Design and Analysis of a Hidden Peer-to-peer Backup Market[*]

Sven Seuken[†], Denis Charles[‡], Max Chickering[‡], Mary Czerwinski[‡]
Kamal Jain[‡], David C. Parkes[†], Sidd Puri[‡], and Desney Tan[‡]

December, 2015

Abstract

We present a new market design for a peer-to-peer (P2P) backup application and provide a theoretical and experimental analysis. In domains such as P2P backup, where many non-expert users find markets unnatural or unexpected, it is often a pragmatic requisite to remove or *hide* the market's complexities. To this end, we introduce a new design paradigm which we call "hidden market design," and show, how a market and its user interface (UI) can be designed to hide the underlying complexities, while maintaining the market's functionality. We enable the P2P backup market using a virtual currency only, and we develop a novel market UI that makes the interaction for the users as seamless as possible. The UI hides or simplifies many aspects of the market, including combinatorial resource constraints, prices, account balances and payments. In a real P2P backup system, we can expect users to update their settings with a delay upon price changes. Therefore, the market is designed to work well even out of equilibrium, by maximizing the buffer between demand and supply. The main theoretical result is an existence and uniqueness theorem, which also holds if a certain percentage of the user population is price-insensitive or even adversarial. However, we also show that the more freedom we give the users, the less robust the system becomes against adversarial attacks. Furthermore, the buffer size has limited controllability via price changes alone and we show how to address this. We introduce a price update algorithm that uses daily aggregate supply and demand data to move prices towards the equilibrium, and we prove that the algorithm converges quickly towards the equilibrium. Finally, we present results from a formative usability study of the market UI, where we found encouraging results regarding the hidden markets paradigm. The market design presented here is implemented as part of a Microsoft research project and an alpha version of the software has been successfully tested.

Keywords: Market design, Peer-to-peer Systems, Backup, User Interface Design.

[†]School of Engineering & Applied Sciences, Harvard University, 33 Oxford Street, Cambridge, MA 02138, {seuken, parkes}@eecs.harvard.edu.
[‡]Microsoft Corporation, One Microsoft Way, Redmond, WA 98052, {cdx, dmax, marycz, kamalj, siddpuri, desney}@microsoft.com

1. Introduction

Reliable, inexpensive, and easy-to-use backup solutions are becoming increasingly important. Individual users and companies regularly lose valuable data because their hard drives crash, their laptops are stolen, etc. Already in 2003, the annual costs of data loss for US businesses alone was estimated to be $18.2 Billion [23]. With broadband connections becoming faster and cheaper, *online backup systems* are becoming more and more attractive alternatives to traditional backup. There are hundreds of companies offering online backup services, e.g., SkyDrive, Idrive, Amazon S3. Most of these companies offer some storage for free and charge fees when the free quota is exceeded. However, all of these services rely on large data centers and thus incur immense costs.

Peer-to-peer (P2P) backup systems are an elegant way to avoid these data center costs by harnessing otherwise idle resources on the computers of millions of individual users: all users must provide some of their resources (storage space, upload bandwidth, download bandwidth, and online time) in exchange for using the backup service. While the total network traffic increases with a P2P solution, the primary cost factors that can be eliminated are 1) costs for hard drives, 2) energy costs for building, running and cooling data centers[1], 3) costs for large peak bandwidth usage, and 4) personnel costs for computer maintenance. A study performed by Microsoft in 2008 showed that about 40% of Windows users have more than half of their hard disk free and thus would be suitable candidates for using a P2P backup system. Our own recent user study [21] found that many users are not willing to pay the high fees for server-based backup and more than half of our participants said they would consider using P2P backup instead. Thus, there is definitely a considerable demand for P2P backup applications. In fact, a series of P2P backup applications have already been deployed in practice (e.g., *Wuala*, *Allmydata*). A drawback of the the existing systems is, that all users are generally required to supply the resources space, upload and download bandwidth in the same ratios.

Our P2P backup system is novel in that it uses a market to allocate resources more efficiently than a non-market-based system could. Furthermore, we provide users with incentives to contribute their resources. This is in contrast to non-price based systems like BitTorrent for example, where numerous research has shown that without proper incentives, file availability rapidly decreases over time until most content finally becomes unavailable [19]. In our system, the relative market prices for the different resources function as compact signals of which resources are currently scarce, and properly motivate those users who value a specific resource least, to provide it to the system in a large quantity. Some users might need most of their own disk space to store large amounts of data and thus prefer to sacrifice some of their bandwidth. Other users might use their Internet connection a lot for services like VOIP or file-sharing and might have a high disutility if the quality of those services were affected. We allow different users with idiosyncratic preferences to provide different resource bundles, and we update prices regularly taking into account aggregate supply and demand of all resources.

The design of a P2P backup market involves a series of unusual challenges, in particular at the intersection of market design and user interface (UI) design. The first and biggest challenge is that users of a backup system do not expect to interact with a market in the first place, and might find a market a very odd concept in this domain. This raises the question of how to display prices to the users if they don't even know they are interacting with a market-based system. Furthermore, users cannot be expected to monitor account balances or to make payments to the system. This challenge arises in many domains, especially in many emerging electronic markets where thousands or millions of non-sophisticated users interact with market-based systems. While these markets often provide

---

[1]In 2008, data centers in the US were responsible for about 3% of the country's energy consumption. Note that a P2P backup system cannot only reduce costs but is also more environmentally friendly due to reduced carbon emissions.

large benefits to the users, they can also be unnatural or complex such that individuals may not have an easy time interacting with them. To address this challenge in a principled way, we introduce a new design paradigm which we call "hidden market design." When designing hidden markets, we attempt to minimize or "hide" the complexities of the market to make the interaction for the user as seamless as possible. A hidden market encompasses both, the design of a UI for the market and the design of the economics of the market. A P2P backup application is particularly well suited to illustrate the hidden markets paradigm because the application targets millions of technically unsophisticated users, in a domain where markets are very unexpected and where many users might find the use of real money unusual. Our proposed design hides many common market aspects from the users.

A second market design challenge arises from the fact that users will only infrequently interact with this market. They will not continuously update their settings, and thus, price changes will only affect supply and demand after a delay. As a consequence, the system will be out of equilibrium most of the time, while trades must be enabled at all times. The third challenge is the combinatorial aspect of the resource supply that is needed for the production of backup services. All users must provide a certain amount of all resources, even if they currently only consume a subset of them. For example, a user who only contributes storage space is useless to the system because no files could ever be sent or received from that user if no bandwidth is provided. We call these combinatorial market requirements the *bundle constraints* because only bundles of resources have value. Displaying the bundle constraints in a simple way is a major challenge for the UI design. Because many of these challenges are quite unusual, providing a simple method of interaction to the users, in a domain where they don't expect a market, requires the development of new techniques for UI and market design.

1.1. Outline and Overview of Results

We present the market and UI design for a P2P backup system and provide a theoretical and experimental analysis of its properties. In Section 2, we introduce the preliminaries of the P2P resource market. We enable the market using a virtual currency only, which avoids the various complications a real-world currency brings along (e.g., state, federal, and international banking laws) and also makes the system more natural to use. In Section 3, we first explain the hidden market design paradigm in more detail and then describe the various elements of the specific market UI we developed for the P2P backup system. In a real P2P backup system, we must expect a delay in users updating their settings upon price changes, and thus the system will be *out of equilibrium* most of the time. In contrast to previous work on data economies, the market is designed to work well even when *not* in equilibrium. In our system, users do not have to continuously update demand and supply and instead periodically choose bounds on their maximum supply and demand. We describe a new slider control, which simplifies the display of the bundle constraints and provides the users with a linear interaction with the system. These sliders guarantee that users can only choose supply ratios that satisfy certain constraints, which enables us to support the market equilibrium with linear prices. The UI exposes the effect of prices to users only implicitly, so as to avoid invoking a mental model of a monetary system, and it completely hides the users' account balances and the payments made in the system.

The economics of the market also involve some unusual design choices. In Section 4, we describe the market design in detail and list a series of properties of our system design that allow us to model the market as an exchange economy, even though production is happening. In Section 5, we begin the analysis of the market equilibrium by advancing a new equilibrium concept, the *buffer equilibrium*. Because the P2P backup market will be out of equilibrium most of the time, we must always have a certain buffer between supply and demand of all resources. We show that the buffer between supply and demand is maximal in the buffer equilibrium, which motivates it as a desirable target concept.

We prove that under very reasonable assumptions, the equilibrium is guaranteed to exist, and is unique. This result also holds if a certain percentage of the user population is price-insensitive or even adversarial. However, we show that the more freedom we give users in choosing their supply settings, the less robust the system becomes against adversarial attacks. Furthermore, we show that the size of the buffer in equilibrium cannot be controlled via price updates alone. We describe which changes in the UI would be necessary to give the market operator control over the buffer size. In Section 6, we introduce a price update algorithm that only requires system-wide supply and demand information to update prices. We prove that the algorithm converges linearly towards the buffer equilibrium when initial prices are chosen close enough to equilibrium prices. Finally, in Section 7, we present results from a formative usability study of our system, evaluating how well users can interact with the new hidden market UI. The results are encouraging and show promise for the hidden market paradigm.

1.2. Related Work

Ten years ago, the research projects OceanStore [14] and FarSite [3] already investigated the potential of distributed file systems using P2P. Both projects, however, did not take the self-interest of individual users into account and did not perform any kind of market design. More recently, researchers have looked at the incentive problem, often with the primary goal to enforce fairness (you get as much as you give). Samsara [5] is a distributed accounting scheme that allows for fairness enforcement. However, it does not enable a system where different users can supply resources in different ratios while maintaining fairness, which is the primary advantage of our market-based system.

The idea to use electronic markets for the efficient allocation of resources is even older than ideas regarding P2P storage systems. Already in 1996, Ygge et al. [25] proposed the use of computational markets for efficient power load management. In the last five years, grid networks and their efficient utilization have gotten particular attention [15]. Fundamental to these designs is that participants are sophisticated users able to specify bids in an auction-like framework. While this assumption seems reasonable in energy markets or computational grid networks, we are targeting millions of users with our backup service and thus we cannot assume that users are able and willing to act as traders on a market when they want to backup their files.

In the last three years, human computer interaction researchers have gotten more interested in topics at the intersection of UI design and economics. Hsieh et al. [11] test whether the use of markets in synchronous communication systems can improve overall welfare. Hsieh et al. [10] explore a similar idea in the domain of question and answer applications where users could attach payments to their questions. While their use of markets is similar in vein to our approach, i.e., using markets to most efficiently allocate resources as is standard in economics [9], in both papers they used a very explicit UI showing monetary prices to the users.

Satu and Parikh [18] compare live outcry market interfaces in scenarios such as trading pits and electronic interfaces. They draw a distinction between trying to blindly replicate the real world in the UI, and locating "defining characteristics" that must be supported. In our work, we adopt this philosophy and attempt to mask the unnecessary affordances in the hopes that the relevant ones become easier to use.

From the UI design point of view, the work that is closest to our approach is *Yoopick*, a combinatorial sports prediction market [8]. This application provides a very intuitive UI for trading on a combinatorial prediction market. The designers successfully hide the complexity of making bets on combinatorial outcomes by letting users specify point spreads via two sliders. This approach is very much in line with the hidden market paradigm.

On the theoretical side, the two papers most similar to our work are by Aperjis et al. [2] and

Freedman et al. [7]. They analyze the potential of exchange economies for improving the efficiency of file-sharing networks. While the domain is similar to ours, the particular challenges they face are quite different. They use a market to balance supply and demand with respect to popular or unpopular files. However, in their domain there is only one scarce resource, namely upload bandwidth, while we design an exchange market for multiple resources. Furthermore, their design does not attempt to hide any of the market aspects from the users.

There exist multiple P2P backup applications that are being used in practice and the application most similar to ours is *Wuala* (`www.wuala.com`). However, we know of no other P2P backup system that uses a market. In the other backup systems, the ratios between the supplied resources space, upload and download bandwidth are fixed, and the same across all users. The advantage of our market-based approach is the additional freedom we give the users. Allowing them to supply different ratios of their resources increases overall economic efficiency and makes the system more attractive for every user. Note that without using a market, this freedom would not be possible, because there would be no mechanism to incentivize the users to supply the scarce resources.

2. The p2p Resource Market: Preliminaries

Our system uses a hybrid P2P architecture where all files are transferred directly between peers, but a dedicated server coordinates all operations and maintains meta-data about the location and health of the files. The role of the server in this system is so small that standard load-balancing techniques can be used to avoid scaling bottlenecks.

Each user in the system is simultaneously a *supplier* and a *consumer* of resources. A peer on the consumer side demanding a service (backup, storage, or retrieval) needs multiple peers on the supplier side offering their resources (space, upload and download bandwidth, and online time). The production process of the server (bundling multiple peers and coordinating them) is essential, turning unreliable storage from individual peers into reliable storage. Each peer on the supplier side offers a different resource bundle while each peer on the consumer side gets the same product, i.e., a backup service with the same, high reliability.

One natural concern about P2P backup is that individual users have a much lower availability than dedicated servers. Thus, a P2P system must maintain a higher file redundancy to guarantee the same file availability as server-based systems. Simply storing multiple file copies would be very costly. Fortunately, we can significantly reduce the replication factor by using *erasure coding* [16]. The erasure code splits up a file into k fragments, and produces $n > k$ new fragments, ensuring that *any* k of the n fragments are enough to reconstruct the file. Using this technique, we can achieve the same high reliability as sever-based systems while keeping replication low. For example, if users are online 12h/day on average, using erasure coding we can achieve a file availability of 99.999% with a replication factor as low as 3.5, compared to simple file replication which would have a factor of 17.

The process for backing up files involves four steps. First, the user's files are compressed. Then the compressed files are automatically encrypted with a private key/password that only the user has access to (via Microsoft LiveID). Then, the encrypted file is erasure coded, and then the individual fragments are distributed over hundreds of peers. Using this process, the security of the P2P backup system can be made as high as that of any server-based system.

Table 1 describes the five high-level operations in the P2P system. Note that all of the system-level processes happen without user interaction. All the user has to do is initiate a backup operation, a retrieval operation, or delete his files when he wants to.

Operation	Description	Resources Required from Suppliers
Backup	When a user performs a backup, file fragments are sent from the consumer to the suppliers.	Download Bandwidth
Storage	Suppliers must persistently store the fragments they receive (until they are asked to erase them	Space
Retrieval	When a user retrieves a backup, file fragments are sent from the suppliers to the consumer.	Upload Bandwidth
Repair	When the server determines a backed up file to be unhealthy, the backup is repaired.	Download and Upload Bandwidth
Testing	If necessary, the server initiates test operations to gather new data about a peer's availability.	Download and Upload Bandwidth

Table 1: Operations and their Required Resources.

Prices, Trading & Work Allocation. All trades in the market are done using a virtual currency. Each resource has a price at which it can be traded and in each transaction the suppliers are paid for their resources and the consumers are charged for consuming services. Prices are updated regularly according to current aggregate supply and demand, to bring the system into equilibrium over time.

Trading is enabled via a centralized accounting system, where the server has the role of a bank. The server maintains an account balance for each user starting with a balance of zero and allows each user to take on a certain maximal deficit. The purpose of the virtual currency is to allow users to do work at different points in time while keeping all contributions and usages balanced over time. Users have a steady inflow of money from supplying resources and outflow of money from consuming services, which varies over time. In *steady state*, when users have been online long enough, their income must equal their expenditure. Users cannot earn money when they are offline but must still pay for their backed up files. Thus, their balance continuously decreases during that time. When using real money, we could simply charge users' credit cards as their balance keeps decreasing. However, as long as we do not use real money, the maximum deficit that users can take on must be bounded. Ultimately, it is a policy decision what happens when a user hits a pre-defined deficit level. Our system will first notify the users (via email and visually in the application) and present options to remedy the situation (e.g., increase supply). Failing this after a reasonable timeout period (e.g., 4 weeks), the users' backups will be deleted. The server is involved in every operation, coordinating the work done by the suppliers and allocating work to those users with the lowest account balances to drive all accounts (back) to zero over time. This is possible because users' steady-state income must equal their expenditure. Thus, when users have been online for a sufficient time period, their account balance is always close to zero.

3. The Hidden Market User Interface

The UI is an essential aspect of the market design because it defines the information flow between the user and the market. The server needs to elicit a user's individual preferences, and a user needs to "experience" the current market prices. However, direct preference elicitation methods (directly

asking the users for their valuations) are infeasible to implement because the amount of communication would be too high, but more importantly, because the majority of users are non-experts and would find such an interaction very complicated, unnatural, and cumbersome. To make the interaction for the user as easy as possible, we design a *hidden market UI* where we attempt to mask as much of the prices, account balances, trading constraints, etc. from the user as possible. To do this, we project a hidden market UI wrapped around the actual market to expose a simplified interface to the user (illustrated in Figure 1). The goal in designing this hidden market UI is to establish a feedback loop between the market and the user, without invoking a mental model of a monetary market.
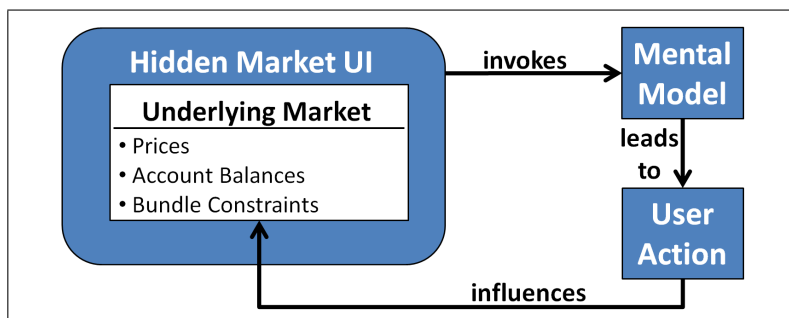

Figure 1: The hidden market UI wraps around the complex underlying market and exposes a simpler interface, invoking a particular mental model in the user, whose actions influence the market.

3.1. What You Give is What You Get

Figure 2 displays the market UI. The user can open this "settings window" to interact with the market. This window is separated into two sides: on the left side, the users can choose how much online backup space they need. On the bar chart the users can see how much they have already backed up and how much free online backup space they have left. On the right side of the window, the users can choose how much of their own resources they want to give up in return. On the top of the right side, the users see the storage path, i.e., where the file pieces from other users are stored on their own computers. Then, for each of the resources of space, upload and download bandwidth, there is a separate slider which the users can move to specify how much of that resource the system should maximally use.[2] Below the sliders the current average online time of the users is displayed.[3] Next to the online time information the system also tells the users the effect of leaving their computer online for one more hour per day (i.e., how much more online backup space they would get in return). This shall make the users aware of the important role of their online time: the longer the users are online, the more useful their supply of space, upload and download bandwidth becomes, and thus the higher their income.

To change anything about their settings, the users can drag the bar chart on the left side up or down, move any of the sliders on the right side, or change how often they are online. Both sides of the window are connected to each other, such that a change on either side affects and dynamically updates the values on the other side as well. The semantics of this connection are important: on average, users must pay for the total consumption chosen on the left side with the supply chosen on

---

[2]The maximum value for these sliders can be determined automatically: the limit for space is simply the free space on the users' hard drives; the bandwidth limits can be determined via speed tests.

[3]To change this value the users have to leave their computer online for more or fewer hours per day than they are currently doing, though we can conceive of schemes in which the application can directly control such settings as power savings and hibernate mode.
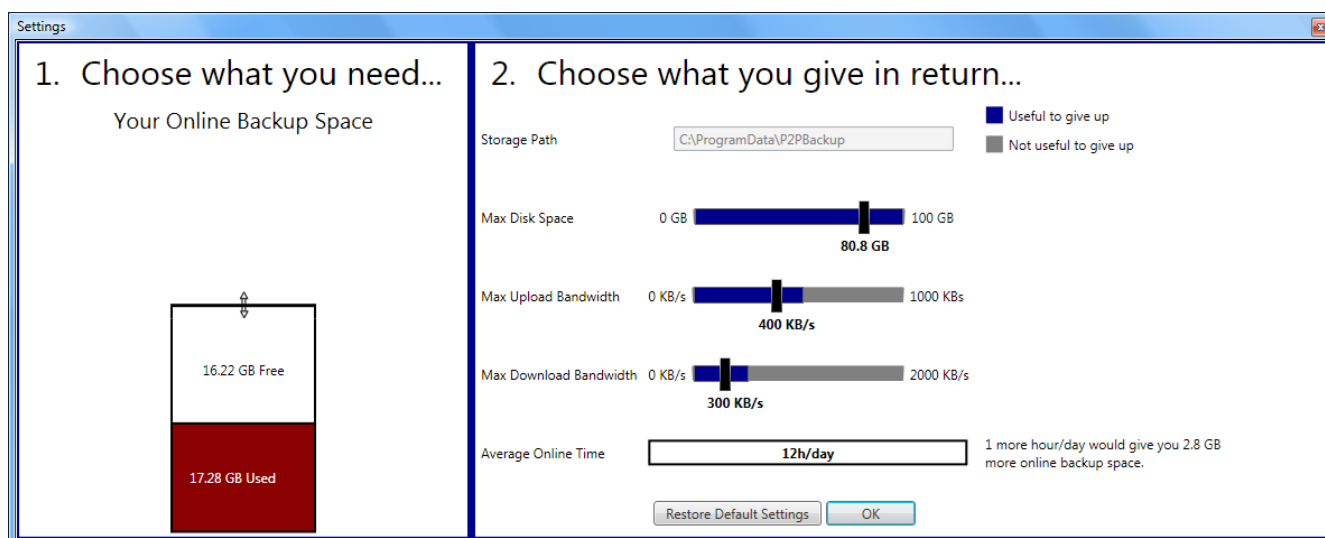
Figure 2: Screenshot of the advanced settings UI. On the left side, the user can choose the desired amount of online backup space. On the right side, the user can fine-tune the supply settings if desired. Account balances, prices and payments are hidden from the user.

the right side. If a user increases any of the sliders on the right then the bar chart on the left grows because the amount of free online backup space increases. If a user decreases a slider then the bar chart on the left shrinks, because the amount of free online backup space decreases. When a user directly drags the bar chart up or down to choose how much free online backup space he wants, then the three sliders on the right side move left or right, proportionally to their previous position.[4]

The UI allows users to express their idiosyncratic preferences over consuming backup services and supplying their resources. For example, if a user needs 20 GB of free online backup space, there are several different slider settings that allow this. Some users might specify to give more space and less bandwidth, others might specify it the other way around, depending on their available resources and individual preferences. Because a user's preferences can change over time this is not a task that can easily be automated. Note that we do not expect users to constantly adjust their settings. Rather, we expect users to choose settings that give them enough online backup space such that they do not have to worry about their settings for a while. However, as they near their quotas, the system will notify them (via an email and visually in the application). At that point, we expect most users to adjust their sliders again, according to their preferences and then current market conditions.

3.2. Combinatorial Aspects of the Market: Bundle Constraints

The first challenge regarding the hidden market design for this application is the combinatorial nature of the market, i.e., the problem that only bundles of resources are useful to the system. In general, the

---

[4]Note that in practice we expect roughly two categories of users: *basic users* will only ever use the left side of the window to choose how much online backup space they need. They either do not care about which resources they give up, or they do not even understand the meaning of upload bandwidth, download bandwidth, etc. The second category of users are the *advanced users*, i.e., those users that understand the meaning and relevance of giving up their own resources and want to control their supply. In a deployed system, the settings window would initially show the left side of the window and only upon clicking an "advanced" button would the right side appear.
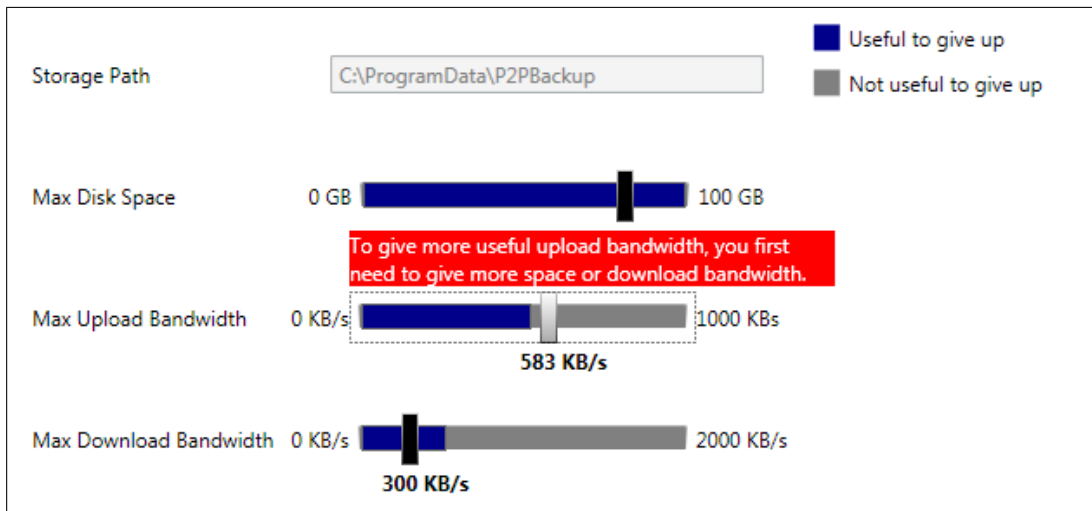
Figure 3: The new slider control provides an indirect visualization of the bundle constraints. When user provides more of one resource than is useful to the system, gets notified via a small popup window.

free online backup space increases when the users increase one of their sliders. However, this is only true for a subset of possible slider positions. In particular, if a user keeps increasing one slider towards the maximum while the other two sliders are relatively low, at some point the online backup space on the left might stop increasing. For example, if users limit their upload bandwidth to 5 KB/s, then increasing their space supply from 50 GB to 100 GB should not increase their online backup space. We would simply never store 100 GB on these users' hard disks because 5 KB/s would not be enough to have a reasonable retrieval rate for all of these file pieces. Thus, for the system to use the whole supply of 100 GB, the users would first have to increase their supply of bandwidth. An analogous argument holds true for other combinations of resources. For example, if a user wanted to give a lot of upload bandwidth but keep the supply of space low, then at some point giving more bandwidth would not be useful. Again, to make use of the download bandwidth, the system would need to store many file pieces on that user's computer which is not possible given the current low limit on space.[5]

Because of these "bundle constraints", we need users to respect certain supply ratios when choosing their supply settings. To provide the users with some visual information regarding how much supply of a resource is "useful to the system" given the current other slider settings, we augmented the traditional slider UI element, building the new slider control shown in Figure 3. The sliders are colored blue and gray, and the legend on the top right of the window explains the color coding. In the blue region, slider movements have an effect on the online backup space because setting the slider to any position inside that region means that the system can effectively use all of the supplied resource. The gray region of the slider is the region where slider movements no longer have an effect on the user's online backup space because giving that much of the resource is "not useful to the system," given the other settings. Because the colors and the legend might be difficult to understand or be overlooked, we also notify the user once the slider is moved from the blue into the gray region with a small pop up message that disappears once the mouse button is released (see Figure 3).

The color-coded sliders provide the user with all the necessary information about the bundle con-

---

[5]These bundle constraints only apply to space, upload and download bandwidth. For "availability" there is no minimum or maximum supply that is useful, independent of the other resources.

straints. When one slider is moved down, the blue regions on the other two sliders first stay the same and eventually decrease. Analogously, when one slider is moved up, the blue regions on the other two sliders first increase and eventually stop increasing. If a user sets the sliders in the same ratios as the system-wide usage of all resources, they are always inside the blue regions. However, requiring this exact ratio from all users is too restrictive, ignoring the system's flexibility in allocating work. For example, the system can allocate more repair and testing operations to users that prefer to give up lots of bandwidth instead of space. Furthermore, the system can estimate how often certain users access their backups and then send file fragments from "cold backups" to users who prefer to give up more storage space rather than bandwidth. To maximize overall efficiency, we make use of this flexibility, and allow every user to supply different ratios of their resources, within certain bounds of the system-wide ratios. In Section 4.2, we explain this concept more formally.

3.3. Exposing/Displaying Market Prices

Because the UI gives users some freedom in choosing their resource supply, we must price the resources correctly. In our system, prices are updated daily depending on aggregate demand and supply, moving the system into equilibrium over time. Without updating prices, we might have a supply shortage for some resources. For example, many users might decide to give lots of disk space and little bandwidth. To counteract a shortage of bandwidth, we would increase the price of bandwidth, incentivizing users to give more bandwidth instead of space. But for this mechanism to work, it is necessary that prices are at least indirectly exposed to users, so they can react and change their supply settings. For example, if the price for upload bandwidth went up relative to download bandwidth, then users might benefit from increasing their upload bandwidth supply a little and in return decreasing their download bandwidth supply a lot.

However, users don't expect monetary transactions in a backup application, which also renders "prices" an unnatural concept. This is why we have chosen to hide the prices in the UI as much as possible. In our UI, a user can "experience" the relative prices indirectly by moving the sliders while observing the bar chart on the left. If a user moves a slider a little and the bar chart only changes a little, this means that the current price for that resource is relatively low. If a user moves a slider a little and the bar chart changes a lot, this means that the current price for that resource is relatively high. This is one of the essential aspects of this hidden market UI: it allows us to communicate the current market prices to a user in a non-explicit way. In particular, users can be unaware of the price-based market underlying the backup system, and yet over time they will notice that for some resources they get more in return than for others. They can then choose the supply combination that is currently best given their preferences. Note that one of the market design goals was to implement a very simple pricing system to provide even non-expert users with a seamless interaction. We achieve this, despite the bundle constraints, by providing the users with a linear interaction with the system, as long as they move the sliders within the blue regions (the bar chart on the left moves up and down linearly when a user moves one of the sliders on the right). More specifically, we expose simple, linear prices to the users, and take care of the bundle constraints by restricting the choices they can make in the UI using the slider controls.

4. Market Design & Economic Model

In this section we introduce a formal economic model to describe the market design in detail and to allow for a theoretical economic analysis of the properties of the P2P market system. At all times, the model is formulated such as to represent the implemented system as closely as possible.

4.1. User Preferences

The economy comprises I users who are simultaneously suppliers and consumers. The set of commodities in the market is denoted $L = \{S, U, D, A, B, \Sigma, R\}$. The first four commodities are space (S), upload bandwidth (U), download bandwidth (D), and availability (A), which are the resources that users supply. The last three commodities are backup service (B), storage service (Σ), and retrieval service (R), which are the services that users consume. By slightly abusing notation, we sometimes use S, U, D, etc. as subscripts, and sometimes they denote the resource domain, e.g., for a particular amount of upload bandwidth u we require that $u \in U$. Each user i has a fixed endowment of the supply resources (defined by the user's hard drive and Internet connection), denoted $w_i = (w_{iS}, w_{iU}, w_{iD}, w_{iA}) \in S \times U \times D \times [0, 1]$.

The next aspect of the model is driven by our UI. Via the sliders, the user selects upper bounds for the supply vector, which we denote $X_i = (X_{iS}, X_{iU}, X_{iD}, X_{iA})$. In return for the supply X_i, the user interface shows the user the maximum demand of services, denoted $Y_i = (Y_{iB}, Y_{i\Sigma}, Y_{iR})$. In Figure 2 the user has currently chosen $X_{iS} = 80.8GB$, $X_{iU} = 400KB/s, X_{iU} = 300KB/s$ and $X_{iA} = 0.5$ as the maximum supply vector.

At any point in time, a certain set of resources from the user are being used, always less than X_i, and a certain set of services is being demanded. We denote user i's current supply as $x_i = (x_{iS}, x_{iU}, x_{iD}, x_{iA})$, and analogously user i's current demand for services as $y_i = (y_{iB}, y_{i\Sigma}, y_{iR})$. The user does not choose x_i and y_i directly via the UI. Instead, the server chooses x_i (obeying the bound X_i) such that user i can afford the current demand y_i which the user simply chooses by backing up files or retrieving them. Note that the UI displays the user's consumption vector in an aggregated way; i.e., instead of listing the services backup, storage, and retrieval separately, we simply display the currently used online backup space ($= 17.28GB$ in Figure 2) and the maximum online backup space that user could consume ($= 33.5GB$ in Figure 2).

In practice, users have a certain cost for opening the settings window and adjusting the settings. Instead of modeling this cost factor directly, we assume that when users open their settings window, they are planning ahead for the whole time period until they plan to open the settings window the next time. While a user might currently consume y_i, he plans for consuming up to Y_i the next time he opens the settings window. He then selects the supply vector X_i that he is willing to give up to get this Y_i. The user cares about how large the bounds on his supply are, because he has negative utility for giving up his resources. To make this more formal, we let $K_i = w_i - X_i$ with $K_i \in S \times U \times D \times A$, denote the vector of resources that the user *keeps*, i.e., his endowment minus the supply he gives up. Note that any changes to X_i translate into changes for K_i and vice versa because the endowment vector w_i is fixed. We only introduce K_i to define a preference relation that is monotone in all components, but we will use the supply vector X_i going forward. We can now specify the user's preference relation over all the resources he keeps, and the services he consumes: $\succeq_i (K_{iS}, K_{iU}, K_{iD}, K_{iA}, Y_{iB}, Y_{i\Sigma}, Y_{iR})$. We make the following assumptions which are all standard in economics (cf. [17], chapters 1-3):

Assumption 1. *Each user's preference relation $\succeq_i (K_{iS}, K_{iU}, K_{iD}, K_{iA}, Y_{iB}, Y_{i\Sigma}, Y_{iR})$ is (i) complete, (ii) transitive, (iii) continuous, (iv) strictly convex, and (v) monotone.*

Strict convexity requires strictly diminishing marginal rates of substitution between two goods, i.e., we need to compensate a user more and more with one good as we take away 1 unit of another good. This is a reasonable assumption because it represents a general preference for diversification. Monotonicity means that all commodities are "goods", i.e., if we give users more of any of the commodities, they are at least as well off as before.[6] Given complete, transitive, and continuous preferences, there

---

[6]Note that we do not assume *strict* monotonicity because we will later assume that service products are perfect

exists a utility function $u_i(K_i, Y_i) = u_i(K_{iS}, K_{iU}, K_{iD}, K_{iA}, Y_{iB}, Y_{i\Sigma}, Y_{iR})$ that represents the preference relation and this utility function is continuous (cf. [17], p.47).

As mentioned before, the only resource that is not subject to the combinatorial bundle constraints, is *availability*: as long as the user's availability is larger than zero, the other resources can be used. To simplify the economic model and pricing of resources, we introduce three new *composite resources* $\overline{S}, \overline{U}$, and $\overline{D}$, incorporating the user's availability into the other resources in the following way:

- $\overline{X}_{i\overline{U}} \in \overline{U} = X_{iU} \cdot X_{iA} \cdot 24 \cdot 60 \cdot 60$.
- $\overline{X}_{i\overline{D}} \in \overline{D} = X_{iD} \cdot X_{iA} \cdot 24 \cdot 60 \cdot 60$.
- $\overline{X}_{i\overline{S}} \in \overline{S} = \overline{\varphi}(X_{iS}, X_{iA}) \approx X_{iS} \cdot X_{iA} \cdot \text{overhead factor}$

Note that this notation denotes *composite* and *not* vector quantities. The definitions for the composite resources upload and download bandwidth are straightforward: we multiply the bound on bandwidth the user supplies (e.g., 300 KB/S) with the user availability $\in [0, 1]$ and then multiply it with 24 hours, 60 minutes and 60 seconds, to calculate how many KBs we can actually send to this user per day. The definition of $\overline{X}_{i\overline{S}}$ is a little more intricate because the user's availability does not enter linearly into the calculation. However, it enters monotonically, i.e., more availability is always better. Here, it suffices to know that the server can compute this function $\overline{\varphi}$ and convert a user's space and availability supply into the new composite resource.

We can now define user i's supply vector for the three composite resources: $\overline{X}_i = (\overline{X_{i\overline{S}}}, \overline{X_{i\overline{U}}}, \overline{X_{i\overline{D}}})$. The advantage of using these "availability-normalized" composite resources is that now, the supply from different users with different availabilities is comparable. For example, 1 unit of $\overline{S}$ from user i with availability 0.5 is now equivalent to 1 unit of $\overline{S}$ from user j with availability 0.9. Obviously, internally user i has to give much more space to make up for his lower availability, but in terms of bookkeeping, we can now operate directly with composites. We define the aggregate supply vector for the composite resources as $\overline{X} = \sum_i \overline{X_i}$, and analogously for Y, $\overline{x}$ and y. We make the following well-known observation (cf. [17], chapter 3) that will be useful later:

Observation 1. *The individual and aggregate supply and demand functions $\overline{X}_i$, Y_i, $\overline{X}$, and Y are homogeneous of degree zero.*

4.2. Production Functions and Slack Constraints

We have already mentioned the important role of the server in our market, i.e., that of combining resources from different suppliers into a valuable bundle. Note that the server is in fact the *only* producer in the market. One can think of this as if every user had access to the same production technology to convert input resources into services. This is crucial for our model and the economic analysis, because it allows us to define an *exchange economy* where the users only exchange factor inputs, despite the fact that production is happening in the market (cf. [17], pp. 582-584). Thus, for each service, we have one production function that defines how many input resources are needed to produce one unit of that service:

- Backup: $f^B : \overline{S} \times \overline{U} \times \overline{D} \to B$
- Storage: $f^\Sigma : \overline{S} \times \overline{U} \times \overline{D} \to \Sigma$
- Retrieval: $f^R : \overline{S} \times \overline{U} \times \overline{D} \to R$

---

complements, which violates strict monotonicity of preferences. We discuss this in more detail in Section 5.

These production functions are defined via the implementation of our system, i.e., the particular production technology that we implemented. For example, they are defined via the particular erasure coding algorithm that is being used, by the frequency of repair operations, etc. Thus, we can now specify a series of properties that these production functions guarantee due to our implementation:

System Property 1. *(Fixed Production Functions) Production functions are fixed and the same for all users.*

System Property 2. *(Additivity) The production functions are additive, i.e., $\forall l \in \{B, \Sigma, R\}$ and for any two resource vector $\vec{x_1}$ and $\vec{x_2}$: $f^l(\vec{x_1} + \vec{x_2}) = f^l(\vec{x_1}) + f^l(\vec{x_2})$.*

System Property 3. *(CRTS) The production functions exhibit constant returns to scale (they are homogeneous of degree 1), i.e., $\forall l \in \{B, \Sigma, R\}$, for any $\vec{x}$, and $\forall k \in \Re : f^l(k \cdot \vec{x}) = k \cdot f^l(\vec{x})$.*

System Property 4. *(Bijectivity) Each production function is bijective, and thus we can take the inverses:*

- $f^{B^{-1}} : B \to \overline{S} \times \overline{U} \times \overline{D}$
- $f^{\Sigma^{-1}} : \Sigma \to \overline{S} \times \overline{U} \times \overline{D}$
- $f^{R^{-1}} : R \to \overline{S} \times \overline{U} \times \overline{D}$

Property 1 holds because the server is the only producer, and because of the way we have defined the composite resources, with any differences between the users' availabilities already considered. Properties 2 (Additivity) and 3 (CRTS) hold because the erasure coding algorithm (which defines the production technology) exhibits these properties.[7] Property 4, the bijectivity of production, holds, because for each service unit, there is only one way to produce it. For example, to backup one file fragment, the erasure coding algorithm tells us exactly how many supplier fragments we need, and the server tells us how much repair and testing traffic we can expect on average per fragment. Furthermore, it is obvious that small changes in the input of the inverse production functions result in small changes in the output. More formally:

System Property 5. *(Continuity) The inverse production functions are continuous.*

Given the inverse functions for the individual services backup, storage, and retrieval, we can define an inverse function for a three-dimensional service vector $(b, \sigma, r) \in B \times \Sigma \times R$:

$$f^{-1}(b, \sigma, r) = f^{B^{-1}}(b) + f^{\Sigma^{-1}}(\sigma) + f^{R^{-1}}(r) \tag{1}$$

Given a demand vector y, we use $f^{-1}(y)$ to refer to the vector of supply resources that are necessary to produce y. Furthermore, we use $f_{\overline{S}}^{-1}(y)$, $f_{\overline{U}}^{-1}(y)$, and $f_{\overline{D}}^{-1}(y)$ to refer to the individual amounts of supply resources that are necessary to produce y.

We now formalize the flexibility we give our users in setting different ratios of their supplied resources. Because of the bundle constraints, a user cannot reduce his supply of resource k towards zero without affecting the supply of his other resources. To determine what ratios are acceptable, i.e.,

---

[7]Note that these two properties only hold approximately and not exactly, and only for file sizes above a certain threshold (approx. 1MB). Very small files are an exception and need special treatment in the implementation, because they are more expensive to be produced (again due to the erasure coding). We take care of this in the implementation by charging users more when they are backing up small files (essentially we have two sets of prices, one for normal files and one for small files).

useful to the system, we look at the system-wide usage of each resource k, i.e.: $f_k^{-1}(y)$. Certainly, if a user provides his resources in the same ratios as the system-wide usage, then all of his supply is usable. However, because the system has flexibility in allocating differnt kinds of work (repair/testing traffic vs. "cold backups" vs. "hot backups"), we can let the users' supply ratios deviate from the system-wide ratios to a limited degree. We let $\gamma > 1$ denote the amount of *slack* we allow users when setting their supply-side sliders. The corresponding *slack constraints*, lower-bounding the supply for resource k, constitute another system property:

System Property 6. *(Slack Constraints) Given slack factor γ, for each resource $k \in \{\overline{S}, \overline{U}, \overline{D}\}$, the user interface enforces the following minimum ratios of supplied resources:*

$$\forall i, \forall l \in \{\overline{S}, \overline{U}, \overline{D}\} \setminus \{k\} : \frac{\overline{X}_{ik}}{\overline{X}_{il}} \geq \frac{1}{\gamma} \cdot \frac{f_k^{-1}(Y)}{f_l^{-1}(Y)} \tag{2}$$

Note that the UI does not actually limit the range of the sliders according to the slack constraints. If a user chooses to supply too little of one resource such that a slack constraint is violated, then the system only uses/considers the maximum amount of the others resource such that the slack constraint binds. The UI visualizes this to the users via the blue regions, which are effectively indirect representation of the slack constraints, showing the user which settings are useful to the system. Thus, Equation 2 correctly models the slack constraints. If we actually limited the range of the sliders, then making larger changes with the sliders (which is necessary to explore the settings space) would be too tedious.

In our implementation, we set $\gamma = 2$. Thus, to give an example, if the system-wide usage ratio of space to upload bandwidth were 6, then each user would have to choose his individual settings with a ratio of space to upload bandwidth of at least $6 \cdot \frac{1}{2} = 3$, and the ratio of of upload bandwidth to space would have to be at least $\frac{1}{6} \cdot \frac{1}{2} = \frac{1}{12}$. How large we can set γ in practice depends on how flexible the system is in terms of allocating work (i.e., how many "cold" vs. "hot" backups there are, how much repair and testing traffic there is, etc.). In practice, the slack factor γ would have to be adjusted over time, when the distribution of work changes. This process could be automated, but here we are not going into the details of this process.

While every individual user is free to choose any supply setting within the slack constraints, of course the aggregate supply of each resource must always be large enough to satisfy current aggregate demand. But if every user chooses a supply setting such that the same slack constraint binds (e.g., every user minimizes his supply of upload bandwidth), then the system does not have enough supply of the corresponding resource. This is were the pricing algorithm comes into play: by regularly updating market prices according to current aggregate demand and supply, we balance the market such that different users will indeed supply *different* ratios of their resources. We discuss this aspect in more detail in Section 5 where we also prove that for any set of user preferences, there always exists a price vector that balances the market and guarantees enough supply of each individual resource.

4.3. Prices and Flow Constraints

In Section 3.2, we have explained how we display the bundle constraints to the users in the UI. The UI automatically enforces that the users only choose supply vectors that satisfy the slack constraints (cf. System Property 6) and this enables us to support an equilibrium with linear prices. We use $p = (p_{\overline{S}}, p_{\overline{U}}, p_{\overline{D}})$ for the prices for supplied composite resources, and $q = (q_B, q_\Sigma, q_R)$ for the demanded services. We require that in steady state, i.e., when a user has been online long enough, he can pay for

his consumption with his supply. In other words, his flow of supplied resources must be high enough to afford the flow of consumed services. We can express this *flow constraint* formally:

$$\overline{X}_i \cdot p = Y_i \cdot q \tag{3}$$

At the same time, the server allocates enough work to user i such that the user's current supply $\overline{x}_i$ is enough to pay for the demand y_i, which leads to a second flow constraint:

$$\overline{x}_i \cdot p = y_i \cdot q \tag{4}$$

We make the following assumption regarding the usage of resources in the system:

Assumption 2. *(Closed System & No Waste) We assume a closed system where no resources are entering or leaving the market, and we assume that no resources are wasted. Thus, the amount of resources required to produce the current aggregate demand is always equal to the current aggregate resource supply, i.e.: $f^{-1}(y) = \overline{x}$.*

Proposition 1. *Given a closed system and no waste of resources (Assumption 2), and given that production functions are additive (System Property 2), the payments from consumer i to the server must equal the payments from the server to the corresponding suppliers, i.e.:*

$$y_i \cdot q = f^{-1}(y_i) \cdot p \tag{5}$$

Proof. From the flow constraint in Equation 4 we know that $\overline{x}_i \cdot p = y_i \cdot q$. By summing over all users on both sides of the equation it follows that $\overline{x} \cdot p = y \cdot q$. Given Assumption 2, we know that $f^{-1}(y) = \overline{x}$. By plugging this into the previous equation, we get $f^{-1}(y) \cdot p = y \cdot q$. From the additivity of the production functions we know that this is equivalent to $\sum_i f^{-1}(y_i) \cdot p = \sum_i y_i \cdot q$. Because each transaction is treated equally in the system (every user is payed the same for the same resources), it follows that $f^{-1}(y_i) \cdot p = y_i \cdot q$. $\square$

Using Proposition 1, we can now re-write the flow constraints for user i as:

$$\overline{X}_i \cdot p = f^{-1}(Y_i) \cdot p \quad \text{and} \quad \overline{x}_i \cdot p = f^{-1}(y_i) \cdot p \tag{6}$$

Thus, from now on, we can omit the price vector q for demanded services and only need to consider price vector p[8], i.e., all what matters are the relative prices of the supply resources. Remember that the UI automatically calculates and adjusts the maximum demand vector Y_i for user i based on the user's supply bound $\overline{X}_i$. In practice, the maximum income is divided by the current average income of the user, and the resulting factor is multiplied with the user's current demand, giving us the maximum demand the user can afford:

System Property 7. *(Linear Prediction for Individual Demand) The system uses a linear demand prediction model for the calculation of a user's maximum demand Y_i:*

$$Y_i = \frac{\overline{X}_i \cdot p}{\overline{x}_i \cdot p} \cdot y_i = \lambda_i \cdot y_i$$

We make the following simplifying assumption:

---

[8]Going forward, please remember that multiplications with p are always dot products, and thus p showing up on the left and the right side of an equation does not cancel out.

Assumption 3. *(Linear Prediction for Aggregate Demand) We assume that with a large number of users, a linear demand prediction is also correct for the aggregate demand vectors, i.e.:*

$$\exists \lambda : Y = \lambda \cdot y$$

This assumption is justified because in practice, such a system would have a large number of users. Let n denote the number of users in the economy, let $Y^n = \sum_{i=1}^{n} Y_i$, $y^n = \sum_{i=1}^{n} y_i$, and let $\mu(\lambda_i)$ denote the mean of the distribution of the λ_i's. Given that the λ_i's are independent from the y_i's, it follows from the strong law of large numbers, that if the number of users n is large enough, then Y^n is linearly predictable by $\mu(\lambda_i) \cdot y^n$ along each dimension to any additive error. More specifically, for any ε and $\delta \geq 0$, for large enough n:

$$Pr[||Y^n - \mu(\lambda_i) \cdot y^n|| \leq \varepsilon] \geq 1 - \delta.$$

5. Equilibrium Analysis

A real-world instance of the P2P backup application would have thousands if not millions of users. Thus, the underlying market would be large enough so that no individual user had a significant effect on market prices. Consequently, users can be modeled as price-taking users and a *general equilibrium* model is suitable to analyze this market. Here we analyze a static equilibrium in which all users adjust their supply bounds to reach target demand bounds, i.e., whenever the price vector p is updated, user i chooses $\overline{X}_i(p)$ and $Y_i(p)$ such as to maximize his utility. While a user does not choose $\overline{x}_i$ (user i's supply that is currently used) and y_i (user i's current demand vector) directly via the UI, these quantities nevertheless depend on current prices, though indirectly, because $\overline{x}_i \leq \overline{X}_i$ and $y_i \leq Y_i$. Thus, while current demand and supply vectors $\overline{x}_i$ and y_i will vary much less with price changes, we must still model them as being dependent on prices, and we use $\overline{x}_i(p)$ and $y_i(p)$ to reflect that. Throughout this section, we assume that System Properties 1 through 7 and Assumptions 1 through 3 hold.

5.1. The Buffer Equilibrium

We begin this section by asking the question what the target equilibrium should be when we are updating prices. Note that there only is an equilibrium pricing problem in the first place because we give users the freedom to supply different ratios of resources. Without any slack, the UI would enforce that every user supplied the resources in the same ratios as system-wide demand for resources, and thus price changes would have no effect. But because we give our users the freedom to choose different supply ratios, we must update prices over time, to avoid situations where we don't have enough supply for a resource to satisfy current demand. But what should be our target?

A standard equilibrium concept in general equilibrium theory is the Walrasian equilibrium, which requires that demand equals supply such that the market clears. Certainly we want to have enough supply to satisfy current demand, i.e.:

$$\overline{x}(p) = f^{-1}(y(p)).$$

But remember that users are not continuously adjusting $\overline{x}_i$, and as a consequence, the system will be *out of equilibrium* most of the time. Thus, our goal should not be to clear the market in equilibrium, but instead to always have some excess supply of all resources, to make sure we can satisfy any demand even out of equilibrium. The larger the "buffer" between the current demand of resources,

i.e., $f^{-1}(y)$, and the maximum supply of resources, i.e., $\overline{X}$, the safer the system, i.e., the more "out of equilibrium" it can cope with before running into trouble. We will use this "size of the supply-side buffer" repeatedly and thus we define it more formally:

Definition 1. *(Size of the Supply-Side Buffer for a Resource) The size of the supply-side buffer for resource l is the ratio of maximum supply to current demand for that resource, and we denote this buffer with $\mathbb{B}_l(p)$:*

$$\mathbb{B}_l(p) = \frac{\overline{X(p)}_l}{f_l^{-1}(y(p))} \tag{7}$$

If we assume that the supply and demand for the individual resources have the same variance, then the best we can do to maximize the safety of the system out of equilibrium, is to maximize the size of the buffer across all three resources.[9] This naturally leads to the definition of the overall size of the supply-side buffer:

Definition 2. *(Overall Size of the Supply-Side Buffer) The size of the overall supply-side buffer $\mathbb{B}(p)$ is the smallest supply-side buffer across all resources, i.e.:*

$$\mathbb{B}(p) = \min_{l \in \{\overline{S}, \overline{U}, \overline{D}\}} \mathbb{B}_l(p) \tag{8}$$

Now the question is, which price vector maximizes the overall supply-side buffer. It is intuitive, that to maximize the overall supply-side buffer, the individual buffers must all be equal (otherwise we might update prices to decrease the largest buffer and increase the samllest buffer). This naturally leads us to the following definition of a "buffer equilibrium":

Definition 3. *(Buffer Equilibrium [Version 1]) A Buffer equilibrium is a price vector $p = (p_{\overline{S}}, p_{\overline{U}}, p_{\overline{D}})$, an aggregate supply vector $\overline{X}(p)$, and an aggregate current demand vector $y(p)$, such that the individual supply-side buffers are the same across all resources, i.e.:*

$$\mathbb{B}_{\overline{S}}(p) = \mathbb{B}_{\overline{U}}(p) = \mathbb{B}_{\overline{D}}(p) \quad \Leftrightarrow \quad \frac{\overline{X}_{\overline{S}}(p)}{f_{\overline{S}}^{-1}(y(p))} = \frac{\overline{X}_{\overline{U}}(p)}{f_{\overline{U}}^{-1}(y(p))} = \frac{\overline{X}_{\overline{D}}(p)}{f_{\overline{D}}^{-1}(y(p))} \tag{9}$$

It seems very reasonable to assume that, as we decrease the price for one resource k, the supply-side buffers for the other two resources will increase. Decreasing p_k makes it less attractive for the users to supply resource k, and makes it relatively more attractive to supply the other resources. If we make this assumption more formally, we can indeed prove that for the supply-side buffer to be maximal, the system must be in a buffer equilibrium, thus justifying the buffer equilibrium as a desirable target concept.

Assumption 4. *(Resource Buffers are Gross Substitutes) We assume that the individual buffer functions $\mathbb{B}_l(p)$ satisfy the gross substitutes condition, i.e., whenever p' and p are such that, for some k, $p_k' > p_k$ and $p_l' = p_l$ for $l \neq k$, we have $\mathbb{B}_l(p') < \mathbb{B}_l(p)$ for $l \neq k$.[10]*

---

[9]If we have specific information about the variance in the supply and demand of certain resources, we would want to target higher buffers on the resources with high variance and lower buffers on resources with low variance. This can easily be incorporated and would only lead to a slightly different equilibrium definition.

[10]Note that this assumption is similar to the more standard assumption that the excess demand function satisfies the gross substitute property, however, they are not equivalent. We assume that, as we decrease the price on one resource, the *ratio* between supply and demand for all other resources will increase, while the standard gross substitutes assumption states that the *difference* between supply and demand for all other resources will increase. Neither assumption implies the other, although both can be true simultaneously.

Proposition 2. *Given Assumption 4 (Resource Buffers are Gross Substitutes), when the overall supply-side buffer $\mathbb{B}(p)$ is maximal, then the market has reached a buffer equilibrium.*

Proof. We present a proof by contradiction. Let's assume that p is a price vector such that the overall supply-side buffer is maximal, but where the resource buffers are not the same across all resources as they must be in the buffer equilibrium. Assume that $k = \arg\max_{l \in \{\overline{S}, \overline{U}, \overline{D}\}} \mathbb{B}_l(p)$, i.e., the buffer for resource k is maximal across all resources. Now, we consider price vector p' where we have decreased the price of resource k slightly and kept the prices of the other resources constant, i.e., $p'_k < p_k$ and $p'_l = p_l \ \forall l \neq k$. Given that the individual resource buffers satisfy Assumption 4, we know that $\mathbb{B}_l(p') > \mathbb{B}_l(p)$, and due to homogeneity of degree zero, it also follows that $\mathbb{B}_k(p') < \mathbb{B}_k(p)$, i.e., the resource buffer size for k has decreased and both other resource buffer sizes have increased. Because of the continuity of users' preferences (Assumption 1) and the continuity of the inverse production function (System Property 5), it follows that $\overline{X}(p)$ and $f^{-1}(y(p))$ are continuous, and thus we can always find a small enough price change from p to p', such that the buffer for resource k is still maximal, but in the process we have increased the buffers for the other two resources. Thus, the overall supply-side buffer is larger for p' than it was before, i.e., $\mathbb{B}(p') > \mathbb{B}(p)$ which violates our assumption that the supply-side buffer with price vector p is maximal, which leads to a contradiction and completes the proof. $\qquad\square$

We have just shown that when the overall supply-side buffer is maximal, then the market has reached a buffer equilibrium. One concern might be that this does not automatically imply that the supply-side buffer will be maximal in *every* buffer equilibrium. However, we will show in Section 5.4 that under certain assumptions, the buffer equilibrium is unique, which removes this concern and implies that the buffer equilibrium is indeed a good target concept. Note that we truly believe that Assumption 4 is satisfied in our domain, and thus, the overall supply-side buffer is indeed maximal in the buffer equilibrium. However, we do not need this assumption going forward. We only used it to provide a formal motivation for the introduction and use of the buffer equilibrium concept, but all statements in the remainder of the paper are also true for the buffer equilibrium, without this assumption.

We now offer an alternative definition of the buffer equilibrium which relates it to the well-known concept of a Walrasian equilibrium:

Definition 4. *(Buffer Equilibrium [Version 2]) A Buffer equilibrium is a price vector $p = (p_{\overline{S}}, p_{\overline{U}}, p_{\overline{D}})$, an aggregate maximum supply vector $\overline{X}(p)$, and an aggregate maximum demand vector $Y(p)$, such that:*

$$\overline{X}(p) = f^{-1}(Y(p))$$

i.e., it is a Walrasian equilibrium defined on the supply and demand bounds chosen by the users.

It is easy to show that the two definitions for the buffer equilibrium are equivalent (proof provided in Section A of the Appendix):

Lemma 1. *Given Assumption 3 (Linear Prediction for Aggregate Demand), the 1. and 2. definitions of the Buffer Equilibrium are equivalent, i.e.:*

$$\mathbb{B}_{\overline{S}}(p) = \mathbb{B}_{\overline{U}}(p) = \mathbb{B}_{\overline{D}}(p) \quad \Leftrightarrow \quad \overline{X}(p) = f^{-1}(Y(p)).$$

5.2. Equilibrium Existence

In this section, we prove that a buffer equilibrium exists in our model. We let $\overline{L} = \{\overline{S}, \overline{U}, \overline{D}\}$ and we use l to index a particular composite resource. We define the vector-valued *relative-buffer function* $Z(p)$ which measures the relative buffer for each individual resource in the following way:

$$Z_l(p) = \frac{\overline{X}_l(p)}{f_l^{-1}(y(p))} - \left(\frac{\sum_k \frac{\overline{X}_k(p)}{f_k^{-1}(y(p))}}{|\overline{L}|}\right) \tag{10}$$

In words, the first term represents the supply to demand ratio of the particular good l. The second term represents the average supply to demand ratio, in our case averaged over the three goods storage space, upload and download bandwidth. Thus, $Z_l(p)$ represents how far the "buffer" between supply and demand for good l is away from the average buffer. We have reached a buffer equilibrium when the buffer is the same for all goods, i.e., when:

$$Z(p) = 0.$$

Lemma 2. *Given that users' preferences are strongly monotone with respect to supply resources, the relative-buffer function $Z(\cdot)$ has the following property: If $p^n \to p$, with $p \neq 0$ and $p_k = 0$ for some k, then for n sufficiently large*

$$\exists l : Z_l(p^n) > Z_k(p^n).$$

Proof. Because $p \neq 0$, for n large enough, there exists a resource l such that $p_l^n > 0$. As the price of resource $k \in \{\overline{S}, \overline{U}, \overline{D}\}$ goes towards zero, due to users' strictly convex and strongly monotone preferences for supply resources, they will supply less and less of k, and supply more of the other resources instead, at least of resource l whose price is bounded away from zero. However, because of the slack constraints, the users cannot reduce their supply of resource k towards zero, or increase their supply of resource l arbitrarily high. Let $\gamma > 1$ denote the slack factor we allow users when setting their preferences. The corresponding slack constraints (see System Property 6), lower-bounding the supply for resource k, are:

$$\forall l \in \overline{L} \setminus \{k\} : \overline{X_{ik}}(p^n) \geq \frac{1}{\gamma} \cdot \frac{f_k^{-1}(y(p^n))}{f_l^{-1}(y(p^n))} \cdot \overline{X_{il}}(p^n)$$

As $p^n \to p$ with $p \neq 0$ and $p_k = 0$, for n large enough, p_k^n will be sufficiently close to zero, such that each user i chooses to supply the minimal amount of resource k that is possible. Thus, at least with respect to *one* of the other resources l or m, the slack constraint will be binding, i.e.,:

$$\forall i : \overline{X_{ik}}(p^n) = \frac{1}{\gamma} \cdot \frac{f_k^{-1}(y(p^n))}{f_l^{-1}(y(p^n))} \cdot \overline{X_{il}}(p^n) \quad \vee \quad \overline{X_{ik}}(p^n) = \frac{1}{\gamma} \cdot \frac{f_k^{-1}(y(p^n))}{f_m^{-1}(y(p^n))} \cdot \overline{X_{im}}(p^n)$$

This does not mean that the slack constraint will be binding for the same resource l or m for every user. In fact, it is possible that user i will minimize his supply of resources k and l, while user j minimizes his supply of resources k and m. However, because every user contributes least to the supply-side buffer for resource k, this implies:

$$\exists l : \frac{\overline{X}_l}{f_l^{-1}(y(p^n))} > \frac{\overline{X}_k}{f_k^{-1}(y(p^n))}$$

and this implies that: $\exists l : Z_l(p^n) > Z_k(p^n)$. $\qquad\square$

Theorem 1. *A buffer equilibrium exists in the* P2P *exchange economy, given that users' preferences are continuous and strictly convex, monotone w.r.t. service products as well as strongly monotone w.r.t. to supply resources.*

Proof. Consider the relative buffer function $Z(p)$. We have noted in Observation 1 that $\overline{X(p)}$ and $y(p)$ are both homogeneous of degree zero, and this implies that $Z(p)$ is homogeneous of degree zero. Thus, we can normalize prices in such a way that all prices sum up to 1. More precisely, denote by

$$\Delta = \left\{ p \in \mathbb{R}_+^L : \sum_l p_l = 1 \right\}.$$

We can restrict our search for an equilibrium to price vectors in Δ. However, the function $Z(p)$ is only well-defined for price vectors in

$$\text{Interior } \Delta = \{ p \in \Delta : p_l > 0 \text{ for all } l \}.$$

To refer to price vectors in Δ that are not in the interior, we use:

$$\text{Boundary } \Delta = \Delta \setminus \text{Interior } \Delta.$$

The proof proceeds in six steps. In the first two steps, we define a correspondence $f(\cdot)$ from Δ to Δ, where we distinguish between price vectors in Interior Δ and in Boundary Δ. In step 3, we show that the correspondence is convex-valued. In step 4, we show that the correspondence is upper hemicontinuous. In step 5, we use all of these results and apply Kakutani's fixed point theorem to conclude that a p^* with $p^* \in f(p^*)$ is guaranteed to exist. Finally, in step 6 we show that any fixed point constitutes an equilibrium price vector. To facilitate notation, we will use q to denote price vectors in the set $f(p) \subset \Delta$.

Step 1: Construction of the correspondence $f(\cdot)$ for $p \in$ Interior Δ. For the definition of this correspondence, we put the resources in an arbitrary but fixed order, and index them by $i, j \in \{1, 2, 3\}$:

$$\forall p \in \text{Interior } \Delta : f(p) = \begin{cases} q \in \Delta, & \text{if } Z(p) = 0 \\ q \in \Delta : q_i = 1 \text{ if } i = \arg\min\{p_j : p_j = \min\{p_1, p_2, p_3\}\}, & \text{if } Z(p) \neq 0 \end{cases}$$

In words, if $Z(p) = 0$, i.e., when the buffer is the same for all resources, then the correspondence $f(\cdot)$ maps p to the set of all price vectors in Δ. If $Z(p) \neq 0$, then the correspondence maps p to a price vector $q \in \Delta$ where one component of q equals 1 and the other two components are equal to 0. More specifically, the correspondence sets that component $q_i = 1$ for which i is the smallest index of the price components p_j that are minimal among p_1, p_2 and p_3. Thus, when $Z(p) \neq 0$, then $f(\cdot)$ maps p to *exactly one* $q \in$ Boundary Δ. Only if $Z(p) = 0$, then $f(p) = \Delta$.

Step 2: Construction of the correspondence $f(\cdot)$ for $p \in$ Boundary Δ.

$$\forall p \in \text{Boundary } \Delta : f(p) = \{ q \in \Delta : q_i = 0 \text{ if } p_i > 0 \}$$

This correspondence maps p to all price vectors $q \in \Delta$ for which a component of q equals 0 when the corresponding component of p is positive. Because $p \in$ Boundary Δ, we know that for some i, $p_i = 0$, and thus $f(p) \neq \emptyset$. Furthermore, for at least one i, $p_i > 0$ and thus $q_i = 0$, which implies that no point from Boundary Δ can be a fixed point.

Step 3: The fixed-point correspondence is convex-valued. Consider first $p \in$ Interior Δ. If $Z(p) = 0$, then $f(p) = \Delta$, and because Δ is a simplex it is obviously convex. When $p \in$ Interior Δ and $Z(p) \neq 0$, then $f(\cdot)$ maps p to exactly one point in Δ, and thus $f(p)$ is trivially convex. Now, if $p \in$ Boundary Δ, then $f(p)$ is a subset of Δ, namely the set of price vectors q where one or two dimensions are equal to 0. These subsets of Δ are themselves simplices, and thus convex, and consequently $f(p)$ is convex.

Step 4: The correspondence $f(\cdot)$ is upper hemicontinous. To show upper hemicontinuity we have to prove that for any sequence $p^n \to p$ and $q^n \to q$ with $q^n \in f(p^n)$ it holds that $q \in f(p)$. We distinguish two cases: $p \in$ Interior Δ and $p \in$ Boundary Δ.

Step 4a: $p \in$ Interior Δ. Consider first a sequence $p^n \to p$ with $Z(p) = 0$. Thus, $f(p) = \Delta$ and for any sequence $q^n \to q$, it is trivially true that $q \in f(p)$. Now consider a sequence $p^n \to p$ with $Z(p) \neq 0$. Because users' preferences are continuous (Assumption 1), we know that $\overline{X(p)}$ and $y(p)$ are continuous, which implies the continuity of $Z(\cdot)$, and thus $\lim_{n \to \infty} Z(p^n) = Z(p)$. Because $Z(p) \neq 0$, for n large enough it must be that $Z(p^n) \neq 0$. Thus, when considering the sequence $p^n \to p$, for n large enough, we only have to consider the second case of the definition of $f(\cdot)$. Let $i^* = \arg\min \{p_j : p_j = \min \{p_1, p_2, p_3\}\}$. It holds that $\lim_{n \to \infty} \min\{p_1^n, p_2^n, p_3^n\} = \min\{p_1, p_2, p_3\}$. Thus, for n large enough, it must be that $\arg\min \left\{p_j^n : p_j^n = \min \{p_1^n, p_2^n, p_3^n\}\right\} = i^*$. Consequently, for n large enough, if $q^n \in f(p^n)$, then $q_{i^*}^n = 1$ which implies that $q_{i^*} = 1$. Thus, if $q^n \to q$ and for all n $q^n \in f(p^n)$, then $q \in f(p)$.

Step 4b: $p \in$ Boundary Δ. Consider $p^n \to p$ and $q^n \to q$ with $q^n \in f(p^n)$ for all n. We show that for any $p_l > 0$, for n sufficiently large we have $q_l^n = 0$ and thus $q_l = 0$ which implies that $q \in f(p)$. If $p_l > 0$, then $p_l^n > 0$ for n sufficiently large. If $p^n \in$ Boundary Δ, then $q_l^n = 0$ by the definition of the correspondence $f(p^n)$, and thus $q_l = 0$. If, however, $p^n \in$ Interior Δ, then Lemma 2 comes into play. Because $p \in$ Boundary Δ, for at least one k we have $p_k = 0$ and thus $p_k^n \to 0$. According to Lemma 2, for n large enough:

$$\exists l : Z_l(p^n) > Z_k(p^n)$$

i.e., there exists a resource l which has a larger buffer than resource k. Thus, $Z_k(p^n) \neq Z_l(p^n)$ and thus $Z(p^n) \neq 0$, which implies that we must only consider the second case of the definition of $f(p^n)$ for $p^n \in$ Interior Δ. If $q^n \in f(p^n)$, then for n large enough $q_k^n = 1$ for a resource k for which $p_k^n \to 0$. Because $p \in$ Boundary Δ, at least one and at most two components of p^n go towards 0. However, because $q^n \to q$, for n large enough, $q_k^n = 1$ for the same resource k, and thus $q_k = 1$, which implies that $q_l = q_m = 0$. Thus, for any $p_l > 0$, $q_l = 0$, which implies that $q \in f(p)$.

Step 5: A fixed point exists. The set Δ is a non-empty, convex and compact set and we have shown that $f(\cdot)$ is a correspondence from Δ to Δ that is convex-valued and upper hemicontinuous. Thus, we can apply Kakutani's fixed-point theorem which says that any convex-valued and upper hemicontinuous correspondence from a non-empty, compact and convex set into itself has a fixed point. We conclude that there exists a $p^* \in \Delta$ with $p^* \in f(p^*)$.

Step 6: A fixed point of $f(\cdot)$ is an equilibrium. Assume that p^* is a fixed point, i.e., $p^* \in f(p^*)$. As we have pointed out in step 2, no price vector from Boundary Δ can be a fixed point. Thus, it must be that $p^* \in$ Interior Δ. In step 1, we already saw that when $Z(p^*) \neq 0$, then $f(p^*) \subset$ Boundary Δ, which is incompatible with $p^* \in$ Interior Δ and $p^* \in f(p^*)$. Thus, for p^* to be a fixed point, it must hold that $Z(p^*) = 0$, and thus any fixed point p^* is an equilibrium price vector.

To summarize, we have shown that a fixed point always exists and that any fixed point is an equilibrium price vector. Thus, given the assumptions of the theorem, a buffer equilibrium is guaranteed to exist. □

5.3. Equilibrium Existence with Price-insensitive or Adversarial Users

So far, we have shown the existence of the buffer equilibrium when all users' preferences satisfy continuity, strict convexity, monotonicity and strong monotonicity w.r.t. supply resources, and update their settings accordingly upon price changes. In practice, however, some users might violate these assumptions, for example, because they don't notice price changes, or because they don't care enough to update their settings immediately. In more extreme cases, some users might purposefully harm the system and try to bring it out of equilibrium by updating their settings in the opposite way than what our assumptions would suggest. We call such users *adversarial* users. For example, an adversarial user could maximize his supply of those resources that currently have a very low price, and minimize his supply of those resources that currently have a very high price. Even though such behavior would certainly hurt the attacking user himself and thus could be called *irrational*, adversarial users do exist in practice, and robustness against adversarial attacks is a common concern.

In this section, we prove that a buffer equilibrium exists, even if a certain percentage of the user population is adversarial. For the analysis, we distinguish between *rational* users whose preferences satisfy our assumptions as before, and who update their settings accordingly upon price changes, and *adversarial* users, whose preferences must not satisfy our assumptions. To derive the maximum percentage of adversarial users that we can tolerate, the following analysis assumes that adversarial users update their settings in such a way as to maximally hurt the system, to bring it out of equilibrium.

We let R denote the set of rational users, and A denote the set of adversarial users. We let Y^R and X^R denote the demand and supply vector of the rational users, and Y^A and X^A denote the demand and supply vector of the adversarial users. Thus, $Y = Y^R + Y^A$ and $X = X^R + X^A$. As before, we let $\gamma > 1$ denote the system's slack constraint. We assume that the maximum demand of the rational users is at least C times larger than the maximum demand of the adversarial users, i.e., $Y^R \geq C \cdot Y^A$, and we derive a minimum bound for C to guarantee the existence of a buffer equilibrium.

As a first step, we show that under certain conditions, when the price of a resource k goes towards zero, there exists a resource $l \neq k$ with a strictly larger resource buffer than k (proof provided in the Appendix, in Section B).

Lemma 3. *Given slack factor γ and given that $Y^R \geq C \cdot Y^A$, if rational users' preferences are continuous and strictly convex, monotone w.r.t. service products as well as strongly monotone w.r.t. supply resources, and if $C > (\gamma^2 + \gamma)$, then for $p^n \to p$ with $p \neq 0$ and $p_k = 0$, for n sufficiently large:*

$$\exists l : \frac{\overline{X_k(p^n)}}{f_k^{-1}(y(p^n))} < \frac{\overline{X_l(p^n)}}{f_l^{-1}(y(p^n))}.$$

Equipped with Lemma 3, it is straightforward to prove the more general Theorem about equilibrium existence with adversarial users.

Theorem 2. *Given slack factor γ and given that $Y^R \geq C \cdot Y^A$, then a buffer equilibrium exists in the P2P exchange economy if $C > (\gamma^2 + \gamma)$ and the rational users' preferences are continuous and strictly convex, monotone w.r.t. service products as well as strongly monotone w.r.t. to supply resources.*

Proof. The theorem follows from the same proof as Theorem 1. The only necessary change is that in step 4b of the proof, instead of using Lemma 2 (which is only applicable when all users are rational), we use the more general Lemma 3. □

What Theorem 2 shows is that the more freedom we give the users in setting their supply (i.e., the larger the slack factor), the less robust is the system against adversarial attacks. This result is actually very relevant and useful for the designer of the P2P backup market. If there is reason to believe that a non-negligible fraction of the population will be adversarial or that many users will not update their prices in a rational way, then Theorem 2 tells the market designer exactly what to do. For example, if the market designer believes that at most 10% of the users will be adversarial, then the formula from the theorem tells us that as long as we give the users a slack factor of 2.5 or less, a buffer equilibrium is guaranteed to exist. In that respect, the theoretical equilibrium analysis actually has a very direct practical impact on the market design.

5.4. Equilibrium Uniqueness

Without any further restrictions on users' preferences, we cannot say anything about the uniqueness of the buffer equilibrium, because the substitution effect and the wealth effect could either go in the same or in opposite directions.[11] The standard equilibrium uniqueness proof for Walrasian equilibria resolves this by assuming that the aggregate excess demand function has the gross substitutes property for *all* commodities [1], which means that a price increase for one commodity causes an increase in the aggregate excess demand for all other commodities. However, that assumption is too strong for our domain for two reasons. First, and most importantly, for the demanded services, the gross substitutes property is violated in a P2P backup system. For example, if the price for storage increases, it is not reasonable to assume that users will now start deleting their backed up files and consume more backup or retrieval operations instead. The reason is simple: every file you back up is then being stored, and you can only retrieve files you have previously backed up. Thus, there are in fact strong complementarities between the demanded services in our domain, and to reflect this, we make the following assumption:

Assumption 5. *(Services are Perfect Complements) We assume that the aggregate demand function $Y(\cdot)$ has the perfect complements property, i.e.:*

$$\forall p, p' \in \mathbb{R}^3_{>0} : \exists \mu \in \mathbb{R} \ s.t. \ Y(p) = \mu \cdot Y(p')$$

A consequence of the perfect complements property is that price changes affect all dimensions of the aggregate demand vector equally. For an individual user, the Leontief utility function would induce the perfect complements property such that resources are consumed in fixed ratios. However, it bears emphasis that we assume perfect complements only for *aggregate* demand, rather than for individual demand, which is a much weaker assumption, and more reasonable due to the law of large numbers.

In contrast to service products, it seems reasonable to assume that supplied resources are substitutes in the sense that a user is happy to shift his supply from one resource to another as prices change. Yet, the strong assumption that supplied resources are gross substitutes might also not hold in our domain. Because services have the perfect complements property, and because services and supplied resources

---

[11]In an exchange economy, a price change always has two effects: first, it changes the relative prices between the goods, causing the substitution effect. Second, it can also change a user's wealth, because his supply might now be more or less valuable, which is called the wealth effect. Without further assumptions, nothing can be said about the net effect of a price change (cf. Sonnenschein-Mantel-Debreu Theorem, [17], pp. 598-606).

are coupled via the flow constraint $\overline{X}_i \cdot p = f^{-1}(Y) \cdot p$, price changes can also have non-substitution effects on the supply of resources. For example, when the price for a resource is decreased, it is not a priori clear that the supply for that resource goes down. It might be, that due to this price decrease, the system just became much more attractive for many users, so that they significantly increase their demand and thus also their supply (of all resources). Thus, we don't want to make assumptions regarding the specific directions of change in the supply and demand functions. We only make an assumption regarding how price changes affect the relative ratios of supplied resources to each other:

Assumption 6. *(Relative Supply Resources are Gross Substitutes) We assume that the aggregate supply function $\overline{X}(p)$ has the relative gross substitutes property, i.e., whenever p' and p are such that, for some k, $p'_k > p_k$ and $p'_l = p_l$ for $l \neq k$, we have $\frac{\overline{X}_k(p')}{\overline{X}_l(p')} > \frac{\overline{X}_k(p)}{\overline{X}_l(p)}$.*

Note that both assumptions are relatively weak. Upon a price decrease for good k, the aggregate supply for k can go up or down, and the demand for all services can also go up or down. All we assume is that when the price for good k is decreased, the relative supply of good k to the other goods decreases, and the demand for services moves up or down proportionally. With these two assumptions, we can now prove that the buffer equilibrium is unique:

Theorem 3. *The buffer equilibrium is unique, given that the aggregate demand function satisfies the perfect complements property (Assumption 5), and that the aggregate supply function satisfies the relative gross substitute property (Assumption 6).*

Proof. Because we make different assumptions regarding the supply and demand sides of our economy, we first separate the supply and demand aspects by introducing an alternative description of the buffer equilibrium:

$$\overline{X} = f^{-1}(Y) \tag{11}$$

$$\Leftrightarrow \left(\overline{X}_{\overline{S}}, \overline{X}_{\overline{U}}, \overline{X}_{\overline{D}}\right) = \left(f_{\overline{S}}^{-1}(Y), f_{\overline{U}}^{-1}(Y), f_{\overline{D}}^{-1}(Y)\right) \tag{12}$$

$$\Leftrightarrow \left(1, \frac{\overline{X}_{\overline{U}}}{\overline{X}_{\overline{S}}}, \frac{\overline{X}_{\overline{D}}}{\overline{X}_{\overline{S}}}\right) = \left(1, \frac{f_{\overline{U}}^{-1}(Y)}{f_{\overline{S}}^{-1}(Y)}, \frac{f_{\overline{D}}^{-1}(Y)}{f_{\overline{S}}^{-1}(Y)}\right) \tag{13}$$

$$\Leftrightarrow \left(\frac{\overline{X}_{\overline{U}}}{\overline{X}_{\overline{S}}}, \frac{\overline{X}_{\overline{D}}}{\overline{X}_{\overline{S}}}\right) - \left(\frac{f_{\overline{U}}^{-1}(Y)}{f_{\overline{S}}^{-1}(Y)}, \frac{f_{\overline{D}}^{-1}(Y)}{f_{\overline{S}}^{-1}(Y)}\right) = 0 \tag{14}$$

We define a new vector-valued function $g(p) = \left(g_{\overline{U}}(p), g_{\overline{D}}(p)\right)$:

$$g_{\overline{U}}(p) = \left(\frac{\overline{X}_{\overline{U}}}{\overline{X}_{\overline{S}}} - \frac{f_{\overline{U}}^{-1}(Y)}{f_{\overline{S}}^{-1}(Y)}\right) \quad \text{and} \quad g_{\overline{D}}(p) = \left(\frac{\overline{X}_{\overline{D}}}{\overline{X}_{\overline{S}}} - \frac{f_{\overline{D}}^{-1}(Y)}{f_{\overline{S}}^{-1}(Y)}\right),$$

which naturally leads to a new equilibrium definition that is equivalent to Definitions 3 and 4:

Definition 5. *(Buffer Equilibrium [Version 3]) A buffer equilibrium is a price vector p and $g(p)$ such that*

$$g(p) = \begin{pmatrix} 0 \\ 0 \end{pmatrix}.$$

We have simplified the problem of finding equilibrium prices to finding the root of the function $g(p)$. Because $\overline{X}(p)$ and $Y(p)$ are homogeneous of degree zero, $g(p)$ is also homogeneous of degree

zero, which implies that collinear price vectors are equivalent, i.e., $\forall \lambda > 0 : g(p) = g(\lambda \cdot p)$. Thus, showing uniqueness of the buffer equilibrium is now equivalent to showing that $g(p) = 0$ has at most one normalized solution. Now, let's assume that $g(p) = 0$, i.e., p is an equilibrium price vector. We show that for any p', $g(p') \neq 0$ unless p and p' are collinear. Because of Assumption 5 (the aggregate demand function has the perfect complements property), a price change affects all dimensions of the demand function equally, i.e., $\exists \mu \in \mathbb{R} : Y(p) = \mu \cdot Y(p')$. Because the production function is bijective and exhibits constant returns to scale, this implies that $f^{-1}(Y(p)) = \mu \cdot f^{-1}(Y(p'))$. Thus, $\forall p, p' \in \mathbb{R}^3_{>0} : \frac{f_{\overline{U}}^{-1}(Y(p))}{f_{\overline{S}}^{-1}(Y(p))} = \frac{f_{\overline{U}}^{-1}(Y(p'))}{f_{\overline{S}}^{-1}(Y(p'))}$, i.e., changes in the demand function $Y(\cdot)$ due to price changes do not affect $g(\cdot)$. Consequently, we only have to consider changes in the supply function $\overline{X}(\cdot)$. Now consider a price vector p' that is not collinear with p. Because of the homogeneity of degree zero, we can assume that $p' \geq p$ and $p_l = p'_l$ for some l. We now alter the price vector p' to obtain a price vector that is collinear to p, and argue about how $g(\cdot)$ changes in the process. We distinguish between three cases:

Case 1: $l = \overline{S}$, i.e., $p'_{\overline{S}} = p_{\overline{S}}$. First, we generate a price vector p'' that is collinear to p, by linearly increasing all components of p until the next two price components are equal, i.e., $p''_k = p'_k$ for $k \neq \overline{S}$. We assume that $k = \overline{D}$ (the case where $k = \overline{U}$ is completely symmetric) such that:

$$p'_{\overline{U}} \geq p''_{\overline{U}} \tag{15}$$
$$p'_{\overline{D}} = p''_{\overline{D}} \tag{16}$$
$$p'_{\overline{S}} \leq p''_{\overline{S}} \tag{17}$$

with at least one of the inequalities being strict. Now we alter p' to obtain p'' in two steps. In the first step, we decrease (or keep unaltered) $p'_{\overline{U}}$ until it equals $p''_{\overline{U}}$. In the second step, we increase (or keep unaltered) $p'_{\overline{S}}$ until it equals $p''_{\overline{S}}$. Because p' and p'' were not collinear, we have changed the price vector in at least one step, and because of Assumption 6, the relative ratio between $\overline{X}_U$ and $\overline{X}_{\overline{S}}$ has decreased in at least one step and has never increased, such that:

$$\frac{\overline{X}_{\overline{U}}(p')}{\overline{X}_{\overline{S}}(p')} > \frac{\overline{X}_{\overline{U}}(p'')}{\overline{X}_{\overline{S}}(p'')} = \frac{\overline{X}_{\overline{U}}(p)}{\overline{X}_{\overline{S}}(p)}$$

Thus, the first term in $g_{\overline{U}}(\cdot)$ has changed, and the second term stayed constant, and $g(p') \neq g(p) = 0$.

Case 2: $l = \overline{U}$, i.e., $p'_{\overline{U}} = p_{\overline{U}}$. First, we generate a price vector p'' that is collinear to p, by linearly increasing all components of p until $p''_k = p'_k$ for $k \neq \overline{U}$. Now we differentiate between two cases:

Case 2a: $k = \overline{D}$ such that:

$$p'_{\overline{S}} \geq p''_{\overline{S}} \tag{18}$$
$$p'_{\overline{D}} = p''_{\overline{D}} \tag{19}$$
$$p'_{\overline{U}} \leq p''_{\overline{U}} \tag{20}$$

with at least one of the inequalities being strict. The remainder of the proof for this case is analogous to the one fore case 1.

Case 2b: $k = \overline{S}$ such that:

$$p'_{\overline{D}} \geq p''_{\overline{D}} \tag{21}$$
$$p'_{\overline{S}} = p''_{\overline{S}} \tag{22}$$
$$p'_{\overline{U}} \leq p''_{\overline{U}} \tag{23}$$

with at least one of the inequalities being strict. Analogously to the proof for case 1, we can show that

$$\frac{\overline{X}_{\overline{U}}(p')}{\overline{X}_{\overline{D}}(p')} < \frac{\overline{X}_{\overline{U}}(p'')}{\overline{X}_{\overline{D}}(p'')} = \frac{\overline{X}_{\overline{U}}(p)}{\overline{X}_{\overline{D}}(p)} \tag{24}$$

For the rest of the proof for this case, we construct a contradiction. Assume that p' is also an equilibrium price vector such that $g(p') = 0$. Because the second term in $g_{\overline{U}}$ and $g_{\overline{D}}$ respectively does not change upon price changes, this implies that:

$$\frac{\overline{X}_{\overline{U}}(p')}{\overline{X}_{\overline{S}}(p')} = \frac{\overline{X}_{\overline{U}}(p)}{\overline{X}_{\overline{S}}(p)} \tag{25}$$

$$\text{and} \quad \frac{\overline{X}_{\overline{D}}(p')}{\overline{X}_{\overline{S}}(p')} = \frac{\overline{X}_{\overline{D}}(p)}{\overline{X}_{\overline{S}}(p)} \tag{26}$$

From Equation (25) it follows that $\overline{X}_{\overline{U}}(p') = \frac{\overline{X}_{\overline{U}}(p)}{\overline{X}_{\overline{S}}(p)} \cdot \overline{X}_{\overline{S}}(p')$ and from (26) it follows that $\overline{X}_{\overline{D}}(p') = \frac{\overline{X}_{\overline{D}}(p)}{\overline{X}_{\overline{S}}(p)} \cdot \overline{X}_{\overline{S}}(p')$. If we put these two results together we get:

$$\frac{\overline{X}_{\overline{U}}(p')}{\overline{X}_{\overline{D}}(p')} = \frac{\overline{X}_{\overline{U}}(p) \cdot \overline{X}_{\overline{S}}(p') \cdot \overline{X}_{\overline{S}}(p)}{\overline{X}_{\overline{S}}(p) \cdot \overline{X}_{\overline{D}}(p) \cdot \overline{X}_{\overline{S}}(p')} = \frac{\overline{X}_{\overline{U}}(p)}{\overline{X}_{\overline{D}}(p)}$$

and this contradicts Equation (24). Thus, $g(p') \neq 0$.

Case 3: $l = \overline{D}$, i.e., $p'_{\overline{D}} = p_{\overline{D}}$. The proof for this case is analogous to the proof for case 2.

In summary, in all three cases we established that $g(p') \neq g(p) = 0$ which shows that p' is not an equilibrium price vector and concludes the equilibrium uniqueness proof. $\square$

5.5. (Un-)Controllability of the Supply-Side Buffer

So far we have shown under what conditions the buffer equilibrium exists and when it is unique. In practice, however, the system will be out of equilibrium most of the time, because users do not continuously adjust their settings, and thus price changes will only affect supply and demand after a delay. This is why in Section 5.1, we have motivated the buffer equilibrium as a desirable target: the buffer between current demand and maximum supply of resources gives the system a certain safety for when it is out of equilibrium. To make sure we can always satisfy new incoming demand, we might like to have at least 25% more supply than current demand, i.e., $\overline{X} \geq 1.25 \cdot f^{-1}(y)$. Unfortunately, the uniqueness of the buffer equilibrium (Theorem 3) has an immediate consequence regarding the limited controllability of the buffer equilibrium:

Corollary 1. *(Limited Controllability of the Market) Given Assumptions 5 and 6, the market operator cannot influence the size of the buffer in the buffer equilibrium by adjusting market prices.*

It turns out that the limited controllability of the buffer equilibrium remains, even without the assumptions that service are perfect complements and that relative supply resources are gross substitutes, thereby strengthening the result from Corollary 1:

Proposition 3. *If each individual user i has a limited planning horizon in that he chooses not to give himself more than a demand-side buffer of λ_i, then there exists a $\Lambda \in \mathbb{R}_{>1}$ such that the market operator cannot achieve a buffer equilibrium with buffer size Λ by adjusting market prices.*

Proof. For the proof we construct a simple counterexample. We choose a Λ such that $\forall i : \Lambda > \lambda_i$. And we let $\lambda_i^* = \max_i \lambda_i$. Now:

$$\forall i : Y_i = \lambda_i \cdot y_i \tag{27}$$

$$\Rightarrow Y = \sum_i \lambda_i \cdot y_i \tag{28}$$

$$\Rightarrow Y \leq \sum_i \lambda_i^* \cdot y_i \tag{29}$$

$$\Rightarrow Y \leq \lambda_i^* \sum_i \cdot y_i \tag{30}$$

$$\Rightarrow Y \leq \lambda_i^* y \tag{31}$$

$$\Rightarrow f^{-1}(y) \leq \lambda_i^* f^{-1}(y) \tag{32}$$

$$\Rightarrow \overline{X} \leq \lambda_i^* f^{-1}(y) \tag{33}$$

Thus, the buffer between supply and demand would be less or equal to λ_i^* which by assumption was strictly less than the buffer Λ that the market operator desired. $\square$

Given the limited controllability of the buffer, it is natural to ask what buffer size to expect in equilibrium. It turns out that, in equilibrium, the supply-side buffer is uniquely determined via the demand-side buffer:

Proposition 4. *In the buffer equilibrium, the size of the supply-side buffer equals the size of the demand-side buffer.*

Proof.

$$\overline{X} = f^{-1}(Y) \tag{34}$$

$$\Leftrightarrow \overline{X} = f^{-1}(\lambda \cdot y) \tag{35}$$

$$\Leftrightarrow \overline{X} = \lambda \cdot f^{-1}(y) \tag{36}$$

Equation (35) follows because of Assumption 3 (linear prediction for aggregate demand). Equation (36) follows from System Properties 3 and 4 (production functions are bijective and exhibit CRTS). $\square$

In words, the size of the buffer depends on how forward-looking the users are. If on average the users give themselves a 25% buffer on the demand side (e.g., a user has currently backed up 20GB and sets the sliders in such a position that his maximum online backup space is 25GB), then the system would also have a 25% buffer on the supply side, i.e., $\overline{X} = 1.25 \cdot f^{-1}(y)$.

Even though the market operator cannot influence the size of the overall supply-side buffer by adjusting market prices, Proposition 4 provides us with a different, yet very natural way to achieve any desired buffer. The market operator simply needs to insist that every user gives himself a certain minimum demand-side buffer. One way to achieve this is to build this requirement into the user interface, i.e., given user i's current demand y_i there would be a minimum demand $Y_i = \lambda_i \cdot y_i$ below which the user could not go:

Proposition 5. *If the market operator can enforce any demand-side buffer for individual users, then he can achieve any desired supply-side buffer size $\Lambda > 1$ in the buffer equilibrium.*

Proof. We let the market operator set all individual user's minimum required demand-side buffers to $\lambda_i = \Lambda$. Then we know from Proposition 4 that the resulting aggregate supply-side buffer will also be at least Λ. $\qquad\square$

Note that enforcing a demand-side buffer of Λ for every individual user *can* result in efficiency losses. A user who, without this restriction, would have chosen a smaller demand-side buffer, now loses some utility. For example, he might now choose a smaller Y_i to avoid having to give up as many resources X_i. Thus, in practice, the desired supply-side buffer Λ would have to be carefully chosen, trading-off a larger supply-side buffer on the one hand, with some efficiency losses for individual users on the other hand.

6. The Price Update Algorithm

In this section we propose and analyze a price update algorithm that is invoked regularly on the server (e.g., once a day), with the goal to move prices towards the buffer equilibrium over time. Our algorithm is oriented at the tâtonnement process as defined by Walras [24]. However, Walras' algorithm only allowed trades at equilibrium prices. In our system, however, we must allow trades at all times, even out of equilibrium.

6.1. The Algorithm

Because users' preferences are homogeneous of degree zero, collinear price vectors are equivalent. Thus, instead of searching for the equilibrium price vector in $\mathbb{R}^3$, we can simplify the task by looking at projective space $\mathbb{RP}^2$:

$$\mathbb{RP}^2 := \left\{ (p_{\overline{S}}, p_{\overline{U}}, p_{\overline{D}}) \in \mathbb{R}^3 \setminus \{0\} : (p_{\overline{S}}, p_{\overline{U}}, p_{\overline{D}}) \sim \lambda(p_{\overline{S}}, p_{\overline{U}}, p_{\overline{D}}) \quad \forall \lambda \in \mathbb{R}_+ \right\}$$

Thus, we can fix the price of an arbitrary good (the numeraire) and normalize the price vector accordingly. Here, we normalize the price of storage space to 1:

$$p = (p_{\overline{S}}, p_{\overline{U}}, p_{\overline{D}}) \sim (1, \frac{p_{\overline{U}}}{p_{\overline{S}}}, \frac{p_{\overline{D}}}{p_{\overline{S}}})$$

In Section 5.4, we have reduced the problem of finding the buffer equilibrium to finding the root of the function $g(p) = \big(g_{\overline{U}}(p), g_{\overline{D}}(p)\big)$ where

$$g_{\overline{U}}(p) = \Big(\frac{\overline{X}_{\overline{U}}}{\overline{X}_{\overline{S}}} - \frac{f_{\overline{U}}^{-1}(Y)}{f_{\overline{S}}^{-1}(Y)}\Big) \quad \text{and} \quad g_{\overline{D}}(p) = \Big(\frac{\overline{X}_{\overline{D}}}{\overline{X}_{\overline{S}}} - \frac{f_{\overline{D}}^{-1}(Y)}{f_{\overline{S}}^{-1}(Y)}\Big).$$

This formulation of the buffer equilibrium is also useful for the price update algorithm, because finding the root of a function is a well-understood mathematical problem. *Newton's method* is probably the best-known root-finding algorithm and converges quickly in practice. However, it requires the evaluation of the function's derivative at each step. Unfortunately, we don't know the function $g(\cdot)$ and thus cannot compute its derivative. Instead, we only get to know individual points in each iteration and can use these points to estimate the derivative. This is exactly what the *secant method* does for a one-dimensional function.

The problem is that $g(p)$ is 2-dimensional, and thus the secant method is not directly applicable. The appropriate multi-dimensional generalization is *Broyden's method* [4], a quasi-Newton method.

Unfortunately, that method requires knowledge of the Jacobian, which we don't know and also cannot even measure approximately. However, we show that one can use an approximation to the diagonal sub-matrix of the Jacobian instead of the full Jacobian matrix. The diagonal sub-matrix of the Jacobian can be approximated by studying changes in the function $g(p)$. This leads to the following quasi-Newton method for multiple dimensions:

Definition 6. *(The Price Update Algorithm)*

$$p_l^{t+1} = \begin{cases} 1 & for \ l = \overline{S} \\ p_l^t - \frac{p_l^t - p_l^{t-1}}{g_l(p^t) - g_l(p^{t-1})} \cdot g_l(p^t) & for \ l = \overline{U}, \overline{D} \end{cases}$$

For the implementation of the price update algorithm in our system we took care of a few special cases (e.g., exactly reaching the equilibrium such that terms cancel out), but we omit the details here.

6.2. Theoretical Convergence Analysis

We begin with the analysis of the convergence of the following iteration rule:

$$x^{(k+1)} = x^{(k)} - D(x^{(k)})^{-1} F(x^{(k)}) \tag{37}$$

where F is a function $F : \mathbb{R}^n \to \mathbb{R}^n$ and D is the diagonal sub-matrix of the Jacobian J of F. We define the matrix L by the rule $J(x) = D(x) + L(x)$, i.e., L comprises of the off-diagonal partial derivatives in the Jacobian. For this iteration rule, the following theorem holds (the proof is provided in the Appendix, in Section D):

Theorem 4. *Let F be a continuously differentiable function. Suppose that in the iteration rule given by equation (37), $x^{(0)}$ is chosen close enough to a root x^* of F, $J(x^*)$ is non-singular, J and D are Lipschitz continuous, and $L(x^*) = 0$. Then the successive iterations $x^{(k)}$ produced by the iteration rule converge to x^*, and the rate of convergence is at least Q-linear.[12]*

The problem one faces when trying to apply the secant method to higher dimensions is that the system of equations provided by $J_k \cdot (x^{(k)} - x^{(k-1)}) \simeq F(x^{(k)}) - F(x^{(k-1)})$ (where J_k is the current estimate of the Jacobian) is under determined. However, if one uses the diagonal approximation to the Jacobian, then the system is fully determined. What Theorem 4 says is that under certain conditions, using the diagonal sub-matrix of the Jacobian instead of the full Jacobian in the given iteration rule, still leads to convergence to a root of the function.

Equipped with Theorem 4, it is now easy to prove that the price update algorithm given in Definition 6 converges to a buffer equilibrium. We only need to consider the update algorithm for resource prices $p_{\overline{U}}$ and $p_{\overline{D}}$ because the price for space remains constant at 1. Consider the function $g(\cdot)$, and as before, J is the Jacobian of $g(\cdot)$, D is the diagonal sub-matrix of J, and L is defined by the rule $J(x) = D(x) + L(x)$.

---

[12]Q-linear convergence means that $\lim_{k \to \infty} \frac{\|x^{(k+1)} - x^*\|}{\|x^{(k)} - x^*\|^q} = \mu$ with $\mu \in (0, 1)$ and $q = 1$. We can in fact prove that the iteration rule exhibits faster than Q-linear convergence: just like Broyden's method, its convergence is locally Q-superlinear (with $q \approx 1.62$, and $\mu > 0$). However, showing this result requires a more intricate argument which is beyond the scope of this paper.

Corollary 2. *Consider the price update algorithm given in Definition 6. If $g(\cdot)$ is a continuously differentiable function, $p^{(0)}$ is chosen close enough to a root p^* of $g(\cdot)$, the Jacobian $J(p^*)$ is non-singular, J and D are Lipschitz continuous, and $L(p^*) = 0$, then the price update algorithm converges to an equilibrium price vector p^*, and the rate of convergence is at least Q-linear.*

Proof. We have shown in Section 5.4 that if we find a price vector p^* such that $g(p^*) = 0$, then we have reached a buffer equilibrium. Thus, we only have to show that the price update algorithm converges to a root of the function $g(\cdot)$. Now, note that the price update algorithm provided in Definition 6 defines a quasi-Newton iteration rule that uses the diagonal sub-matrix of the Jacobian of the function $g(\cdot)$, equivalent to the iteration rule given in equation (37). By Theorem 4, that iteration rule converges locally to a root of $g(\cdot)$, and the rate of convergence is at least Q-linear.[13] □

7. Usability Study

In this section, we describe some of the results from a formative usability study of our system with 16 users.[14] Our main goal in the usability study was to understand whether the market user interface we propose for the P2P backup system is a usable instantiation of the hidden market paradigm. For details on the study set-up, please see Section C in the Appendix.

7.1. Methodology

The purpose of the usability study was to evaluate how users understand the hidden market UI, which mental models are invoked and whether users can successfully interact with the market. Note that during the study, the users interacted with the real P2P backup client software that was connected via TCP to the P2P server application and to 100 other simulated clients. We started the users off with two warm-up tasks. First, they had to perform one backup using the software. Second, they had to open the settings window and answer a series of questions regarding the information they saw.

Upon completion of the warm-up phase, we gave the study participants 11 tasks, each consisting of a user scenario with hypothetical preferences, and a description of the goal setting for that user. We chose tasks with varying complexity and we also tested different mental models in different tasks. For example, Scenario 1 was the most simple one, asking the user to "change the settings such that you have approximately 15 GB of free online backup space available." In contrast, Scenario 11 was rather complex, asking the user to "imagine you are a user who likes to download videos and store them on your computer for a while. Assume that you need 20 GB of your own hard disk space to store the videos, and obviously you need lots of download bandwidth, but you do not care too much about upload bandwidth. Please change your settings so that you have approximately 25 GB of free online backup space available while taking the other constraints into account."

---

[13]One might wonder how restrictive the conditions of Theorem 4 and Corollary 2 are. The condition that the matrices J and D be Lipschitz continuous puts upper bounds on how fast the partial derivatives of the function can change. One can relax this assumption to just that of J and D being Lipschitz continuous in a neighborhood of the root without affecting the conclusions of the theorem and corollary. Local Lipschitz continuity near the neighborhood of the root seems like a plausible condition for $g(\cdot)$ to satisfy because it is hard to envision wild changes in the function near an equilibrium point. The non-singularity of $J(p^*)$ means that our function does not have a higher order zero at the equilibrium point. It is likely that our algorithm would still converge even if this assumption fails, but we do not have a proof of this. The local convergence of our method is an aspect we share with all Newton's methods operating in multiple dimensions, and this is the most worrisome property as well as the hardest to get a handle on. If $\|J(p^*)^{-1}\|$ and Lipschitz constants of J and D around p^* are all small, then the basin of convergence is large. However, it seems that only experimental evidence can validate whether this assumption is reasonable in our situation.

[14]See [21] for the full study.

We asked the users to "think out loud" as they performed each task and we made detailed observations during the tasks. Using the 11 tasks, we tested four different mental models, i.e., aspects of the user's understanding of the market:

1. **Give & Take**: The users understand they must give some of their resources (on the right side) and get a proportional amount of online backup space in return (on the left side). This was tested using tasks 1 and 2. The test was deemed successful if the users adjusted all settings correctly.

2. **Bundling:** The users understand the bundle constraints, i.e., that they cannot provide zero of any resource because only resource bundles have value. This was tested using tasks 3 and 4. The test was deemed successful if the users adjusted all settings correctly.

3. **Prices:** The users understand that different resources can have different "prices" at different points in time. This was tested using tasks 7, 8, and 9. The test was deemed successful if the users adjusted the settings for task 9 correctly (tasks 7 and 8 gave them practice to learn the model and discover the pricing aspect).

4. **Bundling (Learned):** The users understand the bundle constraints after exploring the UI for a while, i.e., after a certain learning period. This was tested using tasks 10 and 11. The test was deemed successful if the users adjusted all settings correctly.

Note that the tasks were set-up such that finding the correct setting by coincidence was unlikely. The correct setting was not a natural focal point so that the user researcher could easily decide whether the participant had truly understood the task (and thus the right mental model had been activated) or not. Of course, the "think out loud" method also helped determining the result of a test. For example, when testing the understanding of the bundle constraints, if a user said something like "I see, I obviously cannot give 5 GB of space without giving any bandwidth, thus I choose to supply the minimum amount of bandwidth I have to give," then this counted as sufficient understanding of the bundle constraints. The rare cases where a user had coincidentally chosen the correct settings but did not display sufficient understanding of the problem were also deemed to be failures in our experiment.

7.2. Results

Table 2 summarizes the results from the usability study, evaluating whether the 4 different mental models have been successfully activated or not. It turns out that the basic aspects of the UI were understood by all users (1: Give & Take). However, the first time the users faced a combinatorial task, e.g., "minimize your upload bandwidth while maintaining at least 15 GB of free online backup space", only 9 out of 16 users completely understood the problem and found the optimal settings. The understanding of the bundle constraints of the market improved towards the end of the study, showing that a certain learning effect had occurred. In particular, 2 of the users that had not understood the bundle constraints at the beginning, understood them well at the end of the study, leading to 11/16 successful outcomes for "Bundling (Learned)".

The most difficult tasks for the users were certainly the ones testing their understanding of prices because this required three steps from them: first, discovering that different resources had different prices, second, understanding the implication for their supply of resources, and then third, choosing the optimal supply settings for themselves given current prices. Only 7 out of 16 users successfully completed all three steps, and thus were deemed to understand the pricing aspect.

One immediate finding is that the performance of the users is uncorrelated with the way we had segmented them into experts or novices in advance (see Table 2). Thus, prior experience with P2P

Category	Mental Model	Experts	Novices	Total
1	Give & Take	8/8	8/8	16/16
2	Bundling	4/8	5/8	9/16
3	Prices	5/8	2/8	7/16
4	Bundling (Learned)	5/8	6/8	11/16

Table 2: Results from the Usability Study: Number of Users Falling into Comprehension Categories

file-sharing software did not seem matter. Instead, anecdotal evidence suggests that those users whose jobs or education involved some mathematical modeling seemed to understand the concepts underlying the UI faster. This makes sense, given that some of the tasks were relatively complex and required a good, somewhat analytical understanding of the UI. However, a factor that is difficult to measure but seemed to play an important role in this study is the users' curiosity, i.e., how much the users liked to play with the sliders until they figured out how the interface worked. This aspect is particularly important for category 4, i.e., the pricing aspect. The less curious users who did not explore the settings space as much as the others were also the ones that did not discover the fact that different resources have different prices, and consequently failed to solve the pricing tasks optimally.

Upon completion of the interactive part of the study we asked the users about their experience with the UI. Despite the fact that almost every user had difficulties with at least one of the tasks, the user feedback was largely positive. Most users thought that the software made it easy to perform the tasks they were given (with a 3.8 average on a 5-point Likert scale, with 1=strongly disagree and 5=strongly agree) and they indicated that they enjoyed using the UI (3.8 average on the same 5-point Likert scale). Most users were pretty confident that they completed the tasks successfully (with an average 4.0 on the same 5-point Likert scale). The users liked the graphical/visual representation of the concepts involved. Despite some difficulties with solving the tasks, the users thought that the UI was "clean, simple, intuitive and easy to use." All users liked the ease of using the bar chart to choose the desired amount of free online backup space. Furthermore, they liked that the UI gave immediate feedback regarding the consequences of their choices. The users primarily disliked that it took them a while to understand the concept and logic behind the sliders.

From the pre-study questionnaire we have seen that for a large number of users, P2P backup systems could be an attractive alternative to server-based systems. However, this still leaves open the question how users perceive the trade-off between a market-based system (that gives users more freedom in choosing different combinations of supplied resources) vs. a non-market-based system (that has a simpler UI). In the post-study questionnaire we asked the users twice to compare the two options. The first time we asked the question, we gave no additional information beforehand. But before asking them for the second time, we described a particular scenario highlighting the fact that the market-based system gives the users more freedom in choosing the supplied resources. The results were that, when asked for the first time, the users already slightly preferred the market-based system (3.3 on a 5-point Likert scale, with 1=definitely prefer the simpler UI and 5=definitely prefer the complex UI). After describing the hypothetical scenario where the non-market-based system would lead to a degraded user experience, the average score rose to 4.0. We interpret these results as follows: a priori, some users do not see the advantage of a market-based system. However, after understanding the possible limitations of the non-market-based system, they realize the benefit of the increased freedom in choosing what to supply, and they value this benefit higher than the disutility from the additional complexity of the UI.

8. Conclusion

In this paper, we have presented the design and analysis of a novel resource exchange market underlying a P2P backup application. We have also used the P2P backup market as a first case study of a new market design paradigm which we call *hidden market design*. We propose hidden markets for the design of electronic systems in domains with many non-experts users and where markets might be unnatural. To successfully hide the market complexities from the users in our system, new techniques at the intersection of market design and user interface design were necessary. At all times, for the model formulation and the theoretical analysis, our focus was on the actual implemented P2P backup system, which we have successfully tested in alpha version.

In contrast to existing P2P backup systems, our design gives users the freedom to supply different ratios of resources. This introduces the problem that without properly motivating the users to supply those resources that are currently scarce, the system might not have enough supply to satisfy demand, which motivates the use of a P2P resource market. While existing work on P2P data economies has generally designed markets that balance supply and demand in equilibrium, our market is designed to work well, even out of equilibrium. The users are not required to continuously update their supply and demand. Instead, we provide a *hidden market UI* that lets them choose bounds on their maximum supply in return for being allowed to consume a certain maximum amount of backup services. The UI completely hides the users' account balances and payments, and only indirectly exposes the current market prices. A key contribution is the new slider control that we developed which we use to display the bundles constraints to the users in an indirect way. The sliders also ensure that the users can only choose supply settings that satisfy certain resource ratio constraints, which allows us to provide the users a linear interaction with the system.

To maximize the safety of the system out of equilibrium, we have declared as our target to maximize the overall size of the buffer between current demand and maximum supply. We have introduced the *buffer equilibrium* concept and shown that, under certain assumptions, the size of the buffer is maximal in the buffer equilibrium. The economic analysis of the market required the introduction of composite resources on the supply side, and the careful study of the system's production technology, to convert the market into a pure exchange economy. In this model, we have proved that a buffer equilibrium is always guaranteed to exist. This result also holds if a certain percentage of the user population is price-insensitive or even adversarial. However, we have shown that the more freedom we give users in choosing their supply settings, the less robust the system becomes against adversarial attacks. We have explained how the theoretical equilibrium analysis actually has an important market design implication. The theorem regarding adversarial users provides the market designer with a concrete formula how large the system's slack factor can be, given a certain belief about the maximal percentage of adversarial users in the population.

To prove uniqueness of the buffer equilibrium, we needed two additional assumptions that are very reasonable in our domain. We have explained why it makes sense to assume that services are perfect complements, and how that affects even the supply of resources via the flow constraints. By making a relatively weak assumption regarding how the relative supply of resources changes upon price changes, we were able to prove uniqueness of the buffer equilibrium. An interesting corollary of the uniqueness result was that the market operator has limited control over the size of the buffer via price updates alone. However, we have shown how changes to the UI design can resolve this problem: by enforcing certain demand-side buffers in the UI, the market operator can ensure any desired supply-side buffer. We have proposed a price update algorithm that only requires daily aggregate supply and demand information, and proved that it converges linearly to the buffer equilibrium, given that initial prices are chosen close enough to equilibrium prices.

To evaluate the hidden market UI, we have performed a formative usability study of our system. Our main goal was to determine whether the UI activates the right mental model, and whether the users can successfully interact with the hidden market. Overall, the results were encouraging and show promise for the hidden market design paradigm. Most users intuitively understood the give & take principle as well as the bundle constraints of the market. It was particularly positive to see that even after the users had used he system for 45 minutes, they had not realized they were interacting with a market-based system, yet were able to complete most of the tasks successfully. This shows that we have successfully hidden the market. The pricing aspect, however, was difficult for some users, i.e., they either never learned that different resources have different values (prices) in the system, or they were unable to exploit this insight properly. We are currently investigating new user interfaces that still hide the market from the users, but provide them with slightly more information and guidance regarding the pricing aspect.

In ongoing work we are also analyzing different ways to monetize the P2P market platform. There is an easy and elegant way to generate revenue while still running the market using a virtual currency: the market operator can charge a small tax on each virtual currency transaction and use the surplus to sell backup services on a secondary market for real money. More specifically, the P2P users would not have to be involved in any real-money transactions and the customers from the secondary market would buy backup services like they would from a centralized data center. If real monetary transaction are made possible and deemed desirable in the P2P system itself, then we can also open the whole market for real monetary payments. On the one side, users will then be able to pay for their consumption of services by either providing their own resources or by paying with real money, and on the other side, users will then also be able to earn real money by supplying their resources. With this design, the market operator could generate revenue by charging a tax on each virtual currency transaction and by charging a tax on each real-money transaction.

We believe that the hidden market design presented in this paper has applicability beyond P2P backup systems, and one such example could be *smart grids*, i.e., the next generation of electricity networks. The main idea of smart grids is to expose the changing market price for electricity to the end users, such that not only supply but also demand becomes price-elastic. Governments and industry labs are currently making large research and development investments for smart grids [6], but it seems that the user interface aspect of these systems is not getting enough attention. We argue that to effectively involve the end-consumers of electricity in these new energy markets, a hidden market UI will be necessary. In general, more and more market-based systems are currently emerging, often with users who are mostly non-experts and might find the market paradigm unnatural in the particular domain. We hope that the ideas we presented in this paper will inspire other researchers to develop similar hidden market designs for novel applications in many other domains.

References

[1] Jr. Alexander S. Kelso and Vincent P. Crawford. Job matching, coalition formation, and gross substitutes. *Econometrica*, 50(6):1483–1504, 1982.

[2] Christina Aperjis and Ramesh Johari. A peer-to-peer system as an exchange economy. In *Proceedings from the Workshop on Game Theory for Communications and Networks (GameNets)*, Pisa, Italy, October 2006.

[3] William J. Bolosky, John R. Douceur, and Jon Howell. The Farsite project: A retrospective. *SIGOPS Operating Systems Review*, 41(2):17–26, 2007.

[4] C. G. Broyden, J. E. Dennis Jr., and Jorge J.Moré. On the local and superlinear convergence of quasi-newton methods. *J. Inst. Math. Appl.*, 12:223–245, 1973.

[5] Landon P. Cox and Brian D. Noble. Samsara: Honor among thieves in peer-to-peer storage. In *Proceedings of the nineteenth ACM symposium on Operating systems principles (SOSP)*, pages 120–132, Bolton Landing, NY, 2003.

[6] U. S. Department Of Energy. Grid 2030: A national vision for electricity's second 100 years. 2003.

[7] Michael J. Freedman, Christina Aperjis, and Ramesh Johari. Prices are right: Managing resources and incentives in peer-assisted content distribution. In *Proceedings of the 7th International Workshop on Peer-to-Peer Systems*, 2008.

[8] Sharad Goel, David Pennock, Daniel M. Reeves, and Cong Yu. Yoopick: A combinatorial sports prediction market. In *Proceedings of the 23rd Conference on Artificial Intelligence (AAAI)*, pages 1880–1881, Chicago, IL, July 2008.

[9] Friedrich A. Hayek. The uses of knowledge in society. *American Economic Review*, 35:519–530, 1945.

[10] Gary Hsieh and Scott Counts. mimir: A market-based real-time question and answer service. In *Proceedings of Computer Human Interfaces (CHI)*, Boston, MA, April 2009.

[11] Gary Hsieh, Robert Kraut, Scott E. Hudson, and Roberto Weber. Can markets help? Applying market mechanisms to improve synchronous communcation. In *Proceedings of the ACM Conference on Computer Supported Cooperative Work*, San Diego, California, November 2008.

[12] L. V. Kantorovich. *Functional analysis and applied mathematics*, volume 6(28), pages 89–185. Uspehi Matem. Nauk (N. S.), 1948.

[13] L. V. Kantorovich and G. P. Akilov. *Functional analysis in normed spaces*, volume 46 of *International Series of Pure and applied Mathematics*. The Macmillan Co., New York, 1964.

[14] John Kubiatowicz, David Bindel, Yan Chen, Steven Czerwinski, Patrick Eaton, Dennis Geels, Ramakrishna Gummadi, Sean Rhea, Hakim Weatherspoon, Westley Weimer, Chris Wells, and Ben Zhao. Oceanstore: An architecture for global-scale persistent storage. In *Proceedings of the Ninth international Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS)*, 2000.

[15] Kevin Lai, Lars Rasmusson, Eytan Adar, Li Zhang, and Bernardo A Huberman. Tycoon: An implementation of a distributed, market-based resource allocation system. *Multiagent Grid Systems*, 1(3):169–182, 2005.

[16] Jin Li and Cha Zhang. Distributed hosting of web content with erasure coding and unequal weight assignment. In *Proceedings of the IEEE International Conference on Multimedia and Expo (ICME)*, pages 27–30, Taipei, Taiwan, June 2004.

[17] Andreu Mas-Colell, Michael D. Whinston, and Jerry R. Green. *Microeconomic Theory*. Oxford University Press, 1995.

[18] Satu S. Parikh and Gerald L. Lohse. Electronic futures markets versus floor trading: Implications for interface design. In *Proceedings of the CHI conference on Human factors in computing systems*, pages 296 – 303, 1995.

[19] J.A. Pouwelse, P. Garbacki, D.H.J. Epema, and H.J. Sips. The bittorrent p2p file-sharing system: Measurements and analysis. In *4th International Workshop on Peer-to-Peer Systems (IPTPS)*, 2005.

[20] Sven Seuken, Denis Charles, Max Chickering, and Sidd Puri. Market Design and Analysis for a P2P Backup System. In *Proceedings of the 11th ACM Conference on Electronic Commerce (EC)*, Cambridge, MA, June 2010.

[21] Sven Seuken, Kamal Jain, Desney Tan, and Mary Czerwinski. Hidden Markets: UI Design for a P2P Backup Application. In *Proceedings of the Conference on Human Factors in Computing Systems (CHI)*, Atlanta, GA, April 2010.

[22] Sven Seuken, David C. Parkes, and Kamal Jain. Hidden market design. In *Proceedings of the 24th Conference on Artificial Intelligence (AAAI)*, Atlanta, GA, July 2010.

[23] David M. Smith. The cost of lost data. *Graziadio Business Report*, 3, 2003. URL http://gbr.pepperdine.edu/033/dataloss.html.

[24] Léon Walras. *Eléments d'économie politique pure; ou, théorie de la richesse sociale (Elements of pure economics; or, the theory of social wealth)*. Corbaz, Lausanne, 1874.

[25] Fredrik Ygge and Hans Akkermans. Power load management as a computational market. In *Proceedings of the 2nd International Conference on Multi-Agent Systems (ICMAS)*, pages 393–400, Kyoto, Japan, 1996.

Appendix

A. Proof of Lemma 1

Proof. We begin by showing the "$\Rightarrow$" direction:
If $\mathbb{B}_{\overline{S}}(p) = \mathbb{B}_{\overline{U}}(p) = \mathbb{B}_{\overline{D}}(p)$ then:

$$\exists \lambda > 1 \text{ s.t. } \forall l : \overline{X}_l(p) = \lambda \cdot f_l^{-1}(y(p))$$

Now, due to Assumption 3 we know that $\exists \delta : Y(p) = \delta \cdot y(p)$. Thus:

$$\Rightarrow \forall l : \overline{X}_l(p) = \lambda \cdot f_l^{-1}(\frac{1}{\delta}Y(p)) \tag{38}$$

$$\Rightarrow \forall l : \overline{X}_l(p) = \lambda \cdot \frac{1}{\delta} \cdot f_l^{-1}(Y(p)) \tag{39}$$

$$\Rightarrow \forall l : \overline{X}_l(p) = \lambda^* \cdot f_l^{-1}(Y(p)) \quad \text{for } \lambda^* = \lambda \cdot \frac{1}{\delta} \tag{40}$$

$$\Rightarrow \overline{X}(p) = \lambda^* \cdot f^{-1}(Y(p)) \tag{41}$$

From the flow constraints (Eqn. 6) we also know that:

$$\overline{X}(p) \cdot p = f^{-1}(Y(p)) \cdot p \tag{42}$$

Equations (41) and (42) can only both be true if $\lambda^* = 1$. Thus, it follows that:

$$\overline{X}(p) = f^{-1}(Y(p)).$$

The "$\Leftarrow$" direction is even simpler to show:

$$\overline{X}(p) = f^{-1}(Y(p)) \tag{43}$$
$$\Rightarrow \overline{X}(p) = f^{-1}(\lambda \cdot y(p)) \tag{44}$$
$$\Rightarrow \overline{X}(p) = \lambda \cdot f^{-1}(y(p)) \tag{45}$$
$$\Rightarrow \mathbb{B}_{\overline{S}}(p) = \mathbb{B}_{\overline{U}}(p) = \mathbb{B}_{\overline{D}}(p) \tag{46}$$

Equation 44 follows because of Assumption 3 (Linear Prediction for Aggregate Demand). Equation 45 follows from System Properties 3 and 4 (Production functions satisfy CRTS and are bijective). $\square$

B. Proof of Lemma 3

Proof. We have shown in the proof for Lemma 2, that for $p^n \to p$ with $p \neq 0$ and $p_k = 0$, for every rational user i, for n large enough, at least one of the slack constraints will bind, i.e.:

$$\forall i \quad \exists l : \frac{\overline{X_{ik}^R(p^n)}}{f_k^{-1}(y(p^n))} = \frac{1}{\gamma} \frac{\overline{X_{il}^R(p^n)}}{f_l^{-1}(y(p^n))}.$$

For the remainder of the proof, we will always consider the supply and demand functions for $p^n \to p$, however, we will write $\overline{X}$ and y instead of $\overline{X(p^n)}$ and $y(p^n)$ to simplify notation. It is possible, that for each rational user, a different slack constraint binds. Let L and M denote the sets of rational users for whom the slack constraints bind for resources l and m, respectively, i.e., $R = L \cup M$. We assume that L and M are disjunct; if for some user, both slack constraints for l and m bind, we can place that user randomly into either L or M. We let $\overline{X_l^L} = \sum_{i \in L} \overline{X_{il}^R}$ and $\overline{X_l^M} = \sum_{i \in M} \overline{X_{il}^R}$. Then:

$$\frac{\overline{X_l^L}}{f_l^{-1}(y)} \geq \frac{\overline{X_m^L}}{f_m^{-1}(y)} \quad \text{and} \quad \frac{\overline{X_m^M}}{f_m^{-1}(y)} \geq \frac{\overline{X_l^M}}{f_l^{-1}(y)}.$$

It is easy to see that at least for one of the resources l or m, the joint supply of that resource from the corresponding set of users L or M must be at least half of the total supply of that resource from the rational users. With out loss of generality, let l be such a resource. Thus:

$$\frac{\overline{X_l^L}}{f_l^{-1}(y)} \geq \frac{1}{2} \cdot \frac{\overline{X_l^R}}{f_l^{-1}(y)}.$$

Remember that for all users $\in L$, the slack constraint for l binds. For all other rational users, we only know that they supply least of resource k. Thus:

$$\gamma \cdot \frac{\overline{X_k^L}}{f_k^{-1}(y)} = \frac{\overline{X_l^L}}{f_l^{-1}(y)} \quad \text{and} \quad \frac{\overline{X_k^M}}{f_k^{-1}(y)} \leq \frac{\overline{X_l^M}}{f_l^{-1}(y)}$$

37

By adding both sides together we get:

$$\gamma \cdot \frac{\overline{X_k^L}}{f_k^{-1}(y)} + \frac{\overline{X_k^M}}{f_k^{-1}(y)} \le \frac{\overline{X_l^R}}{f_l^{-1}(y)}$$

Because $\overline{X_l^L} + \overline{X_l^M} = \overline{X_l^R}$, this is equivalent to:

$$(\gamma - 1) \cdot \frac{\overline{X_k^L}}{f_k^{-1}(y)} + \frac{\overline{X_k^R}}{f_k^{-1}(y)} \le \frac{\overline{X_l^R}}{f_l^{-1}(y)}$$

Because $\frac{\overline{X_l^L}}{f_l^{-1}(y)} \ge \frac{1}{2} \cdot \frac{\overline{X_l^R}}{f_l^{-1}(y)}$, this implies:

$$\left(\frac{\gamma - 1}{2}\right) \cdot \frac{\overline{X_k^R}}{f_k^{-1}(y)} + \frac{\overline{X_k^R}}{f_k^{-1}(y)} \le \frac{\overline{X_l^R}}{f_l^{-1}(y)}$$

$$\Rightarrow \left(\frac{\gamma + 1}{2}\right) \cdot \frac{\overline{X_k^R}}{f_k^{-1}(y)} \le \frac{\overline{X_l^R}}{f_l^{-1}(y)}$$

$$\Rightarrow \frac{\overline{X_k^R}}{f_k^{-1}(y)} \le \left(\frac{2}{\gamma + 1}\right) \cdot \frac{\overline{X_l^R}}{f_l^{-1}(y)}.$$

So far, we have only argued about the rational users, and derived how much smaller the buffer for resource k for these users must be relative to the maximum buffer for resource l or m. Now we turn our attention to the adversarial users as well. Because $Y^R \ge C \cdot Y^A$ we know that $\overline{X^R} \cdot p \ge C \cdot \overline{X^A} \cdot p$. For large enough n, we know that p_k^n is close enough to 0 such that all income must come from supply resources l and m. Thus:

$$\overline{X_l^R} \cdot p_l + \overline{X_m^R} \cdot p_m \ge C \cdot \left(\overline{X_l^A} \cdot p_l + \overline{X_m^A} \cdot p_m\right) \tag{47}$$

Because l was assumed to be the resource with the largest buffer for the rational users, we know that:

$$\overline{X_m^R} \le \overline{X_l^R} \cdot \frac{f_m^{-1}(y)}{f_l^{-1}(y)} \tag{48}$$

For the adversarial users, there is no restriction between the buffers for l and m, except the standard slack constraint, i.e.:

$$\overline{X_m^A} \ge \frac{1}{\gamma} \cdot \overline{X_l^A} \cdot \frac{f_m^{-1}(y)}{f_l^{-1}(y)} \tag{49}$$

If we combine Equations 47, 48 and 49, then we get:

$$\overline{X_l^R} \cdot p_l + \overline{X_l^R} \cdot \frac{f_m^{-1}(y)}{f_l^{-1}(y)} \cdot p_m \ge C \cdot \left(\overline{X_l^A} \cdot p_l + \frac{1}{\gamma} \cdot \overline{X_l^A} \cdot \frac{f_m^{-1}(y)}{f_l^{-1}(y)} \cdot p_m\right) \tag{50}$$

$$\Rightarrow \overline{X_l^R} \cdot p_l + \overline{X_l^R} \cdot \frac{f_m^{-1}(y)}{f_l^{-1}(y)} \cdot p_m \ge C \cdot \overline{X_l^A} \cdot p_l + \frac{C}{\gamma} \cdot \overline{X_l^A} \cdot \frac{f_m^{-1}(y)}{f_l^{-1}(y)} \cdot p_m \tag{51}$$

For the last inequality to be true, a necessary condition is:

$$\overline{X_l^R} \geq \min\{CX_l^A, \frac{C}{\gamma}X_l^A\} \tag{52}$$

$$\Rightarrow \overline{X_l^R} \geq \frac{C}{\gamma}\overline{X_l^A} \tag{53}$$

$$\Rightarrow \overline{X_l^A} \leq \frac{\gamma}{C} \cdot \overline{X_l^R} \tag{54}$$

We have derived above that for rational users, we have:

$$\frac{\overline{X_k^R}}{f_k^{-1}(y)} \leq \left(\frac{2}{\gamma+1}\right) \cdot \frac{\overline{X_l^R}}{f_l^{-1}(y)} \tag{55}$$

For the adversarial users, we have:

$$\frac{\overline{X_k^A}}{f_k^{-1}(y)} \leq \gamma \cdot \frac{\overline{X_l^A}}{f_l^{-1}(y)} \tag{56}$$

If we take these two inequality together we get:

$$\frac{\overline{X_k}}{f_k^{-1}(y)} \leq \frac{\frac{2}{\gamma+1} \cdot \overline{X_l^R} + \gamma \cdot \overline{X_l^A}}{f_l^{-1}(y)} \tag{57}$$

Thus, to get

$$\frac{\overline{X_k}}{f_k^{-1}(y)} \leq \frac{\overline{X_l}}{f_l^{-1}(y)} \tag{58}$$

we need that:

$$\frac{2}{\gamma+1} \cdot \overline{X_l^R} + \gamma \cdot \overline{X_l^A} \leq \overline{X_l} \tag{59}$$

By definition, we have that $\overline{X_l^R} + \overline{X_l^A} = \overline{X_l}$. Because $\gamma > 1$, we know that $\frac{2}{\gamma+1} \cdot \overline{X_l^R} < \overline{X_l^R}$ and $\gamma \cdot \overline{X_l^A} > \overline{X_l^A}$. Thus, the amount by which $\frac{2}{\gamma+1} \cdot \overline{X_l^R}$ is smaller than $\overline{X_l^R}$ is exactly the amount by which $\gamma \cdot \overline{X_l^A}$ can be larger than $\overline{X_l^A}$, for Inequality 59 to hold. Thus, we need:

$$(\gamma - 1) \cdot \overline{X_l^A} \leq (1 - \frac{2}{\gamma+1})\overline{X_l^R} \tag{60}$$

If we now use Equation 54, i.e., $\overline{X_l^A} \leq \frac{\gamma}{C} \cdot \overline{X_l^R}$, it follows that the next inequality implies the previous one:

$$(\gamma - 1) \cdot \frac{\gamma}{C} \cdot \overline{X_l^R} \leq (1 - \frac{2}{\gamma+1})\overline{X_l^R} \tag{61}$$

$$\Leftrightarrow \frac{\gamma^2 - \gamma}{C} \leq \frac{\gamma - 1}{\gamma + 1} \tag{62}$$

Because $\gamma > 1$ and $C > 1$, we can derive the following:

$$(\gamma + 1) \cdot (\gamma^2 - \gamma) \leq C \cdot (\gamma - 1) \tag{63}$$

$$\Leftrightarrow (\gamma - 1) \cdot (\gamma^2 + \gamma) \leq C \cdot (\gamma - 1) \tag{64}$$

$$\Leftrightarrow (\gamma^2 + \gamma) \leq C \tag{65}$$

This completes the proof of the lemma. $\qquad\square$

C. User Study: Set-up

The UI design process included an early exploratory study (with 6 users) and a pilot study (with 6 users). Upon completion of an iterative UI design phase, we recruited 16 users (8 females) from the Greater Seattle area for the usability study. All of the users had some college education and used a computer for at least 10 hours per week. The average age of our participants was about 39, ranging from 22 to 66 years old. None of the users worked for the same company, none of them were usability experts and none of them had used a P2P backup system before. All of the users understood the meaning of "backing up your files" before coming to the study, however only a few of them had used server-based online backup systems before. We recruited two different groups of users: novices and experts. Experts were screened to be users who had used P2P file-sharing software and modified the maximum bandwidth limits of their client in the last 5 years. We also ensured they had some idea about the speeds of an average home broadband connection. Novices were screened such that they did not have technical jobs, were not sophisticated enough to set-up a wireless router by themselves, and had never adjusted the maximum bandwidth limits of a P2P file-sharing client.

In this work we are particularly interested in evaluating the "advanced settings" version of the UI. Thus, our true target group of users was in fact the experts group. However, we included the novice users to make sure we identified all of the problems of the UI or the system in general that might not be found when only testing expert users. We had 8 experts and 8 novices. We ran one participant at a time with each session lasting about 1.5 hours. The users filled out a pre-study questionnaire (20 minutes), completed a series of interactive tasks using the UI (45 minutes), and then completed another post-study survey (20 minutes). We ran the software on a single 3 GHZ Dell computer at full resolution using a 20" 1600x1200 Syncmaster display.

D. Proof of Theorem 3

We discuss some general conditions under which a multi-dimensional Newton iteration converges even if a diagonal approximation is used for the Jacobian. We essentially follow Kantorovich's proof of the local convergence of Newton's method (Kantorovich's theorem [12] and [13] Chapter XVIII).

Definition 7. *Suppose $F : \mathbb{R}^n \to \mathbb{R}^m$. Writing the vector valued function $F(x_1, x_2, \cdots, x_n)$ as*

$$(f_1(x_1, x_2, \cdots, x_n), \cdots, f_m(x_1, x_2, \cdots, x_n))$$

one defines the Jacobian matrix as the $m \times n$ matrix J where $J_{ij} = \partial f_i / \partial x_j$.

We will need the following two results:

Theorem 5. *Suppose $F : \mathbb{R}^n \to \mathbb{R}^m$ is continuously differentiable, and $a, b \in \mathbb{R}^n$. Then*

$$F(b) = F(a) + \int_0^1 J(a + \theta(b - a))(b - a)d\theta,$$

where J is the Jacobian matrix of F.

The above theorem is the second fundamental theorem of calculus. The next theorem extends the triangle inequality obeyed by norms to integrals.

Theorem 6. *If $F : \mathbb{R} \to \mathbb{R}^n$ is integrable over the interval $[a, b]$, then*

$$\left\| \int_a^b F(t)dt \right\| \leq \int_a^b \|F(t)\|dt. \tag{66}$$

We also recall the definition of the operator norm of a matrix.

Definition 8. *If $A \in \mathbb{R}^{m \times n}$, the norm of A is defined as*

$$\|A\| = \max \left\{ \frac{\|Ax\|}{\|x\|} : x \in \mathbb{R}^n, x \neq 0 \right\}.$$

The norm defined above has the following properties:

1. It is a norm on the space $\mathbb{R}^{m \times n}$;

2. $\|Ax\| \leq \|A\|\|x\|$ for all $A \in \mathbb{R}^{m \times n}, x \in \mathbb{R}^n$;

3. $\|AB\| \leq \|A\|\|B\|$ for all $A \in \mathbb{R}^{m \times n}$, $B \in \mathbb{R}^{n \times p}$.

The following is a well-known theorem from Functional analysis.

Theorem 7. *Suppose $J : \mathbb{R}^m \to \mathbb{R}^{n \times m}$ is a continuous matrix-valued function. If $J(x^*)$ is nonsingular, then there exists a $\delta > 0$ such that, for all $x \in \mathbb{R}^m$ with $\|x - x^*\| < \delta$, $J(x)$ is nonsingular and*

$$\|J(x)^{-1}\| < 2\|J(x^*)^{-1}\|.$$

Proof. (Sketch.)The first part follows from the fact that if $J(x^*)$ is non-singular, then $\det J(x^*) \neq 0$ and consequently there is a neighborhood of x^* where the determinant does not vanish (polynomials define continuous maps). The latter part follows from the fact that if the map $x \mapsto J(x)$ is continuous then so is the map $x \mapsto J(x)^{-1}$ whenever the latter map is defined. $\square$

Definition 9. *Suppose $F : \mathbb{R}^n \to \mathbb{R}^m$. Then F is said to be Lipschitz continuous on $S \subseteq \mathbb{R}^n$ if there exists a positive constant T such that*

$$\|F(x) - F(y)\| \leq T\|x - y\|, \text{ for all,} y \in S.$$

This definition can also be applied to a matrix-valued function $F : \mathbb{R}^n \to \mathbb{R}^{m \times n}$ using a matrix norm to $\|F(x) - F(y)\|$.

The usual Newton iteration is phrased as

$$x^{(k+1)} = x^{(k)} - J(x^{(k)})^{-1}F(x^{(k)}). \tag{67}$$

The Newton iteration is known to converge to a root, x^*, of the function F if we start the iteration close enough to x^* (such that the Jacobian is non-singular).

We wish to analyze the convergence of the following update rule:

$$x^{(k+1)} = x^{(k)} - D(x^{(k)})^{-1}F(x^{(k)}), \tag{68}$$

where D is the diagonal sub-matrix of the Jacobian. To this end, we define the matrix L by the rule $J(x) = D(x) + L(x)$, i.e., L comprises of the off-diagonal partial derivatives in the Jacobian.

We will show that if we are in the situation that J and D are Lipschitz continuous and that $L(x^*) = 0$ (is the zero matrix), then the above iteration rule also converges to the root x^* as long as

we start close enough to the root.

Subtracting x^* from both sides of equation (68) and noting that $F(x^*) = 0$ we have

$$x^{(k+1)} - x^* = x^{(k)} - x^* - D(x^{(k)})^{-1}F(x^{(k)})$$
$$= x^{(k)} - x^* - D(x^{(k)})^{-1}\left(F(x^{(k)}) - F(x^*)\right).$$

We now use Theorem 5 to estimate $F(x^{(k)}) - F(x^*)$:

$$F(x^{(k)}) - F(x^*)$$
$$= \int_0^1 J(x^* + \theta(x^{(k)} - x^*))(x^{(k)} - x^*)d\theta$$
$$= \int_0^1 J(x^*)(x^{(k)} - x^*)d\theta$$
$$+ \int_0^1 \left(J(x^* + \theta(x^{(k)} - x^*)) - J(x^*)\right)(x^{(k)} - x^*)d\theta$$
$$= J(x^*)(x^{(k)} - x^*)$$
$$+ \int_0^1 \left(J(x^* + \theta(x^{(k)} - x^*)) - J(x^*)\right)(x^{(k)} - x^*)d\theta.$$

Assuming $L(x^*) = 0$ we have

$$F(x^{(k)}) - F(x^*) = D(x^*)(x^{(k)} - x^*)$$
$$+ \int_0^1 \left(J(x^* + \theta(x^{(k)} - x^*)) - J(x^*)\right)(x^{(k)} - x^*)d\theta.$$

Therefore,

$$\left\|F(x^{(k)}) - F(x^*) - D(x^*)(x^{(k)} - x^*)\right\|$$
$$= \left\|\int_0^1 \left(J(x^* + \theta(x^{(k)} - x^*)) - J(x^*)\right)(x^{(k)} - x^*)d\theta\right\|$$
$$\leq \int_0^1 \left\|\left(J(x^* + \theta(x^{(k)} - x^*)) - J(x^*)\right)(x^{(k)} - x^*)d\theta\right\|$$
$$\leq \int_0^1 \left\|J(x^* + \theta(x^{(k)} - x^*)) - J(x^*)\right\| \|x^{(k)} - x^*\|d\theta$$
$$\leq \int_0^1 T_J\theta\|x^{(k)} - x^*\|^2 d\theta \text{ (using Lipschitz continuity of } J)$$
$$\leq \frac{T_J}{2}\|x^{(k)} - x^*\|^2.$$

We now have

$$x^{(k+1)} - x^*$$
$$= x^{(k)} - x^* - D(x^{(k)})^{-1}(F(x^{(k)} - F(x^*)))$$
$$= x^{(k)} - x^* - D(x^{(k)})^{-1}[D(x^*)(x^{(k)} - x^*)$$
$$+ F(x^{(k)}) - F(x^*) - D(x^*)(x^{(k)} - x^*)]$$
$$= \left(I - D(x^{(k)})^{-1}D(x^*)\right)(x^{(k)} - x^*)$$
$$- D(x^{(k)})^{-1}\left(F(x^{(k)}) - F(x^*) - D(x^*)(x^{(k)} - x^*)\right).$$

Now applying norms on both sides

$$\|x^{(k+1)} - x^*\|$$
$$\leq \left\|\left(I - D(x^{(k)})^{-1}D(x^*)\right)(x^{(k)} - x^*)\right\|$$
$$+ \left\|D(x^{(k)})^{-1}\left(F(x^{(k)}) - F(x^*) - D(x^*)(x^{(k)} - x^*)\right)\right\|$$
$$\leq \left\|I - D(x^{(k)})^{-1}D(x^*)\right\|\left\|x^{(k)} - x^*\right\|$$
$$+ \left\|D(x^{(k)})^{-1}\right\|\left\|F(x^{(k)}) - F(x^*) - D(x^*)(x^{(k)} - x^*)\right\|$$
$$\leq \left\|I - D(x^{(k)})^{-1}D(x^*)\right\|\left\|x^{(k)} - x^*\right\|$$
$$+ \frac{T_J}{2}\left\|D(x^{(k)})^{-1}\right\|\left\|x^{(k)} - x^*\right\|^2.$$

We are assuming that D is also a Lipschitz continuous map:

$$\left\|I - D(x^{(k)})^{-1}D(x^*)\right\| = \left\|D(x^{(k)})^{-1}\left(D(x^{(k)}) - D(x^*)\right)\right\|$$
$$\leq \left\|D(x^{(k)})^{-1}\right\|\left\|D(x^{(k)}) - D(x^*)\right\|$$
$$\leq T_D\left\|D(x^{(k)})^{-1}\right\|\left\|x^{(k)} - x^*\right\|.$$

Thus we have

$$\|x^{(k+1)} - x^*\| \leq \frac{3T}{2}\|D(x^{(k)})^{-1}\|\|x^{(k)} - x^*\|^2,$$

where we have set $T = \max\{T_J, T_D\}$.

If $x^{(k)}$ is sufficiently close to x^*, then: $\|D(x^{(k)})^{-1}\| \leq 2M$,
where $M = \|D(x^*)^{-1}\| = \|J(x^*)^{-1}\|$ by our assumption that $L(x^*) = 0$.

Thus if $x^{(k)}$ is sufficiently close to x^*, then: $\|x^{(k+1)} - x^*\| \leq 3TM\|x^{(k)} - x^*\|^2$.

Moreover, if

$$\|x^{(k)} - x^*\| < \frac{1}{6TM},$$

then

$$\|x^{(k+1)} - x^*\| < \frac{1}{2}\|x^{(k)} - x^*\|.$$

This completes the proof of Theorem 4.