Département Télécommunications et Médiainformatique
Université des Sciences Techniques et Économiques de Budapest

Eurecom
Télécom ParisTech

Analyse des Systèmes Distribués par Théorie des Jeux : Conception et Incitation

Game Theoretic Analysis of Distributed Systems : Design and Incentives

László Toka

Thèse présentée pour obtenir le grade de docteur de Télécom ParisTech
soutenue le 31 Janvier 2011

Directeurs de thèse :

Dr. Attila Vidács

High Speed Networks Lab
Département Télécommunications et Médiainformatique
Université des Sciences Techniques et Économiques de Budapest

Dr. Pietro Michiardi

Département Réseaux et Sécurité
Eurecom
Télécom ParisTech

2011

Résumé

Cette thèse présente les aspects d'incitation des systèmes distribués où une quantité limitée de ressources publiques ou privées doit être répartie parmi les participants égoïstes et autonomes. Notre objectif est de concevoir des mécanismes qui assurent l'efficacité et l'équité de l'allocation des ressources dans tels systèmes. Nous appliquons des modèles d'utilisateurs égoïstes et nous étudions les résultats de nos régimes proposés. Nous proposons également des algorithmes d'optimisation distribués destinés à la mise en œuvre dans la pratique.

Premièrement, nous ciblons les services de sauvegarde dans des systèmes pair-à-pair, c'est-à-dire des réseaux distribués constitués de pairs fonctionnellement égaux, où les utilisateurs sauvegardent leurs données sur les périphériques de stockage sous-utilisées des autres utilisateurs à travers d'Internet. Le système est capable de fonctionner à grande échelle puisque plus d'utilisateurs fournissent plus d'espace de stockage et de bande passante en globale. En outre, la diversité spatiale et propriétaire des hôtes de stockage assurent la disponibilité des données sauvegardées. Toutefois, la gestion des utilisateurs qui ne veulent pas partager leurs ressources locales avec les autres participants a d'importance extrême pour maintenir un système opérationnel. En outre, assurant une haute qualité de service dans un tel réseau pair-à-pair nécessite une conception du système avec soin. Nos nouvelles politiques concernant la redondance de données et la sélection des pairs rendent le service de sauvegarde fiable en échange d'une contribution équitable des ressources des utilisateurs.

Deuxièmement, nous examinons la gestion du spectre d'une façon dynamique qui permet d'allouer des bandes de fréquence pour les fournisseurs de service sans fil séquentiellement. Nous présentons notre conception d'un système distribué sur l'allocation et la tarification avec le but d'établir l'utilisation efficace du spectre, les allocations souples et la compatibilité avec des incitations, compte tenu de l'interférence physique entre les titulaires de fréquence. Notre travail donne un aperçu sur les nouveaux problèmes d'optimisation liés à la répartition du spectre. Nous proposons des solutions heuristiques à ces problèmes, basées sur nos résultats d'analyse. Nous évaluons le système et les algorithmes proposés avec des simulations numériques, et nous concluons que nos heuristiques peuvent être la fondation d'un système d'allocation dynamique distribué.

Abstract

This dissertation studies incentive aspects of distributed systems in which limited private or public resources must be allocated among selfish autonomic participants. Our goal is to design mechanisms which ensure the efficiency and fairness of resource allocation in such systems. We employ selfish user models and we investigate the results of our proposed schemes. We also design distributed optimization algorithms intended for practical implementation.

First, we target backup services in peer-to-peer systems, *i.e.*, distributed networks of functionally equal peers, where users save their backup data on the underutilized storage devices of one another over the Internet. As a main characteristic, no scalability problems arise since more users provide larger overall storage space and bandwidth. Furthermore, spatial and ownership diversity of storage hosts assure the availability of backed up data. However, the management of users not willing to share their local resources with other participants is extremely important to keep the system operational. Moreover, ensuring high quality of service in such a peer-to-peer network requires careful system design. Our novel data redundancy and peer selection policies provide reliable backup service in return for fair resource contribution of the users.

Second, we examine the potential of a dynamic spectrum management framework that enables sequential allocation of frequency bands for wireless service providers. We present our distributed system design on allocation and pricing with the goal of achieving efficient spectrum utilization, flexible allocations and incentive-compatibility, considering physical interference among frequency licensees. Our work provides insights on emerging optimization problems related to the allocation. We suggest heuristic solutions to these problems based on our analytic results. We evaluate the proposed framework and algorithms numerically, and we conclude that our proposed heuristics can be the cornerstones of a flexible distributed dynamic allocation system.

Kivonat

A jelen értekezés olyan elosztott rendszerek ösztönzõ kérdéseit vizsgálja, amelyekben korlátozott magán vagy közös erõforrásokat kell szétosztani önzõ autonóm résztvevõk között. Célunk olyan módszerek megtervezése, amelyek biztosítják a hatékony és méltányos erõforrás-elosztást ilyen rendszerekben. Az önzõ felhasználók leírására modelleket építünk és megvizsgáljuk a javasolt rendszerek teljesítményét. Továbbá olyan elosztott algoritmusokat tervezünk, amelyeket gyakorlati megvalósításban használni lehet.

Elõször biztonsági adatmentõ szolgáltatásokat vizsgálunk egyenrangú rendszerekben, azaz olyan elosztott hálózatokban, amelyekben a felek funkcionális szerepe megegyezik. Ilyen rendszerekben a felhasználók egymás kihasználatlan tároló eszközeire mentik az adataikat az Interneten keresztül. A szolgáltatás fõ jellemzõje az, hogy nem merülnek fel méretezhetõségi problémák, ugyanis több felhasználó nagyobb tárhelyet és sávszélességet nyújt összességében. Továbbá, a tároló eszközök területi és tulajdonosi sokrétûsége biztosítja az elmentett adatok rendelkezésre állását. Ugyanakkor azon felhasználók kezelése rendkívül fontos a rendszer mûködõképességének biztosításának szempontjából, amelyek nem hajlandóak megosztani helyi erõforrásaikat a többi résztvevõvel. Emellett a szolgáltatás színvonalának magasan tartása az egyenrangú hálózatokban gondos rendszertervezést igényel. Az újszerû adat-redundancia és tároló partner-választási szabályaink megbízható biztonsági adatmentõ szolgáltatást biztosítanak a felhasználók számára méltányos erõforrás-hozzájárulás fejében.

Ezután egy dinamikus spektrumgazdálkodás megvalósíthatóságát vizsgáljuk meg, amely lehetõvé teszi frekvenciasávok vezeték nélküli szolgáltatók közötti, egymást követõ kiosztását. Bemutatjuk az általunk javasolt foglalási és árazási keretrendszert, amely a rádióspektrum hatékony felhasználását, rugalmas kiosztását és ösztönzõk megvalósítását célozza, az ugyanazon frekvenciát használók közötti fizikai zavarást figyelembe véve. Munkánk eredményei betekintést nyújtanak az elosztással kapcsolatosan felmerülõ optimalizálási problémák nehézségébe. Ezek kiküszöbölésére heurisztikus megközelítéseket javasolunk, amely megoldások a problémák analitikus vizsgálatán alapulnak. Numerikus szimulációkkal értékeljük a javasolt algoritmusokat, és arra a következtetésre jutunk, hogy az adott heurisztikák egy rugalmas, elosztott dinamikus allokációs rendszer sarokköveit képezhetik.

Acknowledgments

I wish to express my gratitude to my advisors Matteo Dell'Amico, Pietro Michiardi and Attila Vidács for their guidance. I am also thankful for the support of my colleagues at Eurecom and at the High Speed Networks Lab.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Our research work aims at designing distributed systems with focus on fairness and incentives in resource provisioning. The dissertation consists game theoretic models of peer-to-peer (P2P, distributed network of functionally equal peers) backup and decentralized radio spectrum sharing solutions. The inherent selfish behavior of participants is mitigated by well-suited incentive schemes. Analytic and numeric results reflect the performance evaluation of the advised system design frameworks.

1.1 Research background

Nowadays, more and more information technology services and applications turn to the employment of the distributed paradigm because of scalability issues. Self-organizing and distributed systems appear in every domain of telecommunications; these systems, however different in most of the technical aspects, reflect similar incentive issues. Besides the extended research work on technical solutions, the related economic characteristics have also been tackled in growing measure recently.

1.1.1 Related work

Many distributed services currently rely upon altruistic behavior from their users. The phenomenon of selfish individuals who opt out of a voluntary contribution to the common welfare of the group has been widely studied, and is known as the free-rider problem. For example, uncontrolled or excessive free-riding in a P2P file sharing system leads to network congestion at some hot-spot peers and to the degradation of system performance; this phenomenon is indeed a real issue in P2P networks in general. It is thus important to design some mechanisms that encourage peers to contribute resources and reduce free-riding behavior in distributed systems.

A vast research literature tackles the aforementioned problems and propose solutions for distributed systems, such as network access sharing [96, 97], P2P file sharing [11, 29], network

routing [50], packet forwarding in ad-hoc networks [10, 23, 152], spectrum allocation [79], P2P storage and backup [11,33,95,130,139], network content caching [85,86], and network formation [30,49,84]. Designing incentive-compatible system frameworks is considered to be a hot topic in the research community.

1.1.2 Research goals

Our research aims at building specially tailored design for two different types of distributed systems: P2P backup and distributed radio frequency allocation systems. In both cases the non-cooperative selfish behavior of participants can jeopardize the operation. Based on user models, our goal is to design optimal economic incentive solutions that ensure desirable quality of service by fostering cooperation among users or by distributing shared resources efficiently. In the P2P backup system we introduce a barter-based scheme, for spectrum allocation we impose monetary flows between users and the authority. We evaluate the proposed incentive schemes from the user and from the system perspectives, through both analytic and numeric investigations.

In order to reach our goals,

- we model the rational behavior of selfish participants in the investigated systems, accounting for user benefits in terms of application performance, user cost of resource sharing (if applicable), and the heterogeneity of user characteristics, relevant for the service, *e.g.*, the heterogeneity of shared user resources, interference relations among users;
- based on the models, we build system designs with incentive solutions;
- in order to analyze the system models and the proposed incentive schemes, we utilize a broad class of analytic tools, *e.g.*, matching theory models for the P2P backup system, and auction theory for the allocation of radio spectrum;
- we decompose the appeared optimization problems and develop distributed algorithms to solve them;
- to approve the theoretic models, we perform simulations as proofs of concept, and we provide numeric performance evaluation of the proposed solutions.

We strive to create potential implementation solutions of practical, feasible and scalable applications. The frameworks to be designed with embedded incentive schemes and analyzed according to the assumed selfish user behavior, must ensure favorable outcomes in robust systems.

1.1.3 Motivation

The motivation that gives ground to this research work is the lack of reasonable incentives for users in the targeted distributed multi-user systems. An inappropriate economic design of

such a system sets back the possibility to reach a socially optimal outcome in the functioning. Application-specific schemes could enhance the way these distributed systems work.

The first part of the thesis aims a relevant field of research: there is growing need for seamless, secure, reliable and easily accessible online backup as the daily used electronic devices are more and more integrated into the Internet and the increasing transmission rates make possible moving large amount of data. Since instead of public or central resources users exploit those of one another, resource provisioning requires a well-suited incentive scheme.

The second part of the thesis targets radio spectrum allocation, studying auction-based management schemes in a distributed fashion. The main motivation behind the approach we investigate is that the sequence of central auctions to reallocate public resources should be transformed into a more scalable framework. We let the participants trade the acquired resources among themselves in a distributed design without the intervention of a central auctioneer.

1.2 Methodology

Game theory offers a tool-set for modeling individual user preferences, strategies, costs and valuations in distributed systems. We employ graph theory and matching theory to analyze the incentive mechanisms that we propose. Furthermore, we perform numerical evaluations with simulations, written in MATLAB.

1.2.1 Game theory

Distributed systems consist of autonomic participants and limited public or private resources to be distributed among them. It seems reasonable to assume that every user is selfish, *i.e.*, sensitive only to the quality of the experienced service, regardless of the effects of its actions on the other users. On the other hand, the quality of service a user receives depends on the generosity of other users: each user benefits either from the shared capacity of others or its share of the public resources. The framework of game theory [56, 69, 105, 122] is therefore particularly well-suited to study this kind of situation as a non-cooperative game played among users. Analytical investigations tackle user behavior, the existence of best-response strategies, Nash-equilibrium, Pareto-optimal outcomes, incentives, and social welfare.

Since in premature system designs there is no direct incentive to offer own capacity to the others or to fairly share the common wealth, users are motivated to free-ride [53]. If the sharing efforts do not get some kind of proof of appreciation, nobody has interest to cooperate and in extreme case the distributed service fails to exist. Therefore it is necessary to tackle the economic aspects of such systems and to properly design incentive mechanisms for the service in question [12, 31, 52].

The participants of a distributed system are assumed to be strategic. Exactly this is the

core of game theory that attempts to mathematically capture behavior in strategic situations, in which the success of an individual in making choices depends on the choices of others. The rational users change their behavior to maximize their own benefit when taking part in the system. In reality, there may be other types of individuals, *e.g.*, altruistic or malicious, but the predominant majority is assumed to behave strategically.

The combination of universal cooperation leading to optimal overall utility, an individual incentive to defect, and rational behavior provides the essential tension that results in the tragedy of the commons [82] without properly designed incentive schemes. Designing this latter requires complete knowledge about the participants, moreover to attain optimality, each peer should be offered a personalized scheme, and where necessary, enforce (or make payments to ensure) participation. None of these conditions are likely to be achievable in a practical distributed system, where not only the preferences of the individual peers but even their identities might be unknown [12].

As far as economic efficiency is concerned, besides the lack of information concerning the identities and preferences of the individual participants, which is required for the computation of the optimal allocation of resources and cost in a distributed system, an additional challenging issue is the complicated economic modeling of individuals. In several cases, the existence of externalities makes the adoption of a free-market approach using market-defined prices inefficient [12]. Also, different technical constraints limit the design variety of many distributed systems, *e.g.*, currency-based incentive schemes are hard to implement in current file sharing P2P networks. In addition to the challenging economic resource allocation problem, a system designer usually has to deal with the inability to rely on trusted software or on central entities that can monitor and account for peer transactions to ensure that they contribute and consume the amount of resources dictated by an underlying incentive model.

1.2.2 Incentive mechanism design

Incentive schemes must be deployed to align selfish participant behavior with the goals of the system design, *e.g.*, in P2P systems it is essential that peers be compliant with the protocol specification [64, 82, 102, 148]. However, modeling of the economic transactions carried out in a distributed system is, in general, a very complex task. The main reason is that participants should contribute different types of resources (money, power, bandwidth, storage, CPU cycles, content, etc.) with various characteristics, the provision of which generates complex costs taking into account the time spent using the system and in some cases extreme components such as legal risks [12,13]. Early works [54,64,139] made efforts to model the utilities and costs associated with the participation in a P2P file sharing system using game theoretic analysis: they analyze the free-riding problem and the equilibrium of user strategies under several micro-payment mechanisms, as incentive schemes.

Monetary-based incentive mechanisms are widely studied in every area of distributed systems. Currency has a well-defined uniform valuation for every participant and supports flexibility in terms of the time and the amount of counter-contribution for any given resource. In P2P systems the predominant payment scheme approaches belong to micro-payment solutions with clearance infrastructure or to credit-based systems. When publicly available limited resource must be allocated on a set of individuals, auctions seem to be the most suitable frameworks. Extensive research targets a whole range of such systems [79, 96].

When a given distributed system excludes the possibility of applying monetary tools in order to provide incentives for the participants, then barter-based solutions arise. Collaboration among peers is then motivated by resource exchanges, tit-for-tat strategies [11, 29, 101], reputation systems [65,82,104], penalty policies [102]. However, in these models, as a result of excluding monetary means, determining the objective value of resources that are subject to barter appears as an additional issue.

Distributed Algorithmic Mechanism Design (DAMD) aims to create incentives for Internet applications. In an ideal system design selfish nodes would maximize the common welfare [12]. If no centralized authority with total knowledge can make decisions about the system resources, building an incentive scheme becomes a DAMD problem, combining computer science with incentive compatible mechanism design of economics. DAMD provides a useful framework to enforce the proper provision of resources in a distributed system [51].

1.2.3 Matching Theory

Matching theory [58,72,73], a field of combinatorial optimization, provides useful tools to analyze, among several other possible targets, *e.g.*, peer selection in P2P file sharing [57, 88] systems.

1.3 Outline of the dissertation

The first part of the dissertation focuses on P2P systems for backup. Users participate in the system operation by offering private resources (such as storage, online time, bandwidth) for the benefit of the community with the ultimate goal of improving application performance.

We study important design options of data redundancy, data maintenance and data transfer over the network in Chapter 4. We suggest a new procedure for determining data redundancy, and we evaluate its performance compared to currently known techniques in Chapter 7. For their comparison, we define novel metrics that describe the quality of service, *e.g.*, the duration of archiving data and of recovery processes, the probability of backup loss. Determining redundancy is based on the time required to retrieve the backed up data and it guarantees high service quality levels while significantly reducing the applied redundancy, thus storage and bandwidth requirements.

We show the settings that provide high quality of service for backup purposes and, in the meantime, require the least possible shared resources. However, a system with low number of users might not be able to guarantee the appropriate quality of service based only on the shared user resources. Therefore, we examine the effects of introducing a central storage server in order to avoid such situations: we show the cost implications of persistent quality guarantees. In such a hybrid system the central, highly available server might be used to store data in exchange for the reimbursement of costs. Furthermore, it can also be used to restore data from the remaining copies when backup is lost on peers that leave the system. In Chapter 7 we arrive at the conclusion that relatively low occurring costs of such a hybrid system significantly improves the quality of service.

Since the extent of user contributions affects the system, an incentives scheme is required in order to maintain the service: participating users must share disk space, bandwidth and online time, connected to the Internet. In Chapter 5 we present a symmetric peer selection scheme in which users have the ability to selfishly select remote peers they want to exchange data with. Peer characteristics (*e.g.*, online availability, dedicated bandwidth) play an important role and are reflected in the model through a single parameter, termed grade. We show that selecting remote peers selfishly, based on their profiles, creates incentives for users to improve their contribution to the system.

In Chapter 6, we show an efficient algorithm to compute the optimal data transfer scheduling solution for hypothetical cases where future peer uptimes are known. We use the optimal results to evaluate the applied scheduling policy, and we propose practical settings in which the performance of random decisions is close to optimal.

The second part of the dissertation investigates the possibility of allocating radio spectrum among multiple applicants dynamically in a distributed manner. The distributed dynamic spectrum allocation framework, if frequency leasers are coordinated by a well-suited scheme, provides efficient methods for the allocation of the scarce underlying resources, with respect to the general interference conditions.

In Chapter 11, we present a model and the related framework of allocation and pricing that offers a distributed mechanism design, adapted to practical employment issues. Our model handles interference effects without any restricting assumptions, and our framework is scalable and incentive-compatible. We provide both analytical and numerical evaluation (in Chapter 12) of the proposed framework, and in either case we prove this latter to be a suitable approach to efficient and flexible spectrum utilization.

Part I

Peer-to-Peer Backup System

Chapter 2

Introduction

Nowadays, accumulated information results in large, and continuously increasing amount of digital data, a significant part of which is personal and cannot be reproduced easily, therefore their safe storage is essential. Numerous methods and applications are available to simultaneously produce and backup copies, but the difficulty and high costs of the archiving process preclude most users to provide safety to their data.

We study online backup services that offer safe storage of personal data for users with an Internet connection. We suggest an innovative and viable approach to realize this service via a P2P architecture which exploits user resources (storage, online time, bandwidth) with the goal of decreasing the usual cost of backing up data online. Simulations show that such a system has the performance and the reliability similar to those of a costly central storage server.

The reliability of a backup service can be established to different extents. If the system provides data *availability*, then it means that the data is accessible online whenever a request is issued. This implies that any arbitrary part of the data can be fetched promptly by the data owner, therefore guaranteeing low access latency. On the other hand, if the *durability* of data is ensured, the data is safely stored in the system as long as the service is required, but it is not necessarily always accessible, at least not all parts of it. A reliable system recovers the stored data despite failures, without violating the level of data accessibility it guarantees.

2.1 Online data storage services

The advent of cloud computing as a new paradigm to enable service providers with the ability to deploy cost-effective solutions has favored the development of a range of new services, including online storage applications. Due to the economy of scale of cloud-based storage services, the costs incurred by end-users to hand over their data to a remote storage location in the Internet have decreased significantly. Furthermore, the process of backing up user data online are usually performed automatically by a client application: in contrast to on-site backup solutions, user

interaction is minimal, and in case of data loss due to an accident, restoring the original data is a seamless operation.

Commercial online data storage services [2,3,6,7,147] can be broken down into those based solely on capital-intensive and energy-consuming [5,84] server farms [2,3] and those embracing the P2P paradigm [6,7,147].

Amazon S3 [2] is based on large parks of commodity hardware (*i.e.*, a data center) running a custom-built distributed data structure discussed in [37]. Data availability and durability do not come for free: end users are compelled to pay for the amount of space they occupy in the data center and the amount of traffic their contents generate. Furthermore, cloud-based approaches can be subject to failures, as reported in [1]. Many companies base their online storage services (*e.g.*, Dropbox [3]) on the Amazon cloud by building a user-friendly interface to it.

The centralized component that ensures storage for users of Wuala [7] is complemented by storage capacity at all available peers taking part to the application. Generally, data is placed both on servers and on a P2P network with a predetermined amount of redundancy. Servers coordinate the data placement by selecting storage nodes: users must offer an amount of local space inversely proportional to their average online availability, and a minimal dedicated bandwidth capacity [65]. The same servers are involved in constantly checking that these constraints are satisfied.

The long-term storage cost of online services, which are particularly dominant in the context of backup applications, may easily go past that of the traditional offline solutions. Additionally, while data availability is a key feature that large-scale data-centers deployments guarantee, its durability is questionable [147], as reported in [140]. A P2P storage system is a viable alternative to cloud-based solutions: user commodity storage devices are shared (together with some bandwidth resources) with a number of remote users to form a distributed storage system that is resilient to local failures.

2.2 Backup versus storage in a P2P system

General purpose online storage systems optimize latency to individual file access, since users upload their data to the system as a replacement of a local hard drive. Maintaining high data *availability* in a P2P storage system puts high burden on the users due to the intermittent online appearances of storing peers. The online behavior of users is unpredictable and, at large scale, crashes and failures are the norm rather than the exception. As a consequence, a P2P storage application stores large amounts of redundant data to cope with such unfavorable events, *i.e.*, storage space is sacrificed for low access latency.

Despite the enormous amount of work carried out in the research of P2P storage during the last decade, no popular solution has emerged targeting common Internet users. The reason for

this is that a favorable trade-off between the performance and the resource requirement of such a service is hard to find. As Blake *et al.* argue in [20], it is prohibitive to have enough redundancy to keep all data available at all times, although without this, the performance is questionable. The high demand of peer resources might be unsatisfiable by the contribution of the participant peers, even if well-suited incentives are in place to encourage them to share.

On the other hand, an online backup service, as a particular case of online storage, involves the bulk transfer of potentially large quantities of data, both during regular data backups, and especially in case of retrievals. As a consequence, short backup and retrieve times are more important goals to achieve than low access latency. Furthermore, as the backup service assumes a local copy of the data at the user, retrieving the backup is required only when the locally stored copy is lost. At these rather rare events, access latency to specific parts of the backup is simply not an issue since the bulk traffic needed to restore the whole of it would need hours anyway.

Rather than aiming for another general purpose P2P storage system, we therefore focus specifically on a simpler to implement and yet very useful application: backing up the important data of users. Given the above considerations, our backup system design optimizes backup and retrieval times, while guaranteeing data *durability*, *i.e.*, ensuring that the loss of backup remains an unlikely event.

We show in the thesis that providing data durability is less difficult to achieve, in terms of required peer resources, than data availability. We present the drastic reduction of data redundancy, still manageable to ensure the above quality of service guarantees. Furthermore, we demonstrate the possibility of avoiding separate design layers that provide user incentives: our application fosters cooperation among peers *by design*. Moreover, we propose a hybrid design with a data center that improves the system performance to the same level of a costly centralized service, even if our optimizations and incentives fail to ensure the necessary peer resources. Striving to relieve users from payments, our data transfer scheduling policies keep the bandwidth and storage costs of the data center at a minimum.

2.3 Focus of the work

The thesis presents our work on online data backup systems in which users store copies of their files for long-term. We propose a system design that involves user edge devices that confederate by pooling their local resources in a P2P network, and a central storage facility, *i.e.*, a data center. We define a set of performance metrics that describe important quality of service aspects (*e.g.*, data durability, backup and retrieve time). We analyze different design choices of data redundancy, data placement and data transfer scheduling by evaluating the system performance.

We review related works in Chapter 3, organized according to the structure of the thesis. We also give a broad overview on different aspects of P2P storage systems which are loosely related

and/or orthogonal to our contributions, but important for a complete system design.

In Chapter 4, we overview our system design and discuss its key components. We describe in detail our application scenario, and we show why the assumptions underlying a backup application can simplify many problems addressed in the literature.

In order to maintain the durability of backed up data, the degree of *redundancy*, *i.e.*, the amount of additional data in the P2P system that guarantees a backup operation to be considered complete and safe, must be chosen wisely. We present a novel redundancy policy in Section 4.2 that, rather than focusing on short-term data availability, targets short data retrieve times. As such, our method alleviates the storage burden of large amounts of redundant data on client machines.

In order to enhance the performance of the system, we propose to employ a data center to complement the resources offered by peers, if they are not sufficient. For example, detecting a faulty external hard-drive may not be immediate, or obtaining a new piece of equipment upon a crash may require some time; in these cases an *assisted* approach to repair data redundancy on peers, which involves a cloud-based storage service, can significantly reduce the probability of data loss at an affordable cost. Similarly, we show that an assisted backup process mitigates the negative effects of low peer quality, decreasing the data loss probability along with the time to backup. The suggested assisted policies offload the data center by transferring backup data to peers as soon as possible, thereby minimizing data center costs (*i.e.*, storage and bandwidth burden) while sustaining the quality of the backup service.

In Chapter 5, we focus on the *peer selection* process, during which peers choose where to place fragments of backup data they need to store. A global ranking is built among the peers in terms of the *quality* of storage space, representing the online availability and bandwidth, they offer to the P2P system. We model the process as a game in which users selfishly optimize the utility and cost they bear for joining the system by adjusting their shared storage and qualities. We present the objective function that drives the behavior of peers in the game, and we study the algorithmic perspective of the uncoordinated peer selection process.

We show that the game reaches an equilibrium in which the system is *stratified*: peers with similar characteristics cooperate by building bi-lateral links that are used to exchange and store data. As peers improve their contributions, the service they receive from one another becomes less costly, thus, the consequence of system stratification is a natural incentive for peers to improve the quality of resources they offer to other peers.

We further discuss the *scheduling* policies, *i.e.*, deciding how to allocate data transfers between peers. In Chapter 6 we formalize the problem of exchanging multiple pieces of data with intermittently available peers during uploading and retrieving operations with full knowledge of future peer uptime, and we show that it can be solved in polynomial time by reducing it to a maximal flow problem. The full knowledge setting is obviously unrealistic: therefore we

prove that a randomized approach to scheduling completes transfers nearly optimally in terms of duration as long as the system is sufficiently large.

In Chapter 7 we corroborate our theoretical findings with simulations, driven by real availability traces from an instant messaging application where on-line appearances of users show daily and weekly patterns. Extensive numerical simulations, with a distributed algorithm for the peer selection process, show that our techniques are effective in such a realistic scenario with heterogeneous peer availabilities and bandwidths. With the simulation of a complete P2P backup system we show that our proposed design is viable in practical scenarios and we illustrate its benefits in terms of increased performance compared to other system designs.

Our contribution, discussed in detail in Chapter 8, is three-fold:

- we present a P2P backup system design in which data redundancy and its maintenance are adapted to the requirements of a backup service driven by our novel performance metrics, resulting in a significant decrease of resource usage compared to general purpose storage;
- through user-driven peer selection, we introduce embedded incentives for peers to share their local resources and we show the stable configuration of such a system;
- we show the time and bandwidth constraints of data transfers between peers in a P2P system, and we suggest to diminish those inefficiencies for a low cost by integrating a data center storage service in the system.

Chapter 3

Related work

Online data storage solutions, allowing users to store and access their data from any point on the Internet, are commercially popular products. These applications have received much attention from the research community that studied their many facets. In particular, research on distributed P2P storage applications has proliferated in the literature. The complete design of such systems requires considering several problems. Here, we focus on those works that are closely related to our contributions, but also give an overview on topics that are not addressed in the thesis.

The challenge in the design of P2P data storage systems is that a reliable service must be built on *many* unreliable peers. The emphasis is on the high number of participants, because the underlying concept exploits the diversity of peer characteristics. Indeed, the data stored in the system remains *available* online if it is scattered on so many peers that even if most of them are temporarily disconnected, there are enough peers connected to the Internet to serve any data read request. Similarly, the *durability* of storage is ensured if permanent loss, deletion or corruption of data on peers can be corrected by maintaining the spread data with different *repair* techniques. The availability and durability of stored data are achievable only if events that make data parts unavailable or lost are not positively correlated. In this case a system containing numerous peers to store data parts can provide both features by the law of large numbers.

In the following sections we revisit the collection [44] of major ideas that tackle the amount of data redundancy and its maintenance, the placement of data parts on peers, and the related data indexing, security and incentive issues.

3.1 Data redundancy in P2P storage

During the last decade, research focused on the design of general-purpose P2P storage that provides features of traditional file systems. Therefore, a significant amount of work has been devoted to implementing systems with low latency, the most difficult task when building dis-

tributed storage. Numerous solutions emerged, all proposing to add some redundancy to the stored data, but with different methods and extents depending on the design perspectives. We give a brief overview of existing redundancy schemes, we summarize the techniques that are applied to maintain the redundancy, then we argue on the differences between storage and backup systems.

3.1.1 Replication

The simplest redundancy scheme stores copies of data on different peers. If r replicas of the same file are placed on r different peers, the file is available even if $r-1$ of them are offline. Its simplicity made this scheme wide-spread in the first P2P storage system designs, *e.g.*, PAST [43] adopts the replication of files, CFS [35] divides files in blocks and performs replication at the block level.

As described in the following, the redundancy scheme is intertwined with, among other aspects, its repair technique. When a copy is lost, the applicable repair consists of simply copying an available replica to another peer, based on the data placement policy.

The weakness of replication is that the amount of redundant data is significantly larger than in more sophisticated schemes, therefore imposes high storage burden of peers to meet the same data availability and durability guarantees.

3.1.2 Erasure coding

The more sophisticated redundancy schemes apply erasure coding on the data to store. The basic description of the scheme is as follows: data is organized into *backup objects*, each of them is cut into k, equal size original *fragments* which are then transformed into an arbitrary number, but $n > k$, of encoded fragments, again of the same size, to be stored on n remote peers. Any k of these encoded fragments are sufficient to reconstruct the k original fragments, thus the backup object. Therefore the redundancy scheme resists to the erasure of $n-k$ encoded fragments, stored on peers.

The benefit of erasure coding is that less redundancy (defined as $r = \frac{n}{k}$) is required, thus less storage space is consumed than with replication, when providing the same level of reliability [113, 141]. As an illustrative example, let us consider replication with $r = 3$ copies, and erasure coding scheme with $k = 3,\ n = 5$. Both schemes ensure data durability when only two peers lose their assigned data, however the data redundancy rates r are different: $r = 3$ with replication and less than 2 with erasure coding. Note that both schemes imply that replicas or fragments are placed on distinct peers, in order to maximize the diversity of storage locations.

The inconvenience of erasure coding schemes is that repairs are more complex than transferring copies of data. When an encoded fragment is lost, in order to create it again (or a new

one), the encoding process has to be performed again, for which the original data or k redundant fragments are needed. If none of these choices are ensured at one location, the repair of a sole encoded fragment imposes the transfer of a data amount equal to the size of the backup object on the system.

The most widely used erasure codes are based on linear operations on Galois Fields, these solutions are called as Reed-Solomon codes [112]. The main drawback of these latter is the computational complexity regarding the encoding and decoding processes; the remedy is proposed by LT codes [93] that provide linear coding and decoding times, although the reconstruction of a backup object requires slightly more than k encoded fragments. Furthermore, these codes do not predetermine the parameter n, hence the name: rate-less (or fountain) codes. As an alternative, rate-less erasure codes can be constructed based on random linear codes from the area of network coding [55].

The system designs that apply erasure coding to build data redundancy are numerous [8, 17, 18, 40, 67, 81, 91]. The popularity of the scheme is due to its high storage efficiency since, as mentioned above, erasure codes are able to provide the same level of reliability as replication, consuming much smaller storage space. If the major drawback of erasure coding is that accesses and repairs have higher latency than in the replication scheme. If this becomes more important than the savings in storage space, designers apply replication, *e.g.*, for active and for meta-data [18, 81].

3.1.3 Maintaining redundancy

As mentioned earlier, redundant data does not guarantee the availability and durability of data, unless it is maintained during its lifetime: fragments that are lost on peers that crashed or abandoned the system must be repaired by building and storing new fragments.

The repair policy determines when a fragment repair must be performed. Most of the existing techniques have the target of ensuring *prompt availability* at all times. In these P2P storage systems, either *eager* [35, 43] or *lazy* repairs are adopted [18]: offline fragments are repaired either immediately, or only after some delay, respectively. The first policy is simple, but inefficient, since all peer disconnections are considered as losses; the second one is the opposite: it efficiently takes into account both, transient and permanent disconnections, reintegrating fragments on peers that come back online after a short period to avoid unnecessary repairs, but it also requires more sophisticated decision-making.

The decisions that drive lazy repair techniques must ensure the amount of redundancy that guarantees the required availability, and most importantly durability of data. Generally, thresholds on the necessary number of fragments are determined after observing statistics of online, and supposedly alive, but offline peers. System designs apply reactive [18, 27] and/or proactive repairs [36, 47, 121] to always meet these thresholds. Basically, data is stored in the system with

higher redundancy than the repair threshold, and proactive repairs are performed to ensure that the rate always remains above, or reactive repairs are performed when the rate falls below it. The motivation for the reactive policy is to avoid waste of bandwidth, *i.e.*, to perform repairs when they seem to be necessary, while the reasoning behind the proactive scheme is to smooth out bandwidth utilization during the repair process, since reactive repairs might come in bursts.

3.1.4 Feasibility of P2P storage

Among many other approaches, the storage system design of TotalRecall [18] adopts a binomial formula to calculate erasure coding redundancy rate that guarantees low latency through prompt data availability. This rate, as ensuring prompt data availability in general, requires high storage and bandwidth contributions from peers in typical settings.

As an example, in Wuala [7], the usage of a data center is inevitable because the overall storage capacity at peers is not sufficient to sustain the expected data availability for every stored file. As Wuala offers low latency storage services, the system must store data on servers, and the load on peers serves only caching purposes: popular file parts, stored on peers, can be retrieved from multiple sources for free, hence the advantage of the P2P overlay in terms of traffic [94].

Proportionally with the amount of redundant data to store, the repair policy in P2P storage systems necessitates large communication bandwidth from peers. The authors in [20, 21, 113] argue that such a system might be unsustainable in common Internet scenarios with high peer failure rate and low peer bandwidth capacities. The reason for this is, again, that the repair of each lost encoded fragment requires the transfer of a larger amount of data.

To remedy this disadvantage of erasure coding, hybrid schemes might use replicas to perform repairs of encoded fragments avoiding the transfer of the whole backup object. Dimakis *et al.* [39] discusses on the benefits of using network coding in alleviating the costs of data maintenance as opposed to approaches based on erasure coding. On the other hand, they argue that the complexity of the system grows, since different maintenance polices must be applied for replicas and for erasure coded fragments. In fact, redundancy schemes, applying solely erasure coding, might become feasible with more sophisticated codes, built with the need for repairs in mind. When maintenance of redundancy is delegated to peers that do not have a local copy of the backup objects, hierarchical [45] or regenerating [39, 46, 144] coding schemes can be used to lower the amount of required data transfers.

Although storage systems are more often envisaged in the literature than backup systems, data availability requirements, that receive particular importance, are unfortunately cannot be always fulfilled. On the other hand, P2P backup systems, in which data durability is far more important than availability, can be operated for much lower price in terms of redundancy and repairs.

The authors of [89] advocate the use of a timer-based repair policy that separates durability from availability. Moreover, various works [27, 107] determine redundancy as a function of node failure rate in order to guarantee only data durability at the expense of low data availability. Since these works aim to decrease the amount of redundant data in order to alleviate the storage and traffic burden on peers, their focus shifts to the maintenance of the lower, therefore less safe redundancy rate, *i.e.*, to data repair techniques.

Many works devoted to P2P storage target the almost Herculean task of backing up the entire contents of a hard drive, including operating system files. These works propose *convergent encryption*, a technique to achieve data summarization that avoids storing multiple times pieces of data that are common to many users [16, 32, 83], thus ensuring that storage space does not get wasted by saving multiple copies of the same file across the system. In contrast, personal backup usually involves only an important subset of user data to save, thus the amount of overlapping data that could be summarized is plausibly very little. Instead, user backups should encode incremental differences between archive versions. Recently, various techniques have been proposed to optimize computational time and size of these differences [126].

3.2 Data placement

The P2P storage system is built up by the shared resources of participant peers. Their storage space contribution can be organized into an architecture in many different ways, and the structure (or the lack of it) plays a very important role in the system operation, from repair policies to peer incentives. In this section we give an overview on these aspects.

3.2.1 Central or distributed mapping

The first and simplest solution to organize data stored in the system is a centrally coordinated approach. The notion of *tracker*, a server that registers and monitors peers, and furthermore dictates and organizes data placements, was first adapted in P2P file sharing systems [4, 29], where the tracker has a database about the exchanged files, their locations, and the status of peers.

The simplicity of tracker-based design is overshadowed by issues of scalability, robustness and security: even if the server does not store data replicas or fragments [91], as a single point of failure, any overload due to intensive queries or malicious attacks can render it dysfunctional. As such, no data is lost in case the tracker fails on the short term, however since it coordinates data placement and repairs, a long term failure significantly affects the performance of the system.

In F2F [90] and FriendStore [137], peers store their data at their friends to improve data durability. The authors argue that choosing storage partners based on existing social relationships provides incentives for nodes to cooperate and results in a more stable system which, in turn,

reduces the cost of maintaining redundancy. The cost of this approach is decreased flexibility, and backup capacity.

One step toward a distributed placement algorithm is presented in [8]: peers are randomly assigned to groups that manage different partitions of the overall storage. Every peer that belongs to the same group has a consistent view of the attributed partition. This architecture requires trusted peers, therefore it targets corporate networks instead of a common P2P scenario.

The appearance of Distributed Hash Tables (DHTs) [111, 116, 123, 150] made a completely distributed system design for P2P storage possible. DHTs provide consistent mapping between keys and values [77] in a completely decentralized fashion: *e.g.*, in Chord [123] each peer and stored object is identified by a key, and is logically positioned on a ring composed by the possible key values, then each peer is responsible to store the objects, whose keys fall between its key and the preceding key of another peer.

Various DHTs focus on different performance guarantees, and propose design choices accordingly, although the basic mapping functionalities are the same: every peer creates and maintains (when peers connect and disconnect) some links to remote peers in the DHT in order to constitute a structured overlay for fast lookups through simple routing protocols. Furthermore, the peer, that is responsible for a storage object, replicates it on some remote peers with hash keys close to its own, which intuitively makes replication be applied in DHT-based P2P storage [35, 43].

In Pastiche [32], storage peers are selected based on the similarity of their backup: peers save storage space by de-duplicating common backup data.

3.2.2 Peer monitoring

The repair policy, in almost every system design, requires knowledge about the status of peers. In order to determine whether a peer is online, temporarily or permanently offline, an active or passive monitoring system must be in place. This connectivity information is used in order to highlight data parts that are in danger and need repairs.

If a tracker organizes the data placement [91], it is also able to perform monitoring operations easily. In an unstructured distributed P2P system, decentralized monitoring might be probabilistic and the results can be aggregated via a gossip-based protocol [60], which floods the system with peer monitoring information.

In structured overlay networks, distributed monitoring relations among peers may follow the overlay structure. DHTs offer an intuitive policy: each peer monitors a set of its neighbors which also store replicas of storage data assigned to the peer, therefore issues related to data placement, peer monitoring and repair policy, that reacts to the results of monitoring, are solved at once [35, 43, 67, 149].

Correlated online behavior of peers, *e.g.*, daily and weekly online patterns of peers in the same timezone [17], not only affects the performance of redundancy schemes that make a basic

assumption about independent and uncorrelated uptimes, but also renders the measurement of peer availabilities in a distributed manner more difficult.

3.2.3 Fairness

One of the most investigated peculiarities of P2P systems is the need for peer *incentives*. If well-suited mechanisms do not enforce peers to contribute at least as many resources as they consume, then *selfish* peers exploit the shared resources of more altruistic participants without sharing their own. This phenomenon is referred to as *free-riding* and the change of identities in order to escape from badly designed incentives is called *white-washing* [52].

It is obvious that free-riding leads to the collapse of the P2P service on the long term, affecting P2P storage even more drastically than other types of services. If peers insert more data into the system than the storage space they share, the service becomes unsustainable. Enforcing storage *fairness* is therefore imperative: either symmetric storage exchanges must be made between peers in a "tit-for-tat" fashion [32, 91], or multilateral symmetry must be established through a transferable "currency" [33, 64, 71]. While bilateral barters are simple to enforce and control, a currency-based design imposes less constraints on data placement, however, requires a central authority.

A structured approach to data placement, *e.g.*, based on a DHT, renders peer selection *implicit* because data is uniformly stored on peers following the consistent hashing concept. Therefore, DHT-based approaches achieve load balancing by spreading data on every peer with uniform probability, irrespectively of their characteristics. As a consequence, peer heterogeneity in terms of the amount of resources they dedicate to the system has to be taken into account by an additional system layer to elicit their cooperation. This is further complicated with the *asymmetry* in that a peer might store data chunks for a remote peer that is not necessarily storing its data.

3.2.4 Incentives

In effect, peer characteristics directly determine the quality of service in the system. For example, low peer availability would require extensive use of redundancy in order to make data accessible despite massive peer disconnections. In general, large differences can be seen among peers in terms of connectivity, *i.e.*, up-time and communication bandwidth, which may motivate non-random data placement policies that take into account this heterogeneity. Many works [42, 108, 127] improve data availability by carefully selecting peers that will store replicas or fragments of data. Somewhat as a counter-example [62] addresses peer selection strategies under churn, and the authors present a stochastic model of a P2P system and argue on the positive effects of randomization.

If data placement strategies put more storage on peers with better characteristics, selfish peers will not strive to show long uptimes and high connectivity, because in return they would receive more and more data to store, saturating their connection links and storage capacity. Therefore, besides enforcing fairness in terms of used and shared storage capacities, *i.e.*, any peer should offer an amount of local storage space equal to the load it imposes on the system, the fairness should be extended to other resources as well.

A recurrent problem for P2P storage applications is creating incentives to encourage nodes to contribute *more* online time and bandwidth resources [33, 83, 91]. This can be achieved via reputation systems [76] or virtual currency [139]. In general, the idea is to impose fairness spanning every resource type. A commercial example is Wuala, where the amount of local space to be shared for a given amount of online storage is inversely proportional to the peer availability [7]. However, in this scheme highly available peers may have to store data on unavailable peers offering most of the cumulative available storage space.

Toka *et al.* [95] compares two possible mechanisms to manage a P2P storage system: they suggest that either the service use of each peer should be limited to its contribution level (symmetric schemes), or that storage space should be bought from and sold to peers by a system operator that seeks to maximize profit. Using a non-cooperative game model to take into account user selfishness, the authors study these mechanisms with respect to the social welfare performance measure, and give necessary and sufficient conditions for one scheme to socially outperform the other.

While peer selection in most related works is dictated by well-known data indexing-retrieval techniques (*e.g.*, DHT, locality-based), our symmetric data placement scheme (Chapter 5) is based on shared resources. Every user picks remote peers selectively, *i.e.*, peers with high quality are preferred over peers with low quality. The resulting overlay creation process resembles to network creation games [49], especially to the pairwise version of them [30].

Similarly to [101], our key observation is that a network creation game with bilateral agreements can be studied with the tools of *matching theory* [115]. We introduce a generalized stable matching problem that we use as a theoretical basis for peer selection. The study of peer selection in P2P content sharing applications [57] and in P2P storage [118] have pinpointed that user-driven peer selection brings to *system stratification*. We show that this result provides fairness and embedded incentives for users to offer the necessary amount of resources.

An additional benefit of user-driven peer selection is that efficient techniques to perform fast look-up of individual backup files are unnecessary: a restore operation can be completed with the simple knowledge of remote locations with which data exchanges have been performed.

3.3 Reliability of the system

In contrast to storage services, our goal is to ensure data durability with a P2P *backup* system design that neither requires user payments nor external incentive mechanisms to provide safe backup in exchange for as little shared resources as possible. Besides the issues and solutions discussed so far, the reliability of the service highly depends on the speed of data transfers and on the security aspects.

3.3.1 Data transfers

Besides the amount and placement of redundant data, organizing data transfers has received less attention from the research community. The authors of [19] analyzed random backup scheduling by modeling peer uptime as a Markovian process. Their study reached a conclusion that is analogous to what we obtain in Chapter 6: in backups, the completion time of random scheduling converges to the theoretical minimum as the system size grows.

BitTorrent [29] uses fixed-size fragments, and adaptive upper limits on parallel transfers in order to avoid unfinished transfers and the appearance of bottlenecks respectively. Inspired by these design elements of BitTorrent, we applied similar techniques in our backup P2P system.

As mentioned in Chapter 2, we suggest to employ a data center as a remedy for the temporary lack of peer resources. Little work has been done on hybrid approaches to mitigate the shortcomings of P2P systems. To the best of our knowledge no prior work tackles data backup and/or repair operations assisted by a central entity.

AmazingStore [147] improves data availability of cloud-based storage services and reduces their costs by augmenting centralized clouds with an efficient client-side storage system. Peers backup at other peers, besides the servers, with different online patterns to improve the data availability and to serve read requests within the P2P network. Therefore the hybrid system mitigates the issue of the centralized point of failure, and provides resilience to large-scale failures.

FS2You [92] is a peer-assisted system that provides temporary storage and seeding for files in a BitTorrent-like content distribution system with a hybrid structure consisting peers and servers. FS2You does not guarantee data persistence; while its goal is to minimize bandwidth costs, we focus instead on minimizing the storage costs that will be dominant in the long run for a storage system.

3.3.2 Security

While correlation among peer uptimes is a natural phenomenon, losing data on peers *in masses* is somewhat suspicious. The most probable reason for many peers to lose the data stored on them simultaneously is that they all run the same software, *e.g.*, operating system, or they had been created as part of a Sybil attack [41]. Since the reason for these correlated peer failures can

be a vulnerability, creating high risk on the reliability of the system, we see them as a security issue.

Many works [67, 74, 75, 100, 142] tackled this issue, and proposed to place data on peers that are less possibly correlated: those who use different software configurations, who are connected through different network service providers and are far from each other geographically.

Along with losses due to peer failures, replicas or fragments can be destroyed on peers accidentally and voluntarily. In order to detect these events, data integrity checks must be performed periodically. These verifications also ensure that peers are really storing the data assigned to them, hence enforcing fairness in storage.

While common checksums signal accidental corruptions, detecting malicious data deletion requires more sophisticated operations. Two main approaches have been proposed: either peers create self-verifying data blocks with signatures that are cryptographic collision-resistant hash function of the blocks themselves [119,143], or they perform probabilistic challenges via cryptographic protocols toward storage peers that can be answered only by holding the data block [14,91,106]. With the first option, to detect any data modification the peer has to download the block and recompute the signature to perform the comparison.

Since in our backup system we can assume to have a single data owner authorized to read and write it, data confidentiality and access control can be achieved with standard cryptographic techniques between the storing couples. Moreover, for the same reason, consistency guarantees for multiple readers and writers, and anonymity for data publishers and readers are not needed.

Chapter 4

System design

The goal of a P2P backup system is to store data for users safely on remote peers. Every system participant runs a client application on its device, *e.g.*, computer or set-top box, with shared storage capacity, connected to the Internet. The system design must cope with the unavailability of peers and the unpredictable amount of resources dedicated to the system.

We have to consider many design aspects when building our system. This thesis presents innovative elements regarding the data redundancy scheme, the peer selection with related data placement strategies, and the data transfer scheduling policy. In those fields where known techniques provide reasonable solutions, given the purposes and the assumptions of a P2P backup system, our work refers to and reuses mechanisms published in the literature, *e.g.*, erasure coding and repairs.

4.1 Data backup and retrieval

The system design that we present contains a central server, called *tracker*, which is operated by the backup service provider to supply various system management operations, such as peer registration and monitoring. We note that they can also can be implemented in a distributed way using well-known techniques (*e.g.*, DHTs). Such a task is however outside the scope of the thesis. Therefore for simplicity we assume that all of them are carried out by the tracker.

When a peer has to save new data, the *backup phase* begins. During it, the data owner:

- establishes its data to be backed up (stored locally indefinitely unless the device of the peer crashes), divides it into fixed size fragments, then creates additional fragments by encoding the original fragments by erasure coding (Section 4.2);
- queries the tracker for a set of remote peers that are willing to store a fragment of the peer on their devices, *i.e.*, have sufficient unallocated storage capacity, and announces itself at the tracker as a storage candidate in the meanwhile (Section 4.3);

- performs fragment transfers to an online subset of selected peers, striving to complete each of them within a predetermined transfer timeout (Section 4.4).

The phase lasts until backup is completed: the peer reaches a sufficient data redundancy, thus safe backup, by having uploaded enough fragments successfully. Afterwards, the *rearranging phase* is started, during which the peer re-uploads the fragments lost due to peer crashes.

As soon as a peer notices it has lost its local storage due to *e.g.*, device crash, the peer initiates a *retrieval phase*, during which it remains online until its retrieval operation is finished. After it has downloaded enough of its own fragments from remote peers and restored its data, it also downloads fragments of remote peers to store.

The performance metrics we will use to evaluate our novel redundancy, peer selection and transfer scheduling schemes are data loss probability, time to backup and time to retrieve the data.

Definition 1 ***Data loss probability*** *The backup gets lost if not enough fragments can be retrieved from remote peers. The amount of time spent before noticing peer crashes and the speed of retrievals have important impact on the probability of losing data, defining an interval during which data is at risk, because no maintenance of redundancy is carried out.*

Data loss probability (DLP) describes the likelihood of losing so many encoded fragments stored on remote peers that the remaining fragments are insufficient to restore the original data. The DLP depends both on the crash rate of storage peers and the time duration for which the probability is established.

Definition 2 ***Time-To-Backup and Time-To-Retrieve*** *The Time-To-Backup (TTB) (resp. Time-To-Retrieve (TTR)) of a user is the time elapsed while the user uploads (resp. downloads) a number of encoded fragments to (resp. from) remote storage locations which meets the target redundancy (resp. is sufficient to restore the original data). TTR is definable only if the user has backed up enough fragments to make restore possible before starting to retrieve a subset of them.*

We calculate the lower bound for both TTB and TTR. Let us assume a storage service with unlimited bandwidth capacity and uninterrupted online time. If a user backs up its data with this service, its TTB and TTR only depend on the amount of backup data, the bandwidth capacity and the availability of the data owner. Peer i with upload and download bandwidth u_i and d_i, starting the backup of an object of size o at time t, completes its backup at time t', after having spent $\frac{o}{u_i}$ time online. Analogously, i restores a backup object with the same size at t'' after having spent $\frac{o}{d_i}$ time online. We define $minTTB_i^t = t' - t$ and $minTTR_i^t = t'' - t$. We use these reference values, without the indexes when applied generally, throughout the paper to compare the relative performance of our P2P application versus that of such an ideal system.

Note that TTB is generally several times longer than TTR. First, in the retrieval phase, peers are not likely to disconnect from the Internet. Second, most peers have asymmetric lines with

fast downlink and slow uplink; third, backups require uploading redundant data while restores involve downloading an amount of data equivalent to the original backup, as we show later. Because of this unbalance, we argue that it is reasonable to use a redundancy scheme that trades longer TTR (which affects only users that suffer a crash) for shorter TTB (which affects all users).

4.2 Redundancy scheme

In a backup service, the backup copy of data must be durable, *i.e.*, existing for the time during which the user relies on the service. Due to the unreliability of storage locations in the P2P setting we are considering, durability of backup is difficult to achieve. A simple management scheme that ensures the availability of the backup by transferring data every time a node goes offline would cause unbearable overhead. Instead, data replication and redundancy have been suggested as effective means to cope with poor peer availability. If the redundancy is high enough, a sufficient number of peers will be online to make data accessible most of the times.

The right choice of the data redundancy scheme is important in order to achieve an acceptable quality of service in terms of DLP, TTB and TTR. As discussed in Chapter 2, the goal of our backup system is not to provide the continuous online availability of each backed up file. Instead, we aim to ensure the durability and fast accessibility of the *whole* backup. Here we show that these goals are achievable by lower redundancy rates than what is generally applied in P2P storage systems, therefore imposing less storage load (or equivalently allowing for backing up more data).

4.2.1 Data structure

Data redundancy can be achieved by replication or erasure coding. We choose one-level erasure coding (*e.g.*, Reed-Solomon codes [112]) to add redundancy to the backup data stored on peers. In our approach each user organizes its files, to be backed up, into *backup objects*, placeholders with size o. Any byte array of size o can be stored in a backup object, larger files are cut in multiple backup objects. Each backup object is then split into k original *fragments* of size f.

A backup object is akin to the idea of an archive of a predefined portion of user data. Applying this approach simplifies the problem of managing the backup of individual files because we only deal with fixed size objects that we can split always in the same number equal-sized fragments.

The erasure coding scheme encodes each backup object by producing $n = rk$ encoded fragments out of the k original fragments ($n > k$); we term r the redundancy rate of the coding scheme. We regard storing two fragments of the same backup object on a given remote peer as a violation of the redundancy scheme, because otherwise a remote peer crash would cause the loss of multiple fragments. Therefore each encoded fragment must be backed up separately. Thus,

the necessary number of remote peers is $n = r\frac{o}{f}$.

The original backup object can be reconstructed based on any k encoded fragments out of the total n, where the combined size of the k fragments is equal to that of the backup object. In the following, we will denote the necessary number of fragments to restore a backup object as k. The value of r must be set appropriately, depending on the desired backup object durability target, *i.e.*, the probability of being able to reconstruct the original data.

The purpose of placing user data in backup objects before applying erasure coding on them is two-fold. First, it avoids repeating the process of erasure coding on the whole amount of backup data every time the user modifies its files; second, constructing backup objects allows for creating a predefined number of fixed-size fragments out of each of them. If many fragments were created directly from a large amount of backup data, the low number of remote peers might hinder dispersing all of them (each fragment of an object must be stored on a distinct peer). Nevertheless, if more fragments of different backup objects, although belonging to the same user, are uploaded to a remote peer, the durability of the whole backup is compromised. We limit our analysis to the durability of backup objects in the system.

4.2.2 Adaptive redundancy rate

The backup system ensures that when local data is lost, it can be restored from the backup, stored on remote peers, within a given time frame with high probability. Data recovery is not instantaneous due to the finite download capacity of realistic peers. Therefore, we propose to set the redundancy rate in order to ensure a given "data retrieval ability" based on the time required to download at least k fragments and based on the corresponding DLP, instead of focusing on the prompt availability of backed up data.

Our novel adaptive redundancy scheme not only ensures the durability of backup, and provides guarantees that in case of retrieval, the whole backup can be downloaded within reasonable time, but it also adjusts the redundancy rate of each backup object according to the current system characteristics. In this perspective the data redundancy policy that we propose suits the purpose of backup service, and also brings the benefits of significantly lower resource consumption than storage systems.

The scheme determines an adaptive redundancy rate for every user separately, tuning it according to the storage peers of each. For this reason, each peer holds an estimation about its DLP and TTR at any given time. As shown later, the redundancy rate is adjusted continuously with the aim of yielding a target quality of service in terms of these metrics.

Estimating TTR

The length of TTR after a local crash at a given moment in time is hard to predict due to the unreliable nature of remote peers. We introduce the "estimated TTR" (eTTR) as a heuristic guess of TTR which can be computed for peers that have already uploaded at least k fragments.

The eTTR is calculated as the time a peer needs to download k fragments from its kth "fastest" remote peer. For every storage peer, the product of its long-term time average of the probability to find the peer online, called as availability (a) and its average upload bandwidth (u) yields its estimated speed during retrieval. Formally, $a = \frac{1}{t_c - t_0} \int_{t_0}^{t_c} \mathbb{P}^t(\text{peer is online}) dt$, where t_0 denotes the point in time where the peer started to use the service, and t_c is the current time. We approximate the TTR of a peer with the kth largest among these values.

We write our heuristic eTTR in (4.1): let the kth highest au product among the storage peers be that of peer j. When retrieving backup from more storage peers with high availability and uplink capacity, the downlink of the retrieving peer might be saturated with maximum p^d parallel downloads. The lowest threshold on eTTR is given by $minTTR = \frac{o}{d}$, which is attained only if no useless unfinished fragment parts are downloaded with the download capacity d.

$$eTTR = \frac{o}{\min\left(a_j u_j p^d, d\right)} \tag{4.1}$$

Note that in the eTTR formula we implicitly assume that uploading remote peers do not divide their upload bandwidths among parallel connections (with upper limit p^u). We motivate this consideration with the following: crashes are supposedly infrequent, and retrieves do not require to download redundant data. Therefore, the amount of data used for restore is much lower than the backups, rendering concurrent retrieves unlikely, unless the system is already overloaded by backups. In that case, the retrievals could not be possible at all, for the lack of completed backups. Moreover, retrievals receive the highest priority among data transfers, *e.g.*, backup uploads.

Estimating DLP

Upon a crash, a peer with n fragments placed on remote peers can lose its backed up data if more than $n - k$ of them crash as well before k fragments are completely retrieved. Considering a delay D that can pass between the crash event and the beginning of the retrieval phase, we compute the estimated DLP (eDLP) within a total delay of $t = D + eTTR$.

If we consider peer crashes to be memoryless events, with constant probability for any peer and at any time, the times before crash are exponentially distributed stochastic variables with a parametric average $\overline{t}$: a peer crashes by time t with probability $1 - e^{-t/\overline{t}}$. In this case:

$$eDLP = \sum_{i=n-k+1}^{n} \binom{n}{i} \left(1 - e^{-t/\bar{t}}\right)^{i} \left(e^{-t/\bar{t}}\right)^{n-i}. \tag{4.2}$$

We advocate an *adaptive* redundancy scheme that is based on the eTTR, and handles the unavailability of backup data flexibly. The scheme focuses on reaching relatively low TTR values, while keeping the estimated DLP low. After generating encoded fragments, each peer uploads an increasing subset of those until its eDLP *and* eTTR fall below predefined thresholds. Otherwise, more fragments are stored on remote peers, and the *adaptive* redundancy rate is continuously increased.

4.2.3 Redundancy maintenance scheme

Every user keeps a copy of their own backup locally, but all data get lost (stored fragments of remote peers as well) if they crash. In our approach the crashed peer retrieves its backup, remote peers storing on the crashed peer repair their redundancy: backup maintenance is performed by the data owners themselves. To minimize DLP we apply the following operations.

A peer performs *retrieval* after it loses the local copy of its data. A retrieval is successful if the crashed peer downloads at least k complete fragments. A crashed peer always puts priority on retrieving its backup as fast as it can after crash: it downloads k fragments to restore its own data from the peers first, and becomes available to receive fragments from remote peers once it has completed its restore. Similarly, the storage peers prioritize the uploads towards the retrieving peer.

If a peer loses a fragment stored on a crashed remote peer, it initiates the *repair* operation: re-generates the fragment and uploads it to a storage peer. The crashed peer blocks downloads of the re-generated fragments during its retrieve, therefore in order to perform the repair as soon as possible, the lost fragment is possibly uploaded to another storage peer. The amount of repair traffic depends on the storage peer lifetimes and the backup data size. Constructing small fragments, therefore storing on many peers, leads to frequent repairs; if fragments are large, repairs need to be performed rarely but the aggregate traffic is essentially the same in all cases.

If peers start to retrieve their backed up data *instantaneously* after they crash, and remote peers are also informed about the occurred crash (to launch data repairs), the redundancy rate might be lowered significantly without threatening data reliability, since the TTR is negligible compared to peer lifetimes. However, the crashed peer may remain offline for a given time after its crash, postponing its data retrieval, thus risking a fatal drop in the number of redundant fragments still stored on remote peers. In this case, since remote peers are not alerted about the crash, the status of the peer can be easily considered as a temporary offline status and the fragments stored on it are not assumed to be lost instantly.

In our system, peers are notified by the tracker about the crash of a peer, holding their

fragments, only after it has been offline for a predetermined time period w. This period w and the redundancy rate must be well-combined in order to be able to react fast to fragment losses, but without many unnecessary actions triggered by false positives.

4.2.4 Assisted repairs

Peers are not responsible to maintain the data redundancy of any other peer. In order to mitigate the undesired consequences of delayed repairs, we leverage on the reliability of a data center with high, although expensive storage and bandwidth capacity. The tracker *assists* to crashed peers while they are offline, and organizes the repair of their decreased data redundancy with the services of a data center.

After a peer has been offline longer than time w, the tracker alerts the remote peers about the supposed loss of their fragments. Starting from this point, the tracker starts to repair the redundancy of the offline peer *if* its eDLP exceeds a threshold, calculated for a time window that is set to the eTTR of the data center *in addition* to $2 \times w$. The eDLP is calculated for this latter duration because when the peer crashes, it might have already lost fragments on remote peers that had crashed less than w before its crash, without notification.

Repairs by the tracker maintain data redundancy when the data owner is not online to carry out repairs itself. Those are performed as follows. In an emergency situation, the data center downloads fragments from storing peers to be able to create new encoded fragments when it has k complete fragments. Afterwards it resumes the uploads of partly transferred fragments to remote peers, and uploads new fragments to new peers. In case the crashed peer returns online during the tracker-assisted repair, retrieving by tracker has priority from all online storing peers. Similarly, repairing by tracker has priority to backup uploads by the peer.

4.3 Grouping peers by design

Random peer selection leads to storing fragments homogeneously in the P2P overlay, therefore this scheme does not consider peer heterogeneity. Peers that are rarely found online and/or have slow connection, receive less storage load, while the capacities of "good" characteristic peers are saturated. The phenomenon where *selfish* peers, users who strive to decrease any contribution that implies cost, exploit the shared resources of more altruistic participants without sharing their own, is called "free-riding" [52]. The unfairness of loading backup on peers that contribute more resources may easily lead to free-riding, which results in a significant drop of the quality of service.

As it is proved to create incentives for cooperation in P2P file sharing applications, we introduce a tit-for-tat approach in terms of resource sharing. As a first step, we enforce a storage contribution from peers equivalent to their usage in the system. Furthermore, contrary to many

P2P storage applications where data load is spread randomly among remote peers, we exploit the freedom that each peer has when choosing fragment holder peers. Our scheme for the selection of remote storage peers is symmetric, *i.e.*, requires pairwise consent, and every bilateral acceptance is based on the characteristics of the other party, namely on its online availability and upload bandwidth capacity. This framework ensures tit-for-tat in every relevant type of resources.

In Chapter 5 we analyze a system model with user-driven peer selection, and we show the advantages of the scheme. An important consequence of such a data placement policy is that the system is stratified and thus peers receive different quality of service according to their heterogeneous resource contribution. Since users with more shared resources are favored, the system inherently embraces fairness and incentives which mitigates the negative effects of user selfishness. On the other hand, while the peer selection mechanisms result in cooperation incentives, the benefits of the scheme are counterbalanced by a loss in efficiency due to the stricter peer selection norms.

Building on the observed phenomenon in the stability analysis of the model presented in Chapter 5, we define classes for peers with similar availability and bandwidth parameters, and we enforce that peers exchange fragments only inside the classes. This system design choice anticipates the outcome of a potential user-driven peer selection scheme, and enforces that peers follow the strategy they would choose anyway. The benefit is that there is no need to perform complicated algorithms and data transfers every time a new peer arrives in the system.

The peer selection rules in our system therefore are the following. A user may belong to only one class and it selects remote peers randomly within the class. The symmetric contribution of storage and traffic puts less burden on peers that contribute high online availability and bandwidth, thus it creates incentives for users to improve their grades (the product of the average availability and dedicated upload bandwidth of a peer). If a peer expects higher payoff due to a better quality of service, then it may improve its grade to become a member of a higher grade class.

This schemes also brings shortcomings due to the strict symmetry.

Symmetric exchanges force a peer that loses a fragment on one of its crashed remote peers to upload it again to the crashed peer after this latter's retrieval phase, instead of finding a new partner sooner. As an explanatory example let us suppose that peers i and j store one-one fragment for each other, and i subsequently crashes. In this case, j has lost its fragment at i and has to keep the fragment of i to make it possible that i recovers its backup data. Since j cannot reclaim new storage space, and uploading the lost fragment to a new remote peer would imply storing one fragment for the new partner, j should delete the fragment of i. This would lead to a high number of fatal crashes, where the storage peers destroy the fragments of the crashed peers, making backup retrieval impossible. Instead, one or two fragments are uploaded to i: the fragment that i stored on j, if it is needed to recover the backup of i, and the one that

j stored on i before. In this perspective long term symmetric contracts between peers imply less flexibility and longer repairs, in return for creating fairness and incentives.

Furthermore, during the assisted repairs in the symmetric scheme no *new* exchanges are initiated by the data center, since in most cases they would be unsuccessful due to timeout on bilateral transfers (the crashed data owner probably being offline), thus resulting in wasted uploads from data center.

4.4 Assisted backup

We have proposed novel schemes for the most important facets of a P2P backup system to manage limited peer resources in an efficient way. The adaptive-rate redundancy policy imposes a lower storage burden on peers, the grouped peer selection ensures fairness and encourages users to contribute more online time and bandwidth to the service. However, the shared peer resources are finite and might not satisfy the needs of the desired quality of service.

The unavailability and poor connectivity of remote peers may hinder fast backup and retrieval. We apply a timeout on each fragment transfer, so if the fragment is not completely uploaded within the given period counting from the start time of the transfer, *e.g.*, because the two parties are rarely both online at the same time, then it is interrupted and canceled. Furthermore, in order to avoid bandwidth underutilization due to asymmetric bandwidth characteristics, the maximum parallel uploads and downloads for a peer are constrained by upper limits that are adapted to the uplink and downlink capacities of the peer.

Inspired by the assisted repairs, we propose to exploit the hybrid structure of the system, with a central tracker, in order to further decrease the inefficiency of the backup process with remote peers. Besides performing the monitoring and matching of peers and providing assistance to repairs, we consider that the tracker stores fragments in a data center when peer resources are insufficient, in order to improve the quality of service, especially TTB. With the help of the tracker, peers complete their backups sooner.

Storing backup in the data center is costly on the long term, but our system design exploits this option only when necessary and rather as a temporary haven. Once there is an opportunity to offload the data center, backup is moved to peers. Consequently, our system design combines the benefits of both approaches and creates synergy: quick backups, thus immediate safety, and low costs.

The various reasons that motivate backing up on costly data center all amount to the scarcity of distributed resources: the unavailability, the limited storage, the low bandwidth and/or the low number of remote peers that may store the encoded fragments. If any of these constraints exist, despite our redundancy scheme, some peers might not receive an acceptable quality of service in a purely P2P setting. Generally, the unavailability and poor connectivity of remote

peers prolongate TTB, limited storage capacity and small system size restrict the applicable redundancy rate that might be insufficient to ensure a low DLP.

4.4.1 Data center storage

Data centers have characteristics of reliability, bandwidth and availability that are well beyond the capabilities of even very good peers. However, the reliable central resources come with costs: the price of storage is given per storage duration and per data amount, inbound and outbound traffic fees are determined by the transferred data amount.

While long-term storage on data center and related traffic requires payments from the users, excess storage and bandwidth capacity of peers are free, but limited, *e.g.*, uplink/downlink may be bottlenecks in data transfers. On the other hand, storing on peers improves *reliability* since private backup data are not in the hands of a single company that may go bankrupt. Nevertheless, along with the monetary burden, a data center assisted approach is able to ensure the quality of service that is unachievable by a purely P2P setting in most cases.

As seen in Section 4.2, the fragments stored on remote peers must meet a redundancy rate that depends on their characteristics. On the contrary, the centralized nature of the data center implies that a user should not store more than k fragments on it. Moreover, by applying our adaptive redundancy rate scheme, users upload to the data center only a number of fragments that is necessary to meet the targets of DLP and TTR, while they upload as many fragments as possible to remote peers. When estimating DLP and TTR, the fragments backed up on the data center are considered to be always reachable with full download capacity, and perfectly safe from loss.

As soon as the backup phase is completed, with fragments on remote peers and on the data center the "hybrid" rearranging phase is started, which is slightly altered compared to the purely P2P setting. In the rearranging phase a peer continues to upload fragments to remote peers and deletes some of its fragments on the data center if the remaining fragments and the fragments on the peer set altogether guarantee the aimed TTR and DLP. The rational behind offloading the data center is that it is less costly to exchange more encoded fragments with remote peers than the long-term storage of fragments on the data center, both yielding the target DLP and TTR.

4.4.2 Data placement during backup

The bandwidth capacity of each peer must be shared between uploads to the data center and to remote peers during its backup phase. The scheduling strategy directly determines the placement of fragments, distributed between the central and the P2P storage. Our data placement scheme prioritizes uploads toward remote peers and transfers fragments to data center only with excess upload capacity, *i.e.*, which can not be used for uploading toward peers, in order to keep storage

cost as low as possible while exploiting the upload capacity to the fullest.

Thus, fragments are transferred to the data center when the backup operation on remote peers fails temporarily. Our scheme leads to a system where the data center stores the least load possible while guaranteeing that TTB does not grow due to the unavailability of remote peers. If the availability and bandwidth constraints of remote peers cease to exist, more fragments are uploaded to them and the data center is offloaded in the meantime.

The need for central storage capacity may increase if there are not enough suitable peers for exchange. In this case the uploaded fragments cannot be deleted from the data center for a possibly long time. However, when a match can be made between two peers, both storing on the server, the data can be re-transferred between the peers to offload the server.

In case a peer that stores fragments on the data center, crashes during the backup phase, it prioritizes downloads from peers storing its fragments, and only saturates its excess download capacity with transfers from the data center. This is motivated by the following facts: remote peer uplinks may be a bottleneck; remote peers might lose fragments in the meantime and the data center is considered perfectly reliable; and downloading from remote peers implies no traffic cost.

In summary, transferring data to/from the data center during backup and retrieval phases respectively is performed as an additional upload/download thread that saturates the uplink/downlink of the peer. The pseudo-code of the backup phase, assuming at most one fragment on each remote peer $m = 1$, is shown in Algorithm 1. For sake of completeness, we provide the pseudo-code of the rearranging and retrieval phases in Algorithm 3 and 4.

Algorithm 1 Assisted backup phase

Require: *target*: aimed DLP and TTR levels
$\{frag\}$: set of fragments to upload
$\{peer\}$: set of remote peers
p^u: number of maximal parallel uploads
J: list of upload jobs $(peer, frag, comp)$ described by the remote *peer*, the *frag*ment and the transfer *comp*letion percentage, sorted by decreasing order of their *comp*
while $eDLP$ or $eTTR$ is larger than *target* **do**
  Algorithm 2 with p^u, J
  for all $(peer, frag, comp) \in J : peer == 0, comp == 0$ **do** {No more resumable transfers}
    select online *peer* that does not store any of $\{frag\}$
    if |current jobs| $< p^u$ & $peer \neq null$ **then**
      start job $(peer, frag, 0)$
    end if
  end for
  saturate uplink with transferring subset of $\{frag\}$ to data center
end while

Algorithm 2 Transfer fragments

Require: p: number of maximal parallel transfers J: list of jobs
for all $(peer, frag, comp) \in J : comp = 1$ **do** {Remove finished jobs}
  $J = J \setminus (peer, frag, comp)$
end for
for all $(peer, frag, comp) \in J$: expired transfer timeout **do** {Reset prolongated jobs}
  $peer = 0, comp = 0$
end for
for all $(peer, frag, comp) \in J : comp > 0$ **do** {Resume interrupted transfers}
  if |current jobs| $< p$ & *peer* is available for data transfer **then**
    continue job $(peer, frag, comp)$
  end if
end for

Algorithm 3 Assisted rearranging phase

Require: *target*: aimed DLP and TTR levels

$\{frag\}$: set of fragments to upload

$\{peer\}$: set of remote peers

p^u: number of maximal parallel uploads

J: list of upload jobs $(peer, frag, comp)$ described by the remote *peer*, the *frag*ment and the transfer *comp*letion percentage, sorted by decreasing order of their *comp*

if $eDLP$ or $eTTR$ is larger than *target* **then** {Fragments are lost on remote peers}

Algorithm 1 with the same inputs

else

while fragments are stored on the data center **do**

Algorithm 2 with p^u, J

for all $(peer, frag, comp) \in J : peer == 0, comp == 0$ **do** {No more resumable transfers}

select online *peer* that does not store any of $\{frag\}$

if |current jobs| $< p^u$ & $peer \neq null$ **then**

start job $(peer, frag, 0)$

end if

end for

if both $eDLP$ and $eTTR$ are lower than *target* **then**

delete a fragment on the data center

end if

end while

end if

Algorithm 4 Retrieval phase

Require: $\{frag\}$: set of fragments stored on peers and data center
$\{peer\}$: set of remote peers
p^d: number of maximal parallel downloads
J: list of download jobs $(peer, frag, comp)$ described by the remote *peer*, the *frag*ment and the transfer *comp*letion percentage, sorted by decreasing order of their *comp*

while less than k fragments are retrieved per backup object **do**
  Algorithm 2 with p^d, J
  for all $(peer, frag, comp) \in J : comp == 0$ **do** {No more resumable transfers}
    if |current jobs| $< p^d$ & *peer* stores $frag$ required to retrieve at least k **then**
      start job $(peer, frag, 0)$
    end if
  end for
  if data center stores subset of $\{frag\}$ required to retrieve at least k **then**
    saturate downlink with transferring subset of $\{frag\}$ from data center
  end if
end while

Chapter 5

User-driven peer selection

In our system the peers are grouped by design based on their resource characteristics. In this chapter we investigate what would happen if peers were left to select their storage partners freely. We model the process of choosing storage partners as a game where players can vary their storage, online availability and bandwidth contributions, and select remote peers for fragment exchanges strategically. We analyze the game by decomposing the user decisions into the strategies on these previous components and we derive the equilibrium solutions by combining matching theory and game theory: we show that our game can be reduced to a matching problem. We apply the well-known stable fixtures problem [73] to solve it.

5.1 Selection based on grades

We formalize the shared storage space of peers as the number of fragments they can store, denoted by $\hat{c} \in \mathbb{N}$, and we impose symmetry in storage space contribution (as in Chapter 4): a peer must share the same amount of storage capacity that it uses on its remote peers altogether. Moreover, we only allow for bilateral fragment exchanges between peers, and support equal resource contribution, we set the size f of a fragment to a system-wide constant. As a result, peers are compelled to provide fair contribution in terms of storage space: they must offer the same amount of local storage that they use at other peers in the system.

Assuming selfish behavior of users, we suppose that they locally optimize the way they operate in the system. Our redundancy scheme takes the online availability and dedicated bandwidth of remote peers into account to estimate DLP and TTR. We presume that users choose partners among the remote peers in order to attain the required DLP and TTR targets with the lowest adaptive redundancy rate possible. Since remote peers with high availability and fast connectivity ensure short eTTR, and hence lower eDLP, a *selective* peer, that chooses those remote peers, may decrease its data redundancy rate, and consequently the storage capacity it must share, resulting in less bandwidth burden as well.

It is therefore important to consider the *grade* of a peer, which we define as the product of its average online availability (a), and of its upload bandwidth dedicated to the system (u). The grade of a peer characterizes its *effective* upload capability: since it can be exploited only if the peer is online, the average upload bandwidth the peer contributes is less that its uplink capacity, in fact reduced by its availability. Intuitively, the symmetric storage contributions ensure the fairness in the "quantity" of storage a peer offers and gets, while peer grades refer to the "quality" of the shared backup capacities, as we apply the same measures in our redundancy scheme in Section 4.2.

We assume that the grade of a given peer is the same from the perspective of any other peer. Thus, peer grades determine a global ranking that is used during the execution of the peer selection mechanism, defined as follows.

Definition 3 ***Peer selection model***

- *Let $\mathcal{I}$ denote the set of peers taking part in the system; $I = |\mathcal{I}| > 1$.*
- *Every peer $i \in \mathcal{I}$ splits each of its backup objects into k original fragments, out of which it creates $\hat{c}_i$ redundant fragments of the same size, to store on separate remote peers.*
- *Then every peer establishes a set of links, denoted by n_i for peer i: $n_i = \{j \in \{\mathcal{I} \setminus i\}\}$, when peer i stores a fragment on peer j. By policy, no more than one fragment from a backup object can be stored on the same remote peer.*
- *Our symmetric fragment exchange design dictates that $i \in n_j$ if $j \in n_i$,* i.e., *peers i and j are storage partners. Therefore peer i must share at least the local storage capacity equivalent to $|n_i|$ fragments, $|n_i| \leq \hat{c}_i$.*
- *$\hat{g}_i = a_i u_i$ characterizes peer $i \in \mathcal{I}$, and we call it as the* effortless *grade of i,* i.e., *the attributes composing it do not require additional strain from i other than its normal behavior. The effortless grade gives a lower bound on the strategically selected grade g_i of any given peer i.*

5.2 User satisfaction

We extend the above peer selection model in order to reflect the quality of service provided by a set of storage partners. We introduce a game theoretic *value* function that degrades when fragments are not spread among enough remote peers, exposing the peer to higher chance of data loss, as discussed in Section 4.2. Similarly, the value is decreased when storage partners have low grades: up/downloads are faster to better connected peers, thus the required data redundancy is lower with higher grade partners, *i.e.*, the user must share less storage and produces less traffic.

We allow the users to strategically change their contributions, and as a result, their grades. When a user "upgrades" itself from its effortless grade, in order to be able to make links with high grade peers to achieve higher quality service, this improvement implies cost. To this end, besides the value, we define a *cost* function. An effort may consist of, *e.g.*, leaving a storage device online for longer periods, and/or dedicating more connection bandwidth to the service. The more a user increases its grade from the effortless level, the more its effort cost grows.

The *payoff* integrates the above value and cost of users in our P2P backup system. Constructing an objective function that would explicitly account for all aspects of the quality of service is a challenging exercise. Moreover, the optimization of such a function during the peer selection would require sophisticated algorithmic techniques. We define the following heuristic payoff function, striving to obtain reasonable balance on the trade-off between the realistic description of the quality of service and the complexity of the model. Our payoff is composed by two terms, the value of the service and the effort cost; the latter is the difference between the chosen and the effortless grade of the peer.

Definition 4 ***The payoff*** *function that every peer $i \in \mathcal{I}$ maximizes when establishing links to remote peers, is:*

$$p_i = \min\left(|n_i|\underline{g}_i, 1\right) - (g_i - \hat{g}_i),$$

where $|n_i|$ denotes the number of partners of i and $\underline{g}_i = \min_{j \in n_i}(g_j)$ is the lowest grade among them.

A complex value term would consider all the grades in the peer set, but this would lead to an intractable theoretical model. Instead, our heuristic value function reflects the attributes of the partners by their number and the worst grade among them. If either of these values increases, the value improves too. In order to ensure the validity of these heuristics, we normalize grades so that k peers holding the highest possible value could provide a perfectly reliable backup service on their own by storing one of the k original fragments each. In this case, we the suppose that the maximal value is reached, defined for simplicity to be 1. For a perfectly reliable service, we must assume that a peer with the maximal grade $\frac{1}{k}$ never crashes, *i.e.*, the availability of data stored on it is always exactly 1 and its uplink capacity cannot be saturated by the download capability of any peer. We divide the product au of every peer by this theoretical upper bound of reliability, resulting in $g_i \in (0, \frac{1}{k}]\ \forall i \in \mathcal{I}$.

We emphasize the fact that this value function glosses over our adaptive-rate redundancy scheme. When a user has multiple backup objects, fragments within a backup object must be stored on different peers, but it is highly probable that fragments that belong to different backup objects will be stored on the same peer due to a possible lack of sufficient number of remote storage locations. This results in the exchange of multiple fragments between two given peers.

In this chapter, we are oblivious to the data structure presented in our system design. We avoid dealing with multiple backup objects.

With the aforementioned normalization, the grade of a common peer is much lower than 1, therefore redundant fragments of a backup object must be stored on more different peers to reach the maximal value, *i.e.*, quality of service, than what is required by our adaptive rate usually. While in Section 4.2 our goal is to show the minimal redundancy rate that satisfies certain quality requirements, *i.e.*, low data loss probability and short retrieval times, in this chapter, our focus shifts from the required number of remote peers to their grade selection.

Note that having more partners than $\frac{1}{\underline{g}_i}$ does not increase the value beyond 1. It is not decreased either, therefore there is no reason why matches should not be created among more peers having at least a grade $\underline{g}_i$. The payoff function glosses over the fact that these unnecessary matches impose further storage capacity sharing in order to place the exchanged fragments. We will show later that multiple equilibria may exist consequently.

Nevertheless, the value function reflects the characteristics of our redundancy scheme presented in Chapter 4. The adaptive redundancy rate is based on the eTTR which calculates with the kth lowest grade among the remote peers that hold fragments of a backup object. The worst grade among the partners yields a reliable estimate for this latter in most cases. As a consequence, a lower worst grade increases eTTR, thus higher redundancy must be applied, leading to higher resource utilization. For a same level of quality of service, more remote peers must be contacted so that higher grade partners could enhance the eTTR and decrease the usage of storage space and bandwidth.

An informal interpretation of the payoff function is as follows. There are two forces that drive the decision process of each player. On the one hand, as we will show later, the player must increase its grade in order to have high grade partners, hence smaller required peer set which implies less data redundancy, on the other hand, the effort cost drives the player to a low grade.

5.3 The exchange game

Our peer selection model assumes that users optimize their individual payoffs described in Definition 4 when selecting partners for fragment exchanges. We describe the peer selection process by a game [56] during which players selfishly set their logical neighbors and grades. By altering their grades, players strategically change the amount of offered resources (online availability, dedicated upload bandwidth capacities), directly determining the remote peers they are willing to cooperate with in the P2P system and their shared storage.

Definition 5 ***The exchange game*** *is defined by the collection of player strategy sets* $\{\mathcal{S}_i \; \forall i \in \mathcal{I}\}$, *and the payoff function* p *that yields user payoffs* $\{p_i \; \forall i \in \mathcal{I}\}$ *on the combination of the strategy*

sets ($p : \mathcal{S}_1 \times \cdots \times \mathcal{S}_I \to \mathbb{R}^I$). *A strategy of player* $i \in \mathcal{I}$ *consists of a grade* $g_i \in (0, 1]$ and *a set of links* n_i.

Stability is a desirable attribute of a system, therefore we seek equilibrium states of the game, in which users will not change their situation unilaterally. The strategy space that each player can explore does not only consist of the grades they can attain by improving their effortless qualities, but also in the meantime they can choose the remote peers, based on their strategic grades, with whom they want to create exchange partnerships. Since consent is needed from selected peers and they are free to change their grades as well, a stable state of the game is rather hard to characterize at first glimpse. Our setting resembles to network creation games [49], where players create links to one another in the hope of improving their global connectivity, *i.e.*, the ability to reach every other player in the system. In both cases the payoff of every player depends on the set of partners they choose, and the strategies of these latter of course.

The exchange game is in Nash-equilibrium if none of the players can further improve its payoff by itself. In this situation, the strategy, called best response, of player i is g_i^*, n_i^* such that $p_i(g_i^*, g_{-i}^*, n_i^*, n_{-i}^*) \geq p_i(g_i, g_{-i}^*, n_i, n_{-i}^*)$ holds for any $g_i, n_i \; \forall i \in \mathcal{I}$, where g_{-i}^* and n_{-i}^* denote the best response strategy set of the others. Hindered by the complexity of determining best response strategies that create equilibrium, we dissect the joint optimization process: we analyze the grade and peer selection algorithmic problems separately.

By decomposing the optimization problem of each player in the game to find its best response strategy, we assume that player decisions regarding the selection of their own grades and their remote peers are interleaved. For the sake of tractability, we reverse the order of the decisions that the users are assumed to make. First, we cast the peer selection as a stable fixtures problem. Then, we show how the peers are stratified along their grade order in a stable matching. Building on this observation, we provide the best response grade strategies of the users, given an effortless grade profile as input to the game. Once the equilibrium grades are established, the stable fixtures algorithm builds the overlay of fragment exchanges among the peers.

5.4 Stable fixtures problem

During the exchange game, each player picks multiple partners in a dynamically varying environment where player grades may evolve. While strategies are individually made, in a resulting stable overlay of peers matches have to be consensual and no player should be tempted to deviate because another possible match would be better for them. This inspires us to formalize the peer selection step of the above interleaved algorithm as a *matching problem*.

Indeed, matching theory studies algorithmic solutions to find stable pairings in such problems, with "static" player characteristics, *i.e.*, grades in our case. In our exchange game if grades were not strategically improvable, then the game would reduce to a combinatorial matching problem.

We define our setting as a stable fixtures problem, presented by Irving and Scott in [73], which is, in turn, a multi-match variant of the stable roommates problem [72]. Based on Definitions 3 and 4, we illustrate that any possible peer selection setting can be described with a problem instance.

Definition 6 ***The stable fixtures problem*** *describes the peer selection step of our interleaved algorithm, which is essentially a stable matching problem with fixed player grades and selection preferences. The solution to the problem is the neighbor set* n_i $\forall i \in \mathcal{I}$ *(defined in Definition 3).*

A match represents the exchange of two fragments between two players. Capacity $\hat{c}_i$ $\forall i \in \mathcal{I}$ *corresponds to the number of encoded fragments* i *can exchange, thus* $\hat{c}_i$ *is the upper bound on the number of its matches.*

The preference (sorted) list P_i $\forall i \in \mathcal{I}$ *consists of all potential matches of* i*. Based on Definition 4, we define the preference of the match with* j *(denoted by* $P_i(j)$ *by an abuse of notation) as the value of the service when no other matches are made,* i.e., *the increase of payoff* p_i *if the fragment exchange with player* j *is established first among the matches of* i*. The matches are ordered by non-increasing preferences in* P_i*.*

A matching $\mathcal{M}$ *is a set of pairwise matches. Matches are pairwise if for every match* $j \in n_i$, $i \in n_j$ $\forall i, j \in \mathcal{I}$ *holds.* $\mathcal{M}$ *is stable if there are no two nodes* i *and* j *that have an incentive to create a match, possibly revoking matches from other players in the process. More formally, there is no match outside* $\mathcal{M}$*, such that*

- *either* i *has fewer matches than* $\hat{c}_i$ *or* $P_i(j)$ *is greater than* P_i *of at least one of its matches in* $\mathcal{M}$*;* and
- *either* j *has fewer matches than* $\hat{c}_j$ *or* $P_j(i)$ *is greater than* P_j *of at least one of its matches in* $\mathcal{M}$*.*

In order to solve problem instances that are produced in the peer selection step of our interleaved optimization, we use the algorithm presented in [73], in which players propose to one another in a distributed manner to create pairwise matches. We describe the algorithm in the following and write its pseudo-code in Algorithm 5.

The algorithm starts with preference list P_i $\forall i \in \mathcal{I}$ consisting of all the possible matches with remote peers (Line 1), as defined in Definition 6. In order to construct the set of stable matches, every player eliminates those matches from it that cannot be stable. Users initiate match creation, following the decreasing preference order of the possible matches in P_i $\forall i \in \mathcal{I}$, at the corresponding remote peers. These initiatives are called bids.

Throughout the distributed algorithm, players delete those matches from their preference sets that could not figure in stable matching. We define the sets of proposed (A_i) and received

bids ($B_i \; \forall i \in \mathcal{I}$) (Line 2) in which player i collects the currently valid bids, and we describe how matches are decided not to be eligible for stable matching. At the end of the algorithm, only those matches remain in every player preference sets that are reciprocated, hence stable.

Definition 7 ***Target list*** *$A_i = \{j \in P_i\} \; \forall i \in \mathcal{I}$ consists of the valid bids on the top of P_i which have not (yet) been rejected, such that $|A_i| \leq \hat{c}_i$. Adding a new bid to the set A_i is performed by selecting a new possible partner who is not present in A_i yet.*

Each player i makes the most beneficial proposal bids based on its preference list P_i (Line 6), as long as the sum of the matches does not exceed the maximal number of matches (Line 5). At the point when $|A_i| = \hat{c}_i$, player i halts its bidding, until its matches in A_i decrease. Now we define B_i, and present how matches can be dropped from A_i.

Definition 8 ***Received bid list*** *$B_i = \{j \in P_i\} \; \forall i \in \mathcal{I}$ consists of the bids received by i that it has not rejected (yet), such that $|B_i| \leq \hat{c}_i$. Player i rejects a bid,* i.e.*, deletes it from B_i, if a new bid makes $|B_i| > \hat{c}_i$,* i.e.*, more received bids accumulate than the maximal number of matches (Line 8). The least preferable match based on the preference list P_i is dropped then (Line 9). When a match is dropped from B_i, it is also erased from the preference list P_i, and the target (A_j) and preference list (P_j) of the initiator (j).*

Algorithm 5 Stable fixtures algorithm

```
1:  P_i, ĉ_i ∀i ∈ I: preference lists with all potential matches and capacities
2:  A_i = B_i = 0 ∀i ∈ I: empty target and received bid lists
3:  while A_i ≠ P_i ∀i ∈ I do
4:    for all i ∈ I do
5:      while |A_i| ≤ ĉ_i do {bidding}
6:        copy matches from top of P_i into A_i and to B_j of target j
7:      end while
8:      while |B_i| > ĉ_i do {refusing}
9:        delete the least preferred matches from B_i and from A_j, P_j of the bidder
10:     end while
11:   end for
12: end while
```

The algorithm terminates when $\forall i \; |A_i| = \hat{c}_i$, or A_i contains all non-rejected matches from P_i, but their total number is less than $\hat{c}_i$. At this point, target lists and preference lists coincide for every player (Line 3). The following proposition of [73] is a direct consequence of the algorithmic steps and the payoff function, and makes an important observation about the preference lists.

Proposition 1 *If a match of i is not in the preference set P_i at the end of the algorithm, then it is not stable. Thus, if a player has an empty preference list at the end of the algorithm, then it has no stable matches.*

In a stable matching the created matches are pairwise, *i.e.*, each user has the same target matches as accepted bid matches. The authors of [73] argue that at the end of their presented algorithm, user preference lists may not coincide the actual target lists, thus the possibility of a *rotation* is left open, *i.e.*, a cyclic preference order exists where player i would like to create a match with player j more than with player k, while j prefers k to i, and k likes i more than j. They suggest another algorithm that reduces further the preference lists to arrive at the stable matching (or to the conclusion that none exists) by eliminating rotations. In the following we highlight the differences of our scheme compared to that of [73].

5.5 Stable stratification

We are interested in the characteristics of stable matchings that are issued by our algorithm. In this section we show that players are stratified in stability: matches are created only between peers that have similar grades. This is, in fact, a consequence of the particular attributes that the problem instances have during our interleaved algorithm: player preferences are directly determined by their payoffs, indirectly by the grades of other peers. In regard to this, we define the *global preference order* (GPO) of the peers. Building on the GPO, we derive an important proposition that describes the stratification in stable matchings, and we claim that our algorithm always finds one.

The GPO predetermines the possible partners in a stable matching, we define it as the ordered list of the match preferences with different remote peers: $P_i(j)\ \forall j \in \{\mathcal{I} \setminus i\}$ in P_i. Since $\forall i \in \mathcal{I}$, $P_i(j) \geq P_i(k)$ holds for any given pair $j, k \in \{\mathcal{I} \setminus i\}$ if $g_j \geq g_k$ (in case $g_j = g_k$, $P_i(j) = P_i(k)$), GPO is the decreasing order of player grades for every player.

In our system every user divides each of its backup objects into k equal-size fragments, but different number of redundant fragments are created in order to reach the highest possible value of service quality, defined as $\min(|n_i|\underline{g}_i, 1)\ \forall i \in \mathcal{I}$. The next proposition records the relation between user grades and the worst grade among their partners, and we prove that the necessary number of matches in order to reach the same value of the service is not higher for high grade peers than for low grade peers: $\underline{g}_i \geq \underline{g}_j\ \forall i, j \in \mathcal{I}$ if $g_i \geq g_j$.

Proposition 2 *In stable matching if $g_i \geq g_j$, then the two peers can attain the same value of service with $\underline{g}_i \geq \underline{g}_j\ \forall i, j \in \mathcal{I}$.*

Proof: Let us assume that g_h is the lowest grade among the peers in n_j, and by contradiction let $g_h > \underline{g}_i$. Since the grade of i is not lower than that of j, $P_h(i) \geq P_h(j)$ for peer h, and for

every peer in n_j. As a consequence, h has filed a bid not just to j, but to i too, as did every peer in n_j. i must make matches with h and the others, *i.e.*, $h \in n_i$, since it made matches with peers having lower grades as well. Then i reaches the same value of service quality as j without its partners that have lower grades than h, thus those matches are unnecessary. ■

Proposition 2 claims that a higher grade peer has better quality partners than a lower grade peer while reaching the same value of neighborhood. The proof is based on the fact that if a higher grade player is interested in a match with a lower grade one, then the match is made since the latter is also interested. A consequence is that no match is made between two peers, if the difference of their ranks in the GPO is larger than the capacity of the higher grade player. If this were not true, one of the bids of the high grade player to a player better than the low grade peer would have been dropped to make the bid. This could happen only because all of the matches of the rejecting peer are with higher grade players than the high grade peer, contradicting with Proposition 2.

This result shows that users make matches with remote peers that have grades similar to theirs and low grade peers are less popular during the peer selection step of the interleaved algorithm. Since every player is a potential partner for every participant, players with less matches than what is necessary to reach a maximal value neighborhood in a stable matching are at the end of the GPO: if there are at least $\hat{c}_i$ worse peers in GPO than i, then i can have $\hat{c}_i$ matches in a stable matching. Thus, the worst grade peers might realize less matches than their capacity.

We show that the stable fixtures algorithm solves every problem instance during the interleaved algorithm that finds the exchange game equilibrium. The key of the proof is that the bids are always reciprocated by lower grade peers during the algorithm.

Proposition 3 *Our linear-time complexity algorithm always finds a stable matching in a given problem instance with fixed grades.*

Proof: Let the players place their bids in their order in the GPO; note that this deterministic behavior has no effect on the outcome. The highest grade player bids the first $\hat{c}_i$ matches on its preference list, and based on Proposition 2, all of them will be accepted and reciprocated. After the highest grade peer has found its stable matches, all the bids of the other players targeting it are dropped, and the same claim stands for the best grade player of the rest, and so forth, arriving to stable matching $\mathcal{M}$. The algorithm ensures that there is no possible further pairwise match which yields higher payoff than the ones in $\mathcal{M}$. The complexity of our algorithm follows the number of players and their capacities linearly and can be performed asynchronously. ■

The notion of stability in games is often related to the question of the existence of Nash equilibrium. In our exchange game we need to refine the requirements of stability: since link creation is considered to be the "joint" outcome of two players' strategies, pairwise stability has to be ensured as in [30]. Since the requisite of a "pairwise" equilibrium is a stable matching, if no

stable overlay exists for the given player grades, those cannot be a stable solution to the game instance itself. Since player preferences, which follow the GPO, are acyclic as in [98], a stable matching always exists.

5.6 Grade improvement

The stratification of peers based on their grades (shown in Proposition 2) provides direct incentives to low grade users to improve for a better neighborhood during the exchange game. In the grade selection step, users may strategically increase their grades in order to be eligible for exchanging fragments with high grade peers. The equilibrium of the game that we seek is a stable overlay graph representing (logical) neighborhood relations among peers of the system. In this section we show that peers join disjoint groups in equilibrium, according to their initial effortless grades.

In Proposition 2 we claim that the worst partner of a high grade peer has higher grade, than that of an inferior grade peer. Therefore higher grade peers need less matches, *i.e.*, less capacity, than peers with low grades in order to reach the maximal value of Definition 4. The acyclic preferences ensure the existence of the required stable overlay. We make the following claim on the relation of effortless and equilibrium grades, and therefore we show that the effortless grade order inherently predetermines the order of the equilibrium grades, and thus the shared capacities.

Proposition 4 *The equilibrium GPO is the effortless grade order of the players.*

Proof: The equilibrium GPO contains players in decreasing order of their equilibrium grades, in case of a tie, we sort the players corresponding to their effortless grade order. As a contradiction to our claim, let us assume that $g_i > g_j$ although $\hat{g}_i < \hat{g}_j$. Then the payoff of i is higher with g_i than by choosing g_j as strategy, since $\hat{g}_i < \hat{g}_j \leq g_j < g_i$ implies strictly higher effort cost. This is possible only if the payoff of j is improvable, *i.e.*, $1 \geq \underline{g}_i|n_i| > \underline{g}_j|n_j|$ based on Proposition 2, therefore j can increase its payoff by selecting $g_j = g_i$, thus contradicting with our initial assumption about the best response in equilibrium. ■

We now derive the best response grade strategies for players in a one-shot game with full information. First we consider a favorable setting where players have sufficiently high effortless grades, then we give the equilibrium in a general setting.

Proposition 5 *If* $\min_{i \in \mathcal{I}} \hat{g}_i \geq \frac{1}{I-1}$, *then a possible equilibrium is the effortless grand clique: everyone is linked to every other player. Generally, the best response grade strategy is to join a clique according to the effortless grade order rank. This might necessitate improvement on the effortless grade.*

Proof: Let us number the players in decreasing order of their effortless grades, such that player 1 has the highest grade, and player I has the lowest. Many equilibria may exist depending on the effortless grade profile. If $\exists j : (j-1)\hat{g}_j \geq 1$, then players can maximize their payoffs by flocking into one clique: $g_i = \hat{g}_i$, $\underline{g}_i = \hat{g}_j \ \forall i \leq j$. If the inequality holds for the lowest effortless grade in the system, every player may maximize its payoff to 1 by $I-1$ links to the others without improving their grades, since $(I-1)\min_{i\in\mathcal{I}} \hat{g}_i \geq 1$ and $g_i - \hat{g}_i = 0 \ \forall i \in \mathcal{I}$. If a clique is formed on a specific segment of the effortless grades, and we find a player k satisfying the condition in the rest of the player set, *i.e.*, $(k-j-1)\hat{g}_k \geq 1$, then the same claims hold, and so forth.

Depending on the effortless grades of the players, it may happen that no (more) effortless clique can be formed, *i.e.*, no player satisfies the previous condition on the rest of the player set. Let us denote the set of the "remaining" players by $\mathcal{X}$, and let v be the lowest effortless grade player in any effortless clique. We state that all players in $\mathcal{X}$ join the lowest effortless grade clique by improving their grades to $\hat{g}_v$ if $\hat{g}_v \leq \frac{1}{|\mathcal{X}|-1}$, otherwise they create a clique among themselves by increasing their grades to $g_x = \max(\hat{g}_x, \frac{1}{|\mathcal{X}|-1}) \ \forall x \in \mathcal{X}$.

To prove that these strategic grades are stable, let a player lower its grade by Δg in order to save on effort cost. Then, in the first case, it is excluded from the clique of v because the clique members might see a payoff decrease; in the second case other remaining players would follow its grade reduction resulting in a strictly positive payoff drop of $(|\mathcal{X}|-1)\Delta g - \Delta g > 0$ $(|\mathcal{X}| > 2)$. If the lowest grade player i is the only one with $\hat{g}_i < \frac{1}{|\mathcal{X}|-1}$, then i may decrease its grade as far as $(|\mathcal{X}|-1)g_i \geq (|\mathcal{X}|-2)\hat{g}_j \geq 1 - \frac{1}{|\mathcal{X}|-1}$ holds, where j is the second lowest effortless grade player.

If $|\mathcal{X}| = 2$, and there is no effortless clique, a grade improvement to g_x implies $g_x - \hat{g}_i$ effort cost, while the value of its neighborhood grows by $g_x - \hat{g}_i$ at most, thus resulting in a non-increasing payoff, so we suppose that the best response grade strategy is the effortless grade $(g_x = \hat{g}_x \ \forall x \in \mathcal{X})$. For a discussion on homogeneous settings, see Proposition 6. ■

Proposition 5 claims that players unite in groups, driven by their payoffs based on the size and the worst grade member of the clique. If the group size becomes large, *i.e.*, many peers group together, the players might be better off excluding the ones with the worst grade. If the grade-segment becomes too narrow, *i.e.*, low grades are not eligible, the quality of service declines due to the critical number of member peers. This duality is due to the payoff: low grade clique members decrease the value at high grade members in the group, excluding them causes a drop in the value at the remaining players because of the reduced clique size. In equilibrium the two opposing effects are balanced inside disjoint groups.

When players join cliques by increasing their grades, the implementation of a link re-organizing policy is required in order to integrate the newly joining players as equivalent members into the clique, *e.g.*, more clique members than maximal matches per player might cause the rise of "hot spots", when some members do not receive enough matches. Clustering inside clusters may block

some peers from finding sufficient number of available peers.

In contrast to heterogeneous effortless grade settings, if every player has the same initial grade, they have no incentive to improve. The next proposition provides a detailed explanation to this disadvantageous phenomenon.

Proposition 6 *If $\hat{g}_i = \hat{g}\ \forall i \in \mathcal{I}$, the best response strategy is $g_i = \hat{g}\ \forall i \in \mathcal{I}$. Therefore, in homogeneous effortless settings, players have no incentives to improve their grades. The price of anarchy,* i.e.*, the quotient of centrally maximized social welfare and that of the distributed game, is infinite with $\hat{g}_i \to 0\ \forall i \in \mathcal{I}$.*

Proof: If the effortless grades of peers are the same, the best response strategy grades are also identical, because everyone can strictly increase its payoff by decreasing its grade to that of its lowest grade peer. By contradiction, let us assume that the best response strategy is g^*, and we show that one can increase its payoff with a strategy g' such that $g^* > g' \geq g^*(1 - \frac{1}{I-1})$. If this inequality holds the overlay neighborhood is intact because the partners of the deviating player, that are surely less then I in number, would not profit by dropping the link to it, while the payoff of the deviating player is strictly increased. The resulted contradiction supports our claim that no other strategy than $g^* = \hat{g}$ may be best response.

If $\hat{g}_i \to 0\ \forall i \in \mathcal{I}$, the best response strategies are $g_i \to 0\ \forall i \in \mathcal{I}$, resulting in an aggregate payoff arbitrarily close to 0. On the other hand, a central social maximization would result, by linking all players in a clique with $g_i = \frac{1}{I-1}\ \forall i \in \mathcal{I}$, in more than $I\left(1 - \frac{1}{I-1}\right)$ social welfare. Therefore there exists a strictly positive optimal grade in the homogeneous case if $I > 2$, resulting in an arbitrarily large price of anarchy. ■

Proposition 6 shows that during the decentralized optimization, *i.e.*, the exchange game, players do not make links that imply a negative change in their payoff and improve their grades only if the effort is worth it. In a homogeneous setting this never happens. On the other hand, when a central authority maximizes the aggregate payoff, *i.e.*, the social welfare, with the hypothetical ability of changing player grades, it makes a link even if it is beneficial only for one of the matched players, in case the aggregate payoff increases. As a consequence, unfortunately the price of anarchy can be arbitrarily large.

We have finally shown that we obtain a stratification in the peer overlay, analogous to grouped system design we suggested in Chapter 4. By performing the stratification at the tracker side, we simplify the system while nodes have no incentive to deviate from the tracker-suggested behavior in terms of potential storage partners.

Chapter 6

Scheduling data transfers

The scheduling policy organizes data transfers toward remote targets. Future online appearances of remote peers are not predictable and therefore fragment uploads toward them need strategies that attempt to complete the transfers as soon as possible. In this chapter we discuss the importance of scheduling decisions in a P2P backup system, and show how they impact TTB and TTR.

The differences between TTB and $minTTB$ (and between TTR and $minTTR$) are due to the following reasons:

1. because of the unavailability of remote peers, idle time might occur if there is no *online* remote peer to perform upload to or download from;

2. because of the unreliability of remote peers, storing data on them requires applying redundancy to the backup data: this results in a larger amount of data to transfer, hence longer TTB;

3. complete fragments must be uploaded to (resp. downloaded from) selected remote peers, so when the transfers are interrupted because of the temporary unavailability of remote peers, idle waiting may occur before resuming them to finish TTB (resp. TTR);

4. remote peer bandwidth capacities and competing parallel transfers may constrain transfer speed, possibly increasing both TTB and TTR.

In order to quantify the inefficiency of the backup and retrieval phases, we model the time loss due to the lack of online peers to exchange data with. First, we provide the scheduling problem formulation assuming full information about the future online phases of remote peers, with the goal of minimizing the completion time of the necessary fragment transfers. Second, we propose a *randomized* strategy to solve the scheduling problem, if future online phases are not known. We give an average-case analytic estimate on the achieved TTB (or TTR) with the random

scheduling to evaluate its performance on probabilistic inputs of online appearances where only the average availabilities of remote peers are known. Third, we quantify the performance of randomized strategy for realistic cases.

6.1 Scheduling problem with full information

We define fragment transfer decisions as a job scheduling problem.

Definition 9 ***Transfer scheduling problem*** *We call the transfer of a fragment (data amount f) between two given peers over the Internet a job. The job becomes available for "processing" after an online remote peer has been associated with it. If job J is an upload (resp. download) job of peer i, it requires a minimal processing time $t_p^J = \frac{f}{u_i}$ (resp. $t_p^J = \frac{f}{d_i}$). For any given schedule, the completion time of job J is denoted as t_c^J.*

During the backup and retrieval phases, a peer schedules jobs, i.e.*, upload and download transfers. Our objective is to find the schedule that minimizes the latest completion time among the jobs,* i.e.*, to find* $\arg\min_{\mathcal{J}}(\max_{J\in\mathcal{J}} t_c^J)$*, where $\mathcal{J}$ is a set of jobs that satisfy the goals of the backup or the retrieval phase.*

Here we focus on the fragment transfer inefficiencies *only* due to the unavailability of remote peers, and leave the analysis of data redundancy, fragment granularity issues and bandwidth constraints for Chapter 7.

Assumption 6.1.1 *We assume that user online appearances are* a priori *information, furthermore, link capacities and parallel transfers of remote peers do not cause any constraints.*

We model time as slotted, with nodes being online or offline for a whole time-slot, and having an integer upload (u) or download (d) capacity in terms of fragment transfers per time-slot.

We transform the scheduling problem in the following way: in order to find the schedule that minimizes the required time to transfer N fragments (called as *mintime* problem), we determine the maximal number of transferable fragments in T time-slots (denoted as $maxfrag$ problem). Then, the solution for the original problem is the smallest T within which N fragments can be transferred.

Definition 10 ***The*** *mintime* ***and*** $maxfrag$ ***problems*** *are defined as follows: s stands for any scheduling, $t(s)$ gives the duration, and $n(s)$ provides the transferred fragments of a given schedule,*

$$mintime(N) = \min\{T \mid \exists s \ : \ t(s) = T \wedge n(s) \geq N\}; \tag{6.1}$$

$$maxfrag(T) = \max\{N \mid \exists s \ : \ t(s) \leq T \wedge n(s) = N\}. \tag{6.2}$$

The two problems are intertwined in the following way:

Proposition 7 $mintime(N) = \min\{T \mid maxfrag(T) \geq N\}$.

Proof: Let us call $t_1 = mintime(N)$ and $t_2 = \min\{T \mid maxfrag(T) \geq N\}$.

1) $t_1 \geq t_2$. By (6.1), an s_1 exists such that $t(s_1) = t_1$ and $n(s_1) \geq N$, implying that $maxfrag(t_1) \geq N$. Therefore, $t_1 \geq \min\{T \mid maxfrag(T) \geq N\} = t_2$.

2) $t_1 \leq t_2$. By (6.2), s_2 exists s.t. $t(s_2) = t_2$ and $n(s_2) \geq N$. This directly implies that $t_1 = mintime(N) \leq t_2$. ■

We present the following Integer Linear Programming (ILP) problem formulation which is analogous to $maxfrag(T)$ of (6.2).

Definition 11 ***The scheduling problem with full knowledge*** *maximizes the number of transmitted fragments within a given duration T. x_i^t is a variable that encodes scheduling decisions: the number of fragments scheduled with remote peer i in time-slot t. The availability of remote peers is given as constraints: $a_i^t = 1$ if remote peer i is available in time-slot t, 0 otherwise. Another constraint is the maximal number of fragments m that can be placed on each remote peer. The solution of $maxfrag(T)$ can be found by solving the following ILP problem:*

$$\max \sum_{t=0}^{T} \sum_{i=1}^{I} x_i^t \qquad \text{maximize number of transmitted fragments}$$

$$\begin{aligned}
\text{s.t.} \quad & x_i^t = [0, \min(u, d_i)] && \text{transferable fragments in a time-slot to a peer} \\
& x_i^t \leq m a_i^t && \text{transfer to online remote peers} \\
& \textstyle\sum_{t=0}^{T} x_i^t \leq m && \text{no more than } m \text{ fragments on a remote peer} \\
& \textstyle\sum_{i=1}^{I} x_i^t \leq u && \text{no more than } u \text{ transfers within a time-slot.}
\end{aligned}$$

In order to translate the above backup scheduling problem to the retrieval case, we write d instead of u, and m is replaced by the vector of stored fragment number on each remote peer.

ILP problems are usually NP hard to solve. In order to make the solution of $maxfrag(T)$ tractable, we propose to transform the problem into a *maximum flow* formalization. The same underlying problem thus becomes solvable in polynomial time. Our proposed transformation might be useful for analogous job scheduling problems in other P2P settings.

We present the maximum flow formulation of the $maxfrag(T)$ problem in Figure 6.1. Nodes *ts* i with $i = 1, 2, \ldots, T$ represent the time slots up to T. *peer* i with $i = 1, 2, \ldots, I$ depict the remote peers. *ts* i is connected to *peer* j if and only if j is online in time-slot i. Edges with capacity u give the constraints on the uplink capacity of the peer, edges with capacity m describe the maximum number of fragments to be stored on each remote peer. The maximum flow from the *source* to the *target* yields the largest number of fragments that can be uploaded within time T. Similarly to the ILP formulation, the retrieval problem formulation is attained if u is replaced by d, and m is replaced by the stored fragment number on each remote peer respectively.

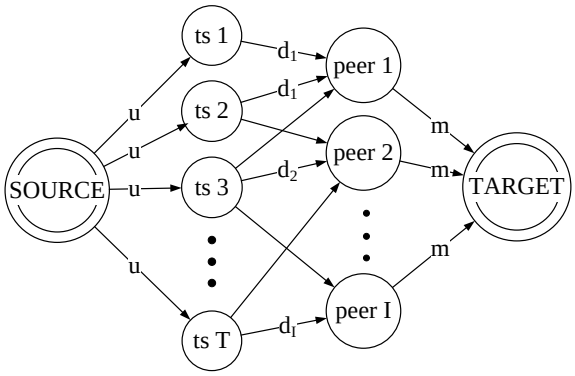


Figure 6.1: Maximum flow problem formulation of data transfer scheduling

We can now iteratively compute $maxfrag(T)$ with growing values of T; Proposition 7 guarantees that the first value T that satisfies $maxfrag(T) \geq N$ will be the desired result for our initial scheduling problem. The original problem, *i.e.*, finding an optimal schedule that minimizes the time to transfer N fragments, can be solved by performing $O(\log T)$ max-flow computations. In fact, an upper bound for the optimal completion time can be found in $O(\log T)$ instances of the max-flow algorithm by doubling at each time the value of T, then the optimal value can be obtained, again in $O(\log T)$ time, by using binary search.

For a flow network with V nodes and E edges, the maximum flow can be computed with time complexity $O\left(VE \log\left(\frac{V^2}{E}\right)\right)$ [63]. In our case, when we have I nodes and an optimal solution of T time-slots, V is $O(I+T)$ and E is $O(IT)$. The complexity of an instance of the algorithm is thus $O\left(IT\left(I \log \frac{I}{T} + T \log \frac{T}{I}\right)\right)$. The computational complexity of determining an optimal schedule in a full information framework is thus $O\left(IT \log T\left(I \log \frac{I}{T} + T \log \frac{T}{I}\right)\right)$.

6.2 Random scheduling without full information

Constructing the optimal scheduling of transfers with remote peers, *i.e.*, ensuring the earliest possible completion time for the required jobs, is impossible without knowing their future online phases. Here, we evaluate the performance of random scheduling and give an upper bound on its efficiency.

Assumption 6.2.1 *We assume that the online appearances of remote peers are independent stochastic variables in every time-slot, and their autocorrelation is* 0*. We suppose that every peer is online in each time-slot with equal probability a: let* $A(i,t)$ *indicate the event when peer* i *is available at time* t*, we define* $\mathbb{P}[A(i,t)] = a \ \forall i,t$ *(i.i.d.).*

We derive the following expected difference between TTB and $minTTB$ based on Assumption 6.2.1 if the limit m of fragments that can be stored on each peer takes the extreme values of 1 or

n. This latter case covers a system where it is possible to store a full copy of each backup object on a server. In both cases we assume that at most one fragment is transferred in each time-slot.

Proposition 8 *If $m = 1$, the estimated lateness of $\min_{\mathcal{J}}(\max_{J\in\mathcal{J}} t_c^J)$ is given by:*

$$\frac{\mathbb{E}(TTB)}{minTTB} \leq \frac{1}{1-(1-a)^{I-n+1}}.$$

Proof: The probability of finding at least 1 online peer to upload the first fragment in a time-slot is $1-(1-a)^I$. Thus, the expected value of the geometrically distributed random variable $\mathcal{X}_1$, which describes the number of Bernoulli trials needed to get one success, is $\mathbb{E}(\mathcal{X}_1) = \frac{1}{1-(1-a)^I}$. Similarly, $\mathbb{E}(\mathcal{X}_2) = \frac{1}{1-(1-a)^{I-1}}$ for the following fragment, and so forth. The expected duration of uploading all fragments is $\mathbb{E}(\mathcal{X}_1+\mathcal{X}_2+\cdots+\mathcal{X}_n) = \mathbb{E}(\mathcal{X}_1)+\mathbb{E}(\mathcal{X}_2)+\cdots+\mathbb{E}(\mathcal{X}_n) = \sum_{i=0}^{n-1}\frac{1}{1-(1-a)^{I-i}} \leq \frac{n}{1-(1-a)^{I-n+1}}$. ■

Proposition 9 *If $m = n$, the probability that less than n fragments are uploaded after T time-slots is lower than* $\exp\left(-\frac{\left(T\left(1-(1-a)^I\right)-n+1\right)^2}{2(1-(1-a)^I)T}\right)$.

Proof: We consider the case where we can upload any number of fragments per peer, and each fragment gets uploaded exactly once. Let $U(t)$ denote the event that a fragment can be uploaded at time t since some peers are online. Then,

$$u = \mathbb{P}[U(t)] = 1-(1-a)^I.$$

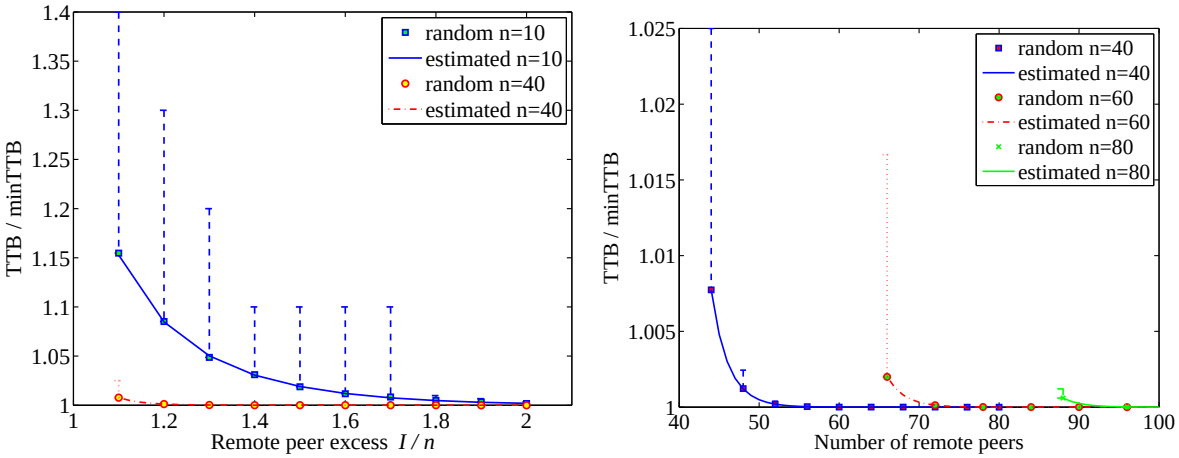
The probability of *not* being able to upload at least n fragments in time-slots from 1 to t is:

$$\mathbb{P}\left[\sum_{i=1}^{t} U(t) \leq n-1\right] = \sum_{i=1}^{n-1}\binom{t}{i}u^i(1-u)^{t-i} \leq \exp\left(-\frac{(tu-n+1)^2}{2tu}\right),$$

by using its Chernoff bound, since $\mathbb{P}[X \leq (1-c)\mathbb{E}X] \leq \exp\left(-\frac{c^2}{2}tu\right)$ and in our case $c = 1-\frac{n-1}{tu}$.

Being a Bernoulli process, the expected number of time slots T to transfer n fragments is $\mathbb{E}(T) = \frac{n}{1-u}$, since the average of the related binomial distribution is $t(1-u)$. ■

Propositions 8 and 9 characterize the theoretical inefficiency of the backup phase due to the unavailability of remote peers. The results show that, in general, as the number of remote peers (I) grows, the larger set of potential targets for scheduling decisions eliminate the probability of being halted due to their unavailabilities. Intuitively, an increasing number of fragments (n) produces counter effects, particularly if there is a strict constraint on the maximal number of fragments that can be scheduled to any given remote peer.



(a) Relative number of peers

(b) Absolute number of peers

Figure 6.2: Backup and retrieval inefficiency with synthetic online phases yielding 0.36 average availability

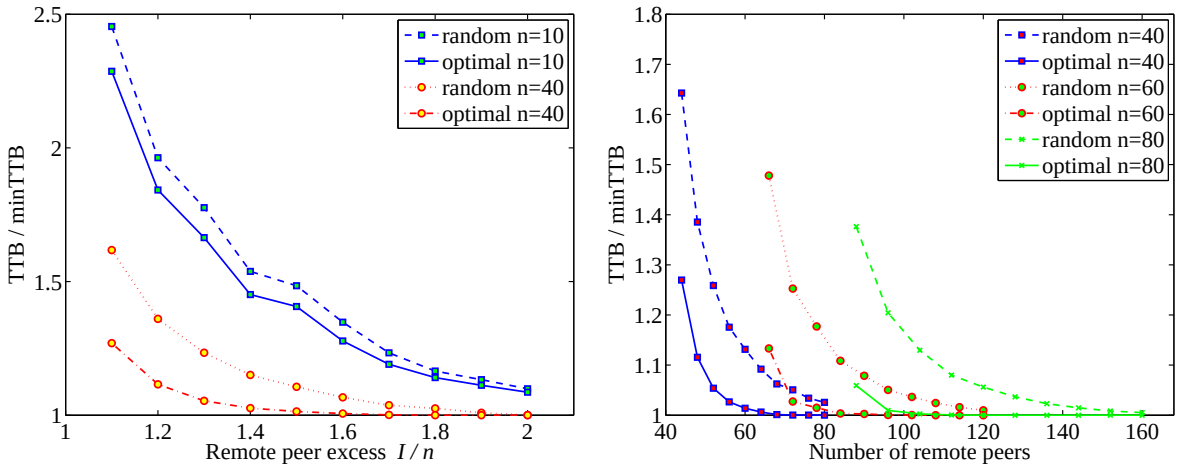
6.3 Evaluation of random scheduling

The unpredictable unavailability of remote peers hinders the minimization of the backup and retrieval phase durations. In this section we evaluate the performance of scheduling fragment transfers toward remote peers randomly, by comparing the results to the optimal outcome. As we show above, if the system design determines a fragment size, well-suited to the number of remote peers, and a sufficiently high redundancy rate, then the performance of random peer selection, *i.e.*, the duration of the backup/retrieval phases, approaches the optimal.

We implement discrete-time scheduling simulations of the above presented problems in MATLAB. Peer online-offline patterns are generated by independent and identically distributed random variables, with the average online availability of our traces (Figure 7.2(a)) as probability. The traces capture user availability, in terms of login/logoff events, from an instant messaging server in Italy for a duration of 3 months [38].

We set the number of fragments to upload to $n = 10, 40, 60, 80$ and the number of remote peers to $I = 1.1n, \ldots, 2n$. In order to mimic our system simulation settings, we let $u = 1$ and $m = 1$, *i.e.*, at most one fragment can be uploaded in a time-slot, and maximum one fragment can be stored on each remote peer, respectively. We run simulations for 1000 generated online pattern inputs, moreover the random scheduling scenario is performed 1000 times on each input. We compare the statistic results of the random scheduling policy to the analytic performance shown in Proposition 8.

On both Figure 6.2(a) and 6.2(b), we plot the median and $95th$ percentile TTB prolongation, *i.e.*, TTB/$minTTB$, of the random scheduling results among the 1000×1000 simulations for each $I - n$ scenario with interconnected markers. The estimated values, based on Proposition



(a) Relative number of peers

(b) Absolute number of peers

Figure 6.3: Backup and retrieval inefficiency with correlated online phases yielding 0.36 average availability

8, are presented by the solid lines for each fragment number case. In Figure 6.2(b) we plot the absolute number of peers on the horizontal axis instead of their relative number compared to the number of fragments in Figure 6.2.

We observe that the analytic estimation overlaps with the median of the results. Furthermore, a relatively large remote peer set (larger than 40 with the given average peer availability) ensures that the performance of the random scheduling policy is close to the optimal, even if the number of fragments to transfer is only slightly less than the peer set size.

Note that the backup and retrieval phases are analogous. We present the simulation results for the backup phase, but substituting k instead of n, and n instead of I, we arrive at the retrieval problem, *i.e.*, downloading k fragments from n sources. Based on the results, applying random scheduling to retrieve fragments from remote peers would yield a TTR close to the $minTTR$.

Our traces exhibit both heterogeneity and correlated user behavior (presented in Figures 7.1(a) and 7.1(b)), reflecting human habits, *e.g.*, most of the users are online during working hours of weekdays, but offline at night. Interested in the effect of this phenomenon on random scheduling, we also evaluate its performance with our correlated traces. We compare the results of the random scheduling policy to the optimal solutions with the same parameters as before. For each experiment pair (optimal and random) of $I - n$, the required number of peer traces n are randomly drawn from our traces, starting at a randomly selected point in time. We set the time slot length to 1 hour; a peer is considered to be online, if it is online during at least half of the time slot, based on its trace.

We repeat the simulation of each $I - n$ case 1000 times for both optimal and random scheduling, moreover the random scheduling scenario is performed 1000 times on each input. The results

are plotted in Figure 6.3, in a similar way as in Figure 6.2, but instead of the estimated values, we plot the median of the 1000 optimal solutions for each $I-n$ case, and only the median of the random scheduling results.

We observe weaker performance of random scheduling when the online-offline sessions of remote peers are provided by the real traces. This is due to the fact that their correlated behavior increases the number of time slots during which only a few, or even none of the peers are online. However, as the number of fragments to transfer grows, the optimal solution gets closer to the theoretic lower bound, *i.e.*, $minTTB$. Moreover, with larger remote peer set, the performance of random scheduling approaches to that of the optimal, irrespective to the number of fragments to transfer.

Therefore, in our analysis we obtain that, as heuristic thresholds, the values of $n = 60$ and $I = 90$ are sufficient to complete backup within a tolerable (around 10%) deviation from $minTTB$ with random scheduling. In our system simulations, described in the following chapter, we apply a value for k according to these thresholds, and therefore we ensure that randomized scheduling decisions impose only a reasonable penalty on TTB and TTR in our results.

Chapter 7

Evaluation of the system design

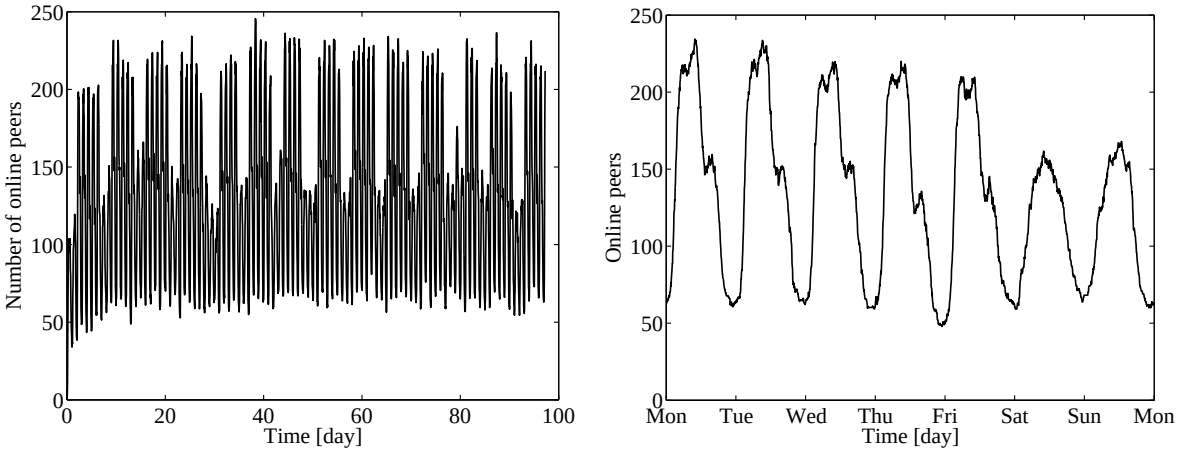
Determining analytically if the quality of service goals are achieved with our system design is hard. In order to numerically analyze how our choices affect performance, we built a discrete-time MATLAB simulator. We use our simulator to experimentally evaluate the effects of applying different policies for data redundancy, peer selection and transfer scheduling. Due to the randomized nature of our algorithms, the results presented in the following are averaged over 10 simulation runs.

7.1 Simulated user settings

Our simulations are synchronous and organized into rounds, where 1 round represents 5 minutes. The total number of rounds is 28000, resulting in more than 3 months of simulated system lifetime.

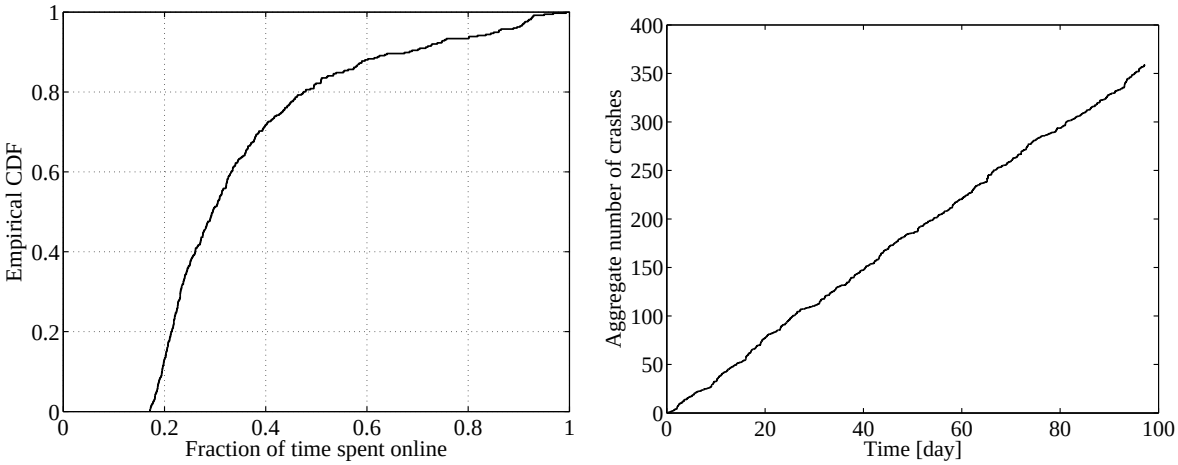
We model the online behavior of a peer with three states: online, offline and crashed. The session lengths of all these states are represented in rounds. The *online behavior* of users, *i.e.*, their patterns of connection and disconnection over time, is difficult to capture analytically. We perform our evaluations on the real application traces, introduced in Chapter 6. We argue that the behavior of regular instant messaging users constitutes a representative case study: for both an instant messaging and an online backup application, users are generally signed in for as long as their machine is connected to the Internet.

The set of simulated users taking part in the system does not vary in time. We only consider users that are online for an average of at least four hours per day (calculated for the whole duration of the measurement), as done in the Wuala online storage application [7]. Once this filter is applied, we obtain the trace of $I = 376$ users. User availabilities are strongly correlated, in the sense that many users connect or disconnect around the same time. As shown in Figures 7.1(a) and 7.1(b), there are strong differences between the number of users connected during day and night and between workdays and weekends. Most users are online for less than 40% of



(a) weeks (b) days

Figure 7.1: Number of online peers



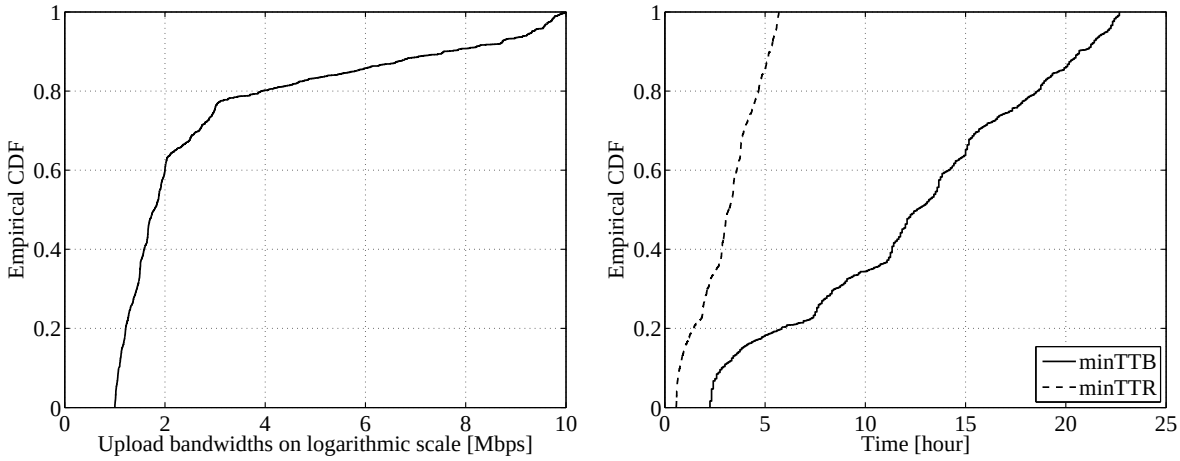
(a) Average peer availabilities (b) Number of peer crashes

Figure 7.2: Peer behavior inputs

the trace, while some of them are almost always connected (Figure 7.2(a)). Some peers appear online for the first time late in the trace.

Peer crash events, *e.g.*, disk failures, are randomly generated from an exponential distribution of lifetimes with the simulation duration as expected value. This leads to a uniform rate of crashes, as represented in Figure 7.2(b). We assume that peers remain online during their retrieval phases, irrespective of their traces.

In our simulations, each peer has a 10GB backup object to save, and it has 50GB available for storing data of other nodes. As mentioned in Chapter 4, every user holds its own backup object locally until it crashes. This setting ensures that the storage capacity of remote peers rarely constrains fragment uploads.



(a) Peer uplink capacities

(b) $minTTR$ and $minTTB$

Figure 7.3: Peer connectivity inputs

We model peer link capacities based on another set of traces: uplink capacity values, between 1Mbps and 10Mbps are randomly drawn from the measurement collection of [110] (Figure 7.3(a)), and we consider downlink capacities to be 4 times larger than uplinks. To avoid bandwidth underutilization due to asymmetric bandwidth characteristics, maximum parallel uploads and downloads for peer i are $p_i^u = 1, p_i^d = 4$ by default, proportionally increasing with $\frac{u_i}{\tilde{u}}$ and $\frac{d_i}{\tilde{d}}$ respectively, where $\tilde{u}, \tilde{d}$ are the median upload and download bandwidths in the system.

In Figure 7.3(b) we show the distribution of the $minTTB$ and $minTTR$, defined in Definition 2: the amount of each user's backup data divided by its *effective* upload and download capacity respectively. During the backup phase each peer is assumed to follow its usual online-offline status variation, and the amount of backup data is generally much larger than what is transferable within a few online phases of a peer. Therefore, the effective upload capacity is approximated by au. On the contrary, the peer remains online in the retrieval phase, thus its effective download capacity is d. The difference between TTR and TTB values, visible in Figure 7.3(b), is due to the asymmetry in download and upload capacities.

Our choices of experimental settings do not impose any kind of correlation or variability during time between peer crash rate, online behavior, bandwidth and storage capacities, and the backup data amount. We think that characterizing the correlations and evolution of these quantities over time is an interesting open problem.

Furthermore, we suppose that the data center is over-provisioned, *i.e.*, it possesses unlimited storage capacity and bandwidth, furthermore it is always available and never crashes.

7.2 Fixed and adaptive redundancy rates

In this section we evaluate our proposed data redundancy scheme by comparing its performance to that of a widely used policy which imposes fixed rates of redundancy for every user. We apply $k = 64$ as the number of original fragments per backup object, and since we organize $10GB$ backup data into 1 backup object, fragments of size $f = 160MB$ are produced.

7.2.1 Prompt data availability and TTR

In many related works on P2P storage systems, the redundancy rate for erasure coding is chosen in order to provide high, system-wide instant data availability, resulting in that every stored file is retrievable, supposing infinite bandwidth links, at any moment in time with high probability. For this, the redundancy scheme aims to ensure that at least k peers that store the fragments of any backup object are online.

This redundancy scheme, what we call *fixed-rate* policy, is presented in [17] for an online P2P storage system, targeting very high data *availability*. The authors model the system as if the encoded fragments were stored on different remote peers with independent online appearances having homogeneous availability. In this setting, data availability can be expressed as:

$$1 - \epsilon = \sum_{i=k}^{n} \binom{n}{i} a^i (1-a)^{n-i}, \tag{7.1}$$

where a is the theoretical availability of peers that compose the system, and ϵ is the probability that the file is unavailable.

The expression to compute the redundancy factor, given a target file availability, an average online time, and the number of original fragments k writes as:

$$r = \left(\frac{\sigma\sqrt{\frac{a(1-a)}{k}} + \sqrt{\frac{\sigma^2 a(1-a)}{k} + 4a}}{2a} \right)^2 \tag{7.2}$$

where σ is the standard deviation of a normal distribution for the required level of file availability.

We set our data availability goal to 0.99. As value of a, we use the system average of peer availabilities, which is 0.36 in our traces. Applying (7.2) of [17], we obtain a redundancy rate of 3.56. As consequence, every peer creates the same data redundancy to its backup, resulting in $n = 228$ encoded fragments.

We plot the average online redundancy (actually available fragments over k) per backup object for every round of the simulation in Figure 7.4. In a setting with independent peers, the applied redundancy rate would ensure permanent data availability ($> 100\%$) most of the time. Since the online behavior of peers is correlated, the measured data availability drops during certain periods in our system, *e.g.*, data is not fully availabile during night.

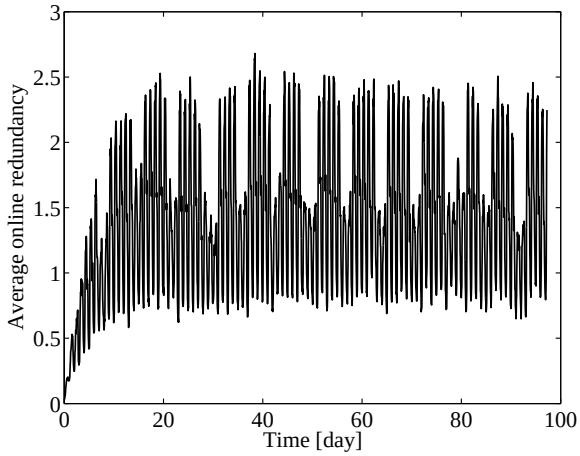


Figure 7.4: Online redundancy with fixed-rate

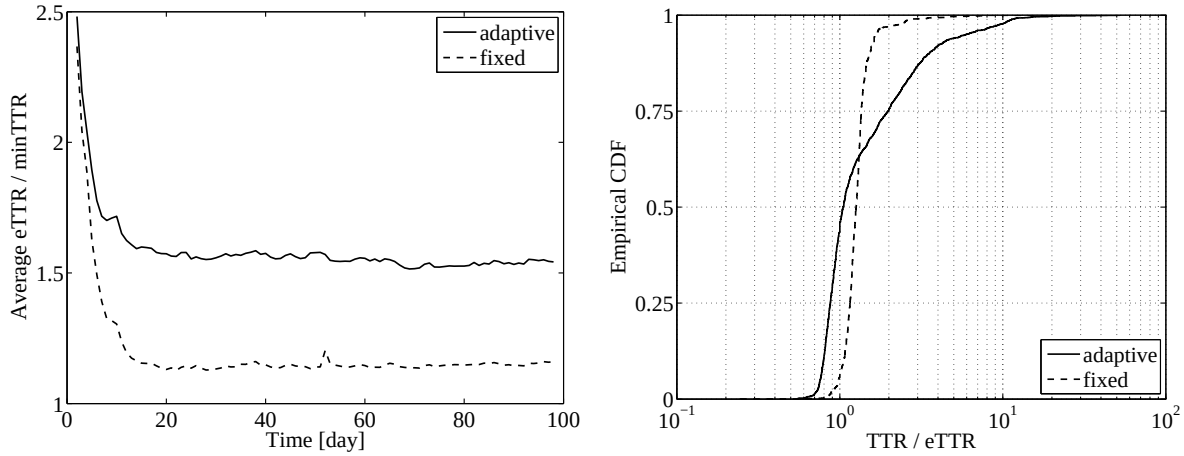
Figure 7.5(a) shows the evolution of the average eTTR when applying the fixed-rate and our adaptive-rate redundancy schemes in a purely P2P setting with random peer selection. As time goes by, more and more fragments are uploaded by every peer, therefore in case of retrieval the choice of download sources widens, possibly containing fast remote peers, hence decreasing estimated TTR. On the long run the average eTTR of adaptive-rates are approximately 30% higher compared to those of fixed-rates. This is due to the fact that our adaptive scheme lowers the redundancy rate, thus worsens the efficiency of retrievals.

In Figure 7.5(b) we evaluate the accuracy of eTTR with the same settings. Plotted eTTR values are calculated when crashes occur, and compared to the subsequently measured TTR. The eTTR gives a fairly good estimate on the TTR: in a few cases it is too pessimistic (ratio of TTR over eTTR is around 0.6), and it proves to be too optimistic with a factor of more than 2 in 25% of the cases.

The eTTR is admittedly an optimistic estimation heuristic: if at least one of the fastest k peers crashes during retrieval and/or most of remote peers are offline when the crash happens, the measured TTR is probably longer than the eTTR. High bandwidth peers, therefore holding low estimates of TTR and low applied redundancy rates further increase this effect, especially in our correlated online behavior setting where a retrieval starting at Friday night might be lengthened due to the lack of online peers until Monday morning. However, as the redundancy rate increases (fixed-rate), the eTTR gets closer to the measured TTR in most cases.

7.2.2 Adaptive redundancy rate scheme

We argued that prompt data availability, *i.e.*, being able to restore all data *instantly* assuming infinite bandwidth, is an excessive goal in a backup system where large amount of data is retrieved in case of need. With our adaptive redundancy scheme (introduced in Section 4.2), we instead



(a) Average eTTR evolution

(b) Measured vs. estimated TTR

Figure 7.5: Analysis of adaptive-, and fixed-rate redundancy schemes

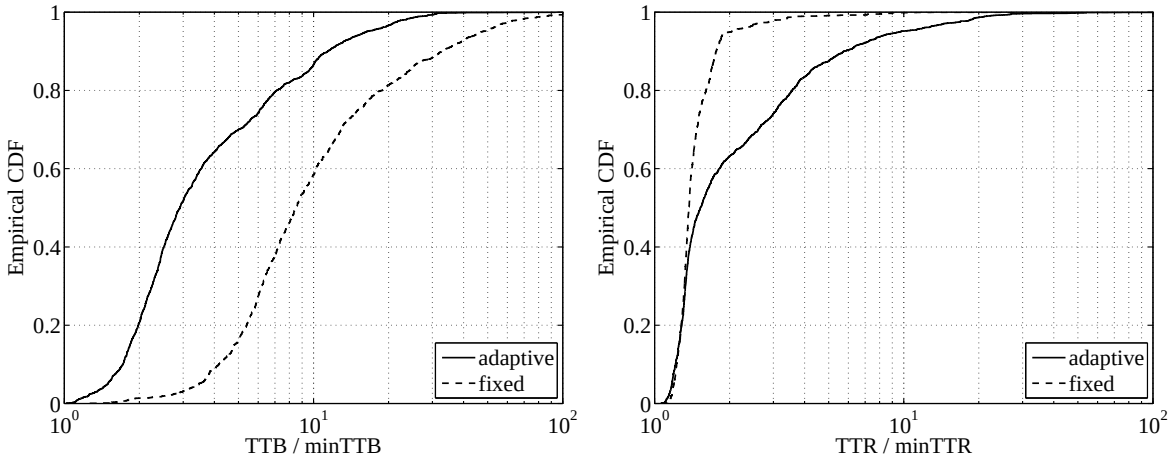
target to guarantee certain levels of data durability and TTR.

Our redundancy scheme requires threshold values for both TTR and DLP, we adopt the following in our system simulations.

- The eTTR target is $\frac{o}{d_i a_i}$ in order to make retrieval time targets proportional to the characteristics of the peer, and also to those of the potential storage partners (due to the peer stratification, described in Chapter 5).

- The eDLP target is 10^{-4}, calculated for the time duration along which the peer does not have valid information about the status of its fragment stored on a remote peer *plus* the eTTR. The former period has the double length of the predetermined time period after which peers are notified by the tracker about the crash of a peer, having been offline for the time being, denoted as w in Section 4.2. We set w to one week in our simulations, therefore the system might not have valid information about the locally and remotely stored fragments of a peer during two weeks, *e.g.*, in case the peer crashes and stays offline for a week after having lost fragments on some remote peers that had been offline for a week prior to the crash. The assisted repairs that we evaluate in the following, motivate the estimation of DLP for $2 \times w + eTTR$ duration.

Simulation results of fixed-rate and our adaptive-rate redundancy schemes are shown in Figure 7.6. We set the fixed-rate redundancy as upper limit on the adaptive-rates. In order to visualize the results, we average the redundancy rates of peers after their backup phases (plotted in Figure 7.7(a)). The significantly lower adaptive redundancy rates result in decreased TTB values, in return of prolongated TTR results. The tail of the distribution in Figure 7.7(a) represents those peers that store their fragments on storage peers with limited bandwidth capacities and



(a) TTB (b) TTR

Figure 7.6: Fixed and adaptive redundancy schemes

online availability, therefore being forced to apply higher redundancy rate then most of the participants in order to obtain the required TTR.

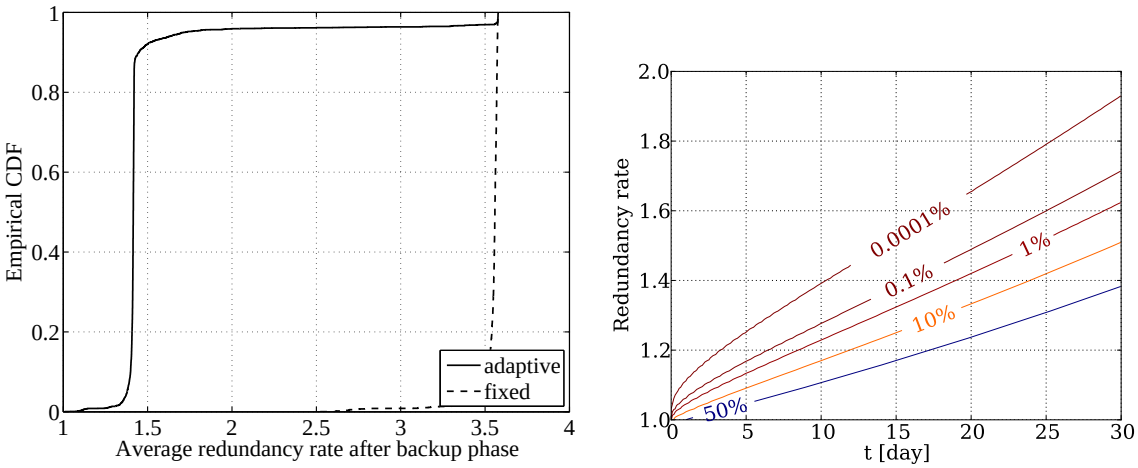
We argue that it is reasonable to trade longer TTR for shorter TTB due to the following reasons. When a user registers to the system, it inevitably suffers from a possibly long TTB, while the length of TTR matters *only* to those who lose their local copy. Furthermore, the crashed peers are probably more willing to wait to complete retrievals, while when backing up, users are more likely to be lazy.

The decay of TTR when switching to adaptive-rates seems less significant than the savings in terms of TTB. This is because the chance to find storing remote peers with large upload capacity is only reasonably lower during the retrieval phases. The observed inefficiency of TTR in both cases is due to data retrievals started at night: crashed peers have to wait until the morning when most of their peers come online.

7.2.3 Data loss results

In Figure 7.7(b), we plot the estimated DLP of Equation (4.2) as a function of the redundancy rate and the delay t. We set the average peer lifetime $\bar{t} = 90$ days, as defined in Section 4.2. When the time without maintenance is in the order of magnitude of weeks, even a slightly lower redundancy rate can increase eDLP by several orders of magnitude.

In order to show the impacts of delayed data maintenance with our lowered adaptive-rates, peers spend a randomly generated length of time (1 week on average) offline after they crash. The time period given to offline peers before starting repairs (denoted by w in Section 4.2) is also set to 1 week, as mentioned above. The tracker starts to perform assisted repairs if its estimated DLP during $2 \times w$ (as explained above) *plus* the estimated retrieval duration of he



(a) Redundancy rate distribution

(b) Estimated DLP

Figure 7.7: Redundancy rates and data losses

tracker increases beyond 10^{-4}.

We categorize data losses in the following way: if the crashed peer

1. has not spent enough time online to upload k fragments until its crash, no online backup system could have saved the data, because data loss is determined by the limits in resources of the data owner;

2. has spent enough time online to upload at least k fragments until its crash, but it has not succeeded: in this case, the limited resources of remote peers are the cause of data loss, since backing up on an always-on data center with plenty of bandwidth would have succeeded;

3. has uploaded at least k fragments, but its backup phase has never completed, then the loss is due to the fact that remote peers crash during the backup phase;

4. has completed its backup phase, but it fails to retrieve at least k fragments after the crash, before its storing peers crash in a fatal number.

We summarize the data loss types in Table 7.1, and we make reference to it in further discussions.

Figure 7.8 represents the number of the observed data losses, labeled based on the above categories. The first row of plots are the results when using our adaptive redundancy scheme. The plots in the second row are the outcomes of the fixed-rate scheme defined in in the beginning of this section. The first column represents the case where the crashed peer appears online right after its crash, and remote peers are notified without delay; the second column shows the delayed maintenance scenario; the third column stands for the simulations where maintenance is delayed, but assisted by the tracker.

Type	Cause of data loss
1	unavoidable
2	inefficient upload
3	elongated TTB
4	elongated TTR

Table 7.1: Data loss types

In all cases the majority of data loss episodes are unavoidable. The consequence is that online backup can only be safe up to a certain unavoidable limit, represented by the time a peer needs to upload its data. All other shortcomings, introduced by the unreliable remote peers, have merely marginal effects. It is our simulated intensity of peer crashes that leads to high absolute number of data loss events: in all of our experiments, due to the inflated peer crash rates, at least 6% of crashed peers could not recover their data.

The majority of data loss events affected peers that crashed before they could upload their data without any redundancy: the backup process is inherently time-consuming, due to the availability and bandwidth of data owners. Thus, users should worry more about completing their backup quickly than about the reliability of their peers. Even after peers upload k fragments, most of the data loss cases are due to low reached data redundancy. In the remaining cases long post-crash offline state, low availability, or poor bandwidth capacity lengthens retrievals, leading to data loss.

When peers go online right after they crash, the number of avoidable losses due to prolongated TTB is negligible in both redundancy schemes. When the data maintenance is delayed, data losses grow significantly with adaptive redundancy rates: peers that remain offline for a long post-crash duration lose their redundancy stored on remote peers. We offer a remedy for this phenomenon with the assisted repairs, with which no data loss occurs once the backup is considered complete.

The difference in redundancy between the high rate used by the fixed baseline and the adaptive approach does not impact significantly the data loss rate, except for the case of non-assisted delayed retrievals. In that case the adaptive-rate seems to provide less safety to the backups.

In Figure 7.9 we show the benefits and the related traffic burden of assisted repairs.

As shown in the third column of Figure 7.8, the number of data losses is lowered back to the case of prompt retrieves and repairs, in return for upload traffic (and intermittent storage load) of the data center, plotted in Figure 7.9(a). We relate the amount of data traffic to the overall size of backups (for all users, without redundancy); we will use the same mapping for data center storage in the following.

The role of the data center is more important when low adaptive-rates are applied, hence it performs more intensive outbound traffic which generates higher cost. On the other hand,

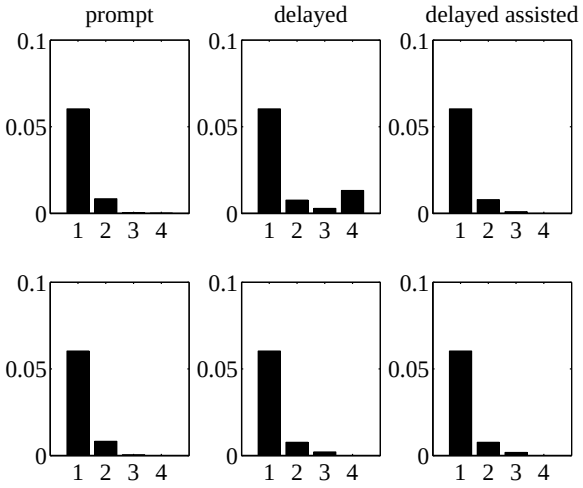


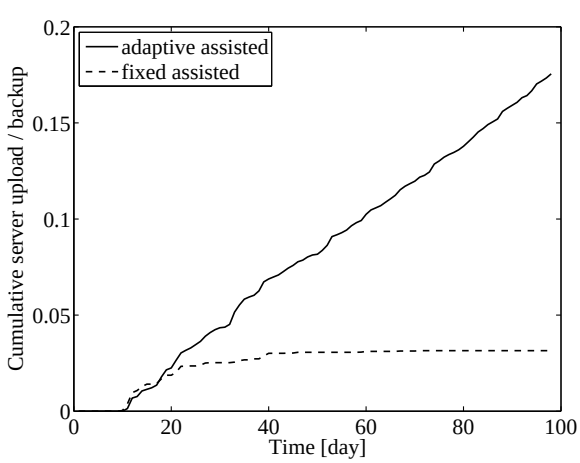
Figure 7.8: Fatal fraction of peer crashes with adaptive-rates (top) and fixed-rates (bottom), loss events are classed according to Table 7.1

the crash-related repair traffic, performed by peers, is proportional to the overall redundant data amount. Therefore it is significantly lower when applying adaptive-rates (Figure 7.9(b)), constituting an advantage of the adaptive scheme. Furthermore, when assistance is in place, TTR values with adaptive-rates are slightly lower (Figure 7.9(c)), since the crashed peers might retrieve their fragments from the data center, if it is holding them to perform the repair, characterized by high availability and download capacity.

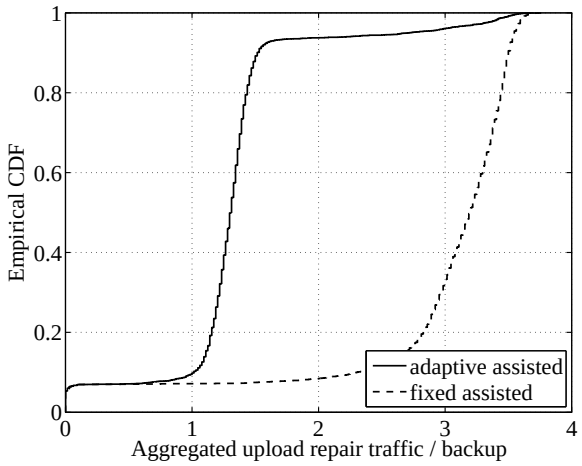
7.3 Evaluation of a grouped P2P system

Motivated by the analytical results, we show the performance of a system where peers, with fixed grades, are classified into predetermined grade classes. In the following system simulations we distinguish two grade classes. With this low number of groups, the classes contain sufficient peers for exchanges, adapted to our setting of $k = 64$, hence no need for grade improvements (defined in Chapter 5). We calculate the value of $a_i u_i$ (a_i: average online availability, u_i: uplink capacity), for each user i, and we classify equal number of peers based on these sorted values in the two classes. The distribution of grades in the system and in the separate classes are plotted in Figure 7.10.

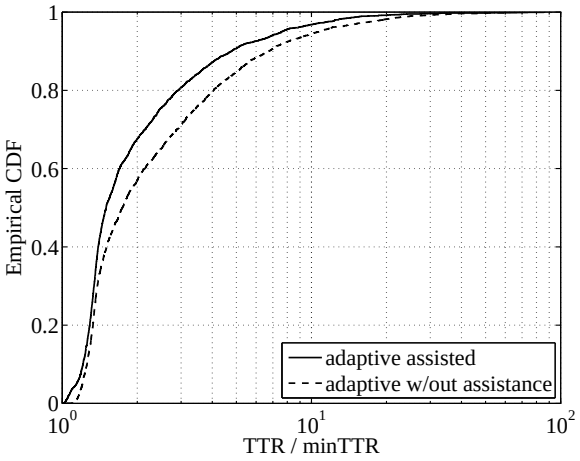
Figure 7.11(a) presents the storage load on peers in a system with adaptive-rate redundancy scheme, assisted repairs and random peer selection. The time-average of stored fragments are characterized by their distributions within the two classes of peers. A direct consequence of high availability and good connectivity, when peer selection is random, leads to excessive burdens on those peers that are found online more often: more fragments are uploaded to them successfully than to peers with lower availability.



(a) Repair traffic of the data center



(b) Repair traffic of peers



(c) TTR

Figure 7.9: The effects of assisted repairs

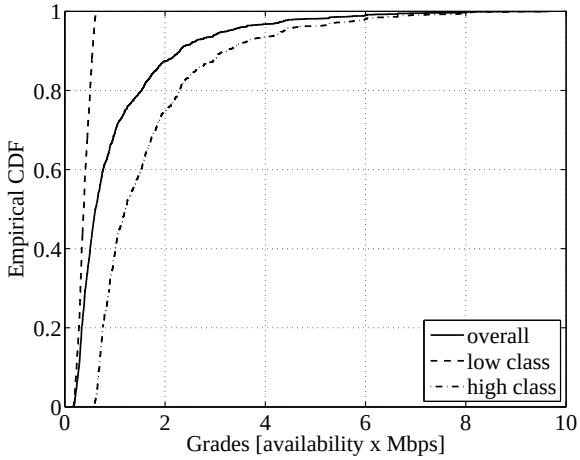
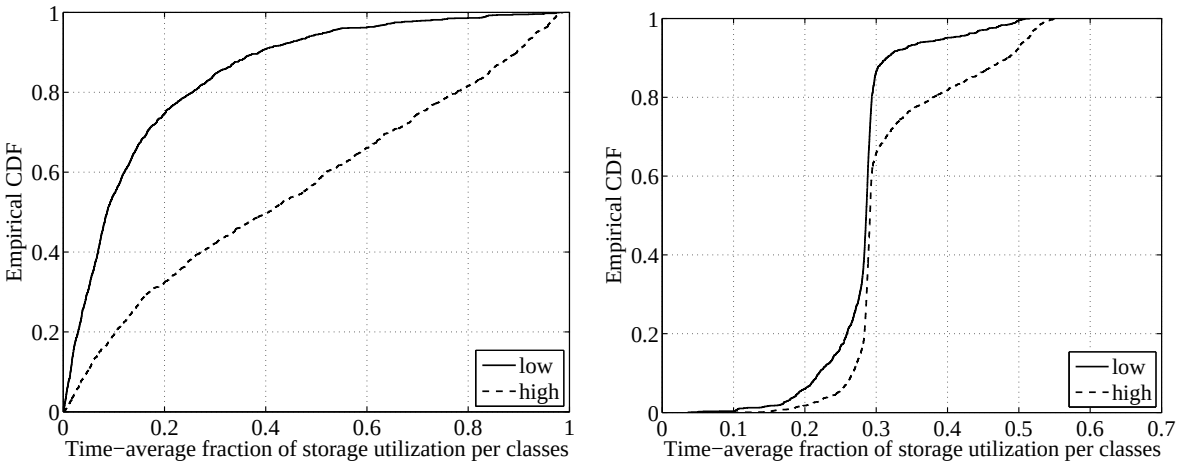


Figure 7.10: Distribution of grades



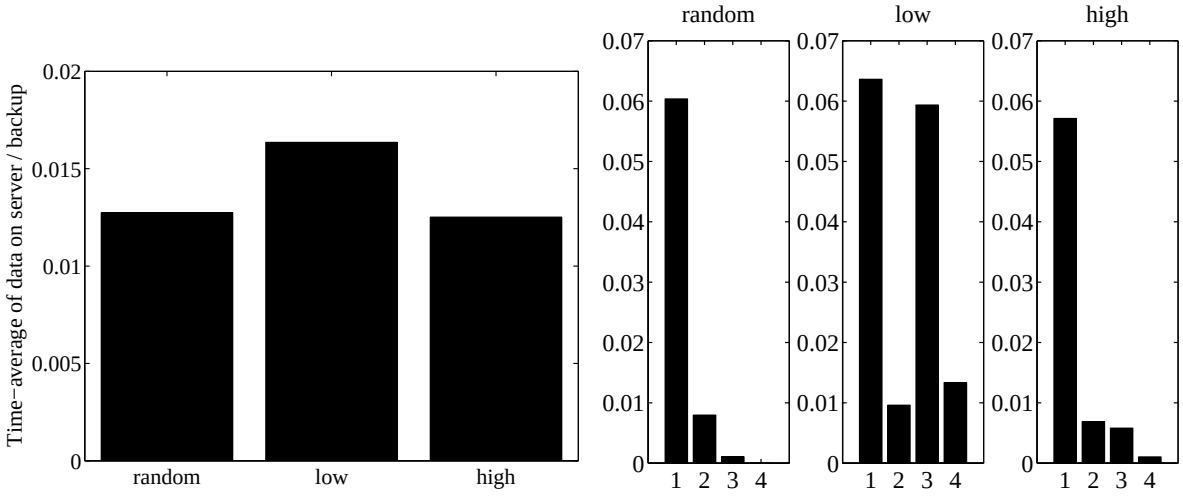
(a) Storage load on peers (random)

(b) Storage load on peers (grouped)

Figure 7.11: Fairness in grouped peer selection

Motivated by this unfairness in a system with heterogeneous peers, we suggested grouped peer selection. We show the storage load on peers and on the data center per classes in Figure 7.11(b) and 7.12(a) respectively. Opposed to the random peer selection outcome, the load on peers becomes balanced among the peers of the two groups with this scheme. On the other hand, the transient data center load (therefore also its upload traffic) is increased: this is due to the restrictions on repairs in symmetric scheme, mentioned in Section 4.4. Moreover, the data center keeps storing fragments if the class size and/or the storage capacity of peers hinder reaching complete backup on remote peers alone. We call the difference between data center load values of the random and grouped peer selection schemes the *price of fairness*: when peers are constrained to store backup fragments on remote peers with similar characteristics and are compelled to offer an amount of local storage space proportional to the amount of (redundant) data they inject in the system, the excess capacity provided by highly available peers cannot be exploited, therefore in a P2P application, some peers can suffer a severe loss in performance or eventually cannot complete their backups.

Figure 7.12(b) characterizes the data losses in systems with symmetric fragment exchanges within groups, due to the same reasons as in Figure 7.8. We experience a significant change in the number of data loss events by switching from random to the grouped peer selection scheme, especially due to long backup phases in the low grade class. While more data losses happen in both groups than in a system applying random peer selection, the high grade group members suffer from a significantly lower number of data losses than low grade peers. Indeed, the fairness introduced by the grouped peer selection does not affect every user in the same extent: fast fragment transfers between the data owner and storer, both high grade peers, results in short backup phases, hence lower probability for fatal crashes than among low grade users.



(a) Data center load per class

(b) Fatal fraction of peer crashes per class, loss events are classed according to Table 7.1

Figure 7.12: Cost of fairness in grouped peer selection

7.4 Evaluation of scheduling policies

In this section we first focus on the effects of discrete scheduling decision making, then we analyze the efficiency improvement yielded by assisted backups, a system design that leverages on a central storage facility to lower the inefficiency of the backup phase uploads.

7.4.1 Effects of discrete scheduling

In our system simulations, each peer assigns the set of remote peers at the beginning of each round that it will transfer data to/from within the round. The data amount to be transferred between peers is determined by the minimal throughput of the bandwidth characteristics of the two parties. The transfer schedule decisions, made every 5 minutes, homogeneously share the link capacity among the connections (limited by the maximal number of parallel up/downloads with remote peers).

The simulation does not allow for additional transfers if an initiated one is finished within a round. This discrete scheduling leads to bandwidth loss if the peer completes at least one fragment transfer and it still has unfinished ones not being active in the given round, although the paired remote peers are online.

In order to ensure that the lack of bandwidth re-allocation within a single round after the end of a fragment transfer does not introduce distortions, we closely observe the time *and* system average of unused bandwidth fraction of peers throughout the rounds. The inefficiency of uploads in a system applying adaptive-rate redundancy scheme, assisted repairs and grouped peer selection, plotted in Figure 7.13, appears negligible.

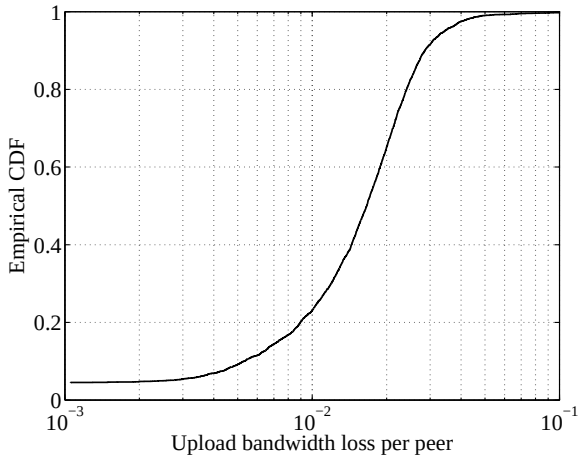


Figure 7.13: Uplink underutilization

7.4.2 Results of assisted backup

As we propose in Section 4.4, when transfers to remote peers are interrupted due to *e.g.*, their unavailability, peers may upload fragments to a data center during their backup phases. We call this system design feature as assisted backups: the data center assists the peers in building their backups.

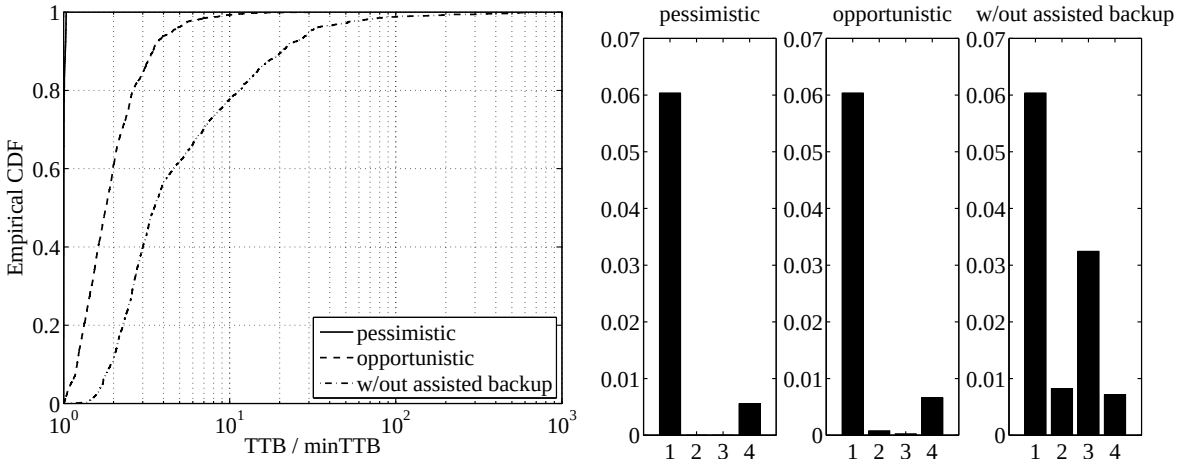
We perform numerical simulations in order to quantify the performance improvement and the related costs of assisted backups in our system. We evaluate two data placement strategies: our scheme (labeled as "opportunistic") that performs uploads to the data center only with excess upload bandwidth, and an alternative that gives priority to uploads toward the data center and backing up to remote peers is started only after having uploaded all original fragments to the data center (thus called "pessimistic").

The pessimistic data placement policy ensures that the TTB of every peer is effectively decreased to its $minTTB$. In this "safer" scheme the backup data is first entirely uploaded to data center to build reliable backup as soon as possible, then the centrally stored data is continuously deleted to save on the storage costs as the backup amount, successfully transferred to remote peers, is growing.

We compare TTB, data loss (Figure 7.14) and TTR (Figure 7.15(a)) results in three different schemes, *i.e.*, pessimistic, opportunistic assisted backup systems, and one without assistance.

Assisted backups mitigate the negative effects of long data transfers toward peers with poor availability and connectivity. While uploading only to remote peers may be slow due their unavailability, in assisted schemes uploading to data center is only constrained by the availability and uplink capacity of the peer, therefore the DLP and TTR targets are reached earlier, hence smaller TTB values.

Due to much longer TTBs, the data loss results (categorized against the cause of the losses,



(a) TTB

(b) Fatal fraction of peer crashes, loss events are classed according to Table 7.1

Figure 7.14: Benefits of assisted backup

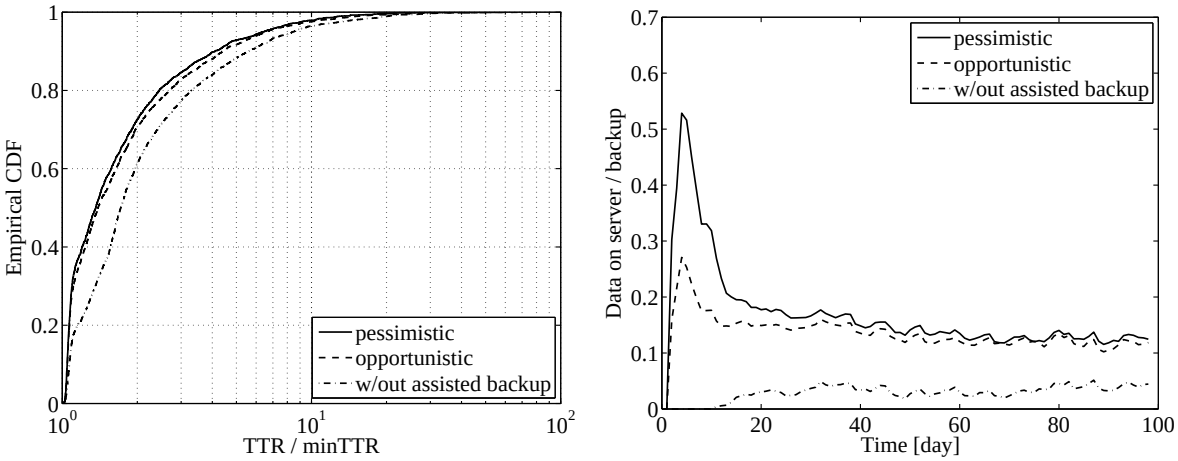
as in Figure 7.8) during the backup phase are significantly worse without assistance. Once the backup is considered complete on remote peers, the achieved DLP and TTR targets ensure the same quality of service in all schemes, *i.e.*, the rates of data loss due to long TTR are similar.

TTR is slightly longer in the schemes where the data center plays less important role during the backup phase (Figure 7.15(a)). Crashed peers that retrieve fragments from the data center experience short TTRs, while the limited upload bandwidth and the short online periods of storing peers hinder fast data retrieval solely from them.

Besides these performance indicators, we investigate the burden of the assisted backup through the following metrics: storage load, outbound and inbound traffic of the data center per peer[1]. We plot the data amount placed on (Figure 7.15(b)), and uploaded/downloaded (Figure 7.16) by the data center as the fraction of the overall backup load.

The central storage is presented in Figure 7.15(b): it grows rapidly in the beginning of the backup phases in the assisted backup cases due to the slow transfers toward peers; then decreases as more fragments are stored on remote peers. The peak is lower than the total backup, due to the fact that peers with fast uploads can outsource their backup to remote peers quickly. After a peer reaches its targets of eDLP and eTTR, further fragment transfers toward remote peers are carried out to minimize costs: storing more fragments on peers for free, and deleting fragments from the data center in exchange decreases the cost paid for the backup service at the data center. In the backup assisted systems more fragments are stored on the data center than in the systems where only the repairs are assisted: besides the transitional storage of repairs, peers, having difficulties with outsourcing their fragments to sufficiently reliable remote peers in

[1]The cost of 1-month storage of 1GB data is 0.1$, the cost of uploading 1GB data is 0.15$ while inbound traffic is free of charge, based on the pricing of Amazon S3 [2] at the time of this writing (2010).



(a) TTR (b) Storage load

Figure 7.15: Data center involvement in different data placement schemes

the necessary number, keep some storage on the date center, further increasing the previously defined "price of fairness".

The upload traffic of the data center (Figure 7.16(a)) can be divided into two categories: assisted repairs of crashed peers and retrievals. In general, central bandwidth costs due to repairs in case of disk crashes are similar, irrespective to the storage load in the data center. On the other hand, if more data is stored in the data center (pessimistic case), it uploads more retrieved fragments, thus the outbound traffic is higher. Storing less data on the data center results in savings in traffic costs as well, since retrievals would be performed from remote peers.

Backup uploads and assisted repair-related downloads (from remote peers to restore the backup object) constitute the inbound traffic of the data center (Figure 7.16(b)). The former type downloads seem to be more important in quantity on the short term, but as the number of assisted repair episodes increases, the inbound traffic related to them becomes dominant.

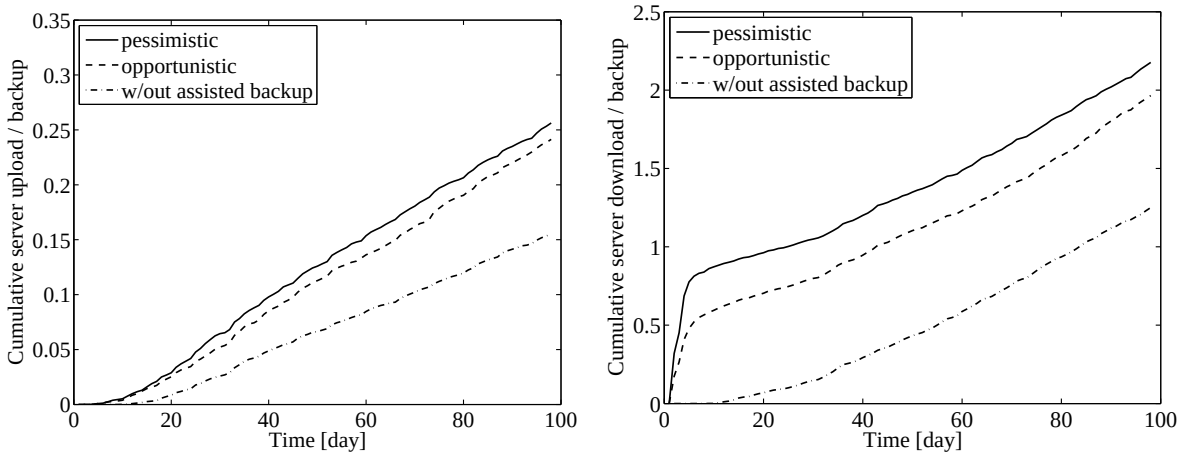
7.5 Choice of parameters

In this section we evaluate the consequences of our fragment size choice and the simulated resource contributions of peers.

7.5.1 Fragment size

In order to show the importance of choosing the right k, we investigate the effects of generating larger and smaller fragments. With $k = 16, 64, 256$, the fragment sizes are $f = 640, 160, 40$ MB respectively.

With the fixed-rate redundancy scheme, when k is increased, lower redundancy rate achieves



(a) Cumulative outbound traffic

(b) Cumulative inbound traffic

Figure 7.16: Data center traffic

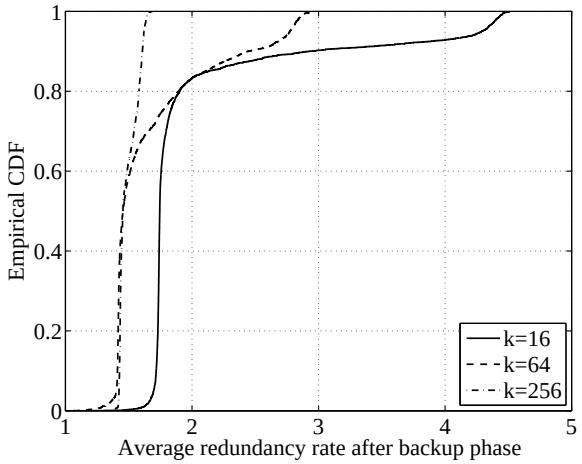
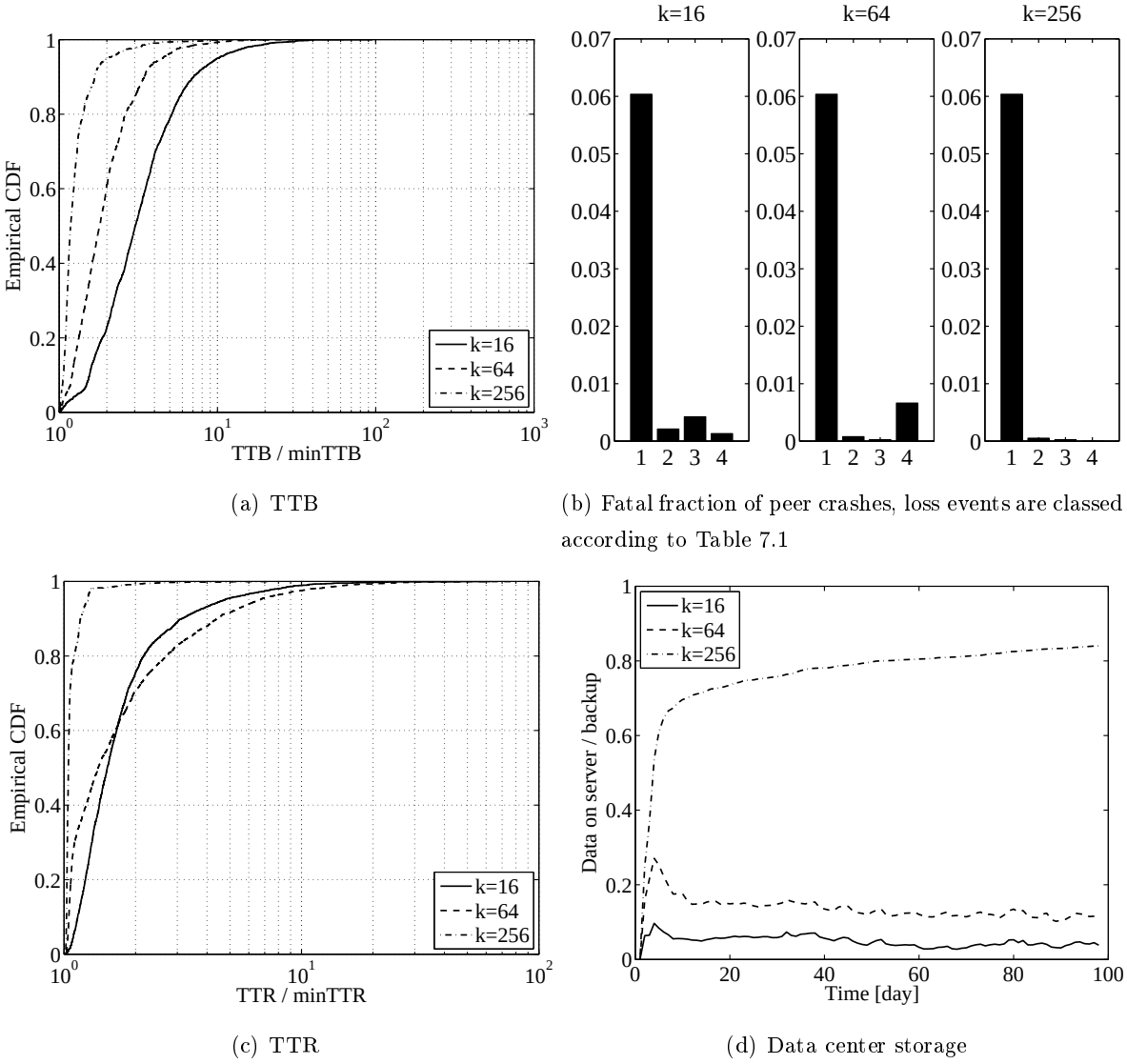


Figure 7.17: Redundancy rate distribution with different fragment sizes

the target data availability [17], therefore TTB decreases, storage and bandwidth burdens drop. Although r is decreased, generally $n = rk$ still increases, therefore a larger peer set is required.

Similarly, in our adaptive-rate scheme, the eDLP and eTTR targets impose a lower rate when k is higher. However, the increasing n is limited by the number of peers in our system, as shown in the applied redundancy rates in systems with adaptive-rates, grouped peer selection and assisted repairs and backups (Figure 7.17). With $k = 256$ most of the fragments must be stored on the data center (Figure 7.18(d)), rendering the redundancy rate very low, although increasing the costs in the meanwhile. Indeed, the fragment size has a lower bound: if fragments are too small, redundant fragments cannot be stored on remote peers in sufficient number.

Our chosen $k = 64$ adapts the expected number of encoded fragments of our redundancy scheme to the simulated system size, *i.e.*, to the number of peers in the grade groups. Transferring



(a) TTB

(b) Fatal fraction of peer crashes, loss events are classed according to Table 7.1

(c) TTR

(d) Data center storage

Figure 7.18: Fragment size analysis

160MB-size fragments also makes the discrete simulation (5-minute rounds) valid, because more frequent scheduling decisions are rarely required.

We apply a 1-day timeout on each fragment transfer: if the fragment is not completely uploaded within one day counting from the start time of the transfer, *e.g.*, because the two parties are negatively correlated in regard to their online sessions, then it is interrupted and canceled. Applying longer timeouts to interrupt slow fragment uploads does not affect TTB results, only if the fragments size is larger than what is generally transferable within the timeout. In order to minimize the number of failed transfers due to expired timeouts, we set the default number of maximal parallel uploads to one, and we prioritize uploads toward the online peer with which the fragment transfer is the closest to completion. In Chapter 6 we show that if the number of remote peers allows for smaller fragments and thus a higher n, increasing k ensures a more

efficient backup phase.

Figure 7.18 represents the simulation results of opportunistic backup assisted systems with our adaptive-rate scheme (with assisted repair) and grouped peer selection. The benefits of high k and reduced redundancy are shorter TTB and TTR, although peers experience similar rates of data loss episodes. The good performance of applying higher k is partly due to the growing involvement of the data center.

In Figure 7.18(d) we plot the storage load of data center as a function of time. The effect of small fragments on the TTB and TTR, with assistance from the data center, is basically that fewer redundant fragments are stored on remote peers. Everything that resides on the data center at the end of the simulations, would remain unsaved in an unassisted case. If the number of peers is low, relatively more load is put on the data center; equivalently, if k is not increased, as the system size grows, more of the central storage can be outsourced to remote peers, thus the larger the system is, the less costly it works.

In summary, $k = 64$ is a good design choice, given our adaptive-rates and the 376 simulated peers, divided into 2 groups. The random scheduling policy performs well with 1 maximal parallel upload and 1-day timeouts. The scalability of the system, *i.e.*, similar TTB, DLP and TTR outcomes and costs, no matter how many peers the system holds, is ensured if peers are sufficient in numbers in order to spread the encoded fragments, which are, in turn, determined by the system design through k.

7.5.2 Simulated user parameters

Instead of 10GB of backup data (resp. 50GB of shared space), we might have considered scenarios of skewed storage demands and capacities. Although in general, with differently skewed demand and capacities, both TTB/$minTTB$ and TTR/$minTTR$ remain similar [128]. Peers that back up huge amount of data would not suffer from transfer inefficiencies due to large n, since we apply backup objects to avoid such issues. It is hard to motivate any correlation between peer crashes, therefore the uplinks of peers with large storage capacity should not impose constraints on retrievals from them.

Without the symmetric fragment exchanges, users that store on large capacity peers along with a lot of other users might be constrained by the downlink capacity of the storing peer. However, transfer timeouts limit the number of exchanges, thus fragment downloads, that a high storage peer can perform in a symmetric peer selection scheme. Therefore selecting remote peers based on the data amount they want to backup or on the storage space they share does not provide any benefits.

In contrast to the heterogeneous peer storage demands and capacities, the presence of extremely low or high grade peers affects the system performance significantly. If low availability and/or poor connectivity remote peers are not grouped separately with respect to their grades,

the TTB and TTR of some users can be relatively longer. Similarly, peer crash rates higher than the simulation settings (expected time to peer crashes is 3 months) would cause more data losses, more intensive maintenance traffic on peers and on the data center. In extreme, crash rates higher than the speed of data repairs would lead to the loss of all backed up data. While further increasing the crash rate seems to be unrealistic, considering lower expected duration for peer membership, and allowing peer departures from the system would affect any simulated crash rate negatively.

Chapter 8

Conclusions

The P2P paradigm applied to backup applications is a compelling alternative to capital-intensive and energy-consuming centralized online solutions that become costly on the long-term. We revisited P2P backup and argued that such an application is viable even if it is based on unreliable storage resources, already deployed at the edge of the network. We proposed a system design that contains a tracker and a data center with storage capacity complementing the users. Because the online behavior of peers is unpredictable and, at large scale, crashes and failures are the norm rather than the exception, we showed that data redundancy, placement and transfer scheduling policies are paramount to achieve short backup and restore times, and high data durability. In this chapter we summarize our contributions in these aspects, and finally, we discuss the perspectives of future deployment and research directions.

8.1 Reliability

We suggested an adaptive scheme that strives to maintain a low data redundancy rate in order to complete backup processes quickly. We have shown that it comes at the expense of increased retrieve times, which we argued to be a reasonable price to pay, especially in light of our study on the probability of data loss. In fact, we determined that the vast majority of data loss episodes are due to incomplete backups. Our experiments illustrated that such events are unavoidable, as they are determined by the limitations of data owners alone: no online storage system could have avoided such unfortunate events. We conclude that short backup times are crucial, far more than the reliability of the P2P system itself. As such, the crux of a P2P backup application is to design mechanisms that optimize such metric.

Data redundancy and its maintenance guarantee the durability of backup by re-arranging stored data after a remote peer crashes or leaves the system. By applying a high redundancy rate, the storage and bandwidth requirements increase linearly. While our results make the redundancy rate that targets prompt data availability seem to be unnecessarily high for backup purposes, we

investigated the dangers of data loss, since with less redundant fragments on peers its probability is higher. Our adaptive rate scheme strives to provide *both* durability and performance at a low redundancy cost, relaxing prompt data availability by requiring that data becomes recoverable within a given time window.

Our scheme requires to quickly react to remote peer crashes in order to avoid data loss. Repairs are performed by either the data owners, which constitutes an implicit incentive for peers to maintain high online availability, or, if they are incapable to react to fragment losses, then by the data center in exchange for some fee. Since we advocate a *backup* system, our underlying assumption about repairs is that the original data is also stored locally until the hardware of the data owner crashes. Therefore repairing lost fragments does not require the owner to download the original pieces of a file to be able to generate new ones and transfer them to online remote peers.

8.2 Fairness

We introduced a distinction in the amount and the quality of storage space contributed by peers, above which users make selfish decisions. In our system, users select remote storage locations selfishly while building the necessary data redundancy. Storing data on those remote peers that provide the highest quality storage, with high availability and fast connectivity, entails the best achievable service. We investigated the uncoordinated approach to data placement in which users select the peers they want to exchange storage capacity symmetrically. By assuming that peers are selfish entities striving to minimize the resources dedicated to the system, we analyzed equilibrium topologies that materialize from the the process with the tool-set of non-cooperative game theory.

A direct consequence of the user-driven symmetric peer selection is that similar characteristic peers will store data fragments for each other. We have shown their expected strategies and we identified a simple decentralized way of implementing our mechanism which is robust against peers that deviate from the normal system operation. We have shown that peers self-organize into cliques according to their attributes. The redundancy rate applied by peers in a high-grade group is lower compared to a low-grade group, hence peers have to dedicate less storage and bandwidth. Additionally, high grade partners provide shorter time to backup data, and shorter retrieval duration, although the increased redundancy rate enhances the chance for low grade peers to find storing remote peers online during their uptime. The system stratification turns out to be the key factor to create incentives for peers to increase their storage characteristics: we showed that in some cases the least contributing players have incentives to improve their availability and connectivity.

The operation of a single peer is simple: it queries the tracker for the required number of

remote peers with similar characteristics, to be able to store its encoded data. As the user number grows, peers with very similar contribution level can be found, but sophisticated measurement techniques are required. Due to the bi-lateral nature of peering agreements, a peer will not be able to "cheat" as any tentative of establishing links to peers with higher quality contributions will fail, preventing free-riding. A secure and unique identification scheme is however required to prevent "spoofing" identities.

The presented user-driven symmetric peer selection provides built-in incentives for users to increase their availability and the bandwidth they offer to the system. However, it also has shortcomings due to the strict symmetric storage relations. As a possible remedy, we investigated assisted system designs where backup data is partly and temporarily placed in a data center. We studied various data placement choices that ensure high quality of service in terms of data loss, backup and retrieval durations. Assisted systems lower the storage and traffic costs of a central service significantly, however the cost can be eliminated completely only if peers contribute sufficient online availability, storage and bandwidth, moreover if they are sufficiently numerously present in the system.

8.3 Efficiency

When a remote peer goes offline before completing a fragment transfer, the user either waits for the remote peer to come back online or restarts the transfer with an other one. In both cases precious time is lost. Essentially this is the reason why we focused on the decisions of fragment transfers between unreliable peers: to improve the efficiency of the backup and retrieve phases. We gave a novel formalization of finding the optimal scheduling and showed that, with full information, the problem can be solved in polynomial time by reducing it to a maximal flow problem. Without full information, optimal scheduling is unfeasible; however, we showed that as the system size grows, the gap between randomized and optimal scheduling policies diminishes rapidly.

The reasonably good performance of random scheduling can be ensured in a P2P backup application with well-suited system design: first, the length of backup phases decreases if the number of remote peers greatly exceeds the number of encoded fragments to store, second, creating small fragments, and thus increasing their cardinality while respecting the previous point, also shortens the archiving and restoring processes, and third, analogously to the first point, with larger applied data redundancy the retrieves are faster.

Based on these observations we suggested to decrease the fragment size according to the potential peer set size: short transfers of small fragments are interrupted less frequently. We also proposed to apply a timeout on transfers which has to be set according to the fragment size. Nevertheless, while random data transfer scheduling yields nearly optimal results, it cannot

eliminate perfectly the inefficiency of the backup and retrieval phases. We summarize our system design elements that answer each of the scheduling inefficiency concerns mentioned in Chapter 6:

1. random scheduling is applied, but the fragment size is adapted to the system size;
2. the applied redundancy rate is lowered as much as possible to minimize the duration of the backup phase (Section 4.2), but keeping it high enough to ensure the efficiency of retrieve phases;
3. fragment granularity issues are mitigated by
   - constructing as small-sized fragments as possible in regard to the *required* and the *available* number of remote peers, since the probability of interruption during their transfer is lower;
   - prioritizing uncompleted transfers that are closer to completion in order to lower the expected number interrupted jobs waiting to resume;
   - among the online storing peers, the peer chooses to retrieve its fragments from those that have already uploaded the largest parts;
   - retrieval uploads receive highest priority at the storing peers;
   - applying timeout after which if the success of a transfer is not confirmed it is canceled, in order to provide the possibility of re-assigning fragments to other remote peers when negatively uncorrelated peers are matched;
4. in order to avoid uplink and downlink bottlenecks, a low cap on the number of parallel uploads is enforced by each peer but they can serve multiple download jobs, moreover all maximal parallel connection limits are adapted to the the connectivity characteristics of each peer.

8.4 Perspectives

The presented work provided a design basis for the implementation of a fully fledged P2P backup application prototype. The elementary *backup* and *retrieve* operations are carried out as follows.

A given user, interested in the backup service, bootstraps in the system. Using a control interface to the tracker, the user indicates the amount of data it wants to backup, and sets a bootstrap value for its online availability and dedicated bandwidth. Provided these characteristics, the tracker offers potential remote peers to make storage exchange links with, based on the findings that selfish users eventually self-organize into cliques holding peers with nearly the same uptime and bandwidth values. Once a stable neighborhood is found, the user faces two

options: it keeps its initial contribution level, with the local storage required to dedicate to the system, or lowers the quantity of shared local storage at the cost of an increased quality. When the decision is made the backup data with a well-suited redundancy ratio that corresponds to the characteristics of the partners, is swapped between the parties.

Every time a user wants to insert new data to the system, the backup operation is carried out. Retrieve works similarly: the user looks up its account at the tracker to find the coordinates of remote peers in order to download the backed up data. Then it contacts online peers to gather the necessary data fragments from them.

A P2P backup system operated by the Internet Service Provider (ISP) offers benefits for both users and ISPs, compared to existing central storage services. Users do not pay for the backup, but exploit unused end-user resources instead. ISPs spare costly inter-ISP traffic due to large data shipments from subscribers to data centers, and/or avoid the burden of maintaining own data centers in order to offer storage service themselves.

Once deployed, the system can be exploited to collect measurements about the natural heterogeneity of user demand in terms of storage requirements, and of realistic bootstrap resource contributions that users would set. This could give an insight about the validity of our assumption about uncorrelated user availability, dedicated bandwidth, shared storage space amount. Furthermore, these conditions would determine the theoretic equilibrium state of the system, the number of cliques, their sizes and the grades within, and also could give answers on the realistic user responses, given to the incentives that the system employs.

Another a measurement research direction is to find the optimal size of backup objects, which in turn requires knowledge about patterns of data production, not only the initial amount of data that a user wants to back after its registration to the system.

Measuring the amount of resources a peer dedicates to the system represents an important issue. The observation of the availability and uplink capacity of remote peers is necessary for our approximation techniques; we assume that monitoring is performed by the tracker and any peer in the system can query it to obtain the qualities of other peers in the system. Indeed, it is common practice (*e.g.*, in Wuala) to rely on a centralized infrastructure to monitor peer resources. However, a decentralized approach to resource monitoring is an appealing research subject, *e.g.*, if the connectivity or availability of a peer is measured by other peers in the system, they may estimate it differently. A nearby peer may experience faster communication, a peer that has positively correlated uptimes with the observed peer will measure higher availability.

Part II

Distributed Dynamic Spectrum Allocation

Chapter 9

Introduction

We propose a distributed spectrum management framework to *allocate* radio frequency bands for wireless service providers dynamically. Our goal is to reach high efficiency in frequency utilization, *i.e.*, to allocate spectrum to those licensees that value it the most. We build a self-organizing scheme in which the participants, *i.e.*, the wireless service providers, manage the allocation of their frequency bands at arbitrary points in time. We give the possibility of choosing the adequate band and the activation time of the license in the hands of the participants.

We model interference effects among the participants by reflecting realistic constraints of their co-existence despite abstract simplification. In case a participant cannot fit on a given frequency band due to excessive interference caused by others, we allow for *exclusions*: a newly allocating participant (as any other participant) may eliminate other, actively operating participant(s) if this action potentially improves the efficiency of spectrum utilization. The exclusions are based on the pricing of spectrum which is managed in a distributed way.

The central authority plays regulative role and controls only the interference and payments of active frequency-leasers. By design, our framework favors the application of modern radio technologies which are interference-tolerant, furthermore, it takes into account the selfishness of participants (in the game theoretic sense) and supports dynamics in allocation demands. The goal is to maximize frequency utilization, which is the most important objective of introducing DSA.

9.1 Static versus dynamic spectrum allocation

Radio spectrum exploitation is historically regulated by governmental authorities, *e.g.*, the Federal Communications Commission in USA. The regulation, lead by these national bodies, results in the static allocation of frequency bands with rigid specification on the geographic operation and on the usage purpose (*e.g.*, broadcast radio/TV, cellular services, wireless LAN) of the license. The growing need for spectrum, generated by many novel applications, claims the revision

of this management scheme, since the current static frequency allocation results in suboptimal spectrum utilization due to well-known reasons.

The capital intensive governmental licenses make the frequency bands, auctioned for long-term, access-limited ("big player syndrome"); moreover, the peak traffic planning causes temporal underutilization in less busy periods. Furthermore, the spatial and spectral restrictions on frequency re-usage due to rigid interference handling policies exclude many potential frequency exploitation opportunities.

The emergence of novel radio technologies enables the application of advantageous spectrum policies wherein allocating spectrum bands for licensees is performed with various spectral, spatial and temporal parameters, thus possibly improving spectrum utilization. While actual spectrum allocation is not efficient because of the aforementioned constraints and fixed frequency ranges for existing services, new generation radio interfaces support flexible transmission frequencies (*e.g.*, dynamic frequency bands in the Long Term Evolution (LTE) project), furthermore the convergence of services makes actual restrictions seem out of date.

A well-suited dynamic spectrum allocation (DSA) framework must offer solution for every key issue. Interference relations among frequency leasers (*e.g.*, caused interference when operating on the same frequency band) must be taken into account without simplifying assumptions, thus reflecting appropriate spectral and spatial constraints when allocating spectrum bands to the licensees. Furthermore, it must fulfill the basic requirements of general resource distribution, when limited resource is to be divided among selfish participants. An ideal framework should not impose temporal constraints on licenses in terms of duration and renewal periods of allocation. Albeit the existing literature that tackles possible spectrum allocation models is vast, our approach provides novelty in many aspects.

9.2 Focus of the work

If licensee candidates are interested in allocating frequency bands for time periods that are not synchronized and do not have the same lengths, auction-based management schemes are not particularly well-suited. We consider the issue generated by *delayed* activation times of selfish frequency leasers. Furthermore, distributed schemes often fail to mitigate the effects of selfishness among the participants. Our distributed allocation framework provides solution for both: possible exclusion of active licensees allows the candidates to enter the spectrum at arbitrary points in time, and our pricing scheme guarantees successful allocation in exchange for adequate payments.

Our contribution also involves excessive discussion on the complexity of decision problems related to exclusions and frequency band selection, moreover well-motivated heuristics are proposed for these latter. We compare the performance of the heuristic algorithms, that are built

on the observations of analytically tractable scenarios, in numerical evaluations. We prove that the proposed strategies are light-weighted and perform well in realistic situations.

We overview the related work in Chapter 10.

In Chapter 11 we present our framework through the introduction of the node and the interference models, moreover we introduce the allocation and pricing policies and define the notion of node arrival sequence. We discuss the complexity of allocation decisions, generated by our framework, and propose reasonable policy choices for node exclusion strategies built on the insights to the algorithmic characteristics of the frequency band selection.

In Chapter 12 we evaluate the proposed heuristics numerically. Chapter 13 concludes our work.

Chapter 10

Related work

Since the appearance of enabling technologies, the management of systems implementing DSA has received much attention. Therefore many related works have appeared so far. We keep our focus on papers that present allocation and pricing solutions for DSA and we do not consider the large field of research on underlying technological issues.

10.1 Central allocation

The management of DSA was first discussed in [87]. The authors presented the concepts of DSA as an alternative to fixed allocation schemes. They showed the potential for gains in spectrum efficiency and several issues related to improvements to the fairness and effectiveness of the allocation scheme.

The seminal work of Buddhikot and Ryan [22] initiated the sequence of papers from Buddhikot *et al.* focusing on spectrum allocation and pricing. All of them present models in which a central spectrum broker allocates governmental licenses of spectrum for short leasing times. They define the concept of coordinated DSA and the spectrum broker (*i.e.*, who owns the spectrum and leases it), along with different allocation algorithm types (online vs. batched), the authors of [22] also introduce the important notion of interference *conflict graph*, and the cascading effects among frequency leasers on blocked list. The authors also provide some linear programming formulation of the spectrum allocation with feasibility constraints: maximal service vs. minimal interference, maximal broker revenue vs. *max-min fairness*.

In [117] the authors go further and analyze pricing issues: they propose models on auction-based and peak-load pricing depending on demand and supply. Furthermore, Buddhikot *et al.* [125] propose fast heuristic algorithms to perform the central allocation by optimizing the same metrics as before: obtaining maximal satisfied demand or minimal interference. Based on the *conflict graph*, the authors arrive to well-known NP-hard graph theory problems (coloring and cutting problems respectively), and provide nice theorems and proofs on the efficiency of

their heuristic algorithms under some interference graph assumptions.

The work in [146] highlights the weaknesses of the widely-employed interference modeling tool, *i.e.*, the pairwise conflict graph, and they show how to derive this latter from physical interference models so that it produces near-optimal allocation.

10.2 Distributed allocation

Another platform dealing with spectrum allocation was initiated by Cao, Zheng and Zhao. In their first papers [24, 151], they introduce distributed algorithms to allocate spectrum by local coordination and collaborative sharing among users through bargaining. Their solution reaches optimal frequency assignment after topology changes faster than a central optimization or a distributed graph coloring technique as in [109], starting from scratch. However, users are assumed to cooperate in order to improve social welfare, defined on spectrum utilization and *max-min fairness*, thus selfishness and the need for incentives are not taken into account. Interference is handled by the conflict graph approach.

In [25], Cao and Zheng argue further on the efficiency, convergence time and communication overhead of the proposed scheme, and deduce lower bounds on system performance characteristics, such as fairness level, and upper bound on complexity.

10.3 Spectrum auctions

Subramanian *et al.* give a general bidding framework, where the broker strives to maximize its revenue, in [124]. This paper presents a revolutionary work in the sense that the main objective of the broker is revenue-maximizing, and spectrum allocation is carried out on both pairwise and physical interference models. Moreover, the framework allows for heterogeneous channels and general complementary bidding functions. The authors propose greedy algorithms for the NP-hard allocation problems; theorems about approximation bound are given as well.

In [59] Gandhi *et al.* switch from the distributed framework to a central, i.e, auction-based, allocation scheme in which the objective is maximizing the revenue. This work is similar to that of [124], although some of the assumptions are highly restrictive: the model presented in [59] supposes pairwise interference conflict graph, piece-wise linear bidding functions, and homogeneous non-overlapping channels. Under these assumptions, the authors formulate the allocation and pricing as linear programming problems, and give approximation bounds of their heuristics. They also show the trade-off between revenue and fairness, and the difference between global market-clearing price and discriminatory pricing schemes.

An early work on user pricing is presented in [70], where the authors provide a model of the profit of a service provider, based on stochastic user acceptance assumptions. The conclusion

they arrive at is that eventually the iterative bidding for spectrum increases its utilization.

Then Cao and Zheng present the SPARTA framework in [26]. The focus of the proposed system is stability of dynamic allocation, which is achieved by interference-aware admission control and demand shaping. The framework supports outage-avoidance with adaptive algorithms based on demand statistics and interference conflict graph. Performance evaluation is given for binary demand shaping.

Rodriguez *et al.* show in [114] the efficiency improvements of DSA for specific application scenarios, where services have complementary busy hours and inter-cell interference. They consider video entertainment services and cellular telephony with co-existing Universal Mobile Telecommunications System (UMTS) and Digital Video Broadcast (DVB) terrestrial networks. The spectrum broker periodically holds inter-related auctions of short-term spectrum licenses in different cells.

The pricing model presented in [68] applies a second-price sealed bid auction of transmission rights in each time slot. The user with the highest bid, who then pays the price equal the second highest bid, gets the right for a given time slot. The authors show the existence of Nash equilibrium of the bidding strategies in the two-user case. The authors of [15] also present a game theoretic model where the spectrum broker *sequentially* allocates frequency bands among users by second-bid auctions. They derive worst-case efficiency values under some restrictive assumptions.

Zheng *et al.* return to auction-theory by proposing to implement the Vickrey-Clarke-Groves (VCG) mechanism [28, 66, 138] for spectrum allocation in [153]. Truthfulness and strategy-proofness are achieved by computationally-efficient heuristic algorithms, while maximizing broker's revenue or the social welfare. Their TRUST framework, presented in [154], deploys double auctions to reach spectrum allocation efficiency and truthfulness.

Interestingly, the DSA system presented in [79] also performs allocation and pricing by VCG mechanism. The authors propose a general spatio-temporal model with physical interference modeling. Frequency leasers participate in one-shot multi-bid auctions and obtain frequency usage rights for prices that maximize broker revenue or social welfare.

10.4 Secondary spectrum usage

Many further related works consider pricing issues towards secondary users in DSA models.

The framework in [120] relies on the VCG mechanism in a sealed-bid knapsack auction when determining spectrum allocation, but in the presented economic model the authors also account for the interaction between wireless service providers and users, and determine dynamic pricing rules to capture their conflict of interest. On the other hand they do not discuss interference issues.

[9] analyzes spectrum reselling, and the balance between the income and the incurred cost due to interference caused by the buyers.

The authors of [145] extend the VCG mechanism with new pricing schemes in order to avoid spectrum reselling loss due to colluding secondary users. They introduce virtual groups of users based on the conflict graph, and price the spectrum accordingly; the approach they take is favorable from many perspectives, however due to computation complexity, it does not seem scalable.

The work [48] examines the case of reselling spectrum from a different angle: the setting is modeled as a Stackelberg (leader-follower) game among the spectrum owner, the primary, and the secondary users. The revenue increase of the broker is provided by the higher frequency utilization.

An other game theoretic tool is exploited in [103], where the authors consider the spectrum allocation as a potential game between cooperative users, and as such, best response dynamic results in pure Nash equilibrium.

The authors of [78] present a study on the available spectrum for secondary devices. They show that secondary devices can utilize very little of partially used spectrum, if conservative access policies are adopted to minimize interference with primary users. Therefore they propose virtual frequency bundling, where secondary devices build reliable channels by combining multiple, randomly selected, unreliable frequencies.

Our framework, that we present in the next chapter, combines beneficial aspects of the above works: our goal is to create a *distributed* allocation scheme where spectrum bands are assigned based on the outcomes of *auctions*. Our model creates synergy in the sense that frequency bands are efficiently utilized, but there is no need to perform high complexity pricing calculations centrally, at periodic auctions. We avoid to oversimplify interference issues, and the disadvantages of applying VCG mechanisms.

Chapter 11

System model

We investigate the possibility of allocating radio spectrum among multiple applicants dynamically in a distributed manner. The actual spectrum allocation policies, *i.e.*, governmental licenses for frequency bands sold for long-terms, are not efficient because peak traffic planning causes temporal underutilization in less busy periods, furthermore, the spatial and spectral restrictions on frequency re-usage due to rigid interference handling policies exclude many potential frequency exploitation opportunities.

The emergence of novel radio technologies enables allocating spectrum bands for licensees with various spectral, spatial and temporal parameters, thus possibly improving spectrum utilization. The framework that we present allows for potential frequency leasers to reserve spectrum bands according to their needs at any time, anywhere. The management of licensees is built to be as decentralized as possible in our distributed dynamic spectrum allocation (DDSA) model. Our most important goal is to distribute and utilize radio spectrum efficiently.

11.1 Distributed spectrum allocation model

Our proposed allocation framework builds upon the following criteria: the spectrum utilization should be maximal, while sustaining the rule-of-thumb that in case of "conflict of interest" the frequency bands are allocated to those who hold the highest utility. Thus, in our model, the fairness receives new connotation, *i.e.*, unlike the *max-min fairness* presented in [22, 24] that assures a bit of the spectrum for every participant, in our model the one who pays more gets the frequency band. This approach yields fairness towards the spectrum band itself because this latter is going to be used and exploited by the leaser holding the highest utility. Note that we assume that the price that a leaser is willing to pay for a spectrum band, *i.e.*, its utility, is justified by the extent to which the leaser plans to utilize the frequency. One may consider the possibility of distributing a part of the spectrum among the frequency leasers in order to reach the desired *max-min fairness* while our framework is only responsible for the allocation of the

rest of the spectrum.

In this section we introduce our model: first, we present a simple way to describe the economic perception of participants, related to the spectrum allocation, second, we argue on applying a general interference model, third, our allocation scheme, then our pricing policy are defined and discussed.

11.1.1 Node description

Our model consists of participants which are distinguishable entities exploiting radio spectrum at fixed, and/or confined geographic locations, *i.e.*, base stations of wireless service providers, private radio systems. Multiple participants may belong to the same Network Service Provider or Radio Access Network, *e.g.*, to a mobile phone service provider that operates on different geographic locations, but these participants are still considered as different nodes. We denote the set of the nodes by $\mathcal{I} = \{1, 2, \ldots, I\}$.

As in [79], we characterize the possible frequency leasers, *i.e.*, the nodes, with their frequency band demand and their utility describing their willingness to pay for acquired frequency shares. Let q_i denote the size of the contiguous frequency band required by node i. In order to model interference-tolerance, we also define the "bearable" interference level for each participant: α_i^f stands for the maximal interference level node i can bear from the other nodes $j \neq i$ on frequency f. Interference may occur if the same frequency is used by other nodes than i, and it is defined as the *maximal* measured Signal to Interference-plus-Noise Ratio (SINR) value on the operating area of i and denoted and approximated by $\sum_{j \in \mathcal{I}} \omega_{ji}^f$, ω_{ji}^f being the interference caused by node j at node i (by definition $\omega_{ii}^f = 0 \; \forall i \in \mathcal{I}$). The frequency-dependent interference level depends on many aspects: geographic distance, transmission power, applied technologies, coding, type of radio transmitters, etc.

Our model takes into account interference as a source for spectrum utilization degradation, since too noisy spectrum, due to other nodes using the same frequency in nearby regions, cannot be utilized. Kovacs *et al.* [79] decompose interference in three components. The first depends on the geographic location and size of the operation regions. That is what the authors call the *geographic* coupling. It is zero if there is no overhearing at all, and the maximal value means that the radio transmission is heard undamped between any pair of nodes. The second is the measure of how much different radio technologies affect one another: the level of disturbance (or jamming) between nodes is captured by the *technology* coupling that reflects the positions and types of radio transmitters. As an example, a carefully designed micro-cell structure with directed antennas causes much limited interference than a central unidirectional transmitter placed in the middle of the region. If two nodes have the same spectrum slice within the same region, and the technology coupling is zero, they do not cause interference to each other at all, while if it is maximal the spectrum is ruined for one of the nodes. The third component is the

transmission power of every node. The cumulative effect of the geographic and radio technology coupling between any two nodes, operating on the same spectrum with the given transmission powers is simply the product of the three factors, and these factors can be asymmetric as well.

We denote by u_i the utility of a frequency slot for node i $\forall i \in \mathcal{I}$; its value is based on the expected income of the node, provided that it gets the necessary quality spectrum band of q_i slots in order to launch its service. We assume that the overall utility of a node is homogeneously divided over its required frequency slots.

11.1.2 Interference model

As the majority of the related works, we assume that the frequency spectrum, denoted by F, can be divided into small predefined sized, non-overlapping, homogeneous spectrum slots, denoted by f in general. Let F_i denote the frequency band allocated by node i $\forall i \in \mathcal{I}$. We denote the size of a band F_i by $|F_i|$, and we denote by $\mathcal{F}^f$ the set of nodes that allocate frequency slot f within their bands, *i.e.*, $\mathcal{F}^f = \{i : f \in F_i, \forall i \in \mathcal{I}\}$.

Many works in the literature tackling DSA state that even this simplified multi-unit goods, multi-buyer allocation is more complex than a simple combinatorial problem. One main reason for the complexity, usually ascertained as NP-hard, is that a feasible allocation must meet the frequency interference requirements.

Two main approaches exist to model the interference relations among participants. Closer to reality, the *physical interference model* considers signal attenuation formulas to take spatial and transmitting power parameters into account in order to establish the interference values. Many prior works on spectrum management simplifies this problem by assuming that radio interference can be modeled by a *conflict graph* where nodes represent the participants, and an edge exists between two given nodes if they can not utilize the same frequency band without facing serious performance diminution due to high interference. Although a conflict graph gives the opportunity to implement graph coloring and graph cutting algorithms in order to generate feasible allocations, it has been shown to have important drawbacks as well. Authors of [146] argue that the conflict graph is unable to model aggregated (cumulative) interference, moreover [79] reasons on the asymmetric nature of interference, *i.e.*, one of two connected nodes in the conflict graph may tolerate the interference generated by the other node, while this is not true for the opposite direction of this relation.

An interesting work [146] attempts to generate optimized conflict graphs from physical interference models so that one can apply well-developed graph-based spectrum allocation solutions on top of various practical physical interference models. Our model, although reflects several concepts of [146], follows the general interference model deployed in [79]. Since significant complexity is reached when optimizing spectrum allocation centrally, we make the case of a distributed scheme that fulfills the maximal spectrum utilization with the highest utility requisites. Note

that even if the central authority is relieved from the burden of computation-intensive allocation and pricing algorithms, it is still required in our framework to maintain regulations in terms of interference and for accounting purposes.

11.1.3 Distributed allocation — one-way exclusion

In our system the spectrum allocation is carried out by the nodes themselves, thus, there is no need for central auctioneer to handle bids and node preferences, and to calculate optimal allocation as in auction-based frameworks. The distributed allocation and pricing make the system flexible in terms of possible allocation of spectrum at any time without a centrally announced or periodical auction. We present the allocation rules here, and we evaluate their performance in a selfish environment in the following section about pricing.

The work presented in this paper focuses on sequential *arrivals*. We assume that nodes activate their operation at arbitrary points in time, and we call these actions as *arrivals*, moreover, the node that has allocated a frequency band the most recently as *newcomer*. We make the case of a distributed allocation that allows for delayed node arrivals and we investigate the effects of such a novel approach. Note that we do not consider any simultaneous arrivals. Nodes initiate spectrum allocation attempts at their arrivals.

Four types of outcome could occur in our distributed setting when a newcomer node i makes an allocation attempt. These are the followings:

Seamless

If node i demands the use of a given frequency band F_i and *after* allocating it every active node j that is present on any frequency slot of the band ($j \in \mathcal{F}^f \ \forall f \in F_i$, including i) perceives less interference than its tolerance level, then the frequency allocation of the newcomer is seamless. Formally, if $\forall f \in F_i \ \alpha_j^f \geq \sum_{k \in \mathcal{F}^f} \omega_{kj}^f \ \forall j \in \mathcal{F}^f$, then the newcomer acquires the right to exploit F_i without further actions.

Exclusive

If the above condition of seamless arrival is not fulfilled, the exclusion of some nodes is necessary due to unbearable interference constraints. In our framework, disturbed nodes attempt to exclude other nodes that are causing unbearable interference to them. We make the case of self-maintaining interference levels, *i.e.*, the nodes themselves need to lower their perceived interference by excluding other nodes from the specific frequency slot, if necessary. That is what we call the one-way exclusion: if the interference experienced by the newcomer node i is higher than α_i^f, i buys out the node(s) that cause(s) a part of the interference.

Defensive

If the interference that node i perceives is not higher than α_i^f, but on the other hand any other node j of the active nodes on F_i cannot support the increased interference due to i, following the same one-way exclusion policy, j may make an attempt on excluding i in order to keep the interference under its required level α_j^f. In this case i has to resist buy-out attempts in order to allocate F_i successfully.

Exclusive and defensive

If i performs an exclusive allocation, and any of the remaining nodes is over-interfered, than the allocation of i is both exclusive and defensive.

If the allocation of a newcomer involves buy-out attempts, the excluding node(s) pays off the interfering node(s) on a frequency slot, but not *interfered* ones, *i.e.*, exclusion is one-way. As a direct consequence, no interference threshold information needs to be known about other nodes. Note that if a node is excluded from a frequency slot, it is forbidden by policy to try to allocate it again until the node that excluded it is still active.

Since exclusive allocation may happen at any time (unlike in the models in which central auctions are held on predefined dates), service interruption may easily occur. Therefore a delay might be applied after an exclusive allocation so the ex-leaser could use the frequency during a short time after its exclusion while preparing for the outage or a frequency band re-allocation. Licensed spectrum can also be "marketed" through advertising selling-willingness in exclusive situations, allowing *voluntary* exclusion of active nodes from their unwanted allocated frequency band.

Our framework supports dynamics in many aspects: a node may decide to buy up frequency bands at interfering nodes in order to improve ω_i^f on its own geographic service location, so our model supports intentions to increase quality of service, moreover spectrum slices can be merged and divided, over the unit size threshold. Each node maintains its utility about the spectrum slice it holds, and in the voluntary case this value starts to drop enabling new nodes to "buy out" the actual leaser; in the exclusive case the actual leaser gets overbid, but receives compensation for the exclusion based on its advertised utility. The truthful bidding and utilities are direct consequences of the system rules; this aspect is evaluated in the next section.

11.2 Pricing directives

In this section we present our second-price policy and the management role of the authority.

11.2.1 Second-price auctions

If any exclusion is performed at the allocation attempt of a newcomer node, we assume that the two parties, *i.e.*, the newcomer and an actual leaser of the given frequency band, both issue their bids, then the higher wins and pays the second bid. After both nodes have issued their bids, the possible outcomes are the following.

Successful buy-out

If the bid of the newcomer node is higher, it pays the lower bid to the authority and the frequency leaser node is excluded. The excluded node receives financial compensation from the authority.

Successful defense

If the bid of the frequency leaser is higher, the authority does not impose any fee on it, and the attempting node is *implicitly* excluded without any compensation.

This pricing rule provides the efficiency of resource allocation, *i.e.*, buyers with higher bids get the right to use the frequency spectrum.

Definition 12 ***Bidding*** *When node i attempts to exclude node j on some frequency slot f with bid b_i, if $b_i > b_j$, where b_j is the* defense bid *of j, then j is excluded, and i pays $b_j \leq u_j - c_j^f$ to the authority. Furthermore, the authority pays c_j^f, the sum of the expenses of j paid for prior exclusions of other interfering nodes on frequency slot f, to j as a compensation after its exclusion. By paying b_j to the authority, c_i^f increases,* i.e., *$c_i^f := c_i^f + b_j$, which lowers the budget of i for further exclusion bids. An exclusion attempt is unsuccessful if $b_i < b_j$: neither i nor j pays the authority and i is implicitly excluded in this case.*

We assume that the authority launches the distributed allocation framework for a time period with predetermined and announced length. Furthermore, we consider that the authority maintains a reserve price (denoted by c_a) for all spectrum units at the beginning, and resets it at the end of each period in order to avoid speculation on initially free spectrum bands. We add that the authority should fictionally cause interference (denoted by ω_a^f) to every node on the spectrum at the beginning of each period so that interference-tolerant nodes have competitive advantage, provided by the allocation rules, over those nodes that do not bear this level of interference. "Pioneer" nodes, those who demand the use of given frequency slots first in the current period from the authority, pay the predefined reserve price directly to the authority, if they can not bear the imaginary interference that the authority creates on the slot in question. Formally, if $\alpha_i^f < \omega_a^f$, i pays c_a price for each frequency unit to the central authority, and acquires the right to exploit the frequency, otherwise i does not have to pay anything in order to use the frequency.

Frequency band units are advertised with the utility of the actual leaser, spread over all its leased units: the frequency units hold the same unit utility. In this sense there is no synergy of spectrum, since the sum of the affordable bets that a given node can make for two disjoint frequency bands cannot exceed a potential bet for their aggregate, nor it is possible in the other way.

11.2.2 Utility-based pricing and rationality

We now refine the definition of the utility. Since, in our assumption, the authority re-launches the DDSA framework periodically, the utility of the spectrum is high at the beginning of such a period, and 0 at the end of the period, when it is auctioned on the reserve price again by the authority. Therefore the defined utility also depends on the time remaining until the end of the actual period. The length of the DDSA periods is to be determined by the authority; it may scale from the order of minutes to the order of decades, even to infinity. Nodes determine their utility based on their expected income provided that they get the necessary quality and quantity spectrum resource in order to launch their services. This decision also requires knowledge about the date of the end of the period since income needs to be discounted accordingly.

Definition 13 ***Utility*** *The present value of the future incomes of i at time t is defined by*

$$\Theta_i^{PV}(t) = \int_t^T \Theta_i(\tau) e^{-r_i(\tau - t)} d\tau, \tag{11.1}$$

where $\Theta_i(\tau)$ denotes the continuous income of i at time τ due to the service provided on the frequency band which meets both quantity and quality requirements. Furthermore, T stands for the end date of the current DDSA period, and r_i expresses the continuously compounded rate of node i since $r_i = \ln(1 + r_i^a)$, r_i^a being the annual interest rate of i.

Regarding to Definition 13 the following parameters and variables have outstanding importance:

- The concept of DSA responds to frequency utilization efficiency issues in the presence of time-varying service demand. Therefore we consider evolving economic perception of nodes by introducing $\Theta_i(t)$ as a time-variant parameter for node $i \; \forall i \in \mathcal{I}$. As a direct consequence of Definition 13, the utility introduced above equals to $\Theta_i^{PV}(t)$, thus it changes in time, allowing for frequency trades even in an invariant node set.

- r_i^a and $r_i \; \forall i \in \mathcal{I}$ are normally defined by risk-free investments in economics, however in our case where nodes might have already deployed investments (introducing amortization), this discount rate is supposed to be different and even time-variant for every node.

- T represents the time ahead until the end of the actual leasing period. If $T \to \infty$, *i.e.*, the authority launches the DDSA framework for only once, $\Theta_i^{PV} = \frac{\Theta_i}{r_i}$ by assuming constant $\Theta_i(t) = \Theta_i$ and r_i.

Note that our approach of pricing is slightly different to those, who propose time-based pricing. This latter requires nodes to pay to the authority a leasing fee for every time unit they utilize the spectrum, and the time-unit payments might be determined by the bidding rules. In this case, bids are determined by instantaneous $\Theta_i(t)$ $\forall i \in \mathcal{I}$ and $\forall t \in [0, \infty]$, since rational nodes would not overbid their true utilities. On the other hand, in our setting, it is the combination of $\Theta_i(t)$ and r_i $\forall i \in \mathcal{I}$ that defines the maximal bid, thus not the prompt utilities decide about the resource allocation, but the income estimations for longer terms.

Because of this long term evaluation of the spectrum, we assume that exclusive re-allocations will happen relatively rarely. Cheap resource, typically a node operating on a frequency band with low α^f, is not likely to be bought out, because possibly highly interfered spectrum does not provide high utility for a newcomer. On the other hand, if a given radio resource is highly valuable, it is probably allocated on a high price, which moderates further intensive buy-outs. In both cases a slight over-bidding beyond the utility of the peer could lead to higher payout on the long run. At the acquisition the buyer risks a possible overpayment (if other nodes place bids between the revealed and true utility of the winner), but a possible buy-out from a newcomer may raise its income by the difference between the true and revealed value due to the exclusive re-allocation pricing. However, the higher the revealed bid is, the less likely it is going to be overbid: the expected revenue from reselling leasing rights is discussed in the following section.

We assume that the nodes are autonomous and selfish, thus they try to maximize their payoffs. The payoff is, by definition, the "realized" utility of the spectrum.

Definition 14 ***Payoff*** *denoted by p_i^f $\forall i \in \mathcal{I}, \forall f \in F_i$ is given as $p_i^f = u_i - c_i^f$, where c_i^f stands for the difference of expenses and incomes paid for frequency unit f. Note that p_i^f is the maximal defense bid of i on f.*

If a node does not succeed in allocating the required frequency band with the interference constraints fulfilled, its utility becomes 0. In order to assure that the rationality condition always holds, *i.e.*, $p_i^f \geq 0$ $\forall i \in \mathcal{I}$ and $\forall f \in F_i$, prior excluding costs should be paid back to excluded nodes. An *explicitly* excluded node i receives c_i^f altogether, directly from the excluding node and from the authority. An *implicitly* excluded node, which cannot afford the necessary buyouts (on all of its frequency bands) in order to decrease the perceived interference from the newcomer nodes, receives the reimbursement of its prior costs c_i^f from the authority $\forall f \in F_i$, and the node is excluded.

We assume therefore that excluded nodes get back their excluding payments from the authority in order to clear their balances. However, we define utilities and costs with respect

to "amortization" in time and by tuning the amortization factor, a desired time-unit spectrum utilization fee can be deducted from the balances by the authority.

11.2.3 Incentive compatibility

Truthful bidding, *i.e.*, incentive compatibility, is a direct consequence of the pricing policy. In the next proposition we state that our pricing rules incite truthful bidding at selfish nodes.

Proposition 10 *Nodes bid with their real utility (u) for spectrum in DDSA.*

Proof: At a possible buyout, the actual owner node i and the interested parti(es) (let j be the one with the highest utility) are going to play a second-price auction. Let c_i^f denote the price that i paid for the frequency unit f, subject to the auction. While the authority knows the utility of i based on its former bids, j does not know it. If j therefore decides to bid beyond its u_j with $u'_j > u_j$, and if $u'_j > u_i - c_i^f > u_j$, then the fee to pay to the authority results in a negative payoff. Therefore, the well-known truthfulness property of second-price auctions [120] makes newcomers bid their true utilities as dominant strategies at the exclusion attempts. ■

Notice, that the second-price auction might be replaced by a type of sequential first-price auction if the interested buyer j does not intend to reveal its utility u_j. In this case j bids on the given frequency band by incrementing its bet in each iteration. This process results in an outcome similar to that of the second-price auction: the node holding the highest utility gets the resource approximately on the second highest utility.

11.2.4 Fairness and efficiency

We show that nodes causing high interference and/or being sensitive to interference are punished with high costs. Therefore this approach leads to an efficient spectrum allocation, where frequency bands are allocated to the most valuable leasers at the highest possible price. We hereby present a proposition that is in line with the efficiency consequences of central allocation mechanisms, applied in other works, *e.g.*, [80].

Definition 15 ***Interference-friendliness*** *Let us call node i interference-friendly, if its interference tolerance is high, moreover if the interference that is caused by i to every other node is low. Formally, i is more interference-friendly than k, if* $\alpha_i^f > \alpha_k^f$ *and* $\omega_{ij}^f < \omega_{kj}^f$ *and* $\omega_{ji}^f < \omega_{jk}^f \; \forall j \in \mathcal{I} \setminus i, k$.

Proposition 11 *Less interference-friendly nodes pay relatively more for the spectrum.*

Proof: Let us examine how each one of two nodes i and k holding the same utility would allocate a given frequency unit f at the same time, at the same geographic location. Let us

suppose $\alpha_i^f > \alpha_k^f$ and $\sum_{j\in\mathcal{I}} \omega_{ij}^f < \sum_{j\in\mathcal{I}} \omega_{kj}^f$. Our assumption is that k gets f, and we show that in this case i can also get f at a lower price. Based on the assumption k is only disturbed by the interference of other nodes up to α_k^f, thus the quality of f would be sufficient for i as well. Since k paid the possible exclusions of other nodes in order to decrease the interference level below the threshold from its utility u_k, this could be also done by i, since $u_i = u_k$. Also, the interference that k causes to others is higher then i would cause, therefore i could resist to even less intensive re-allocation attempts of other nodes than k does. ■

The important implication of our utility-based allocation and pricing framework is that it ensures fairness despite the fact that exclusions are only unidirectional. Those nodes that cause high interference must hold high defense bids because interfered nodes try to buy them out. The allocation mechanism assures that nodes with high utility may get interference-free frequency while they cause high interference for other nodes. In this sense, the one-way exclusion policy offers results similar to the VCG mechanism applied in [79,80]: service providers that cause high interference are punished with high payables, and frequency bands are allocated to the most valuable leasers at the highest possible price.

In the model of [80], a central entity performs the spectrum allocation by focusing on different aspects, such as efficiency and profit, and the same agent determines the pricing based on the VCG mechanism. Since we consider a distributed mechanism, our framework does not require the costly steps of calculating feasible allocation settings and the implied prices, as implemented in [80]. Instead, distributed heuristic allocation is carried out through iterations of node interactions in our model. Before delving into the evaluation, we provide the algorithmic steps that every node executes in our framework, furthermore we show related results and issues.

11.3 Node exclusion strategies and their consequences

The sequence of node "arrivals" has critical importance. Although if a node is excluded from a frequency band, it can retry at an other one or even on the same when its excluder vanishes, here we show that node characteristics predetermine the success of its spectrum allocation. Selecting a frequency band to allocate and then the disturbing nodes to exclude is not straightforward, in this section we evaluate the effects of sequential node arrivals on our proposed model from the node exclusion perspective and show its important properties.

11.3.1 Node exclusion problem

When newcomer node i arrives, by policy, at first it excludes the nodes that cause interference to it, if necessary. These latter must be bought-out one-by-one by bidding over their defense bids for the given frequency slot. We define two notations, that will be frequently used in the following.

Definition 16 ***Node roles*** *On frequency slot f we define the set of*

- disturbing nodes *as the group of nodes whose exclusion by newcomer node i is necessary in order to assure that the cumulative interference on f is kept below α_i^f: $\mathcal{D}_i^f \subseteq \mathcal{F}^f$ such that $\sum_{j \in \mathcal{F}^f \setminus \mathcal{D}_i^f} \omega_{ji}^f \leq \alpha_i^f$;*
- excluding nodes *as a group of nodes from which exclusion attempts are expected because their perceived interference, increased by newcomer node i, is higher than their tolerance levels: $\mathcal{E}_i^f \subseteq \mathcal{F}^f \setminus \mathcal{D}_i^f$ such that for $\mathcal{C}_i^f = \mathcal{F}^f \setminus \mathcal{D}_i^f \setminus \mathcal{E}_i^f$ (called as the set of* coexistent nodes*) $\forall k \in \mathcal{C}_i^f \sum_{j \in \mathcal{C}_i^f} \omega_{jk}^f \leq \alpha_k^f$ and $\forall k \in \mathcal{E}_i^f \sum_{j \in \mathcal{C}_i^f} \omega_{jk}^f > \alpha_k^f$.*

Figure 11.1 describes the relation of the defined sets to one another.

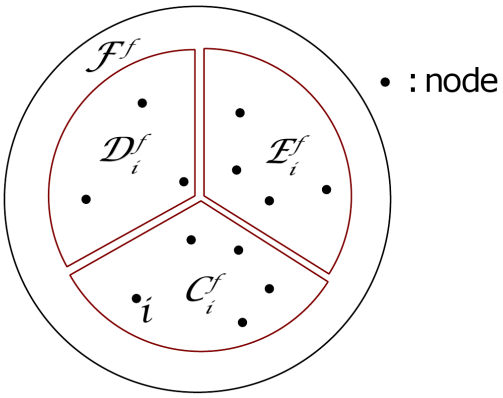


Figure 11.1: Sets $\mathcal{F}^f$, $\mathcal{C}_i^f$, $\mathcal{D}_i^f$ and $\mathcal{E}_i^f$

The intuition behind the system design which implies the aforementioned policy, *i.e.*, at first, it is the newcomer that needs to exclude interfering nodes then the remaining active nodes may perform their attacks against the newcomer, is based on the fact that attacking other nodes with exclusion bids causes high monetary costs for a node. While the sum of exclusion bids are subtracted from the budget, the defense bid is intact when resisting to an exclusion attempt. Therefore we make the case of favoring the nodes already present on the spectrum by this reasonable policy choice.

The newcomer supposedly strives to pick exclusion targets by minimizing exclusion costs, while excluding them alleviates the cumulative interference to a bearable value. We prove that the problem of selecting an optimal set of nodes that cause excess interference for node i on a given frequency slot f is hard.

Proposition 12 *Optimizing the set of disturbing nodes is an NP-complete problem.*

Proof: The problem is formulated as follows. For given $\{\omega_{ji}^f, u_j\}$ $\forall j \in \mathcal{I}$ and for α_i^f, u, is there any $\mathcal{D}_i^f \subseteq \mathcal{I}$ which satisfies $\sum_{j \in \mathcal{D}_i^f} \omega_{ji}^f \geq \sum_{j \in \mathcal{I}} \omega_{ji}^f - \alpha_i^f$ and $\sum_{j \in \mathcal{D}_i^f} u_j \leq u$? This is equivalent to the knapsack problem, which is known to be NP-complete [61]. ■

By policy, after the newcomer has excluded the necessary number of adversaries, the remaining nodes may attempt to exclude the newcomer, if it violates their interference thresholds. Supposing that the newcomer could choose the set of nodes that may make exclusion attempts on it, the problem of finding the optimal set would be solvable. Under the notion of "optimal", we mean the set of nodes, against which the defensive bid of the newcomer is minimal.

Proposition 13 *Optimizing the set of excluding nodes is solvable in polynomial time.*

Proof: The problem is formulated as follows: for given $\{\omega_{jk}^f, u_j, \alpha_j^f\}$ $\forall j,k \in \mathcal{I}$ and u, is there any $\mathcal{E}_i^f \subseteq \mathcal{I}$ which satisfies $\sum_{j\in\mathcal{E}_i^f} \omega_{jk}^f \geq \sum_{j\in\mathcal{I}} \omega_{jk}^f - \alpha_k^f$ $\forall k \in \mathcal{I} \setminus \mathcal{E}_i^f$ and $\max_{j\in\mathcal{E}_i^f} u_j \leq u$? This can be solved simply by adding the nodes for which $u_j \leq u$ into $\mathcal{E}_i^f$, and checking if the first condition holds. ■

However, the above solution lacks the perspective of the excluding nodes: nodes that might not be disturbed by the newcomer are supposed to be excluded, therefore possibly unnecessary implicit exclusions may happen. In reality, only those nodes will attempt to exclude the newcomer, that are members of the set of *excluding nodes*.

Conjecture 1 *Finding $\mathcal{E}_i^f$ such that for $\mathcal{C}_i^f$ ($\mathcal{C}_i^f = \mathcal{F}^f \setminus \mathcal{D}_i^f \setminus \mathcal{E}_i^f$) $\forall k \in \mathcal{C}_i^f \sum_{j\in\mathcal{C}_i^f} \omega_{jk}^f \leq \alpha_k^f$ and $\forall k \in \mathcal{E}_i^f \sum_{j\in\mathcal{C}_i^f} \omega_{jk}^f > \alpha_k^f$ with the minimum value of $\max_{j\in\mathcal{E}_i^f} u_j$ is suspected to be a hard problem.*

Hindered by the difficulty of the exclusion problem, we propose the following policy choice. Exclusion of nodes is carried out in the increasing order of their "interference prices", *i.e.*, for newcomer node i: $\forall j \in \mathcal{D}_i^f$ and $\forall k \notin \mathcal{D}_i^f$ $\frac{u_j - c_j^f}{\omega_{ji}^f} \leq \frac{u_k - c_k^f}{\omega_{ki}^f}$. Furthermore, we propose that exclusion attempts, targeting the newcomer, should arrive from nodes in the increasing order of their interference thresholds, *i.e.*, $\forall j \in \mathcal{E}_i^f$ and $\forall k \notin \mathcal{E}_i^f$ $\alpha_j^f \leq \alpha_k^f$. In case of equality, by policy, those nodes are included in the sets that cause higher interference or arrived sooner respectively.

11.3.2 Insights about exclusions in a simplified scenario

In this section we present important considerations about node exclusions. First, we refer to the above findings on the efficient resource allocation. As a reminder, we call a node i more interference-friendly than j, if i tolerates interference more than j, moreover the interference that is caused by i to every other node is lower than that of j.

Building on the interference-friendliness, and its effect on the necessary utility, for analytical tractability, we make the following assumption about the reduction of parameter space. The assumed case restricts the analysis for nodes that operate at the same geographic location and having the same "technology couplings" [79].

Assumption 11.3.1 *The values of interference a given node causes to others are the same for all nodes ($\omega_{ij}^f \equiv \omega_i^f$ $\forall i,j \in \mathcal{I}$) and we assume increasing utilities with increasing level of caused*

interference ($u_i = const_1 \omega_i^f$) *and decreasing level of tolerance* ($u_i = const_2 \frac{1}{\alpha_i^f}\ \forall i$). *We suppose that* $const_1 = 1$ *and* $const_2 = \min_{i \in \mathcal{I}} {\alpha_i^f}^2$, *setting* $u_i = \omega_i^f = \frac{\min_{i \in \mathcal{I}} {\alpha_i^f}^2}{\alpha_i^f}$.

Given the above assumption, we make the following claim about the nodes that will engage in exclusions with newcomers. Let us denote the newcomer node as *new*, and the node with the minimum interference threshold present on the investigated frequency slot f before the arrival of *new* by *min*.

Proposition 14 *When nodes that allocate a given frequency slot* f *are described by Assumption 11.3.1, nodes min and new are always involved if exclusion happens.*

Proof: We write the following formulas by including *new* into $\mathcal{F}^f$. We distinguish two possible cases.

When $\alpha_{new}^f \geq \alpha_{min}^f$, exclusion is needed if $\sum_{i \in \mathcal{F}^f} \frac{\text{const}_2}{\alpha_i^f} > \alpha_{min}^f + \frac{\text{const}_2}{\alpha_{min}^f}$ while $\sum_{i \in \mathcal{F} \setminus \{new\}} \frac{\text{const}_2}{\alpha_i^f} \leq \alpha_{min}^f + \frac{\text{const}_2}{\alpha_{min}^f}$. In this case node *min* is the only member of $\mathcal{D}_{new}^f \cup \mathcal{E}_{new}^f$ since $\alpha_{min}^f + \frac{\text{const}_2}{\alpha_{min}^f}$ is strictly increasing in α_{min}^f (if $\text{const}_2 = \min_{i \in \mathcal{I}} {\alpha_i^f}^2$).

When $\alpha_{new}^f < \alpha_{min}^f$ the proposition follows from assumption and the previous reasoning, however, *min* may not be the only member of $\mathcal{D}_{new}^f \cup \mathcal{E}_{new}^f$. ■

Therefore, the possible outcomes of the arrival of *new* are:

- no exclusion happens;
- old node *min* excludes *new*;
- old node *min* is implicitly excluded, because $u_{min} - c_{min}$ is too low to exclude *new*;
- old node(s) holding the lowest interference threshold(s) are excluded by *new* (however *new* may be implicitly excluded by an other node than *min* if the budget of *new* is exhausted during the exclusions).

In order to reflect the difficulty of the underlying problem, here we show that Proposition 12 holds even in a simple scenario described by Assumption 11.3.1 if we do not follow the policy choice proposed above about selecting nodes to $\mathcal{D}_i^f$.

Proposition 15 *Optimizing the set of disturbing nodes described by Assumption 11.3.1 is an NP-complete problem.*

Proof: The problem transforms to the following: for given $\{\omega_j^f\}\ \forall j \in \mathcal{I}$ and α_i^f, u, is there any $\mathcal{D}_i^f \subseteq \mathcal{I}$ which satisfies $\sum_{j \in \mathcal{D}_i^f} \omega_j^f \geq \sum_{j \in \mathcal{I} \setminus \{i\}} \omega_j^f - \alpha_i^f$ and $\sum_{j \in \mathcal{D}_i^f} \omega_j^f \leq u$? The problem is in NP because it can be demonstrated by a good $\mathcal{D}_i^f$; and it is NP-hard because the subset sum problem can be reduced to it with $u = \sum_{j \in \mathcal{I} \setminus \{i\}} \omega_j^f - \alpha_i^f$ [61]. ■

Seen the complexity of the node exclusion problems, in the following section we keep Assumption 11.3.1.

11.3.3 The saturation of a frequency slot

In this section we show that once a frequency slot is "saturated", *i.e.*, the sum of the interference of active nodes nearly exceeds the tolerance thresholds of some nodes, the high accumulated interference is unlikely to drop below a certain level. This level is the minimum interference threshold among the nodes that have allocated the given frequency slot, denoted by α^f_{min}.

Proposition 16 *After the cumulative interference has reached α^f_{min} on frequency slot f, it will not drop to lower values due solely to exclusions.*

Proof: We prove the claim in an indirect way. Let the cumulative interference be lower than α^f_{min}, after having exceeded it. For the cumulative interference to decrease, it follows that some node i (without the loss of generality) has left the frequency slot. But node i is excluded from the frequency slot only if, as a conservative condition, $\sum_{j\in\mathcal{F}^f}\omega^f_j > \alpha^f_{min}+\omega^f_{min}$ where $\mathcal{F}^f$ still includes i, and ω^f_{min} is the interference caused by the node having α^f_{min}. However, we know that the cumulative interference after the exclusion is lower than α^f_{min}, so

$$\alpha^f_{min} > \sum_{j\in\mathcal{F}^f}\omega^f_j - \omega^f_i > \alpha^f_{min}+\omega^f_{min}-\omega^f_i \geq \alpha^f_{min},$$

thus arriving at contradiction. ■

The following proposition about a saturated frequency slot state follows from the previous claim.

Proposition 17 *After the frequency slot gets "saturated" with nodes that are all more frequency-friendly than a node with α^f_{min}, later on no more than one such node with α^f_{min} can allocate the frequency slot simultaneously.*

Proof: We give an indirect proof. We make the assumption that there is exactly one node with α^f_{min} (the least frequency-friendly node) present on the saturated frequency slot f, then an other one arrives. For a seamless allocation $\sum_{j\in\mathcal{F}^f}\omega^f_j < \alpha^f_{min}+\omega^f_{min}$ must hold with $\mathcal{F}^f$ already containing the second node with α^f_{min}, where ω^f_{min} is the interference caused by a node having α^f_{min}. Before the arrival of this latter, the left side of the inequality was lower by ω^f_{min}, therefore the inequality $\sum_{j\in\mathcal{F}^{f\prime}}\omega^f_j < \alpha^f_{min}$ stood for $\mathcal{F}^{f\prime}=\mathcal{F}^f\setminus\{min\}$. However, the frequency slot was supposedly "saturated", thus contradicting with Proposition 16. ■

Building on the observations about saturating frequency slots, in the next section we present our estimations for the *expected lifetime* of nodes with different characteristics, under Assumption 11.3.1. The expected lifetime, *i.e.*, the estimated number of new node arrivals that a given node can "survive" by successfully allocating its frequency band, is considered to be an important metric, since it gives an estimate on the duration of the frequency allocation with positive payoff. While selfish nodes are supposed to be interested about the least costly allocation, they also concerned about the duration they can benefit of an advantageous allocation.

11.3.4 Queuing model of a frequency slot

In the following, we focus on node arrivals on a particular frequency slot f and we determine the expected lifetime of the newcomers after their successful allocation.

We denote the actual newcomer node that successfully allocates f as *new*. Based on Proposition 14 and our policy choice, *new* is able to operate as long as the present less frequency-friendly nodes and the equally frequency-friendly, but older nodes are not excluded. These latter remain active on f until they are able to exclude interfering newcomers. Therefore, *new* stays active on f on average for a duration proportional (assuming steady rate of node arrivals) to

$$\mathbb{E}(D) = \frac{t \sum_{j \in \mathcal{F}, \alpha_j^f < \alpha_i^f} (u_j - c_j) + \min_{j \in \mathcal{F}} \alpha_j^f - \sum_{j \in \mathcal{F}} \omega_j^f}{\bar{\omega^f}},$$

where t is a constant describing the average interference value that can be excluded by investing a monetary unit ($t \approx 1$ if Assumption 11.3.1 holds), the multiplied term is the sum of the budgets of nodes that are equally or less frequency-friendly than i; $\min_{j \in \mathcal{F}} \alpha_j^f - \sum_{j \in \mathcal{F}} \omega_j^f$ is the amount of further cumulative interference that the actual nodes can tolerate on f at the arrival of i; and $\bar{\omega^f}$ is the average interference of the ulterior newcomers. The first term in the nominator describes the number of arrivals i survives, the second and third terms extend it by the actual cumulative interference gap, and the denominator transforms the result into the expected arrivals that i will last.

In order to give estimates on node lifetimes in further details, we define the following queuing model:

Definition 17 ***Queuing model*** *Let $\mathcal{Y}$ be the following queuing system:*

- *nodes arrive in separated queues according to their types;*
- *a node arrives to the queuing system if successfully allocates a frequency slot;*
- *a node is served if it has to leave the frequency slot, because it is excluded (explicitly or implicitly).*

$\mathcal{Y}$ is a First-In-First-Out *(FIFO) queuing system if among nodes of the same type the older one is involved in exclusions with the newcomer first. Furthermore, $\mathcal{Y}$ is a* priority *queuing system, if more interference-friendly nodes make bids and are excluded sooner than less interference-friendly nodes.*

We consider f as a *FIFO priority* queuing system. We categorize nodes, described by Assumption 11.3.1, into types, based on their parameters. We refer as z to a category $\mathcal{Z}$ and for $z, z' : z < z'$ if $\omega_i^f > \omega_j^f \ \forall i \in \mathcal{Z}$ and $\forall j \in \mathcal{Z}'$. We also use this reference for the parameters $\omega_z^f, \alpha_z^f, u_z$ of the nodes of a given type z. Let S_z be the service time, and let n_z be the first node

(*i.e.*, the node being in the queue for the longest duration) in the queue of nodes with type z. Based on Definition 17, if queues lower than z are all empty, then node n_z is under service. We derive the service time S_z for nodes of type z.

Proposition 18 *The service time of n_z without preemption is*

$$S_z = \min(\min_{j<z}(W_z(j) + R_j), S_z^+),$$

where $W_z(j)$ is the number of newcomers of types at least z with whom the cumulative interference reaches the tolerance level of node type j; R_j is the number of node arrivals until the arrival of node with type j (not necessarily resulting in successful allocation); and S_z^+ is the service time if every newcomer is more interference-friendly than n_z. Moreover,

$$\mathbb{E}(W_z(j)) = \frac{\alpha_j^f - \omega^f}{\overline{\omega^f}_z},$$

where $\omega^f = \sum_{j \in \mathcal{F}^f} \omega_j^f$ at the beginning of the service period of n_z, and $\overline{\omega^f}_z$ is the average interference of newcomers with $\alpha_{new}^f \geq \alpha_z^f$. R_j depends on the distribution of node types in the system and the frequency band selection strategies.

Proof: If newcomers, that arrive after the beginning of the service period of n_z, are more interference-friendly than n_z ($\alpha_{new}^f \geq \alpha_z^f$), the newcomers seamlessly allocate f as long as $\sum_{j \in \mathcal{F}^f} \omega_j^f \leq \alpha_z^f + \omega_z^f$ ($new \in \mathcal{F}^f$). Otherwise n_z attempts to exclude new, therefore

$$\mathbb{E}(S_z^+) = \frac{t(u_z - c_z^f) + \alpha_z^f - \omega^f}{\overline{\omega^f}_z},$$

where c_z^f is the average cost of a node of type z after it allocated f.

If a newcomer is less interference-friendly than n_z ($\alpha_{new}^f < \alpha_z^f$), and $\sum_{j \in \mathcal{F}^f} \omega_j^f \leq \alpha_{new}^f + \omega_{new}^f$, then the service of n_z is preempted. On the other hand, if $\sum_{j \in \mathcal{F}^f} \omega_j^f > \alpha_{new}^f + \omega_{new}^f$, then the new excludes n_z and the service of n_z is over. ■

Note that if every newcomer chooses the frequency slot f with the highest $\mathbb{E}(D^f)$, then f will see more frequent arrivals, thus decreasing $\mathbb{E}(D^f)$, *i.e.*, its attractiveness. Based on the findings presented above, with higher α_{new}^f, $\mathbb{E}(D_{new}^f)$ is greater. Therefore, statistically, less interference-friendly nodes are excluded sooner than interference-friendly ones, provided that they can successfully allocate the frequency slot. In the following section we discuss the importance of frequency band selection, and we present our algorithm with the proposed heuristics that are evaluated numerically in Chapter 12.

11.4 Frequency band selection algorithms

In this section first, we provide the pseudo-code of the frequency band selection algorithm that the selfish nodes would perform, second, we give possible heuristics to apply, and third, we outline the implemented algorithms.

11.4.1 The frequency band selection and node-exclusion algorithm

Newcomer selfish nodes strive to allocate the required size frequency band by spending the minimum on occurring costs for the maximal expected lifetime. Each node i may perform exclusions up to its budget of u_i on each of its frequency slots, but one should maintain sufficient defense bid against exclusion attempts. At exclusion, in the worst case node i needs to set its bid b_i at the target node j to u_j. This is the case when node j has not excluded other nodes earlier. As the number of nodes that should be excluded in order to fulfill the interference requirements of i grows, its competitiveness worsens, as $u_i - c_i$ falls.

In order to find the cheapest frequency band, newcomer i tries to position its F_i on the spectrum, so that, on the one hand, the cost of exclusion of other nodes ($\mathcal{D}_i^f$) would be minimal; on the other hand the cost of defense against excluding nodes ($\mathcal{E}_i^f$) would cost the least possible on the average of all the frequency slots of F_i. In the following proposition we give a necessary condition for successful allocation.

Proposition 19 *Node i is able to allocate an adequate spectrum band if $\exists F_i : |F_i| = q_i$ and $\forall f \in F_i$:*

$$u_i \geq \sum_{j \in \mathcal{D}_i^{f*}} \left(u_j - c_j^f\right) + \max_{j \in \mathcal{E}_i^{f*}} \left(u_j - c_j^f\right), \textit{where}$$

$$\mathcal{D}_i^{f*} = \arg\min_{\mathcal{D}_i^f} \sum_{j \in \mathcal{D}_i^f} \left(u_j - c_j^f\right) \textit{ and } \mathcal{E}_i^{f*} = \arg\min_{\mathcal{E}_i^f} \max_{j \in \mathcal{E}_i^f} \left(u_j - c_j^f\right),$$

provided that the node exclusion policy requires the newcomer to exclude first the interfering nodes, then the old nodes may make attempts to exclude the newcomer if needed.

Proof: The proposition states that if node i has higher frequency slot utility than the sum of two terms: the minimal aggregated cost of excluding a set of nodes that interfere with i (first term on the right side of the inequality) and the necessary defense bid to implicitly exclude the set of remaining nodes that are disturbed by i (second term on the right side of the inequality), then node i can surely acquire a suitable frequency band. The necessity of the condition is ensured by minimizing the exclusion cost and defense bid through the optimal choice of sets $\mathcal{D}_i^{f*}$ and $\mathcal{E}_i^{f*}$. The condition also provides sufficiency, in case the set of excluding nodes can be chosen by i, since the first term assures that node i can exclude others if necessary, and the second term prevents any node in set $\mathcal{E}_i^{f*}$ from excluding node i. ■

If a newcomer considers only the actual costs of a successful allocation, then in order to find optimality, it performs the frequency band selection algorithm described by its pseudo-code in Algorithm 6. In Line 2, the newcomer computes the incurred exclusion costs (in Line 5) and expected exclusion attempts (in Line 8) on each frequency slot (Line 3) of every adequate size frequency band. After the optimization has found the node sets $\mathcal{D}_i^{f*}, \mathcal{E}_i^{f*}$, with which node i must engage in exclusions, on all frequency slot units, in Line 12 the algorithm finds the cheapest frequency band. Provided that node the utility of i, which is homogeneously distributed over its frequency slots, *i.e.*, $u_i^f = u_i^{f'}\ \forall f, f' \in F_i$, is sufficient on each frequency slot of band F_i^*, i successfully allocates F_i^* (Line 14).

If such a F_i^* does not exist (Line 16), the newcomer node cannot allocate the required band, thus it is excluded from the spectrum. However, by design, it attempts to exclude nodes j in the increasing order of their "interference prices" ($\frac{u_j - c_j}{\omega_{ji}^f}$), if the frequency slot is over-interfered for i. Furthermore, if i succeeds to lower its perceived interference level below its threshold α_i^f, i is assumed to resist exclusion attempts in the increasing order of α_j^f of old nodes, thus implicitly excluding them or raising their costs until i is definitely bought out on every frequency slot.

Algorithm 6 Channel-selection and node-exclusion algorithm

1: *newcomer node* i: q_i, u_i, α_i^f
2: **for all** $F_i \in F$ such that $|F_i| = q_i$ **do**
3: **for all** $f \in F_i$ **do**
4: **for all** $\mathcal{D}_i^f$ **do**
5: $\mathcal{D}_i^{f*} = \arg\min_{\mathcal{D}_i^f} \sum_{j \in \mathcal{D}_i^f} \left(u_j - c_j^f\right)$
6: **end for**
7: **for all** $\mathcal{E}_i^f$ **do**
8: $\mathcal{E}_i^{f*} = \arg\min_{\mathcal{E}_i^f} \max_{j \in \mathcal{E}_i^f} \left(u_j - c_j^f\right)$
9: **end for**
10: **end for**
11: **end for**
12: $F_i^* = \arg\min_{F_i} \sum_{f \in F_i} \left(\sum_{j \in \mathcal{D}_i^{f*}} \left(u_j - c_j^f\right) + \max_{j \in \mathcal{E}_i^{f*}} \left(u_j - c_j^f\right)\right)$
13: **if** $\forall f \in F_i^*\ u_i \geq \sum_{j \in \mathcal{D}_i^{f*}} \left(u_j - c_j^f\right) + \max_{j \in \mathcal{E}_i^{f*}} \left(u_j - c_j^f\right)$ **then**
14: successful allocation: i buys out nodes in $\mathcal{D}_i^{f*}$, and implicitly excludes nodes in $\mathcal{E}_i^{f*}\ \forall f \in F_i^*$
15: **else**
16: unsuccessful allocation: i buys out subset of nodes in $\mathcal{D}_i^{f*}$, and implicitly excludes subset of nodes in $\mathcal{E}_i^{f*}\ \forall f \in F_i^*$
17: **end if**

11.4.2 Optimization heuristics

The complexity of finding the optimal frequency band allocation drives our attention to potential heuristics. The difficulty stems from first, the NP-completeness of the underlying node exclusion problem in Line 5 of Algorithm 6; second, from the suspected hardness of determining the minimal defense bid against excluding nodes, third, from the fact that optimality does not necessarily confined to allocating the required frequency band for the least possible cost, but nodes may be also concerned about future outcomes due to further node arrivals, specifically about estimated future costs of exclusions and the expected lifetime. This latter transforms the frequency band selection to a *stochastic optimization* problem.

In order to overcome these issues and to keep the algorithm runtime low, finding the least costly frequency band and excluding other nodes, *i.e.*, minimizing own excluding costs and the defense bid (by assigning the excluding nodes) in Lines 5 and 8 of Algorithm 6 respectively, are carried out by the aforementioned "greedy" policies. The greedy method targets interfering nodes in the increasing order of their "interference prices", *i.e.*, $\frac{u_j - c_j}{\omega_{ji}^f}$, then the excluding nodes file their bids in the increasing order of their thresholds, *i.e.*, α_j^f.

Given these rules, one can determine the feasibility and the cost of the allocation of a given frequency band. However, the stochastic optimization might still require counter-intuitive frequency band selection. In the following we discuss our propositions about heuristics for this latter. We set the results of the *cost-minimizing* frequency band selection strategy as reference for comparison with the *interference-aware* and the *deliberate* heuristics.

Definition 18 ***Frequency band selection*** *Node i attempts to allocate F_i^*. We define a frequency band selection strategy cost-minimizing, if*

$$F_i^* = \arg\min_{F_i} \sum_{f \in F_i} \left(\sum_{j \in \mathcal{D}_i^{f*}} \left(u_j - c_j^f \right) + \max_{j \in \mathcal{E}_i^{f*}} \left(u_j - c_j^f \right) \right);$$

interference-aware, if

$$F_i^* = \arg\min_{F_i} \sum_{f \in F_i} \sum_{j \in \mathcal{F}f} \omega_{ji}^f;$$

deliberate, if

$$F_i^* = \arg\min_{F_i} \sum_{f \in F_i} \sum_{j \in \mathcal{F}f} (\omega_{ji}^f - (u_j - c_j)).$$

The cost-minimizing strategy runs a simple minimum search with the expected cost numbers that the above node selection heuristics provide. The interference-aware strategy takes into account *only* the inferred cost of excluding interfering nodes by considering the *actual* aggregate interference. This is motivated by the relatively costly exclusions. On the other hand, the deliberate strategy calculates both on the aggregate interference *and* the remaining budget of the other nodes, which is reasoned by the findings in Chapter 12.

11.4.3 Implemented heuristic algorithms

In this section we provide the pseudo-code of the algorithms that we developed, and we show the feasibility and simplicity of the proposed heuristics.

While our framework aims at maximizing the utilization of radio spectrum, the allocation is optimized by the nodes. For a brute-force exhaustive search, the frequency band selection and node-exclusion algorithm requires node i to evaluate exponential number (in $|\mathcal{D}_i \cup \mathcal{E}_i|$) of values, where $|\mathcal{D}_i|$ (resp. $|\mathcal{E}_i|$) denotes the cardinality of the group consisting such node js for which $\omega_{ji}^f > 0$ (resp. $\omega_{ij}^f > 0$). However, following the node selection heuristics and sorting node js based on their interference prices on each frequency slot, *i.e.*, $\frac{u_j - c_j^f}{\omega_{ji}^f}$ (resp. based on their interference tolerance levels α_j^f), the algorithm complexity reduces significantly to $o(|F||\mathcal{D}_i||\mathcal{E}_i| \log(|\mathcal{D}_i| + |\mathcal{E}_i|))$, where $|F|$ stands for the number of frequency slots on the spectrum. Furthermore, the proposed band-selection heuristics (interference-aware and deliberate) are down to $o(|F||\mathcal{D}_i|)$.

While the central authority is relieved from the burden of computation-intensive allocation and pricing algorithms, it is still needed in our framework for monitoring and accounting purposes. The communication overhead related to our algorithm consists of the allocation (F_i), the interference (ω_{ij}^f from any node i to any node j) and the actual channel utilities. This information requires maintenance and updates after every change in the allocation and node set respectively, and should be also closely observed by the authority.

Generally, in cases where selfish decisions drive the system to a stable outcome, this latter is suboptimal compared to the result of a central allocation based on full information. This difference is called the *price of anarchy*, which surely appears in the context of our distributed scheme as well. Since the main goal of our framework is to exploit the spectrum with the highest utility, we do not intend to find the allocation and pricing which maximizes the aggregate user payoff. However, the *price of anarchy* may appear in the context of the authority income. Hindered by the analytical complexity of the evaluation of the framework, we provide numerical results in the following chapter.

Chapter 12

Evaluation

In order to evaluate our framework under practical settings, we built a custom MATLAB simulator and performed case studies of our model for a scenario conform to Assumption 11.3.1 and for a more realistic one. As evaluation, we compared the outcome of the simulations applying the different frequency band selection strategies, presented in Section 11.4. In this section we describe the setting of the numerical evaluations, the metrics of our investigation and the results.

12.1 Simulation setting

In the simulation we assume that nodes arrives sequentially with their allocation demands and each of them executes Algorithm 7 in the respective order of their arrivals. Node arrivals are organized into rounds: only one node arrives in a round and may perform simple buyouts for exclusions. After the newcomer node has finished its required exclusions, the other nodes may carry out their exclusion attempts on the newcomer, if necessary. Since the sequence of node arrivals highly affects the final allocation outcome, this latter is presented statistically for multiple randomly generated arrival sequences. The algorithm takes the interference matrix, describing interference relations for every possible node combination, as input. Each node i, *i.e.*, a given service provider in a given area, is characterized by: the required frequency band size q_i, its utility u_i per frequency slot, moreover the associated costs c_i^f it has paid for exclusion of interfering nodes on frequency slot f. We do not account for dynamics in node utilities.

In the first scenario, the restrictive Assumption 11.3.1 holds: nodes operate on the same geographic area, and each node causes the same interference on the other nodes. Moreover, the interference threshold parameters are inversely proportional, and node utility values are proportional to the caused interference for each node. Here, we suppose that every node competes for only one frequency slot ($q_i = 1\ \forall i \in \mathcal{I}$), we assume 10 different types of nodes and 10 nodes from each type. The size of the spectrum ($|F|$) is 5 frequency slots. The applied node parameters are shown in Table 12.1: Type 1 nodes are the least interference-friendly nodes, but also holding

Algorithm 7 Implemented heuristic algorithm for spectrum allocation

newcomer node i: q_i, u_i, α_i^f
for all $F_i \in F$ such that $|F_i| = q_i$ **do**
  for all $f \in F_i$ **do**
    $\mathcal{D}_i^{f*} = \{j : \sum_{k \notin \mathcal{D}_i^{f*}} \omega_{ki}^f \leq \alpha_i^f, \frac{u_j - c_j^f}{\omega_{ji}^f} \leq \frac{u_k - c_k^f}{\omega_{ki}^f} \forall k \notin \mathcal{D}_i^{f*}\}$
    $\mathcal{E}_i^{f*} = \{j : \sum_{k \notin \mathcal{E}_i^{f*}} \omega_{kj}^f > \alpha_j^f, \alpha_j^f \leq \alpha_k^f \forall k \notin \mathcal{E}_i^{f*}\}$
  end for
end for
if cost-minimizing **then**
  $F_i^* = \arg\min_{F_i} \sum_{f \in F_i} \left(\sum_{j \in \mathcal{D}_i^{f*}} \left(u_j - c_j^f \right) + \max_{j \in \mathcal{E}_i^{f*}} \left(u_j - c_j^f \right) \right)$
else if interference-aware **then**
  $F_i^* = \arg\min_{F_i} \sum_{f \in F_i} \sum_{j \in \mathcal{F}^f} \omega_{ji}^f$
else if deliberate **then**
  $F_i^* = \arg\min_{F_i} \sum_{f \in F_i} \sum_{j \in \mathcal{F}^f} (\omega_{ji}^f - (u_j - c_j))$
end if
if $\forall f \in F_i^*$ $u_i \geq \sum_{j \in \mathcal{D}_i^{f*}} \left(u_j - c_j^f \right) + \max_{j \in \mathcal{E}_i^{f*}} \left(u_j - c_j^f \right)$ **then**
  successful allocation: i buys out nodes in $\mathcal{D}_i^{f*}$, and implicitly excludes nodes in $\mathcal{E}_i^{f*}$ $\forall f \in F_i^*$
else
  unsuccessful allocation: i buys out subset of nodes in $\mathcal{D}_i^{f*}$, and implicitly excludes subset of nodes in $\mathcal{E}_i^{f*}$ $\forall f \in F_i^*$
end if

the highest utilities.

	α	ω	u
Type 1	2	0.50	0.50
Type 2	3	0.33	0.33
Type 3	4	0.25	0.25
Type 4	5	0.20	0.20
Type 5	6	0.17	0.17
Type 6	7	0.14	0.14
Type 7	8	0.13	0.13
Type 8	9	0.11	0.11
Type 9	10	0.10	0.10
Type 10	11	0.09	0.09

Table 12.1: Node parameters in the simplified scenario

In the second scenario our simulation setting is inspired by the simplistic example presented in [79]. Nodes are located at the same geographic region, as illustrated on Figure 12.1; the region has the diameter of 20 kms, furthermore we suppose that GSM-like, UMTS-like, UWB-like and DVB-T-like service providers operate in the region. As in the previous example, we strive to analyze the effects of the framework on very interference-friendly (UWB-like), and extremely not interference-friendly (DVB-T-like) nodes.

We emphasize that the following settings are hypothetical and only approximately reflect practical considerations. We provide the geographical ($\epsilon = 640$ within the given area) and technological coupling parameters of [79] (in Table 12.2, the row technologies cause the depicted disturbance to the column technologies); interference parameters are given by these multiplicative values. We propose dividing the spectrum into 5 MHz unit size blocks, therefore the demand of the different service providers are practically as follows: GSM-like 2 frequency slots; UMTS-like 3 frequency slots; UWB-like 4 frequency slots; DVB-T-like 2 frequency slots. We determine the interference thresholds based on the different radio technique characteristics involved in the simulation, thus establishing the set of α (in Table 12.3). We assume that the size of the spectrum to be allocated is $|F| = 20$ frequency slots, and 25 nodes appear from each type. We do not consider the frequency dependence of interference relations between nodes. Node utilities (in Table 12.4) are decided in favor of the performance representation of our framework.

12.2 Evaluation metrics

For each scenario we are interested in the success of spectrum allocation for each type of node and in the cumulative cost that actually active nodes will bear for successful allocation. We

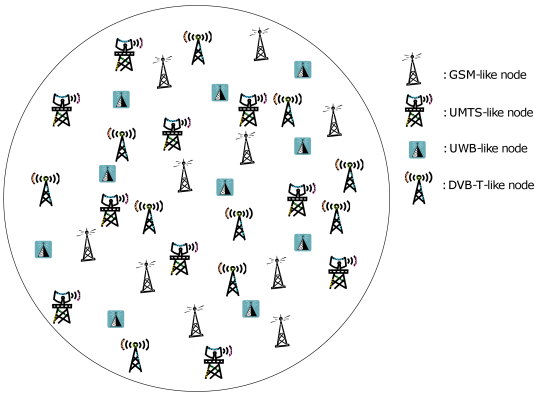


Figure 12.1: Nodes in the second scenario

	GSM-like	UMTS-like	UWB-like	DVB-T-like
GSM-like	0.486	0.010	0.001	0.501
UMTS-like	0.010	0.050	0.001	0.501
UWB-like	0	0	0	0
DVB-T-like	0.100	0.005	0.001	0.794

Table 12.2: Technology coupling between nodes

$\times 10^{-10}$	GSM-like	UMTS-like	UWB-like	DVB-T-like
power	3.09	2.06	0.000001	0.0732
α	0.49	41.1	3.71	0.00421

Table 12.3: Transmission power and interference tolerance thresholds $\left[\frac{mW}{kHz}\right]$

	GSM-like	UMTS-like	UWB-like	DVB-T-like
u	10	10	1	100

Table 12.4: Node utilities per type

directly derive the following results:

- authority income: $\sum_{i \in \mathcal{I}} \sum_{f \in F} c_i^f$, *i.e.*, the sum of fees that the nodes spend on exclusions (we assume that a time-based license fee is introduced by the authority, imposed on the licensees proportionally to these costs);
- average lifetime for each type of node, *i.e.*, the number of arrivals for which a node, given its type, can allocate the required frequency band.

We represent the results statistically for 100 simulation runs by plotting the mean and the standard deviation (as a confidence interval) of the above metrics. The simulation results and discussion about the acquired plots are shown in the next section.

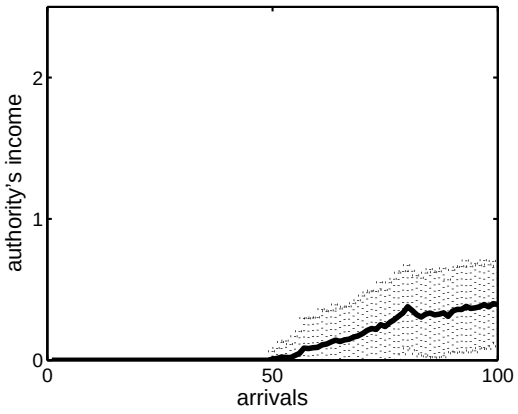
12.3 Simulation results

In the first scenario, *i.e.*, in the simplified setting, the cost-minimizing, interference-aware and deliberate frequency band selection strategies perform as reflected in Figure 12.2.

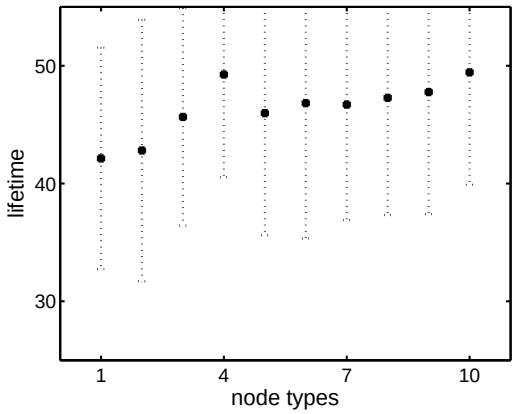
In terms of the authority income, *i.e.*, aggregate exclusion costs of active nodes, the deliberate frequency band selection strategy (Figure 12.2(e)) performs better than the two other strategies. From this observation, it directly follows that selfish nodes would not opt for the usage of this strategy, because of the high costs, unless their expected lifetimes were longer. As it can be seen on Figures 12.2(d) - 12.2(f), this is exactly the case for nodes of type 1, which are the main cost-payers due to their relatively large budget. For type-1 nodes, applying the interference-aware frequency band selection results in possibly lower costs, however shorter lifetimes. On the other hand, with deliberate frequency band selection, they select bands where they achieve longer lifetime by costly exclusion of nodes (most probably nodes of types 2 and 3 based on Figure 12.2(f)).

Comparing the cost-minimizing and interference-aware strategies in the perspective of the authority there is not much difference in income. On the other hand, the perception of nodes differs a lot: because of the failure of type-1 node allocation attempts, the other types of nodes have higher lifetime in case every node implements the interference-aware frequency band selection. After the exclusion of type-1 nodes, newcomers cannot afford to exclude the numerous active nodes of other types, thus they do not fit into the "crowded" frequency slots, explaining the decaying income of the authority.

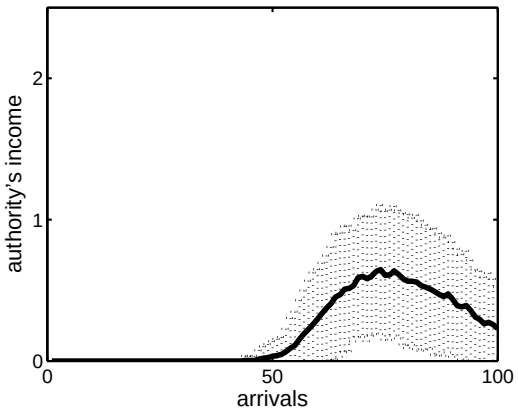
The outcomes in the second scenario are shown in Figure 12.3. Contrary to the results of the first scenario, in this setting the authority income is less and less if nodes switch to interference-aware and deliberate frequency band selection strategies from the cost-minimizing selection. This is due to the failure of type-4 (DVB-T-like) node allocation attempts. Interestingly, while with interference-aware strategy type-1 nodes (GSM-like) may expect longer lifetime (Figure



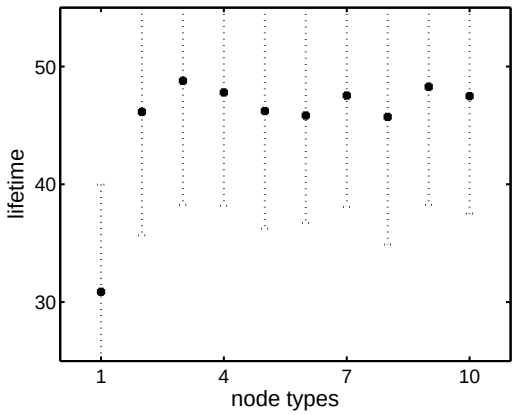
(a) Cost minimizing frequency band selection



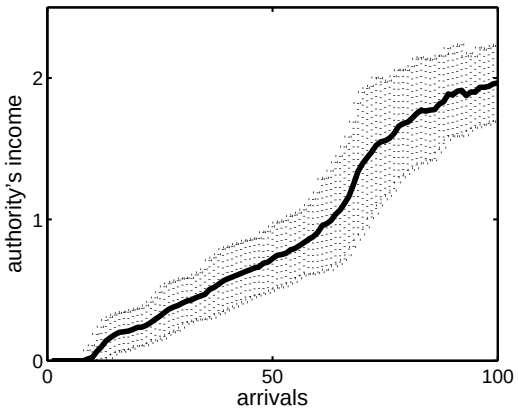
(b) Cost minimizing frequency band selection



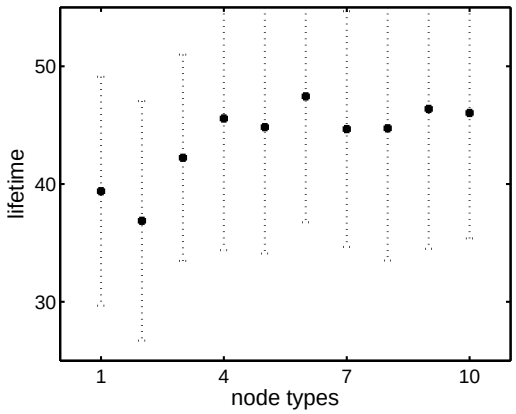
(c) Interference-aware frequency band selection



(d) Interference-aware frequency band selection



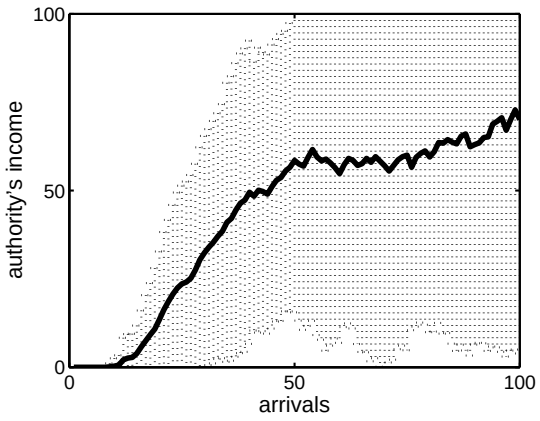
(e) Deliberate frequency band selection



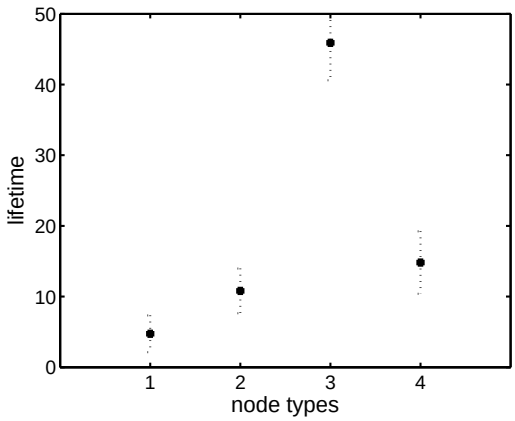
(f) Deliberate frequency band selection

Figure 12.2: Authority income and node lifetimes with different frequency band selection strategies in the first scenario

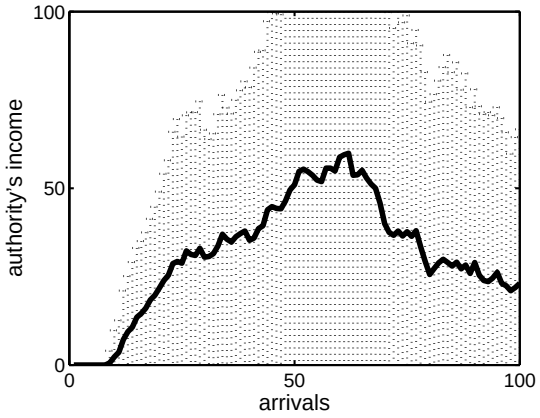
12.3(d)), the difference is less observable in Figure 12.3(f) despite the fact that type-4 nodes are even more punished in the latter case. Since tpye-2 (UMTS-like) and type-3 (UWB-like) nodes experience similar lifetimes irrespective to the applied frequency band selection, the interference-aware strategy proposes a fair trade-off among the income and the allocation success of type-1 and type-3 nodes in this setting.



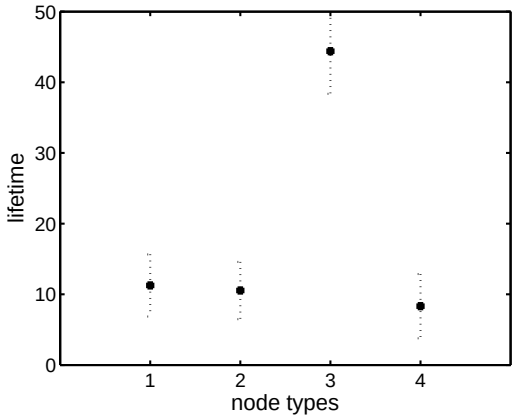
(a) Cost minimizing frequency band selection



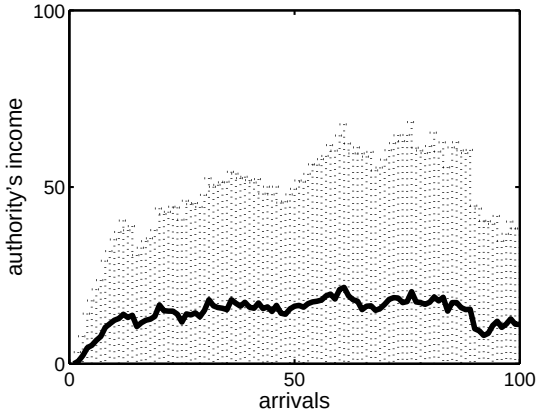
(b) Cost minimizing frequency band selection



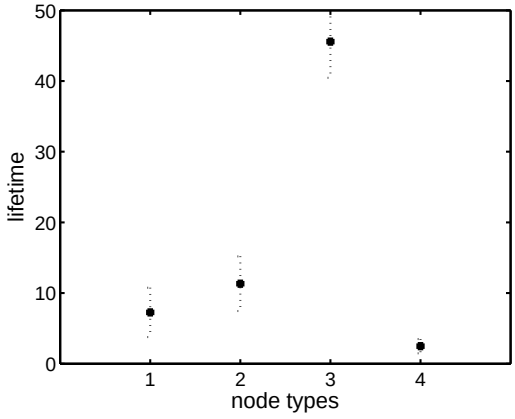
(c) Interference-aware frequency band selection



(d) Interference-aware frequency band selection



(e) Deliberate frequency band selection



(f) Deliberate frequency band selection

Figure 12.3: Authority income and node lifetimes with different frequency band selection strategies in the second scenario

Chapter 13

Conclusions and perspectives

We presented our work on dynamic spectrum allocation and we proposed a distributed framework for the allocation and pricing of radio spectrum among competing wireless service providers. We applied a general point-to-point interference model which characterizes interference dependencies among the participants realistically. After having emphasized the drawbacks of the implementation of a central allocation scheme because of its synchronization constraints and excessive complexity, we proposed a distributed scheme in which the participants follow simple rules in order to allocate and price the radio spectrum among themselves.

In line with practical considerations, we suppose that different nodes express spectrum allocation demands delayed in time. Therefore, in terms of allocation, we introduced the concept of one-way exclusion, which allows the nodes to buy out interfering nodes at the beginning of their activation period, in order to assure unperturbed operation. We described the potential re-allocations and emphasized on the asynchronous, self-organizing and self-stabilizing attributes of our distributed dynamic spectrum allocation framework. In line with practical considerations, we defined the dynamic utility of participants for spectrum based on discounted future income estimations. We completed our framework by pricing policies: we considered second-price auctions with payment division when re-selling spectrum between the authority and the participants.

We evaluated the proposed framework analytically and numerically. We showed that the model is incentive-compatible, thus assures truthfulness of spectrum valuation reporting in the game theoretical sense. We also proved that our preliminary goals about fairness and efficiency on spectrum utilization are fulfilled: less interference-friendly nodes (that cause high interference at others and/or do not tolerate interference from others) pay relatively more for the spectrum than the rest.

We have shown that the one-way exclusions introduce hard optimization problems that nodes face during allocation, therefore we proposed reasonable policy choices for node exclusions. We also gave estimates on the success of spectrum allocation for given types of nodes in terms of costs and expected lifetime. We have given necessary and sufficient conditions for successful allocation

and we proposed heuristic band selection strategies based on our findings in the queuing model that we built. We designed algorithms for the optimization of node payoff, and we evaluated the different algorithmic choices in numeric simulations.

The main advantages of our framework and heuristics are temporal flexibility and capability for fast, local response to any variation in the node set, while respecting the efficiency requirements of spectrum allocation. The distributed allocation and pricing scheme is thus a well-engineered environment for selfish (in a game theoretic sense) participants, designed to maximize the utility at which the spectrum is exploited at any given point in time.

Chapter 14

Summary

We summarize our contributions and discuss their practical application possibilities below.

14.1 The design and analysis of a P2P backup system

In Chapter 4, our contributions target multiple facets of a P2P backup system. First, we presented a data redundancy scheme that adapts the resource contribution of users to a low level while ensuring a fair quality backup service. We proposed to determine the data redundancy rate with a method that is based on estimations of data loss probability and necessary time to retrieve backup data. The scheme guarantees high service quality for backup purposes, and keeps the storage and bandwidth requirements imposed on users low. Second, we proposed a method to group peers based on their availability and bandwidth characteristics and to exchange fragments within the groups in a symmetric way. This peer selection ensures fairness in terms of quality of service and resource contribution. Third, we proposed a hybrid system architecture that contains a data center in order to complement P2P resources. The data center improves the quality of service by assisting to the backup phases and to the repairs of lost fragments for a reasonable cost.

In Chapter 5, we analyzed a model of peer selection in P2P backup systems in which users have the ability to selfishly select remote peers they want to exchange data with in a symmetric scheme. We described user "selfishness" with a game theoretic [105] payoff model. Furthermore, we defined a non-cooperative game, termed as exchange game, built on the payoff function to reflect the selfish context of user-driven peer selection. We proposed to reduce the exchange game with fixed grades to a matching problem. Based on the peer selection preferences induced by the payoff, we showed that matches are created between peers with similar grades. We proved the existence of equilibrium in the exchange game, and we gave the best-response user strategies in respect to grade and remote peer selection.

In Chapter 6, we showed an efficient algorithm to compute the optimal data transfer schedul-

ing solution for hypothetical cases where future peer uptimes are known. We used the optimal results to evaluate the applied scheduling policy, and we proposed practical settings in which the performance of random decisions is close to optimal.

In Chapter 7, we provided numerical evaluation of the system design choices that we suggested.

The application possibilities of the research results are potentially high, even on a short term. The findings of the extensive investigation on P2P backup systems consists of both theoretic and practical results. The presented combination of matching-, and game theory models may provide new insights on existing problems in distributed systems. The defined performance metrics and the novel data redundancy calculation approach may affect other P2P storage system designs, not necessarily only for backup purposes. In addition, the implemented prototype can provide a basis for developing practical backup applications. These latter can be commercialized shortly for end-users devices, can be deployed on subscriber set-top-boxes of Internet service providers, or in corporate networks of companies.

14.2 Distributed dynamic spectrum allocation

In Chapter 11, we investigated the possibility of allocating radio spectrum among multiple applicants dynamically in a distributed manner. We defined a spectrum allocation framework, and we showed that the pricing rules ensure the essential characteristics of resource distribution: rationality (user payoff cannot be negative), incentive compatibility (users bid with their true utilities), and less interference-friendly nodes pay relatively more for the spectrum. We proved that making an ideal exclusion strategy, *e.g.*, optimizing the set of disturbing nodes, is an NP-complete problem. We suggested heuristic node exclusion strategies, and we gave a necessary condition for successful allocation. We suggested a heuristic algorithm with various frequency band selection strategies.

In Chapter 12, we evaluated our heuristics in simulations.

The analysis of spectrum allocation gives insights about the potential future radio frequency management schemes. We showed a distributed, incentive-compatible model with goals of ensuring fairness and efficiency on spectrum utilization, although we highlighted the algorithmic complexity of exclusions and frequency band selection. The allocation and pricing scheme is a well-engineered environment for selfish participants, provides temporal flexibility and capability for fast, local response to any variation in the frequency demand. However, its practical deployment will be probably deferred in the future at a time when the scarcity of radio spectrum will significantly worsen the wireless services.

Bibliography

[1] Allthingsdistributed. http://www.allthingsdistributed.com/.

[2] Amazon s3. http://aws.amazon.com/s3.

[3] Dropbox. https://www.dropbox.com/.

[4] Napster. http://www.napster.com.

[5] Slashdot, nada. http://tech.slashdot.org/article.pl?sid=08/07/16/1515211.

[6] Ubistorage. http://www.ubistorage.com/.

[7] Wuala. http://wua.la/en/home.html.

[8] A. Adya, W. Bolosky, M. Castro, R. Chaiken, G. Cermak, J.Douceur, J. Howell, J. Lorch, M. Theimer, and R. Wattenhofer. Farsite: Federated, available, and reliable storage for an incompletelytrusted environment. In *USENIX Symposium on Operating Systems Design and Implementation (OSDI)*, 2002.

[9] A. Al Daoud, M. Alanyali, and D. Starobinski. Secondary pricing of spectrum in cellular CDMA networks. In *IEEE International Symposium on New Frontiers in Dynamic Spectrum Access Networks (DySPAN)*, 2007.

[10] E. Altman, A. A. Kherani, P. Michiardi, and R. Molva. Non-cooperative forwarding in ad-hoc networks. In *IFIP NETWORKING*, 2005.

[11] K. Anagnostakis and M. Greenwald. Exchange-based incentive mechanisms for peer-to-peer file sharing. In *IEEE International Conference on Distributed Computing Systems (ICDCS)*, 2004.

[12] P. Antoniadis, C. Courcoubetis, and R. Mason. Comparing economic incentives in peer-to-peer networks. *Computer Networks*, 46(1):133–146, 2004.

[13] P. Antoniadis, C. Courcoubetis, and B. Strulo. Incentives for content availability in memory-less peer-to-peer file sharing systems. *SIGecom Exchanges*, 5(4):11–20, 2005.

[14] G. Ateniese, R. Di Pietro, L. Mancini, and G. Tsudik. Scalable and efficient provable data possession. In *SecureComm*, 2008.

[15] J. Bae, E. Beigman, R. Berry, M. Honig, and R. Vohra. On the efficiency of sequential auctions for spectrum sharing. In *International Conference on Game Theory for Networks (GameNets)*, 2009.

[16] C. Batten, K. Barr, A. Saraf, and S. Treptin. pStore: A secure peer-to-peer backup system. Technical Report MIT-LCS-TM-632, MIT Laboratory for Computer Science, 2001.

[17] R. Bhagwan, S. Savage, and G. M. Voelker. Understanding availability. In *International Workshop on Peer-To-Peer Systems (IPTPS)*, 2003.

[18] R. Bhagwan, K. Tati, Y. Cheng, S. Savage, and G. M. Voelker. Totalrecall: System support for automated availability management. In *ACM/USENIX Symposium on Networked Systems Design and Implementation (NSDI)*, 2004.

[19] Y. Birk and T. Kol. Coding and scheduling considerations for peer-to-peer storage backup systems. In *IEEE International Workshop on Storage Network Architecture and Parallel I/Os*, 2007.

[20] C. Blake and R. Rodrigues. High availability, scalable storage, dynamic peer networks: Pick two. In *USENIX Workshop on Hot Topics in Operating Systems (HotOS)*, 2003.

[21] W. J. Bolosky, J. R. Douceur, D. Ely, and M. Theimer. Feasibility of a serverless distributed file system deployed on an existing set of desktop pcs. In *SIGMETRICS*, 2000.

[22] M. Buddhikot and K. Ryan. Spectrum management in coordinated dynamic spectrum access based cellular networks. In *IEEE International Symposium on New Frontiers in Dynamic Spectrum Access Networks (DySPAN)*, 2005.

[23] L. Buttyán and J.-P. Hubaux. Stimulating cooperation in self-organizing mobile ad hoc networks. *Mobile Networks and Applications*, 8(5):579–592, 2003.

[24] L. Cao and H. Zheng. Distributed spectrum allocation via local bargaining. In *SECON*, 2005.

[25] L. Cao and H. Zheng. On the efficiency and complexity of distributed spectrum allocation. In *International ICST Conference on Cognitive Radio Oriented Wireless Networks (CrownCom)*, 2007.

[26] L. Cao and H. Zheng. Stable and efficient spectrum access in next generation dynamic spectrum networks. In *IEEE International Conference on Computer Communications (INFOCOM)*, 2008.

[27] B. Chun, F. Dabek, A. Haeberlen, E. Sit, H. Weatherspoon, M. F. Kaashoek, J. Kubiatowicz, and R. Morris. Efficient replica maintenance for distributed storage systems. In *ACM/USENIX Symposium on Networked Systems Design and Implementation (NSDI)*, 2006.

[28] E. H. Clarke. Multipart pricing of public goods. *Public Choice*, 11(1), 1971.

[29] B. Cohen. Incentives build robustness in bittorrent. In *Workshop on the Economics of Peer-to-Peer Systems (NetEcon)*, 2003.

[30] J. Corbo and D. Parkes. The price of selfish behavior in bilateral network formation. In *ACM Symposium on Principles of Distributed Computing (PODC)*, 2005.

[31] C. Courcoubetis and R. Weber. Incentives for large peer-to-peer systems. *IEEE Journal on Selected Areas in Communications*, 24(5):1034 – 1050, may. 2006.

[32] L. Cox and B. Noble. Pastiche: Making backup cheap and easy. In *USENIX Symposium on Operating Systems Design and Implementation (OSDI)*, 2002.

[33] L. Cox and B. Noble. Samsara: Honor among thieves in peer-to-peer storage. In *ACM Symposium on Operating Systems Principles (SOSP)*, 2003.

[34] A. Csoma, L. Toka, and A. Vidacs. A peer-to-peer backup system with incentives. In *International Conference on Telecommunications and Signal Processing*, 2010.

[35] F. Dabek, M. F. Kaashoek, D. Karger, R. Morris, and I. Stoica. Wide-area cooperative storage with CFS. In *ACM Symposium on Operating Systems Principles (SOSP)*, 2001.

[36] A. Datta and K. Aberer. Internet-scale storage systems under churn – a study of the steady-state using markov models. In *IEEE International Conference on Peer-to-Peer Computing (P2P)*, 2006.

[37] G. DeCandia, D. Hastorun, M. Jampani, G. Kakulapati, A.Lakshman, A. Pilchin, S. Sivasubramanian, P. Vosshall, and W. Vogels. Dynamo: Amazon's highly available key-value store. In *ACM Symposium on Operating Systems Principles (SOSP)*, 2007.

[38] M. Dell'Amico, P. Michiardi, and Y. Roudier. Back to the future: On predicting user uptime. Technical report, arXiv, 2010.

[39] A. G. Dimakis, P. B. Godfrey, M. J. Wainwright, and K. Ramchandran. Network coding for distributed storage systems. In *IEEE International Conference on Computer Communications (INFOCOM)*, 2007.

[40] R. Dingledine, M. J. Freedman, and D. Molnar. The free haven project: Distributed anonymous storage service. In *Proceedings of Designing Privacy Enhancing Technologies: Workshop on Design Issues in Anonymity and Unobservability*, 2000.

[41] J. R. Douceur. The Sybil attack. In *International Workshop on Peer-To-Peer Systems (IPTPS)*, 2002.

[42] J. R. Douceur and R. Wattenhofer. Competitive hill-climbing strategies for replica placement in a distributed file system. In *International Conference on Distributed Computing (DISC)*, 2001.

[43] P. Druschel and A. Rowstron. PAST: A large-scale, persistent peer-to-peer storage utility. In *USENIX Workshop on Hot Topics in Operating Systems (HotOS)*, 2001.

[44] A. Duminuco. *Data redundancy and maintenance for peer-to-peer file backup systems.* PhD thesis, Eurecom, 2009.

[45] A. Duminuco and E. Biersack. Hierarchical codes: How to make erasure codes attractive for peer-to-peer storage systems. In *IEEE International Conference on Peer-to-Peer Computing (P2P)*, 2008.

[46] A. Duminuco and E. Biersack. A practical study of regenerating codes for peer-to-peer backup systems. In *IEEE International Conference on Distributed Computing Systems (ICDCS)*, 2009.

[47] A. Duminuco, E. Biersack, and T. En-Najjary. Proactive replication in distributed storage systems using machine availabilityestimation. In *Conference on emerging Networking EXperiments and Technologies (CoNEXT)*, 2007.

[48] A. O. Ecran, J. Lee, S. Pollin, and J. Rabaey. A revenue enhancing Stackelberg game for owners in opportunistic spectrum access. In *IEEE International Symposium on New Frontiers in Dynamic Spectrum Access Networks (DySPAN)*, 2008.

[49] A. Fabrikant, A. Luthra, E. Maneva, C. H. Papadimitriou, and S. Shenker. On a network creation game. In *ACM Symposium on Principles of Distributed Computing (PODC)*, 2003.

[50] J. Feigenbaum, V. Ramachandran, and M. Schapira. Incentive-compatible interdomain routing. In *ACM Conference on Electronic Commerce (EC)*, 2006.

[51] J. Feigenbaum and S. Shenker. Distributed algorithmic mechanism design: Recent results and future directions. In *International Workshop on Discrete Algorithms and Methods forMobile Computing and Communications*, 2002.

[52] M. Feldman, K. Lai, I. Stoica, and J. Chuang. Robust incentive techniques for peer-to-peer networks. In *ACM Conference on Electronic Commerce (EC)*, 2004.

[53] M. Feldman, C. Papadimitriou, J. Chuang, and I. Stoica. Free-riding and whitewashing in peer-to-peer systems. *IEEE Journal on Selected Areas in Communications*, 24(5):1010 – 1019, may. 2006.

[54] D. Figueiredo, J. Shapiro, and D. Towsley. Incentives to promote availability in peer-to-peer anonymity systems. In *IEEE International Conference on Network Protocols (ICNP)*, 2005.

[55] C. Fragouli, J.-Y. Le Boudec, and J. Widmer. Network coding: an instant primer. *SIGCOMM Compution Communication Review*, 36(1):63–68, 2006.

[56] D. Fudenberg and J. Tirole. *Game Theory*. MIT Press, 1991.

[57] A. Gai, F. Mathieu, F. de Montgolfier, and J. Reynier. Stratification in p2p networks: Application to BitTorrent. In *IEEE International Conference on Distributed Computing Systems (ICDCS)*, 2007.

[58] D. Gale and L. S. Shapley. College admissions and the stability of marriage. *The American Mathematical Monthly*, 69(1):9–15, 1962.

[59] S. Gandhi, C. Buragohain, L. Cao, H. Zheng, and S. Suri. A general framework for wireless spectrum auctions. In *IEEE International Symposium on New Frontiers in Dynamic Spectrum Access Networks (DySPAN)*, 2007.

[60] A. J. Ganesh, A.-M. Kermarrec, and L. Massoulié. Peer-to-peer membership management for gossip-based protocols. *IEEE Transactions on Computers*, 52(2):139–149, 2003.

[61] M. R. Garey and D. S. Johnson. *Computers and Intractability; A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., 1990.

[62] P. B. Godfrey, S. Shenker, and I. Stoica. Minimizing churn in distributed systems. *SIGCOMM Compution Communication Review*, 36(4):147–158, 2006.

[63] A. V. Goldberg and R. E. Tarjan. A new approach to the maximum flow problem. In *ACM Symposium on Theory of Computing (STOC)*, 1986.

[64] P. Golle, K. Leyton-Brown, I. Mironov, and M. Lillibridge. Incentives for sharing in peer-to-peer networks. In *ACM Conference on Electronic Commerce (EC)*, 2001.

[65] D. Grolimund, L. Meisser, S. Schmid, and R. Wattenhofer. Havelaar: A Robust and Efficient Reputation System for Active Peer-to-PeerSystems. In *Workshop on the Economics of Peer-to-Peer Systems (NetEcon)*, 2006.

[66] T. Groves. Incentives in teams. *Econometrica*, 41(4):617–631, 1973.

[67] A. Haeberlen, A. Mislove, and P. Druschel. Glacier: Highly durable, decentralized storage despite massive correlatedfailures. In *ACM/USENIX Symposium on Networked Systems Design and Implementation (NSDI)*, 2005.

[68] A. T. Hoang and Y.-C. Liang. Dynamic spectrum allocation with second-price auctions: When time is money. In *International ICST Conference on Cognitive Radio Oriented Wireless Networks (CrownCom)*, 2008.

[69] J. Hofbauer and K. Sigmund. *Evolutionary Games and Population Dynamics*. Cambridge University Press, 1998.

[70] O. Ileri, D. Samardzija, and N. Mandayam. Demand responsive pricing and competitive spectrum allocation via a spectrum server. In *IEEE International Symposium on New Frontiers in Dynamic Spectrum Access Networks (DySPAN)*, 2005.

[71] J. Ioannidis, S. Ioannidis, A. D. Keromytis, and V. Prevelakis. Fileteller: paying and getting paid for file storage. In *International Conference on Financial Cryptography*, 2003.

[72] R. W. Irving. An efficient algorithm for the "stable roommates" problem. *Journal of Algorithms*, 6(4):577–595, 1985.

[73] R. W. Irving and S. Scott. The stable fixtures problem - a many-to-many extension of stable roommates. *Discrete Applied Mathematics*, 155(17):2118–2129, 2007.

[74] F. Junqueira, R. Bhagwan, A. Hevia, K. Marzullo, and G. Voelker. Surviving internet catastrophe. In *USENIX Annual Technical Conference*, 2005.

[75] F. Junqueira, R. Bhagwan, K. Marzullo, S. Savage, and G. M. Voelker. The phoenix recovery system: rebuilding from the ashes of an internet catastrophe. In *USENIX Workshop on Hot Topics in Operating Systems (HotOS)*, 2003.

[76] S. Kamvar, M. Schlosser, and H. Garcia-Molina. The eigentrust algorithm for reputation management in p2p networks. In *ACM World Wide Web Conference (WWW)*, 2003.

[77] D. Karger, E. Lehman, T. Leighton, R. Panigrahy, M. Levine, and D. Lewin. Consistent hashing and random trees. In *ACM Symposium on Theory of Computing (STOC)*, 1997.

[78] V. Kone, L. Yang, X. Yang, B. Y. Zhao, and H. Zheng. On the feasibility of effective opportunistic spectrum access. In *ACM/USENIX Internet Measurement Conference (IMC)*, 2010.

[79] L. Kovács and A. Vidács. Interference-tolerant spatio-temporal dynamic spectrum allocation. In *IEEE International Symposium on New Frontiers in Dynamic Spectrum Access Networks (DySPAN)*, 2007.

[80] L. Kovács, A. Vidács, and B. Héder. Spectrum auction and pricing in dynamic spectrum allocation networks. *The Mediterranean Journal of Computers and Networks*, 4(3):125–138, 2008.

[81] J. Kubiatowicz, D. Bindel, Y. Chen, S. Czerwinski, P. Eaton, D. Geels, R. Gummadi, S. Rhea, H. Weatherspoon, W. Weimer, C. Wells, and B. Zhao. Oceanstore: An architecture for global-scale persistent storage. In *ACM Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS)*, 2000.

[82] K. Lai, M. Feldman, I. Stoica, and J. Chuang. Incentives for cooperation in peer-to-peer networks. In *Workshop on the Economics of Peer-to-Peer Systems (NetEcon)*, 2003.

[83] M. Landers, H. Zhang, and K. Tan. Peerstore: Better performance by relaxing in peer-to-peer backup. In *IEEE International Conference on Peer-to-Peer Computing (P2P)*, 2004.

[84] N. Laoutaris, G. Smaragdakis, A. Bestavros, and J. W. Byers. Implications of selfish neighbor selection in overlay networks. In *IEEE International Conference on Computer Communications (INFOCOM)*, 2007.

[85] N. Laoutaris, G. Smaragdakis, K. Oikonomou, I. Stavrakakis, and A. Bestavros. Distributed placement of service facilities in large-scale networks. In *IEEE International Conference on Computer Communications (INFOCOM)*, 2007.

[86] N. Laoutaris, G. Zervas, A. Bestavros, and G. Kollios. The cache inference problem and its application to content and request routing. In *IEEE International Conference on Computer Communications (INFOCOM)*, 2007.

[87] P. Leaves, S. Ghaheri-Niri, R. Tafazolli, L. Christodoulides, T. Sammut, W. Staht, and J. Huschke. Dynamic spectrum allocation in a multi-radio environment: concept and algorithm. In *International Conference on 3G Mobile Communication Technologies*, 2001.

[88] D. Lebedev, F. Mathieu, L. Viennot, A.-T. Gai, J. Reynier, and F. de Montgolfier. On using matching theory to understand p2p network design. In *International Network Optimization Conference*, 2007.

[89] G. Lefebvre and M. J. Feeley. Separating durability and availability in self-managed storage. In *ACM SIGOPS European Workshop*, 2004.

[90] J. Li and F. Dabek. F2f: reliable storage in open networks. In *International Workshop on Peer-To-Peer Systems (IPTPS)*, 2006.

[91] M. Lillibridge, S. Elnikety, A. Birrell, M. Burrows, and M. Isard. A cooperative internet backup scheme. In *USENIX Annual Technical Conference*, 2003.

[92] F. Liu, Y. Sun, B. Li, B. Li, and X. Zhang. Fs2you: Peer-assisted semipersistent online hosting at a large scale. *IEEE Transactions on Parallel and Distributed Systems*, 21(10):1442–1457, 2010.

[93] M. Luby. Lt codes. In *IEEE Symposium on Foundations of Computer Science (FOCS)*, 2002.

[94] T. Mager. Measurement study of Wuala, a distributed social storage service. Master's thesis, Eurecom, 2010.

[95] P. Maille and L. Toka. Managing a peer-to-peer data storage system in a selfish society. *IEEE Journal on Selected Areas in Communications*, 26(7):1295 –1301, 2008.

[96] P. Maillé and B. Tuffin. Pricing the internet with multibid auctions. *IEEE/ACM Transactions on Networking*, 14(5):992–1004, 2006.

[97] P. Maille and B. Tuffin. Analysis of price competition in a slotted resource allocation game. In *IEEE International Conference on Computer Communications (INFOCOM)*, 2008.

[98] F. Mathieu. *Acyclic Preference-Based Systems*, pages 1165–1203. Springer Verlag, 2010.

[99] P. Michiardi and L. Toka. Selfish neighbor selection in peer-to-peer backup and storage applications. In *International Conference on Parallel and Distributed Computing (Euro-Par)*, 2009.

[100] S. Nath, H. Yu, P. B. Gibbons, and S. Seshan. Subtleties in tolerating correlated failures in wide-area storage systems. In *ACM/USENIX Symposium on Networked Systems Design and Implementation (NSDI)*, 2006.

[101] G. Neglia, G. L. Presti, H. Zhang, and D. Towsley. A network formation game approach to study bittorrent tit-for-tat. In *NET-COOP*, 2007.

[102] T. Ngan, A. Nandi, A. Singh, D. Wallach, and P. Druschel. On designing incentives-compatible peer-to-peer systems. In *International Workshop on Future Directions in Distributed Computing*, 2004.

[103] N. Nie and C. Comaniciu. Adaptive channel allocation spectrum etiquette for cognitive radio networks. *Mobile Networks and Applications*, 11(6):779–797, 2006.

[104] P. Nurmi. A bayesian framework for online reputation systems. In *Advanced International Conference on Telecommunications and Internet and Web Applications and Services*, 2006.

[105] M. J. Osborne and A. Rubinstein. *A Course in Game Theory.* The MIT Press, July 1994.

[106] N. Oualha, M. Önen, and Y. Roudier. A security protocol for self-organizing data storage. In *SEC*, 2008.

[107] L. Pamies-Juarez and P. Garcia-Lopez. Maintaining data reliability without availability in p2p storage systems. In *SAC*, 2010.

[108] L. Pamies-Juarez, P. García-López, and M. Sánchez-Artigas. Rewarding stability in peer-to-peer backup systems. In *ICON*, 2008.

[109] C. Peng, H. Zheng, and B. Y. Zhao. Utilization and fairness in spectrum assignment for opportunistic spectrum access. *Mobile Networks and Applications*, 11(4):555–576, 2006.

[110] M. Piatek, T. Isdal, T. Anderson, A. Krishnamurthy, and A. Venkataramani. Do incentives build robustness in BitTorrent? In *ACM/USENIX Symposium on Networked Systems Design and Implementation (NSDI)*, 2007.

[111] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker. A scalable content-addressable network. In *ACM Conference on Applications, technologies, architectures, and protocols for computer communication (SIGCOMM)*, 2001.

[112] I. S. Reed and G. Solomon. Polynomial codes over certain finite fields. *Journal of the Society for Industrial and Applied Mathematics*, 8(2):300–304, 1960.

[113] R. Rodrigues and B. Liskov. High availability in DHTs: Erasure coding vs. replication. In *International Workshop on Peer-To-Peer Systems (IPTPS)*, 2005.

[114] V. Rodriguez, K. Moessner, and R. Tafazolli. Market driven dynamic spectrum allocation over space and time among radio-access networks: Dvb-t and b3g cdma with heterogeneous terminals. *Mobile Networks and Applications*, 11(6):847–860, 2006.

[115] A. E. Roth and M. A. O. Sotomayor. *Two-sided matching: A study in game-theoretic modeling and analysis.* Cambridge University Press, 1990.

[116] A. Rowstron and P. Druschel. Pastry: Scalable, decentralized object location and routing for large-scale peer-to-peer systems. In *IFIP/ACM International Conference on Distributed Systems Platforms*, 2001.

[117] K. Ryan, E. Aravantinos, and M. M. Buddhikot. A new pricing model for next generation spectrum access. In *International Workshop on Technology and Policy for Accessing Spectrum (TAPAS)*, 2006.

[118] K. Rzadca, A. Datta, and S. Buchegger. Replica placement in p2p storage: Complexity and game theoretic analyses. In *IEEE International Conference on Distributed Computing Systems (ICDCS)*, 2010.

[119] T. S. J. Schwarz and E. L. Miller. Store, forget, and check: Using algebraic signatures to check remotely administered storage. In *IEEE International Conference on Distributed Computing Systems (ICDCS)*, 2006.

[120] S. Sengupta, M. Chatterjee, and S. Ganguly. An economic framework for spectrum allocation and service pricing with competitive wireless service providers. In *IEEE International Symposium on New Frontiers in Dynamic Spectrum Access Networks (DySPAN)*, 2007.

[121] E. Sit, A. Haeberlen, F. Dabek, B. gon Chun, H. Weatherspoon, R. Morris, M. F. Kaashoek, and J. Kubiatowicz. Proactive replication for data durability. In *International Workshop on Peer-To-Peer Systems (IPTPS)*, 2006.

[122] J. M. Smith. *Evolution and the Theory of Games*. Cambridge University Press, 1982.

[123] I. Stoica, R. Morris, D. Liben-Nowell, D. R. Karger, M. F. Kaashoek, F. Dabek, and H. Balakrishnan. Chord: a scalable peer-to-peer lookup protocol for internet applications. *IEEE/ACM Transactions on Networking*, 11(1):17–32, 2003.

[124] A. Subramanian, M. Al-Ayyoub, H. Gupta, S. Das, and M. Buddhikot. Near-optimal dynamic spectrum allocation in cellular networks. In *IEEE International Symposium on New Frontiers in Dynamic Spectrum Access Networks (DySPAN)*, 2008.

[125] A. Subramanian, H. Gupta, S. R. Das, and M. M. Buddhikot. Fast spectrum allocation in coordinated dynamic spectrum access based cellular networks. In *IEEE International Symposium on New Frontiers in Dynamic Spectrum Access Networks (DySPAN)*, 2007.

[126] K. Tangwongsan, H. Pucha, D. G. Andersen, and M. Kaminsky. Efficient similarity estimation for systems exploiting data redundancy. In *IEEE International Conference on Computer Communications (INFOCOM)*, 2010.

[127] J. Tian, Z. Yang, and Y. Dai. A data placement scheme with time-related model for p2p storages. In *IEEE International Conference on Peer-to-Peer Computing (P2P)*, 2007.

[128] L. Toka, M. Dell'Amico, and P. Michiardi. Online data backup : a peer-assisted approach. In *IEEE International Conference on Peer-to-Peer Computing (P2P)*, 2010.

[129] L. Toka, A. Korosi, and A. Vidacs. On distributed dynamic spectrum allocation for sequential arrivals. In *IEEE Symposia on New Frontiers in Dynamic Spectrum Access Networks (DySPAN)*, 2010.

[130] L. Toka and P. Maillé. Managing a peer-to-peer backup system: Does imposed fairness socially outperform a revenue-driven monopoly? In *International Workshop on Grid Economics and Business Models*, 2007.

[131] L. Toka and P. Michiardi. Analysis of user-driven peer selection in peer-to-peer backup and storage systems. In *International Workshop on Game Theory in Communication Networks (GameComm)*, 2008.

[132] L. Toka and P. Michiardi. A dynamic exchange game. In *ACM Symposium on Principles of Distributed Computing (PODC)*, 2008.

[133] L. Toka and P. Michiardi. Uncoordinated peer selection in p2p backup and storage applications. In *IEEE Global Internet Symposium*, 2009.

[134] L. Toka and P. Michiardi. Analysis of user-driven peer selection in peer-to-peer backup and storage systems. *Telecommunication Systems*, 2010. in press.

[135] L. Toka and A. Vidács. Distributed dynamic spectrum management. In *International Conference on Telecommunications and Signal Processing*, 2009.

[136] L. Toka and A. Vidács. General distributed economic framework for dynamic spectrum allocation. *Computer Communications*, 32(18):1955 – 1964, 2009.

[137] D. N. Tran, F. Chiang, and J. Li. Friendstore: Cooperative online backup using trusted nodes. In *International Workshop on Social Network Systems*, 2008.

[138] W. Vickrey. Counterspeculation, auctions, and competitive sealed tenders. *The Journal of Finance*, 16(1):8–37, 1961.

[139] V. Vishnumurthy, S. Chandrakumar, and E. Sirer. KARMA: A secure economic framework for peer-to-peer resource sharing. In *Workshop on the Economics of Peer-to-Peer Systems (NetEcon)*, 2003.

[140] R. Wauters. Online backup company carbonite loses customers' data, blames and sues suppliers. TechCrunch, http://techcrunch.com/2009/03/23/online-backup-company-carbonite-loses-customers-data-blames-and-sues-suppliers/, 2009.

[141] H. Weatherspoon and J. D. Kubiatowicz. Erasure coding vs. replication: A quantitative comparison. In *International Workshop on Peer-To-Peer Systems (IPTPS)*, 2002.

[142] H. Weatherspoon, T. Moscovitz, and J. Kubiatowicz. Introspective failure analysis: Avoiding correlated failures in peer-to-peer systems. In *IEEE Symposium on Reliable Distributed Systems*, 2002.

[143] H. Weatherspoon, C. Wells, and J. D. Kubiatowicz. Naming and integrity: self-verifying data in peer-to-peer systems. In *International Workshop on Future Directions in Distributed Computing*, pages 142–147. Springer-Verlag, 2003.

[144] Y. Wu, R. Dimakis, and K. Ramch. Deterministic regenerating codes for distributed storage. In *Annual Allerton Conference on Communication, Control, and Computing*, 2007.

[145] Y. Wu, B. Wang, K. J. R. Liu, and T. C. Clancy. A multi-winner cognitive spectrum auction framework with collusion-resistant mechanisms. In *IEEE International Symposium on New Frontiers in Dynamic Spectrum Access Networks (DySPAN)*, 2008.

[146] L. Yang, L. Cao, and H. Zheng. Physical interference driven dynamic spectrum management. In *IEEE International Symposium on New Frontiers in Dynamic Spectrum Access Networks (DySPAN)*, 2008.

[147] Z. Yang, B. Y. Zhao, Y. Xing, S. Ding, F. Xiao, and Y. Dai. Amazingstore: Available, low-cost online storage service using cloudlets. In *International Workshop on Peer-To-Peer Systems (IPTPS)*, 2010.

[148] B. Yu and M. P. Singh. Incentive mechanisms for peer-to-peer systems. In *International Workshop on Agents and Peer-to-Peer Computing*, 2003.

[149] Z. Zhang, Q. Lian, S. Lin, W. Chen, Y. Chen, and C. Jin. Bitvault: a highly reliable distributed data retention platform. *SIGOPS Operating Systems Review*, 41(2):27–36, 2007.

[150] B. Y. Zhao, L. Huang, J. Stribling, S. C. Rhea, A. D. Joseph, and J. D. Kubiatowicz. Tapestry: A resilient global-scale overlay for service deployment. *IEEE Journal on Selected Areas in Communications*, 22:41–53, 2004.

[151] J. Zhao, H. Zheng, and G.-H. Yang. Distributed coordination in dynamic spectrum allocation networks. In *IEEE International Symposium on New Frontiers in Dynamic Spectrum Access Networks (DySPAN)*, 2005.

[152] S. Zhong, J. Chen, and Y. Yang. Sprite: a simple, cheat-proof, credit-based system for mobile ad-hoc networks. In *IEEE International Conference on Computer Communications (INFOCOM)*, 2003.

[153] X. Zhou, S. Gandhi, S. Suri, and H. Zheng. eBay in the sky: strategy-proof wireless spectrum auctions. In *ACM International Conference on Mobile Computing and Networking (MobiCom)*, 2008.

[154] X. Zhou and H. Zheng. TRUST: A general framework for truthful double spectrum auctions. In *IEEE International Conference on Computer Communications (INFOCOM)*, 2009.

Annexe A

Synthèse en Français

A.1 Introduction

Mon travail de recherche vise à concevoir des systèmes distribués avec un accent sur l'incitation d'approvisionnement des ressources. La thèse propose des solutions pour la sauvegarde des données dans un réseau pair-à-pair (P2P, un réseau des éléments fonctionnellement égales) et pour la distribution décentralisée du spectre radioélectrique par des modèles de la théorie des jeux. Le comportement inhérent égoïste de participants est atténué par les régimes d'incitation bien adapté. Des résultats analytiques et numériques reflètent la performance des conceptions de système conseillées.

A.1.1 Le contexte

Aujourd'hui, de plus en plus de services et d'applications de technologie de l'information utilisent le paradigme de répartition afin de garantir la robustesse à grange échelle. Des systèmes distribués et auto-organisé, bien que différents dans la plupart des aspects techniques, reflètent les questions d'incitation similaires. De nombreux services distribués actuellement comptent sur le comportement altruiste de leurs utilisateurs. La présence des individus égoïstes qui se retirent de la contribution volontaire au bien commun du groupe a été largement étudiée, et est connue comme le problème de "free-riding" (resquillage). Il est important de concevoir des mécanismes qui encouragent les utilisateurs à contribuer aux ressources communes et à réduire l'effet du comportement abusif dans les systèmes distribués.

Outre les travaux de recherche étendus sur les aspects techniques, les caractéristiques économiques des systèmes ont également été abordées dans la littérature récemment. Diverses solutions d'incitation sont proposées pour de nombreux systèmes distribués, tels que le partage d'accès au réseau [96, 97], le partage de fichiers à P2P [11, 29], le routage [50], la transmission de paquets dans les réseaux ad-hoc [10, 23, 152], l'attribution des radio fréquences [79], le stockage de don-

nées à P2P [11, 33, 95, 130, 139], la mise en cache du réseau de contenu [85, 86], et la formation de réseau [30, 49, 84].

A.1.2 Motivation

On attaque des systèmes distribués de multiutilisateurs qui ne peuvent pas être exploités d'une façon socialement optimale sans une conception adéquate. Des mécanismes d'incitation spécifiques doivent être déployés pour améliorer la qualité de service pour les utilisateurs du système. Selon la nature décentralisée des systèmes, ces incitations doivent éviter de se fier en grande partie sur des entités centrales.

La première partie de notre travail vise à un domaine de recherche pertinent : la besoin pour la sauvegarde de données en ligne dans une façon transparente, sûre, fiable et facilement accessible croît étant donné que des appareils électroniques utilisées quotidiennement sont souvent connectées à l'Internet et les taux de transmission augmentent, rendant les transferts de données volumineux possible. Comme les utilisateurs du système de sauvegarde à P2P exploitent des ressources de l'un à l'autre au lieu des ressources publiques ou centrales, l'encouragement du partage des ressources privées nécessite un système d'incitation bien adapté.

La deuxième partie de l'œuvre a comme l'objectif l'étude d'un système de gestion distribué à base d'enchères pour l'attribution de fréquences radio. La motivation principale de l'enquête est que la séquence d'enchères centrales qui a le but de réaffecter les ressources publiques devrait être transformée en un cadre adapté à un système de grande échelle. Par conséquent, dans le modèle présenté les participants font le commerce des ressources acquises entre eux dans une conception distribuée sans l'intervention d'un agent central.

A.2 Les buts

On vise à la construction des conceptions spécialement adaptées à deux types de système distribué : la sauvegarde de données à P2P et l'attribution des fréquences radio. Dans tous les deux cas le comportement non coopératif, égoïste des participants peut mettre en péril l'opération. Sur la base de modèles d'utilisateur, notre objectif est de concevoir des solutions d'incitation économique qui garantissent la qualité de service souhaitée, et d'évaluer leur performance à la fois par des études analytiques et numériques.

Après la reconnaissance des particularités de chaque domaine d'application, nous définissons des modèles d'utilisateur qui reflètent leurs avantages en termes de performances du système, leurs coûts de partage des ressources (dans le cas échéant), et leurs caractéristiques importantes du point de vue du système étudié, par exemple la hétérogénéité des ressources partagés ou les relations d'interférence entre les utilisateurs. Nous nous adressons à l'allocation des ressources par la construction de régime d'incitation nouveau pour les systèmes étudiés. Afin de favoriser la

coopération entre les utilisateurs soit des échanges des ressources, soit des paiements monétaires sont appliquées.

Pour évaluer les modèles de systèmes et les régimes d'incitation proposés, nous utilisons un vaste choix d'outils d'analyse : la théorie de couplage avec le modèle du système de sauvegarde à P2P, et la théorie d'enchères pour l'attribution du spectre radioélectrique. Nous décomposons les problèmes d'optimisation qui apparaissent et nous développons des algorithmes distribués pour les résoudre. Nous effectuons des simulations de nos modèles théoriques afin de prouver les concepts pour arriver enfin à la mise en œuvre des applications pratiques. Nos régimes d'incitation intégrés assurent les résultats favorables dans les systèmes robustes.

En Section A.4.1, nous présentons une conception de système à P2P qui adapte la contribution des ressources des utilisateurs à un faible niveau tout en assurant une qualité acceptable du service de sauvegarde. En Section A.4.2, nous analysons un modèle de sélection de pair dans les systèmes de sauvegarde à P2P où les utilisateurs ont la possibilité de choisir égoïstement les pairs avec qu'ils veulent échanger des données bilatéralement. En Section A.4.3, nous étudions la possibilité d'attribution du spectre radioélectrique parmi les demandeurs dynamiquement dans une manière distribuée.

A.3 Méthodologie

Nous modelons les systèmes distribués avec l'outil-ensemble de *la théorie des jeux* [56, 69, 105, 122], un moyen de la modélisation des préférences d'utilisateurs, leurs stratégies, leurs coûts et leurs évaluations. Nous investissons analytiquement le comportement des utilisateurs égoïstes, l'existence de meilleures stratégies et d'équilibre. Nous construisons des incitations [12, 31, 52] pour aligner la conduite des participants égoïste avec les objectifs de la conception du système.

Nous utilisons aussi la théorie des graphes et la théorie de couplage pour analyser les mécanismes d'incitation proposées. La théorie de couplage [58,72,73] est un domaine de l'optimisation combinatoire, et elle offre des outils utiles pour étudier, entre plusieurs autres cibles possibles, la sélection des pairs dans un système de partage de fichiers à P2P [57,88].

Enfin, nous effectuons des évaluations numériques avec des simulations écrits en MATLAB.

A.4 Résultats

A.4.1 La conception d'un système de sauvegarde à P2P

Propositions 1: [34,128] Nous avons proposé une conception de système à P2P qui adapte la contribution de ressources des utilisateurs à un niveau faible tout en assurant la qualité acceptable de service de sauvegarde.

On a étudié les systèmes à P2P qui proposent de service de sauvegarde de données, où les utilisateurs enregistrent leurs données sur les périphériques de stockage inexploitées d'autres utilisateurs à différents endroits sur Internet, gratuitement (Figure A.1). En conséquence principale, le système fonctionne à grande échelle (en augmentant du nombre d'utilisateurs l'espace de stockage global s'accroit), outre la répartition géographique ainsi que la diversité de la propriété des hôtes de stockage assurent une grande sécurité pour les données stockées. Le système doit être conçu pour maintenir une grande durabilité de la sauvegarde et pour assurer la récupération des fichiers locaux lors d'un événement de perte de données.

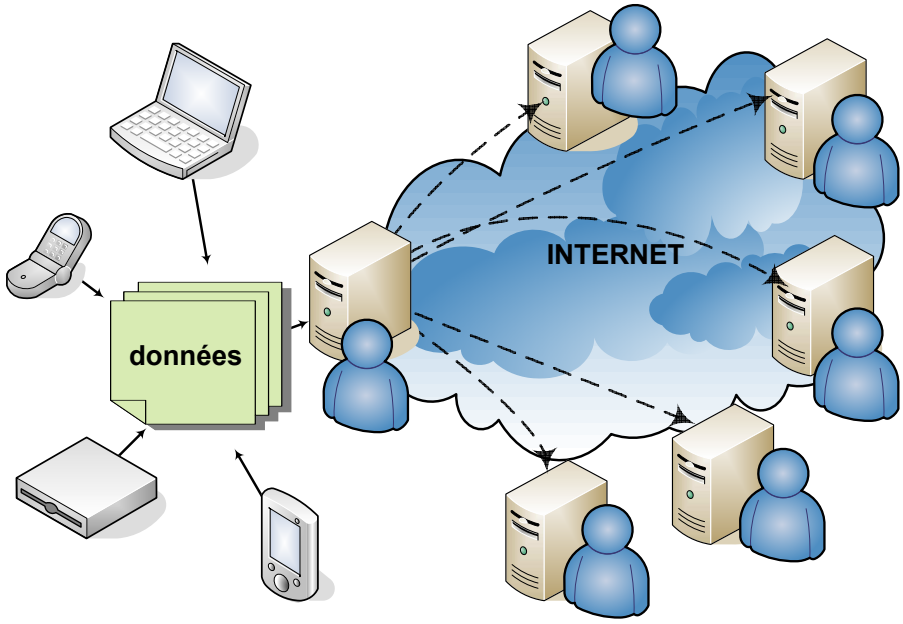


FIGURE A.1: L'application de sauvegarde, en cours d'exécution sur l'ordinateur de l'utilisateur connecté à l'Internet, stocke les données de l'utilisateur en toute sécurité en diffusant des copies sur d'autres ordinateurs participants. Après une perte de données locales, le logiciel récupère les données demandées depuis les partenaires de stockage.

Stocker des données sur des pairs temporairement indisponibles entraine la perte temporaires des données. Pour remédier à ce phénomène, les données doivent être conservées en plusieurs exemplaires, de manière redondante. Le régime appliqué pour créer la redondance de données est le codage d'effacement (erasure coding) [141], où les données à sauvegarder sont divisées en k fragments qui sont ensuite transformés pour créer $n > k$ fragments redondants. Ces derniers sont stockés sur les pairs à distance, desquels n'import quels mais au moins de k fragments, sont suffisants pour restaurer les données originales. La sauvegarde se perd si moins de k fragments peuvent être récupérés depuis les pairs à distance en cas de besoin.

La gestion de la redondance des données, et les opérations du réseau affectent largement la

qualité du service. L'étude de ces aspects a été le premier sujet de nos recherches. Afin d'évaluer les différentes conceptions de système, d'abord on a défini de simples métriques de performance pour décrire la qualité du service de sauvegarde. La première mesure reflète la longévité des données par la probabilité de perdre autant de fragments codés parmi les n fragments stockés sur les pairs à distance durant un temps donné T, que les fragments restants sont insuffisants pour restaurer les données originales.

Definition 19 *La probabilité de perte de données (DLP) est donnée par la valeur de la fonction de répartition de t durant laquelle $n-k+1$ fragments sont perdus $F_t^{n,k}(T)$ à la durée du temps T. $F_t^{n,k}$ dépend de n pairs de stockage à distance, de leur taux d'accidents, et de k.*

La deuxième série de mesures reflètent la durée de l'archivage des données et celle du processus de récupération.

Definition 20 *Le TTB (resp. TTR) d'un utilisateur est le temps écoulé avant que l'utilisateur finisse le téléchargement d'un nombre de fragments codés vers (resp. depuis) les pairs de stockage à distance qui satisfont la redondance cible (resp. sont suffisants pour rétablir des données originales). TTR est définissable seulement si l'utilisateur a sauvegardé au moins k fragments avant de commencer à récupérer une partie parmi eux.*

Dans un système de stockage idéal avec une capacité illimitée et sans interruption de temps en ligne, le TTB et TTR d'un utilisateur ne dépendent que de la quantité de données de sauvegarde, de la capacité de bande passante et de la disponibilité du propriétaire des données. On a défini *minTTB* et *minTTR* d'un utilisateur comme des bases de référence pour la sauvegarde et la restauration qui limitent ses TTB et TTR. On a utilisé ces valeurs de référence tout au long de la thèse pour comparer la performance relative du système à P2P par rapport à celle d'un tel système idéal.

Definition 21 *Les* minTTB *et* minTTR *d'un utilisateur coïncident respectivement avec ses TTB et TTR dans un système idéal. Un utilisateur i avec des bandes passantes montante et descendante u_i et d_i en commençant le téléchargement de données avec la taille o au moment t achève sa sauvegarde au temps t', après avoir passé $\frac{o}{u_i}$ de temps en ligne. De manière analogue, i restaure un objet de sauvegarde avec la même taille jusqu'au moment t'' après avoir passé $\frac{o}{d_i}$ temps en ligne. On a défini $minTTB(i,t) = t' - t$ et $minTTR(i,t) = t'' - t$.*

On a évalué les différentes options de redondance de données et de sa gestion dans des simulations numériques basées sur ces mesures. En outre, on a préconisé un système de redondance *adaptative* qui est basé sur les estimations de DLP et TTR, et détermine le taux de redondance $\frac{n}{k}$ en vue de ces valeurs. Ce régime n'assure pas la disponibilité permanante des données de sauvegarde, mais il se concentre sur la réalisation d'un TTR relativement court, tout en conservant un bas DLP estimé.

Proposition 1.1 *[128] On a proposé de déterminer le taux de redondance des données $\frac{n}{k}$ avec une méthode qui est basée sur les estimations de DLP et TTR. Ce régime garantit une haute qualité de service de sauvegarde, et ne demande qu'une contribution faible de ressources de stockage et de bande passante aux usagers.*

La méthode fonctionne comme suit. Après avoir généré des fragments codés, chaque pair en télécharge un sous-ensemble croissant jusqu'à ce que ses estimations de DLP *et* TTR tombent au-dessous des seuils prédéfinis. Sinon, plusieurs fragments sont stockés sur les pairs à distance, et le taux de redondance *adaptative* est continuellement augmenté. Nous proposons une estimation heuristique pour les mesures DLP et TTR, puisque les pertes de fragments stocké et la durée du TTR après un accident local à un moment donné sont difficiles à prévoir pour des raisons de la nature peu fiable des pairs à distance.

L'estimation de TTR (eTTR) est calculée comme la période de temps dont un utilisateur a besoin pour télécharger k fragments depuis le pair de stockage qui a le $k$ième plus élevé produit de la moyenne de probabilité d'être trouvé en ligne, appelé la disponibilité (a) et de la moyenne bande passante en montante (u). Formellement, $a = \frac{1}{t_c - t_0} \int_{t_0}^{t_c} \mathbb{P}^t(\text{en ligne}) dt$, où t_0 désigne le moment où le pair a commencé d'utiliser le service, et t_c est l'heure actuelle. Notre heuristique eTTR s'écrit comme (A.1) où j est le $k$ième plus "rapide" pair de stockage. La bande passante du pair peut être saturé avec maximum de p^d téléchargements parallèles en récupérant des fragments, et le plus bas TTR ($minTTR = \frac{o}{d}$) est atteint si aucun fragment téléchargé n'est incomplet et la capacité de téléchargement est d. L'eTTR peut être calculé pour les pairs qui déjà stocke au moins k fragments aux pairs, le nombre de fragments originaux.

$$eTTR = \frac{o}{\min\left(a_j u_j p^d, d\right)} \tag{A.1}$$

Après un accident, un pair qui a placé n fragments sur des pairs à distance peut perdre ses données sauvegardées si plus que $n - k$ d'entre eux s'écrasent avant que k fragments soient complètement récupérés. Considérant un délai qui peut s'écouler entre l'incident du crash et le début de la phase de récupération, on estime DLP (eDLP) durant un délai total de $t = retard + eTTR$ en considérant des crashs comme des événements sans mémoire, ayant une probabilité constante pour tous les pairs à tout moment. Par conséquent, le temps avant un crash, comme un variable stochastique, suit une distribution exponentielle avec une moyenne paramétrique $\overline{t}$: un pair s'écrase avant le temps t avec une probabilité de $1 - e^{-t/\overline{t}}$:

$$eDLP = \sum_{i=n-k+1}^{n} \binom{n}{i} \left(1 - e^{-t/\overline{t}}\right)^i \left(e^{-t/\overline{t}}\right)^{n-i}. \tag{A.2}$$

Les résultats de simulation de redondance à taux fixe (le régime qui vise à assurer qu'au moins k fragments de tous les utilisateurs sont en ligne avec une probabilité de 99%, basé sur la moyenne

de disponibilité des pairs) et de notre redondance adaptative sont présentés dans la Figure 7.6. Les valeurs eTTR sont calculées lorsque accidents se produisent, et elles sont comparées aux TTR mesurés à la suite : l'eTTR donne une estimation acceptable (Figure 7.5(b)). Avec des taux adaptatifs significativement plus bas, les valeurs de TTB diminuent, en revanche les résultats de TTR sont plus élevés, offrant un remède à tous les utilisateurs contre des TTB longs, et punissant avec des TTR prolongés *uniquement* ceux qui perdent leur copie locale.

Afin de maintenir un système de sauvegarde à P2P, les utilisateurs doivent partager trois types de ressources avec d'autres participants : l'espace de disque, de la bande passante et du temps de connexion au réseau. Lorsque la sélection des pairs est aléatoire, les pairs avec une haute disponibilité et avec une bonne connectivité reçoivent une charge excessive de stockage, car ils sont disponibles en ligne plus souvent, et alors plus de fragments sont téléchargés vers eux que vers des pairs à faible accessibilité. Par conséquent, en s'appuyant sur notre régime de redondance de données, nous nous sommes concentrés sur l'équité parmi les pairs : on propose une économie de troc pour encourager le partage des ressources afin d'atteindre une haute qualité de service.

Proposition 1.2 *[34, 128] On a proposé une méthode qui assure l'équité en termes de qualité de service et de la contribution des ressources. La méthode regroupe les pairs en fonction de leur disponibilité et les caractéristiques de leur bande passante et impose des échanges de fragments symétriques à l'intérieur des groupes.*

Notre plan de redondance adaptative considère des estimations basées sur le produit de la disponibilité (a) et de la bande passante moyenne (u) des pairs à distance. On écrit ce produit par $g_i = a_i u_i$, et on l'appel le *grade* du pair i. Nous avons montré que le regroupage des pairs selon leur grades et l'introduction des échanges symétriques de fragments au sein des groupes cause moins de stockage et moins de charge de trafic chez les pairs qui contribuent une haute disponibilité en ligne et de large bande passante que dans un système où la sélection des pairs est au hasard (Figure 7.11).

En outre, la qualité du service ce que chaque participant peut recevoir est limité par sa contribution : les pairs de grade bas ne peuvent pas exploiter les abondantes ressources de ceux de haut grade. En conséquence, le nombre d'événements de perte de données change en remplaçant le régime aléatoire de sélection des pairs par le plan de regroupage (Figure 7.12(a)), en particulier pour des raisons de longues phases de sauvegarde dans les grades bas. On catégorise les pertes de données de la manière suivante : si le pair écrasé

1. n'avait pas passé assez de temps en ligne à télécharger k fragments jusqu'à son crash, alors son accident de la perte de données est inévitable : pas de système de sauvegarde en ligne pourrait avoir sauvegardé les données, car la perte de données est déterminé par les limites des ressources du propriétaire de données ;

2. avait passé suffisamment de temps en ligne à télécharger au moins k fragments jusqu'à son crash, mais il n'y a pas réussi : dans ce cas, les ressources limitées des pairs à distance sont la cause de perte de données, étant donné que la sauvegarde sur un fiable centre de données avec une large bande passante aurait y réussi ;

3. avait transféré au moins k fragments, mais sa phase de sauvegarde n'a jamais terminé, la perte de données est à cause d'un TTB allongé et du fait que les pairs à distance peuvent tomber en panne ;

4. a terminé sa phase de sauvegarde, mais il n'a pas réussi à récupérer au moins k fragments après son crash, avant que ses pairs de stockage se soient écrasés dans un certain nombre fatal (TTR prolongé).

Alors que plus de pertes de données se produisent dans les deux groupes que dans un système de sélection aléatoire des pairs, les membres du groupe de haut grade souffrent d'un nombre de pertes de données sensiblement inférieur que les pairs avec des grades bas. En effet, l'équité mis en place par la sélection regroupée des pairs n'affecte pas tous les utilisateurs dans la même mesure : des transferts rapides de fragments entre le propriétaire des données et les stockeurs, tous de haut grade, résultent à courtes phases de sauvegarde, alors la probabilité de perte est plus faible que chez les utilisateurs de grade bas.

Un système à P2P peut être incapable de garantir la qualité de service approprié à cause du bas nombre d'utilisateurs et/ou de la quantité des ressources partagées d'utilisateur, par exemple, de la capacité de stockage insuffisante. Par conséquent, on a examiné les effets de l'introduction d'un serveur central de stockage pour éviter une telle situation (provisoire) : on a montré les coûts d'une qualité persistante. Dans un tel système hybride le serveur fiable (de haute disponibilité) peut être utilisé pour stocker des données en échange du remboursement des frais. On a exprimé les coûts relativement faibles qui se produisent dans tels systèmes hybrides.

Proposition 1.3 *[128] On a proposé une architecture de système hybride dans lequel un serveur central complète des ressources à P2P afin d'améliorer la qualité du service en aidant les phases de sauvegarde et de réparation des fragments perdus pour un coût raisonnable.*

Le serveur central peut rapidement récupérer et stocker des fragments dans les cas où la sauvegarde est compromise quand de nombreux fragments sont perdus pendant que le propriétaire des données est temporairement hors-ligne, excluant la possibilité d'entretien de redondance effectué par lui. Figure 7.8 représente le nombre de pertes de données observées, étiquetés selon les catégories ci-dessus. Le premier rang des figures montre les résultats des simulations où la redondance adaptative est utilisée, les résultats du système à taux fixe défini ci-dessus sont dans le deuxième rang. La première colonne représente le cas où les pairs s'apparaissent en ligne toute

de suite après leurs crashs et les pairs à distance sont informés sans délai ; la deuxième colonne montre un scénario où les pairs écrasés demeurent hors ligne pour une semaine en moyenne, et les autres n'en sont informés qu'après une semaine ; la troisième colonne représente les simulations où l'entretien de redondance est également retardée, mais assisté par un serveur.

Comme la troisième colonne de la Figure 7.8 le montre, le nombre de pertes de données est abaissé au niveau du cas des récupérations et réparations instantanées, en échange du trafic (et de la charge de stockage intermittent) du serveur, tracé dans la Figure 7.9(a). Le rôle du serveur est plus important quand les bas taux adaptatifs sont appliqués, par conséquent il effectue de plus intense trafic montant qui génère un coût plus élevé. D'autre part, le trafic de réparation générée par des crashs, effectuée par les pairs, est proportionnel à la quantité globale de données redondantes. Par suite, il est nettement plus faible lors de l'application des taux adaptatifs (Figure 7.9(b)), qui constitue un avantage de notre régime d'adaptation.

Avec les sauvegardes assistées, les pairs peuvent télécharger des fragments vers le serveur pendant leur phase de sauvegarde si les pairs à distance ne sont pas disponibles temporairement. Dans notre régime (étiquetés comme "opportuniste") téléchargements vers le serveur sont réalisés seulement avec l'excès de bande passante montante. Une alternative donne la priorité aux téléchargements vers le serveur et la sauvegarde sur des pairs à distance est démarrée seulement après avoir transféré tous les fragments originaux au serveur (ainsi appelée le régime "pessimiste"). Le régime pessimiste garantit que le TTB de tous les pairs est effectivement réduit à leurs $minTTB$. Dans ce plus "sûr" régime les données sont d'abord entièrement transférées au serveur pour construire une sauvegarde fiable dès que possible, et puis ces données stockées sont constamment supprimés pour économiser sur les frais de stockage tant que le montant de sauvegarde transféré avec succès vers les pairs à distance est en croissance.

Les sauvegardes assistées atténuent les effets négatifs des transferts de données vers les pairs de faible disponibilité et connectivité (Figure 7.14). Lors d'un téléchargement vers des pairs à distance leur indisponibilité peut empêcher les transferts, dans un régime de sauvegarde assistée les téléchargements vers le serveur sont seulement limitées par la disponibilité et la capacité de bande passante montante du pair, donc les objectifs de DLP et TTR sont atteints plus tôt, pour cette raison des valeurs de TTB plus petites peuvent être obtenues. En raison de TTB beaucoup plus longue, les résultats de perte de données (classées par la cause de pertes, comme dans la Figure 7.8) au cours de la phase de sauvegarde sont pire sans assistance. Une fois la sauvegarde est considérée complète sur les pairs à distance, les objectifs atteints de DLP et TTR assurent la même qualité de service dans tous les régimes, les taux de perte de données en raison de long TTR sont similaires.

Le stockage central coûteux (Figure 7.15(b)) agrandit rapidement au début des phases de sauvegarde dans le cas de sauvegarde assistée en raison des transferts lents vers les pairs ; ensuite il décroît comme des fragments additionnels sont stockés sur les pairs à distance. Après qu'un

pair atteint ses objectifs d'eDLP et d'eTTR, les transferts suivants de fragments vers des pairs à distance sont effectués pour la minimisation des coûts : stocker plus de fragments sur les pairs gratuitement, et effacer des fragments du serveur diminuent le coût payé pour le service de sauvegarde. Dans les systèmes de sauvegarde à l'assistance plus fragments sont stockés sur le serveur que dans les systèmes où seulement les réparations sont assistées : outre le stockage transitoire des réparations, les pairs qui ont des difficultés à télécharger leurs fragments dans un nombre nécessaire vers des pairs à distance maintiennent de l'espace de stockage sur le serveur. Le prix de "l'équité" du régime regroupés de sélection des pairs implique donc les coûts plus élevés, mais raisonnables, dans ces systèmes.

Dans notre conception du système, les transferts de données sont programmés au hasard en direction des pairs à distance dans le réseau. On a étudié cette politique de planification par l'analyse de modèles théoriques, et on a validé nos estimations analytiques sur la durée des transferts de fragments en simulant les paramètres de pratique.

Proposition 1.4 *On a proposé d'utiliser un algorithme efficace du problème de flot maximum pour calculer la planification optimale pour les cas hypothétiques où les séances en ligne des pairs sont connues préalablement. On a utilisé les résultats optimaux pour évaluer la politique de planification aléatoire, et on a proposé des paramètres pratiques dans lesquelles l'exécution des décisions au hasard est presque optimale.*

Pendant les phases de sauvegarde et de récupération, un pair planifie ses téléchargements de fragments. L'objectif est de trouver la planification avec la durée minimale de temps pour finir les transferts de fragments en satisfaisant les objectifs de la sauvegarde ou de la récupération. Afin de trouver la planification qui minimise le temps nécessaire pour le transfert de N fragments (appelé en tant que le problème *mintime*), on a déterminé le nombre maximal de fragments transférables pendant T créneaux horaires (noté comme problème $maxfrag$). Ensuite, la solution pour le problème initial est le plus petit T dans lequel N fragments peuvent être transférés.

Definition 22 *Les problèmes mintime et* $maxfrag$ *sont définis comme suit :* s *est une planification arbitraire,* $t(s)$ *donne la durée, et* $n(s)$ *fournit les fragments transférés de la planification* s*,*

$$mintime(N) = \min\{T \mid \exists s \ : \ t(s) = T \wedge n(s) \geq N\}; \tag{A.3}$$

$$maxfrag(T) = \max\{N \mid \exists s \ : \ t(s) \leq T \wedge n(s) = N\}. \tag{A.4}$$

Les deux problèmes sont liés étroitement de la manière suivante :

Proposition 20 $mintime(N) = \min\{T \mid maxfrag(T) \geq N\}$.

La formulation suivante de programmation linéaire en nombres entiers (ILP) est analogue à $maxfrag(T)$ (A.4).

Definition 23 *Le problème de planification en pleine connaissance maximise le nombre de fragments transmis dans un délai d'une durée donnée T. x_i^t est une variable qui code les décisions de la planification : le nombre de fragments à transférer vers le pair à distance i en tranche de temps t. La disponibilité des pairs à distance est donnée à titre contraintes : $a_i^t = 1$ si le pair à distance i est disponible en tranches de temps t, 0 sinon. Une autre contrainte est le nombre maximal de fragments m qui peuvent être placé sur chaque pair à distance. La solution de $maxfrag(T)$ peut être trouvée en résolvant le problème ILP suivant :*

$$
\begin{array}{lll}
\max \sum_{t=0}^{T} \sum_{i=1}^{I} x_i^t & & \textit{maximiser le nombre de fragments transmis} \\
s.t. & x_i^t = [0, \min(u, d_i)] & \textit{fragments transférables à un pair} \\
& x_i^t \leq m a_i^t & \textit{transfert aux pairs en ligne} \\
& \sum_{t=0}^{T} x_i^t \leq m & \textit{pas plus de } m \textit{ fragments sur un pair} \\
& \sum_{i=1}^{I} x_i^t \leq u & \textit{pas plus de } u \textit{ transferts dans un créneau horaire.}
\end{array}
$$

Afin de traduire le problème de planification au-dessus pour le cas de récupération, on écrit d au lieu de u et m est remplacé par le vecteur des nombres de fragments stockés sur chaque pair à distance.

Nous avons transformé le problème ILP ci-dessus à un problème de flot maximum. Le même problème donc devient résoluble en temps polynomial. On présente la formulation de flot maximum du problème $maxfrag(T)$ en Figure 6.1. Les nœuds *ts* i avec $i = 1, 2, \ldots, T$ représentent les intervalles de temps jusqu'à T ; les nœuds *peer* i avec $i = 1, 2, \ldots, I$ représentent les pairs à distance. *ts* i est relié à *peer* j si et seulement si pair j est en ligne au créneau horaire i. La capacité u des arcs donne la contrainte sur la bande passante montante du pair, la capacité m des arcs décrit le nombre maximal de fragments stockés sur chaque pair à distance. Le flot maximal de la *source* au *target* donne le plus grand nombre de fragments qui peuvent être téléchargés durant temps T. Similairement à la formulation ILP, la formulation du problème de récupération est obtenue si u est remplacé par d et m est remplacé par le nombre de fragments stockés sur chaque pair à distance respectivement.

On peut calculer $maxfrag(T)$ itérativement pour des valeurs de T croissantes ; Proposition 20 garantit que la première valeur T qui satisfait $maxfrag(T) \geq N$ est le résultat de notre problème de planification initial. Le problème initial, trouver une planification optimale qui minimise le temps de transfert de N fragments, peut être résolu en effectuant $O(\log T)$ calculs de flot maximum. Pour un réseau d'écoulement avec V nœuds et E arcs, le débit maximal peut être calculé avec la complexité du temps $O\left(VE \log\left(\frac{V^2}{E}\right)\right)$ [63]. Dans notre cas, lorsque nous avons I nœuds et une solution optimale de T créneaux horaires, V et E sont $O(I+T)$ et $O(IT)$ respectivement.

On a simulé les planifications optimales et aléatoire pour certains $I - n$ paramètres (en répétant la simulation 1000 fois, le scénario de planification aléatoire est effectué 1000 fois dans chaque cas). On a tracé la médiane des solutions optimales et aléatoires pour chaque $I - n$ cas dans la Figure 6.3. Comme le nombre de fragments à transférer augmente, la solution optimale se rapproche à la limite inférieure théorique, à $minTTB$. De plus, avec un plus grand ensemble de pairs à distance, le rendement de la planification aléatoire devient similaire à celui d'optimale, quel que soit le nombre de fragments à transférer. Par exemple, les valeurs de $n = 60$ et $I = 90$ sont suffisantes pour une sauvegarde complète avec une prolongation tolérable (environ 10%) de $minTTB$ en utilisant la planification aléatoire.

A.4.2 La sélection des pairs par l'utilisateur dans un système de sauvegarde à P2P

Propositions 2: [99,131–134] On a analysé un modèle de sélection des pairs dans les systèmes de sauvegarde à P2P où les utilisateurs ont la possibilité de choisir égoïstement leurs pairs à distance avec lesquels ils veulent échanger des fragments dans un schéma symétrique.

Dans notre modèle symétrique de système les utilisateurs choisissent "égoïstement" des pairs sur lesquelles ils stockent des données. Ces partenaires de stockage sont choisis en fonction de leurs caractéristiques (la disponibilité en ligne et la bande passante montante dédiée au système) qui sont reflétées par un seul paramètre, appelé le grade des pairs. En cas de volonté mutuelle, les deux utilisateurs offrent la même quantité d'espace de stockage l'un à l'autre.

Definition 24 *Le modèle de sélection des pairs est comme suite :*

- *Soit $\mathcal{I}$ l'ensemble des pairs qui participent dans le système ; $I = |\mathcal{I}| > 1$.*
- *Chaque pair $i \in \mathcal{I}$ divisechacun de ses objets de sauvegarde dans k fragments originaux, dont il crée $\hat{c}_i$ fragments redondants de la même taille, pour les stocker sur des pairs à distance différents.*
- *Ensuite, chaque pair établit un ensemble de liens, notée par n_i pour le pair i : $n_i = \{j \in \{\mathcal{I} \setminus i\}\}$, où pair i stocke un fragment sur pair j. On ne permet qu'un fragment d'un objet de sauvegarde d'être stocké sur le même pair à distance.*
- *La conception symétrique des échanges de fragments stipule que $i \in n_j$ si, et seulement si $j \in n_i$, alors les pairs i et j sont des partenaires de stockage. Par conséquent pair i doit partager une capacité locale de stockage équivalente à $|n_i|$ fragments, $|n_i| \leq \hat{c}_i$.*
- *Son grade $g_i = a_i u_i$ caractérise chaque pair i in$\mathcal{I}$. $\hat{g}_i$ représente le grade* sans-effort *de pair i, parce que les attributs qui le composent ne nécessitent pas une pression supplémentaire de i autre que son comportement normal.*

Le modèle de sélection des pairs est en ligne avec la conception présentée du système de sauvegarde à P2P : les échanges symétriques de fragments sont effectués avec des pairs à distance choisis basé sur leurs grades. La qualité du service est déterminée par les attributs des partenaires : leur nombre et leurs grades. Si l'une de ces valeurs augmente, la qualité du service croît aussi (jusqu'à une certaine limite). Le coût du service consiste à l'amélioration du grade du pair, s'il choisit d'augmenter sa contribution en ressources.

Proposition 2.1 *[99,131–134] On a proposé un modèle réaliste, mais analytiquement traitables par la théorie des jeux [105] pour décrire les gains (Définition 25) de l'utilisateur égoïste.*

Definition 25 *Le gain que tous les pairs maximisent lorsqu'ils établissent des liens à des pairs à distance, est composé de deux termes, la valeur du service et le coût des efforts :*

$$p_i = \min\left(|n_i|\underline{g}_i, 1\right) - (g_i - \hat{g}_i),$$

où $|n_i|$ désigne le nombre de partenaires de i et $\underline{g}_i = \min_{j \in n_i}(g_j)$ est le grade le plus bas parmi eux.

En outre, on a défini un jeu non coopératif, appelé du jeu de l'échange, fondé sur la fonction de gain pour refléter le contexte égoïste de la sélection des pairs conduite par l'utilisateur.

Definition 26 *Le jeu de l'échange est défini par la collection des stratégies de joueurs $\{\mathcal{S}_i \; \forall i \in \mathcal{I}\}$, et la fonction de gain p : $\{p_i \; \forall i \in \mathcal{I}\}$ défini sur la combinaison des stratégies ($p : \mathcal{S}_1 \times \cdots \times \mathcal{S}_I \to \mathbb{R}^I$). Une stratégie du joueur $i \in \mathcal{I}$ se compose d'un grade $g_i \in (0, 1]$ et d'un ensemble de liens n_i.*

Le jeu de l'échange est en équilibre si tous les pairs font des stratégies qui leur donnent le plus haut gain, étant donné les stratégies des autres pairs. Pour déterminer ces stratégies (de meilleure réponse), on dissèque le processus d'optimisation conjointe que les pairs stratégiques envisage : nous analysons les problèmes algorithmiques de sélection des pairs et de grade séparément. En décomposant le problème d'optimisation de chaque joueur dans le jeu pour trouver sa stratégie de meilleure réponse, on suppose que les décisions concernant la sélection de leurs grades et de leurs pairs à distance sont entrelacées. Pour une discussion plus simple d'abord on analyse la sélection des pairs comme un problème de "fixtures" stable [73].

Proposition 2.2 *[99, 131–134] On a proposé de simplifier le jeu de l'échange avec des grades fixés à un problème de couplage. Basé sur des préférences de sélection des pairs générées par la fonction de gain, on a montré que les liens sont créés entre des pairs avec des grades similaires.*

L'algorithme de "fixtures" stable [73] résout tous les problèmes de sélection des pairs qui surviennent au sein de l'optimisation entrelacée qui trouve l'équilibre du jeu de l'échange. Il s'agit, en fait, une conséquence directe des caractéristiques particulières des problèmes possibles : les préférences des joueurs sont directement déterminées par leurs gains, indirectement par les grades des autres pairs. Pour cette raison, pendant l'algorithme de "fixtures" stable les propositions pour construire un lien provenant d'un pair de grade haut sont toujours réciproqué par des pairs de grade bas.

Le phénomène de stratification où les pairs sont liés selon leur ordre de grade, résulte à une situation où un pair de grade haut n'a pas de partenaires d'un grade plus bas qu'un pair de grade bas. Par conséquent, des pairs de grade bas trouveraient moins de partenaire que leur capacité. Après avoir formulé le problème algorithmique de la sélection des pairs, on a étudié la

sélection de grade dans le jeu de l'échange. On a prouvé que la stratification crée des incitations pour les utilisateurs de grade bas afin d'améliorer leur contribution au système. Lorsque les utilisateurs égoïstes sont encouragés à augmenter leur contribution consacrée au système, en termes de disponibilité en ligne et de bande passante dédiée au système, la qualité globale du service offert par le système améliore sensiblement.

Proposition 2.3 *On a prouvé l'existence de l'équilibre dans le jeu de l'échange, et on a donné les stratégies d'utilisateur (les meilleures réponses) en ce qui concerne la sélection de grade et de pairs à distance.*

Proposition 21 *Si* $\min_{i\in\mathcal{I}} \hat{g}_i \geq \frac{1}{I-1}$, *alors la grande clique sans-effort est un équilibre possible : tous les pairs sont reliés. En général, la meilleure stratégie de grade est de rejoindre une clique selon le rang dans l'ordre de grade sans effort; ce pourrait nécessiter l'amélioration du grade sans effort de certains pairs.*

Les joueurs se réunissent en groupes, guidés par leurs gains basés sur la taille et le pire grade dans la clique. Si le groupe devient grand, beaucoup de pairs se regroupent ensemble, alors les joueurs pourraient obtenir meilleurs gains en excluant les pairs des grades pires. Si la variété des grades d'un groupe devient trop étroite, par exemple si des pairs de faibles grades ne sont pas admissibles, la qualité de service diminue en raison du nombre critique des pairs membres. Cette dualité est produite par la fonction de gain : les membres de bas grades diminuent le gain des membres de hauts grades dans la clique, par contre leur exclusion provoque une baisse de gain au joueurs restants en raison de la taille réduite de la clique. Dans l'équilibre des deux effets opposés sont équilibrés à l'intérieur des groupes.

Contrairement aux paramètres de grade sans effort hétérogènes, si chaque joueur a le même grade initial, ils ne sont pas incités à l'améliorer : si $\hat{g}_i = \hat{g}\ \forall i \in \mathcal{I}$, la meilleure stratégie de réponse est de $g_i = \hat{g}\ \forall i \in \mathcal{I}$.

A.4.3 Allocation dynamique distribuée du spectre

Propositions 3: [129, 135, 136] Nous avons étudié la possibilité d'attribution du spectre radio-électrique entre de multiples demandeurs dynamiquement d'une manière distribuée.

Les politiques actuelles d'attribution du spectre, c'est-à-dire les licences gouvernementales pour les bandes de fréquences vendues à long termes, ne sont pas efficaces parce que la planification du trafic de pointe entraîne la sous-utilisation temporelle durant les périodes creuses, en outre, les restrictions spatiales et spectrales de la réutilisation des fréquences sont trop rigides en raison de la politique de traitement des interférences, excluant de nombreuses possibilités d'exploitation de fréquences. L'émergence de nouvelles technologies permet l'attribution des bandes

de spectre pour les titulaires de licence avec différents paramètres spectrales, spatiales et temporelles, ce qui pourrait améliorer l'utilisation du spectre.

Definition 27

- *Le spectre radio, notée par F, est divisé en petits créneaux de fréquence homogènes de taille prédéfinie, désigné par f.*
- *Les loueurs de fréquence ($\mathcal{I}$), appelés les nœuds, sont des entités distinctes qui exploitent du spectre radioélectrique au fixe, et/ou aux positions géographiques confinées, par exemple, des stations de base du service des fournisseurs sans fil, des systèmes de radio privées.*
- *Les nœuds demandent des bandes de fréquence de taille donnée (q_i $\forall i \in \mathcal{I}$ exprimé en créneaux de fréquence) et ils sont prêts à payer une somme d'argent (appelée utilité $q_i u_i$ $\forall i \in \mathcal{I}$) pour cela.*
- *F_i désigne la bande de fréquences attribuées au nœud i $\forall i \in \mathcal{I}$, dont la taille est $|F_i|$ et $\mathcal{F}^f$ est l'ensemble des nœuds qui attribuent le créneau de fréquence f au sein de leurs bandes, c'est-à-dire $\mathcal{F}^f = \{i : f \in F_i, \forall i \in \mathcal{I}\}$.*
- *L'interférence peut se produire si le même créneau de fréquence est utilisé par plusieurs nœuds. Elle est définie comme le* maximal *rapport de signal à l'interférence plus bruit (SINR) mesuré sur la zone d'exploitation d'un nœud i et noté et approché par $\sum_{j \in \mathcal{I}} \omega_{ji}^f$, où soit ω_{ji}^f l'interférence causée par nœud j au nœud i sur f.*
- *Le niveau d'interférence maximal global que nœud i peut supporter provenant d'autres nœuds $j \neq i$ sur le créneau de fréquence f est désigné par α_i^f.*

Le niveau d'interférence dépend de nombreux aspects : la distance géographique entre les nœuds, leurs puissances de transmission, les technologies appliquées, les codages, les type d'émetteurs radio, etc. On modèle l'interférence par les relations ω et les seuils α, tous dépendant de la fréquence. Les nœuds allouent dynamiquement les bandes de fréquences en payant des frais l'un à l'autre et à l'autorité si ces aspects de fréquence le rendent nécessaire, c'est-à-dire la coexistence de nœuds sur un créneau de fréquence n'est pas possible.

Definition 28 *L'attribution de fréquence et les règles de tarification assurent que les nœuds arrivant séquentiellement peuvent exclure d'autres, si nécessaire en raison de contraintes d'interférence insupportable. À toute exclusion des deux parties interférentes, c'est-à-dire celui qui vient d'arriver et le locataire de fréquence de la bande de fréquence exécutent une vente aux enchères au second prix : les deux nœuds émettent leurs offres, le plus élevé gagne et paie la deuxième pari. Les résultats possibles sont les suivants.*

- *Le rachat à succès : si l'offre du nœud qui vient d'arriver est plus élevée, il paie l'autre offre à l'autorité et le nœud qui a loué la fréquence est exclu.*

- *La défense à succès : si l'offre du locataire de fréquence est plus élevée, l'autorité n'impose pas de taxe sur lui, et le nœud arrivant est exclu* implicitement.

Lorsque nœud i essaie d'exclure nœud j sur un certain créneau de fréquence f avec une offre b_i, si $b_i > b_j$, où b_j est l'offre de défense de j, alors j est exclue, et i paie b_j à l'autorité. Par ailleurs, $b_j \leq u_j - c_j^f$ où c_j^f représente la somme des dépenses de j payés pour les exclusions précédentes des nœuds interférents sur le créneau de fréquence f. En payant b_j à l'autorité, c_i^f augmente, c'est-à-dire $c_i^f := c_i^f + b_j$, ce qui réduit le budget de i pour les offres d'exclusion supplémentaire. Une tentative d'exclusion échoue si $b_i < b_j$: ni i ni j paie l'autorité et i est implicitement exclue dans ce cas.

L'attribution et la tarification distribuées rendent le système souple en termes d'allocation du spectre qui devient possible à tout moment sans une vente aux enchères centrales annoncées préalablement ou périodiquement. La règle de tarification entraine l'efficacité d'utilisation du spectre, soit les acheteurs avec des offres plus élevées obtiennent le droit d'utiliser le spectre des fréquences. De plus, elle assure l'équité malgré le fait que les exclusions sont unidirectionnelles : les nœuds qui provoquent des interférences hautes sont la cible de tentatives d'exclusion, mais les nœuds ayant une utilité importante peuvent obtenir un créneau de fréquence sans interférences tant qu'ils provoquent de hautes interférences pour les autres nœuds.

Definition 29 *Convivialité d'interférence signifie une haute tolérance d'interférence et basse interférence causée à d'autres nœuds. Nœud i est plus amical d'interférence k, si $\alpha_i^f > \alpha_k^f$ et $\omega_{ij}^f < \omega_{kj}^f$ et $\omega_{ji}^f < \omega_{jk}^f$ $\forall j \in \mathcal{I} \setminus i, k, f \in F$.*

Proposition 3.1 *[135,136] On a montré que les règles d'attribution et la tarification de la Définition 28 assurent la rationalité (le gain ne peut pas être négatif), la compatibilité aux incitations (les nœuds soumettent des offres de leur véritable utilité u), et en plus les nœuds moins amicales d'interférence paient relativement plus pour le spectre.*

Les nœuds exclus reçoivent une compensation de l'autorité ou des nœuds qui les excluent, par conséquent le gain $p_i^f = u_i^f - c_i^f$ (c_i^f est la somme des prix payés dans les exclusions précédentes) de tous les nœuds sur n'importe quel créneau de fréquence f est toujours non-négatif. En plus, la propriété de vérité des ventes aux enchères au second prix fait les participants proposer leurs utilités réelles.

À un rachat, le locataire i et le(s) nœud(s) intéressé(s) (soit j avec la plus grande utilité) vont jouer une vente aux enchères au second prix. Soit c_i^f le prix que i a payés pour le créneau de fréquence f qui est le sujet de la vente aux enchères. Aucune des utilités des nœuds ne sont connues, mais l'utilité du locataire actuel a une limite inférieure : son coût cumulé c_i^f. Si j décide donc de soumettre une offre au-delà de son utilité $u'_j > u_j$, et si $u'_j > u_i > u_j$ le prix à payer à l'autorité résulte dans un gain négatif.

Supposons que deux nœuds i et k ayant la même utilité veulent attribuer le même créneau de fréquence f au même temps et au même emplacement géographique. Soit $\alpha_i^f > \alpha_k^f$ et $\sum_{j\in\mathcal{I}} \omega_{ij}^f < \sum_{j\in\mathcal{I}} \omega_{kj}^f$. Si k alloue f, alors i pourrait également obtenir f à un prix inférieur : k est perturbé par des interférences d'autres nœuds jusqu'à α_k^f, donc la qualité de f serait suffisant pour i aussi. Si k a exclu d'autres nœuds afin de diminuer le niveau d'interférence dessous de son seuil, i pourrait également effectuer ces exclusions, car $u_i = u_k$. En plus, l'interférence provoquée par k pour les autres est plus élevée que celle de i, alors i recevrait de moins intensives tentatives de réaffectation depuis d'autres nœuds que k.

La séquence des "arrivées" des nœuds a une importance primordiale, mais les caractéristiques des nœuds prédéterminent en partie le succès de leurs attribution du spectre. Toutefois, la sélection d'une bande de fréquences à allouer, puis les nœuds interférant à exclure n'est pas simple.

Definition 30 *Les nœuds allouant un créneau de fréquence f sont classés dans les ensembles suivants par le nœud arrivant :*

- les nœuds interférants *: le groupe de nœuds dont l'exclusion par le nouveau nœud i est nécessaire afin d'assurer que l'l'interférence cumulative sur f est maintenue en dessous α_i^f : $\mathcal{D}_i^f \subseteq \mathcal{F}^f$, $\sum_{j\in\mathcal{F}^f\setminus\mathcal{D}_i^f} \omega_{ji}^f \leq \alpha_i^f$;*
- les nœuds excluants *: le groupe de nœuds depuis lesquels des tentatives d'exclusion sont attendus en raison de leur haute interférence perçue, augmenté par le nouveau nœud i en-dessus de leurs niveaux de tolérance : $\mathcal{E}_i^f \subseteq \mathcal{F}^f \setminus \mathcal{D}_i^f$, $\mathcal{C}_i^f = \mathcal{F}^f \setminus \mathcal{D}_i^f \setminus \mathcal{E}_i^f$ (le groupe des nœuds* co-existants*) $\forall k \in \mathcal{C}_i^f \sum_{j\in\mathcal{C}_i^f} \omega_{jk}^f \leq \alpha_k^f$ et $\forall k \in \mathcal{E}_i^f \sum_{j\in\mathcal{C}_i^f} \omega_{jk}^f > \alpha_k^f$.*

Proposition 3.2 *[129] On a montré que l'optimisation du groupe des nœuds interférants à exclure est un problème NP-complet. On a suggéré des stratégies heuristiques pour les exclusions de nœuds.*

Le problème est formulé comme suit. Soit $\{\omega_{ji}^f, u_j\}$ $\forall j \in \mathcal{I}$ et α_i^f, u. Est-ce qu'il y a $\mathcal{D}_i^f \subseteq \mathcal{I}$ qui satisfait $\sum_{j\in\mathcal{D}_i^f} \omega_{ji}^f \geq \sum_{j\in\mathcal{I}} \omega_{ji}^f - \alpha_i^f$ et $\sum_{j\in\mathcal{D}_i^f} u_j \leq u$? Ceci est équivalent au problème de sac à dos, ce qui est connu pour être NP-complet [61].

Être empêché par la difficulté du problème, on propose que l'exclusion des nœuds soit effectuée dans l'ordre croissant de leur "prix d'interférence", c'est-à-dire pour le nouveau nœud i : $\forall j \in \mathcal{D}_i^f$ et $\forall k \notin \mathcal{D}_i^f$ $\frac{u_j - c_j^f}{\omega_{ji}^f} \leq \frac{u_k - c_k^f}{\omega_{ki}^f}$. En outre, les tentatives d'exclusion ciblant le nouveau arrivant, devrait initialiser par des nœuds dans l'ordre croissant de leurs seuils de tolérance d'interférence, c'est-à-dire $\forall j \in \mathcal{E}_i^f$ et $\forall k \notin \mathcal{E}_i^f$-re $\alpha_j \leq \alpha_k$.

Les nouveaux nœuds égoïstes cherchent à allouer la bande de fréquence avec la taille requise pour des dépenses minimales pendant la durée de vie maximale attendue. Chaque nœud i peut effectuer des exclusions de son budget de u_i sur chacun de ses créneaux de fréquences, mais

entretemps ils doivent maintenir une offre de défense suffisante contre les tentatives d'exclusion. À l'exclusion, dans le pire des cas nœud i doit élever son offre b_i à u_j du nœud cible j (si le nœud j n'a pas exclu d'autres nœuds préalablement). Comme le nombre de nœuds qui devraient être exclus afin de satisfaire les exigences d'interférence de i grandit, sa compétitivité tombe avec $u_i - c_i$.

Afin de trouver la bande de fréquence la moins coûteuse, le nouveau arrivant i essaie de positionner sa F_i sur le spectre, de sorte que, d'une part, le coût de l'exclusion d'autres nœuds ($\mathcal{D}_i^f$) serait minimal ; d'autre part le coût de défense contre les exclusions d'autres nœuds ($\mathcal{E}_i^f$) coûterait le moins possible en moyenne sur les créneaux de fréquences de F_i.

Proposition 3.3 *[129] On a donné une condition nécessaire pour l'attribution réussie. Nœud i est capable d'allouer une bande de fréquences adéquates si $\exists F_i : |F_i| = q_i$ et $\forall f \in F_i$:*

$$u_i \geq \sum_{j \in \mathcal{D}_i^{f*}} \left(u_j - c_j^f\right) + \max_{j \in \mathcal{E}_i^{f*}} \left(u_j - c_j^f\right), \text{ahol}$$

$$\mathcal{D}_i^{f*} = \arg\min_{\mathcal{D}_i^f} \sum_{j \in \mathcal{D}_i^f} \left(u_j - c_j^f\right) \text{ et } \mathcal{E}_i^{f*} = \arg\min_{\mathcal{E}_i^f} \max_{j \in \mathcal{E}_i^f} \left(u_j - c_j^f\right),$$

à condition que le nouveau arrivant excluent d'abord les nœuds interférents, et puis les autres nœuds font des tentatives d'exclusions de nouveau arrivant en cas de besoin.

On a proposé un algorithme heuristique (dans l'Algorithme 7) avec des stratégies différentes de sélection de bandes de fréquence :

- *minimisant des coûts : cette stratégie cherche le plus bas coût attendu basé sur la sélection heuristique de nœuds ;*
- *consciente d'interférence : ne prend en compte* que *l'interférence cumulée* actuelle *sur chaque bande de fréquence ;*
- *délibérée : compte à la fois l'interférence cumulée* et *le reste du budget d'autres nœuds.*

Nous avons évalué numériquement le modèle proposé, et nous avons comparé les résultats des stratégies heuristiques dans un exemple simple (Figure 12.3). Le revenu de l'autorité diminue si les nœuds changent à la stratégie consciente d'interférence ou délibérée de la stratégie minimisant des coûts. Cela est dû à l'échec des tentatives d'allocation des nœuds type-4 (DVB-T). Tandis que les nœuds type-1 (GSM) peuvent obtenir une vie plus longue (Figure 12.3(d)) avec la stratégie consciente d'interférence, la différence est moins observable dans la Figure 12.3(f) malgré le fait que les nœuds type-4 sont encore plus punis dans ce dernier cas. Comme les nœuds tpye-2 (UMTS) et type-3 (UWB) expériencent une durée de vie similaire quelle que soit la stratégie de sélection de la bande de fréquence appliquée, la stratégie consciente d'interférence propose un juste compromis entre les revenus et la réussite d'allocation avec ces paramètres.

Les scénarios des simulations simples nous montrent que le système d'attribution du spectre dynamique distribué est une méthode appropriée à l'allocation du spectre efficace et flexible.

A.5 Application des résultats

Les résultats de notre recherche sur les systèmes de sauvegarde à P2P contiennent à la fois des résultats théoriques et pratiques. La combinaison présentée des modèles des théories de couplage et des jeux offre une nouvelle perspective dans les problèmes des systèmes distribués. Les indicateurs de performance définis et la nouvelle approche de calcul de redondances de données peuvent affecter d'autres modèles de système de stockage à P2P, pas uniquement des services de sauvegarde. En outre, notre prototype mis en œuvre peut servir à la base pour développer des applications pratiques de sauvegarde. Ces dernières peuvent être commercialisées et déployées rapidement sur les appareils des utilisateurs, sur le set-top-boxes d'abonnés des fournisseurs de service Internet, ou sur des réseaux d'entreprise.

L'analyse de l'attribution du spectre donne un aperçu sur les futurs projets potentiels de gestion des fréquences radio. On a montré un modèle de distribution enrichi par des incitations compatibles et avec les objectifs d'assurer l'équité et l'efficacité de l'utilisation du spectre. On a souligné la complexité algorithmique d'exclusions et de sélection de la bande. Le régime d'attribution et la tarification sont conçus pour les participants égoïstes, et ils fournissent la flexibilité temporelle et la capacité de réponse rapide et locales à toute variation de la demande de fréquences.

Remerciements

Le travail présenté dans les Sections A.4.1 et A.4.2 a été effectué à Eurecom. Je tiens à remercier Pietro Michiardi et Matteo Dell'Amico pour leur aide. La recherche démontrée en Section A.4.3 a été réalisée au Laboratoire HSN de l'Université des Technologies et Economiques de Budapest. Je suis reconnaissant pour les conseils fournis par Attila Vidács.