters described in the legend of Figure 2.3. Figure 2.3 shows the expected download time as a function of the bundle size. For seven of the scenarios ($1/R \in [500, 1100]$), increasing K to its optimal value, $K = 3$, leads to a decrease in the

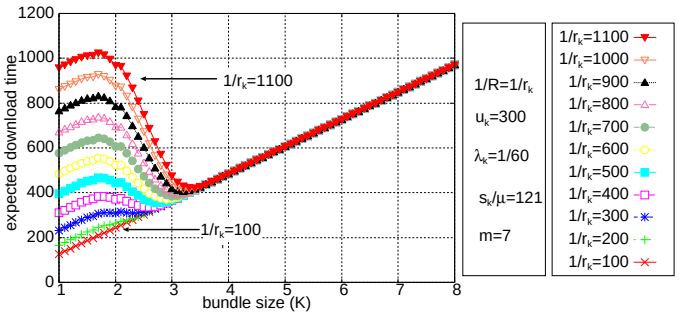


Figure 2.3. Bundles may reduce download time.

expected download time, while setting $K = 1$ is the best strategy for the remaining four. In each curve, as K increases the mean download time first increases, then decreases and finally increases again. The initial performance degradation occurs because small bundles may increase service times without sufficiently increasing the busy period. Figure 2.3 also shows that the benefits of bundling increase as the value of R decreases.

2.4 Experimental evaluation

In this section, we report on controlled experiments using real BitTorrent clients to validate the two main conclusions of our study: 1) bundling improves availability, and 2) bundling reduces download times when publishers are highly unavailable. We use an instrumented version of the Mainline BitTorrent client [50] and experiment with private torrents deployed on Planetlab. Our experimental setup thus emulates realistic wide-area network conditions, client implementation artifacts, and the impact of realistic upload capacity distributions and arrival patterns that are difficult to capture in an analytic model.

2.4.1 Experimental setup

Our experiments were conducted using approximately 150 Planetlab hosts and two hosts at the University of Massachusetts at Amherst one of which is designated as the controller of the experiment and another as a Bittorrent tracker. The controller

generates peer arrivals, publisher arrivals, and publisher departures by dispatching via `ssh` a command to start or stop BitTorrent clients on randomly chosen unused Planetlab hosts. At the end of the experiment, the controller collects the remote traces logged by the instrumented BitTorrent clients. We use the traces to determine when each download started and completed.

Experimental parameters Our experiments consist of torrents that publish either a single file of size $S = 4$ MB or a bundle of K files of aggregate size KS. The peer arrival rate for a bundle is assumed to be the sum of the arrival rates of its constituent files. The uplink capacity of each peer is $\mu = 33$ KBps ($\mu = 50$ KBps in §2.4.3). The publisher's upload capacity is 50KBps for individual as well as bundled torrents. There is one publisher that alternates between being on and off. The peer arrival rate λ and on/off behavior of the publisher (R and U) are varied according to the experimental goals as described below.

Note that, whereas in our model we assume that the file is infinitely divisible, in our experiments we consider files divided into blocks of 256KB. Therefore, when publishers are intermittent the system is not guaranteed to be stable for all range of parameters, i.e., for large periods of time the population may grow unbounded. In particular, the stability condition in [40], which states that the average capacity of the publisher (measured in blocks per second) is larger than the average arrival rate of peers λ (measured in peers per second) is not satisfied for $K \geq 6$ in Figure 2.6 ([40] is further discussed in §4.5). Nonetheless, throughout our experiments with intermittent publishers the number of peers in the system oscillated around its mean, i.e., the steady state system stability does not affect our results for the parameters and time spans considered below.

2.4.2 Bundling improves availability

Our model suggests that bundling increases availability by increasing busy period lengths and thereby reducing the need for a stable publisher. As an extreme case, we consider a publisher that initiates a swarm and then goes offline never to come back. We examine how long the swarm remains available after the publisher goes offline. We ensure that the publisher stays online long enough for at least one peer to fully download the file. Each peer leaves the system immediately after downloading the file.

We set $\lambda = 1/150$ peers/second for each file and all other parameters to their default values, and study how the availability of the publisher-less swarm varies with the level of bundling K.

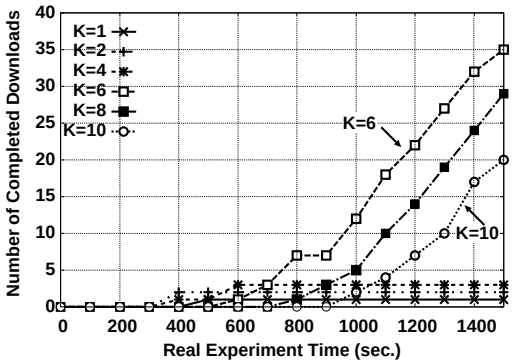


Figure 2.4. Availability of seedless swarms and the tradeoff in the choice of the bundle size.

Figure 2.4 shows the number of peers served between 0 and 1500 seconds of the experiment for $K = 1, 2, 4, 6, 8$ and 10. No peer completes its download in the first 300 seconds of the experiment: the publisher is either waiting for the first peer to arrive or is serving the first peer in each case. However, when the first peer completes its download and the publisher goes offline, the curves for $K = 1, 2, 4$ exhibit a very different trend compared to $K = 6, 8, 10$. For $K = 1, 2, 4$, only a small number of additional peers are able to complete their download before parts of the content

become unavailable. On the other hand, for $K = 6, 8, 10$, the number of completed downloads increases linearly even in the abscence of a publisher, i.e., the swarm is self-sustaining (self-sustainability is formally defined and further discussed in the next chapter).

At steady state, the length of time the swarm remains self-sustaining after the publisher goes offline is given, by definition, by the mean congestion period, $B(m)$. To compute $B(m)$ we use eq. (2.15) with $\mu = 33$KBps, $s = 4$MB and $\lambda = 1/150$ peers/s.

To set the threshold coverage we use results derived in the next chapter, on the mean number of peers necessary to attain a self-sustainability of 0.9 (see Figure 3.5 and Appendix A.3). For K varying between 1 and 8, the threshold coverage m is (10, 11, 12, 13, 13, 13, 14, 14). This leads to the following values of $B(m)$ for $K = 1$ to 8, (0, 0, 6, 115, 1110, 3796, 8484, 21264). These values capture the fact that for $K \geq 6$ the swarms remained self-sustaining throughout our measurement.

Although the system goes from being unavailable to being available as K increases from 4 to 6, further increasing K only increases download times. The mean peer download time when $K = 10$ is roughly 66% larger than that for $K = 6$ (not shown in Figure 2.4). This suggests a delicate tradeoff in choosing K—it should be large enough to bridge gaps in publisher unavailability, but beyond that point bundling only increases download times. We study this tradeoff in more detail next.

2.4.3 Bundling can improve download time

In this section, we consider an intermittently available publisher with capacity 100KBps that alternately remains on and off. The duration of on and off intervals are drawn from exponential distributions, with means of 300s and 900s respectively. The peer arrival rate for each file is $\lambda = 1/60$ peers/second and the capacity of each

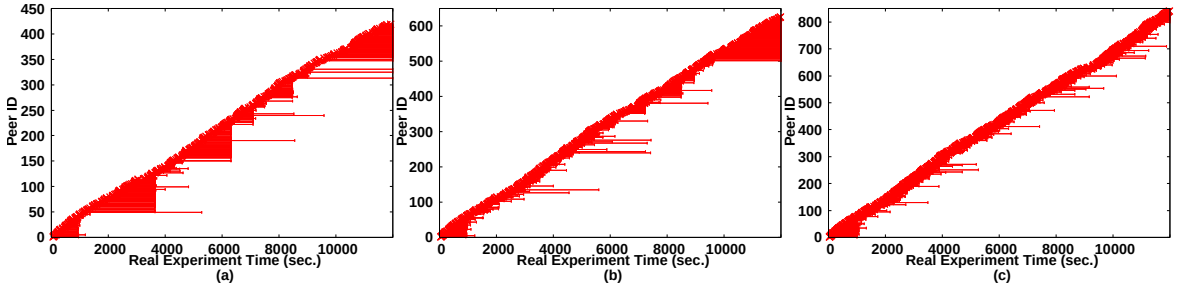


Figure 2.5. An intermittent publisher: (a) K=2; (b) K=3; (c) K=4. Each line starts when a peer arrives and ends when it leaves. As K increases, blocking probability decreases.

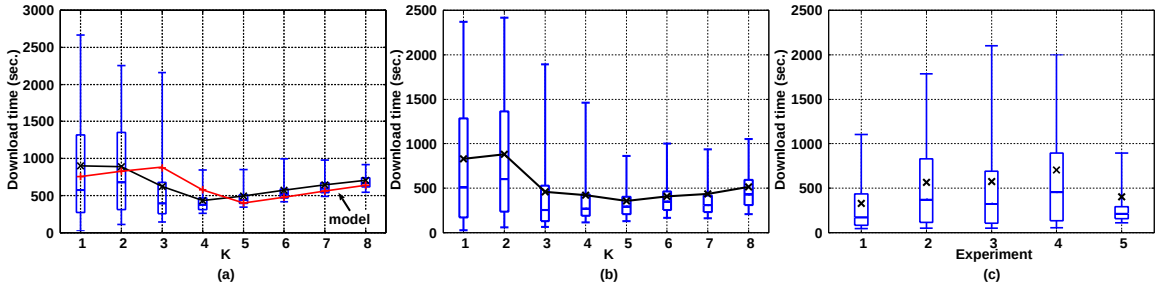


Figure 2.6. Download time versus bundling strategy. (a) exponential up and down times; (b) heterogeneous upload rates; (c) heterogeneous demand ($\lambda_i = \frac{1}{8i}$, $i = 1, \ldots, 4$), files bundled in experiment 5.

peer is μ=50 KBps. We study how the mean peer download time varies with the level of bundling.

Figures 2.5(a)–(c) show peer arrivals and departures over time. Each line segment starts at the instant that the peer arrives and terminates when the peer departs. For each value of K the experiment lasts for 10 runs of 1200s each. Figure 2.5(a) shows that for $K = 2$, many peers complete their downloads at roughly the same time. These bulk departures indicate that the swarm is not self-sustaining. They occur because extant as well as newly arriving peers get stuck soon after the publisher goes off, and must wait until the publisher reappears to serve the missing blocks allowing them to complete their downloads. On the other hand, setting K=3 (Figure 2.5(b)) reduces the likelihood of peers being blocked, and setting $K = 4$ (Figure 2.5(c))

nearly eliminates blocking as the swarm sustains itself during periods of publisher unavailability.

Figure 2.6(a) shows the mean download time as a function of K. For $K = 1$ and 2, the mean download time remains large as it is dominated by the times peers spend waiting for the publisher. When $K = 3$, the mean download time reduces significantly, however the distance between the 5th and 95th percentiles remains large as the download times are still partly determined by peers waiting for the publisher to reappear. The optimal bundle size is $K = 4$. The mean and the median download time are the lowest for this value of K as bundling eliminates gaps in publisher availability. For the same reason, the distance between the 5th and 95th percentiles is smaller when $K = 4$ than when $K \leq 3$. For values of $K > 4$ the download time increases linearly with respect to K as the download time is dominated by the time to actively download increasingly bigger bundles.

2.4.3.1 Evaluation of the analytical model

Next, we validate our analytical model (Section 2.3.3.3) against the experimental results above. We compute the mean download time adapting (2.16) to account for the fact that there is only one publisher in the system to obtain

$$E[T^{(b)}] = Ks/\mu + P^{(b)}/R, \text{where } P^{(b)} = \frac{\exp\left(-R\sum_{i=0}^{\infty}\frac{\exp(-\frac{K^2\lambda s}{\mu})(\frac{K^2\lambda s}{\mu})^i}{i!}B^{(b)}(i,m)\right)}{UR+1} \tag{2.18}$$

The derivation of (2.18) is in Appendix A. Setting $s/\mu = 80$s, $\lambda = 1/60$ peers/s, $1/r = 900$ arrivals/s, $u = 300$s and setting the threshold coverage as described in §2.4.2, our model predicts the results observed in Figure 2.6(a) well. The model leads to an optimal bundle size of $K = 5$, whereas the optimal observed in the experiments was $K = 4$. The model correctly captures the trend of the download time curve.

2.4.3.2 Heterogeneous upload rates

Next, we repeat the above experiment with heterogeneous peer upload capacities. The upload rate distribution was taken from the measured data used to generate Figure 1 in the BitTyrant study [70]. The average upload rate is 280KBps and the median is 50KBps. Using realistic peer upload capacities does not qualitatively change the behavior of the system (compare Figures 2.6(a) and Figures 2.6(b)). However, the optimal bundle size is now $K = 5$. This is consistent with the increase in the average upload capacity compared to the values obtained from the experiments with homogeneous capacities (μ=50 KBps). The larger upload capacity implies that a larger bundle is needed to increase the length of its busy periods so as to make the swarm self-sustaining during periods of publisher unavailability—a conclusion that agrees with our model.

2.4.3.3 Heterogeneous file popularites

Next, we study the impact of bundling when different files have different popularities. We consider a bundle of four files. We assume that the popularities of the files inside the bundle are distributed as follows: $\lambda_1 = 1/8$, $\lambda_2 = 1/16$, $\lambda_3 = 1/24$ and $\lambda_4 = 1/32$. We run 5 experiments, the first four correspond to swarms with individual files (experiments 1, 2, 3 and 4) and the last one to a bundle including all four files (experiment 5). In experiment i ($1 \leq i \leq 4$) we set λ_i as described above, and in experiment 5 we set $\lambda = \sum_{i=1}^{4} \lambda_i = 1/3.84$. All other parameters are set to their default values.

The mean download times are illustrated in Figure 2.6(c). The boxplots and lines show the distribution quartiles and 5th and 95th percentiles. For the individual files, as we move to the right in Figure 2.6(c) (i.e., as the popularity of the files decreases) the mean download time increases. When we consider a bundle of four files (experiment 5, extreme right in 2.6(c)) the mean download time is 405s, and the

distance between the 5th and 95th percentiles is smaller than the one observed for any of the individual files. The mean download time of the bundle is larger than the mean download time of 329s experienced for file 1 in isolation but smaller than the mean download times for files 2, 3 and 4 in isolation. These results are explained as follows. File 1 is the most popular, so the cost of downloading more content outweighs the availability benefit of bundling. However, for the less popular files 2, 3, and 4, bundling reduces the download time by keeping the swarm self-sustaining during periods of publisher unavailability. In summary, if contents have different popularities, bundling may increase the download times of peers downloading the most popular contents but can benefit those downloading unpopular files. In this example, bundling slightly increases the download times of 48% of peers who download the most popular content but significantly benefits the majority of the population.

2.4.3.4 Arrival patterns

Our model as well as experiments so far assume that peers arrive according to a time invariant Poisson process. To evaluate the sensitivity of our conclusions to the Poisson assumption, we repeated experiments similar to those in Figure 2.6 using scaled versions of real arrival patterns observed in our measurement traces collected in §2.2. We found that using trace-driven arrivals does not qualitatively change our conclusions (refer to Appendix A.3 for details).

However, we believe our model's conclusions may not hold if the mean arrival rate is not steady for a long enough duration of time. In particular, our model will overestimate the length of the busy period and consequently availability if the arrival rate decreases significantly before the end of the busy period determined by the current arrival rate. Nevertheless, we found a significant number of swarms with relatively steady arrival rates in our measurement traces. For example, out of the 1,155 swarms associated with the TV show "Lost", 911 were published more than

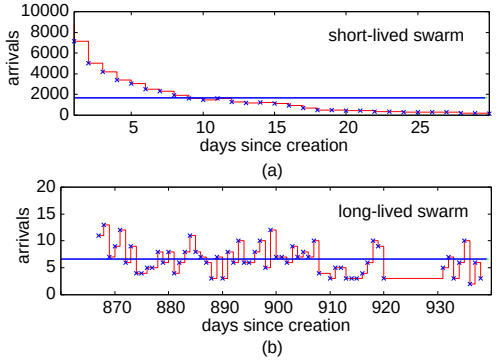


Figure 2.7. Typical peer arrival patterns of short-lived and long-lived swarms.

one month before we started our measurement. Figure 2.7(a) shows a typical new swarm in its first month and a typical swarm two years after its creation. The arrival rates of old swarms show much less variation compared to the arrival rates of new swarms. Our model can be used to predict the availability, download times, and the impact of bundling for such swarms.

2.5 Related work

To our knowledge, this chapter presents the first analytical model for content availability in BitTorrent-like swarming systems. In addition, we address the availability and performance implications of bundling, which received little attention in the realm of BitTorrent [89, 52, 41].

Ramachandran et al. [74] study the blocked leecher problem, where extant as well as arriving peers may have to wait for a long period of time for some blocks of the file that are no longer available. To address the problem, they propose BitStore, a token-based incentive architecture to obtain the missing blocks cached at other peers that had previously downloaded the file.

Qiu and Srikant [73] building upon earlier work by Veciana et al. [95] present a fluid model to analyze the download time performance of BitTorrent in steady-state. In contrast, our model accounts for both performance and availability similar in spirit to *performability* [43]. A naive adaptation of the fluid model in [73] to

bundles suggests strictly longer download times under bundling [89], whereas our model shows that bundling can decrease download times by improving availability, i.e., peers can download more content in less time. This idea was further pursued by Lev-tov et al. [52] and Carlsson et al. [17]. In these two works, the authors have proposed bandwidth allocation schemes, in the realm of unmanaged and managed swarms, respectively, such that if peers exchange content that they did not request, mean download times can decrease due to availability gains.

Hajek and Zhu [40, 108] studied the stability of peer-to-peer systems. They have shown that in a system in which peers adopt random peer selection and rarest first block selection, there is a non-negligible probability that one block of the file might become very rare, and in this case the server needs to send that block to every peer that arrives to the system. Although some experiments conducted in this chapter correspond to unstable systems, the steady state stability did not affect our results for the parameters and time spans considered (see §2.4.1 for details).

Many recent works have studied performance and fairness of a single swarm [50, 56, 12, 31]. Collaboration across swarms was studied by Guo et al. [39] suggesting many unexplored inter-torrent opportunities for block exchanges. Yang et al. [104] propose subtle modifications to the Mainline BitTorrent client so as to leverage such opportunities. Their mechanism provides incentives for peers to remain online as seeds after completing downloads. Piatek et al. [71] suggest that propagating peer reputations limited to one hop can incent exchanges across swarms. Sirivianos et al. [85] propose an architecture where a commercial content provider provides "credits" to incent more cooperation between peers. Bundling is complementary to inter-swarm collaboration based on micropayment schemes to improve content availability. Micropayment schemes require a central bank to enable transactions and a tracking mechanism across swarms for peers to locate each other. In contrast, bundles are

easy to set up and require no change to existing trackers or clients and is already in widespread use.

Our work opens up several avenues of future work. First, bundling may increase the traffic in the network, which motivates studying the implications of bundling for ISP pricing as well as its impact on content locality. Second, although our work sheds some light on what files make good candidates for bundling, more work is needed to understand how a content provider should optimally bundle files to meet performance or cost objectives, especially when the demand for a bundle may be different from the aggregate demand for its constituent files.

2.6 Discussion

In this section we discuss our key modeling assumptions as well as alternatives to bundling.

2.6.1 Assumptions and Their Validation

Next, we discuss the assumptions adopted to yield a tractable model.

Steady state assumption: Our model assumes a Poisson arrival of peers at a fixed rate. It has been shown in §2.4.3.4 that long-lived swarms have relatively stable mean arrival rates over periods of months. Our model can be used to analyze the impacts of bundling on content availability and mean download times for such swarms. More generally, note that our model qualitatively predicts one of the findings observed in the wild, namely that bundled content is more available. The quantitative validation of the model was done using controlled PlanetLab experiments (see §2.4).

Fixed threshold coverage: Our model assumes a fixed threshold coverage. In practice, content might become unavailable even when there are more than m peers in the system, i.e., the threshold coverage is a random variable. Nevertheless, in §2.4 we showed the predictive power of our model in scenarios of practical interest,

parameterizing the threshold coverage m using results to be presented in the upcoming chapter.

2.6.2 Alternatives to Bundling

Altruistic lingering and exogenous incentive mechanisms for peers to linger in the system are alternatives to bundling. Both altruistic lingering and exogenous incentives yield increased peer residence times, which increases content availability. However, as pointed out in §2.3.3.4, if small and unpopular files are bundled with large and popular ones, bundling leads to significant availability gains for the requesters of the unpopular files with negligible impact to the requesters of the popular ones. Such gains are due to the increased replication of the unpopular content, and might be difficult to attain through altruistic lingering alone (see §2.3.3.4).

2.7 Conclusions

Peer-to-peer swarming in BitTorrent scales impressively to tolerate massive flash crowds, but falls short on availability. For instance, our measurements indicate that half of the swarms are unavailable half of the time—an observation that does not bode well for the increasing commercial interest in integrating swarming with server-based content dissemination. Our work is a first step towards developing a foundational understanding of content availability in swarming systems.

By viewing BitTorrent as a queueing system, we were able to model content availability. The model suggests two important implications for bundling of content, a common practice among swarm publishers today. First, bundling improves content availability. Second, when the publisher is highly unavailable, bundling reduces the download time experienced by peers to obtain unpopular content. The latter implication is particularly intriguing as peers take less time to download more content. Although the model makes several simplifying assumptions, we were able to em-

pirically validate its conclusions through large-scale controlled experiments with the Mainline BitTorrent client over Planetlab.

CHAPTER 3

MODELING SWARM SELF-SUSTAINABILITY: THE STABLE PUBLISHERS CASE

By leveraging resources provided by clients, peer-to-peer swarming decreases costs to publishers, and provides scalability and system robustness. However, there is a limit on how much savings can be gained from swarming techniques. For example, in the case of unpopular content, peers must rely on the publisher in order to complete their downloads. In this chapter, we investigate such a dependence of peers on a publisher. To this goal, we define swarm self-sustainability as the fraction of time in which all blocks are available among peers (excluding the publisher).

Both this chapter and the previous one are focused on the problem of content availability. Whereas in the previous chapter we considered a scenario with intermittent publishers, in this chapter we consider a stable publisher and quantify swarm self-sustainability. In addition, whereas in the previous chapter we assumed that the file was inifinitesimally divisible, in this chapter we consider a file divided into finite size blocks.

3.1 Introduction

We consider a scenario where each swarm includes one stable publisher that is always online and ready to serve content. The corresponding system is henceforth referred to as a hybrid peer-to-peer system, since peers can always rely on the publisher if they cannot find blocks of the files among themselves. If all blocks are available among the peers, the swarm is referred to as *self-sustaining*. Quantifying swarm

self-sustainability, defined as the fraction of time during which the swarm can sustain itself without a publisher, is useful for provisioning purposes. The larger the swarm's self-sustainability, the less the dependency of peers on the publisher, and the lower the bandwidth needed by the publisher to serve the peers.

The primary contribution of this chapter is a model to study swarm self-sustainability. We use a two-layer model to quantify swarm self-sustainability as a function of the number of blocks in the file, the mean upload capacity of peers and the popularity of a file. The upper layer of our model captures how user dynamics evolve over time, while the lower layer captures the conditional probability of a given number of blocks being available among the peers given a fixed upper layer population state. Our model is flexible enough to account for large or small numbers of blocks in the file, heterogeneous download times for different blocks, and peers residing in the system after completing their downloads. We derive closed-form expressions for the distribution of the number of blocks available among the peers and apply them to show that self-sustainability increases as a function of the number of blocks in the file. The derived expressions involve sums and differences of large numbers, and are amenable to numerical errors. Hence, we present an efficient algorithm to compute the swarm self-sustainability that avoids these problems. We then numerically investigate the minimum popularity needed to attain a given self-sustainability level. Finally, we validate the estimates made by the model against detailed simulations.

The remainder of this chapter is organized as follows. In §3.2 we propose our model and in §3.3 we present an efficient algorithm to solve the proposed model followed by analytical results in §3.4. In §3.5 we evaluate our model against experiments. In §4.5 we present related work, in §3.7 we discuss some limitations and caveats of our model and §4.7 concludes.

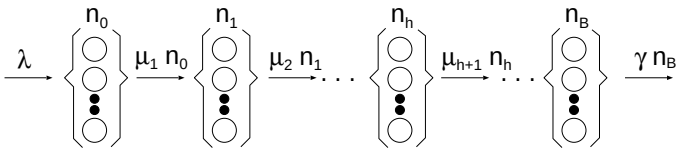


Figure 3.1. User dynamics. In stage h, there are n_h users, each user owning h blocks, $0 \leq h \leq B$.

3.2 Model

In this section, we present our model to estimate swarm self-sustainability. Our model is hierarchical. The upper layer characterizes user dynamics, and the lower layer comprises a performance model used to quantify the distribution of blocks available among the peers, for a given state of the upper layer model.[1] We present each of the layers, in §3.2.1 and §3.2.2, respectively, and then introduce the metric of interest in §3.2.3.

3.2.1 User Dynamics Model

A file consists of B blocks. Requests for a file arrive according to a Poisson process with rate λ. We further assume that the time required for a user to download its j^{th} block is a random variable with mean $1/\mu_j$, $1 \leq j \leq B$. After completing their downloads, peers remain in the system for a time with mean $1/\gamma$.

We model user dynamics with $(B+1)$ M/G/∞ queues in series. Each of the first B M/G/∞ queues models the download of a block, and capture the self-scaling property of BitTorrent swarms, i.e., each peer brings one unit of service capacity to the system. The last queue captures the residence time of seeds (see Figure 3.1).

The system state is characterized by a $(B+1)$-tuple, $\mathbf{n} = (n_0, n_1, n_2, \ldots, n_B)$, where n_h represents the number of customers in queue h, i.e., the number of users that

[1]Each layer of the model is self-containted. In Appendix B.11 we provide an alternate description of our model, integrating the user dynamics into the lower layer and explicitly capturing the evolution of the blocks owned by each user (users signatures) in time.

have downloaded h blocks of the file, $0 \leq h \leq B$. We denote by $\mathbf{N}$ the random variable characterizing the current state of the upper layer model and by $\mathbf{n}$ its realization. The number of peers in the system is referred to as n, $n = \sum_{i=0}^{B} n_i$.

Peers arrive according to a Poisson process with rate λ to queue 0 and transit from queue h, also referred to as stage h, to queue $h+1$ $(0 \leq h \leq B-1)$ with rate μ_{h+1}, the download rate of the $(h+1)^{th}$ block downloaded by a peer. The mean residence time in queue B captures the mean time that peers remain in the system after completing their downloads, $1/\gamma$. Setting $\gamma = \infty$ models the case where all peers immediately leave the system after completing the download. Throughout this chapter, unless otherwise stated, we assume that the mean download times of all blocks are the same, $1/\mu_j = 1/\mu$, $1 \leq j \leq B$. Nevertheless, all results are easily extended to the case where the mean time it takes for a user to download its j^{th} block is $1/\mu_j$.

Let $\pi(n_0, \ldots, n_B)$ be the joint steady state population probability distribution, $\pi(n_0, \ldots, n_B) = P(\mathbf{N} = (n_0, \ldots, n_B))$, of finding n_h users in the h^{th} queue, $0 \leq h \leq B$, and let $\pi_h(n_h) = P(N_h = n_h)$, $h = 0, \ldots, B$, be the corresponding marginal population probability distributions of the individual queues. The steady state distribution of the queueing system has the following product form,

$$\pi(n_0, \ldots, n_{B-1}, n_B) = \prod_{h=0}^{B} \pi_h(n_h) = \frac{(\lambda/\gamma)^{n_B}}{n_B!} e^{-\lambda/\gamma} \prod_{h=0}^{B-1} \left(\frac{\rho^{n_h}}{n_h!} e^{-\rho} \right) \tag{3.1}$$

where $\rho = \lambda/\mu$ is the *load* of the system (refer to Table 3.1 for notation).

3.2.2 Performance Model For a Given Population State

We now describe the lower layer of the model. Given the current population state, $\mathbf{n} = (n_0, \ldots, n_B)$, our goal is to determine the distribution of the number of blocks available among the peers. We begin by stating our key modeling assumption.

parameters	
λ	mean arrival rate of peers (peers/s)
$1/\mu$	mean time to download a block (s)
$\rho = \lambda/\mu$	mean load of the system (per stage)
B	number of blocks in file
$1/\gamma$	mean residence time of seeds
variables	
n_h	number of users that own h blocks
$\mathbf{n}$=$(n_0,\ldots,n_B)$	upper layer state
$\pi(\mathbf{n})$	steady state probability of state $\mathbf{n}$
$n = \sum_{i=0}^{B} n_i$	number of peers in the system
V	number of blocks available among the peers
metrics	
$p(v) = P(V = v)$	probability of v blocks being available among peers
A=$p(B)$	swarm self-sustainability

Table 3.1. Table of notation. Vectors are denoted by bold face symbols. Unless otherwise stated, $\gamma = \infty$ in which case $n_B = 0$. When referring to block availability, it is subsumed availability *among peers* (excluding publisher).

Uniform and independent block allocation: In steady state, the set of blocks owned by a randomly selected user in stage h is chosen uniformly at random among the $\binom{B}{h}$ possibilities and independently among users.

A user u in stage h, $0 \leq h \leq B$, has a signature $s_{h,u} \in \{0,1\}^B$, defined as a B bit vector where the i^{th} bit is set to 1 if the user has block i and 0 otherwise. Each user in stage h owns h blocks and has one of $\binom{B}{h}$ possible signatures.

Under the uniform and independent block allocation assumption, signatures are chosen uniformly at random and independently among users; the latter is clearly a strong assumption since in any peer-to-peer swarming system the signatures of users are correlated. Nevertheless, in §3.5 we show that the effect of such correlations on our metric of interest, swarm self-sustainability, is negligible in many interesting scenarios. Therefore, we proceed with our analysis under such an assumption.

Let $S_{h,u}$ be the random variable denoting the signature of the u^{th} user in stage h, and $s_{h,u}$ its realization, $1 \leq u \leq n$. The sample space of the lower layer model,

$\Omega_{\mathbf{n}}$, for a given state of the upper layer, $\mathbf{n}$, is the set of all $\{0,1\}^{Bn}$ bit vectors in which element $B(u-1)+i$ equals one if the u^{th} user has block i, and zero otherwise, $1 \leq u \leq n$, $1 \leq i \leq B$. An element in $\Omega_{\mathbf{n}}$ is the concatenation of n bit vectors of size B each. $\Omega_{\mathbf{n}}$ has cardinality $|\Omega_{\mathbf{n}}| = \prod_{h=0}^{B} \binom{B}{h}^{n_h}$. Then, under the uniform and independent block allocation assumption,

$$P(S_{1,1}{=}s_{1,1},\ldots,S_{B,n_B}{=}s_{B,n_B}|\mathbf{N}=\mathbf{n}){=}\frac{1}{|\Omega_{\mathbf{n}}|} \tag{3.2}$$

In the next section, we relate the upper and lower layer models, showing how (3.1) and (3.2) yield the key metric of interest, namely, swarm self-sustainability.

3.2.3 Self-Sustainability

We now define the key metric of interest, swarm self-sustainability. Let V denote the steady state number of blocks available among the peers. Denote by $p(v)$ the steady state probability that v blocks are available among the peers,

$$p(v) = P(V=v) = \sum_{\mathbf{n}\in\mathbb{N}^{B+1}} P(V=v|\mathbf{N}=\mathbf{n})\pi(\mathbf{n}) \tag{3.3}$$

Definition 3.2.1. *Swarm self-sustainability, A, is the steady-state probability that the peers have the entire file,*

$$A = p(B) \tag{3.4}$$

Definition 3.2.1 together with equation (3.3) yields,

$$A = \sum_{\mathbf{n}\in\mathbb{N}^{B+1}} P(V=B|\mathbf{N}=\mathbf{n})\pi(\mathbf{n}){=} \tag{3.5}$$

$$= \sum_{\mathbf{n}\in\mathbb{N}^{B+1}} P(V=B|\mathbf{N}=\mathbf{n})\frac{(\lambda/\gamma)^{n_B}}{n_B!}e^{-\lambda/\gamma}\prod_{h=0}^{B-1}\left(\frac{\rho^{n_h}}{n_h!}e^{-\rho}\right) \tag{3.6}$$

The second equality in (3.5) follows from (3.1). $P(V=B|\mathbf{N}=\mathbf{n})$ is obtained from (3.2) (see Appendix B.1).

If peers leave the system immediately after concluding their downloads, we refer to the swarm self-sustainability as A_∞. Swarm self-sustainability, A, when $\gamma < \infty$, is obtained from A_∞ as follows,

$$A = 1 - (1 - A_\infty)\exp(-\lambda/\gamma), \qquad \gamma < \infty \tag{3.7}$$

The above follows because a block is unavailable among the peers if it is unavailable among the leechers and there are no seeds in the system.

Note that A is expressed through (3.5) as an infinite sum. In what follows, we approximate A by its truncated version, $A^{(N)}$, considering only population states in which there are no more than N users in the system,

$$A^{(N)} = \sum_{\mathbf{n}\in\mathbb{N}^B s.t. n\leq N} P(V = B|\mathbf{N} = \mathbf{n})\pi(\mathbf{n}) \tag{3.8}$$

The value of N is based on the desired error tolerance η, and is chosen as described at the end of §3.3. A naive use of (3.8) yields an inefficient algorithm to compute $A^{(N)}$ by exploring a number of states that grows exponentially with respect to the file size. This problem is addressed in the next section, where we provide an efficient algorithm to evaluate $A^{(N)}$.

In the rest of this chapter, we refer to the truncated self-sustainability, $A^{(N)}$, simply as self-sustainability, the distinction between $A^{(N)}$ and A being clear from the context. In addition, since A is readily obtained from A_∞ using (3.7), henceforth we focus on the case $\gamma = \infty$ and make the dependence of $p(v)$ on γ explicit, denoting it by $p(v;\gamma)$, whenever $\gamma < \infty$.

3.3 An Efficient Solution Algorithm to Evaluate Self-Sustainability

In this section we present an efficient algorithm to compute swarm self-sustainability in polynomial time. The key insight consists of aggregating the states in the upper

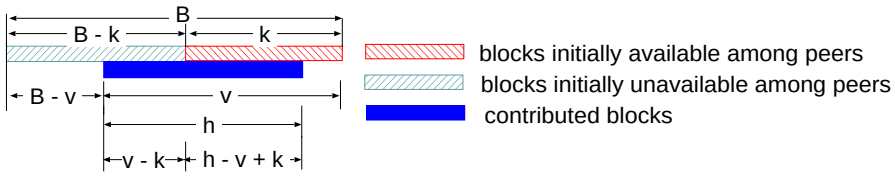


Figure 3.2. Recursion to compute probability of v blocks being unavailable among the peers. There are initially k blocks unavailable among the peers, and v after a user contributes h blocks.

layer of the model in such a way that the lower layer metrics are computed once per aggregate rather than once per state. The algorithm relies on three observations about our model, the first related to the performance model for a given population state (lower layer model) and the last two related to the user dynamics (upper layer model).

Let $\psi_h(k, v)$ be the probability that, in a system in which k blocks are initially available among peers, v blocks become available among the peers after an additional user contributes h blocks. Then, $\psi_h(k, v)$ is characterized by a hypergeometric distribution,

$$\psi_h(k,v) = \frac{\binom{k}{h-(v-k)}\binom{B-k}{v-k}}{\binom{B}{h}}, \quad k = 0, 1, \ldots, B-1, B, \quad v = \max(k,h), \ldots, B-1, B \tag{3.9}$$

Equation (3.9) follows because there are $\binom{B-k}{v-k}$ ways in which $v-k$ blocks of the additional user do not overlap with the k previously available blocks, and there are $\binom{k}{h-(v-k)}$ ways in which the other $h-v+k$ blocks can overlap with previously available blocks (see Figure 3.2). A recursion to compute $\psi_h(k, v)$ is presented in Appendix B.2.

Our second observation regards the steady state probability that a randomly selected user is in stage h, denoted by $\sigma(h)$ $(0 \leq h \leq B-1)$. It can be shown that $\sigma(h) = 1/B$, $0 \leq h \leq B-1$. This is a consequence of the assumption that the download times of all blocks have the same mean, $1/\mu$. Note that, in general, if a

user downloads its $(h+1)^{th}$ block at rate μ_{h+1}, $0 \leq h \leq B-1$, the model can be easily parameterized by setting $\sigma(h) = (1/\mu_{h+1})/(\sum_{i=1}^{B} 1/\mu_i)$.

Our third and last observation concerns the total number of users in the system. The total population is characterized by a Poisson random variable with mean $B\lambda/\mu$. This follows from the fact that the sum of B independent Poisson random variables, with mean λ/μ, is a Poisson random variable with mean $B\lambda/\mu$.

Denote by $p_n(v)$ the probability that v blocks are available among the peers conditioned on the presence of n users in the system,

$$p_n(v) = P(V = v \Big| |\mathbf{n}| = n) \tag{3.10}$$

It follows from the discussion in the previous paragraph that

$$p(v) = \sum_{n=0}^{\infty} p_n(v) e^{-B\rho} (B\rho)^n / n!$$

The truncated version of $p(v)$, $p^{(N)}(v)$, is

$$p^{(N)}(v) = \sum_{n=0}^{N} p_n(v) e^{-B\rho} (B\rho)^n / n! \tag{3.11}$$

It remains to show how to compute $p_n(v)$. This is accomplished by making use of our first two observations, as summarized by the following lemma.

Lemma 3.3.1. *The probability of v available blocks, conditioned on n users in the system, $p_n(v)$, satisfies the following recursion,*

$$p_n(v) = \begin{cases} \sum_{h=0}^{\min(v,B-1)} \sum_{k=v-h}^{v} p_{n-1}(k)\psi_h(k,v)/B, & n \geq 1 \\ 1, & n = 0, v = 0 \\ 0, & n = 0, v \neq 0 \end{cases} \tag{3.12}$$

In Appendix B.3, we show that (3.12) correctly computes (3.10). Next, we further simplify recursion (3.12). Changing the order of the summations and adapting their limits accordingly yields

$$p_n(v)=\sum_{k=0}^{v} p_{n-1}(k) \sum_{h=v-k}^{B-1} \psi_h(k,v)/B, \qquad n \geq 1 \tag{3.13}$$

The base cases are $p_0(v)$=1 if $v = 0$ and $p_0(v)$=0 if $m \neq 0$. In Appendix B.4 we derive the following result,

$$\sum_{h=m-k}^{B-1} \psi_h(k,v) = \begin{cases} (B+1)/(B-k+1), & 0 \leq m \leq B-1 \\ k/(B-k+1), & v = B \end{cases} \tag{3.14}$$

Equation (3.14) is key to further simplifying (3.13). Replacing (3.14) into (3.13) yields, after algebraic manipulation,

Theorem 3.3.1. *The probability of v blocks being available among the peers, $p(v)$, equals*

$$p(v) = \sum_{n=0}^{\infty} p_n(v) e^{-B\rho} (B\rho)^n / n! \tag{3.15}$$

where $p_n(v)$ satisfies the following recursion, $0 \leq v < B$,

$$p_n(v)= \begin{cases} 1/B^n, & n \geq 1, v = 0 \\ p_n(v-1) + p_{n-1}(v)((B+1)/(B(B-v+1))), & n \geq 1, B > v > 0 \\ 1, & n = 0, v = 0 \\ 0, & n = 0, v \neq 0 \end{cases} \tag{3.16}$$

and $p_n(B) = 1 - \sum_{v=0}^{B-1} p_n(v)$. The approximation $p^{(N)}(v)$ for (3.15) is obtained by truncating the infinite sum at N and is computed in time $O(NB)$.

Theorem 3.3.1 yields an efficient algorithm to evaluate swarm self-sustainability. The algorithm has complexity $O(NB)$, since $p_n(v)$ is computed for $0 \leq n \leq N$ and

$0 \leq m \leq B$. Note also that once the elements $p_n(v)$ are computed for a fixed B, one can obtain the self-sustainability for different values of ρ in time $O(B)$.

Let $\boldsymbol{\varepsilon}(N)$ be the truncation error, $\boldsymbol{\varepsilon}(N) = p(B) - p^{(N)}(B)$. The maximum number of users in the system, N, can be chosen as a function of $\boldsymbol{\varepsilon}(N)$,

$$\boldsymbol{\varepsilon}(N) = \sum_{n=N+1}^{\infty} p(v) e^{-B\rho} (B\rho)^n / n! \leq \sum_{n=N+1}^{\infty} e^{-B\rho} (B\rho)^n / n! = 1 - \sum_{n=0}^{N} e^{-B\rho} (B\rho)^n / n! \tag{3.17}$$

If ρB is large ($\rho B > 1000$), the Poisson distribution is well approximated by a normal distribution. In this case, N can be chosen so that $1 - \Phi((N - B\rho)/\sqrt{B\rho}) \leq \eta$, where η is the desired error tolerance and $\Phi(\cdot)$ is the standard normal cdf.

Theorem 3.3.1 assumes $\mu_h = \mu$, $0 \leq h \leq B - 1$. If that is not the case, self-sustainability can be computed in time $O(NB \log B)$ using an alternative recursion. We refer the reader to Appendix B.12 for details.

3.4 Model Analysis

We derive closed-form expressions for the probability that v blocks are available among the peers in the system and for the mean number of available blocks, in §3.4.1 and §3.4.2, respectively. The closed-form expressions are useful in order to gain insight on how different system parameters impact self-sustainability. In §3.4.2 we use the closed-form expressions to compute the minimum popularity to attain a given self-sustainability level. In §3.4.3 we show that self-sustainability increases with file size. However, the closed-form expressions may lead to numerical problems, if used to compute the self-sustainability for large files (e.g., $B > 500$), since they involve sums and subtractions of large numbers. This is why the recursion presented in §3.3 is useful.

In order to simplify the closed-form expressions, in the remainder of this section we assume that $\gamma = \mu$, i.e., peers, after completing their downloads, linger in the

system as seeds for an interval with duration drawn from an exponential distribution with mean $1/\gamma$. Recall that when γ is finite, we make the dependence of $p_n(v)$ on γ explicit, and denote it by $p_n(v;\gamma)$.

3.4.1 Self-Sustainability Closed-Form Expression

Similar arguments to those in §3.3 (Theorem 3.3.1) yield, for $\gamma = \mu$ and $0 \le n \le N, 0 \le v \le B$,

$$p_n(v;\mu) = \begin{cases} 1/(B+1)^n, & n \ge 1, v = 0 \\ p_n(v-1;\mu) + p_{n-1}(v;\mu)(1/(B-v+1)), & n \ge 1, 0 < v \le B \\ 1, & n = 0, v = 0 \\ 0, & n = 0, v \ne 0 \end{cases} \tag{3.18}$$

This recursion can be solved (see Appendix B.8), to obtain [2]

$$p_n(v;\mu) = \binom{B}{v} \sum_{l=0}^{v} \binom{v}{l} (-1)^l (B-v+l+1)^{-n}, \qquad 1 \le n, \qquad 0 \le v \le B \tag{3.19}$$

In particular, the probability that all blocks are available among the peers, conditioned on the number of users in the system, is $p_n(B;\mu) = \sum_{l=0}^{B} \binom{B}{l} (-1)^l (l+1)^{-n}$. Using this expression we derive, in Appendix B.9 the corresponding unconditional probability, namely, the swarm self-sustainability,

$$p(B;\mu) = \sum_{l=0}^{B} \binom{B}{l} (-1)^l e^{-(B+1)\rho l/(l+1)} = 1 - \sum_{l=1}^{B} \binom{B}{l} (-1)^{l+1} e^{-(B+1)\rho l/(l+1)} \tag{3.20}$$

We now interpret (3.20) using the inclusion/exclusion principle, which allows us to apply the Bonferroni inequalities in §3.4.2 and §3.4.3. The term $\exp(-(B+1)\rho l/(l+1))$ is the probability that l specific (*tagged*) blocks are unavailable among the peers (refer

[2] The expression corresponding to equation (3.19) for the case $\gamma = \infty$ is found in Appendix B.6.

to Appendix B.10 for the derivation). So, $\binom{B}{l}\exp(-(B+1)\rho l/(l+1))$ is the sum of the probabilities that *any* l blocks are unavailable among the peers. Therefore, as a consequence of the inclusion/exclusion principle, the probability that *at least* one block is unavailable among the peers equals the rightmost summation in (3.20), and $p(B;\mu)$ is its complement.

In what follows, we use the above closed-form expression to analyze the mean number of blocks unavailable among the peers as well as the impact of the file size on the self-sustainability.

3.4.2 Minimum Load to Attain Self-Sustainability

We now provide a simple expression to estimate the minimum load necessary to attain high self-sustainability. The result relies on approximating swarm self-sustainability using the mean number of available blocks among the peers. The mean number of available blocks among the peers, $E[V]$, is $E[V] = B(1-q)$, where q is the probability that a tagged block is unavailable among the peers,

$$q = \exp(-\rho(B+1)/2) \tag{3.21}$$

The expression of q is readily obtained from (3.20). Note that the mean number of unavailable blocks, Bq, equals the first term ($l = 1$) in the rightmost summation in (3.20). This observation coupled with an application of the Bonferroni inequality [96] to (3.20) implies that $1 - p(B;\mu) \leq B - E[V]$. When $E[V] \approx B$, the upper bound $B - E[V]$ provides an approximation to the fraction of time that the swarm is not self-sustaining, $1 - p(B;\mu)$.

Next, we present a simple alternative derivation of (3.21). Let q_∞ be the probability that a tagged block is unavailable among leechers (excluding seeds). In order to compute q_∞, note that the mean time that a leecher holds a tagged block is $\sum_{l=0}^{B-1} l/(\mu B) = (B-1)/(2\mu)$ and the rate at which leechers acquire a tagged block is

given by a Poisson process with mean rate λ. Therefore, $q_\infty = \exp(-\rho(B-1)/2)$. In general, a tagged block is unavailable among the peers if no leecher owns the block and there are no seeds in the system. The probabilities of these two events are $\exp(-\rho(B-1)/2)$ and $\exp(-\rho)$, respectively. Their product yields (3.21).

We now use the above results to provide a simple expression to estimate the minimum load, $\rho^\star$, necessary to attain a given self-sustainability level, $A^\star$, when $\gamma = \infty$. It follows from the discussion in the previous paragraph that if $\gamma = \infty$ the probability that a tagged block is unavailable among the peers is $q_\infty = \exp(-\rho(B-1)/2)$. For values of q_∞ close to 0 ($q_\infty \leq 0.01$), $p(B) \approx 1 + E[V] - B = 1 - Bq_\infty$, as indicated in the beginning of this section. This approximation, in turn, can be used to select the load $\rho^\star$ to attain self-sustainability level $A^\star$,

$$\rho^\star \approx \left[2\log\left(B/(1-A^\star)\right)\right]/(B-1), \qquad \gamma = \infty \tag{3.22}$$

We further study (3.22) in §3.5.2.2, where we compare the results obtained with this approximation against those obtained with recursion (3.16), that provides exact results.

3.4.3 The Impact of File Size on Self-Sustainability

Increasing file size, B, increases the mean download time of peers, B/μ. In this section, we show that such an increase in the residence time of peers yields larger swarm self-sustainability. Theorem 3.4.1 states the result for $\gamma = \mu$, $B \geq 4$ and $\rho \geq 1.6$ and in §3.5 we provide evidence that it also holds when $\gamma = \infty$ and for small values of ρ.

Theorem 3.4.1 (File Size Impact)**.** *If $B \geq 4$, $\rho \geq 1.6$ and $\gamma = \mu$, self-sustainability increases with the file size, B.*

Proof. We denote the swarm self-sustainability for a given value of B and ρ as $\hat{p}(B, \rho)$. The Bonferroni inequalities [96], which generalize the inclusion/exclusion principle, applied to (3.20), yield upper and lower bounds on $\hat{p}(B, \rho)$,

$$1 - Be^{-(B+1)\rho/2} \leq \hat{p}(B, \rho) \leq 1 - Be^{-(B+1)\rho/2} + (B(B-1)/2)e^{-2(B+1)\rho/3} \quad (3.23)$$

It is easy to show that if $\rho \geq 1.6$ and $B \geq 4$ then

$$1 - Be^{-(B+1)\rho/2} + (B(B-1)/2)e^{-(B+1)\rho 2/3} \leq 1 - (B+1)e^{-(B+2)\rho/2} \quad (3.24)$$

from which the result follows by comparing $\hat{p}(B, \rho)$ and $\hat{p}(B+1, \rho)$. □

The key insight of Theorem 3.4.1 can be easily explained in terms of the busy periods of the proposed model. The busy period is defined as an uninterrupted interval during which the swarm is self-sustaining. As the file size increases, the number of blocks that need to be maintained increases linearly but the busy period of the system increases exponentially (see Chapter 2). Indeed, as the file size increases, the availability gain compensates the overhead to maintain a larger number of blocks, and self-sustainability increases.

3.5 Evaluation

In this section we (a) validate the proposed model, against detailed simulations, showing that despite the simplifying assumptions considered in our model, it captures how self-sustainability depends on different system parameters and (b) report results on the minimum popularity to attain a given self-sustainability level.

3.5.1 Experimental Setup

Our simulation experiments were conducted using the Tangram-II modeling environment [28]. Tangram-II is an event-driven, object oriented modeling tool. The

three main objects in our simulations are the tracker, the peer and the seed. Their implementations are based on the official BitTorrent protocol description [20, 50]. Every time a peer enters the system, receives a block or leaves, we record the event in our logs, the current timestamp, peer id, and signature (see §3.2.2).

3.5.1.1 Simulator and Protocol Descriptions

When a peer P joins the system, it receives a random list of fifty other peers from the tracker, which constitutes its *peer set*. Throughout the simulation, as peers leave the system the size of the peer set of P may dwindle to less than twenty. Once the peer set size is less than twenty, P requests additional neighbors from the tracker. The set of peers to whom P offers content blocks is a subset of its peer set, referred to as the *active peer set*.

BitTorrent proceeds in rounds of ten seconds. By the end of each round, peer P runs the tit-for-tat incentive mechanism. According to this mechanism, P reciprocates contents with those neighbors that contributed in the previous round. P selects r of those peers that contributed in the last round to add to its active peer set ($r \leq 4$). In the next round, the active peer set of P will consist of the r aforementioned peers plus $5 - r$ additional peers selected uniformly at random out of its peer set. This random selection of peers is referred to as *optimistic unchoke*, performed to allow peers to get bootstrapped as well as to let them learn about new neighbors. Finally, peers select blocks to download using the rarest first algorithm, except for the first four blocks, which are chosen uniformly. Each block of the file is divided into sixteen sub-blocks. After selecting the blocks to download, peers can get different sub-blocks (of the same block) from different neighbors concurrently. A block can be uploaded after all its sub-blocks are downloaded, assembled and checked using the block hash key.

In our experiments we observe that self-sustainability decreases with the size of the active peer sets. That is because, if the active peer sets are large, each peer splits its bandwidth across many other peers and blocks take longer to get replicated in the network. As mentioned above, we select an active peer set of size five, which is adopted by many BitTorrent implementations.

In our experiments the seed behaves as a standard peer, except that it (i) initially owns all blocks and (ii) is altruistic, hence does not execute the tit-for-tat algorithm. We did not implement the mechanism used by peers to download their last block, also known as *end game mode* [20]. This is inconsequential, though, since the end game mode does not significantly affect the steady state behavior of the system (see §3.5.2.1 and [11]).

3.5.1.2 Experimental Parameters

The configuration of our experiments consists of torrents that publish a file of size S divided into B blocks of size s, $s = 256$KB, a typical block size in BitTorrent. The number of blocks in the file, B, takes values 16, 50, 100 and 200, which corresponds to files of size 4MB, 12MB, 25MB and 51MB, respectively. Note that if a swarm is constituted of multiple files, which can be separately downloaded, we are interested in analyzing the self-sustainability of each individual file. A file of size 51MB already yields self-sustainability larger than 0.9 for $\lambda > 0.05$ peers/min (see Figure 3.4). Simulations to analyze such a steep increase in self-sustainability (see Figure 3.4) quickly requires prohibitively large execution times and significant variability in the metrics of interest across runs. For this reason, we focused on quantitatively validating our model for files with up to 200 blocks, but also use the model to analyze files of size greater than 200 blocks.

The uplink capacity of each peer is 39KBps, which corresponds to $\mu = 39/256 = 0.15$ blocks/s, a typical effective capacity for BitTorrent peers [70, Figure 1]. The

publisher maximum upload capacity is the same as that of a peer. A publisher that contributes the same capacity as an ordinary peer might correspond to either a domestic user or a commercial publisher that supports a large catalog of titles and only provides enough capacity for each swarm so as to allow peers to complete their downloads at rate μ. The peer arrival rate is varied according to the experimental goals between $(.25, .5, 1, 2, \ldots, 9)$ peers/min, as described next.

3.5.2 Model Validation

Our analytical model makes a number of simplifying assumptions as described in §3.2. In what follows, our goal is to show that even with those simplifying assumptions, discussed in §3.5.2.1, our model still captures swarm self-sustainability in a realistic setting, as shown in §3.5.2.2.

3.5.2.1 Validating Model Assumptions

Our aim in this section is to validate (a) that the mean download times of blocks are roughly the same (an exception being the first block downloaded by the peers), i.e., $\mu_h = \mu$, $2 \leq h \leq B$ (see §3.2.1) and (b) that the signatures of the users are uniformly distributed (see §3.2.2).

Figure 3.3(a) shows the download time of the h^{th} block downloaded by a peer. The boxplots and lines show the distribution quartiles and minimum and maximum values. Crosses indicate means download times of blocks, which are approximately the same, except for the first and last blocks. In particular, even though our simulator ignores the *end game mode* [11], in general peers do not experience difficulty finding a neighbor from whom to download their last block. The median of the last block is roughly the same as the one observed for the other blocks, and the mean is only slightly larger. The first block requested by a peer, however, takes longer to be downloaded. This happens because a peer can only download its first block after

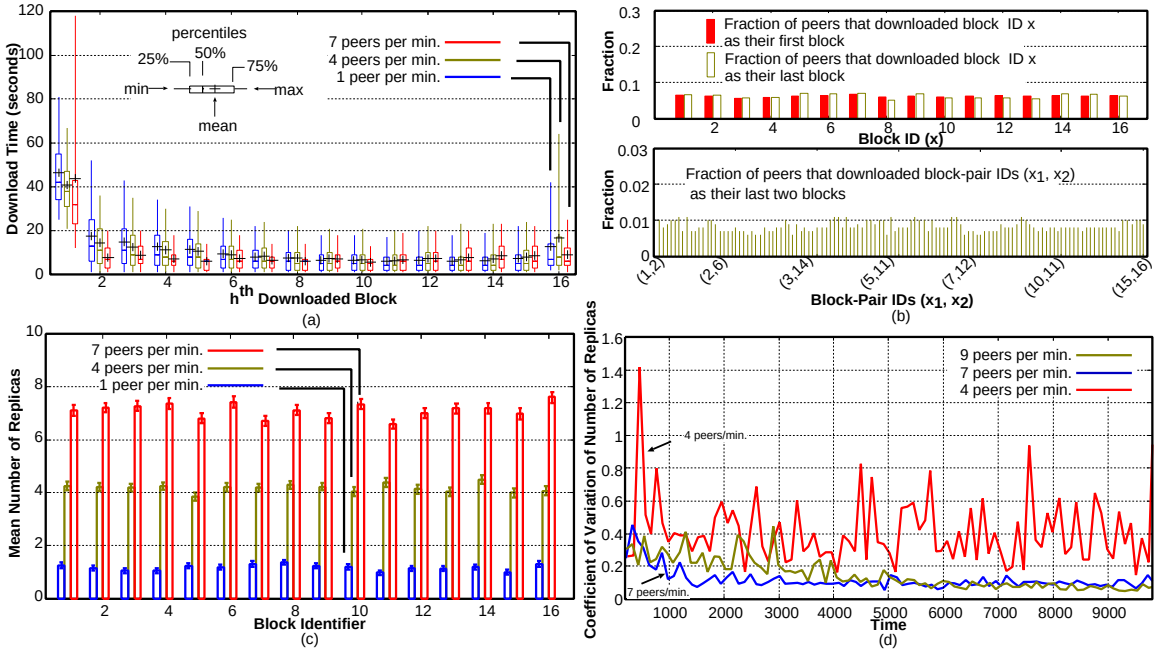


Figure 3.3. (a) Download times of h^{th} block downloaded by a peer. The boxplots show the four quartiles and crosses indicate means. (b) Top: shaded (resp., light) rectangles are fractions of peers that download block x as their first (resp., last) block. Bottom: fraction of peers that download block-pair (x_1, x_2) as their last two blocks. (c) Mean number of replicas of each block with 95% confidence intervals. (d) Coefficient of variation of number of replicas of blocks versus time.

being optimistically unchoked (see §3.5.1.1). Although this affects the time spent by peers in stage zero of our model and as a consequence the total download time, it is inconsequential to our self-sustainability estimates (the time that peers remain in stage zero of the upper layer model has no influence in our results). While BitTorrent peers download their first block, they cannot contribute to self-sustainability as they have no content to provide (see §3.5.1.1).

Our second goal is to study the users' signatures distribution (§3.2.2). For this purpose, we validated that the first two and last two blocks downloaded by a user are indeed uniform and then studied one of the consequences of the uniform and independence assumption, namely, that the number of replicas of each block in the system is well balanced.

Figure 3.3(b) (top) shows, for each block, the fraction of peers that downloaded that block as their first block (shaded bars). The figure was generated from independent samples: every 500 seconds, one user owning one block was randomly selected, and the identity of its block was recorded (the same procedure was repeated for users owning all but one block of the file (light bars)). Similarly, Figure 3.3(b) (bottom) shows, for every $B(B-1)$ block-pair, the fraction of peers that downloaded that pair as their first two blocks. Figure 3.3(b) indicates that the first and last blocks downloaded by users, as well as the first downloaded block-pair, are approximately uniformly distributed (the same procedure was repeated for users owning all but two blocks, with similar results).

Figure 3.3(c) shows the mean number of replicas of each block (including the one stored at the publisher) for $\lambda = 1$ peer/min, 4 peers/min and 7 peers/min. The mean number of replicas of each block is around $\rho(B-1)/2+1$, which corresponds to a well balanced system (see §3.4.2). Figure 3.3(d) corroborates this claim by showing the coefficient of variation of the number of replicas of blocks as a function of time. Let $r_{i,t}$ be the number of replicas of block i at time t. The mean number of replicas of blocks at time t is $\mu_t = \sum_{i=1}^{B} r_{i,t}/B$ and the coefficient of variation is $c_t = \sqrt{(\sum_{i=1}^{B}(r_{i,t}-\mu_t)^2/B)}/\mu_t$. Figure 3.3(d) indicates that throughout the simulation, the coefficient of variation is usually smaller than 0.8, which means that the number of replicas of blocks has a low variance. In a system where users signatures are uniform and independent we would observe similar behavior.

3.5.2.2 Validating Model Estimate of Self-Sustainability

To study how content popularity impacts self-sustainability, we simulated BitTorrent in the setting described in §3.5.1.2, varying the arrival rate of peers, λ, from 1 peer/minute to 8 peers/minute, in increments of 1 peer/minute, while maintaining all other parameters fixed. Equivalently, this corresponds to an increase in the load,

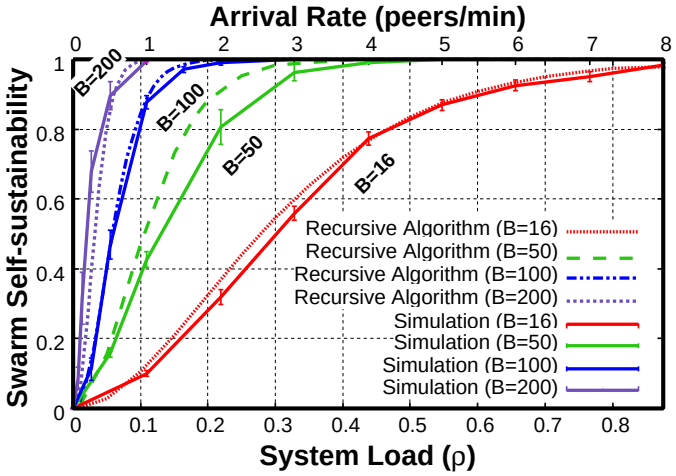


Figure 3.4. Model validation. Swarm self sustainability as a function of the system load. Results obtained with recursive algorithm and simulations (with 95% confidence intervals) are plotted with dotted and solid lines, respectively.

$\rho = \lambda s/\mu$, from 0.1 to 0.8. Each simulation lasted for 10,000s. Twenty one independent simulations were executed for each value of λ, and used to compute 95% confidence intervals. The same experiment is repeated for $B = 16$, 50, 100 and 200.

Figure 3.4 shows self-sustainability, A, as a function of the content popularity, λ, for $B = 16, 50, 100$ and 200. For unpopular contents, $\lambda = 1$ peer/min, and small files, $B = 16$, swarm self-sustainability is around 0.1 and the publisher needs to frequently provide blocks that are unavailable among the peers. As the popularity of the files increases, swarm self-sustainability increases and content is available even in the absence of the publisher. For $\lambda = 8$ peers/min, the fraction of time at which the publisher needs to provide blocks to peers is close to zero.

Figure 3.4 indicates that the results obtained with our model are close to those obtained through simulation. Even assuming that the mean download times of all blocks are the same ($\mu_h = \mu$, for $1 \leq h \leq B$), the model was able to capture the self-sustainability observed in our simulations. We also repeated the simulations with heterogeneous peer upload capacities, for $B = 16$ (the upload rate distribution taken

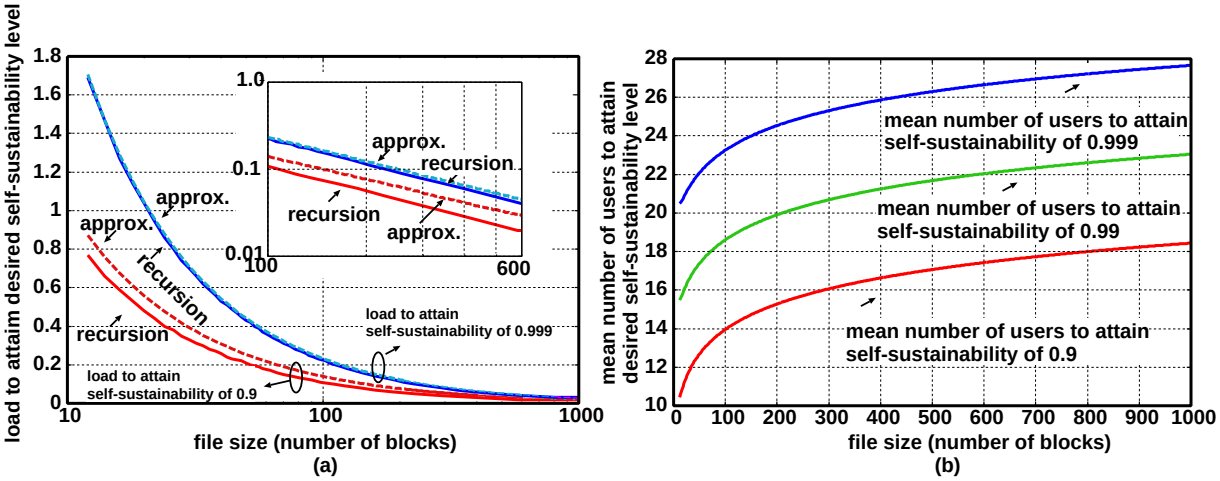


Figure 3.5. Larger files yield increased availability. (a) x-axis, file size. y-axis, necessary load ($\rho^\star = \lambda/\mu$) to attain self-sustainability greater than 0.9 (red) and 0.999 (blue). The results obtained using the approximation (3.22) are also shown. (b) the mean number of users in the system, $B\rho^\star$, to attain a desired self-sustainability level.

from the measured data used to generate Figure 1 in the BitTyrant study [70]) and our results did not qualitatively change (details in Appendix B.12).

Consider now the impact of file size on swarm self-sustainability. Figure 3.4 shows that for a fixed content popularity, as file size increases, self-sustainability increases. This is in accordance with Theorem 3.4.1, and reflects the fact that, as file size increases, peers stay longer in the system and the coverage, defined as the mean number of users in the system, increases. The higher the coverage, the greater the self-sustainability of the swarm. In fact, as file size increases, the number of blocks that needs to be maintained by the publisher increases linearly but the availability gain increases exponentially, as discussed in Chapter 2.

3.5.3 Popularity to Attain High Self-Sustainability

We now address the following question: what is the minimum file popularity (or load) needed to attain a given self-sustainability? Answering this question is useful

not only for publisher dimensioning but also for other strategic decisions such as how to distribute and bundle files across multiple swarms.

Recall that the load is defined as $\rho = \lambda/\mu$. Figure 3.5 shows the minimum load, $\rho^\star$, necessary to achieve high self-sustainability, $A^\star$ ($A^\star = 0.9, 0.999$), for file sizes varying between 2MB and 256MB ($B = 8, \ldots, 1,000$). This figure illustrates self-sustainability as predicted by the recursions contained in Theorem 3.3.1, eq. (3.16) and the approximation (3.22). Figure 3.5(a) suggests that when the goal is to find the minimum popularity to attain a high self-sustainability level, equation (3.22) can be used to approximate $\rho^\star$. However, if the goal is to compute self-sustainability under different loads and for different files sizes, as illustrated by Figure 3.4, the recursion provided by Theorem 3.3.1 needs to be used.

Figure 3.5(a) indicates that the popularity, $\rho^\star$, needed to attain a high degree of self-sustainability increases as the file size, B, decreases. In particular, the zoom in the figure shows that $\log \rho^\star$ is linear in $\log B$. The comments made at the end of §3.5.2 to explain Figure 3.4 also apply here. Peers take longer to download larger files, which increases block availability. Nevertheless, the benefits of leveraging peer-to-peer swarming can be noted even for small files. Figure 3.5(a) shows that for a file of 4MB ($B = 16$), an arrival rate of 8 peers/min (which corresponds to a load of 0.8) already yields a very high self-sustainability.

More insights on how file size impacts self-sustainability are obtained from Figure 3.5(b). Figure 3.5(b) plots the mean number of users in the system, also referred to as the mean coverage (see Chapter 2), necessary to attain a high level of self-sustainability. The curves in Figure 3.5(b) correspond to the the respective ones in Figure 3.5(a) multiplied by B. The main insight shown in this example is that the coverage necessary to attain a given self-sustainability level slowly increases as a function of B. As the file size increases, a slightly larger population suffices to attain a given self-sustainability level. For instance, for a file of size 10 a coverage of 20 is

necessary to attain self-sustainability of 0.999, whereas a coverage of 29 suffices to achieve the same self-sustainability if the file has 1,000 blocks.

3.6 Related Work

3.6.0.1 Modeling of Peer-to-Peer Swarming Systems

This chapter presents the first analytical model for publisher dependency estimation. We are unaware of related analytical work that analyzed swarm self-sustainability taking into account the fact that files are divided into multiple blocks, a very fundamental characteristic of these systems.

For large populations, Massoulié and Vojnovic [56] used a coupon collector model to show that asymptotically the distribution of blocks across the population is well balanced, and does not critically depend on the block selection algorithm used by the peers. Qiu and Srikant [73] and Fan et al. [32] also considered the large population regime, and used fluid approximations and differential equations to model the system assuming that the efficiency is always high.

In this chapter we are particularly interested in the small population regime. For small populations, Markov Chain (MC) models have been proposed by Yang and Veciana [95], providing insights on the performance of the system but not dealing with the problem of block availability among peers. Norros and Reittu [76], using a different model, studied the dissemination of a two-block file in a closed network accounting for the availability of the blocks. In this chapter, we consider an open network and propose a model which can be used to estimate self-sustainability of files of arbitrary size.

Here we studied the implications of the content popularity on the self-sustainability of swarms. In a real time setting, Leskela et al. [51] pointed out a phase transition in the stability of the peer-to-peer system as a function of the content popularity. In contrast, our system is always stable. Norros et al. [65] imply a phase transition of

the mean broadcast times as a function of the departure rate of seeds. In this chapter we are concerned with self-sustainability.

3.6.0.2 Hybrid Peer-to-Peer Swarming Systems

The literature on the use of peer-to-peer swarming systems for enterprise content delivery is rapidly growing [69, 27, 78]. The methodology usually consists of defining an optimization problem to be solved by the publishers and then showing how different system parameters affect the optimal bandwidth allocation strategy. The approach we take in this chapter is different. We are interested in the *minimum* fraction of time that the publisher must be active so as to guarantee that all blocks are always available.

Ioannidis and Marbach [44] study how quickly the bandwidth available at the server has to grow as the number of users increases. For this purpose they consider two query propagation mechanisms, the random walk and the expanding ring. Here, on the other hand, we assume that peers can always find the blocks they need in case they are available. While [44] focuses on the control plane and it's asymptotic analysis, here we focus on the data plane and account also for small files.

Hajek and Zhu [40] study the *stability* of hybrid peer-to-peer systems. In Appendix B.11 we show how the system described in this section relates to the one in [40]. Whereas Hajek and Zhu [40] show that if the publisher capacity is smaller than the arrival rate of peers the system is unstable, in the model presented in this chapter it is assumed that the publisher capacity is large enough to guarantee stability (see §3.7), and we are interested in swarm *self-sustainability*.

Wong et al. [100] and Susitaival et al. [86] propose models for content availability in BitTorrent without accounting for the fact that files are divided into blocks. Our model differs from [100, 86] in that we (a) consider a hybrid peer-to-peer system, in which a publisher is always available and (b) account for the fact that the file is

divided into blocks. Finally, explicit scheduling of blocks exchanges to minimize peer download times was studied by Mundinger et al. [63]. In this chapter we assume that peers exchange blocks using only local information, as in BitTorrent.

3.6.0.3 Balls and Bins

The derivation of some of our results fit into the balls and bins framework. Each user is allocated a set of blocks (balls) each of which must correspond to a different identifier (bin). In the context of balls and bins, a set of balls each of which must be allocated in a different bin is referred to as complex [48]. Previous work on the allocation of complexes into bins appears in Mirakhmedov and Mirakhmedov [62], Kolchin et al. [48, Chapter VII], and references therein. In particular, the definition of $\psi_h(i, m)$ was inspired by [16, Figure 1].

In a peer-to-peer setting, balls and bins were used by Simatos et al. [84] to study the duration of the regime during which the system is saturated because capacity is smaller than demand. The scenario studied in this chapter differs from [84] in several aspects. For instance, [84] considers a finite population of peers.

3.7 Discussion

Next, we discuss the assumptions adopted to yield a tractable model.

Uniformity and independence assumptions: In the performance model presented in §3.2.2 we assume that the signatures of users are drawn uniformly and independently at random. In particular, we do not account for correlations among users' signatures. Although such correlations are present in practice, our simulations have indicated that the independence assumption is appropriate in order to capture the self-sustainability of swarms.

In the user dynamic model presented in §3.2.1 we make the following assumptions: (i) peers arrive according to a Poisson process with rate λ (steady state as-

sumption), (*ii*) the download times of all blocks have the same mean, $1/\mu$ (smooth download assumption) and (*iii*) users leave the system immediately after completing their downloads, $\gamma = \infty$ (self-regarding users assumption). We discuss each of these in turn.

Steady state assumption: It has been shown in §2.4.3.4 that a vast number of long-lived swarms have relatively stable mean arrival rates over periods of months. Our model can be used to predict the self-sustainability of such swarms.

Smooth download assumption: Under this assumption, the capacity of the system scales perfectly with the number of users. Our simulations indicate that the mean download time of the first and last blocks are slightly larger than the others. Although our model has the flexibility to capture such asymmetries (see observation two in §3.3), we show that their implications in the estimates of self-sustainability are not significant (see §3.5.2.1).

Self-regarding users assumption: Our model has the flexibility to account for users that stay in the network after completing their downloads. However, in today's BitTorrent users have no incentive to stay in the system after obtaining the files of their interest. Therefore, we focus on the worst case scenario in which users, not having incentives to linger in the system after completing their downloads, depart immediately.

Finally, in our simulations we consider a publisher that is always online and that behaves like a typical peer.

Typical peer-like publisher: Our simulations indicate that if the publisher has the same capacity as a typical peer, the smooth download assumption holds and the swarm self-sustainability estimates of our model are accurate given the parameters and time spans considered in this chapter. Coping with intermittent publishers and devising dynamic bandwidth allocation strategies according to which the smooth download assumption holds is non trivial [40], and is subject of future work.

3.8 Conclusion

In this chapter we investigated the dependency of peers on a publisher that leverages peer-to-peer techniques for the dissemination of both popular and unpopular content. In particular, the latter deserve special attention, since unpopular content can represent a significant fraction of demand and revenue [5]. We believe that devising strategies for disseminating large catalogs of files leveraging peer-to-peer techniques is an important and interesting research area, and we see our model as a first attempt to shed light into the intrinsic advantages and limitations of peer-to-peer swarming systems for the dissemination of such catalogs.

CHAPTER 4

RECIPROCITY

Reciprocity is one of the fundamental pillars supporting peer-to-peer systems. In essence, the principle of reciprocity states that participants must contribute to the system with resources such as bandwidth or memory in order to accomplish their tasks. The primary goals of this chapter are to present a new foundational understanding of reciprocity in peer-to-peer systems, and to show how practical design decisions used to incentivize reciprocity impact performance.

We are motivated by systems in which users exchanging contents incur almost zero costs to replicate those contents. Examples are peer-to-peer swarming systems such as BitTorrent [20] and content trading systems such as TitleTrader [90]. In the former, users demand files and exchange blocks, while in the latter users demand and exchange non splittable commodities such as DVDs. Henceforth, we focus on peer-to-peer swarming systems such as BitTorrent.

4.1 Introduction

In BitTorrent, reciprocity happens naturally between users interested in the same content at the same time. Those users join a swarm and exchange blocks of files among themselves. Swarming systems also support reciprocity between users interested in different contents. That is because a swarm may be associated with a bundle of files, as discussed in Chapter 2. In this case, users joining a swarm can download only a subset of the files in the swarm and may already own some files when arriving to the system. In addition, as discussed in §1.2, users may download more content than they

originally sought, for bartering purposes or just for curiosity's sake, possibly guided by a recommendation system.

Reciprocity mechanisms broadly classify into two types. In the case of direct reciprocity, users follow the principle of *I scratch your back and you scratch mine*. In the case of indirect reciprocity, the return may come from a peer other than the recipient, i.e., peer A provides content to B at a rate equal to that at which content is being provided to it by peer C, regardless of whether or not B and C are the same individual. Users follow the principle of *give and you shall be given* [66]. Both direct and indirect reciprocity are used in peer-to-peer systems.

BitTorrent implements a direct reciprocity tit-for-tat incentive mechanism. The mechanism maintains a long term exchange between two peers only if they both benefit from it. Even though BitTorrent does not strictly require that the two peers exactly match each other's flows rates, it has been suggested that imposing this requirement leads to greater robustness [53] and FairTorrent [82] already employs it.

Indirect reciprocity is implemented in peer-to-peer systems such as eMule [30] and PACE [6] that rely on credits to accomplish fair exchanges. Other systems such as PeerTrust [103] take advantage of indirect reciprocity through reputation mechanisms. In some cases indirect reciprocity may be the only feasible solution [1]. Nevertheless, in the realm of peer-to-peer content distribution, direct reciprocity has its own advantages.

Direct reciprocity systems are simpler to implement than their credit based counterparts. In credit based systems users need to either rely on a bank (single point of failure) or store information in local files about credits received or given (prone to hacking). Inspired by the simplicity of direct reciprocity, we ask the following two questions,

- what is the loss of efficiency for enforcing direct rather than indirect reciprocity?

- how can trackers help users achieve efficient direct reciprocity schedules?

To answer the first question, we formulate a model of users with content and content demands. Using this model, we show that a class of indirect reciprocity schedules can be replaced by direct reciprocity schedules, provided that *(1)* users are willing to obtain undemanded content for the purpose of barter and *(2)* they are willing to use up to twice the bandwidth resources that they would have used under indirect reciprocity.

To address the second question, we propose a *broker-based* architecture (consider a broker to be a sophisticated BitTorrent tracker) and study, through simulation, several schemes where the broker provides greater amounts of information to users for the purpose of understanding the value of this information.

Our key contributions are the following.

Bound on loss of efficiency: We show that the loss of efficiency due to direct reciprocity, measured in terms of number of transmissions made by each user, is at most two for a class of indirect reciprocity schedules in which nodes do not relay content.

System design: We propose mechanisms based on brokers to perform match-making between users and to issue recommendations on content value for bartering. We validate experimentally how they enable efficient direct reciprocity exchanges.

The remainder of this chapter is organized as follows. In Section 4.2 we establish our main result concerning the loss of efficiency due to direct reciprocity. Sections 4.3 and 4.4 provide the system design and experimental results. Section 4.5 discusses the related literature and Section 4.7 concludes.

4.2 Loss of Efficiency due to Direct Reciprocity

In this section we prove that the loss of efficiency due to direct reciprocity is at most two for a class of relayless systems. To appreciate the nature of our result in

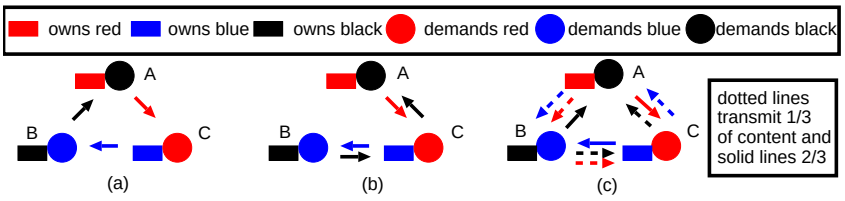


Figure 4.1. (a) Indirect reciprocity cycle (all nodes transmit 1 content) is converted into direct reciprocity network incurring loss of efficiency of (b) 2 (node C relays black and transmits blue); (c) 4/3 (all nodes transmit 4/3 of content).

a simple setting, consider the cycle shown in Figure 4.1(a), in which node v owns content c_v and requests content c_{v-1}, $0 \leq v \leq 2$, with subtraction done modulo 3. Under an indirect reciprocity constraint, every node transmits one content so as to have all demands satisfied. Beginning with the same workload, we now seek a direct reciprocity schedule. If *(a)* nodes are willing to receive one additional content for bartering purposes and *(b)* at least one node can transmit two contents simultaneously, one such schedule is shown in Figure 4.1(b). The loss of efficiency, measured as the maximum ratio of number of transmissions by a node in the direct and indirect reciprocity schedules, is two. That is because node C transmits content blue and black, the latter being received from B, for bartering purposes, and relayed to A. In the schedule of Figure 4.1(c) the loss of efficiency is 4/3.

Clearly, the loss of efficiency due to direct reciprocity is no more than two for any cycle. The direct reciprocity schedule consists of transmitting all contents in clockwise direction, except for one content, used for bartering purposes, that flows counterclockwise. In what follows, we show that the bound of two on the loss of efficiency extends from cycles to a more general class of networks.

4.2.1 Model

We define a network to be $N = (V, C, S, D, E)$ where V is a finite vertex set, C is a set of contents and S identifies the set of sources for each content, $S : C \to 2^V$. S_c is the set of sources for content c and C_v is the set of contents that node v initially has,

variable	description
V	set of users
C	set of contents
C_v	contents initially owned by v (v's endowment)
$S : C \to 2^V$	set of sources for each content
$D : C \to 2^V$	set of users that demand each content
$E \subset V^2 \times C \times \mathbb{R}^+$	set of labeled edges; each edge e transmits
$\{e = (v_1, v_2, c, r)\}$	content c from v_1 to v_2 at rate r
$N(V, C, S, D, E)$	dissemination network
$p_N(j)$	incoming edges to user j in network N
$s_N(j)$	outgoing edges from user j in network N
L	loss of efficiency (eq. (4.1))
$\overline{L}$	system loss of efficiency (eq. (4.2))

Table 4.1. Table of notation.

i.e., $C_v = \{c : v \in S_c\}$. C_v is also referred to as node v's endowment. D_c identifies the set of users that demand content c, $D : C \to 2^V$.

Content flows are characterized by $E \subset V^2 \times C \times \mathbb{R}^+$. If $e = (v_1, v_2, c, r) \in E$, then there exists a flow of content c from $v_1 = t(e)$ to $v_2 = h(e)$ at rate r. We use the notation $(t(e), h(e), c(e), r(e)) \in E$ to represent such an element. Note that unlike a directed graph, there may be several edges carrying different flows at different rates from one node to another. In this chapter, we use the terms *node* and *user* interchangeably. Table 4.1 contains the notation.

Throughout the proofs, it is assumed that all files are of the same size and users can exchange fractions of files. When v_1 transmits a fraction f of file c to v_2, represented by edge $e = (v_1, v_2, c, r)$, it is assumed that rate $r(e) = f$ files/slot is adopted. Therefore, all file transfers can occur in parallel, in one time slot (these assumptions are removed in simulations).

Let $p_N(j)$ denote the set of incoming edge flows to $j \in V$ in network N, $p_N(j) = \{e \in E : h(e) = j\}$. Similarly, $s_N(i)$ is the set of outgoing edge flows from $i \in V$.

The number of transmissions from node i to node j is $\tau_{i,j}$,

$$\tau_{i,j} = \sum_{e \in E} r(e) \mathbf{1}\{t(e)=i, h(e)=j\},$$

where $\mathbf{1}\{\mathcal{P}\}$ is an indicator variable equal to 1 if $\mathcal{P}$ is true. The number of transmissions by node i is $\tau_{i,\star}$, where $\tau_{i,\star} = \sum_{e \in s_N(i)} r(e)$. The number of transmissions to node j is $\tau_{\star,j}$, where $\tau_{\star,j} = \sum_{e \in p_N(j)} r(e)$.

We say that a network is a *dissemination network* if

1) content flows: for every node-demand pair (v, c), $v \in D_c$, there exist paths from sources of content c to v consisting of edge flows, which jointly carry all content c;

2) no redundancy: nodes don't receive redundant content over different edges (but might receive different parts of a content over different edges).

A dissemination network is an *indirect reciprocity network* if, for all $i \in V$, $\tau_{i,\star} = \tau_{\star,i}$, and a *direct reciprocity network* if, for all $i, j \in V$, $\tau_{i,j} = \tau_{j,i}$.

4.2.2 Main Result

Let the loss of efficiency at node v, L_v, be defined as the ratio of number of transmissions made by v in a direct reciprocity schedule, $\tau^d_{v,\star}$, over the number of transmissions in an indirect reciprocity schedule, $\tau^i_{v,\star}$. The loss of efficiency due to direct rather than indirect reciprocity, L, is the maximum of L_v across all nodes,

$$L = \max_{v \in V} \tau^d_{v,\star} / \tau^i_{v,\star} \tag{4.1}$$

A node that transmits content which is not in its endowment is also referred to as a *relay*. A relayless network is one which does not have any such nodes. In a relayless network, $c(e) \in C_{t(e)}$ for every edge $e \in E$. Next, we prove that given a relayless indirect reciprocity network, there is always a direct reciprocity one such that $L \leq 2$,

Theorem 4.2.1. *The loss of efficiency due to direct reciprocity in relayless networks is at most 2.*

4.2.3 Proof Overview

The proof of the main result is based on repeatedly transforming cycles in an indirect reciprocity dissemination network N that do not satisfy direct reciprocity into sets of direct reciprocity cycles without doubling the flow rate in/out of a node. For this purpose, we execute Algorithm 1.

To simplify presentation, we assume that, in the indirect reciprocity network that is input to Algorithm 1, each content might flow into a node only over one edge. Still, fractional allocations are allowed in the direct reciprocity network that is output by the algorithm. If nodes can't split their traffic among multiple paths, the loss of efficiency may be more than two (see Appendix C.2).

Algorithm 1 is constructed to satisfy two properties,

1) efficiency: the number of transmissions by each node in the direct reciprocity network is at most twice the number in the relayless indirect reciprocity network;

2) correctness: the direct reciprocity network is valid. For every node-demand pair (v, c) there exists a set of edges from sources of c to v. Node v receives all content c through these edges, each of which carries non-redundant fractions of c.

While studying the latter we will show that when reducing each cycle from indirect to direct reciprocity all nodes (a) receive the commodities that they request in the

Algorithm 1 IndirectToDirectReciprocity

Input: indirect reciprocity network without relays, N
Output: direct reciprocity network N'

1: $i \leftarrow 0$
2: **while** $E \neq \emptyset$ **do**
3: $\quad i \leftarrow i + 1$
4: $\quad$ construct cycle $\mathcal{C}_i$ from N [N might be rewired] (§4.2.4.1)
5: $\quad$ reduce $\mathcal{C}_i$ to a direct reciprocity network N_i (§4.2.4.2)
6: $\quad$ in network N, 1) remove edges in $\mathcal{C}_i$, 2) add to the endowment of nodes in $\mathcal{C}_i$ the contents that they received in step 5
7: **end while**
8: output N' is the combination of $N_1, \ldots, N_i$

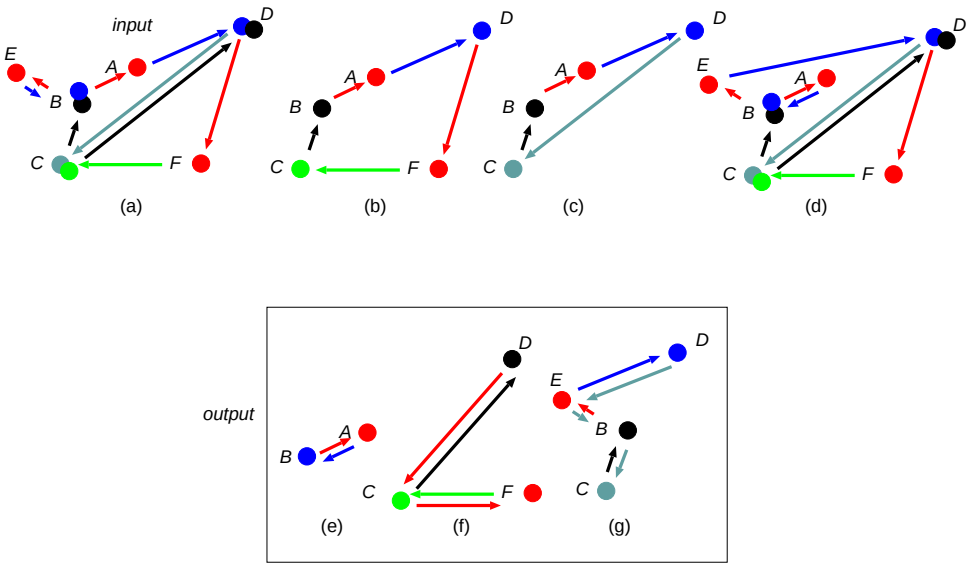


Figure 4.2. Illustrating the proof: (a) Input: indirect reciprocity network with no relays. Sources are implicitly identified by the tail of edges. (b) First identified cycle and (c) the corresponding short circuited cycle. Note that B receives a commodity involved in cycle (c), blue, via edge E-B that is not in (c). This prevents A from selecting blue as bartering commodity to be sent to B. To avoid such restriction, the indirect reciprocity network is reconstructed, as shown in (d). (e) First direct reciprocity network constructed from (d). (f) Second direct reciprocity network [from D-C-F in (d), using red as bartering commodity]. (g) Third direct reciprocity network [from D-E-B-C in (d), using dark green as bartering commodity]. The output consists of (e), (f) and (g).

cycle plus (b) possibly additional bartering commodities that they did not originally request in the input network.

Illustrative Example: Let us now present a simple example to illustrate how Algorithm 1 (detailed in the next section) works and why it satisfies the two desired properties. The example has only six nodes and five contents (Figure 4.2) but brings out the key insights in the conversion from indirect to direct reciprocity.

The first step consists of constructing a cycle $\mathcal{C}$ from the indirect reciprocity network (line 4 in Algorithm 1). Cycle $\mathcal{C}$ is constructed so as to satisfy the following *decoupling property*. If a node receives a given commodity through $\mathcal{C}$, say c, no other nodes in $\mathcal{C}$ can receive c via edges of N that do not appear in $\mathcal{C}$. In §4.2.4.1 we show that it is always possible to construct such cycle. Once the cycle is constructed, one

or more of the commodities involved in the cycle can be used for bartering purposes. As a consequence of this fact, the correctness property holds throughout the proof.

In our simple example, the intermediary steps to construct the first cycle, $\mathcal{C}_1$ (see Figure 4.2(e)), are illustrated in Figures 4.2(b)-(e). First, a cycle is identified (see Figure 4.2(b)). If there are edges joining two nodes that are not adjacent in the cycle, referred to as chords, use them to short-circuit the cycle. In our example, chord D-C is used to short-circuit cycle A-D-F-C-B into A-D-C-B (see Figure 4.2(c)). The remaining cycle still does not satisfy the decoupling property described above. Note that B receives blue from E, the latter not being present in the cycle. This prevents A from selecting blue as bartering commodity to be sent to B. To avoid such restriction, the indirect reciprocity network is reconstructed, as shown in (d) (see details in §4.2.4.1). In the reconstructed network, cycle A-B satisfies the decoupling property.

Our purpose now is to convert the remaining indirect reciprocity cycle into a direct reciprocity network. In our simple example, the indirect reciprocity cycle A-B (see Figure 4.2(e)) is already a direct reciprocity cycle. In §4.2.4.2 we show that it is always possible to reduce the former into the latter while satisfying the efficiency property (line 5). Such reduction is referred to as CycleReduce (see Algorithm 2). We then repeat the steps above to obtain the network in Figure 4.2(f).

Note that after reducing each indirect reciprocity cycle into a direct reciprocity network, we end up finding a direct reciprocity network in which all cycle nodes receive their required commodity, plus eventually some amount of the other commodities involved in the cycle. However, this never causes problems of double-reception of content, because of the decoupling property on the original cycle. Thus, the residual schedule, where 1) the edges in the original cycle have been removed, and 2) nodes are assumed to be sources of whatever they received thanks to CycleReduce, plus what they previously were sources of (line 6), is still a valid indirect reciprocity

schedule. The procedure can then be repeated on such schedule. Finally, combining the resulting direct reciprocity networks together (Figures 4.2(e), 4.2(f) and 4.2(g)) yields the desired direct reciprocity network (line 8).

4.2.4 Proof Details

Next, we provide the remaining details of the proof.

4.2.4.1 Constructing an Indirect Reciprocity Cycle From the Indirect Reciprocity Network

Starting from the indirect reciprocity network, we construct a cycle as follows. First, (i) pick a source v_1 and (ii) choose a content c_2 that v_1 is a leaf of. Then, (iii) traversing the edge corresponding to c_2 backwards, again (i) find a source v_2 of c_2. Repeat (i), (ii) and (iii) until finding, at step (i), a source that has already been incorporated into the cycle. Identify the cycle of sources, $v_i \overset{c_i}{\rightarrow} v_{i-1} \overset{c_{i-1}}{\rightarrow} \dots v_{i-m} \overset{c_{i-m}}{\rightarrow} v_i$, where v_i is source of content c_i. The set of sources is $\mathcal{S}$, $\mathcal{S} = \{v_i, \dots, v_{i-m}\}$.

Given the cycle of sources, we proceed by removing its chords. If there exists a source $v_j \in \mathcal{S}$ that is also a receiver of c_k with c_k distinct from c_{j+1}, and $v_l \in \mathcal{S}$ that is also a source of c_k, short-circuit the cycle by introducing the edge $v_l \overset{c_k}{\rightarrow} v_j$. Repeating this procedure, we obtain a cycle which cannot be further reduced, also referred to as a minimal cycle.

Next we proceed to remove violations of the decoupling property due to nodes within $\mathcal{S}$ receiving some commodity, say c_j, from a node outside of $\mathcal{S}$. To this aim, given a minimal cycle we search for a node v_{j-1}, that receives c_j, and violates the decoupling property. If there is no such node, we can proceed to the next step. Otherwise, v_{j-1} is also a receiver of c_k distinct from c_j. Content c_k is provided to v_{j-1} by $v_m \notin \mathcal{S}$ and to v_{k-1} by v_k, where $v_k, v_{k-1} \in \mathcal{S}$. Now replace edges (v_m, v_{j-1}) and (v_k, v_{k-1}) by (v_m, v_{k-1}) and (v_k, v_{j-1}), respectively. Since this replacement strictly reduces the length of the cycle, which now involves all nodes in $\mathcal{S}$ but v_k, repeated ap-

plication of the procedure results in a minimal cycle verifying the decoupling property. Next, we reduce minimal indirect reciprocity cycles into direct reciprocity dissemination networks.

4.2.4.2 Reducing Indirect Reciprocity Cycles to Direct Reciprocity Networks

We now focus on reducing an indirect reciprocity cycle to a set of direct reciprocity cycles without more than doubling the bandwidth requirement, as described in the proof of the following lemma.

Lemma 4.2.1. *Any indirect reciprocity cycle can be converted into a direct reciprocity network incurring a loss of efficiency of at most two.*

Proof. We consider two cases.

Case i) There exists at least one content that can only be provided by one node. This content can be used for bartering purposes and it is easy to see that in this case the lemma holds (see Figure 4.1(a) and Figure 4.1(b)).

Case ii) Every content can be provided by at least two nodes. We describe a transformation that shifts the amount of content delivered over indirect reciprocity cycles to direct reciprocity cycles (see Figure 4.3).

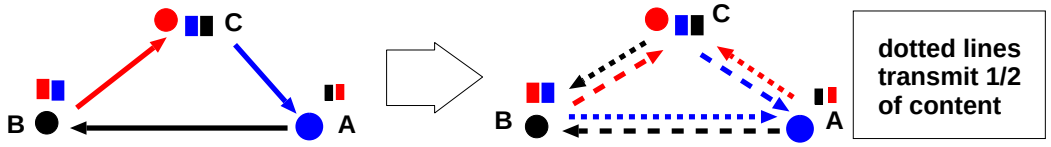


Figure 4.3. Indirect to direct reciprocity. Every content is offered by 2 nodes.

The key insight of our transformation is to relate cycles to perfect matchings in bipartite graphs. Let $G = (V, C, S, D, E)$ be a cycle in a dissemination network N. Let c_{v_j} be the content demanded by v_j in G. Now suppose that U is a copy of V and let $B = (V, U, E_B \subseteq V \times U)$ denote a bipartite graph representing possible content

exchanges between nodes in the cycle. Edge $(v_i, v_j) \in E_B$ if v_i is a potential source for v_j in the dissemination network G. With a slight abuse of notation, $(v_i, v_j) \in E_B$ if $c_{v_j} \in s_G(v_i)$.

A *matching* on B consists of a subset $M \subseteq E_B$ such that each node in $V \cup U$ is incident to at most one edge in M. A *perfect matching* is a matching where each node in V is incident to exactly one edge in M. Given a cycle G we can construct a corresponding perfect matching on B. Conversely, given a perfect matching M on B we can construct a set of cycles G (v_j is a successor of v_i in G if $(v_i, v_j) \in M$).

Our cycle reductions rely on the following result about perfect matchings, the proof of which is based on Hall's theorem [13, Thm. III.7] (see Appendix C.3). In what follows $|s_B(u)|$ denotes the number of incident edges to vertex u in a bipartite graph B.

Lemma 4.2.2. *Consider a bipartite graph $B = (V, U, E_B)$ where $|V| = |U| = n$, $n \geq 3$, that satisfies the following: (i) $(v_i, u_i) \in E_B$, $i = 1, \ldots, n$, (ii) $(v_{i+1}, u_i) \notin E_B$, $i = 1, \ldots, n$ and (iii) $|s_B(u)| \geq 2$, for all $u \in U$. Then, there exists at least two* distinct *perfect matchings M_1 and M_2 on B.*

We now use Lemma 4.2.2 to reduce cycles to direct reciprocity networks. This is accomplished through Algorithm 2. The input to Algorithm 2 is a cycle as well as two matchings, M_1 and M_2. The existence of the two matchings is guaranteed by Lemma 4.2.2. Sample inputs and outputs to Algorithm 2 are illustrated in Figure 4.4.

In the input to Algorithm 2, each node v_i transmits content c_{v_i} at rate r. Algorithm 2 outputs a set of cycles. Each node v_i is a member of two of these cycles and transmits half of content c_{v_i}, at rate $r/2$, through one of the cycles and the other half, at rate $r/2$, through the other cycle (see Figure 4.4).

The first cycle generated by Algorithm 2, A_1, contains fewer than n nodes, where n is the number of nodes in the input cycle. If A_1 is either a direct reciprocity cycle or satisfies Case *i)*, we are done with this cycle. Otherwise, apply Algorithm 2 to

Algorithm 2 CycleReduce

Input: indirect reciprocity cycle $A = (V_a, C_a, S_a, D_a, E_a)$ and matchings M_1 and M_2
Output: set of indirect reciprocity cycles $A_1, \ldots, A_k$

choose an arbitrary edge $e \in M_2$ such that $e \notin M_1$
$M_2 \leftarrow M_2 \setminus \{e\}$; $V_1 \leftarrow \{t(e)\}$; $E_1 \leftarrow \{e\}$; $i \leftarrow 1$
while $M_1 \cup M_2 \neq \emptyset$ **do**
    {construct cycle $A_i = (V_i, C_a, S_a, D_a, E_i)$ as follows}
    while $h(e) \notin V_i$ **do**
        remove e' from $M_1 \cup M_2$ such that $t(e') = h(e)$
        insert $t(e')$ into V_i and e' into E_i
        $e \leftarrow e'$
    end while
    identify cycle A_i
    re-add edges in E_i not in A_i into original matchings
    if $M_1 \cup M_2 \neq \emptyset$ **then**
        remove arbitrary edge e from $M_1 \cup M_2$
        $i \leftarrow i + 1$; $V_i \leftarrow \{t(e)\}$; $E_i \leftarrow \{e\}$
    end if
end while

A_1. Repeated applications of Algorithm 2 produces a cycle of size three within $n-2$ steps. It is trivial to reduce this three node cycle to direct reciprocity cycles. This increases the amount of flow carried by direct reciprocity cycles by at least $3r/2^{n-2}$.

At the end of this repeated application of the CycleReduce algorithm, we are left with a set of cycles. We can now apply *Case i)* and *Case ii)* (see page 84) to these cycles with the consequence that flow will continue to be shifted over to direct reciprocity cycles. This can be repeated until all flow is being carried over direct reciprocity cycles.

Two questions remain: *(1)* Do these transformations result in all content flows being shifted from indirect reciprocity cycles to direct reciprocity cycles? *(2)* Is the bandwidth required by the direct reciprocity flows no more than double that required by the original cycle? The answer to *(1)* is *yes* as any indirect reciprocity cycle can be reduced by repeated application of the above algorithm resulting in a positive decrease in indirect reciprocity flow. The answer to *(2)* is also *yes* as the algorithm above replaces cycles by a set of smaller cycles; bandwidth requirements are only

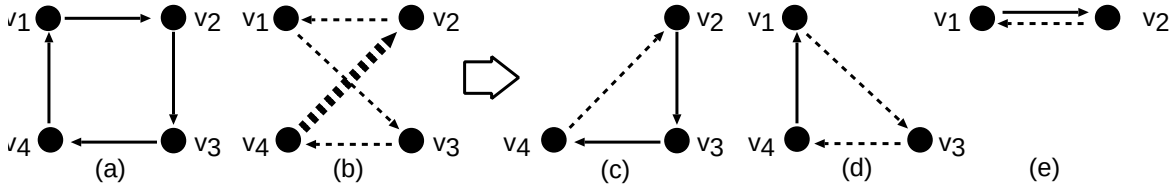


Figure 4.4. Illustration of the CycleReduce algorithm: the input is a cycle C and two matchings, (a) M_1 and (b) M_2; the first edge selected by the algorithm, $e = (v_4, v_2) \in M_2 \setminus M_1$, is marked in bold; (c) the first cycle constructed by the algorithm, A_1, has fewer nodes than C; e is the only edge from M_2 in this cycle; (d) the second cycle, A_2, and (e) the third cycle, A_3. Every node is in two cycles constructed by the algorithm. Hence, if each cycle transfers half of the contents in C, nodes receive the same amount of content in C as in the superposition of A_1, A_2 and A_3.

doubled (and in only some cases) at the last step when a direct reciprocity cycle is created. □

4.2.5 Tightness of the Bound

To show that the bound of two on the loss of efficiency is tight, consider a cycle of size $|V|$. Each node v owns content c_v and demands c_{v-1}, with subtraction done modulo $|V|$. Let $T_{|V|}$ be the minimum aggregate number of transmissions in a direct reciprocity schedule for a cycle of size $|V|$. The argument presented at the beginning of this section implies that $T_{|V|} \leq 2(|V| - 1)$ (Figure 4.1(b)). It can also be shown using linear programming and duality that $T_{|V|} \geq 2(|V| - 1)$ (see Appendix C.1). Therefore, there is at least one node, v, that transmits at least $2(|V| - 1)/|V| = 2 - (2/|V|)$ contents (when $|V| = 3$, Figure 4.1(c) illustrates a scheme where all nodes transmit exactly 4/3 contents). As $|V| \to \infty$ the number of transmissions from v tends to two, which yields the result.

4.2.6 General Network Topologies

The extension of our main result for general network topologies is left open.

Conjecture 4.2.1. *The loss of efficiency due to direct reciprocity is at most 2.*

In Appendix C.4 we present a battery of networks that must be handled by any proof of the conjecture, indicating when and where some tentative approaches fail.

4.3 System Design

Our purpose in this and the upcoming sections is to study the impact of design choices, such as how peers are matched, on the efficiency of the system. Our key metric for efficiency is the number of transmissions executed by each user before leaving the network. We first assume that users have only local information to make their decisions, and later consider brokers to facilitate exchanges.

4.3.1 Overview

We study a time slotted system. At every time slot, users are paired and content exchanges take place between paired users. Each user has the capacity to send one content per time slot. A content may correspond to a file or a block of a file. If file F is chopped into a set S_F of n_F blocks, S_F=$\{c_{F,1}, \ldots, c_{F,n_F}\}$, users interested in F, when arriving, demand all contents in S_F. For the purposes of the following two sections in most cases it suffices to let $n_F = 1$ for all files. In Section 4.4.2 we discuss a scenario requiring $n_F > 1$.

All users follow a direct reciprocity tit-for-tat strategy, according to which one content is sent only if another one, not yet owned, is received. We consider networks where all members are of the same type, either selective, semi-selective, or non-selective. An exchange between selective users must involve the transmission of demanded content in both directions whereas non-selective users exchange content even if what they receive was not demanded. An exchange between semi-selective users occurs if at least one of the users in the pair receives demanded content.

We now relate semi-selective exchanges to our main result on the loss of efficiency due to direct reciprocity being two. Let the system loss of efficiency, $\overline{L}$, be de-

fined as the ratio of the aggregate number of transmissions in the direct and indirect reciprocity schedules,

$$\overline{L} = (\sum_{v \in V} \tau^d_{v,\star})/(\sum_{v \in V} \tau^i_{v,\star}) \tag{4.2}$$

Starting from a workload that can be satisfied using indirect reciprocity, we have $\sum_{v \in V} \tau^i_{v,\star} = \sum_{c \in C} |D_c|$. If, under direct reciprocity, all users adopt semi-selective exchanges, for every two transmissions by paired users, at least one delivers demanded content, $\sum_{c \in C} |D_c| \geq \sum_{v \in V} \tau^d_{v,\star}/2$. Therefore, $\overline{L} \leq 2$. Note however, that due to the lack of a central controller some users may still transmit more than twice the number of contents than they would in an indirect reciprocity scheme (i.e., L may be greater than two) and/or the system may deadlock before all users receive all the desired contents. One of our goals in the following section is to identify scenarios under which semi-selective exchanges suffice to allow almost all users to have their demands satisfied, even when users have incomplete information about the system.

4.3.2 Workloads

We consider two types of workloads in our experiments, cycle and Zipfian workloads.

4.3.2.1 Cycle Workload

In a cycle with $|V|$ users and non-overlapping demands, each user v owns content c_v and demands content c_{v-1}, where subtraction is done modulo $|V|$. Unless otherwise stated, the number of contents is $|C| = |V| = 200$ and the number of contents initially owned and demanded by each user equals one.

In a cycle with overlapping demands, multiple users may request the same content and some users, referred to as contentless, arrive to the network without bringing any contents. In particular, we assume that user v demands content $c_{\lceil v/5 \rceil - 1 \bmod \lceil V/5 \rceil}$ and, if $v \bmod 5 = 0$, owns content $c_{v/5}$. The number of contents in the network is $|C| = |V|/5 = 40$. In a network supporting indirect reciprocity, contentless users

may work as relays. In a network enforcing direct reciprocity, one needs to build mechanisms to bootstrap such users, as described in Section 4.4.2.

4.3.2.2 Zipfian Workload

In a Zipfian workload, each user demands content k with probability $p_k = 1/k$, and owns content $C - k - 1$ with probability p_k, $k = 0, \ldots, C - 1$. Note that the most demanded content is also the scarcest one. Unless otherwise stated, we set the number of contents $|C| = 100$, number of users $|V| = 200$ and endowment $|C_v| = 10$, $0 \leq u \leq |V| - 1$.

4.3.3 Brokers: the Public Board and the Matchmaker

While analyzing each workload we begin by assuming that users are randomly paired and, if multiple contents are available for exchange, ties are broken arbitrarily. We then progressively add complexity to the system, by letting users decide which contents to send based on a public board and allowing a matchmaker to decide how users are paired, as described next.

4.3.3.1 The Public Board

When users have multiple contents to offer, a public board may help them decide which content to transmit. We consider a public board that informs, for each content, at each time slot, the number of users that own and demand it. If a user has multiple contents to offer to its neighbor, the candidate with highest ratio of number of copies demanded to the number of replicas available is selected. Conversely, if a user needs to evict a content from its cache, it evicts the one with the lowest ratio.

4.3.3.2 The Matchmaker

A matchmaker may pair users based on the contents that they own and demand. We assume the matchmaker uses a maximum weight matching algorithm to pair

users with weights set as follows. Users are randomly divided into two subsets V_1 and V_2 of equal size (if the network has an odd number of users, one user is randomly discarded). There is an edge from a user $v_1 \in V_1$ to $v_2 \in V_2$ with weight 2, 1 and $\epsilon = 0.1$ if v_1 and v_2 can establish a selective, semi-selective and non-selective exchange, respectively. There is no edge between pairs of users that cannot exchange contents. By prioritizing matches between users that can exchange contents, the matchmaker increases the throughput of the system, measured by the number of useful exchanges per slot.

4.4 Bartering and Brokers

Our goal now is to illustrate *(a)* the system benefits from users prefetching contents even when the contents are not of immediate interest and *(b)* the impact of brokers on the performance of users. For this purpose, we analyze both the cycle and the Zipfian workloads.

4.4.1 Bartering: The Benefits of Content Prefetching

We now evaluate the implications of users downloading undemanded contents, for bartering purposes. In the cycle workload, bartering is essential. If users are selective, no exchanges can occur and the system remains deadlocked forever. On the other hand, our experiments indicate that if users are not selective they can have their demands satisfied in an average of 83.63 time slots, even without a broker.

In a Zipfian workload, bartering decreases the chance that a content becomes unavailable before all requestors are able to download it, simply because more contents get replicated if users take advantage of their exchange opportunities. Note, however, that in the face of lack of information about which content to barter or which user to contact, bartering may lead to an increase in the number of transmissions without a significant increase in the availability of the contents or in the download times.

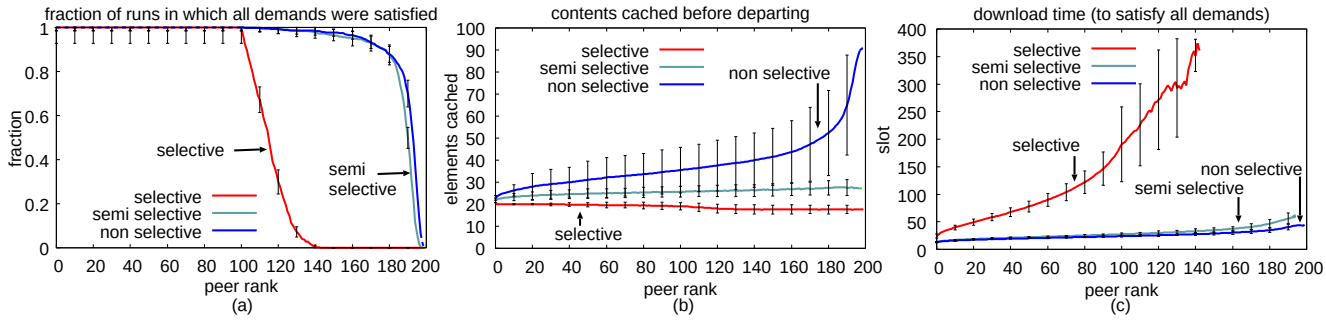


Figure 4.5. Zipfian workload. Users benefit from prefetched content and semi-selective exchanges present a good compromise between performance and overhead: (a) fraction of users that satisfied all demands; (b) elements cached when leaving the network (equals transmissions plus endowment); (c) download times.

To illustrate the above observations, we consider a Zipfian workload where the number of contents initially demanded and owned by each user is 10 (last column of Table 4.2), users don't have constraints on their cache sizes or the time they spend in the system, and the public board is available. We repeat our experiments 1000 times, each experiment ending when either all users satisfy their demands or a deadlock is reached. Figure 4.5(a) shows the fraction of users that were able to complete the download, Figure 4.5(b) shows the number of contents in the cache when the user left the network and Figure 4.5(c) shows the time that each user took to download all the demanded contents. The x-axis of Figures 4.5(a)-(c) list the users in order of download times to satisfy all demand. For instance, in Figure 4.5(a), the 10^{th} user to leave the system was always able to have its demand satisfied, the 120^{th} user was able to have its demand satisfied in roughly 30% of the simulation runs with selective users, while the 200^{th} was never able to download all the requested contents. Note that some curves in Figures 4.5(b) and 4.5(c) are not monotonically increasing because the metrics showed are conditioned on the peers concluding their downloads. The higher the peer rank, the smaller are the chances that the peer satisfies all its demands and the smaller the number of samples collected.

	workload			
graph	cycle	cycle	Zipfian	Zipfian
leave network?	no	no	yes	yes
contentless users	no	yes	no	no
overlapped demand	no	overlapping	overlapping	overlapping
demand per user	1	1	2	10
exchange strategy	outcome			
selective	*generally unfeasible*			
matchmaker	deadlock	deadlock	deadlock in few iterations	$L \approx 0$
no matchmaker	deadlock	deadlock	deadlock in few iterations	frequent deadlocks
semi-selective	*generally feasible, incurring small loss of efficiency*			
matchmaker	$L \leq 2$	$L_S \leq 2$	$\overline{L} \leq 2$ frequent deadlocks	$L \approx 0$ infrequent deadlocks
no matchmaker	$\overline{L} \leq 2$	$\overline{L}_S \leq 2$	$\overline{L} \leq 2$ frequent deadlocks	$\overline{L} \leq 2$ infrequent deadlocks
non-selective	*generally feasible, incurring large loss of efficiency*			
matchmaker	$L \leq 2$	$L_S \leq 2$	$L \geq 2$	$L \approx 0$
no matchmaker	$L \geq 2$	$L_S \geq 2$	$L \geq 2$	$L \geq 2$

Table 4.2. Direct reciprocity design space: exchange strategies, workloads and respective outcomes. L: user loss of efficiency; $\overline{L}$: system loss of efficiency; L_S: user loss of efficiency of sources; $\overline{L}_S$: system loss of efficiency of sources.

The performance of non-selective and semi-selective users is better than the performance of the selective ones. The fraction of completed downloads is higher (Figure 4.5(a)) and the download times are smaller (Figure 4.5(c)) when users are non-selective or semi-selective as opposed to selective. To achieve high performance, the overhead incurred by non-selective users is the additional number of contents cached before leaving the system (Figure 4.5(b)). Semi-selective users present the best outcome, high performance (small download times and large number of conclusions) and low overhead (small number of contents cached before leaving the system).

Even though with selective exchanges a significant fraction of users are not able to satisfy their demands, most users download a considerable amount of the requested contents. We observed that across our simulations 90% of selective and semi-selective users were able to conclude all their downloads. This fraction drops to 50% for selective users. Nevertheless, even with selective exchanges more than 90% of the users were able to download at least eight out of the ten demanded contents.

4.4.2 Brokers: The Role of Information on the Loss of Efficiency

We now analyze the role of brokers to recommend contents to users (public board) and pair them efficiently (matchmaker). As in the previous section, we begin by analyzing the simple cycle workload. In cycles with semi-selective exchanges, the system loss of efficiency, $\overline{L}$, is at most two. However, without a matchmaker some users may need to transmit more than two contents to their neighbors. This happens, for instance, if in the first three time slots user v_2, with endowment c_2, is matched with v_3, v_4 and v_1, in that order, transmitting three contents, c_2, c_3 and c_1, respectively. In contrast, with a matchmaker each user transmits at most two contents. In the first (resp., second) time slot users with even (resp., odd) index transmit useful content to users with odd (resp., even) index, and the loss of efficiency incurred by each user is exactly two. Finally, if users are not selective, the system loss of efficiency may be higher than two, which happens, for instance, if in the first time slot user $v_{i \bmod V}$ is matched with user $v_{i+2 \bmod V}$, $0 \leq i \leq |V| - 1$.

If the cycle involves contentless users (second column of Table 4.2) the observations made in the paragraph above still apply to the network involving only the sources. However, if all users are to receive all demanded contents, a mechanism to bootstrap the contentless users is essential. One solution consists of giving them credits that can then be swapped for other contents.

A second solution, more in line with optimistic unchoke in BitTorrent, consists in allowing some users to send a small number of blocks to their contentless neighbors, without being immediately reciprocated. If the contentless neighbors are able to complete their downloads only after offering blocks back to the original donor (such schedule being enforced by the matchmaker), direct reciprocity will still hold.

In Zipfian networks brokers are particularly important when users have limited cache sizes and delay constraints. Consider a system in which each user can cache 10 files and departs the network either when it has satisfied its demand or after 40 time

slots have elapsed, whichever comes first. As soon as a user leaves the system, it is replaced by a new one. The other parameters are those described in the last column of Table 4.2.

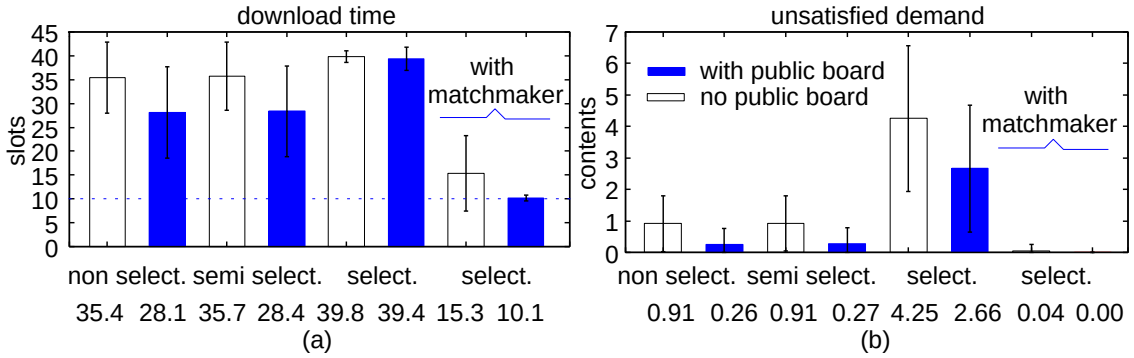


Figure 4.6. Benefits of a public board and a matchmaker with a Zipfian workload are significant. Maximum residence time of users is 40 and cache size 10.

Figure 4.6(a) shows the download times experienced by the users with non-selective, semi-selective, and selective exchanges. The more selective users are, the larger the delays they incur. Figure 4.6(b) shows the fraction of unsatisfied demand, which also increases as users become more selective. As one might expect, the use of the public board, both to decide which content to send as well as to decide which content to evict, plays a key role in this scenario. The public board reduces the delay as well as the fraction of unsatisfied demand, as observed comparing the blue and white bars, corresponding to the system with and without the public board.

More surprising is the effect of the matchmaker (bars on the right of Figure 4.6). The use of a matchmaker coupled with a public board yields an optimal delay of 10 time slots with all users having their demands satisfied. The rationale behind this outcome is as follows. While the public board helps unpopular contents replicate early on, the matchmaker guarantees that all users are matched in such a way that every time slot is used to transmit demanded contents.

The scenario above drastically changes if users request only 2 contents when joining the network (third column of Table 4.2). In this case, even with brokers semi-selective exchanges no longer suffice to guarantee that all users get all demanded contents. Non-selective exchanges must be employed, and the system loss of efficiency may be greater than two.

4.5 Related Work

The literature on incentives in peer-to-peer systems focusing on its relation to freeriding [82], clustering [49] and system design [71] is vast. Nevertheless, there appears to be no previous study on the *fundamental* pros and cons of direct and indirect reciprocity for the dissemination of digital goods.

In the economics literature, direct reciprocity bartering schemes are considered precursors of more sophisticated economies involving money and indirect reciprocity. Arpejis et al. [6] argue that prices can also play an important role in peer-to-peer systems. The authors compare implicit prices in bilateral exchanges (direct reciprocity) with explicit prices in multilateral exchanges (indirect reciprocity) and propose a system that implements the latter. Our work, in contrast, suggests that in the context of peer-to-peer file sharing, direct reciprocity has its own advantages and might suffice to implement an efficient and incentive-compatible system.

In the economics terminology, identifying an indirect reciprocity schedule is referred to as market clearing [1]. In this chapter, we assumed that an indirect reciprocity schedule is provided. Future work consists of efficiently finding such schedule, perhaps using distributed deadlock detection algorithms [19].

Our work is related to those on scheduling in single swarm peer-to-peer systems [63]. Reciprocity constraints were first incorporated in peer to peer schedules by Figueiredo et al. [34]. While [34] focus on cooperative versus non cooperative systems, we focus on direct versus indirect reciprocity. In addition, we address

multi-commodity scheduling, which has received little attention in the realm of peer-to-peer systems (one exception being [55]).

4.6 Discussion

In this section we discuss the key simplifications adopted in our model and simulations.

Homogeneous file sizes assumption: Under this assumption, all files have identical sizes. Alternatively, one can interpret the units of exchange as blocks of the files. In this case, users that arrive to the system demanding a file are mapped into requesters for all blocks of that file.

Perfect information assumption: Our model assumes that users have perfect information about the demands and ownerships of every other user. We have shown through simulations that even if users have local information, the bound of two on the loss of efficiency can still be attained in scenarios of practical interest.

Time slotted system and pairwise contacts assumption: In our simulations we assumed a time slotted system in which users are paired every time slot. In BitTorrent, users exchange content simultaneously with up to five neighbors. Considering this and other more complicated topologies in our simulations is subject of future work.

Static network assumption: Our model assumes a static network. In particular, users reside in the network until all demands are satisfied. In our simulations we considered the dynamic network case, in which users depart the network as soon as they conclude their downloads.

4.7 Conclusion

In this chapter we provided new foundational results on reciprocity mechanisms. In the context of our proposed model, we showed that the loss of efficiency due to

direct reciprocity is at most two in networks without relays. Then, using simulations, we indicated that in many situations direct reciprocity can lead to high performance with marginal overhead. In particular, we identified that in cycles and Zipfian workloads it is crucial that users download contents for the purpose of bartering, and a low loss of efficiency may be achieved if brokers are available to perform matchmaking between users and to issue recommendations on content value. Although our model is based on a number of simplifying assumptions, such as all contents having the same size and all peers following the same set of rules, we believe that it sheds important insights on the fundamental limitations and potentials of direct reciprocity. Future work consists of finding a proof or a counter-example for the conjecture on the loss of efficiency due to direct reciprocity being two in general networks.

CHAPTER 5

ALGORITHMIC ASPECTS, EQUILIBRIA AND COMPETITION

In Chapter 2 we studied how bundling affects availability and in Chapter 3 we considered the dependence of peers on a stable publisher, introducing the notion of self-sustainability. What if such stable publishers are run by enterprises? And what if enterprises take advantage of bundling in order to increase the self-sustainability of their swarms? These are two of the questions addressed in this chapter, in which we consider algorithmic and economic issues in the realm of enterprise peer-to-peer swarming.

5.1 Introduction

In this chapter we consider the use of swarming systems by enterprises. For that purpose, the swarming system needs to be coupled with credit and copyright mechanisms (e.g., DRM [97]), which ensure that users gain access to a file only after paying for its access. In this case, users download bundles but have limited access to the files. In order to obtain full access, they need a passcode, which is made available only after payment.

The idea of using bundling to promote unpopular content is not new. In the economics literature bundling was proposed decades ago as a mechanism to increase sales and extend monopoly power. For information goods, bundling has been shown to be advantageous because it permits firms to smooth demand for multiple goods. Traditionally it has been assumed that information goods have zero marginal cost for

production and dissemination. In this chapter, however, we are particularly interested in the impact of these costs on the publishers, which may be non-negligible if, for instance, we consider publishers of a large number of unpopular contents.

In this chapter we show how and when bundling, coupled with a swarming mechanism, helps a publisher increase utility, defined as its revenue minus costs (see §5.3). We are interested both in the monopoly and the duopoly cases. When there is only one enterprise involved, we consider both algorithmic as well as economic issues faced by the enterprise when deciding how to bundle and set prices. When multiple enterprises are involved in the market, the bundling strategy of one enterprise impacts the outcome of the others. Note that different firms may offer overlapping, partially overlapping (or weak substitutable), or non-overlapping contents. In this multi-firm scenario, game theory emerges as a natural tool to study the possible market outcomes.

We ask the following four questions, the first from an algorithmic standpoint and the other three from an economic one:

1. if there is only one enterprise involved, how to bundle the files so as to attain a given self-sustainability level for each swarm?

2. still in the case of a monopoly, for a given bundling strategy does there exist an equilibrium and, if so, what is it?

3. how does the monopoly equilibrium change as functions of the bundling strategy?

4. in the case of a duopoly, what is the impact of the parameters of the system on the Nash Equilibrium?

In this work we assume that users have uniform valuations over files and that each user, when accessing the system, is interested in a single file. Users are reluctant

to download bundles since the download of a bundle may lead to increased delays. Publishers, on the other hand, are willing to push bundles to the users since this increases the availability of the contents. We propose a model that captures the utilities of users and publishers as functions of the prices and bundling strategies. Using this model, we derive properties of the system equilibrium and perform a sensitivity analysis of the publishers' utility as a function of the bundling strategy.

For the duopoly, we consider a special parameterization of the utility functions for users and publishers and plug them into normal form game matrices (for which the Nash equilibrium is guaranteed to exist). Using these matrices, we show how different system parameters such as willingness of users to download bundles and level of overlap of contents between publishers affect the Nash equilibrium.

We provide the following answers to our initial questions:

- in its most general form, the problem of determining the optimal bundling strategy is NP hard. However, if users express how much they are willing to wait in order to download each file, the problem can be solved using a greedy strategy;

- for the monopoly, we establish conditions for existence and uniqueness of an equilibrium;

- we present scenarios under which the optimal bundling level can be easily determined. In particular, if the cost to the publisher is proportional to the number of bundles it offers and does not depend on demand, the publisher's utility as a function of the bundling level has a unique local maximum which can be easily determined using a gradient descent strategy;

- for the duopoly, we show that different games emerge as a function of the system parameters. This multitude of possible games may be regarded as a sign of the complexity of the problem in hand.

5.2 Algorithmic Aspects Concerning Bundling In The Monopoly

How to combine files into bundles? In this section, we consider two different formulations of this problem. The first assumes static bundles while the second encompasses dynamic bundles and embraces individual users preferences over files. Whereas the first yields an NP hard problem, a greedy strategy suffices to solve the second.

5.2.1 Minimization of Download Time with Constraint on Availability is NP Hard

We consider a set S of F files, $|S| = F$. File i has size s_i blocks, $s_i \in \mathbb{Z}$. Requests arrive for file i at rate λ_i, and peers download content at rate μ.

Let M be the maximum number of bundles that can be managed by the publisher. We assume that each file must be included in one and only one bundle. The decision variables are:

- B, the number of bundles;
- x_{ib}, for $1 \leq i \leq F$, $1 \leq b \leq M$,

$$x_{ib} = \begin{cases} 1, & \text{if content } i \text{ is assigned to bundle } b \\ 0, & \text{otherwise} \end{cases}$$

Note that the download time for bundle b is

$$\sum_{i \in S} x_{ib} s_i / \mu$$

Let $S_{b(i)}$ be the size of the bundle that contains file i is in, and $s_{b(i),j}$ be the size of the j^{th} file in the bundle that contains file i is in. The goal of the publisher is to minimize the expected bundle download time. The objective function is

$$\min(1/\mu)\sum_{i=1}^{F}\lambda_i S_{b(i)} = \tag{5.1}$$

$$= \min(1/\mu)\sum_{i=1}^{F}\lambda_i(s_{b(i),1}+s_{b(i),2}+\ldots+s_{b(i),N(i)}) = \tag{5.2}$$

$$= \min(1/\mu)\sum_{b=1}^{B}(\sum_{i\in S}x_{ib}\lambda_i)(\sum_{i\in S}x_{ib}s_i) \tag{5.3}$$

Let $s_{(b)}$ and $\lambda_{(b)}$ be the size and request rate for bundle b.

$$s_{(b)} = \sum_{i\in S}x_{ib}s_i, \qquad \lambda_{(b)} = \sum_{i\in S}x_{ib}\lambda_i \tag{5.4}$$

It follows from (3.22) that, for large values of self-sustainability A,

$$A \approx 1 - s_{(b)}e^{-\lambda_{(b)}(s_{(b)}-1)/2\mu}, \qquad 1\le b\le B \tag{5.5}$$

(5.5) is obtained from (3.22) replacing $\rho^\star$, B and $A^\star$ by $\lambda_{(b)}/\mu$, $s_{(b)}$ and A, respectively.

Therefore, the publisher problem is stated as follows.

PUBLISHER PROBLEM: Obtain B and $\{x_{ib}\}$ so as to

$$\text{minimize} \sum_{i=1}^{F}\lambda_i S_{b(i)} \tag{5.6}$$

where

$$x_{ib} \in \{0,1\}, \qquad 1\le i\le F, 1\le b\le B \tag{5.7}$$

$$\sum_{b=1}^{B}x_{ib} = 1, \qquad 1\le i\le F \tag{5.8}$$

$$B \le M \tag{5.9}$$

$$s_{(b)} = \sum_{i\in S}x_{ib}s_i, \qquad 1\le b\le B \tag{5.10}$$

$$\lambda_{(b)} = \sum_{i\in S}x_{ib}\lambda_i, \qquad 1\le b\le B \tag{5.11}$$

$$1 - s_{(b)}e^{-\lambda_{(b)}(s_{(b)}-1)/2\mu} \ge \kappa_b, \qquad 1\le b\le B \tag{5.12}$$

In the PUBLISHER PROBLEM, each file is in one and only one bundle (5.8), the number of bundles is bounded by its maximum (5.9), the size and request rate for each bundle are defined in (5.10) and (5.11), and the self-sustainability of bundle b is lower bounded by κ_b (5.12).

Proposition 5.2.1. *The* PUBLISHER PROBLEM *is NP hard.*

Proof: Let us consider the problem of deciding if the set of feasible solutions of the PUBLISHER PROBLEM is empty. Equivalently, the problem consists of deciding if the optimal download time is bounded (assuming that if the problem is not feasible, the download time is infinite). We refer to such decision problem as BUNDLE. Note that the problem of finding the optimal bundling, i.e. the PUBLISHER PROBLEM, must be harder than BUNDLE, since solving the optimization problem leads to a solution to BUNDLE.

We reduce PARTITION to BUNDLE. An instance of the PARTITION problem consists on a finite set A and size $z(a) \in \mathbb{Z}^+$ for each $a \in A$. The question is: is there a subset $A' \subseteq A$ such that $\sum_{a\in A'} z(a) = \sum_{a\in A-A'} z(a)$? This problem is known to be NP hard [37].

Given an instance $x = (A, (z(a)))$ of the partition problem we transform it into an instance $f(x) = (S, (\lambda_i), (s_i), M, (\kappa_b), \mu)$ of the BUNDLE problem as follows. Let $|S| = |A|$ and $\lambda_i = s_i = z(i)$ for all i. Let $\mu = 0.1$. Let $\overline{z} = \sum_i z(i)/2$. Set $M = 2$ and

$$\kappa_1 = \kappa_2 = 1 - \overline{z}e^{-\overline{z}(\overline{z}-1)/2\mu} \tag{5.13}$$

Then, $x \in$ PARTITION iff $f(x) \in$ BUNDLE.

($\Rightarrow$) If $x \in$ PARTITION there is a way to split the contents in subsets A' and $A - A'$ such that $\sum_{a\in A'} z(a) = \sum_{a\in A-A'} z(a)$.

$$\sum_{a\in A'} z(a) = \sum_{a\in A-A'} z(a) = \overline{z} = \underbrace{\sum_{i\in S} x_{i1}s_i}_{=s_{(1)}} = \underbrace{\sum_{i\in S} x_{i2}s_i}_{=s_{(2)}} \tag{5.14}$$

But $s_i = \lambda_i = z(i)$ for all i, hence $1 - s_{(b)}e^{-\lambda_{(b)}(s_{(b)}-1)/2\mu} = 1 - \overline{z}e^{-\overline{z}(\overline{z}-1)/2\mu}$ for $b = 1, 2$. Therefore the feasible set of solutions to the BUNDLE problem is non empty.

($\Leftarrow$) If $f(x) \in$ BUNDLE then the set of feasible solutions is non empty. Note that, since $\mu = 0.1$ and $\lambda_{(b)}, s_{(b)} \in \mathbb{Z}$, then $1 - s_{(b)}e^{-\lambda_{(b)}(s_{(b)}-1)/2\mu}$ is strictly increasing in both $\lambda_{(b)}$ and $s_{(b)}$. Hence, the only way to satisfy the constraints imposed by $f(x)$,

$$1 - s_{(1)}e^{-\lambda_{(1)}(s_{(1)}-1)/2\mu} \geq 1 - \overline{z}e^{-\overline{z}(\overline{z}-1)/2\mu} \tag{5.15}$$

$$1 - s_{(2)}e^{-\lambda_{(2)}(s_{(2)}-1)/2\mu} \geq 1 - \overline{z}e^{-\overline{z}(\overline{z}-1)/2\mu} \tag{5.16}$$

is to divide the contents into two subsets, each with equal sum of file sizes. Therefore, $x \in$ PARTITION. $\square$

5.2.2 Reservation Download Times Yield Greedy Solution

In the economics literature, *reservation prices* are used to express what the customer is willing to pay for a product. The consumer surplus is the difference between the reservation price and what he actually has to pay. To illustrate the potential of bundling in this context, we borrow an example from [35, page 13]. Assume that a restaurant offers wine and pizza. There are four consumers with the reservation prices given in Table 5.1.

Consumer	Price Willing To Pay For Wine	Price Willing To Pay For Pizza	Price Willing to Pay For Bundle
1	9.0	1.5	10.5
2	8.0	5.0	13.0
3	4.5	8.5	13.0
4	2.5	9.0	11.5

Table 5.1. Individual reservation prices for wine, pizza and bundle.

Note the negative correlation between the reservation prices for wine and pizza. If separate pricing is used, the best that the firm providing these goods can do is to

set the price of the wine as 8.0, the price of the pizza as 8.5 (see Table 5.2), and get a revenue of $8 \times 2 + 8.5 \times 2 = 33$, since two units of each product would be sold. Now consider pure bundling: if the price of the bundle is set as 10.5, all four consumers would buy the bundle, and the revenue is now 42. This is 27.3 percent higher than the revenue with separate pricing.

Wine			Pizza		
Price	# of Consumers	Revenue	Price	# of Consumers	Revenue
9.0	1	9.0	1.5	4	6.0
8.0	**2**	**16.0**	5.0	3	15.0
4.5	3	13.5	**8.5**	**2**	**17.0**
2.5	4	10.0	9.0	1	9.0

Table 5.2. Revenue as a function of prices of wine and pizza. Bold entries correspond to optimal pricing strategies.

5.2.2.1 Reservation Download Times and Dynamic Bundles

Let us now consider the scenario where a user expresses his willingness for a file in terms of the download time that he is willing to tolerate in order to download it. Similar to the idea of reservation prices in the economics literature, we introduce the idea of a *reservation download time*. Assuming that a user has a download capacity of C Mbps, the reservation download time could be expressed, for instance, as a multiplicative factor $f > 1$ meaning that the user is willing to wait for fS/C seconds in order to download a file of size S Mb.

If reservation prices are used jointly with reservation download times, users would have flexibility to express both the relevance (through the reservation price) and the urgency at which they need a file (through the reservation download time).

We focus on a population of users that have large reservation download times. Their residence time in the system consists of a *service time* spent in downloading the requested content and a *spare time* which can be used to download content they did not originally sought. How should they use their spare time, i.e., what content

should they download after they complete downloading the requested file? To address this question, publishers need to trade-off between

1. "pushing" unpopular content to users, so that whenever any other user requests that content, it is available among peers;

2. sending to each user content that is of interest to that user, since the user may eventually decide to buy one of the contents that it owns.

Next, we show that a greedy strategy can be used by the server to decide on how to propose files to users. We reduce the problem to fractional knapsack. Assume that each file f is associated to a *value* v_f which takes into account the tradeoff presented in bullets 1 and 2 above. Then,

Proposition 5.2.2. *Consider a set of F files with sizes $s_1, s_2, \ldots, s_F$ and with values $v_1, \ldots, v_F$ to a given user U. Then, the problem of deciding which files to send to user U with reservation download time T can be solved using a greedy strategy where the i^{th} file to be sent is the one with the i^{th} largest ratio $v_k D_k/s_k$ where D_k is an estimate of the download rate at which file k can be downloaded by user U.*

Proof: Follows immediately from the discussion above and the fact that a greedy strategy is optimal to solve the fractional knapsack problem [21]. The fraction $v_k D_k/s_k$ corresponds to the utility per second of sending file k to user U. The knapsack capacity is T. □

Note that if users are not allowed to download fractions of files, the problem reduces to 0-1 knapsack, which is NP hard even though good approximation heuristics are known [37].

5.3 The Economic Model

We now introduce the economic model which will be used in the analysis of equilibria between peers and publishers to be presented in the upcoming sections. We

consider H publishers that offer C contents (files) to an infinite population of users. Each publisher, h, provides a set $\mathcal{N}_h$ of files, $|\mathcal{N}_h| = N_h$, through T_h torrents, in bundles of size $K_h = N_h/T_h$. Each file provided by a publisher is a member of exactly one of its bundles.

We do not restrict K_h to take integer values. If the optimal value $K^\star$ is not an integer, one way to implement it is to let the publisher's bundling strategy change over time. The optimal value $K^\star$ can be used to decide the fraction of time that $\lceil K^\star \rceil$ and $\lfloor K^\star \rfloor$ are adopted.

Each user accessing the system is interested in a single file. Users requesting content i arrive at rate $\Lambda^{(i)}$ but only a fraction $x^{(i)}$ of the users download the desired content.

Let $i_h(l)$ be the index of the l^{th} content provided by h, $1 \leq l \leq N_h$ and $i_h(l) \in C$. If two publishers p and q offer the same content, $i_p(m) = i_q(n)$ for some $1 \leq m \leq N_p$ and $1 \leq n \leq N_q$. Let $x_h^{(i)}$ be the fraction of users requesting content i that download it from publisher h. Publisher h experiences demand $\lambda_h^{(i_h(l))} = \Lambda^{(i_h(l))} x_h^{(i_h(l))}$ for its l^{th} content. Note that $\sum_{h=1}^{H} x_h^{(i)} = x^{(i)}$.

The vector $\underline{\lambda}_h = (\Lambda^{(i_h(1))} x_h^{(i_h(1))}, \ldots, \Lambda^{(i_h(N_h))} x_h^{(i_h(N_h))})$ is referred to as the *publisher h demand* vector. $\underline{\lambda} = (\underline{\lambda}_1, \ldots, \underline{\lambda}_h)$ is the *population demand* vector and $\underline{\Lambda} = (\Lambda^{(1)}, \ldots, \Lambda^{(C)})$ is the vector of *popularities*. The vector of *bundling strategies* is given by $\underline{K} = (K_1, \ldots, K_h)$.

Our key assumptions are the following:

- (A1) (reluctance to bundling) users are reluctant to download unrequested content;

- (A2) (minimize download time) users want to minimize the time to download requested content;

- (A3) (self sustaining swarms) users that download the same file or bundle from the same publisher help each other; if enough users collaborate with each other, the swarm sustains itself even in the absence of the publisher;
- (A4) (bandwidth costs) if the swarm is not able to sustain itself, the publisher needs to allocate a minimum bandwidth to each user;
- (A5) (rewards for serving users) the publisher receives a reward (which may be monetary) for each user served;
- (A6) (infinite capacity) the publisher bandwidth capacity is infinite.

5.3.1 Users' Utility

To capture assumptions (A1) and (A2), we model the cost function of a typical user as the sum of its download and monetary costs. For a user that receives content at rate μ bps, the download cost of a bundle of K_h files from publisher h, $D(K_h)$, is given by

$$D(K_h) = \alpha K_h/\mu \tag{5.17}$$

Here the term K_h/μ captures the average download time and α the download cost to a user in units of currency per second. α accounts for the reluctance of users to download bundles. The smaller the value of α the longer the users are willing to wait in order to download the desired files. Note that both μ and α can in principle vary across users, but here we assume a homogeneous population.

Publishers charge a flat price for each file that they provide, i.e., prices are independent of content. Publisher h sets price p_h for each of its contents and the *vector of prices* is $\underline{p} = (p_1, \ldots, p_h)$.

The user's utility function for the download of a bundle of K_h files (one of them being the file requested) from publisher h, $U_c(h)$, is

$$U_c(h) = V - D(K_h) - p_h \tag{5.18}$$

Here V is a random variable uniformly distributed in the range $[v_{min},\ v_{max}]$, $0 \leq v_{min} < v_{max} < \infty$, which characterizes valuations of files by users. The valuations are assumed to be independent across users.

Users download the desired content from h if (a) that publisher has the content, (b) $U_c(h) > 0$ and (c) $U_c(h) \geq U_c(h')$ for all $h' \neq h$. In case of a tie, demand is equally split among publishers. Clearly, the pricing and bundling strategies of a publisher impacts the demand (hence utility) experienced by the others. This naturally characterizes a game between publishers, who are coupled through users demands. To describe this game, we introduce the function $q_h^{(i_h(l))}(\cdot)$ which determines the demand experienced by publisher h for its l^{th} content as a function of all publishers' strategies

$$q_h^{(i_h(l))}(\underline{p}, \underline{K}) = \lambda_h^{(i_h(l))} = \Lambda^{(i_h(l))} x_h^{(i_h(l))} \tag{5.19}$$

We present two possible definitions of $q_h^{(i_h(l))}(\cdot)$ in Sections 5.4 and 5.5 (equations (5.24) and (5.47)).

Note that even though users download bundles, they only have access to the requested file. Hence, the download cost in (5.18) is proportional to K_h but the price charged by the publisher is for the single file.

5.3.2 Publishers' Utility

The aggregate costs incurred by publisher h to make all of its N_h/K_h bundles available must capture assumptions (A3) and (A4) and are given by $G_h(\underline{\lambda}_h, K_h)$. In some cases it may be convenient to make explicit the cost for publisher h to make

its j^{th} bundle available, $C_h^{(j)}(\underline{\lambda}_h, K_h)$. The definition of $C_h^{(j)}(\underline{\lambda}_h, K_h)$ is postponed to Section 5.4.

The publishers' revenue, $F_h(\underline{\lambda}_h, K_h)$, related to assumption (A5), is the standard one used in the economics literature:

$$F_h(\underline{\lambda}_h, K_h) = p_h \sum_{l=1}^{N_h} \Lambda^{(i_h(l))} x_h^{(i_h(l))} = p_h \sum_{l=1}^{N_h} \lambda_h^{(i_h(l))} \tag{5.20}$$

The publisher's utility, $U_h(p_h, K_h)$, is given by the difference between its revenue and its costs

$$\begin{aligned} U_h(p_h, K_h) &= p_h \sum_{l=1}^{N_h} \lambda_h^{(i_h(l))} - G_h(\underline{\lambda}_h, K_h) & (5.21) \\ &= p_h \sum_{l=1}^{N_h} \lambda_h^{(i_h(l))} - \sum_{j=1}^{T_h} C_h^{(j)}(\underline{\lambda}_h, K_h) & (5.22) \end{aligned}$$

5.3.3 Problem statement

The objective of publisher h is to maximize its utility,

$$\max_{K_h, p_h} U_h(p_h, K_h) = \max_{K_h, p_h} F_h(\underline{\lambda}_h, K_h) - G_h(\underline{\lambda}_h, K_h) \tag{5.23}$$

subject to the constraints posed by (5.19).

5.4 Monopoly

In this section we study the single publisher case. We are interested in understanding the impact of the publisher pricing strategy on the equilibrium, defined as the point where the publisher's profit is maximized. Henceforth, the index s is dropped for notational convenience.

Variable	Description
	Client Parameters
V	value of a file, uniform random variable [0,1]
$D(K)$	cost of download for the user
μ	download rate
	Publisher Parameters
$U(\lambda, K)$	utility (profit)
$C(\lambda, K)$	cost per bundle
$p(\lambda, K)$	price
K	number of files per bundle
N	number of files
	Population Parameters
Λ	total arrival rate per file
x	fraction of users that buy a file
$x_0 = 1 - (\alpha K/\mu)$	fraction of users that buy a file when $p = 0$
$\lambda = \Lambda x$	demand per file

Table 5.3. Table of notation.

5.4.1 Optimization Problem

Our goal is to pose a simplified version of the general problem (5.23). For that purpose, we assume that the popularities of all the files are the same ($\Lambda^{(i)} = \Lambda$ for all i).

A user downloads a bundle and purchases a file if $v - D(\lambda, K) - p > 0$ (equation (5.18)), where v is a realization of the random variable V. Letting $v_{min} = 0$ and $v_{max} = 1$, V is uniformly distributed in [0,1] (§5.3.1). Hence, the fraction x of the population for which $v > D(K) + p$ is $1 - g$ where $g = D(K) + p = (\alpha K/\mu) + p$. The demand as a function of the price is given by

$$x(p, K) = \frac{q(p, K)}{\Lambda} = [1 - \alpha K/\mu] - p = x_0 - p \tag{5.24}$$

where $q(\cdot)$ was introduced in (5.19). We have dropped all the superscripts present in (5.19) because the popularities of all the files are now assumed to be the same. Also, we explicitly account for the dependence of x on p and K. Note that when the

price is zero demand is maximized and $x_0 = 1 - \alpha K/\mu$. Alternatively, the price as a function of the demand is

$$p(x, K) = (1 - \alpha K/\mu) - x, \tag{5.25}$$

which implies that the maximum feasible bundle size is μ/α. Since demand x and price p are coupled through (5.24) and (5.25) we take K and x as control variables rather than K and p. Substituting (5.25) into (5.23) yields the following objective function for the publisher

$$\max_{K,x} U(x, K) \tag{5.26}$$

where (5.27)

$$U(x, K) = N\Lambda p(x, K)x - \frac{N}{K}C(\lambda, K) \tag{5.28}$$

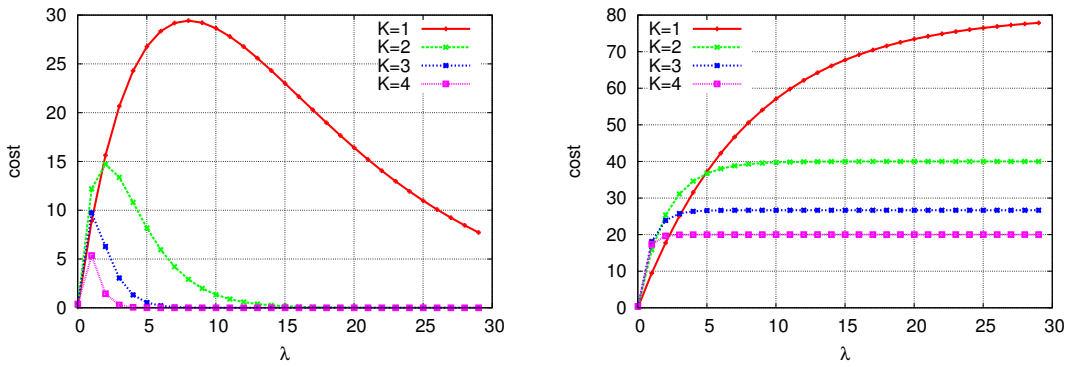
To solve the problem, we must determine the cost function $C(\lambda, K)$. To illustrate, in the following section we consider three possible functions, the first being the most realistic but also the most complex.

5.4.2 Cost Functions

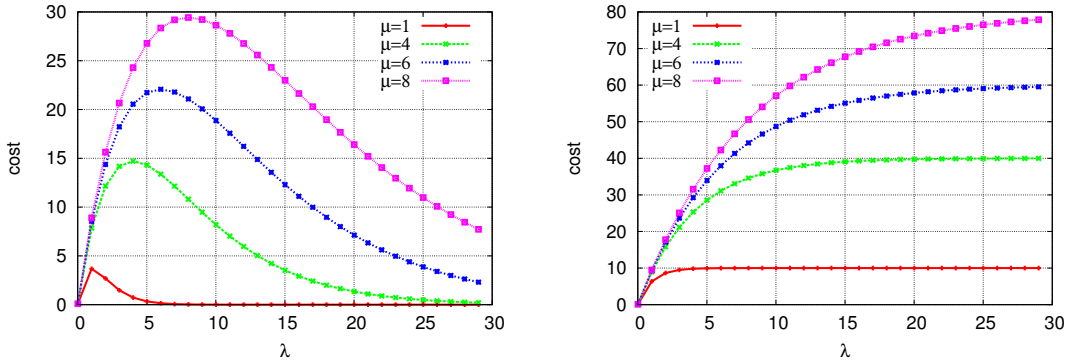
The goal of this section is to determine the cost function $C(\lambda, K)$. For this purpose we rely on a simple version of the model proposed and analyzed in Chapter 2. Assuming that peers arrive according to a Poisson process with rate $K\lambda$ and remain online for an exponentially distributed period of time equal to K/μ, and denoting by X the number of peers in the system,

$$P\{X = n\} = \frac{(K^2\lambda/\mu)^n e^{-K^2\lambda/\mu}}{n!} \tag{5.29}$$

We now briefly present the three cost functions studied in this chapter.



(a) active publisher (b) busy period



(c) active publisher (d) busy period

Figure 5.1. Cost for the publisher as a function of the arrival rate. (a) and (b) varying K [$\mu = 1$]; (c) and (d) varying μ [$K = 1$].

Active publisher: The publisher must be active when the number of peers in the system is small so as to provide missing blocks. In particular, assuming that the publisher is active only when there is one peer in the system, the probability of finding an *active publisher* is obtained by setting $n = 1$ in (5.29). The publisher's cost function per swarm is given by

$$C(\lambda, K) = cK^2 \frac{\lambda}{\mu} e^{-K^2 \lambda/\mu} \tag{5.30}$$

where c represents the cost per time unit incurred by an active publisher.

Busy period: The previous cost function is applicable if the server is active when there is *exactly one* customer in the system. For a second cost function, we assume that the publisher is active whenever there is *at least one* leecher in the system.

$$C(\lambda, K) = cP\{X > 0\} = c\Big(1 - e^{-K^2\lambda/\mu}\Big) \tag{5.31}$$

Note that this cost function is concave in λ.

Number of torrents: Finally, the last cost function we consider is where the per swarm cost to the publisher is constant

$$C(\lambda, K) = c \tag{5.32}$$

5.4.2.1 Numerical Evaluation

In Figure 5.1 we illustrate the first two cost functions described above. In all cases $c = \mu$, which means that the costs are proportional to the allocated bandwidth, which in turns equals that of a typical peer. We start with the configuration with 10 files ($N = 10$), 1 bundle ($K = 1$) and download rate $\mu = 8$ Mbps. Figure 5.1(a) shows the *active publisher* cost (§5.4.2) as a function of λ, as K varies from 1 to 4. For a fixed value of K, the costs for the publisher first increases but then sharply decreases, which captures the fact that when the mean population size is above a certain threshold the self-sustainability level is high. Note that the cost decreases as K increases: bundling leads to an increase in the arrival rate and in the residence time of peers, which maps into reduced publisher costs. The same rationale applies to Figure 5.1(b) where we consider the *busy period* cost function (§5.4.2). However, in Figure 5.1(b) costs are bounded away from zero as λ increases. Finally, in Figure 5.1(c) and (d) we observe that as μ increases server costs also increase. That's because the server contributes with a capacity equal to that of a typical peer ($c = \mu$).

5.4.3 Equilibrium for Fixed Bundling Strategy

We now establish the existence and uniqueness of the equilibrium in the monopoly case. Since in this section we consider a fixed value of K, we don't make explicit the dependency of any of the functions on K.

Definition: An equilibrium is a point x^* that maximizes the utility (5.26), i.e.,

$$x^* = \arg\max_{x\in[0,x_0]} U(x) = \arg\max_{x\in[0,x_0]} \{N\Lambda p(x)x - \frac{N}{K}C(x\Lambda)\} \tag{5.33}$$

In this section we assume that $C(x)$, as well as its first three derivatives, are continuous, $C(0) = 0$ and $C(x) > 0$ for $x > 0$. Moreover we assume that there exists at least one value $y \in [0, x_0]$ such that $U(y) > 0$ and that $U(x)$ is not constant in any subinterval of $[0, x_0]$.

Proposition 5.4.1. *The monopoly has an equilibrium with positive utility in* $(0, x_0)$.

Proof. $U(x)$ is continuous in the interval $[0, x_0]$, so it has a global maximum x^*. At the global maximum x^*, the utility is positive ($U(x^*) > 0$). As a consequence x^* belongs to $(0, x_0)$ because $U(0) = 0$ and $U(x_0) < 0$. □

The *publisher's revenue*, $F(x)$, is given by (5.20), $F(x) = N\Lambda p(x)x = N\Lambda[(1 - \alpha K/\mu) - x]x$ and the *publisher's marginal revenue* is $\frac{dF(x)}{dx} = N\Lambda[-2x + 1 - \alpha\frac{K}{\mu}]$. The *publisher's cost per swarm*, $C(x\Lambda)$, is given by one of the equations (5.30)-(5.32), the *publisher's cost* is $\frac{N}{K}C(x\Lambda)$ and the *publisher's marginal cost* is $\frac{N}{K}\frac{dC(x\Lambda)}{dx}$.

We observe that a necessary condition for y to be an equilibrium is that $U'(y) = 0$ since the equilibrium falls inside $[0, x_0]$. This means that at the equilibrium the marginal revenue has to equal the marginal cost:

$$\left.\frac{dF(x)}{dx}\right|_{x=x^*} = N\Lambda\left[-2x^\star + 1 - \alpha\frac{K}{\mu}\right] = \left.\frac{dC(x\Lambda)}{dx}\frac{N}{K}\right|_{x=x^*} \tag{5.34}$$

Among the roots of $U(x) = 0$ there are local minima and local maxima that are not equilibria. In what follows we provide sufficient conditions for the existence of a single local maximum and then of a unique equilibrium[1].

[1]Note that even in presence of multiple local maxima, there can be a unique global maximum point and so a unique equilibrium.

Proposition 5.4.2. *If $U'(x) = 0$ has at most two roots in $[0, x_0]$ then there is a unique equilibrium.*

Proof. If $U'(x) = 0$ has exactly one root within $[0, x_0]$ then this root is the equilibrium (and the unique maximum). If $U'(x) = 0$ has exactly two roots within $[0, x_0]$, they cannot be both local maxima points of $U(x)$. For the sake of contradiction, assume that both roots are local maxima. Then, there would have to exist a third root that "locally" minimizes $U(x)$ within $[0, x_0]$. $U'(x) = 0$ would have to have three roots within $[0, x_0]$ (contradiction). So, if $U'(x) = 0$ has exactly two roots within $[0, x_0]$, one of them is a maximum and the other is a minimum. □

Proposition 5.4.3. *If the publisher marginal cost function $\frac{N}{K}C'(\Lambda x)$ is convex in x then there is a unique equilibrium.*

Proof. The publisher marginal cost function $\frac{N}{K}C'(\Lambda x)$ being convex in x corresponds to $C'''(\Lambda x) \geq 0$. Since $U'''(x) = -\frac{N}{K}C'''(\Lambda x)$ it follows that $U'''(x) \leq 0$. Hence $U''(x)$ is non increasing and $U'(x)$ can change sign at most twice. As we assumed that $U(x)$ is not constant in any interval, $U'(x) = 0$ has at most two roots and the result follows from Proposition 5.4.2. □

We observe that the hypothesis of Proposition 5.4.3 is satisfied by the cost function (5.31), as proven in Appendix D.1, but not by function (5.30) which increases and then decreases. For this case the following result can be applied (hypotheses check for the function (5.30) in Appendix D.1).

Proposition 5.4.4. *If the publisher cost $\frac{N}{K}C(\Lambda x)$ is concave in $[0, \overline{x}]$ and convex in $(\overline{x}, x_0]$ and the corresponding marginal cost $\frac{N}{K}C'(\Lambda x)$ is convex in $[0, \overline{x}]$ then there is a unique equilibrium.*

Proof. Due to Proposition 5.4.3, $U'(x) = 0$ has at most two roots in $[0, \overline{x}]$. Cost convexity in $(\overline{x}, x_0]$ (i.e., marginal cost being non decreasing in the interval $(\overline{x}, x_0]$)

and the fact that the utility cannot be constant in any interval implies that $U'(x) = 0$ has at most one root in $(\overline{x}, x_0]$.

We now prove that if $U'(x) = 0$ has two roots in $[0, \overline{x}]$ it has no roots in $(\overline{x}, x_0]$. By assumption, $C'(\Lambda x)$ is decreasing and convex in $[0, \overline{x}]$. Also, $F'(x) = N\Lambda[-2x + 1 - \alpha\frac{K}{\mu}]$. Therefore, if $U'(x) = F'(x) - C'(\Lambda x) = 0$ has two roots in $[0, \overline{x}]$ it follows that $C'(\Lambda\overline{x}) > F'(\overline{x})$. Since in $(\overline{x}, x_0]$ the marginal cost is non decreasing, $C'(\Lambda x) \geq C'(\Lambda\overline{x})$ $\forall x \in (\overline{x}, x_0]$. The marginal revenue $F'(x)$ is always decreasing. Therefore, $U'(x) = 0$ has no roots in $(\overline{x}, x_0]$.

Note that if $U'(x) = 0$ has at most one root in $[0, \overline{x}]$ it could have another root in $(\overline{x}, x_0]$.

It has been shown that $U'(x) = 0$ has at most two roots in $[0, x_0]$. The rest of the proof then follows from Proposition 5.4.2. □

5.4.4 The Impact of Bundling

We now study the impact of bundling on the publisher's utility in equilibrium. The three cost functions introduced in §5.4.2 are considered.

5.4.4.1 Active Publisher

Given the active publisher cost function (5.30),

$$\frac{\partial}{\partial x}F(x,K)\bigg|_{x=x^\star} = N\Lambda\Big[-x + [(1-\alpha K/\mu) - x]\Big]\Big|_{x=x^\star} \tag{5.35}$$

and

$$\frac{\partial}{\partial x}\frac{N}{K}C(x,K)\bigg|_{x=x^\star} = cNK\frac{\Lambda}{\mu}\Big[1 - \frac{xK^2\Lambda}{\mu}\Big]e^{-K^2x\Lambda/\mu}\bigg|_{x=x^\star} \tag{5.36}$$

Substituting the two equations above into (5.34) yields

$$\Big[-2x^\star + 1 - \alpha K/\mu\Big] - \frac{cK}{\mu}\Big[1 - \frac{x^\star K^2\Lambda}{\mu}\Big]e^{-K^2x^\star\Lambda/\mu} = 0 \tag{5.37}$$

The roots of (5.37), which correspond to candidate values for the demand in equilibrium, $x^\star$, can be obtained using the following fix point equation:

$$x^{(i+1)} = \frac{1}{2\mu}\left((\frac{K^2\Lambda x^{(i)}}{\mu} - 1)cKe^{-\frac{K^2 x^{(i)}\Lambda}{\mu}} + \mu - \alpha K\right) \tag{5.38}$$

Now we derive insights on how α affects the system equilibrium through a numerical experiment. We set the parameters of the model as follows: $\Lambda = 1$, $\mu = 1$, $c = 12$ and $N = 100$. We set α equal to 0.02, 0.01, 0.0002 and 2×10^{-7}. Table 5.4 displays the utility values presented in the table were normalized and are given by $U(x^\star, K) - 24$, where $U(x^\star, K)$ is obtained through (5.26).

The same trend is observed in all columns of Table 5.4. For small values of K, an increase in the bundling level leads to a significant reduction in the costs for the publisher and a corresponding utility increase. However, for larger values of K increasing the bundling level leads to a decrease in demand which negatively impacts the utility.

Table 5.4 also shows the equilibrium value of x. When K is small, the publisher sets high prices to cope with its costs, hence demand is small. As K increases, prices decrease and demand increases. Finally, increasing K further leads to a demand decrease due to the reluctance of users to download bundles.

For a fixed value of K, as α decreases we note an increase in the utility perceived by the publisher. This happens because a higher fraction of the population joins the system. The optimal bundle size for $\alpha = 0.02, 0.01, 0.0002$ and 2×10^{-7} is equal to 5, 5, 6, 7, respectively (bold face elements in Table 5.4). Finally, note that we haven't included in Table 5.4 the results for $K = 1, 2$ because the system admits no equilibrium in those cases. For the results presented in the previous section regarding the existence of the equilibrium, we assumed that there is at least one value of x in the range $[0, x_0]$ for which $U(x) > 0$. This condition is not satisfied for $K = 1, 2$ in this example.

α	0.02		0.001		0.0002		2×10^{-7}	
K	$x^\star$	$\overline{U(x^\star, K)}$	$x^\star$	$\overline{U(x^\star, K)}$	$x^\star$	$\overline{U(x^\star, K)}$	$x^\star$	$\overline{U(x^\star, K)}$
3	0.5007	-19.18514	0.502489	-18.80194	0.503847	-18.49608	0.504187	-18.41979
4	0.5321	-0.041375	0.533499	0.171757	0.534574	0.342649	0.534843	0.385383
5	**0.4963**	**0.490159**	**0.498821**	**0.738963**	0.500763	0.938880	0.501249	0.988931
6	0.4940	0.403533	0.497010	0.700839	**0.499410**	**0.939980**	0.500009	0.999885
7	0.4930	0.304900	0.496500	0.651225	0.499300	0.930049	**0.499999**	**0.999930**
8	0.4920	0.206400	0.496000	0.601600	0.499200	0.920064	0.499999	0.999920
9	0.4910	0.108100	0.495500	0.552025	0.499100	0.910081	0.499999	0.999910
10	0.4900	0.010000	0.495000	0.502500	0.499000	0.900100	0.499999	0.999900

Table 5.4. Equilibrium varying K and α (Active publisher cost).

5.4.4.2 Busy Period

When the cost function is given by the busy period, we can proceed as in the last section and solve the problem using a fix point algorithm. However, in this case we can express the equilibrium demand, $x^\star$, as a function of K in terms of the Lambert $W(\cdot)$ function. The Lambert $W(\cdot)$ function is defined as the inverse of $f(w) = we^w$. Since $f(w)$ is not injective $W(\cdot)$ is multivalued; nevertheless, $W(\cdot)$ takes real values only on two of its branches, referred to as $W_0(\cdot)$ and $W_{-1}(\cdot)$ [36].

The demand in equilibrium, $x^\star$, is defined implicitly as a function of K by substituting (5.31) into (5.34). First, we define the function $H(x, K)$ as follows

$$\begin{aligned} H(x,K) &= \frac{\partial}{\partial x}F(x,K) - \frac{\partial}{\partial x}\frac{N}{K}C(x,K) \\ H(x,K) &= N\Lambda[-2x + 1 - \alpha\frac{K}{\mu}] - \frac{N}{K}ce^{-K^2x\Lambda/\mu}K^2\frac{\Lambda}{\mu} \end{aligned}$$

Then, solving $H(x^\star, K) = 0$ for $x^\star$ yields

$$x^\star = \frac{\left(2\,W\left(-\frac{cK^3\Lambda e^{1/2\frac{\Lambda K^2(-\mu+\alpha\,K)}{\mu^2}}}{2\mu^2}\right)\frac{\mu^2}{K^2} + \Lambda\,\mu - \Lambda\,K\alpha\right)}{2\Lambda\mu} \tag{5.39}$$

We make three observations about this solution. First, the argument of the function W decreases if $\frac{2\mu}{3\alpha} \leq K \leq \frac{\mu}{\alpha}$. But $W_0(z)$ and $W_{-1}(z)$ take real values if and only

if $z \in [-1/e, +\infty)$ and $z \in [-1/e, 0)$, respectively [36, Lemma 2.10]. Therefore, once we find (using $W_0(z)$) a value $K' \geq \frac{2\mu}{3\alpha}$ for which $x^\star$ is an imaginary number we don't need to further search for solutions $K^\star > K'$.

Second, substituting $x^\star$ into (5.26) and taking the derivative of (5.26) with respect to K yield

$$\lim_{K \to 0} \frac{dU}{dK} = -\frac{\Lambda(\alpha + c)N}{2\mu} < 0 \tag{5.40}$$

Third, we can compute $\frac{dx^\star}{dK}$ using the implicit function theorem,

$$\frac{dx^\star}{dK} = -\frac{\partial H/\partial K}{\partial H/\partial x^\star} = \frac{(c\mu - 2\,\Lambda\,K^2 x^\star c)\, e^{-\frac{K^2 \Lambda\, x^\star}{\mu}} + \alpha\,\mu}{-2\,\mu^2 + cK^3 \Lambda\, e^{-\frac{K^2 \Lambda\, x^\star}{\mu}}} \tag{5.41}$$

which leads to

$$\lim_{K \to \infty} \frac{dx^\star}{dK} = -\frac{\alpha}{2\mu} < 0 \qquad \lim_{K \to \infty} \lim_{x^\star \to 0} \frac{dU}{dK} = -\infty \tag{5.42}$$

In all the numerical experiments we conducted we found at most three roots for $\frac{dU}{dK} = 0$. Based on the second observation above, if $\frac{dU}{dK} = 0$ has at most three roots in $[0, K'')$, $K'' \leq \mu/\alpha$, and $x^\star$ takes only real values in that interval then $U(x^\star, K)$ has at most one local maximum in $(0, K'')$.

5.4.4.3 Number of Torrents

Motivated by the cost function (5.31) we consider the case where $e^{-K^2 x \Lambda/\mu}$ is negligible (equation (5.32)). Therefore

$$\frac{\partial U}{\partial x} = \Lambda - \alpha \frac{K}{\mu}\Lambda - 2x\Lambda; \quad x^\star(K) = \frac{1 - \alpha\frac{K}{\mu}}{2}; \quad \frac{dx^\star}{dK} = \frac{-\alpha}{2\mu}$$

where the last expression agrees with (5.42). Also,

$$\frac{dU}{dK} = \frac{N\left(-\alpha\, x^\star \Lambda + cK^{-2}\,\mu\right)}{\mu} \tag{5.43}$$

$$\frac{d^2U}{dK^2} = \frac{N}{2}\left(\frac{\alpha^2}{\mu^2}\Lambda - \frac{4c}{K^3}\right); \qquad \frac{d^3U}{dK^3} \geq 0 \tag{5.44}$$

We conclude that, restricting to $K > 0$, $\frac{dU}{dK}$ is convex (note that $\frac{dU}{dK}$ is not defined for $K = 0$), $\lim_{K\to 0+} \frac{dU}{dK} = +\infty$ and $\lim_{K\to+\infty} \frac{dU}{dK} = +\infty$ (compare to (5.42)). The convexity of $\frac{dU}{dK}$ implies that U has at most two critical points and at most one local maximum in $(1, \mu/\alpha)$. To find the optimal value of K, $K^\star$, one may search for (at most two) roots of $\frac{dU}{dK}$ in the interval $(1, \mu/\alpha)$ and include them in set I. The optimal value of K is given by $K^\star = \arg\max_{K\in I\cup\{1,\mu/\alpha\}} U(x^\star(K), K)$.

5.5 Competition

We now consider the case where multiple publishers compete in the market. The publishers are coupled through the demand. The actions of one publisher affect the demand perceived by the others.

We assume publishers are price-takers. The market price for each file is $p > 0$. The only strategic decision is the bundling level. In addition, we consider the simplified scenario where each publisher has only two options: either to bundle all its files or not to bundle at all.

Users' utilities are given by (5.18),

$$U_c(h) = V - \alpha K/\mu - p \tag{5.45}$$

We assume that each user requests a single content. Recall that in Section 5.3.1 we defined V as uniformly distributed in the range $[v_{min}, v_{max}]$, $0 \leq v_{min} < v_{max} < \infty$. Here, we let $v_{min} = \alpha/\mu + p$ and $v_{max} = M + \alpha N/\mu + p$ where M is a constant, and N is the number of files offered by each publisher ($N_h = N$ for all h). Note that each user is willing to download the requested file and pay its market price. A fraction x of users tolerates downloading bundles,

$$x = 1 - \frac{\frac{\alpha N}{\mu} - \frac{\alpha}{\mu}}{M + \frac{\alpha N}{\mu} - \frac{\alpha}{\mu}} = \frac{M\mu}{M\mu + \alpha(N-1)} \tag{5.46}$$

If $N > 1$, for any $M > 0$ there is one single value of α which leads to demand x, $0 \leq x \leq 1$. The demand x decreases as the reluctance of users to bundles α increases.

Let $\mathcal{O}^{(i)}$ be the set of publishers that offer content i, $\mathcal{O}^{(i)} = \{h : i \in \mathcal{N}_h\}$. The sets of publishers that offer bundled and unbundled versions of content i are $\mathcal{B}^{(i)} = \{t \in \mathcal{O}^{(i)} : K_t = N\}$ and $\mathcal{N}^{(i)} = \mathcal{O}^{(i)} \setminus \mathcal{B}^{(i)}$, respectively. We now define $q(\cdot)$ (equation (5.19)). When multiple publishers offer content i, demand is equally split across those publishers that do not bundle. This happens because, if $s, t \in \mathcal{N}^{(i)}$ and $u \in \mathcal{B}^{(i)}$, then $K_s = K_t = 1$, $K_u = N$ and $U_c(s) = U_c(t) > U_c(u)$ (equation (5.18)). If all the publishers offering content i resort to bundling, demand splits equally among them since $U_c(s) = U_c(t)$ for all $s, t \in \mathcal{B}^{(i)}$. Therefore,

$$q_h^{(i)}(\underline{K}) = \begin{cases} \frac{\Lambda^{(i)}}{|\mathcal{N}^{(i)}|}, & h \in \mathcal{N}^{(i)} \\ \frac{x\Lambda^{(i)}}{|\mathcal{B}^{(i)}|}, & h \in \mathcal{B}^{(i)} \text{ and } \mathcal{N}^{(i)} = \emptyset \\ 0, & \text{otherwise} \end{cases} \tag{5.47}$$

where $q(\cdot)$ does not depend on p because all *publishers are price takers* and all users are willing to pay the market price.

The utility of publisher h is given by (5.21). We now focus on publisher h costs. If the average demand per swarm is smaller than a threshold τ, the publisher incurs an aggregate cost c for all swarms. Otherwise, the swarms provided by publisher h are self sustaining, and costs are zero (analogous to standard information goods in the economics literature).

$$G_h(\underline{\lambda}_h, K_h) = \begin{cases} c, & \sum_{l=1}^{N} \lambda^{i_h(l)} K_h / N < \tau \\ 0, & \text{otherwise} \end{cases} \tag{5.48}$$

	not bundle	bundle
not bundle	A, W	B, X
bundle	C, Y	D, Z

Table 5.5. General rewards for the bundling game. Cell entries R, C represent the revenues for the row and column players, respectively.

This cost function resembles the active server cost function presented in Section 5.4.2. However, the continuous function introduced in (5.30) for each swarm is now replaced by the above step function that captures costs aggregated across several swarms.

5.5.1 Duopoly

From here on we consider the special case where there are only two publishers in the market. Let $\mathcal{C}_1 = \{i : 1 \in \mathcal{O}^{(i)} \text{ and } 2 \notin \mathcal{O}^{(i)}\}$, $\mathcal{C}_2 = \{i : 2 \in \mathcal{O}^{(i)} \text{ and } 1 \notin \mathcal{O}^{(i)}\}$ and $\mathcal{C}_{1,2} = \{i : 1 \text{ and } 2 \in \mathcal{O}^{(i)}\}$ be the sets of contents offered solely by publisher 1, solely by publisher 2, and by both, respectively.

Each user seeks content that is provided either by publisher 1, publisher 2 or both. Requests arrive with rate $\Lambda_1 = \sum_{j \in \mathcal{C}_1} \Lambda^{(j)}$, $\Lambda_2 = \sum_{j \in \mathcal{C}_2} \Lambda^{(j)}$ and $\Lambda_B = \sum_{j \in \mathcal{C}_{1,2}} \Lambda^{(j)}$, respectively.

The normal form game is shown in Table 5.5. We list publisher 1's strategies as rows and publisher 2's as columns. For each of the four cells, we give a pair of payoffs to the two publishers (in units of currency per second), first to publisher 1 and then to publisher 2, obtained using equations (5.21), (5.47) and (5.48). Given that $K_h = N$ if all the contents are bundled and $K_h = 1$ otherwise,

- $A, W = [p(\Lambda_i + \Lambda_B/2) - c\mathbf{1}_{(\Lambda_i + \Lambda_B/2)/N < \tau}]^+$
- $B, Y = [p(\Lambda_i + \Lambda_B) - c\mathbf{1}_{(\Lambda_i + \Lambda_B)/N < \tau}]^+$
- $C, X = [p\Lambda_i x - c\mathbf{1}_{\Lambda_i x < \tau}]^+$

	not bundle	bundle
not bundle	$p(\Lambda_1 + \frac{\Lambda_B}{2}) - c$, $p(\Lambda_2 + \frac{\Lambda_B}{2}) - c$	$p(\Lambda_1 + \Lambda_B) - c$, $p(\Lambda_2 x) - c$
bundle	$p(\Lambda_1 x) - c$, $p(\Lambda_2 + \Lambda_B) - c$	$p(\Lambda_1 + \frac{\Lambda_B}{2})x$, $p(\Lambda_2 + \frac{\Lambda_B}{2})x$

(a)

	not bundle	bundle
not bundle	$4, 4$	$14, 0$
bundle	$0, 14$	$5, 5$

(b)

Table 5.6. Almost completely overlapping contents and the publisher's dilemma: (a) general game structure; (b) numerical example where $\Lambda_B = 20$, $\Lambda_i = 0$, $x = 0.5$, $c = 6$, $p = 1$, $N = 10$, $\tau = 2.1$.

- $D, Z = [p(\Lambda_i + \Lambda_B/2)x - c\mathbf{1}_{(\Lambda_i+\Lambda_B/2)x<\tau}]^+$

where $i = 1$ in cells A, B, C and D and $i = 2$ otherwise. $\mathbf{1}_C$ is an indicator variable equal to 1 if C is true and 0 otherwise. Note that all rewards are positive. Henceforth, this should be assumed even when the $^+$ symbol is omitted.

5.5.2 Almost completely overlapping contents

We now consider the case where $\Lambda_i \approx 0$ $(i = 1, 2)$ and $N \gg 0$. We assume that the following conditions hold:

- (C1) $(\Lambda_i + \Lambda_B)/N < \tau$: not bundling leads to publishing costs;
- (C2) $\Lambda_i x < \tau$: bundling but relying on a small population of users still leads to publishing costs;
- (C3) $(\Lambda_i + \frac{\Lambda_B}{2})x > \tau$: bundling and relying on a population of users with size greater than a critical threshold leads to negligible publishing costs.

Even though we write conditions (C2)-(C3) as a function of the variable x, they can be as well rewritten as a function of the parameter α (see comments following equation (5.46)). Given (C1)-(C3) the normal form game is depicted in Table 5.6(a).

For a numerical example, consider the following parameters: $\Lambda_B = 20$, $\Lambda_i = 0$, $x = 0.5$, $c = 6$, $p = 1$, $N = 10$, $\tau = 2.1$. The normal form game in this case is given in Table 5.6(b). Publishers are better off if both bundle. However, the Nash

equilibrium of this game is (not bundle, not bundle), which is not Pareto optimal. A Nash equilibrium which is not Pareto optimal is referred to as a social trap.

We refer to a game in which bundling consists of a dominant strategy for both publishers and where the resulting equilibrium (not bundle, not bundle) is not Pareto optimal as a *publisher's dilemma*. A publisher's dilemma happens when $B > D > A > C$ and $Y > Z > W > X$ (Table 5.5) and is an example of a prisoner's dilemma [83].

In what follows, we show that if in addition to (C1)-(C3) the arrival rate of requestors $\Lambda_B \gg 0$ and α, the reluctance to bundling is close to 0 (hence x is close to 1), then complete overlap ($\Lambda_i = 0, i = 1, 2$) favors a publisher's dilemma. For that purpose, we add a condition on the arrival rate of clients,

- (C4) $\frac{2c}{p} < \Lambda_B < \frac{2c}{p(1-x)}$

In particular, (C4) is valid for the example in Table 5.6.

Proposition 5.5.1. *If conditions (C1)-(C4) hold then there exists an $\Lambda_i^\star$ such that if $0 \leq \Lambda_i \leq \Lambda_i^\star$ $(i = 1, 2)$ the game in Table 5.6 is a publisher's dilemma.*

Proof. Under the assumptions of the proposition, the condition for a publisher's dilemma $C < A < D < B$ maps into $[p\Lambda_1 x - c]^+ < [p(\Lambda_1 + \frac{\Lambda_B}{2}) - c]^+ < p(\Lambda_1 + \frac{\Lambda_B}{2})x < [p(\Lambda_1 + \Lambda_B) - c]^+$. If $\Lambda_1^\star = \max(0, \Lambda_1^{\star\star})$, $\Lambda_1^{\star\star} = \min(\frac{c}{px}, \frac{c - p(1-x)\frac{\Lambda_B}{2}}{p(1-x)})$ and $0 \leq \Lambda_1 \leq \Lambda_1^\star$: (1) the first term $[p\Lambda_1 x - c]^+ = 0$ because $\Lambda_1 \leq \frac{c}{px}$; (2) the second term $[p(\Lambda_1 + \frac{\Lambda_B}{2}) - c]^+ > 0$ since (C4) implies $p\frac{\Lambda_B}{2} > c$; (3) the relation $[p(\Lambda_1 + \frac{\Lambda_B}{2}) - c]^+ < p(\Lambda_1 + \frac{\Lambda_B}{2})x$ is satisfied since $\Lambda_1 < \Lambda_1^{\star\star}$ and (C4) implies $0 < p\frac{\Lambda_B}{2} - c < p(\Lambda_B/2)x$ so that even if $\Lambda_1 = 0$ still the relation holds; (4) (C4) also leads to $0 < p(\Lambda_B/2)x < p\Lambda_B - c$ hence $p(\Lambda_1 + \frac{\Lambda_B}{2})x < [p(\Lambda_1 + \Lambda_B) - c]^+$.

The condition $X < W < Z < Y$ follows similarly. □

As a matter of fact, note that $\Lambda_i = 0$ favors (C1) and (C2) and $\Lambda_B \gg 0$ favors (C3). If $N \gg 0$, (C1) holds even when $\Lambda_B \gg 0$.

	not bundle	bundle
not bundle	0, 0	$p(\Lambda_1 + \Lambda_B)$, 0
bundle	0, $p(\Lambda_2 + \Lambda_B)$	0, 0

(a)

	not bundle	bundle
not bundle	0, 0	4, 0
bundle	0, 4	0, 0

(b)

Table 5.7. Partially overlapping contents and the bundle off game: (a) general game structure; (b) numerical example where $\Lambda_B = \Lambda_1 = \Lambda_2 = 10$, $x = 0$, $c = 16$, $p = 1$, $N = 6$, $\tau = 5$.

Key insight: If a publisher distinguishes itself by providing exclusive content ($\Lambda_i > 0$) it is less likely to end up in a social trap consisting of a non Pareto equilibrium where no publisher bundles.

5.5.3 Partially overlapping contents

We now consider the case where content is only partially overlapping and users are reluctant to download bundles ($x \approx 0$). We replace (C1) by the following two conditions and add condition (C5):

- (C1′) $(\Lambda_i + \frac{\Lambda_B}{2})/N < \tau$: not bundling leads to publishing costs if the population of requestors is small;
- ($\neg$C1) $(\Lambda_i + \Lambda_B)/N > \tau$: if the population of requestors is large, costs are zero;
- (C5) $c > p(\Lambda_1 + \Lambda_2 + \frac{\Lambda_B}{2})$: publishing costs, when incurred, are high.

Alternatively, (C1′) and ($\neg$C1) can be interpreted as conditions over the number of files offered by each publisher: the number of files (N) offered by each publisher falls in the range $(\Lambda_i + \frac{\Lambda_B}{2})/\tau < N < (\Lambda_i + \Lambda_B)/\tau$.

Conditions (C1′), ($\neg$C1) and (C5) lead to the game shown in Table 5.7(a). The reluctance of users to download bundles forces the publishers that bundle to go off the market (the revenue of publishers that bundle is zero). For $\Lambda_B = \Lambda_1 = \Lambda_2 = 10$, $x = 0$, $c = 16$, $p = 1$, $N = 6$, $\tau = 5$, we obtain the game shown in Table 5.7(b). This game has three pure equilibria: (not bundle, not bundle), (bundle, not bundle) and (not bundle, bundle).

We refer to a game in which (1) there are at least two pure strategy equilibria, e_1 and e_2; (2) each publisher prefers a different pure strategy equilibrium and (3) both e_1 and e_2 are Pareto optimal as a *bundle off game*.

Note that the bundle off game is an example of a degenerate hawk dove game [83]. The conditions for a bundle off game are (Table 5.5) $B > D \geq C \geq A$ and $Y > Z \geq X \geq W$. Next, we show that reluctance of users to download bundles favors a bundle off game:

Proposition 5.5.2. *If conditions [(C1′), (¬C1) and (C5)] hold then there is a threshold Δ such that if $x < \Delta$ the game in Table 5.5 is a bundle off game.*

Proof. If conditions (C1′), (¬C1) and (C5) are met and $x < \tau/(\Lambda_i + \frac{\Lambda_B}{2})$ then the game in Table 5.5 reduces to the one shown in Table 5.7(a), which is a bundle off game. □

Key insight: If users are reluctant to download bundles and the costs to provide unbundled files are high, we may expect that in the equilibrium only one publisher offers an unbundled version of the contents it has.

5.6 Related Work

In their seminal work on bundling, Adams and Yellen [2] showed that a firm that provides two products can take advantage of bundling by increasing the number of customers that buy its products. Since then, there is a growing literature on bundling (for a survey, see [35]). More recently, certain authors have also considered strategic bundling [99, 64]. However, none of these works considered bundling of files in a peer-to-peer network.

Salinger [80] was the first to point out that bundling is a demand smoother in the case of homogeneous demands. This is a consequence of the law of large numbers: if all buyers draw their value for the goods from the same probability distribution, average valuation converges to the mean as the number of goods increases. Therefore,

if the firm sets the price of the bundle to the mean times the number of elements in the bundle, virtually every customer will be willing to buy and profit is maximized. Later, Bakos and Brynjolfsson [9] reached the same conclusion in the context of information goods, i.e., goods that have almost zero cost to be replicated. This indicates that firms that cannot price discriminate may still use bundling to increase its sales. The same reasoning applies if demand is negatively correlated [68, Section 12.6].

Closely related to our work are those by Bakos and Brynjolfsson [10], Fay and Mackie-Mason [33] and Croson and Sainders [25]. They were the first to consider bundling of information goods under competition. What distinguishes our work from these is the fact that in a peer-to-peer system the replication costs depend on the number of users that possess the content. In addition, we take into account performance factors related to network metrics such as download time and availability which were not considered in previous work.

5.7 Discussion

Next, we discuss the assumptions adopted to yield a tractable model.

Homogeneous request rates and file sizes assumption: We assume all files have the same request rate and size. Although the general model presented in the beginning of this chapter allows for heterogeneous request rates and file sizes, the extension of the analytical results to this more general case is non-trivial.

Uniform price assumption: We assume that users pay a fixed price for each requested file. This is roughly the model adopted, for instance, by iTunes [7].

Price-takers assumption (in duopoly): In the duopoly, we assume publishers are price-takers. Future work consists of studying the joint pricing and bundling equilibrium.

5.8 Conclusion

We considered enterprises that rely on swarming systems to disseminate files and proposed a model to capture the tradeoffs in the choices of prices and bundling strategies. Using the proposed model, we showed conditions for existence and uniqueness of the equilibrium and how this equilibrium changes as a function of the bundling strategy. In the multi-firm case, we explained the effects of system parameters on the Nash equilibrium. Future work consists on analyzing the Bertrand-Nash equilibrium of the duopoly and using multi agent reinforcement learning to analyze the multi-firm case.

CHAPTER 6

CONCLUSION AND FUTURE WORK

In this thesis we have analyzed mechanisms, models and algorithms to leverage the use of swarming for the dissemination of a catalog of files. We have quantified how bundling impacts availability (Chapter 2), the dependence of peers on publishers (Chapter 3) and the loss of efficiency due to direct reciprocity and the corresponding download of unrequested content (Chapter 4). We have also studied algorithmic and economic aspects faced by publishers (Chapter 5). Although in this thesis we have posed and answered fundamental questions in the realm of peer-to-peer swarming systems for the dissemination of catalogs of files, the problems addressed here are far from exhausting all the questions related to that topic. In what follows, we point out some of the many avenues for future work.

6.1 Future Work

6.1.1 Publisher Capacity Allocation and Stability

The model presented in Chapter 3 is useful to estimate the dependence of peers on a publisher when the publisher allocates a fixed bandwidth to the swarm. If the bandwidth allocated by the publisher is too small, though, or if the publisher allocates its bandwidth dynamically, then the *last block problem*, as described by Hajek et al. [40], can occur.

The last block problem occurs when only one block is missing among peers. If the publisher provides this block to a peer, the peer is likely to complete its download and consequently depart the system immediately. That is because peers acquire all

blocks except the missing one very quickly. If all peers rely on the publisher in order to download their last block, the system goes into a client-server mode.

Question: *How does bundling affect the stability of the swarms?*

In addition, the bandwidth allocation strategies considered in the literature [69, 27] do not account for dynamic replication of content in the network. The joint bandwidth allocation-content placement problem is still open.

Question: *How to optimally solve the joint bandwidth allocation-content placement problem?*

6.1.2 The Nature of the Content

The decisions of how to combine contents and which content to send to each users depend on how contents relate one to another. In this thesis we have not taken into account the correlations among contents in order to issue recommendations or to bundle. Future work consists of taking such correlations, which can be captured, for instance, using a correlation matrix, into account.

Question: *How to bundle and issue recommendations to users based on content correlations as well as on system metrics, such as availability?*

6.1.3 Reciprocity and Barter

In Chapter 4 we pointed out that the loss of efficiency due to direct reciprocity in a special class of relay-less networks is bounded by a factor of two. We also discussed matchmaking and content recommendations as a means to improve loss of efficiency. In what follows, we describe possible future work in this field.

6.1.3.1 Loss of Efficiency Due To Direct Reciprocity

Question: *What is the (per-node) loss of efficiency due to direct reciprocity?*

The problem above in its more general setting is still open. A related problem concerns showing that the system loss of efficiency, measured as the collective number

of transmissions required in the direct reciprocity schedule divided by the number of transmissions in the indirect reciprocity schedule, is less than two.

Question: *What is the system loss of efficiency due to direct reciprocity?*

6.1.3.2 Content Value for Bartering

In Chapter 4 we proposed the use of a public board to assist users in deciding what and when to barter. The public board was used to break ties, and users were assumed to transmit one entire file every time they switched contents. In general, though, users might be able to exchange fractions of contents and the public board might be used to decide how to weight the amount to be sent of each content.

Question: *How to assist users in deciding what and when to barter?*

Users might be willing to download content that they did not initially sought, for bartering purposes, if such content turns out to be interesting to them or related to contents that are of their interest. Therefore, recommendation systems that take into account both users interests as well as system performance are needed.

6.1.4 The Universal Swarm and Wireless Swarming

In its full generality, the main question addressed in this thesis is how to take advantage of collaborations across users interested in different files so as to satisfy their demands, in a simple, efficient and incentive compatible way. This general idea of allowing collaborations among users of different swarms has been referred to as universal swarming [93, 106, 107]. The universal swarm might involve both wired and wireless users, and the contact opportunities among the wireless users might be dictated by their mobility patters. Although there has been recent progress on dissemination and replication strategies for peer-to-peer wireless systems [45, 3, 18, 75], the following general question is open.

Question: *How to leverage swarming systems to disseminate content to users that are intermittently connected in space (due to mobility) and time (due to their online patterns)?*

APPENDIX A

CONTENT AVAILABILITY AND BUNDLING

Background

Our results rely on those reported by Browne and Steele [15] on the busy period of an M/G/∞ queue where the customer initiating the busy period has an exceptional residence time.

Let customers arrive according to a Poisson process with rate β. If we allow customers initiating a busy period to draw their residence times from a distribution $H(\cdot)$ with Laplace transform $h(\cdot)$ and mean θ while all other customers draw their residence times from a distribution $G(\cdot)$, the expected busy period length is given by

$$E[B] = \theta + \sum_{i=1}^{\infty} \frac{\beta^i}{i!} \int_0^{\infty} (1 - H(x)) \Big[\int_x^{\infty} \Big(1 - G(u)\Big) du \Big]^i dx \tag{A.1}$$

When $G(x) = 1 - e^{-x/\alpha}$, i.e., all customers except the first draw their service times from an exponential distribution with mean α, the equation above reduces to

$$E[B] = \theta + \sum_{i=1}^{\infty} \frac{(\beta\alpha)^i \alpha[1 - h(i/\alpha)]}{i!i} \tag{A.2}$$

If the customer initiating a busy period also draws its service time from an exponential distribution,

$$E[B] = \theta + \alpha\theta \sum_{i=1}^{\infty} \frac{(\beta\alpha)^i}{i!(\alpha + i\theta)} \tag{A.3}$$

Finally, if $\theta = \alpha$,

$$E[B] = (e^{\beta\alpha} - 1)/\beta \tag{A.4}$$

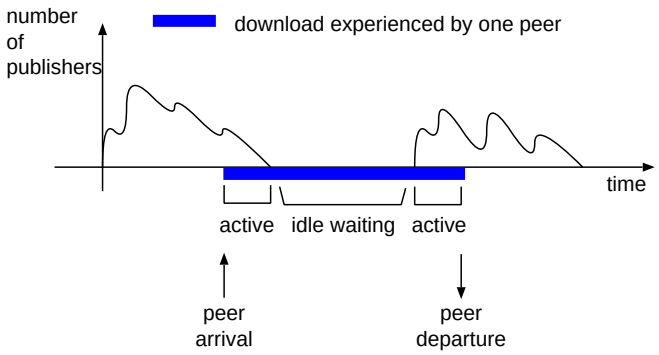


Figure A.1. Publishers only availability. Content download spans multiple busy periods.

A.1 Proof of Theorem 2.3.1

Accounting for publishers only corresponds to setting the threshold coverage to infinity. A threshold coverage of infinity yields a lower bound on availability and an upper bound on average download time since content is assumed to be available only if there is at least one publisher in the system.

A.1.1 Availability

The availability accounting for publishers only is given by the fraction of time in which there is at least one publisher in the system,

$$\frac{(e^{ru}-1)/r}{1/r+(e^{ru}-1)/r} \tag{A.5}$$

A.1.2 Download Time

In this section we quantify the mean download time assuming peers are patient and content is available only when publishers are online. Note that the download time may span multiple busy periods (see Figure A.1).

Let I be the initial number of online publishers when a peer starts its download. We consider a Markov Chain with state space $\{X_i\}(i \geq -1)$. State X_i $(i \geq 1)$

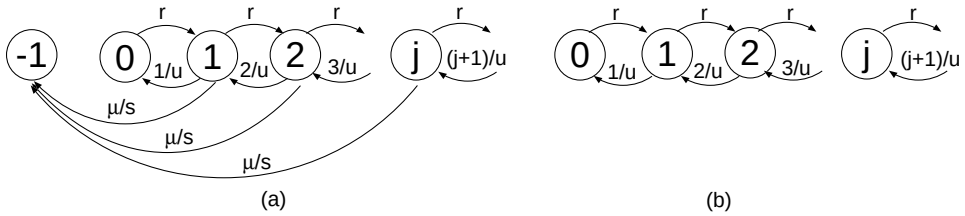


Figure A.2. (a) Markov chain of original process; (b) Markov chain of modified process.

corresponds to a system in which there are i publishers online and the download is in progress. State X_0 corresponds to no publishers online and the user is idle waiting, and state X_{-1} corresponds to the download completion. The transition rates are as follows

a) $X_i \xrightarrow{r} X_{i+1}$, $i \geq 0$

b) $X_i \xrightarrow{i/u} X_{i-1}$, $i \geq 1$

c) $X_i \xrightarrow{\mu/s} X_{-1}$, $i \geq 1$

Transitions a) and b) correspond to arrivals and departures of publishers. Transition c) corresponds to a download completion.

Let the mean download time, conditioned on j publishers being online in the system when the peer arrives, be $E[T|I = j]$. The download time, C, has mean $E[T]$ equal to

$$E[T] = \sum_{j=0}^{\infty} \frac{e^{-\rho}\rho^j}{j!} E[T|I = j] \tag{A.6}$$

where $\rho = ru$.

Our results rely on those derived by Crescenzo et al. [24] on a birth-death system with catastrophes. Transitions a), b) and c) correspond to the birth, death and catastrophes in [24, Section 3], respectively.

To present our results it is useful to introduce the system consisting only of transitions a) and b), which corresponds to an $M/M/\infty$ queue. We refer to the state space of the modified system as $\{\widehat{X}_i\}(i \geq 0)$.

a) $\widehat{X}_i \xrightarrow{r} \widehat{X}_{i+1}$, $i \geq 0$

b) $\widehat{X}_i \xrightarrow{i/u} \widehat{X}_{i-1}$, $i \geq 1$

It follows from [23, eq. (41)] that [1]

$$E[T|I=j] = \frac{1}{\mu/s}\left(1 + \frac{j!}{\rho^j}\frac{q_j}{1-q_0}\right) \tag{A.7}$$

where q_j, $j \geq 0$, is μ/s times the Laplace transform of the probability of there being j publishers in the modified system at time t given that initially there are none, evaluated at point μ/s.

Computing q_0

We first compute q_0. The number of customers in an $M/M/\infty$ queue, n, at time t, given that the queue is initially empty is given by a Poisson random variable with rate $\rho(1-e^{-t/u})$ (see [24, Section A4] and [22, 54]). Therefore, the probability of the queue being empty at time t is

$$e^{-\rho(1-e^{-t/u})} \tag{A.8}$$

The Laplace transform of (A.8) is

$$\widehat{\pi}_0(\varsigma) = \int_0^\infty e^{-\varsigma t - \rho(1-e^{-t/u})} dt \tag{A.9}$$

Using [38, eq. 3.331.1] and the series representation of the gamma function [38, eq. 8.354], together with (A.9), yields

$$q_0 = (\mu/s)\widehat{\pi}_0(\mu/s) = e^{-\rho}(\mu/s)u\sum_{k=0}^{\infty}\frac{\rho^k}{k!}\frac{1}{k+u\mu/s} \tag{A.10}$$

[1] Note that in [24, Proposition 1] there is a typo: the term $E[T_{j,r}]$ is missing in the right hand side of eq. (30).

Computing q_j

Arguments similar to those used to derive the Laplace transform of the probability of there being zero users in the system at time t can be used to derive the Laplace transform of the probability of there being j users in the system at time t, $j \geq 0$. For details, refer to [24, Section A4].

$$q_j = \frac{\mu}{s}\frac{\rho^j}{j!}e^{-\rho}\frac{\mu/s}{1/u}B\left(j+1, \frac{\mu/s}{1/u}\right)\Phi\left(\frac{\mu/s}{1/u}, \frac{\mu/s}{1/u}+j+1; \rho\right) \tag{A.11}$$

where B is the Beta function and Φ the Kummer function (see [24, Section A4]).

$E[T|L=j]$ is obtained by substituting (A.10) and (A.11) into (A.7). Finally, $E[T|L=j]$ together with (A.6) yield $E[T]$,

$$\boxed{E[T] = \frac{1}{\mu/s}\left(1 + \sum_{j=0}^{\infty}\frac{q_j}{e^{\rho} - (\mu/s)u\sum_{k=0}^{\infty}\frac{\rho^k}{k!}\frac{1}{k+u\mu/s}}\right)} \tag{A.12}$$

A.2 Proof of Theorem 2.3.2

We now consider a threshold coverage of zero. Note that whereas it is straightforward to verify that an infinite threshold coverage yields a lower bound on availability (see Definition 2.3.1), it is not as immediate to show that a threshold coverage of zero yields a corresponding upper bound. Although increasing the threshold coverage decreases the mean duration of the first busy period, it also increases the number of extant peers that request for service in the subsequent busy periods. In the next section, we show that the decrease in the mean duration of the first busy period outweights the increase of the mean duration of the subsequent periods, i.e., we show that a threshold coverage of zero yields an upper bound on availability. This upper bound, in turn, yields a lower bound on the mean download time.

A.2.1 Availability

When the threshold coverage is zero, the availability accounting for publishers and peers is given by the fraction of time in which there is at least one peer in the system. For simplicity, we assume that the residence time of publishers is drawn from an exponential distribution with mean s/μ, same as peers (otherwise, results from [15] are applicable). Therefore, the availability, $\overline{A}$, is

$$\overline{A} = \frac{(e^{(r+\lambda)(s/\mu)} - 1)/(r+\lambda)}{1/r + (e^{(r+\lambda)(s/\mu)} - 1)/(r+\lambda)} \tag{A.13}$$

We now show that the above expression indeed constitutes an upper bound on the availability. If the threshold coverage is greater than zero, peers might get *blocked while downloading a content* when the number of online peers reaches the threshold coverage. In this case, peers need to wait for a publisher to return in order to resume their downloads. Next, we show that *in the system with blocking (i.e., with threshold coverage greater than zero) peers experience decreased availability (i.e., decreased average duration of busy periods) as compared to peers in the system without blocking.*

We start from the system without blocking (threshold coverage of zero). Consider the first instant of time t at which there are no publishers online and the number of peers reaches the value $m > 0$. All extant peers and peers that arrive after t, and before a publisher returns, are referred to as exceptional peers. The time until the system becomes empty after t is referred to as the residual busy period.

Now consider the system with blocking (threshold coverage $m > 0$). Exceptional peers remain idle waiting from t until a publisher returns. When the publisher returns, they immediately start to receive service. This constitutes a new busy period. The first customer of such a busy period is a virtual customer, whose residence time equals the maximum of the residence times of all exceptional peers and the only publisher.

The mean residence time of the virtual customer is smaller than the mean residual busy period (see Figure A.3). That is because arrivals which occur far apart from

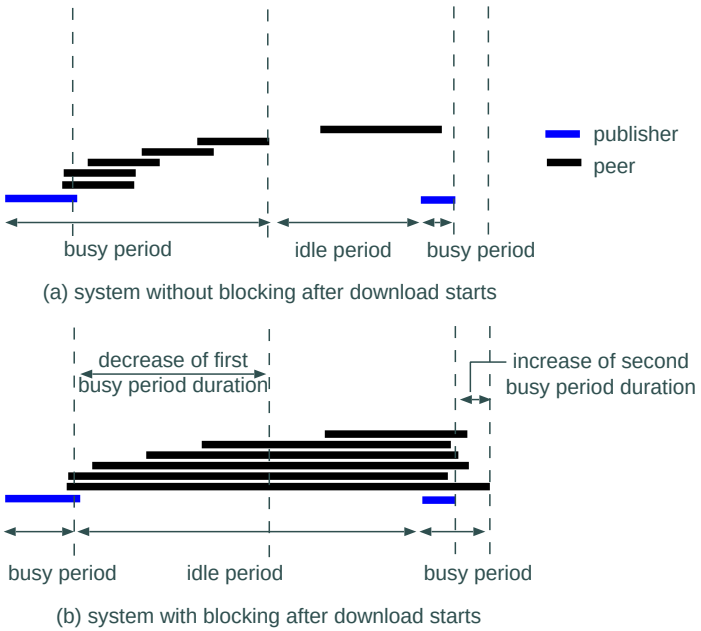


Figure A.3. System with blocking yields a net decrease in mean busy period duration and a corresponding decrease in availability.

each other (as in a residual busy period) yield increased makespan compared to that obtained if all arrivals occur at the same instant (as in a virtual customer). Therefore, the system without blocking (threshold coverage of zero and residual busy periods) yields busy periods that have mean duration larger than the system without blocking (threshold coverage $m > 0$ and virtual customers).

A.2.2 Download Time

The mean download time is given by,

$$(1 - A)(1/r) + \text{mean active download time}$$

Note that the mean active download time may span multiple busy periods. However, it is lower bounded by s/μ, therefore the mean download time is lower bounded by

$$(1 - \overline{A})(1/r) + s/\mu$$

Other Proofs

Throughout the proofs, let $\hat{\lambda} = \max\{\lambda_k\}$, $\check{\lambda} = \min\{\lambda_k\}$, $\hat{s} = \max\{s_k\}$, $\check{s} = \min\{s_k\}$.

Derivation of equation (2.10)

Proof. We use equation (A.1) to obtain (2.10). Let the download time of customers that arrive during the busy period be given by

$$X = \begin{cases} X_1, & \text{with probability } q_1 \\ X_2, & \text{with probability } q_2 = 1 - q_1 \end{cases}$$

where $E[X_i] = \alpha_i$. Then,

$$G(u) = 1 - q_1 e^{-\frac{1}{\alpha_1}u} - q_2 e^{-\frac{1}{\alpha_2}u} \tag{A.14}$$

and

$$E[B] = \theta + \sum_{i=1}^{\infty} \frac{\beta^i}{i!} \int_0^{\infty} I(z,i)dz \tag{A.15}$$

where

$$I(z,i) = (1 - H(z)) \sum_{j=0}^{i} \binom{i}{j} \Big[\Big(q_1 \frac{e^{-\frac{1}{\alpha_1}z}}{\frac{1}{\alpha_1}} \Big)^j \Big(q_2 \frac{e^{-\frac{1}{\alpha_2}z}}{\frac{1}{\alpha_2}} \Big)^{i-j} \Big] \tag{A.16}$$

Substituting (A.16) into (A.15) and using integration by parts yields

$$E[B] = \theta + \sum_{i=0}^{\infty} \frac{\beta^i}{i!} \sum_{j=0}^{i} \binom{i}{j} \frac{q_1^j}{\frac{1}{\alpha_1^j}} \frac{q_2^{i-j}}{\frac{1}{\alpha_2^{i-j}}} \Big[\frac{1 - h(\frac{j}{\alpha_1} + \frac{i-j}{\alpha_2})}{\frac{j}{\alpha_1} + \frac{i-j}{\alpha_2}} \Big] \tag{A.17}$$

and if customers initiating a busy period draw their service times from an exponential distribution with mean θ, $h(s) = \theta^{-1}/(\theta^{-1} + s)$, which yields (2.10). □

Proof of Lemma 2.3.1

Proof. Since all the terms in B_k, $k = 1, \ldots, K$, are lower bounded and upper bounded by terms that do not depend on K, $E[B_k]$ is bounded hence $E[B_k] = \Theta(1)$.

To show that $\log E[B^{(b)}] = \Theta(K^2)$ we consider a special process, with busy period duration $B^\star$, where the following conditions hold,

- during the busy period, customers arrive with rate β, where $\beta = R + \sum_{k=1}^{K} \lambda_k$,
- the residence times of all customers, excluding the first, arriving in a busy period are drawn from an exponentially distributed random variable with mean α,
- the residence time of the first customer in a busy period is drawn from an exponentially distributed random variable with mean θ,
- α, β and θ are upper bounded by

$$\alpha \leq K\hat{s}/\mu \qquad \beta \leq K\hat{\lambda} + R \qquad \theta \leq U + K^2\hat{s}\hat{\lambda}/(R\mu) \tag{A.18}$$

- α, β and θ are lower bounded by

$$\alpha \geq K\check{s}/\mu \qquad \beta \geq K\check{\lambda} \qquad \theta \geq U \tag{A.19}$$

The average busy period, $E[B^\star]$, of the special process is given by (A.3).

First, we show that $\log E[B^\star] = O(K^2)$. Since $1/(\alpha + i\theta) \leq 1/\alpha$ it follows from (A.3) that

$$E[B^\star] \leq \theta + \alpha\theta \sum_{i=1}^{\infty} \frac{(\beta\alpha)^i}{i!\alpha} = \theta\alpha \sum_{i=0}^{\infty} \frac{(\beta\alpha)^i}{i!\alpha} = \theta\alpha e^{\alpha\beta} \tag{A.20}$$

Given that $\alpha\beta \leq K^2\hat{s}\hat{\lambda}/\mu + o(K^2)$,

$$\lim_{K\to\infty} \frac{\log E[B^\star]}{K^2} \leq \lim_{K\to\infty} \frac{\log(\theta\alpha e^{K^2\hat{s}\hat{\lambda}/\mu})}{K^2} < \infty \tag{A.21}$$

implying that $E[B^\star] = O(K^2)$.

Next, we show that $\log E[B^\star] = \Omega(K^2)$. First, note that

$$\frac{1}{\alpha + i\theta} \geq \frac{1}{(i+1)\max(\alpha,\theta)} \tag{A.22}$$

Putting (A.22) and (A.3) together yields

$$\begin{aligned} E[B^\star] &\geq \theta + \alpha\theta \sum_{i=1}^{K} \frac{(\Lambda\alpha)^i}{i!(i+1)\max(\alpha,\theta)} && \text{(A.23)} \\ &\geq \frac{\theta}{\Lambda} \sum_{i=1}^{K} \frac{(\Lambda\alpha)^{i+1}}{(i+1)!\max(\alpha,\theta)} && \text{(A.24)} \\ &= \frac{\theta}{\beta} \frac{1}{\max(\alpha,\theta)} \Big(e^{\beta\alpha} - 1 - \beta\alpha \Big) && \text{(A.25)} \end{aligned}$$

Since $\alpha\beta \geq K^2 \breve{s}\breve{\lambda}/\mu$,

$$\lim_{K\to\infty} \frac{K^2}{\log E[B^\star]} \leq \lim_{K\to\infty} \frac{K^2}{\log\left(\theta \frac{\exp(K^2\hat{s}\hat{\lambda}/\mu) - 1 - (K^2\breve{s}\breve{\lambda})/\mu}{K^2\hat{s}\hat{\lambda}/\mu + K\hat{\lambda}\theta + o(K^2)}\right)} < \infty \tag{A.26}$$

Therefore, $\log E[B^\star] = \Theta(K^2)$.

To extend the result above to the parameterization of $E[B^{(b)}]$ made in Section 2.3.3.1 we proceed as follows. For the upper bound, $\log E[B^{(b)}] = O(K^2)$, consider a modified process in which the residence times of all customers arriving during a busy period are drawn from an exponential random variable with mean $\alpha = K\hat{s}$. Denote the busy period of the modified process by $\widehat{B}$. Noting that conditions (A.18) hold it follows from (A.20)-(A.21) that $\log E[\widehat{B}] = O(K^2)$. Since $\lim_{K\to\infty} E[\widehat{B}] \geq \lim_{K\to\infty} E[B^{(b)}]$, $\log E[B^{(b)}] = O(K^2)$. The lower bound, $\log E[B^{(b)}] = \Omega(K^2)$, follows similarly.

Finally, given that $E[N^{(b)}] = E[\Lambda B^{(b)}]$ we also have

$$\log E[N^{(b)}] = \Theta(K^2) \tag{A.27}$$

□

Accounting For the Impact of Peers That Are Served When Busy Period Starts

Consider the group of peers that wait for a publisher to arrive and immediately begins to be served when the busy period starts. In the following proof, we account for the possible impact of this group of peers on the duration of the busy period.

Proof. When a publisher arrives to start a busy period a group of peers that were waiting for a publisher immediately begins to be served. This group is modeled using a virtual customer, with residence time Y. Y is the maximum of all the residual download times of the queued peers that enter simultaneously into service and the residence time of the publisher,

$$Y = \max\{X_1, \ldots, X_L, X_{L+1}\}$$

where $X_1, \ldots, X_L$ are exponential random variables with mean s/μ and X_{L+1} is exponential with mean u. L is a geometric random variable with support $\{0, 1, \ldots\}$ and parameter $r/(\lambda + r)$, denoting the number of Poisson arrivals with rate λ in an exponentially distributed interval of average length $1/r$. Let $h(s)$ be the Laplace transform of Y and $\theta = E[Y]$.

To fully parameterize equation (A.2) it remains to determine $h(s)$. To this end, let the random variable Y_f be defined as

$$Y_f = \max\{X_1, X_2, \ldots, X_f, Z\} \tag{A.28}$$

where $X_1, \ldots, X_f$ are exponential random variables with mean s/μ and Z is an exponential random variable with mean u.

Recall that $\theta = E[Y]$,

$$E[Y] = \sum_{f=0}^{\infty} E[Y|L=f]P(L=f) \tag{A.29}$$

$$= \sum_{f=0}^{\infty} E[Y_f]P(L=f) \tag{A.30}$$

Next, we compute $E[Y_f]$. The cdf of Y_f is $F(x) = P(Y_f \leq x) = (1-e^{-\frac{\mu}{s}x})^f(1-e^{-(1/u)x})$, therefore

$$F(x) = \left(1 + \sum_{j=1}^{f}(-1)^j C(f,j)e^{-\frac{\mu}{s}jx}\right)(1-e^{-(1/u)x}) \tag{A.31}$$

We also have that

$$E[Y_f] = \int_{x=0}^{\infty}(1-F(x))dx \tag{A.32}$$

Then, substituting (A.31) into (A.32) yields

$$E[Y_f] = \frac{u}{C(f+(1/u)(s/\mu),f)} + \frac{s}{\mu}\Big(\Gamma + \Psi(f+1)\Big) \tag{A.33}$$

where $\Gamma = 0.5772...$ is the Euler's constant, $\Psi(z)$ is the di-gamma function,

$$\Psi(z) = \frac{d}{dz}\ln\Gamma(z), \qquad \Gamma(z) = \int_{t=0}^{\infty} e^{-t}t^{z-1}dt \tag{A.34}$$

and $C(m,n)$ is the generalized binomial coefficient,

$$C(m,n) = \Gamma(m+1)/(\Gamma(n+1)\Gamma(m-n+1))$$

Next, our goal is to compute $h(s)$. Let $h_f(s)$ be the Laplace transform of Y_f. Then, conditioning on the number of queued customers, $h(s)$ is

$$h(s) = \sum_{f=0}^{\infty} h(s|L=f)P(L=f) \tag{A.35}$$

$$= \sum_{f=0}^{\infty} h_f(s)P(L=f) \tag{A.36}$$

We now turn to the computation of $h_f(s)$. Let $W_{(i)}$ be the i^{th} order statistic among $\{X_1, \ldots, X_f, Z\}$. Consider the event

$$W_{(1)} \leq \ldots \leq W_{(i^*-1)} \leq Z \leq W_{(i^*+1)} \leq \ldots \leq W_{(f)}$$

We compute the Laplace transform of Y_f conditioning on the value of i^*,

$$h_f(s) = \sum_{j=1}^{f+1} P(i^*=j)h_f(s|i^*=j) \tag{A.37}$$

where $P(i^*=j)$ equals

$$\frac{1/u}{(f-j+1)(\mu/s)+1/u} \prod_{i=1}^{j-1} \frac{(\mu/s)(f-i+1)}{(f-i+1)(\mu/s)+1/u}$$

Let U_i denote an exponential random variable with rate $(f-i+1)(\mu/s)+(1/u)$, $1 \leq i \leq i^*$, and V_i denote an exponential random variable with rate $i(\mu/s)$, $1 \leq i \leq f-i^*+1$. Then

$$h_f(s|i^*=j) = E[e^{-s\max\{W_{(1)},\ldots,W_{(i^*)},\ldots,W_{(f)}\}}|i^*=j] =$$

$$= E[e^{-s(\sum_{i=1}^{j} U_i + \sum_{i=1}^{f-j+1} V_i)}] = \prod_{i=1}^{j} E[e^{-sU_i}] \prod_{i=1}^{f-j+1} E[e^{-sV_i}]$$

$$= \prod_{i=1}^{j} \frac{(f-i+1)(\mu/s)+(1/u)}{(f-i+1)(\mu/s)+(1/u)+s} \prod_{i=1}^{f-j+1} \frac{i(\mu/s)}{i(\mu/s)+s}$$

Substituting this last expression into (A.37) yields $h_f(s)$. □

Extensions of Lemma 2.3.1

Proof. The extension of Lemma 2.3.1 to cope with Zipfian demand and to account for the impact of the peers that wait for a publisher and immediately start to be serviced when the publisher returns are described next.

1. *Zipfian demand:* let η_k be the ratio of the mean busy period duration of the bundled swarm over the mean busy period duration of individual swarm k.

$$\eta_k = \frac{E[B^{(b)}]}{E[B_k]}, \qquad k = 1, \ldots, K$$

   $B^{(b)}$ and B_k are defined similarly as in Lemma 2.3.1, except that $K\breve{\lambda}p_K \leq \beta \leq K\hat{\lambda} + R$. Following an argument similar to the one to prove Lemma 2.3.1, it can be shown that $\eta_k = e^{\Theta(K^2)}$.

2. *waiting peers:* the idle periods have mean duration $1/R$ and the mean number of peers that arrive during each of them is upper bounded by $K\hat{\lambda}$. Each peer requires mean service time upper bounded by $K\hat{s}/\mu$ once the busy period starts. The longest mean service time among the waiting peers is upper bounded by $K^2\hat{s}\hat{\lambda}/(R\mu)$. Therefore, to cope with the impact of peers that wait for a publisher and immediately start to be serviced when the publisher returns, we consider a virtual customer whose mean residence time is upper bounded by $U + K^2\hat{\lambda}\hat{s}/(R\mu)$. Under such conditions, Lemma 2.3.1 still holds (note that the residence time of the first customer is allowed to vary between U and $U + K^2\hat{\lambda}\hat{s}/(R\mu)$ in Lemma 2.3.1).

□

Proof of Theorem 2.3.3

Proof. Since all the terms in P_k, $k = 1, \ldots, K$, are lower bounded and upper bounded by terms that do not depend on K, $E[B_k]$ is bounded hence $P_k = \Theta(1)$.

We rewrite $-\log P^{(b)}$ as

$$\begin{aligned} -\log P^{(b)} &= -\log \frac{1/R}{E[B^{(b)}]+1/R} \\ &= -\log(1/R)+\log(e^{\Theta(K^2)}+1/R) \end{aligned}$$

where the last equality follows from Lemma 2.3.1.

We now show that $-\log P^{(b)} = \Theta(K^2)$. First, we show that $-\log P^{(b)} = O(K^2)$,

$$\lim_{K\to\infty} \frac{-\log P^{(b)}}{K^2} = \kappa_1 + \lim_{K\to\infty} \frac{\log(e^{\Theta(K^2)}+1/R)}{K^2} < \infty \tag{A.38}$$

Then, we show that $-\log P^{(b)} = \Omega(K^2)$,

$$\lim_{K\to\infty} \frac{K^2}{-\log P^{(b)}} = \lim_{K\to\infty} \left[\kappa_2 + \frac{\log\left(e^{\Theta(K^2)}+1/R\right)}{K^2}\right]^{-1} < \infty \tag{A.39}$$

from which we conclude that $-\log P^{(b)} = \Theta(K^2)$. □

Derivation of equation (2.13)

Proof. The mean download time of a peer is constituted by two components: the mean active download time and mean idle waiting. The mean active download time is Ks/μ and the mean idle waiting is the probability of arriving to the system when content is unavailable, $(1/R)\Big/(1/R + E[B^{(b)}])$, times the mean time for a publisher to arrive, $1/R$. According to Lemma 2.3.1, $E[B^{(b)}] = e^{\Theta(K^2)}$. □

Proof of remark following equation (2.13)

If $K = \sqrt{\log(1/R)}$ then $E[T]/E[T^{(b)}] = \Theta\Big(R^{-c}(\sqrt{\log(1/R)})^{-b}\Big)$, $b \geq 0, c > 0$.

Proof. If $K = \sqrt{\log(1/R)}$,

$$\lim_{K\to\infty} \frac{E[T]}{E[T^{(b)}]} = \frac{\frac{s}{\mu}+e^{K^2}}{K\frac{s}{\mu}+e^{K^2}e^{-\Theta(K^2)}} = \tag{A.40}$$

$$= \Theta(e^{cK^2}/K^b) = \Theta\Big(\frac{1}{R^c(\sqrt{\log(1/R)})^b}\Big), b \geq 0, c > 0$$

where the first equality follows from Theorem 2.3.3. □

Derivation of equation (2.14)

Proof. Consider the first instant of time t at which there are no publishers online. The time until the system becomes empty after t is referred to as the residual busy period. Due to the memoryless property of the exponential random variable, the virtual customer that starts the residual busy period is characterized by a random variable $Y = \max\{X_1, \ldots, X_n\}$ where $X_1, \ldots, X_n$ are exponential random variables with mean s/μ. Therefore, Y is an hypoexponential distribution with parameters $(s/\mu, s/(2\mu), \ldots, s/(n\mu))$, which has Laplace transform $\prod_{i=1}^{n}(i\mu/s)/(s + i\mu/s)$ and mean $\sum_{i=1}^{n} s/(i\mu)$. Equation (A.2) can be used to compute $B(n, 0)$ for any value of n (eq. (2.14)).

Let us denote by $T_{i,j}$ the time it takes for a residual busy period which starts with i peers to reach a population size of $j < i$ peers, where $B(i, j) = E[T_{i,j}]$. For $n > l$ and $n > k > l$, we have that $T_{n,l} = T_{n,k} + T_{k,l}$. Therefore, in general $E[T_{n,l}] = E[T_{n,k}]+E[T_{k,l}]$ and in particular, $E[T_{n,l}] = E[T_{n,0}]-E[T_{l,0}]$. Equation (2.14) and $B(n, l) = B(n, 0) - B(l, 0)$ provide a way to compute $B(n, l)$ for arbitrary values of n and $l < n$. □

Equation (2.14) *and congestion periods [79]*

We now relate (2.14) to [79, eq. (10)]. Let $\rho = \lambda s/\mu$. Given $C > 0$,

$$B(C+1,C) = B(C+1,0) - B(C,0) = \tag{A.41}$$

$$\begin{aligned}
&= \frac{1}{(C+1)\mu/s} + \frac{s}{\mu}\sum_{i=1}^{\infty}\left(\frac{\lambda s}{\mu}\right)^i \frac{i!C!i}{i!i(C+1+i)!} \\
&= \frac{1}{(C+1)\mu/s} + \frac{s}{\mu}\sum_{i=1}^{\infty}\left(\frac{\lambda s}{\mu}\right)^i \frac{C!}{(C+1+i)!} \\
&= \frac{1}{(C+1)\mu/s} + \frac{s}{\mu}\sum_{i=C+1}^{\infty}\left(\frac{\lambda s}{\mu}\right)^{i-C} \frac{C!}{(1+i)!} \\
&= \frac{s}{\mu}\sum_{i=C}^{\infty}\left(\frac{\lambda s}{\mu}\right)^{i-C} \frac{C!}{(1+i)!} \\
&= \frac{C!}{\lambda\rho}\sum_{i=C+1}^{\infty}\rho^{i-C-1}\frac{1}{i!} \\
&= \frac{C!}{\lambda\rho^C}\sum_{i=C+1}^{\infty}\frac{\rho^i}{i!} \qquad \text{(A.42)}
\end{aligned}$$

where the last expression corresponds to (10) in [79].

Proof that $B(m) = e^{\Theta(K^2)}$

$$B(m) = \sum_{i=0}^{\infty}\frac{e^{\lambda s/\mu}\left(\frac{\lambda s}{\mu}\right)^i}{i!}B(i,m) \tag{A.43}$$

The corresponding quantity for the bundled swarm, $B^{(b)}(m)$, is obtained by replacing in the above expression s and λ by Ks and $K\lambda$, respectively.

First, we show that

$$\log\left(B^{(b)}(m)\right) = O(K^2) \tag{A.44}$$

It follows from (2.14) that

$$B^{(b)}(i,m) \le B^{(b)}(i,0) = O(Ke^{(\lambda s/\mu)K^2}) \tag{A.45}$$

The fact that $B^{(b)}(m)$ is a weighted sum of $B^{(b)}(i,m)$ $(i \ge 0)$, together with (A.45), yields (A.44).

Next, we show that

$$\log\left(B^{(b)}(m)\right) = \Omega(K^2) \tag{A.46}$$

Given the threshold coverage m, the congestion period when the system begins with i users is given by $B(i,m)$,

$$B(i,m) = \sum_{k=m}^{i-1} B(k+1,k) \tag{A.47}$$

Therefore, when $i > m$, the congestion period $B(i,m)$ is lower bounded by $B(m+1,m)$,

$$B(i,m) \geq B(m+1,m), \qquad i > m \tag{A.48}$$

$B^{(b)}(m+1,m)$ is given by (A.42),

$$B^{(b)}(m+1,m) = \frac{m!}{\Lambda\nu^m}\sum_{i=m+1}^{\infty}\frac{\nu^i}{i!} = \frac{m!}{\Lambda\nu^m}\left(e^{\nu} - \sum_{i=0}^{m}\frac{\nu^i}{i!}\right), \tag{A.49}$$

$$\text{where} \qquad \Lambda = K\lambda, \tag{A.50}$$

$$\nu = K^2\frac{\lambda}{\mu} \tag{A.51}$$

Hence,

$$B^{(b)}(m) \geq \left(1 - \sum_{j=0}^{m} e^{-K^2\frac{\lambda}{\mu}}\frac{(K^2\frac{\lambda}{\mu})^j}{j!}\right)B^{(b)}(m+1,m) \tag{A.52}$$

$$= \underbrace{\left(1 - \sum_{j=0}^{m} e^{-K^2\frac{\lambda}{\mu}}\frac{(K^2\frac{\lambda}{\mu})^j}{j!}\right)}_{(*)}\underbrace{\left(\frac{m!}{\Lambda\nu^m}\left(e^{\nu} - \sum_{i=0}^{m}\frac{\nu^i}{i!}\right)\right)}_{(**)}$$

$$\geq \frac{e^{K^2\frac{\lambda}{\mu}}}{(K^2\frac{\lambda}{\mu})^m K\lambda} - 2(m!)\sum_{j=0}^{m}(K^2\frac{\lambda}{\mu})^j/j! \tag{A.53}$$

where (*) equals $\sum_{j=m+1}^{\infty} e^{-K^2\frac{\lambda}{\mu}}(K^2\lambda/\mu)^j/j!$ and (**) is a lower bound on $B^{(b)}(i,m), i \geq m+1$, and equals $B^{(b)}(m+1,m)$.

(A.53) yields

$$\lim_{K\to\infty} \frac{e^{\lambda K^2/\mu}/((K^2\lambda/\mu)^m(K\lambda))}{B^{(b)}(m)} \leq \tag{A.54}$$

$$\leq \lim_{K\to\infty} \frac{e^{\lambda K^2/\mu}/((K^2\lambda/\mu)^m(K\lambda))}{\frac{e^{K^2\lambda/\mu}}{(K^2\lambda/\mu)^m K\lambda} - 2(m!)\sum_{j=0}^{r}(K^2\lambda/\mu)^j/j!} \tag{A.55}$$

$$= \lim_{K\to\infty} \frac{1}{1 - \frac{(K^2\lambda/\mu)^m K\lambda}{e^{K^2\lambda/\mu}} 2(m!)\sum_{j=0}^{m}(K^2\lambda/\mu)^j/j!} = 1$$

where the last equality is derived using l'Hopital rule,

$$\lim_{K\to\infty} \frac{(K^2\lambda/\mu)^m(K\lambda)}{e^{K^2\lambda/\mu}} \sum_{j=0}^{m}(K^2\lambda/\mu)^j/j! = \lim_{K\to\infty} \frac{\kappa_1}{\kappa_2 e^{K^2\lambda/\mu}} = 0$$

where κ_1 and κ_2 are constants functionally independent of K.

Therefore, if follows from (A.55) and (A.56) that

$$B^{(b)}(m) = \Omega(e^{\lambda K^2/\mu}/((K\lambda)(K^2\lambda/\mu)^m))$$

which implies that

$$\log\left(B^{(b)}(m)\right) = \Omega\left(\log\left(e^{\lambda K^2/\mu}/((K\lambda)(K^2\lambda/\mu)^r)\right)\right) = \Omega(K^2)$$

Derivation of equation (2.16)

Proof. The probability that a request leaves without being served, P, is

$$P = \frac{1/r}{E[B] + 1/r} \tag{A.56}$$

We now compute $E[B]$, the expected length of the busy period of a system which cycles between three phases: (1) one or more publishers are available; (2) no publisher is available but content is still available; (3) the content is not available.

Consider an instant of time t at which the system transitions to Phase 2. The time until the number of peers reaches the value $m > 0$ is referred to as the residual busy period. We denote by $B(m)$ the expected length of a residual busy period. Assuming peers reached steady state before the number of publishers went to zero,

$$B(m) = \sum_{i=0}^{\infty} \frac{e^{-\frac{\lambda s}{\mu}} \left(\frac{\lambda s}{\mu}\right)^i}{i!} B(i, m) \tag{A.57}$$

The system starts in (1). It cycles between (1), (2), (1), (2) and so on up to reaching phase (3). Let us denote the number of times that it goes through (1) and (2), before reaching (1) for the last time and finally (2) and (3), by a geometrically distributed random variable C. C has support $\{0, 1, \dots\}$ and success probability p,

$$p = P\{X > B(m)\} = e^{-rB(m)} \tag{A.58}$$

where X is an exponentially distributed random variable with rate r. Therefore,

$$E[C] = \frac{1-p}{p} = \frac{1 - e^{-rB(m)}}{e^{-rB(m)}} \tag{A.59}$$

Phase (1) takes on average $\frac{e^{ru}-1}{r}$, the busy period of a population formed only by publishers. Phases (1) and (2) together, when not followed by (3), take on average

$$\frac{e^{ru} - 1}{r} + E[X|X < B(m)] \tag{A.60}$$

where

$$E[X|X < B(m)] = \frac{\frac{1}{r} - \left(\frac{1}{r} + B(m)\right) e^{-rB(m)}}{1 - e^{-rB(m)}} \tag{A.61}$$

The expected active period is

$$E[B] = E[C]\Big(\frac{e^{ru} - 1}{r} + E[X|X < B(m)]\Big) + \frac{e^{ru} - 1}{r} + B(m) \tag{A.62}$$

Substituting (A.57), (A.59) and (A.61) into (A.62) yields

$$E[B] = \frac{e^{r(B(m)+u)} - 1}{r} \tag{A.63}$$

Replacing (A.63) into (A.56) leads to the desired result. □

Derivation of equation (2.18)

Proof. The derivation is similar to the one of (2.16). Let

$$B^{(b)}(m) = \sum_{i=0}^{\infty} \frac{e^{-\frac{K^2\lambda s}{\mu}} \left(\frac{K^2\lambda s}{\mu}\right)^i}{i!} B^{(b)}(i, m) \tag{A.64}$$

Let X be an exponentially distributed random variable with rate R. Then, with a slight abuse of notation, the expected active period is, similarly to (A.62),

$$E[B^{(b)}] = E[C]\Big(U + E[X|X < B^{(b)}(m)]\Big) + U + B^{(b)}(m) \tag{A.65}$$

where

$$E[X|X < B^{(b)}(m)] = \frac{\frac{1}{R} - \left(\frac{1}{R} + B^{(b)}(m)\right)e^{-RB^{(b)}(m)}}{1 - e^{-RB^{(b)}(m)}} \tag{A.66}$$

Substituting (A.66) and (A.59) into (A.65) yields

$$E[B^{(b)}] = \frac{e^{RB^{(b)}(m)}(UR + 1) - 1}{R} \tag{A.67}$$

and (2.18) follows from (2.3) and (A.67). □

Altruistic lingering (see § 2.3.3.4)

The availability and download time theorems hold in the scenario with altruistic lingering since adding a constant to the residence time of each peer does not change the asymptotics of the system when $K \to \infty$.

To model altruistic lingering we use equation (A.1). Let S be an exponentially distributed random variable with mean u, representing the residence time of publishers. Let L be the sum of two exponentially distributed random variables with means $\beta_1 = s/\mu$ and $\beta_2 = 1/\gamma$ representing the residence time of peers after starting the download. The distribution of L is

$$\begin{aligned} P(L \leq x) &= L(x) = \\ &= \frac{(1/\beta_1)(e^{-\frac{1}{\beta_1}x} - 1) - (1/\beta_2)(e^{-\frac{1}{\beta_2}x} - 1)}{(1/\beta_2) - (1/\beta_1)} \end{aligned} \tag{A.68}$$

if $\beta_1 \neq \beta_2$ and Erlang(2, β_1) otherwise. A customer arriving in an active period draws its residence time X as follows,

$$X = \begin{cases} L, & \text{with probability } q_1' \\ S, & \text{with probability } 1 - q_1' \end{cases} \tag{A.69}$$

where $q_1' = \lambda/(\lambda + r)$. Therefore, we parameterize (A.1) setting $\beta = \lambda + r$, $H(x) = S(x) = 1 - e^{-ux}$, $\theta = u$, $G(x) = (1 - q_1')S(x) + q_1'L(x)$.

For the bundled swarm, $\beta = \Lambda + R$, $H(x) = S(x) = 1 - e^{-Ux}$, $\theta = U$, $q_1' = \Lambda/(\Lambda + R)$, $\beta_1 = S/\mu$ and $\beta_2 = 1/\gamma$.

A.3 Trace driven arrivals

In this section we report results obtained using peer arrival patterns observed in real swarms. We selected two files from the 2008 Olympic game opening ceremony (which we refer to as files A and B). We scaled the arrival rates so that the swarms in isolation were not popular but as a bundle were popular enough to be self-sustaining. In this Planetlab experiment, we set publisher up and down times, U and D, to be exponentially distributed with mean of 500s and 1500s, respectively. Both files have size S=10 MB and we set the capacity of peers and publishers as 50KBps and 100KBps, respectively.

For each of the three scenarios that we considered [(1) bundle, (2) file A isolated and (3) file B isolated] we ran a Planetlab experiment for 12 hours. The download distributions for the three considered scenarios are shown in Figure A.4. The dots mark the mean and the box plots show the 1st, 2nd and 3rd quartiles and, the 5 and 95 percentiles. Note that the mean and variance in the bundled case are significantly smaller than the ones for the two individual files, with an improvement in download time of 39% and 41%, respectively.

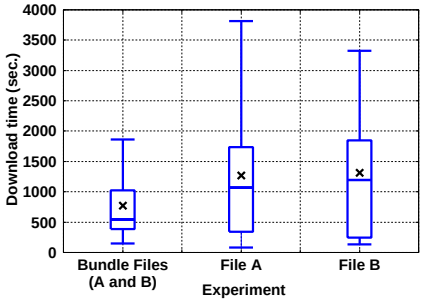


Figure A.4. Experiment using trace-driven arrivals (Mean, quartiles, 5th and 95th percentiles)

The impact of the threshold coverage

In this section we further study the impact of the threshold coverage on our estimates of the mean download time. Tables A.1 and A.2 show the mean number of peers necessary to attain a self-sustainability level of 0.9 and 0.999, respectively, as shown in Figure 3.5.

Although the assumptions in Chapter 3 are slightly different from the ones considered in Chapter 2 (in particular, in Chapter 3 it is assumed that there is a publisher that is always online), the download times predicted by our model using the threshold coverage suggested in Chapter 3 are in accordance to the ones observed in our experiments (see Figure 2.6(a)). In part, this is because our model is not very sensitive to slight variations in the threshold coverage, as illustrated in Figure A.5, obtained using the same parameters as Figure 2.6(a), and varying the threshold coverage m between 8 and 24.

file size (number of blocks)	suggested threshold coverage
12-18	10
20-32	11
34-58	12
60-102	13
104-172	14
174-288	15
290-482	16
484-800	17
802-1000	18

Table A.1. Suggested threshold coverage for different file sizes (target self-sustainability of 0.9)

file size (number of blocks)	suggested threshold coverage
12-22	20
24-46	21
48-86	22
88-150	23
152-254	24
256-428	25
430-714	26
716-1000	27

Table A.2. Suggested threshold coverage for different file sizes (target self-sustainability of 0.999)

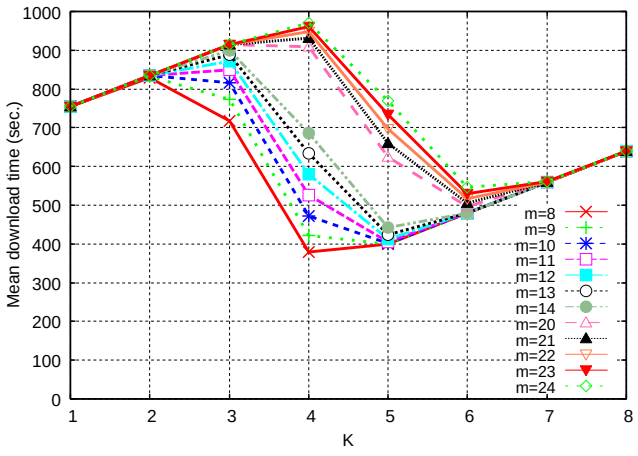


Figure A.5. The impact of the threshold coverage

APPENDIX B

SELF-SUSTAINABILITY

B.1 An Expression of $P(V = B|\mathbf{N} = \mathbf{n})$

We now show how to obtain $P(V = B|\mathbf{N} = \mathbf{n})$ from (3.2). Recall that, given an upper layer state $\mathbf{n}$, $\Omega_{\mathbf{n}}$ is the lower layer sample space. Denote by $\mathbf{S}$ the random variable that represents the lower layer state, and by $\mathbf{s}$ its realization. Then

$$P(V = B|\mathbf{N} = \mathbf{n}) = \sum_{\mathbf{s}:V=B,\mathbf{s}\in\Omega_{\mathbf{n}}} P(\mathbf{S} = \mathbf{s}|\mathbf{N} = \mathbf{n}) \overset{(3.2)}{=} \sum_{\mathbf{s}:V=B,\mathbf{s}\in\Omega_{\mathbf{n}}} 1/|\Omega_{\mathbf{n}}| \qquad \text{(B.1)}$$

Therefore, the problem of computing $P(V = B|\mathbf{N} = \mathbf{n})$ is reduced to that of counting the states in which all blocks are available among the peers. Using the inclusion/exclusion principle, $P(V = B|\mathbf{N} = \mathbf{n}) = |\Omega_{\mathbf{n}}|^{-1} \sum_{i=0}^{B-1} (-1)^i \binom{B}{i} \prod_{j=1}^{B-1} \binom{B-i}{j}^{n_j}$. In general, $P(V = v|\mathbf{N} = \mathbf{n})$ is also obtained using the inclusion/exclusion principle [98, Section 4.2].

B.2 Recursion to Compute $\psi_h(k, v)$

Consider the scenario where k blocks are available among the peers and an additional user contributes h blocks. $\psi_h(k, v)$ is the probability that v blocks are available among the peers after accounting for the blocks contributed by the additional user. $\psi_h(k, v)$ can be recursively computed,

$$\psi_h(k, v) = \psi_{h-1}(k, v-1)\frac{B-v-h}{B-h+1} + \psi_{h-1}(k, v)\frac{v}{B-h+1}, \qquad 0 \le k, v \le B, \quad 0 < h \le B \qquad \text{(B.2)}$$

The base cases are $\psi_0(k,v) = 0$ if $v \neq k$ and $\psi_0(k,v) = 1$ if $v = k$. Note that the above recursion is convenient to avoid numerical problems since it only involves additions and multiplications of probabilities.

For presentation convenience, we consider an arbitrary ordering of the h blocks contributed by the additional user. After contributing the first $h-1$ blocks, there are two cases to consider, (i) v blocks are available among the peers [and the h^{th} block overlaps with a previously available block, an event which happens with probability $v/(B-h+1)$] or (ii) $v-1$ blocks are available [and the h^{th} block does not overlap with previously available blocks, an event which happens with probability $(B-v-h)/(B-h+1)$]. Cases (i) and (ii) correspond to the first and second terms in (B.2), respectively.

B.3 Proof of Lemma 3.3.1

We now show that (3.12) and (3.10) are equivalent. To this purpose, we consider an extended representation of the upper layer states. In such representation, the upper layer state, $\mathbf{n}'$, is defined as follows.

Consider the users in the system ordered uniformly at random. The number of blocks owned by the i^{th} user is denoted by n'_i. The upper layer state is represented by vector $\mathbf{n}'$. The dimension of $\mathbf{n}'$ is $1 \times n$, where n is the number of users in the system.

Note that $\mathbf{n}$ is inferred from $\mathbf{n}'$. Let $n_l(\mathbf{n}')$ be the number of users that have l blocks when the state is $\mathbf{n}'$. Since any permutation of the users in the system is equally likely, the steady state probability of state $\mathbf{n}'$ is $\pi(\mathbf{n}')$,

$$\pi(\mathbf{n}') = \prod_{l=0}^{B-1} \left[\frac{\rho^{n_l(\mathbf{n}')}}{n_l(\mathbf{n}')!} e^{-\rho} \right] \frac{1}{n!}$$

The steady state probability of state $\mathbf{n}'$, conditioned on the event that there are n users in the system, is

$$\frac{\pi(\mathbf{n}')}{\sum_{\forall \mathbf{n}':|\mathbf{n}'|=n}\pi(\mathbf{n}')}=\frac{\pi(\mathbf{n}')}{e^{-B\rho}(B\rho)^n/n!}=\frac{1}{B^n\prod_{l=0}^{B-1}n_l(\mathbf{n}')!} \tag{B.3}$$

The first equality follows from the fact that a superposition of B Poisson processes with rate ρ is a Poisson process with rate $B\rho$.

The key idea of the proof consists of partitioning the state space into sets $G_h(n)$, $0 \leq h \leq B-1$, $0 \leq n$. Set $G_h(n)$ contains states in which there are n users in the system and $n'_n = h$. $G_h(n)$ is defined as $G_h(n) = \{\mathbf{n}' : n'_n = h \text{ and } 0 \leq n'_j \leq B-1, 1 \leq j < n\}$. The set containing all states in which there are n users in the system, $G(n)$, is

$$G(n) = \cup_{h=0}^{B-1} G_h(n) = \{\mathbf{n}' : 0 \leq n'_j \leq B-1, 1 \leq j \leq n\} \tag{B.4}$$

Note that $G_h(n)$ is obtained from $G(n-1)$ by adding to each element of $G(n-1)$ a user that owns h blocks,

$$G_h(n) = \{(\mathbf{n}', 0) + h\mathbf{1}_n : \mathbf{n}' \in G(n-1)\} \tag{B.5}$$

$\mathbf{1}_n$ denotes a $1 \times n$ vector in which all elements equal zero, except element n, which equals one. Let $p(v|\mathbf{n}')$ be the probability of v blocks being available among the peers when the upper layer state is $\mathbf{n}'$. If $\mathbf{n}' \in G_h(n)$ $(n > 0)$,

$$p(v|\mathbf{n}') = \sum_{k=0}^{B} p(k|\mathbf{n}' - h\mathbf{1}_n)\psi_h(k, v), \qquad \mathbf{n}' \in G_h(n) \tag{B.6}$$

According to definition (3.10),

$$p_n(v)=\left[\sum_{\mathbf{n}'\in G(n)} p(v|\mathbf{n}')\pi(\mathbf{n}')\right] \Big/ \left[\sum_{\mathbf{n}'\in G(n)} \pi(\mathbf{n}')\right] = \sum_{\mathbf{n}'\in G(n)} p(v|\mathbf{n}')\frac{1}{B^n \prod_{l=0}^{B-1} n_l(\mathbf{n}')!} \tag{B.7}$$

where the first and second equalities follow from (B.4) and (B.3), respectively. Hence,

$$\begin{aligned}
p_n(v) &= \sum_{h=0}^{B-1} \sum_{\mathbf{n}'\in G_h(n)} p(v|\mathbf{n}')\frac{1}{B^n \prod_{l=0}^{B-1} n_l(\mathbf{n}')!} && \text{(B.8)}\\
&= \sum_{h=0}^{B-1} \sum_{\mathbf{n}'\in G_h(n)} \sum_{k=0}^{B} p(k|\mathbf{n}'-h\mathbf{1}_n)\psi_h(k,v)\frac{1}{B^n \prod_{l=0}^{B-1} n_l(\mathbf{n}')!} && \text{(B.9)}\\
&= \sum_{h=0}^{B-1}\sum_{k=0}^{B} \underbrace{\sum_{\mathbf{m}'\in G(n-1)} p(k|\mathbf{m}')\frac{1}{B^{n-1}\prod_{l=0}^{B} n_l'!}}_{p_{n-1}(k)} \psi_h(k,v)\frac{1}{B} && \text{(B.10)}\\
&= \sum_{h=0}^{B-1}\sum_{k=0}^{B} p_{n-1}(k)\psi_h(k,v)\frac{1}{B} && \text{(B.11)}
\end{aligned}$$

where the first, second and third equalities follow from (B.4), (B.6) and (B.5), respectively. Note that the summands in (B.10) for which $h > v$ or $k \in [0, v-h-1]\cup[v+1, B]$ are equal to zero, since in these cases $\psi_h(k,v) = 0$. Therefore, (B.10) yields (3.12).

B.4 Probabilistic Derivation of (3.14)

Next, we provide a probabilistic derivation of (3.14). An algebraic proof can be obtained using the Sigma package [81], applying a paradigm called creative telescoping [105].

Henceforth, we consider the case $v < B$ (the case $v = B$ follows similarly). The probabilistic interpretation for (3.14) follows from the connection between $\psi_h(k,v)$ and the hypergeometric distribution. Suppose we have an urn containing B balls, $B-k$ of which are white (unavailable blocks) and k are black (available blocks).

$\psi_h(k,v)$ is the probability of selecting without replacement h balls, of which $v-k \leq h$ are white.

Consider now another experiment, namely selecting without replacement all balls from the urn. Let J_w be the round in which the w^{th} white ball is selected, $0 < J_1 < J_2 < \ldots < J_{B-k} < B+1$. Let W_b be the number of black balls selected between the b^{th} and $b+1^{th}$ white, plus one. Equivalently, W_b is the number of elements in the set $S = \{n \in \{0,\ldots,B\} : \text{ exactly } b \text{ white balls are selected before ball } n\}$. Then $W_0 = J_1$, $W_1 = J_2 - J_1$, ..., $W_{k-1} = J_k - J_{k-1}$, and $W_{B-k} = B+1-J_i$. Clearly, $\sum_{b=0}^{B-k} W_b = B+1$. By symmetry, $E[W_b] = (B+1)/(B-k+1)$ $(0 \leq b \leq B-k)$.

Now let $v-k$ be given, $v-k \in \{0,1,\ldots,v\}$. Let $\mathbf{1}_h$ be the indicator equal to 1 if exactly $v-k$ white balls are selected among the first h balls. Note that $E[\mathbf{1}_h] = \psi_h(k,v)$. Also, from the definition of W_{v-k}, we have $W_{v-k} = \sum_{h=0}^{B} \mathbf{1}_h$. Therefore, $E[W_{v-k}] = \sum_{h=0}^{B} E[\mathbf{1}_h] = \sum_{h=0}^{B} \psi_h(k,v) = (B+1)/(B-k+1)$.

B.5 Proof of Theorem 3.3.1

Replacing (3.14) into (3.13) yields,

$$p_n(v) = \begin{cases} 1/B^n, & n \geq 1, v = 0 \quad (i) \\ \sum_{k=0}^{v} p_{n-1}(k)(B+1)/(B(B-k+1)), & n \geq 1, 0 < v < B \quad (ii) \\ 1 - \sum_{l=0}^{B-1} p_n(l), & n \geq 1, v = B \quad (iii) \\ 1, & n = 0, v = 0 \quad (iv) \\ 0, & n = 0, v \neq 0 \quad (v) \end{cases} \tag{B.12}$$

In case (ii), $p_n(v) = p_{n-1}(v)(B+1)/(B(B-v+1)) + \sum_{k=0}^{v-1} p_{n-1}(k)(B+1)/(B(B-k+1)) = p_{n-1}(v)(B+1)/(B(B-v+1)) + p_n(v-1)$ which yields (3.16).

B.6 Expression of $p_n(v)$ When $\gamma = \infty$

If $\gamma = \infty$, the closed-form expression for $p_n(v)$ is $p_n(v)=\binom{B}{v}\left(\frac{1+B}{B}\right)^n \sum_{l=0}^{v}\binom{v}{l}(-1)^l(B-v+l+1)^{-n}$, $n \geq 1, B > v \geq 0$, and $p_n(B) = 1 - \sum_{v=0}^{B-1} p_n(v)$.

B.7 Self-Sustainability with Heterogeneous Download Times

We now present an efficient algorithm to compute swarm self-sustainability if the block download times, μ_h, are not necessarily equal for all h, $1 \leq h \leq B$.

Denote by $a_{h,r}(v)$ the probability that v blocks are available among the peers conditioned on the presence of r users in layer h and all users being in layers h up to B,

$$a_{h,r}(v) = P(V = v \Big| n_h = r; n_i = 0, i < h), \qquad 0 \leq h \leq B, \qquad 0 \leq r \leq M \quad \text{(B.13)}$$

M is the maximum number of users per layer. The following recursion correctly computes $a_{h,r}$, $0 \leq h \leq B$, $0 \leq r \leq M$,

$$a_{h,r}(v)=\begin{cases} \sum_{i=0}^{v} a_{h,r-1}(i)\psi_h(i,v), & r \geq 1, h \neq B \quad (i) \\ \sum_{r=0}^{\infty} a_{h+1,r}(v)\pi_{h+1}(r), & r = 0, h \neq B \quad (ii) \\ 1, & r = 0, h = B, v = 0 \quad (iii) \\ 0, & r = 0, h = B, v \neq 0 \quad (iv) \end{cases} \quad \text{(B.14)}$$

We approximate A by its truncated version, $\overline{A}^{(M)}$, considering only states in which there are up to M users in each layer of the system. Recall that N is the maximum number of users in the system, $N = O(BM)$. A naive use of (B.14) yields an algorithm to compute $\overline{A}^{(M)}$ in time $O(B^3M) = O(B^2N)$.

In what follows, we show how to compute $\overline{A}^{(M)}$ in $O(NB\log B)$. For this purpose, we re-write the sum in (B.14)(i) as a convolution, which is computed in time $O(B\log B)$. Let

$$\widehat{a}_{h,r-1}(k) = a_{h,r-1}(k)k!(B-k)!, \quad 0 \le k \le B \tag{B.15}$$

$$\widehat{\psi}_h(v-k) = 1/((v-k)!(h-(v-k))!), \quad 0 \le k \le v \tag{B.16}$$

Then, the following convolution correctly computes (B.14)(i), $\widehat{a}_{h,r} = \widehat{a}_{h,r-1} \otimes \widehat{\psi}_h$. $a_{h,r}$ is obtained from $\widehat{a}_{h,r}$ as

$$a_{h,r}(v) = \begin{cases} (\widehat{a}_{h,r}(v))/\left((v-h)!(B-v)!\binom{B}{h}\right), & 0 \le v \le B-1 \\ 1 - \sum_{i=0}^{B-1} a_{h,r}(i), & v = B \end{cases} \tag{B.17}$$

B.8 Derivation of (3.19)

We now show that (3.19) follows from (3.18). The case $v = B$ follows trivially. For $v < B$, we show that (3.19) satisfies (3.18), i.e., $p_n(v) - p_{n-1}(v)(1/(B-v+1)) = p_n(v-1)$,

$$\begin{aligned} & p_n(v) - p_{n-1}(v)(1/(B-v+1)) = \\ = \; & \binom{B}{v}\sum_{l=0}^{v}\binom{v}{l}(-1)^l(B-v+l+1)^{-n} - \frac{\binom{B}{v}\sum_{l=0}^{v}\binom{v}{l}(-1)^l(B-v+l+1)^{-n+1}}{B-v+1} \\ = \; & \binom{B}{v}\sum_{l=0}^{v}\binom{v}{l}(-1)^{l+1}(B-v+l+1)^{-n}\frac{l}{B-v+1} \qquad \text{(B.18)} \\ = \; & \binom{B}{v-1}\sum_{l=0}^{v-1}\binom{v-1}{l}(-1)^l(B-v+l+2)^{-n} = p_n(v-1) \qquad \text{(B.19)} \end{aligned}$$

B.9 Derivation of (3.20)

We now derive (3.20) from $p_n(B) = \sum_{l=0}^{B} \binom{B}{l}(-1)^l(l+1)^{-n}$.

$$\begin{aligned} p(B) &= \sum_{n=0}^{\infty}\sum_{l=0}^{B} p_n(B)e^{-(B+1)\rho}((B+1)\rho)^n/n! = \qquad \text{(B.20)} \\ &= \sum_{l=0}^{B}\binom{B}{l}(-1)^l e^{-B\rho}e^{(B+1)\rho/(l+1)}\underbrace{\sum_{n=0}^{\infty} e^{-(B+1)\rho/(l+1)}((B+1)\rho/(l+1))^n/n!}_{=1} \end{aligned}$$

B.10 Probability that l tagged blocks are unavailable

The probability that l tagged blocks are unavailable, when $\gamma = \mu$, is $\exp\left(-\rho\left(\frac{l(B+1)}{l+1}\right)\right)$.

Let $\alpha_{h,r}$ be the probability that there are l tagged unavailable blocks, conditioned on all users being in stages $h, h+1, \ldots, B-1, B$, stage h having r users (similar in spirit to recursion presented in Appendix B.7). α_h is the corresponding metric, but with no conditioning on the number of users in layer h. Given l, the following recursion yields $\alpha_{h,r}$.

$$\alpha_{h,r} = \begin{cases} (i) & 1 & h{=}B, r{=}0 \\ (ii) & \alpha_{h,r-1}\prod_{i=0}^{h-1}\frac{B-l-i}{B-i} & h \le B, r \ge 1 \\ (iii) & \sum_{m=0}^{\infty}\alpha_{h+1,m}\pi_{h+1}(m) & h \le B, r = 0 \end{cases} \qquad \text{(B.21)}$$

Our goal is to find the expression of $\alpha_{h,0}$,

$$\alpha_{h,0} = \sum_{m=0}^{\infty}\alpha_{h+1,m}\pi_{h+1}(m) \qquad \text{(B.22)}$$

From (B.21)(ii),

$$\alpha_{h+1,m} = \alpha_{h+1,m-1}\prod_{i=0}^{h}\frac{B-l-i}{B-i} \therefore \alpha_{h+1,m} = \alpha_{h+1,0}\Big(\prod_{i=0}^{h}\frac{B-l-i}{B-i}\Big)^m \qquad \text{(B.23)}$$

Hence, replacing (B.23) into (B.22),

$$\alpha_{h,0} = \sum_{m=0}^{\infty} \alpha_{h+1,0} \Big(\prod_{i=0}^{h} \frac{B-l-i}{B-i} \Big)^m \frac{e^{-\rho}\rho^m}{m!} = \alpha_{h+1,0} \sum_{m=0}^{\infty} \Big[\underbrace{\prod_{i=0}^{h} \frac{B-l-i}{B-i}}_{\beta_{h+1}} \Big]^m \frac{e^{-\rho}\rho^m}{m!} \tag{B.24}$$

Let

$$\beta_h = \prod_{i=0}^{h-1} \frac{B-l-i}{B-i} = \frac{\binom{B-h}{l}}{\binom{B}{l}} \tag{B.25}$$

Therefore,

$$\alpha_{h,0} = \alpha_{h+1,0} \sum_{m=0}^{\infty} (\beta_{h+1}\rho)^m \frac{e^{-\rho}}{m!} = \alpha_{h+1,0} e^{-x} \sum_{m=0}^{\infty} (\beta_{h+1}\rho)^m \frac{e^{-\rho+x}}{m!} = \alpha_{h+1,0} e^{-x} \tag{B.26}$$

where $-\rho + x = -\beta_{h+1}\rho$,

$$x = \rho - \beta_{h+1}\rho = \rho(1-\beta_{h+1}) \tag{B.27}$$

So,

$$\alpha_{h,0} = \alpha_{h+1,0} \exp(-\rho(1-\beta_{h+1})) = \alpha_{h+1,0} \exp(-\rho(1 - \frac{\binom{B-h-1}{l}}{\binom{B}{l}})) \tag{B.28}$$

The solution of recursion (B.28) is

$$\alpha_{h,0} = \begin{cases} (i) \quad 1, & h=B \\ (ii) \quad \prod_{a=h}^{B-1} \exp(-\rho(1 - \frac{\binom{B-a-1}{l}}{\binom{B}{l}})), & 0 \le h < B \end{cases} \tag{B.29}$$

If $h = 0$,

$$\alpha_{0,0} = \prod_{a=0}^{B-1} \exp(-\rho(1 - \frac{\binom{B-a-1}{l}}{\binom{B}{l}})) = \exp(-\rho(\left(\sum_{a=0}^{B-1} 1\right) - \left(\sum_{a=0}^{B-1} \frac{\binom{B-a-1}{l}}{\binom{B}{l}}\right))) \quad \text{(B.30)}$$

The hockey stick pattern of the Pascal triangle, $\sum_{a=0}^{B-1} \binom{B-a-1}{l} = \binom{B}{l+1}$, together with (B.30), yields

$$\alpha_{0,0} = \exp\left(-\rho\left(B - \frac{\binom{B}{l+1}}{\binom{B}{l}}\right)\right) = \exp\left(-\rho\left(\frac{l(B+1)}{l+1}\right)\right) \quad \text{(B.31)}$$

B.11 Integrating the User Dynamics into the Lower Layer

Next, we provide an alternative description of our model which explicitly characterizes the evolution of the signature of each user in time.

Assume that each user selects uniformly at random one of the blocks among those that he does not have. Each user then contacts other users uniformly at random for opportunities to download the selected block. The time between contacts initiated by a specific (tagged) user is characterized by a Poisson process of rate μ. If the contacted user has the requested block, it is transfered immediately. Otherwise, the tagged user is instantaneously re-directed to the publisher, who then transfers the block. All transfers are assumed to be instantaneous.

The system described above can be fully characterized using the signatures introduced in our lower layer model. Let S be the lower layer state, i.e., S is a $\{0,1\}^{Bn}$ bit vector in case there are n users in the system, (see §3.2.2). $n_i(S)$ is the number of users in stage i when the system state is S. Let $h(S,u)$ be the stage of user u when the system state is S.

Let $\mathbf{\Omega}_n$ be the union of the state spaces of the lower layer model corresponding to an the upper layer in which there are n users in the system, $\mathbf{\Omega}_n = \bigcup_{\mathbf{n}:|\mathbf{n}|=n} \mathbf{\Omega}_{|\mathbf{n}|}$. Let $\mathbf{\Omega}'$ be the union of all possible state spaces of the lower layer model, $\mathbf{\Omega}' = \bigcup_{i=0}^{\infty} \mathbf{\Omega}_i$.

Starting from state S, let T_0 be the state resulting from an arrival and $T_{u,b}$ be the state resulting from user u concluding the download of block b.

Let $\mathbf{0}$ denote a vector of lenght B with all elements equal to zero; $\mathbf{e}_i$ denotes a vector with its i^{th} element equal to 1 and all other elements equal to zero; $S \setminus u$ denotes the bit vector S after the removal of the bits corresponding to user i (bits $(u-1)B+1$ up to uB). We assume that after a user leaves the system, the remaining users are re-indexed accordingly.

$$\begin{aligned} T_0 : \mathbf{\Omega}_n &\rightarrow \mathbf{\Omega}_{n+1} \\ S &\rightarrow S.\mathbf{0} \end{aligned} \tag{B.32}$$

(B.33)

If $h(S,u) < B-1$

$$\begin{aligned} T_{u,b} : \mathbf{\Omega}_n &\rightarrow \mathbf{\Omega}_n \\ S &\rightarrow S + \mathbf{e}_{B(u-1)+b} \end{aligned} \tag{B.34}$$

If $h(S,u) = B-1$

$$\begin{aligned} T_{u,b} : \mathbf{\Omega}_n &\rightarrow \mathbf{\Omega}_{n-1} \\ S &\rightarrow S \setminus u \end{aligned} \tag{B.35}$$

Note that transformation $T_{u,b}$ explicitly characterizes the evolution of the signature of each user in time. (B.34) corresponds to the download of a block and a signature update whereas (B.35) corresponds to a download completion.

The entries of the generator matrix $Q = (q(S, S') : S, S' \in \mathbf{\Omega}')$ are

$$q(S, T_0(S)) = \lambda \tag{B.36}$$

$$q(S, T_{u,b}(S)) = \mu/(B - h(S, u)) \tag{B.37}$$

Equations (B.36)-(B.37) provide an integrated description of the upper and lower layer models. This model description is similar in spirit to [40]. While Hajek and Zhu [40] are interested in studying the *stability* of a system that resembles the one described above, the system considered here is always stable, and we are interested in its *self-sustainability*.

B.12 Heterogeneous Capacities

Figure B.1 shows the simulation results considering heterogeneous peer upload capacities, for $B = 16$. The upload rate distribution was taken from the measured data used to generate Figure 1 in the BitTyrant study [70]. The mean capacity of the peers was adjusted to 39KBps. After the normalization, peers with capacity larger than 110KBps were excluded.

Figure B.1 was generated from eight simulations that lasted 100,000s each. That is why the confidence interval in Figure B.1 is smaller than the one in Figure 3.4 (generated from twenty independent simulations that lasted 10,000s each). Figure B.1 shows that our results did not qualitatively change for peers with heterogeneous capacities.

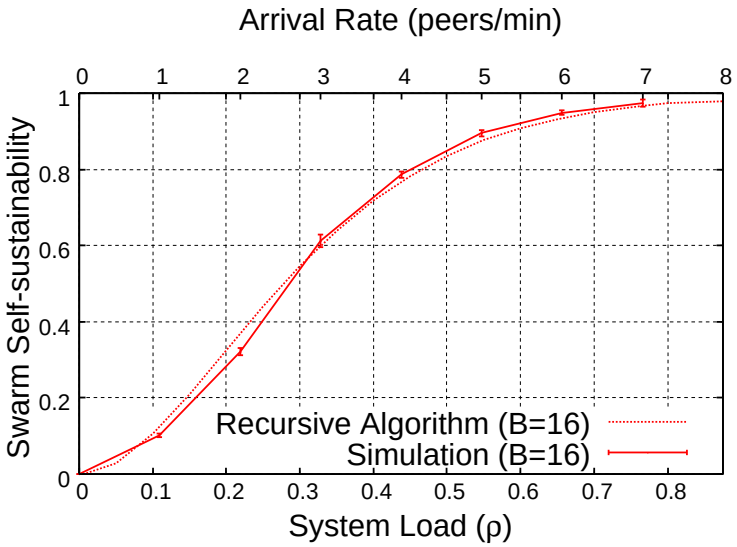


Figure B.1. Heterogeneous peer capacities

APPENDIX C

RECIPROCITY

C.1 Tightness of the Bound

In this section we prove that our bound of two on the loss of efficiency is tight. For this purpose, we work with a cycle of n nodes in which node v requests content c_{v-1} and owns content c_v, subtraction done modulo $|V| = n$. In particular, we show that it is not possible to satisfy all demands with less than $T_n = 2(n-1)$ transmissions.

In an indirect reciprocity schedule all nodes in the cycle can have their demands satisfied with one transmission. In the direct reciprocity schedule, as $n \to \infty$, if $T_n = 2(n-1)$ then $\frac{T_n}{n} \to 2$, i.e., at least one node transmits two contents. The loss of efficiency is two. In what follows, we show that $T_n \geq 2(n-1)$.

In any valid direct reciprocity schedule, let the fraction of content c_k sent from node i to j be denoted by $\Pi_{i,j,k}$. We denote by $d(k)$ the demand of node i,

$$d(i) = i - 1 \pmod{n} = \begin{cases} i-1, & 1 \leq i \leq n-1, \\ n-1, & i = 0 \end{cases} \tag{C.1}$$

In general, we denote by $r = x \pmod{n}$ the integer r such that (a) $0 \leq r \leq n-1$ and (b) $qn + r = x$ where $x, q \in \mathbb{Z}$ and $r, n \in \mathbb{N}$.

Note that even though each node may send redundant content across multiple links, at least one set of edge disjoint paths must exist from every source to every destination. Henceforth, we refer to the edge from node i to node j in which content c_k is transmitted as $\pi_{i,j,k}$, when considering only the paths in which non redundant

content is transfered, and $\pi^e_{i,j,k}$ when referring to the amount of additional content c_k transmitted through edge (i, j). The total amount of content c_k sent from i to j is

$$\Pi_{i,j,k} = \pi_{i,j,k} + \pi^e_{i,j,k} \tag{C.2}$$

Let $\mathcal{D}$ denote the domain of $\pi_{i,j,k}$ and $\pi^e_{i,j,k}$,

$$\mathcal{D} = \{(i, j, k) \in V \times V \times V : i \neq j, j \neq k\} \tag{C.3}$$

Any valid optimal direct reciprocity schedule satisfies the following properties:

1. the total amount of content transfered, $\sum_{(i,j,k)\in\mathcal{D}} \pi_{i,j,k} + \sum_{(i,j,k)\in\mathcal{D}} \pi^e_{i,j,k}$, is minimized;

2. the amount of content sent from node i to node j equals the amount sent from j to i;

3. if node i is neither a source nor a sink for content c_k, i.e., if $i \neq k$ and $d(i) \neq k$, the amount of (essential) content flowing towards i is equal to the amount sent by node i;

4. if node i is a source for content c_k, the amount of (essential) content c_k flowing from i is 1;

5. if node j is a destination for content c_k, the amount of content c_k flowing towards j is 1.

Equations (C.4)-(C.8) in the linear program below reflect properties 1-5, respectively.

$$\min \sum_{(i,j,k)\in\mathcal{D}} \pi_{i,j,k} + \sum_{(i,j,k)\in\mathcal{D}} \pi^e_{i,j,k} \tag{C.4}$$

$$\sum_{k=0,k\neq j}^{n-1} \pi_{i,j,k} + \pi^e_{i,j,k} = \sum_{k=0,k\neq i}^{n-1} \pi_{j,i,k} + \pi^e_{j,i,k} \qquad (i,j) \in V \times V, i \neq j \tag{C.5}$$

$$\sum_{j=0,k\neq j,i\neq j}^{n-1} \pi_{i,j,k} = \sum_{j=0,i\neq j,d(j)\neq k}^{n-1} \pi_{j,i,k} \qquad (i,k) \in V \times V, \; k \neq i, k \neq d(i) \tag{C.6}$$

$$\sum_{j=0,j\neq i}^{n-1} \pi_{i,j,i} = 1 \qquad i \in V \tag{C.7}$$

$$\sum_{i=0,j\neq i}^{n-1} \pi_{i,j,d(j)} = 1 \qquad j \in V \tag{C.8}$$

$$\pi_{i,j,k} \geq 0 \qquad (i,j,k) \in \mathcal{D} \tag{C.9}$$

$$\pi^e_{i,j,k} \geq 0 \qquad (i,j,k) \in \mathcal{D} \tag{C.10}$$

We now seek the dual of the above linear program. Letting $\underline{\mu}$ denote the vector of Lagrange multipliers,

$$\underline{\mu} = (\underline{w} \quad \underline{x} \quad \underline{y} \quad \underline{z}) \tag{C.11}$$

there are

1. $n(n-1)$ constraints of type (C.5), each one associated to a variable $w_{i,j}$;
2. $n(n-2)$ constraints of type (C.6), each one associated to a variable $x_{i,k}$;
3. n constraints of type (C.7), each one associated to a variable y_i;
4. n constraints of type (C.8), each one associated to a variable z_i.

The Lagrangian is given by $L(\underline{\pi}, \underline{\mu}) =$

$$= \sum_{(i,j,k)\in\mathcal{D}} \pi_{i,j,k} + \sum_{(i,j,k)\in\mathcal{D}} \pi^e_{i,j,k} \tag{C.12}$$

$$+ \sum_{\substack{i,j\in V\times V,\\ i\neq j}}^{n} w_{i,j}\Big(\sum_{k=0,k\neq j}^{n-1} \pi_{i,j,k} + \pi^e_{i,j,k} - \sum_{k=0,k\neq i}^{n-1} \pi_{j,i,k} + \pi^e_{j,i,k}\Big) \tag{C.13}$$

$$+ \sum_{\substack{i,k\in V\times V,\\ i\neq k, k\neq d(i)}} x_{i,k}\Big(\sum_{j=0,k\neq j,i\neq j}^{n-1} \pi_{i,j,k} - \sum_{j=0,i\neq j,d(j)\neq k}^{n-1} \pi_{j,i,k}\Big) \tag{C.14}$$

$$+ \sum_{i\in V} y_i\Big(\sum_{j=0,j\neq i}^{n-1} \pi_{i,j,i} - 1\Big) \tag{C.15}$$

$$+ \sum_{j\in V} z_j\Big(\sum_{i=0,j\neq i}^{n-1} \pi_{i,j,d(j)} - 1\Big) \tag{C.16}$$

Deriving the Lagrangian with respect to $\pi_{i,j,k}$ and setting the resulting expression to be smaller than or equal to zero we obtain

$$\begin{aligned} &w_{i,j}\mathbf{1}_{k\neq j} - w_{j,i}\mathbf{1}_{k\neq j} + x_{i,k}\mathbf{1}_{\substack{k\neq j\\ i\neq k, k\neq d(i)}} - x_{j,k}\mathbf{1}_{\substack{k\neq d(j)\\ j\neq k, k\neq d(i)}} + \\ &+ y_i\mathbf{1}_{k=i} + z_j\mathbf{1}_{k=d(j)} \leq 1, \quad (i,j,k)\in V\times V\times V, i\neq j \end{aligned}$$

Similarly, considering the derivative with respect to $\pi^e_{i,j,k}$,

$$w_{i,j}\mathbf{1}_{k\neq j} - w_{j,i}\mathbf{1}_{k\neq j} \leq 1, \quad (i,j,k)\in V\times V\times V, i\neq j$$

Therefore, the dual of the linear program shown in the beginning of this section is

$$\max \sum_{i=0}^{n-1} y_i + z_i = \min - \sum_{i=0}^{n-1} y_i + z_i \tag{C.17}$$

$$\begin{aligned} &- w_{i,j}\mathbf{1}_{k\neq j} + w_{j,i}\mathbf{1}_{k\neq j} - x_{i,k}\mathbf{1}_{\substack{k\neq j\\ i\neq k, k\neq d(i)}} + x_{j,k}\mathbf{1}_{\substack{k\neq d(j)\\ i\neq k, k\neq d(i)}} - \\ &\quad - y_i\mathbf{1}_{k=i} - z_j\mathbf{1}_{k=d(j)} \geq -1, \quad (i,j,k)\in\mathcal{D} \end{aligned} \tag{C.18}$$

$$- w_{i,j}\mathbf{1}_{k\neq j} + w_{j,i}\mathbf{1}_{k\neq j} \geq -1, \quad (i,j)\in V\times V, i\neq j \tag{C.19}$$

Denoting by $p^\star$ the optimal solution of the primal and given a feasible solution $\underline{\mu}$ for the dual, [14, Section 5.1.3] yields

$$p^\star \geq \begin{bmatrix} \mathbf{0}_{n(2n-3)} & -\mathbf{1}_{2n} \end{bmatrix} \underline{\mu} = -\sum_{i=0}^{n-1} y_i + z_i \tag{C.20}$$

where $\mathbf{0}_{n(2n-3)}$ denotes a vector of $n(2n-3)$ zeros and $\mathbf{1}_{2n}$ denotes a vector of $2n$ ones. Henceforth, to simplify the notation, we work solely with odd cycles (n is assumed to be odd). Note that working only with odd cycles suffices for our purposes.

We show that the following constitutes a feasible solution for the dual,

1. $y_i = z_i = -\frac{n-1}{n}$
2. $w_{i,j} = -w_{j,i} = \frac{n-2[(j-i)(\mathrm{mod}\ n)]}{2n}$
3. $x_{i,k} = \frac{2[(k-i)\ (\mathrm{mod}\ n)\ -(n-1)/2]}{n}$

which together with (C.20) yields

$$p^\star \geq \sum_{i=0}^{n-1} 2\frac{n-1}{n} = 2(n-1) \tag{C.21}$$

To show that 1-3 above lead to a feasible solution for the dual, we have to determine that constraints (C.18)–(C.19) are satisfied. It is easy to check that (C.19) is always satisfied. To show that (C.18) is also, we work case by case with the valid tuples $(i,j,k) \in V \times V \times V$.

- if $k = i$ and $i = d(j)$,

  then $w_{i,j} = \frac{n-2}{2n} = -w_{j,i}$, $x_{i,j} = x_{j,k} = 0$, and $y_i = z_i = -\frac{n-1}{n}$. Constraint (C.18) reduces to

$$-\frac{n-2}{n} + \frac{2(n-1)}{n} = 1 \geq -1 \tag{C.22}$$

- if $k = i$ and $i \neq d(j)$,

  constraint (C.18) reduces to

$$-w_{i,j}+w_{j,i} - y_i \geq -1 \tag{C.23}$$

  But $y_i = -(n-1)/n$ and $w_{i,j} = \frac{n-2(j-i)(\text{mod } n)}{2n}$ therefore $-w_{i,j}+w_{j,i} - y_i$ equals

$$\frac{(j-i)(\text{mod } n)}{n} - \frac{(i-j)(\text{mod } n)}{n} + \frac{n-1}{n} \tag{C.24}$$

  If $j > i$,

$$\begin{aligned} -w_{i,j}+w_{j,i} - y_i &= \frac{n-1+j-i-(n+i-j)}{n} = \\ &= \frac{2j-2i-1}{n} \geq -1 \end{aligned} \tag{C.25}$$

  If $j < i$,

$$-w_{i,j}+w_{j,i} - y_i = \frac{2n-1+2j-2i}{n} \geq -1 \tag{C.26}$$

- if $k \neq i$ and $k \neq d(j)$,

  constraint (C.18) reduces to

$$\begin{aligned} &-w_{i,j}+w_{j,i}-x_{i,k}\mathbf{1}_{k\neq d(i)}+x_{j,k}\mathbf{1}_{k\neq d(i)} \geq -1, \\ &(i,j,k) \in V \times V \times V, i \neq j, k \neq j \end{aligned} \tag{C.27}$$

  There are two subcases of interest,

1. $k \neq j$, $k \neq d(i)$, equation (C.27) reduces to

$$-w_{i,j} + w_{j,i} - x_{i,k} + x_{j,k} \geq -1,$$

If $j > i$, the expression above equals

$$2j - 2i - n - 2\Big[(k-i)(\text{mod } n) - (k-j)(\text{mod } n)\Big] \geq -n \qquad \text{(C.28)}$$

and if $j < i$,

$$2j - 2i + n - 2\Big[(k-i)(\text{mod } n) - (k-j)(\text{mod } n)\Big] \geq -n \qquad \text{(C.29)}$$

We start analyzing the case $j > i$ and the three possible subcases, (a) $k > j > i$, in which case (C.28) reduces to $-n \geq -n$, (b) $j > k > i$, when (C.28) reduces to

$$2j - 2i - n - 2\Big[k - i - (n + k - j)\Big] = n \geq -n \qquad \text{(C.30)}$$

and (c) $j > i > k$, when (C.28) reduces to

$$2j - 2i - n - 2\Big[(n + k - i) - (n + k - j)\Big] = -n \geq -n \qquad \text{(C.31)}$$

The case $j < i$ follows similarly.

2. $k \neq j$, $k = d(i)$, equation (C.27) reduces to

$$-w_{i,j} + w_{j,i} \geq -1 \qquad \text{(C.32)}$$

It can be verified using an argument similar to the one above that this inequality always holds.

- if $k \neq i$ and $k = d(j)$,

  constraint (C.18) reduces to

$$-w_{i,j} + w_{j,i} - x_{i,k} - z_j \geq -1,$$
$$(i, j, k) \in V \times V \times V, i \neq j \tag{C.33}$$

  If $j > i$,

$$2(j - i) - n - 2[(k - i) \pmod{n}] + 2(n - 1) \geq -n$$

  or

$$(j - i) - [(j - 1 - i) \pmod{n}] + 2(n - 1) \geq 0$$

  If $j - 1 - i \geq 0$ then $2n - 1 \geq 0$ and if $j - 1 - i < 0$ the left hand side of the above expression equals $n - 1 \geq 0$. If $j < i$,

$$2(j - i) + n - 2[(k - i) \pmod{n}] + 2(n - 1) \geq -n$$

  or

$$(j - i) - [(j - 1 - i) \pmod{n}] + n \geq 0$$

  If $j - 1 - i \geq 0$ then $n + 1 \geq 0$ and if $j - 1 - i < 0$ the left hand side of the above expression equals $-n + 1 + n = 1 \geq 0$.

Since all valid $(i, j, k) \in V \times V \times V$ tuples lead to feasible solutions for the dual, we conclude that the optimal value of the primal, $p^\star$, is $p^\star \geq 2(n - 1)$, which concludes the proof.

C.2 Splittable and unsplittable contents

Our main result assumes a fluid traffic model. Fractional allocations are realizable if the time frame over which nodes interact is longer than the one over which they

switch schedules. In those cases, nodes can split their uplink capacities across multiple paths.

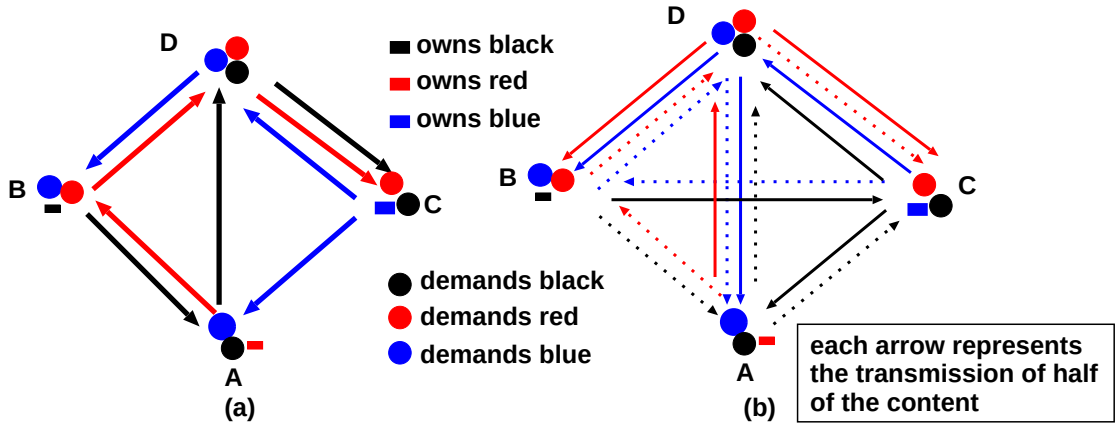


Figure C.1. Converting from (a) indirect reciprocity to (b) direct reciprocity.

Nevertheless, if nodes interact for very short intervals, with duration close to the one to transmit a block (assumed to be the minimum unit of exchange), there are workloads for which the loss of efficiency may be greater than two. Figure C.1(a) illustrates a network satisfying indirect reciprocity which can only be converted into a network satisfying direct reciprocity if nodes are allowed to allocate fractions of their outgoing flows to different paths (Figure C.1(b)).

C.3 Main Result

We now prove the lemmas used in our main result.

Lemma C.3.1. *Given an indirect reciprocity dissemination network N*

1. *N can be partitioned into a set of cycles*
2. *N is a dissemination network if and only if no cycle partition of N contains a cycle consisting solely of edge flows carrying the same content.*

Proof. 1) is is a consequence of the fact that the flow rate into each node equals the flow rate out. 2) follows from the definition of a dissemination network. □

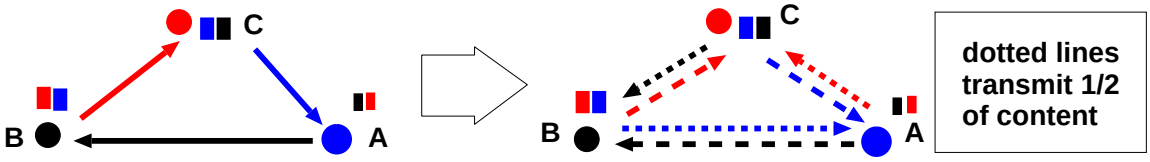


Figure C.2. Simple cycle converted from indirect to direct reciprocity.

Hall's theorem provides sufficient conditions for the existence of a perfect matching in a bipartite graph B, [13, Theorem III.7].

Theorem C.3.1 (Hall's theorem)**.** *A bipartite graph B with vertex sets V_1 and V_2 contains a perfect matching from V_1 to V_2 iff*

$$|s_B(S)| \geq |S| \quad \forall S \subseteq V_1.$$

Using Hall's theorem, we derive the following lemma,

Lemma C.3.2. *There exist at least two perfect matchings M_1, M_2 such that $M_1 \neq M_2$ whenever $n \geq 3$.*

Proof. First note that one matching is $M_1 = \{(v_i, u_i) : i = 1, \ldots, n\}$. We prove the existence of the second perfect matching by induction.

Basis step. Consider the case where $n = 3$. It is easily verified that $E = M_1 \cup \{(v_1, u_2), (v_2, u_3), (v_3, u_1)\}$ and that $M_2 = \{(v_1, u_2), (v_2, u_3), (v_3, u_1)\}$.

Induction step. Suppose that the lemma holds for all values of $n \leq k$. We establish it for $n = k + 1$. There are two cases.

Case i) There exists at least one $v \in V$ such that $|s_B(v)| = 1$. Wlog assume that this is v_n. Construct the bipartite graph $B' = (V \setminus \{v_n\}, U \setminus \{u_n\}, E \setminus E(V, \{u_n\}))$. It is easy to verify that the conditions needed for the inductive hypothesis to hold are satisfied and that there are two perfect matchings. Take the matching M' which includes at least one edge not of the form (v_i, u_i). The second perfect matching for B is $M_2 = M_2' \cup \{(v_n, u_n)\}$.

Case ii) $|s_B(v)| \geq 2$ for all $v \in V$, i.e., every node can serve content to at least two other nodes. We focus on the bipartite graph B' obtained by removing the edge (v_n, u_n) from E, i.e., $B' = (V, U, E' = E \setminus \{(v_n, u_n)\})$. If B' satisfies the conditions needed to apply Hall's Theorem, we are done as this ensures a perfect matching that does not include the edge (v_n, u_n). This matching, M_2, must also be a perfect matching for B. Suppose that the conditions are not satisfied and that there exists a set $S \subseteq V$ such that $|s_{B'}(S)| < |S|$. As a consequence of the construction of B', S takes the form $S = \{(v_{i_1}, \ldots, v_{i_2}, v_n\}$ and $\Gamma_{B'}(S) = \{u_{i_1}, \ldots, u_{i_2}\}$ for some $1 \leq i_1 < i_2 < n$. Note that if we define $V'' = \{(v_{i_1}, \ldots, v_{i_2}\}$ and $U'' = \{u_{i_1}, \ldots, u_{i_2}\}$, then $s_{B'}(V'') = U''$ and we can define yet another bipartite graph $B'' = (V'', U'', \{e \in E' : t(e) \in V'', h(e) \in U''\})$. Furthermore, it is easy to show that the assumptions needed to apply the inductive step holds and that B'' contains two perfect matchings, one of which, M'', which does not include all edges of the form (v_i, u_i). Thus the perfect matching for B', which is also a perfect matching for B is $M'' \cup \{(v_i, u_i) : i < i_1; i_2 < i\}$. Note that it differs from M_1, because of how M'' was constructed. This concludes the proof. □

C.4 Issues When Coping With Relays

We conjecture that the bound of two on the loss of efficiency due to direct reciprocity extends from realyless networks to a more general class of networks. Next, we illustrate some of the issues that arise when coping with relays. The presentation is not very rigorous in nature, its main goal being to point out a battery of networks that must be handled by any proof of the conjecture.

C.4.1 Backtracking

Figure C.3(a) shows an indirect reciprocity schedule of exchanges involving two cycles, one cycle of size two between nodes v_1 and v_2, and one cycle of size three

between nodes v_1, v_2 and v_3. There are three contents in the network, r, g, and b. Node v_1 requests contents r and g, node v_2 requests b and g, and node v_3 requests b. Under an indirect reciprocity schedule, node v_3 transmits one content, and nodes v_1 and v_2 transmit two contents each. Beginning with this set of exchanges, we seek to transform this schedule into a direct reciprocity schedule. One such schedule is shown in Figure C.3(b). Note that the number of transmissions in both schedules is five, so there is no loss of efficiency in this example.

In the transformation shown from Figure C.3(a) to Figure C.3(b), the first cycle removed is $v_2 \xrightarrow{r} v_1 \xrightarrow{b} v_2$, which is already a direct reciprocity cycle. The second cycle removed is $v_1 \xrightarrow{g} v_2 \xrightarrow{b} v_3 \xrightarrow{g} v_1$. This cycle is transformed into direct reciprocity using swarming, resulting in the schedule shown in Figure C.3(b).

In general, the order in which cycles are removed from the indirect reciprocity network may lead to scenarios in which circular forwarding of the same commodity occurs, also referred to as *conflicts*. For example, if cycle $v_1 \xrightarrow{g} v_2 \xrightarrow{r} v_1$ is removed first, then the transformation of the second cycle, $v_1 \xrightarrow{b} v_2 \xrightarrow{b} v_3 \xrightarrow{g} v_1$, yields a conflict as illustrated in Figure C.3(c). In this example, the conflict can be avoided by swapping edges $v_1 \xrightarrow{g} v_2$ and $v_1 \xrightarrow{b} v_2$ before reducing the second cycle. However, we are unaware of a systematic way to avoid conflicts. In particular, some conflicts might involve multiple hops (see Figure C.4).

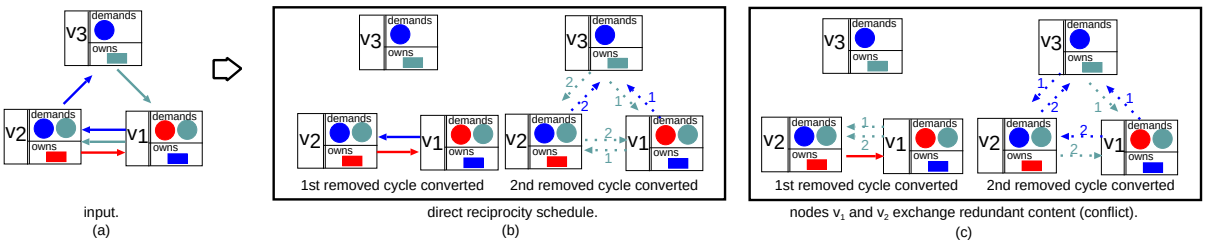


Figure C.3. Simple example of when backtracking solution works.

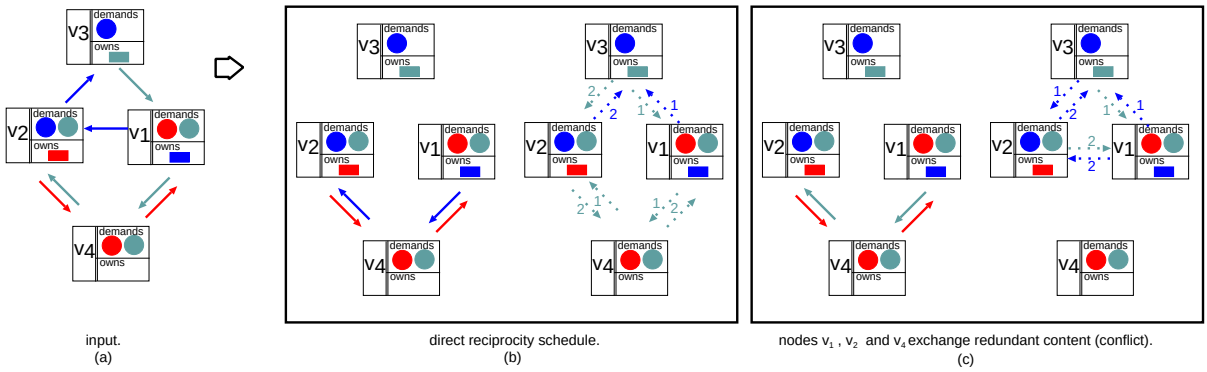


Figure C.4. Simple example of when backtracking solution does not work.

C.4.2 Other Issues

Figure C.5 illustrates in the upper part an indirect reciprocity network and in the lower part the correspoding direct reciprocity network. In what follows, we point out some of the subtleties that might come across during such conversion.

Splitting content: If the cycle ABDC (Figure C.6(a)) is removed first and converted into a direct reciprocity network as shown in Figure C.6(b) there is no other cycle that can be removed without leading to an inconsistency. That's because after the removal of cycle ABDC all nodes have contents red and blue, and some of them still need black without having any commodity to use for bartering.

Choosing cycles in order: If we change the order in which the indirect reciprocity cycles are selected, a solution is illustrated in Figure C.6(e)-(h). Note that there must be coordination between B and A regarding which part of content blue is sent using the first and third cycles.

Figure C.6(i)-(l) also illustrates the importance of the order at which cycles are treated. Note that network C.6(l) cannot be converted into one satisfying direct reciprocity without leading to an inconsistency. This is in contrast to the network shown in Figure C.2 which can be converted into one satisfying direct reciprocity.

Valid cycles and invalid cycles: Let a valid cycle be one such that a path of color c in this cycle starts in the source of color c and ends in a leaf of the tree that

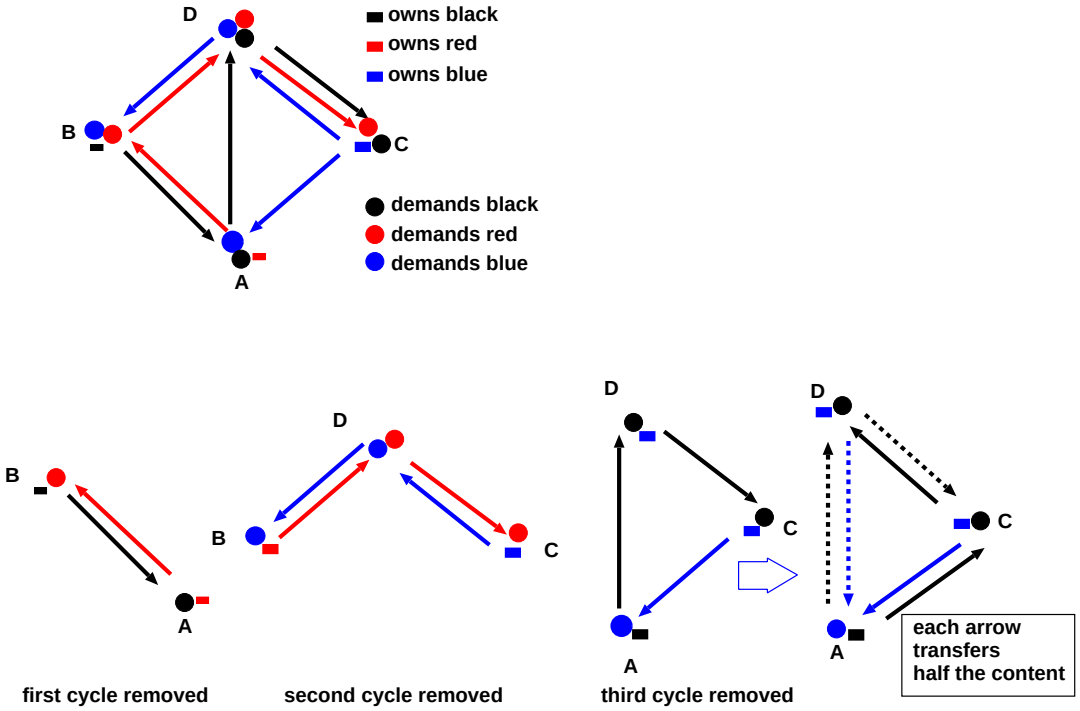


Figure C.5. Cycles removed in the right order.

disseminates c. Even if we stick only to valid cycles we may reach inconsistent states. Figure C.7 shows a network which, after the removal of two valid cycles, becomes inconsistent.

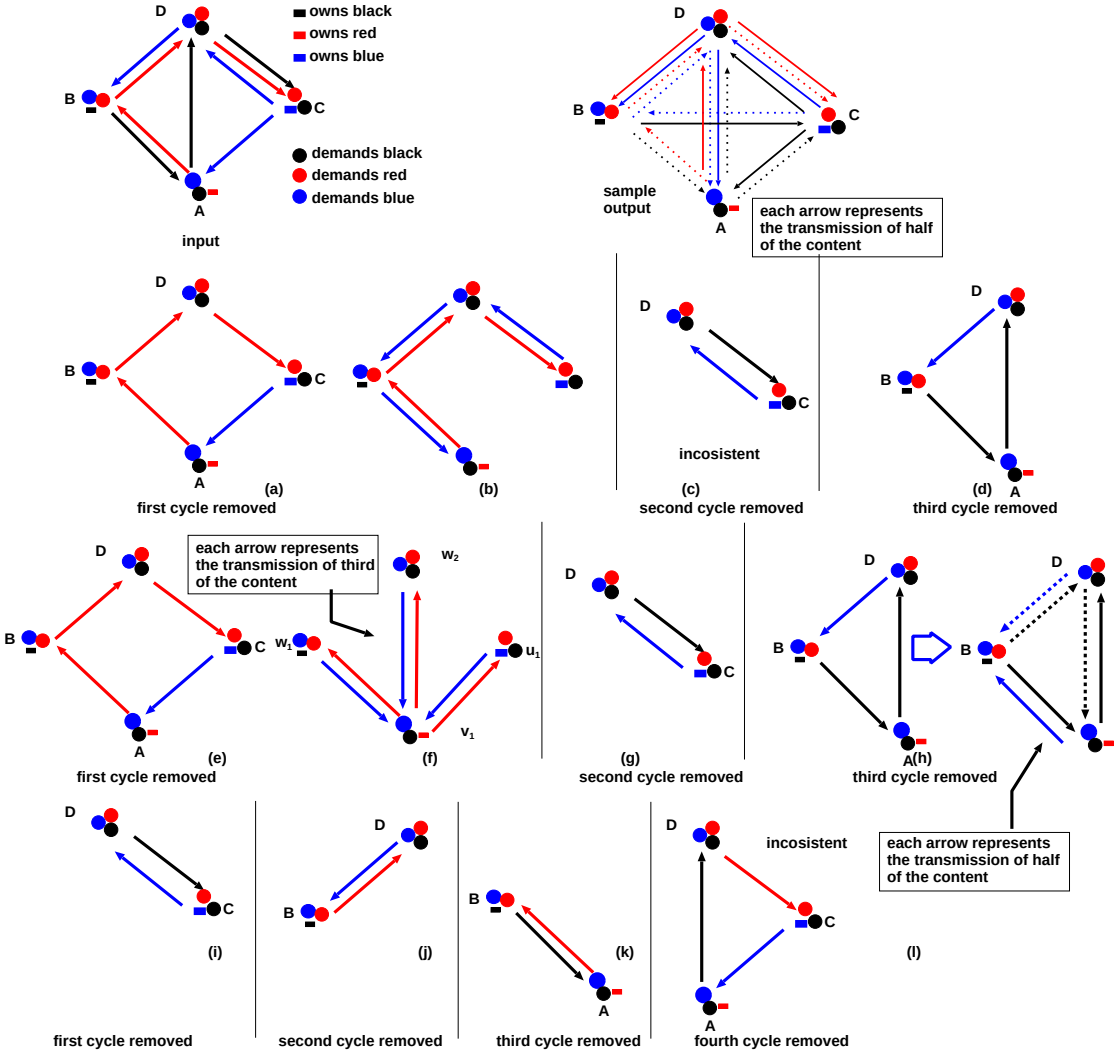


Figure C.6. The importance of removing cycles in the right order.

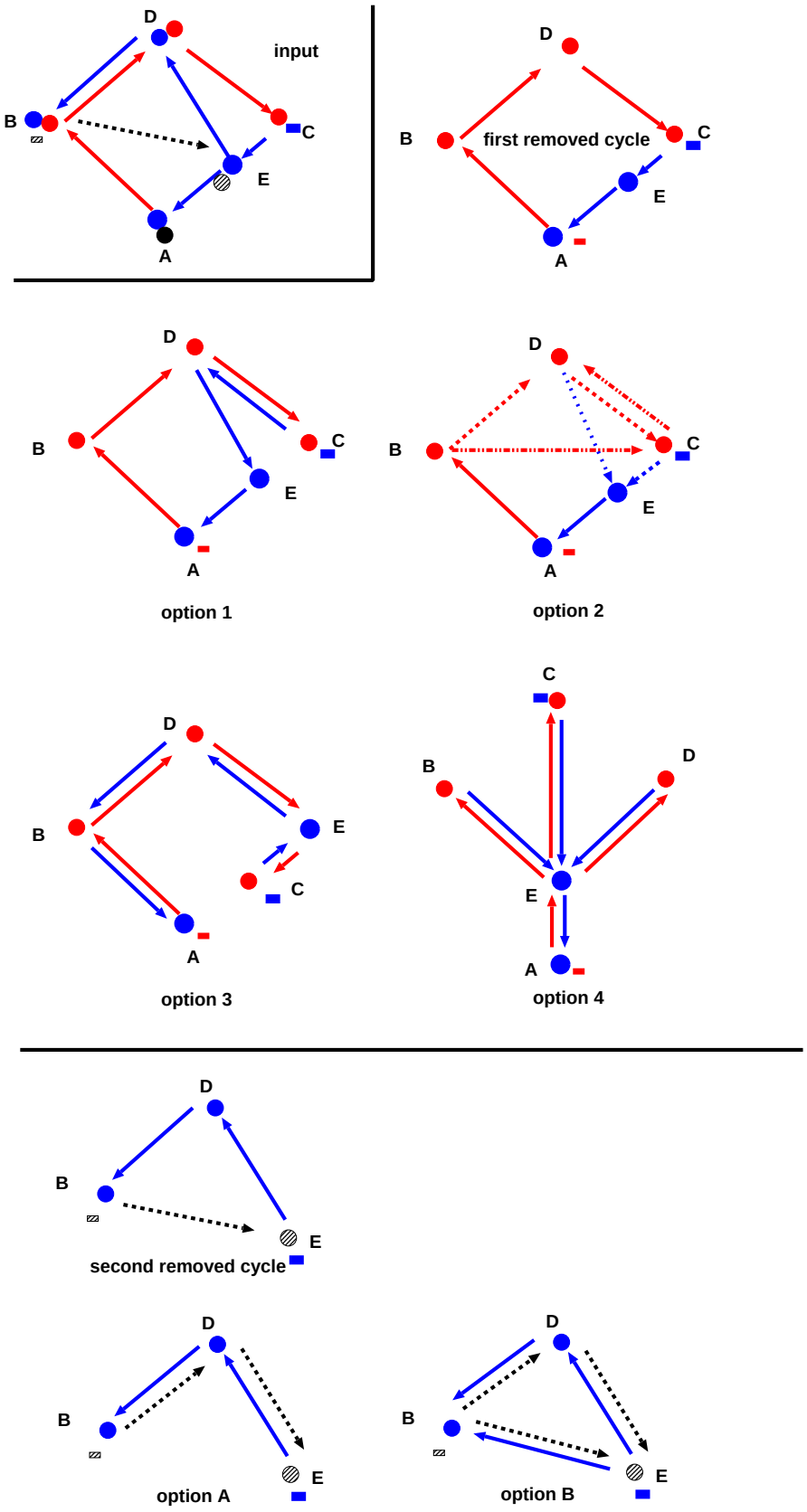


Figure C.7. Removing only valid cycles might still yield inconsistencies.

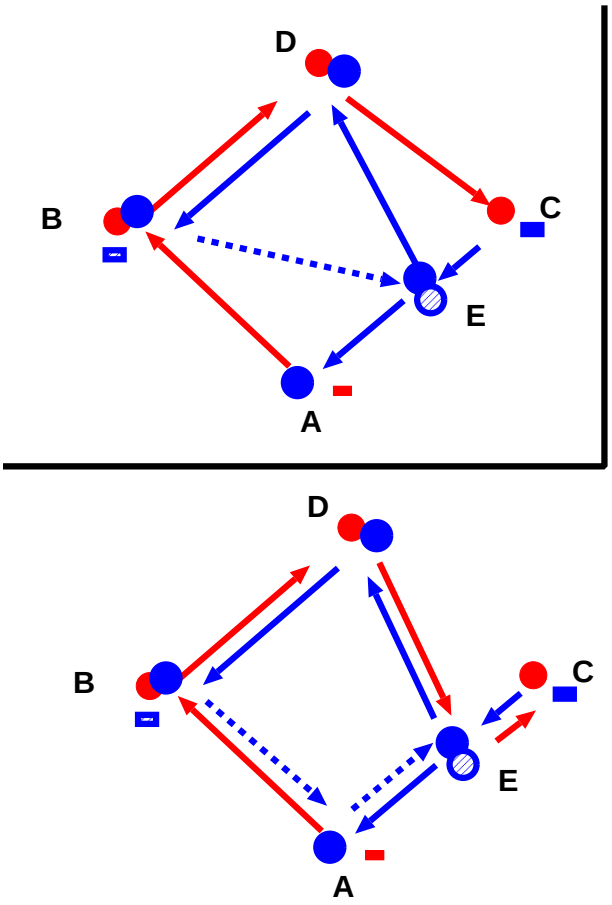


Figure C.8. A valid conversion from indirect to direct reciprocity. Removing the outer cycle from the indirect reciprocity network, though, would lead to inconsistencies.

C.5 The Feasibility of Indirect Reciprocity Schedules I

Consider a set of users, V, and a set of contents, C. Let D_v denote the set of contents that node v demands and, as before, $s(v)$ the set of contents that node v has, $s(v), D_v \subset C$. Assume also that node $v \in V$ has a bandwidth equal to D_v. This can be interpreted as if each node has unit bandwidth to transfer a file. We now ask the following question: when does there exist an indirect reciprocity assignment for such a workload?

We construct the following capacitated directed graph.

$$\begin{aligned} V &= \{s,t\} \cup V_1 \cup V_2 \cup V_3 \\ V_1 &= \{(v, D_v, s(v)) : v \in V\} \\ V_2 &= \{(v,c) : c \in D_v, v \in V\} \\ V_3 &= \{u_c, w_c : c \in C\} \end{aligned}$$

and

$$\begin{aligned} E &= \{(s,v) : v \in V_1\} \\ &\quad \cup \{((v, D_v, s(v)), (v', c)) : \\ &\qquad (v, D_v, s(v)) \in V_1, (v', c) \in V_2, c \in s(v)\} \\ &\quad \cup \{((v, D_v, s(v)), u_c) : \\ &\qquad (v, D_v, s(v)) \in V_1, c \in D_v \cup s(v)\} \\ &\quad \cup \{(u_c, w_c) : c \in C\} \\ &\quad \cup \{(w_c, (v,c)) : (v,c) \in V_2, c \in D_v\} \\ &\quad \cup \{(v,t) : v \in V_2\} \end{aligned}$$

Note that each node in V has a counterpart in V_1 with an indication of what content it can serve, $D(v) \cup s(v)$. Similarly, each node in V_2 consists of a node-

demand pair; a unit flow to $(v, c) \in V_2$ represents satisfying the demand of node v for content c. We will observe shortly, the set of nodes V_3 ensures that content flows from the sources into the network.

The edge between s and a node $v \in V_1$ is given a capacity equal to the number of demands associated with v, $|D_v|$. Edges between V_1 and V_3, between V_3 and V_2, and between V_1 and V_2 are given infinite capacities. Edges between nodes in V_2 and t have capacity one. Last, the capacity of edge (u_c, w_c) is $n_c - 1$ where $n_c = \sum_{v \in V} \mathbb{1}_{c \in D_v}$ is the number of users demanding c. Note that the capacity $n_c - 1$ edges between nodes in $\{u_c\}$ and $\{w_c\}$ restrict non source nodes from supplying all copies of c to nodes demanding it. In particular, at least one unit of content has to traverse a link between a node in V_1 and a node in V_2; this occurs only if the corresponding nodes in V consist of one source of content c and a demander of that content. We make the following claim.

Theorem C.5.1. *The min cut of the above graph is $\sum_{c \in C} n_c$, iff there exists an indirect reciprocity schedule that satisfies all of the demands of the users in V.*

Proof. ($\Rightarrow$) If there exists an indirect reciprocity schedule that satisfies all users in V, it is easy to check that the corresponding flow satisfies all the constraints imposed above and that the min cut of the graph is $\sum_{c \in C} n_c$.

($\Leftarrow$) The min cut of the above graph is $\sum_{c \in C} n_c$ in two cases,

- the flow corresponds to a legal indirect reciprocity schedule, and the result holds;
- the flow corresponds to an illegal indirect reciprocity schedule, which needs to be ruled out. The schedule is illegal if it involves self-loops, i.e., nodes sending content to themselves. We now show how to convert such schedule into a legal one. If node A sends content c to itself, there are also nodes T and U such that T sends content c to U. That is because the capacity of edge (u_c, w_c) being equal to $n_c - 1$ yields at least one copy of the content being sent directly from a

source to a destination. To rule out the self loop involving A, T sends content c to A who sends it to U.

To illustrate how illegal schedules are ruled out in the proof above, consider the network shown in Figure C.9(a). The corresponding flow is shown in Figures C.10 and C.9(b). To remove the self loop of node 3 (Figure C.9(b)), node 1 sends content black to 3 that forwards it to 4 (Figure C.9(a)).

C.6 The Feasibility of Indirect Reciprocity Schedules II

We have the following alternative feasibility conditions for indirect reciprocity. The set of demanded contents is C. Assume node capacity c_u for each user u, streaming rate r_i for commodity i, denote R_i the set of users who want to receive commodity i, and S_i the set users who own commodity i initially. Then all commodity demands can be satisfied provided the following conditions hold:

For all set I of commodities, the following two conditions are necessary for feasibility,

1. $\sum_{u \in \cup_{i \in I}\{S_i \cup R_i\}} c_u \geq \sum_{i \in I} r_i |R_i|$
2. $\sum_{u \in \cup_{i \in I} S_i} c_u \geq \sum_{i \in I} r_i$

Assume further that

3) $c_u = \sum_{i \in D_u} r_i$ (where D_u is the set of items demanded by u),

Conditions 1-3 are necessary and sufficient for indirect reciprocity to hold.

From here on we assume $r_i = 1$, $i = 1, \ldots, |C|$. The three conditions above imply,

1. the joint capacity of sources and relays for a given set of contents is greater than the demand for that set of contents (if $r_i = 1$ for all i, this condition is vacuous);
2. there is a perfect matching between demanded contents and sources (see below);

3. the capacity of a node is equal to the number of items demanded by that node.

To see why the above conditions are sufficient, we construct the following capacitated directed graph,

$$\begin{aligned} V &= \{s, t\} \\ V_1 &= \{(c) : c \in C\} \\ V_2 &= \{(v, C_v, d) : d \in D_v, d \text{ is owned by at least one node}\} \end{aligned}$$

and

$$\begin{aligned} E &= \{((c), (v', C_v, d)) : (c) \in V_1, (v', C_v, d) \in V_2, c \in C_v\} \\ &\quad \cup \{(s, v), v \in V_1\} \\ &\quad \cup \{(u, t), u \in V_2\} \end{aligned}$$

Each element in V_2 consists of a node-ownership-demand triple; a unit flow to $(v, C_v, d) \in V_2$ represents node v satisfying demand of content c and receiving content d as a compensation. All edges have capacity one. We make the following claim.

Theorem C.6.1. *The min cut of the above graph is $|C|$ iff there exists an indirect reciprocity schedule that satisfies all of the demands of the users in V.*

Proof. Follows from Hall's theorem [13, Theorem III.7] and condition 2 above. Note that relays can be arbitrarily added to the network associated to the graph above.

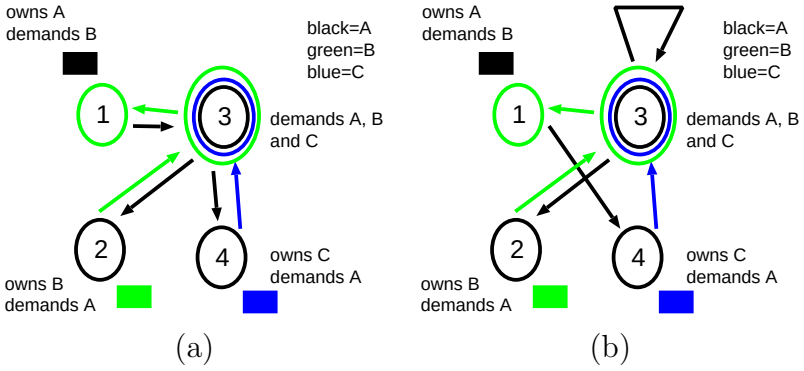


Figure C.9. Legal and illegal schedules.

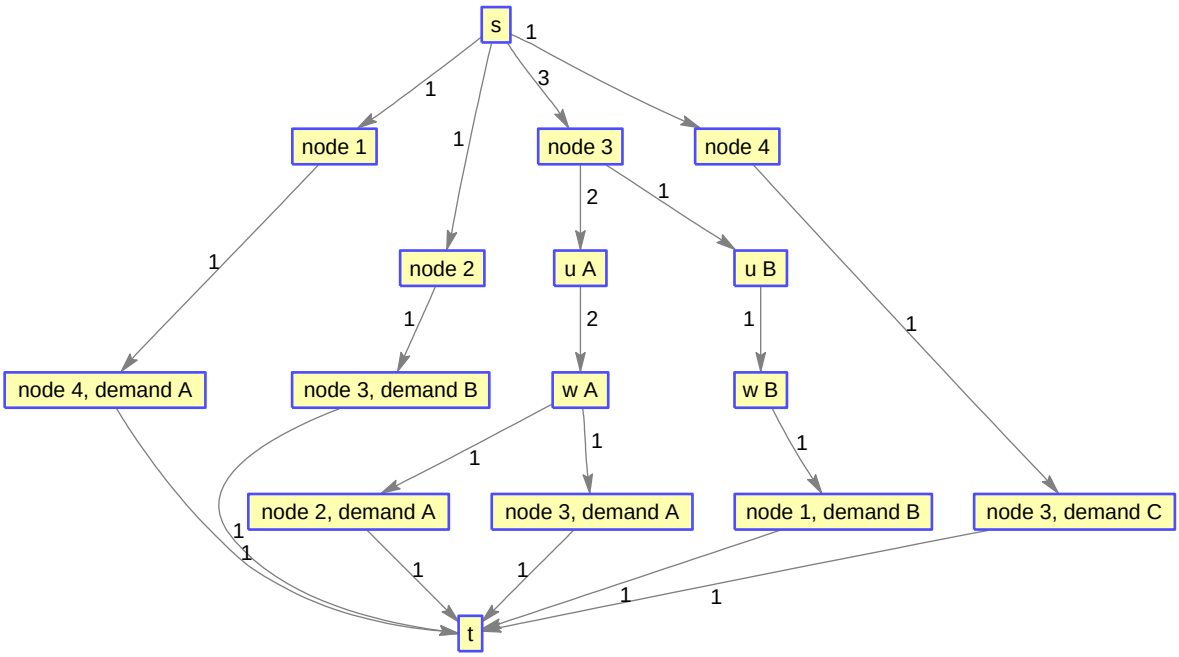


Figure C.10. Network flow corresponding to an illegal schedule (Figure C.9(b)) but which can be converted into a legal one (Figure C.9(a)).

APPENDIX D

EQUILIBRIA AND COMPETITION

D.1 Equilibrium uniqueness

In this section we verify that the hypothesis of Propositions 5.4.3 and 5.4.4 are satisfied respectively for functions (5.31) and (5.30). In order to check properties like convexity/concavity or monotonicity we can ignore multiplicative factors or scaling factors as long as they are positive. For (5.31) we consider $f_1(x) = 1 - e^{-x}$. The corresponding marginal cost is convex, because the the third derivative is $f_1'''(x) = e^{-x} > 0$.

For (5.30) we consider $f_2(x) = xe^{-x}$. The derivatives of interest are: $f_2''(x) = e^{-x}(x-2)$ and $f_2'''(x) = e^{-x}(3-x)$. f_2 is concave in $[0,2]$ and convex in $(2,\infty)$. Its derivative is convex in $(0,3)$.

BIBLIOGRAPHY

[1] Abraham, David J., Blum, Avrim, and Sandholm, Tuomas. Clearing algorithms for barter exchange markets: Enabling nationwide kidney exchanges. In *ACM Conference on Electronic Commerce* (2007).

[2] Adams, W., and Yellen, J. Commodity bundling and the burden of monopoly. *Quarterly Journal of Economics 90* (1976), 475–498.

[3] Altman, E., Nain, P., and Bermond, J.C. Distributed storage management of evolving files in delay tolerant ad hoc networks. In *INFOCOM* (2009).

[4] Amazon. Using BitTorrent with Amazon S3. http://aws.amazon.com/.

[5] Anderson, Chris. *The Long Tail: Why the Future of Business is Selling Less of More*. Hyperion, 2006.

[6] Aperjis, C., Freedman, M. J., and Johari, R. Peer assisted content distribution with prices. In *CONEXT* (2008).

[7] Apple. itunes, 2011. `http://www.apple.com/itunes/`.

[8] Astri. Astri Labs. `http://www.astri.org/`.

[9] Bakos, and Brynjolfsson. Bundling information goods: Pricing, profits and efficiency. *D. Hurley, B.Kahin, and H. Varian (Eds.), The Economics of Digital Information Goods* (1998).

[10] Bakos, Y., and Brynjolfsson, Erik. Bundling information goods. *Management Science 45* (1999), 1613–1630.

[11] Bharambe, A., Herley, C., and Padmanabhan, V. Analyzing and improving BitTorrent network performance mechanisms. In *Infocom* (2006).

[12] Bharambe, Ashwin R., Herley, Cormac, and Padmanabhan, Venkata N. Some observations on bittorrent performance. In *ACM SIGMETRICS'05* (2005).

[13] Bollobas, B. *Modern Graph Theory*. Springer Verlag, 1998.

[14] Boyd, S., and Vandenberghe, L. *Convex Optimization*. Cambridge University Press, 2004.

[15] Browne, S., and Steele, J. Transient behavior of coverage processes with applications to the infinite server queue. *J. Appl. Prob. 30 (3)* (1993), 589–601.

[16] Burger, A. P., and van Vuuren, J. H. Balanced minimum covers of a finite set. *Discrete Mathematics 307* (2007), 2853–2860.

[17] Carlsson, N., Eager, D., and Mahanti, A. Using torrent inflation to efficiently serve the long tail in peer-assisted content delivery systems. In *Networking* (2010).

[18] Chaintreau, Augustin, Boudec, Jean-Yves Le, and Ristanovic, Nikodin. The age of gossip: spatial mean field regime. In *SIGMETRICS* (2009).

[19] Chandy, and Misra. Distributed deadlock detection. *Transactions on Computer Systems*, 2 (1983), 144–156.

[20] Cohen, Bram. Incentives build robustness in BitTorrent. In *P2PEcon* (2003).

[21] Cormen, Thomas, Leiserson, Charles, Rivest, Ronald, and Stein, Clifford. *Introduction to Algorithms*. McGraw Hill, 2002.

[22] Cox, D., and Miller, H. *The Theory of Stochastic Processes*. Chapman and Hall, 1965.

[23] Crescenzo, A., Giorno, V., Nobile, A., and Ricciardi, L. A note on birth-death processes with catastrophes. *Statistics and Probability Letters 78* (2008), 2248–2257.

[24] Crescenzo, A., Giorno, V., Nobile, A., and Ricciardi, L. On the first-visit-time problem for birth and death processes with catastrophes. *arXiv* (2008).

[25] Croson, David, and Saunders, Adam. Competition and cooperation in the bundled software market. In *Workshop on Information Systems and Economics* (2004).

[26] Cuevas, R., Kryczka, M., Cuevas, A., Kaune, S., Guerrero, C., and Rejaie, R. Is content publishing in bittorrent altruistic or profit-driven? In *CoNEXT* (2010).

[27] Das, S., Tewari, S., and Kleinrock, L. The case for servers in a peer-to-peer world. In *ICC* (2006).

[28] de Souza e Silva, E., Leao, R. M. M., and Figueiredo, D. R. An integrated modeling environment for computer systems and networks. *Performance Evaluation Review 36*, 4 (2009), 64–69.

[29] Elberse, Anita. Bye bye bundles: The unbundling of music in digital channels. In *Harvard Business School Technical Report* (2010).

[30] eMule project. http://www.emule-project.net/.

[31] Fan, Bin, Chiu, D., and Lui, John. Stochastic analysis and file availability enhacement for bt-like sharing systems. In *WQoS* (2006), pp. 85–91.

[32] Fan, Bin, Chiu, Dah-Ming, and Lui, John. Stochastic differential equation approach to model peer-to-peer. In *ICC* (2006).

[33] Fay, and Mackie-Mason. Competition between firms that bundle information goods. In *Proceedings of the 27th Annual Telecommunications Policy Research Conference* (1999).

[34] Figueiredo, Dimitri De, Venkatachalam, Balaji, and Wu, S. Bounds on the performance of p2p networks using tit-for-tat strategies. In *IEEE Conference on P2P Systems* (2007), pp. 11–18.

[35] Fuerderer, Ralph, Herrmann, Andreas, and Wuebker, Georg. *Optimal Bundling*. Springer, 1999.

[36] Galidakis, I. N. On an application of lambert's w function to infinite exponentials. *Complex Variables and Elliptic Equations 49* (2004), 759–780.

[37] Garey, Michael, and Johnson, David. *Computers and Intractability*. Freeman and Company, 1979.

[38] Gradshteyn, I., and Ryzhik, I. *Tables of Integrals, Series and Products (7th edition)*. Academic Press, 2007.

[39] Guo, Lei, Chen, Songqing, Xiao, Zhen, Tan, Enhua, Ding, Xiaoning, and Zhang, Xiaodong. A performance study of Bittorrent-like peer-to-peer systems. *JSAC 25(1)* (2007), 155–169.

[40] Hajek, B., and Zhu, J. The missing piece syndrome in peer-to-peer communication. In *ISIT* (2010).

[41] Han, J., Chung, T., Kim, S., Kim, H., Kwon, T., and Choi, Y. An empirical study on content bundling in bittorrent swarming system. In *arXiv:1008.2574v1* (2010).

[42] Han, Jinyoung, Chung, Taejoong, Kim, Hyunchul, Kwon, Ted, and Choi, Yanghee. Systematic support for content bundling in bittorrent swarming. In *INFOCOM* (2010).

[43] Haverkort, B. R., Marie, R., and Rubino, G. *Performability Modelling*. Wiley, 2001.

[44] Ioannidis, S., and Marbach, P. On the design of hybrid peer-to-peer systems. In *SIGMETRICS* (2008).

[45] Ioannidis, Stratis, Massoulie, Laurent, and Chaintreau, Augustin. Distributed caching over heterogeneous mobile networks. In *SIGMETRICS* (2010).

[46] IPOQUE. Ipoque internet study. `http://www.ipoque.com/news_&_events/internet_studies/internet_study_2007`.

[47] Kaune, Sebastian, Cuevas, Ruben, Tyson, Gareth, Mauthe, Andreas, Guerrero, Carmen, and Steinmetz, Ralf. Unraveling bittorrent's file unavailability: Measurements, analysis and solution exploration. In *arXiv:0912.0625v1* (2009).

[48] Kolchin, V., Sevastyanov, B., and Chistyakov, V. *Random Allocations*. V. H. Winston and Sons, 1978.

[49] Legout, A., Liogkas, N., Kohler, E., and Zhang, L. Clustering and sharing incentives in bittorrent systems. In *ACM SIGMETRICS'07* (2007).

[50] Legout, Arnaud, Liogkas, Nikitas, and Kohler, Eddie. Rarest first and choke algorithms are enough. In *IMC* (2006).

[51] Leskela, L., Robert, P., and Simatos, F. Stability properties of linear file sharing networks. *arXiv.org* (2009).

[52] Lev-tov, N., Carlsson, N., Li, Zongpeng, Williamson, C., and Zhang, Song. Dynamic file-selection policies for bundling in bittorrent-like systems. In *IWQoS* (2010).

[53] Levin, Dave, LaCurts, Katrina, Spring, Neil, and Bhattacharjee, Bobby. Bittorrent is an auction: Analyzing and improving Bittorrent incentives. In *SIGCOMM* (2008).

[54] Mandjes, M.R.H., and Zuraniewsky, P. M/g/infinity transience, and its applications to overload detection. *CWI Technical Report* (2009).

[55] Massoulie, L., and Twigg, A. Rate optimal schemes for peer-to-peer live streaming. *Performance Evaluation 65* (2008), 804–822.

[56] Massoulie, Laurent, and Vojnovic, Milan. Coupon replication systems. In *SIGMETRICS* (2006).

[57] Menasche, Daniel Sadoc, Massoulie, Laurent, and Towsley, Don. Reciprocity and barter in peer-to-peer systems. In *INFOCOM* (2010).

[58] Menasche, Daniel Sadoc, Neglia, Giovanni, Towsley, Don, and Zilberstein, Shlomo. Strategic reasoning about bundling in swarming systems. In *GameNets* (2009).

[59] Menasche, Daniel Sadoc, Rocha, Antonio, de Souza e Silva, Edmundo, Leao, Rosa Maria Meri, Towsley, Don, and Venkataramani, Arun. Modeling chunk availability in peer-to-peer swarming systems. In *Performance Evaluation Review* (2009).

[60] Menasche, Daniel Sadoc, Rocha, Antonio, Li, Bin, Towsley, Don, and Venkataramani, Arun. Content availability and bundling in swarming systems. In *CONEXT* (2009).

[61] Menasche, Daniel Sadoc, Rocha, Antonio A., de Souza e Silva, Edmundo, Leao, Rosa M., Towsley, Don, and Venkataramani, Arun. Estimating self-sustainability in peer-to-peer swarming systems. *Performance Evaluation 67* (2010), 1243–1258.

[62] Mirakhmedov, Saidbek S., and Mirakhmedov, Sherzod M. On asymptotic expansion in the random allocation of particles by sets. *Journal of Theoretical Probability* (2009).

[63] Mundinger, J., Weber, R., and Weiss, G. Optimal scheduling of peer-to-peer file dissemination. *Journal of Scheduling* (2008).

[64] Nalebuff, Barry. Bundling as an entry barrier. *The Quarterly Journal of Economics 119* (2004), 159–187.

[65] Norros, I., Prabhu, B. J., and Reittu, H. On uncoordinated file distribution with non-altruistic downloaders. In *ITC-20* (2007).

[66] Nowak, M A, and Sigmund, K. Evolution of indirect reciprocity. *Nature 437* (2005), 1291–1298.

[67] Page, Will. More long tail debate: mobile music no, search yes, 2008. `http://longtail.typepad.com/`.

[68] Perloff, Jeffrey. *Microeconomic: Theory and Applications with Calculus*. Pearson International, 2008.

[69] Peterson, R. S., and Sirer, E. G. Antfarm: efficient content distribution with managed swarms. In *NSDI* (2009).

[70] Piatek, Michael, Isdal, Tomas, Anderson, Thomas, Krishnamurthy, Arvind, and Venkataramani, Arun. Do incentives build robustness in Bittorrent? In *NSDI* (2007).

[71] Piatek, Michael, Isdal, Tomas, Krishnamurthy, Arvind, and Anderson, Thomas. One hop reputations for peer to peer file sharing workloads. In *NSDI* (2008).

[72] Pyka, Andreas, and Scharnhorst, Andrea. *Innovation Networks: New approaches in modelling and analysis*. Springer, 2009.

[73] Qiu, D., and Srikant, R. Modeling and performance analysis of Bittorrent-like peer-to-peer networks. In *SIGCOMM* (2004).

[74] Ramachandran, A., das Sarma, A., and Feamster, N. Bitstore: An incentive compatible solution for blocked downloads in bittorrent. In *Proc. Joint Workshop on The Economics of Networked Systems and Incentive-Based Computing* (2007).

[75] Reich, Joshua, and Chaintreau, Augustin. The age of impatience: optimal replication schemes for opportunistic networks. In *CONEXT* (2009).

[76] Reittu, Hannu, and Norros, Ilkka. Toward modeling of a single file broadcasting in a closed network. In *IEEE Workshop on Spatial Stochastic Models in Wireless Networks* (2007).

[77] Resnikoff, Paul. Bundling Wins: Coachella's Three Day Sellout... , 2010. `http://www.digitalmusicnews.com/stories/041510coachella`.

[78] Rimac, Ivica, Elwalid, Anwar, and Borst, Sem. On server dimensioning for hybrid peer-to-peer content distribution networks. In *P2P'08* (2008).

[79] Roijers, F., Mandjes, M., and Berg, J. Analysis of congestion periods of an m/m/inf-queue. *Performance Evaluation 64* (2007), 737–754.

[80] Salinger, M. A. A graphical analysis of bundling. *Journal of Business 68* (1995), 85–98.

[81] Schneider, C. Symbolic summation assists combinatorics. *Sem.Lothar.Combin. 56* (2007), 1–36.

[82] Sherman, A., Nieh, J., and Stein, C. Fairtorrent: Bringing fairness to peer-to-peer systems. In *CONEXT* (2009), pp. 133–144.

[83] Shor, Mikhael. Dictionary of game theory terms, game theory .net, 2009. `http://www.gametheory.net/dictionary/`.

[84] Simatos, Florian, Robert, Philippe, and Guillemin, Fabrice. Analysis of a queueing system for modeling a file sharing principle. In *SIGMETRICS* (2008).

[85] Sirivianos, M., Park, J. H., Chen, R., and Yang, X. Free-riding in bittorrent networks with the large view exploit. In *IPTPS* (2007).

[86] Susitaival, Riikka, Aalto, Samuli, and Virtamo, Jorma. *Lecture Notes in Computer Science*. Springer, 2006, ch. Analyzing the dynamics and resource usage of peer-to-peer file sharing by a spatio-temporal model.

[87] Takacs, Lajos. *Introduction to the Theory of Queues*. Oxford University Press, 1962.

[88] Technicolor. Technicolor Research Labs. http://www.thlabs.net/.

[89] Tian, Y., Wu, D., and Ng, K.-W. Analyzing multiple file downloading in bittorrent. In *ICCP* (2006).

[90] Titletrader, swap your stuff. http://www.titletrader.com/.

[91] Torrent Finder. http://torrent-finder.com/.

[92] Torrent Freak. Twitter uses BitTorrent for server deployment. http://torrentfreak.com/twitter-uses-bittorrent-for-server-deployment-100210/.

[93] Towsley, Don. The internet is flat. In *PODC* (2008).

[94] Ubuntu. Download Ubuntu using BitTorrent. http://torrent.ubuntu.com:6969/.

[95] Veciana, G., and Yang, X. Service capacity of peer-to-peer networks. In *Infocom* (2004).

[96] Wikipedia. Boole's inequality. http://en.wikipedia.org/wiki/Boole%27s_inequality.

[97] Wikipedia. Entry on Digital Rights Management, 2010.

[98] Wilf, Herbert. *Generatingfunctionology*. Acad. Press, 1994.

[99] Wilson, Robert. Strategic models of entry deterrence. In *Handbooks in Economics* (1994).

[100] Wong, Sulan, and Altman, Eitan. Restricting internet access: Ideology and technology. In *Communication Systems and Networks* (2010).

[101] Wu, Di, Liu, Yong, and Ross, Keith. Queueing network models for multi-channel p2p live streaming systems. In *INFOCOM* (2009).

[102] Wu, Di, Liu, Yong, and Ross, Keith. Modeling and analysis of multi-channel p2p live video systems. *IEEE/ACM Transactions on Networking* (2011).

[103] Xiong, L., and Liu, L. Peertrust: Supporting reputation-based trust for p2p electronic communities. *IEEE Tran. Knowl. Data Eng.* (2004).

[104] Yang, Y., Chow, A., and Golubchik, L. Multi-torrent: a performance study and applications. *Int. J. Advanced Media and Communication 4(1)* (2010).

[105] Zeilberger, Doron. The method of creative telescoping. *Journal of Symbolic Computation archive 11* (1991), 195–204.

[106] Zheng, Xiaoying, Cho, Chunglae, and Xia, Ye. Algorithms and stability analysis for universal-swarming-based content distribution.

[107] Zhou, Xia, Ioannidis, Stratis, and Massoulie, Laurent. On the stability and optimality of universal swarms. In *SIGMETRICS* (2011).

[108] Zhu, J., and Hajek, B. Stability of a peer-to-peer communication. In *PODC* (2011).